Université de Lille
École Doctorale MADIS

Thèse de Doctorat

Spécialité **Informatique**

présentée par
Omar Darwiche Domingues

Exploration in Reinforcement Learning: Beyond Finite State-Spaces

---

Exploration en apprentissage par renforcement : au-delà des espaces d'états finis

sous la direction de **Michal Valko** et de **Emilie Kaufmann**

Soutenue publiquement le **18 mars 2022** à **Villeneuve d'Ascq**, devant le jury composé de

Emmanuel **Rachelson**	Professeur, ISAE-SUPAERO	Rapporteur & Président
Marcello **Restelli**	Professeur associé, Politecnico di Milano	Rapporteur
Aurélien **Garivier**	Professeur, École Normale Supérieure de Lyon	Examinateur
Matthieu **Geist**	Professeur, Univ. de Lorraine, Google Research	Examinateur
Emilie **Kaufmann**	Chargée de recherche (HDR), CNRS, Univ. de Lille	Directrice de thèse
Michal **Valko**	Chargé de recherche (HDR), Inria, DeepMind	Directeur de thèse
Christina **Lee Yu**	Professeur assistant, Cornell University	Invitée
Ronald **Ortner**	Professeur associé, Montanuniversität Leoben	Invité

Centre de Recherche en Informatique, Signal et Automatique de Lille (CRIStAL),
UMR 9189 Équipe Scool, 59650, Villeneuve d'Ascq, France

Université de Lille

Inria

CRIStAL
Centre de Recherche en Informatique, Signal et Automatique de Lille

To my mom, whose unconditional support and hard work allowed me to get here. And to my grandma, who always said to me: "if you can't make it, then who else can?" and "who studies too much goes crazy".[1] All I can hope is that I made it and did not go crazy!

[1] "Se você não conseguir, quem mais consegue?" e "quem estuda demais fica louco!".

Acknowledgements

There are many people without whom this work would not have been possible. Michal and Emilie, my advisors and friends, gave me support and freedom to explore ideas, inspired me with their energy and intelligence, were always there to ask the tough questions that fuel research, and always there for the social events that keep life bright in a not-always-so-bright world.[2] I am extremely thankful for the opportunities you gave me and the amazing times we had, but I'd like to highlight two particular moments that show how cool you are. Emilie, when you welcomed me in Truchtersheim, showed me some cities around, and prepared vegan *tarte flambées*, it was incredible! And Michal, always knowing me and my passion for free food very well, brought me free snacks from DeepMind when I couldn't be there during my remote internship, this is unforgettable!

Pierre[3] joined the team-family as a postdoc, became my third advisor, an amazing collaborator and friend, without whom many (many!) results presented in this thesis would not have been found, and several gatherings at l'Écart[4] would not have taken place. Thank you for all the discussions, debates, beers, Bernsteins, desserts, brainstorms, left or right?, pizza, S^2, beers.

Matteo helped me discover that reinforcement learning is painfully fun[5], encouraged me to explore kernel-based ideas that appear quite a lot in this thesis, gave me great teaching opportunities[6], and is one of the few people with whom I can really talk about the best kind of pop music. Thank you!

I am also immensely thankful to the members of the jury: Emmanuel, Marcello, Aurélien, Matthieu, Christina, and Ronald. I really admire your work, and it was a great honor to have you on the jury. Special thanks to Emmanuel and Marcello, the *rapporteurs*, for their detailed feedback on the manuscript and the many typos they've spotted! I also appreciated the discussions I had with Emmanuel after the defense about some research topics!

[2] As we all know, things got more complicated after (and including) 2020.

[3] Also known as Pedro, the *theorem-proving machine*, and one of the most brilliant people I have ever met.

[4] A wonderful bar in Lille, la meilleure ville.

[5] During my master's, when he was teaching RL with Alessandro, I can never forget that he replied to an email that I sent right after Christmas about some weird stuff that people do with policy gradients.

[6] Ghana! I remember the cool music we listened to while preparing the exercises.

I thank Lucas and Denise, who are amazing people that I consider as my family in Paris, who were there for the joyful moments and also for the difficult ones, welcomed me uncountable times into their home, either for simple weekends among friends or for their delicious Christmas dinners and joyful New Year's Eve champagne. Xuedong (Javier Francesco Samuel and I forgot what the other name is, probably Carlinhos), a friend and thesis-brother, who was (almost) always there for a walk in Lille, a beyond-meat burger at Holy Moly, a pizza at IT, a strawberry-milkshake-at-Five-Guys-that-tastes-almost-like-the-missing-Häagen-Dazs-à-la-fraise, and, most importantly, to help me find the best apartments in Loos while making me angry about his weird political views and question whether we can know if anything is real. Thank you for that and for all your very thoughtful words in Spanish! Jean, my full-of-energy thesis-half-brother and friend, thank you for all your good (and bad) advice, the best (!) parties in Paris, the times you received me at Facebook-the-office-not-the-social-network (which were of course for academic reasons and not for free food, but thanks for the free food), the nice pictures in Vancouver and beyond, the party hats, the easily recognizable tall-man drawings in skribbl (+Xuedong here too), that one very cool time at Ground Control, and for all the party lights you take wherever you go (I'm trying a metaphor here, but it's late and this manuscript is due tomorrow). Reda (Khobz), another friend and thesis-cousin, thank you for all your thé-à-la-menthe-flavored good vibes, for taking care of the magic key to the vending machine, for teaching me very cool words (among which I can pronounce probably two), and for all the pessimistically optimistic discussions about research! Yannis, a friend who also has the *crème de la crème* of vibes, thank you for the great parties and music, for l'Écart et al., and for the expertise you shared on deep RL and beyond!

I would also like to thank several other friends: Gildo (for all the advice, japanese restaurants, and phone calls), Vinícius (Marco Luque, who sometimes has a great musical taste, special thanks for all the *caronas Swift*), Pedro (Mineiro, who has many interesting alternative discussion topics!), Omar C. (who inspires me to be *chic* in several aspects of life), Pedro (Capixa, for all the great times, I remember when you took an Uber instead of the subway so that we could drink iced-tea-related-beverages), Daniel (Coquinho, thanks for the problem-solving meetings at Unicamp, and all those weird-in-a-great-sense moments at Centrale), Daniel (Dello, also for the great adventure times at Centrale), Victor (Creuzinho, for the great times as neighbors at Centrale and at the *Caverna*, and the pizzas at the *Cantina do Belo*), Denis (a great example of a person who does magic with code), Gabrielle, Lisa, Chloé, Laura, and Nate. I also thank Diego, Vitor (Bassi), Gabriel, Gustavo (Magrelo, Guga), Milena (who always has incredible dreams to share), and André, who, despite the distance (and time!), are still close friends.

I would also like to thank all the amazing people that I was lucky to meet at Inria (in the great SequeL/SCOOL team) or at Inria-adjacent places/situations who were not yet mentioned: Philippe P. (without whom the team wouldn't be there!), Rémi M., Alessandro, Debabrota, Jill-Jênn, Odalric, Olivier, Rémy D., Sadegh, Edouard L. (with whom it was a great pleasure to

collaborate, and who inspired me with the quality of his research, code, and presentations), Jean-Bastien (whose idea led to a great collaboration), Edouard O. (with whom I got my first teaching assistant opportunity), Daniele C. (thanks for the amazing pasta and graphs!), David (thanks for the *croustillons hollandais*/puff-puffs), Ronan, Florian, Guillaume, Lilian, Mathieu, Antoine, Achraf, Hassan, Nathan, Dorian, Hippolyte, Johan, Pierre P., Sarah, Geoffrey, Clémence, Andrea, Rianne, Nicolas, Mariana, Mahsa, César, Riccardo, Alena, Matheus, Timothée, Yoan (Vancouver!), Julien S., Julien T., Marc, Fabien, Patrick, Évrard, and Pierre-Alexandre. I'm also very thankful to Amélie, Lucile, and Charlotte for all their support in Inria. Additionally, this work was made possible by the DELTA project, coordinated by Anders, with whom I was also lucky to collaborate.

During my PhD, I had the chance to do an internship at DeepMind (thanks Michal!), where I could meet, collaborate, and learn from a lot of incredible people. Special thanks to Rémi, Corentin, Bilal, Jean-Bastien, Alaa, Florent, Daniel, Bernardo, Mohammad Azar, Florian, Miruna, Rana, among many other people that made this experience amazing!

Before my PhD, I also had the chance to work on research projects without which I would probably not have arrived at this thesis. Many thanks to Dalton and Darli (with whom I worked at Unicamp), and Vincent (with whom I did an internship) for all their support and for everything I learned from them. Many years ago, before starting my university studies, there were people who helped me look at the world in a way that still guides me today and, in that sense, I thank Glória, Fernando, and Hugo. Speaking of university, I am highly thankful to the University of Campinas, that provided the most amazing intellectual and social environment for me to grow since my early undergraduate years.

Art! Art is important in life. It brings more colors to the world, it is capable of resonating with our happiness, and making a few kinds of sadness enjoyable. In that sense, I would like to thank Taylor Swift, for her amazing work throughout the years, and especially all her released albums between 2019 and 2021, which brought me a lot of joy while working on this thesis.

There were some especially difficult moments in my life during my PhD. One of those was around June 2021, so I'd like to give many thanks to the people who were extremely helpful at the time: Yole and Mozart, Sara, Fátima, Rosa, Jamile, and Carlos (Bifi).

Finally, I conclude these acknowledgements by thanking my family. My mom, Nádia Darwiche, and my grandma, Maria Conceição Ferreira Domingues, are the most amazing people I have ever met (and I was lucky to meet them very early in my life!): if I ever manage to do something useful for the world, it's hugely thanks to them. My dad, Paulo Sérgio Domingues, who inspires me with his tranquility and his ability to do many cool things, from playing the guitar to renovating apartments by himself, and on whose support I know I can always count. I would also like to thank many other people from my family: Jamile (who came to France and helped my mom prepare a wonderful *pot de thèse*!), Ana Maria, Silas (a *grand chef*, with an unparalleled

humor and great style), Rosa (Rosinha, I hope you'll stop smoking), João, Samira, Aichi, Sara, Paulo, Ibraim, Letícia, Xará (my friend), Fátima, Solange, Adilson, Júlia (Jules), Layla, Nasser, Yasmin, Camila, Bárbara, and all my other cousins!

Abstract

Reinforcement learning (RL) is a powerful machine learning framework to design algorithms that learn to make decisions and to interact with the world. Algorithms for RL can be classified as offline or online. In the offline case, the algorithm is given a fixed dataset, based on which it needs to compute a good decision-making strategy. In the online case, an agent needs to efficiently collect data by itself, by interacting with the environment: that is the problem of exploration in reinforcement learning. This thesis presents theoretical and practical contributions to online RL. We investigate the worst-case performance of online RL algorithms in finite environments, that is, those that can be modeled with a finite amount of states, and where the set of actions that can be taken by an agent is also finite. Such performance degrades as the number of states increases, whereas in real-world applications the state set can be arbitrarily large or continuous. To tackle this issue, we propose kernel-based algorithms for exploration that can be implemented for general state spaces, and for which we provide theoretical results under weak assumptions on the environment. Those algorithms rely on a kernel function that measures the similarity between different states, which can be defined on arbitrary state-spaces, including discrete sets and Euclidean spaces, for instance. Additionally, we show that our kernel-based algorithms are able to handle non-stationary environments by using time-dependent kernel functions, and we propose and analyze approximate versions of our methods to reduce their computational complexity. Finally, we introduce a scalable approximation of our kernel-based methods, that can be implemented with deep reinforcement learning and integrate different representation learning methods to define a kernel function.

Résumé

L'apprentissage par renforcement (*reinforcement learning*, RL) est un paradigme de l'apprentissage automatique qui nous permet de concevoir des algorithmes qui apprennent à prendre des décisions et à interagir avec le monde. Les algorithmes de RL peuvent être classés comme hors ligne ou en ligne. Dans le cas hors ligne, l'algorithme dispose d'un ensemble de données fixe, avec lequel il doit calculer une bonne stratégie de prise de décision. Dans le cas en ligne, l'agent doit collecter efficacement des données par lui-même, en interagissant avec l'environnement : c'est le problème que l'on appelle exploration en apprentissage par renforcement. Cette thèse présente des contributions théoriques et pratiques sur le RL en ligne. Nous étudions la performance dans le pire des cas des algorithmes de RL dans des environnements finis, c'est-à-dire, ceux qui peuvent être modélisés avec un nombre fini d'états, et où l'ensemble des actions qui peuvent être prises par un agent est aussi fini. Cette performance se dégrade à mesure que le nombre d'états augmente, alors qu'en pratique, l'espace d'états peut être arbitrairement grand ou continu. Pour résoudre ce problème, nous proposons des algorithmes à noyaux qui peuvent être implémentés pour des espaces d'états généraux, et pour lesquels nous proposons des résultats théoriques sous des hypothèses faibles sur l'environnement. Ces algorithmes reposent sur une fonction noyau qui mesure la similarité entre différents états, qui peut être définie sur des espaces d'état arbitraires, y compris des ensembles discrets et des espaces euclidiens, par exemple. De plus, nous montrons que nos algorithmes à noyaux sont capables d'apprendre dans des environnements non stationnaires en utilisant des fonctions noyau dépendantes du temps, et nous proposons et analysons des versions approximatives de nos méthodes pour réduire leur complexité de calcul. Finalement, nous introduisons une autre approximation de nos méthodes à noyaux, qui peut être implémentée avec des algorithmes d'apprentissage par renforcement profond et intégrer de différentes méthodes d'apprentissage de représentation pour définir un noyau.

Contents

1 Introduction **1**

1.1 Overview . . . 1
1.2 Markov Decision Processes . . . 2
1.3 Sampling Models . . . 8
1.4 Evaluating Reinforcement Learning Algorithms . . . 9
1.5 Contributions . . . 10

2 Planning with a Generative Model **13**

2.1 Model-Based Q-Value Iteration . . . 14
2.2 `SparseSampling`: Planning in Arbitrary State Spaces . . . 16
2.3 `SmoothCruiser`: Planning in Regularized MDPs . . . 21
2.4 Discussion and Bibliographical Remarks . . . 27

3 Online Interaction with Finite MDPs **31**

3.1 Performance Criteria . . . 32
3.2 Lower Bounds: Key Ideas & Hard MDP Instances . . . 34
3.3 Lower Bound on the Regret . . . 38
3.4 Lower Bound on the Sample Complexity . . . 41
3.5 Lower Bounds: Extensions . . . 44
3.6 Upper Bound on the Regret of `UCBVI` . . . 47
3.7 Discussion and Bibliographical Remarks . . . 53

4 A Kernel-Based Approach to Exploration in Continuous MDPs **57**

4.1 Kernel-Based Reinforcement Learning for Exploration . . . 58
4.2 Regret Analysis of `Kernel-UCBVI` . . . 61
4.3 Comparison to Lower Bounds & Related Work . . . 66
4.4 `KeRNS`: An Extension of `Kernel-UCBVI` to Non-Stationary MDPs . . . 67
4.5 Regret Analysis of `KeRNS` . . . 70
4.6 Reducing the Computational Complexity . . . 73
4.7 Experiments . . . 77

4.8 Discussion and Bibliographical Remarks . . . 81

5 Exploration without Rewards & Applications to Deep RL 85
5.1 Reward-Free Exploration in Finite MDPs . . . 86
5.2 Kernel-Based Bonuses for Exploration in Deep RL . . . 90
5.3 Related Work . . . 94
5.4 Experiments . . . 96
5.5 Discussion and Bibliographical Remarks . . . 104

6 Conclusion 107
6.1 Main Contributions & Directions for Future Work . . . 107
6.2 Software for Reinforcement Learning Research . . . 108

A Complements on Chapter 2 111
A.1 Proof of Theorem 2.6: Sample Complexity of `SmoothCruiser` . . . 111
A.2 Proof of Theorem 2.7: Consistency of `SmoothCruiser` . . . 115
A.3 Technical Lemmas . . . 123

B Complements on Chapter 3 125
B.1 Change of Distribution: Proof of Lemma 3.6 . . . 125
B.2 PAC-MDP Lower Bound: Proof of Corollary 3.10 . . . 126
B.3 Technical Lemmas for Lower-Bound Proofs . . . 127
B.4 Complements on the proof of Theorem 3.14 (Regret of `UCBVI`) . . . 128

C Complements on Chapter 4 133
C.1 Definitions . . . 133
C.2 Proof of Theorem 4.7 . . . 134
C.3 Proof Sketch for Theorem 4.12: Regret of `KeRNS` . . . 147
C.4 Proof of Theorem 4.14: Regret of `Kernel-UCBVI+RTDP` . . . 149
C.5 Detailed Description of `RS-KeRNS` . . . 152
C.6 Proof Sketch for Theorem 4.16: Regret of `RS-KeRNS` . . . 157
C.7 Technical Lemmas . . . 158

List of Figures 162

List of Algorithms 164

List of Tables 165

References 167

Chapter 1

Introduction

1.1 Overview

As living beings interacting with the world, we are constantly faced with decision-making problems. Which path to take in order to find food? What is the best strategy to fight a pandemic? Some of those problems can be modeled as seeking the behavior that maximizes a utility function, such as the amount of gathered food and the number of saved lives.[1]

Reinforcement learning (RL) provides a mathematical and algorithmic framework for utility maximization in which an *agent* interacts with an *environment* by taking *actions* and receiving *rewards*. In this framework, the utility is defined as the sum of rewards obtained throughout the interaction with the environment. Mathematically, the environment is modeled as a Markov decision process (MDP), and the agent is defined as a decision rule (or *policy*) that selects actions based on the history of its previous interactions and its knowledge about the current state of the environment.

Exploration in reinforcement learning A key challenge in RL is that the agent has little or no prior knowledge about the consequences of its actions on the environment: all it can do is collect samples from the environment by taking actions and observing their consequences. The performance of an RL algorithm is usually measured with respect to how many samples are required from the environment for it to learn a good policy, or with respect to the intensity of the "mistakes" it makes during learning. In order to achieve an optimal performance, it must handle two objectives simultaneously: *learn how the environment behaves* and *learn how to act optimally* while interacting with the environment. The first objective is called *exploration* and the second *exploitation*, and we refer to the task of balancing these objectives as the *exploration-exploitation*

[1] Nevertheless, the questions of how to define a utility function and whether we should aim for utility maximization at all might be subject to societal, political and philosophical considerations.

dilemma. To gain some intuition about this dilemma, imagine that you are at restaurant with free food, where you can eat pizza, falafel, or spaghetti all'arrabbiata. Assume that you have never tasted any of those dishes before. You start by trying the pizza and, unsurprisingly, you think it tastes good. At that moment, you must either decide to keep eating pizza (that is, to *exploit* your current knowledge) or to try other dishes to find out if they are better than pizza (to *explore* and gather more knowledge). Since your eating capacity is limited, you need to optimally balance exploration and exploitation in order to have the best possible experience in such situation. Reinforcement learning algorithms need to handle a similar dilemma when trying to maximize rewards in an unknown environment with limited resources.

Beyond finite state-spaces At every time step, the environment with which an agent interacts is described by a state variable s belonging to a state set (or space) $\mathcal{S}$. The complexity of exploration is related to the size of the set $\mathcal{S}$, which can be its cardinality (if it is finite) or some notion of dimension. For instance, if $\mathcal{S}$ is finite and no prior information is given about the environment, an agent is required to visit every reachable state $s \in \mathcal{S}$ in order to learn a good policy: otherwise, it might miss a state with very high rewards. This thesis studies theoretical and practical aspects of exploration in reinforcement learning, when the state space $\mathcal{S}$ is very large, and possibly continuous. With continuous state spaces, it is not possible to visit every single state $s \in \mathcal{S}$ in finite time: thus, we need to make assumptions on the regularity of the MDP. Here, we study regularity through *kernel functions* that measure the similarity between states. The intuition is that once the agent visits a given state s, it also gains information about all other states that are similar to s and avoids the need to visit every state. In the example above, we might have prior information saying that pizza is similar to spaghetti all'arrabbiata, which might prevent you from spending time exploring the spaghetti if you already tried pizza and found out you dislike tomato sauce.

In the next sections, we formalize the concepts of agent and environment and we define the performance criteria that are used to evaluate RL algorithms. Then, we end this chapter with an overview of the contributions presented in this thesis.

1.2 Markov Decision Processes

The environment with which an agent interacts is modeled as a *Markov decision process* [Put94], to which we refer as MDP. At every time $t \in \mathbb{N}^*$, the environment is in a state $s \in \mathcal{S}$ and the agent takes an action $a \in \mathcal{A}$, where $\mathcal{S}$ and $\mathcal{A}$ are the sets of possible states and actions, respectively. As a consequence, the agent receives a (possibly random) reward with expectation $r_t(s, a)$, where $r_t : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a (mean) reward function, and the state of the environment is changed according to a probability kernel $p_t(\cdot|s, a)$. An MDP is thus defined as the tuple

$\mathcal{M} := (\mathcal{S}, \mathcal{A}, \mu, (p_t, r_t)_{t\in\mathbb{N}^*})$, where μ is the probability measure on the set of states at $t = 1$, representing the distribution of initial states.

Assumption 1.1. *Unless explicitly stated, we assume in this thesis that the mean reward functions $(r_t)_{t\in\mathbb{N}^*}$ are known by the agent, and that there exists a constant $r_{\max} > 0$ such that $\sup_{t,s,a} |r_t(s,a)| \leq r_{\max}$. In Section 2.1, we see that the main challenge of a reinforcement learning algorithm is to handle the unknown transitions of the environment, and relaxing the assumption of known rewards is usually straightforward.*

Let $b_t = (s_1, a_1, \ldots, s_{t-1}, a_{t-1}, s_t)$ be a history of previous states and actions up to time t, where (s_i, a_i) are the state and action at time i, let $\mathcal{B}_t$ be the set of all possible histories b_t at time t and let $\mathcal{B} := \bigcup_{t\in\mathbb{N}^*} \mathcal{B}_t$ be the set of all histories. A *history-dependent policy*, denoted by $\boldsymbol{\pi}$, is a mapping from $\mathcal{B}$ to $\Delta(\mathcal{A})$, the set of probability distributions over the action set $\mathcal{A}$, and represents a rule to decide which action to take at time t, given a history b_t. A *Markov policy* π is a mapping from $\mathbb{N}^* \times \mathcal{S}$ to $\Delta(\mathcal{A})$ and represents a decision rule that does not take the history into account, besides the current state s_t and the time t. A Markov policy is *deterministic* if, instead of giving a probability distribution over actions, it returns a single action, *i.e.*, it is a mapping from $\mathbb{N}^* \times \mathcal{S}$ to $\mathcal{A}$. Furthermore, a *stationary* deterministic Markov policy is defined as a mapping from states $\mathcal{S}$ to actions $\mathcal{A}$. We denote by Π_{H}, Π_{M}, Π_{MD}, and Π_{SD} the sets of history-dependent, Markov, deterministic Markov, and stationary deterministic Markov policies, respectively.

1.2.1 Probabilistic Model

Infinite-Horizon

A policy $\boldsymbol{\pi}$ interacting with an MDP $\mathcal{M}$ defines a stochastic process denoted by $(S_t, A_t)_{t\in\mathbb{N}^*}$, where S_t and A_t are the random variables representing the state and action at time t. As explained by [LS20], the Ionescu-Tulcea theorem ensures the existence of a probability space $(\Omega, \mathcal{F}, \mathbf{P})$ such that

$$\mathbf{P}\left[S_1 \in \cdot\right] = \mu(\cdot), \quad \mathbf{P}\left[S_{t+1} \in \cdot | B_t, A_t\right] = p_t\left(\cdot | S_t, A_t\right), \text{ and } \quad \mathbf{P}\left[A_t \in \cdot | B_t\right] = \boldsymbol{\pi}\left(\cdot | B_t\right),$$

for any $t \in \mathbb{N}^*$, where $B_t = (S_1, A_1, \ldots, S_{t-1}, A_{t-1}, S_t)$. Additionally, we define by $\mathcal{F}_t$ the σ-algebra generated by B_t, and denote $\mathcal{F}_t := \boldsymbol{\sigma}(B_t)$. For a history-dependent policy $\boldsymbol{\pi}$, we denote by $\boldsymbol{\pi}_{t,B_t}$ the restriction of $\boldsymbol{\pi}$ to the subset of possible histories in $\mathcal{B}$ that start with the sequence B_t. Often, the past state-action pairs B_t will be implicitly defined by the context, and we denote $\boldsymbol{\pi}_{t,B_t}$ simply as $\boldsymbol{\pi}_t$.

Notice that the probability measure $\mathbf{P}$ depends on the policy $\boldsymbol{\pi}$ and on the MDP $\mathcal{M}$. When necessary, we explicit this dependence by denoting $\mathbf{P}$ as $\mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}$. We denote by $\mathbf{P}^{B_t}_{\boldsymbol{\pi},\mathcal{M}}$ the pushforward measure of B_t under $\mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}$.

Finite-Horizon

In this thesis, we will also consider trajectories $(S_t, A_t)_{t\in\mathbb{N}^*}$ that are split into *episodes* of length (or *horizon*) $H \in \mathbb{N}^*$. Consider the bijection $m : \mathbb{N}^* \times \{1, \dots, H\} \to \mathbb{N}^*$ such that $m(t, h) = (t-1)H + h$, and let $S_h^t := S_{m(t,h)}$, $A_h^t := A_{m(t,h)}$, and $B_h^t := B_{m(t,h)}$. We refer to the sub-trajectory $(S_h^t, A_h^t)_{h\in\{1,\dots,H\}}$ as the t-th *episode* generated by the policy $\boldsymbol{\pi}$. Also, we denote by $\mathcal{F}_h^t := \boldsymbol{\sigma}(B_h^t)$ the σ-algebra generated by B_t, and by $\boldsymbol{\pi}_h^t := \boldsymbol{\pi}_{m(h,t)}$ the restriction of the policy $\boldsymbol{\pi}$ to the subset of histories starting with B_h^t.

1.2.2 Performance Criteria

In RL, the utility of a policy $\boldsymbol{\pi}$ in an MDP $\mathcal{M}$ is defined as the weighted sum of rewards gathered by taking the actions prescribed by $\boldsymbol{\pi}$ in $\mathcal{M}$:

$$\mathcal{U}^\pi = \sum_{t=1}^{\infty} \gamma^{t-1} r_t(S_t, A_t),$$

where $\gamma \in [0, 1]$ is a constant controlling the preference of the agent for short-term rewards, and is called *discount factor*. We are interested in finding policies that maximize the expected utility $\mathbf{E}[\mathcal{U}^\pi]$, and, to that end, it is convenient to introduce *value functions*. The value of $\boldsymbol{\pi}$ starting from a state s at time t is defined as

$$V_t^{\boldsymbol{\pi}}(s) := \mathbf{E}_{\boldsymbol{\pi}}\left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}(S_{t'}, A_{t'}) \middle| S_t = s\right]$$

and $s \mapsto V_t^{\boldsymbol{\pi}}(s)$ is the value function of $\boldsymbol{\pi}$ at time t. Similarly, we define the action-value function $(s, a) \mapsto Q_t^{\boldsymbol{\pi}}(s, a)$, also known as *Q-function*, as the value of the policy that selects action a in state s at time t and follows the policy $\boldsymbol{\pi}$ thereafter:

$$Q_t^{\boldsymbol{\pi}}(s, a) := \mathbf{E}_{\boldsymbol{\pi}}\left[\sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}(S_{t'}, A_{t'}) \middle| S_1 = s, A_t = a\right].$$

Since we are dealing with infinite sums, additional conditions are required for V^π and Q^π to be well-defined, both mathematically and in terms of which kind of properties we would like for the policy $\boldsymbol{\pi}$ that maximizes $\mathbf{E}[\mathcal{U}^\pi]$. Different conditions will lead to different performance criteria, and the most common criteria are introduced below. Except for the finite-horizon criterion, we assume that the rewards and transition probabilities are time-invariant, *i.e.*, $r_t(s, a) = r(s, a)$ $p_t(\cdot|s, a) = p(\cdot|s, a)$, where r is a fixed reward function and p is a fixed transition kernel.

Finite Horizon In situations where the agent is evaluated for its performance during a finite amount of time steps $H \in \mathbb{N}^*$, we set $r_t = 0$ for $t > H$. The discount factor is often set to $\gamma = 1$ in this setting.

Discounted Reward If the agent has a preference for short-term rewards, but still considers the long-term impact of its actions, we can take a discount factor γ that is strictly smaller than 1, which defines the discounted-reward criterion.

Average Reward If the agent is evaluated for its performance during a large (possibly infinite) amount of time and with no preference for short-term rewards, we take $\gamma \to 1$ and use the average-reward criterion, where the policy is evaluated by $\lim_{\gamma\to 1}(1-\gamma)V_1^{\boldsymbol{\pi}}(s)$ [Ber11, Chapter 4].

Stochastic Shortest Path This setting considers $\gamma = 1$ and applies to MDPs where there exists an absorbing state s_{g} (*i.e.*, a state such that $p(s_{\mathrm{g}}|s_{\mathrm{g}}, a) = 1$ for any action a, also called *goal state*) such that $r(s_g, \cdot) = 0$ and $r(s, a) \leq 0$. We can interpret r as a negative cost function. If the cost in a state s is seen as the average time spent on a transition between s and its possible next states, a policy that maximizes the sum of rewards must reach the absorbing state s_{g} as quickly as possible. Hence, we seek the *shortest path* to the goal state s_{g}.

In this thesis, we focus mostly on the finite-horizon criterion (except for Chapter 2, where we consider the discounted-reward criterion) for mathematical convenience. Indeed, this criterion imposes fewer constraints on the reinforcement-learning algorithms, by avoiding infinite sums. Nevertheless, some of the algorithmic ideas introduced for finite-horizon problems might be adapted to infinite-horizon discounted-reward problems. This is due to the fact that

$$\left|\sum_{t=1}^{\infty}\gamma^{t-1}r(S_t, A_t) - \sum_{t=1}^{H}\gamma^{t-1}r(S_t, A_t)\right| \leq \frac{\gamma^H}{1-\gamma}r_{\max}.$$

Hence, for $H \geq \log_\gamma \frac{\varepsilon(1-\gamma)}{r_{\max}}$, the value of a policy varies at most by ε from the finite-horizon to the discounted-reward criterion, provided that we use a discounted factor $\gamma < 1$ and assume that the transitions and rewards are time-invariant.

1.2.3 Characterization of Optimal Policies

A history-dependent policy $\boldsymbol{\pi} \in \Pi_{\mathrm{H}}$ is *optimal* if $V_1^{\boldsymbol{\pi}}(s) = V_1^*(s) := \sup_{\boldsymbol{\pi}'\in\Pi_{\mathrm{H}}} V_1^{\boldsymbol{\pi}'}(s)$ for all $s \in \mathcal{S}$, where V_1^* is the *optimal value function*. Notice that $\boldsymbol{\pi}$ is optimal if and only if it maximizes the expected utility $\mathbf{E}\left[\mathcal{U}^{\boldsymbol{\pi}}\right]$ for any possible initial distribution μ. It can be shown that, for any $\boldsymbol{\pi} \in \Pi_{\mathrm{H}}$ and for each $s \in \mathcal{S}$, there exists a Markov policy $\pi' \in \Pi_{\mathrm{M}}$ such that $V_1^{\boldsymbol{\pi}}(s) = V_1^{\pi'}(s)$ [Put94, Theorem 5.5.3]. Hence, if there exists an optimal policy for an MDP $\mathcal{M}$, there exists an optimal *Markov* policy for $\mathcal{M}$, since $\sup_{\boldsymbol{\pi}\in\Pi_{\mathrm{H}}} V_1^{\boldsymbol{\pi}}(s) = \sup_{\pi\in\Pi_{\mathrm{M}}} V_1^{\pi}(s)$.

To characterize optimal policies and optimal value functions, we will use the *Bellman operator*, and we consider the finite-horizon and the discounted-reward criteria. Let $\mathcal{V}$ be the space of bounded functions from $\mathcal{S}$ to $\mathbb{R}$ equipped with the infinity norm $\|\cdot\|_\infty$, such that $\|f\|_\infty :=$

$\sup_{x\in\mathcal{S}} |f(x)|$. The Bellman operator at time t is defined as $\mathcal{T}_{t,\gamma} : \mathcal{V} \to \mathcal{V}$ such that

$$\left[\mathcal{T}_{t,\gamma} V\right](s) = \sup_{a\in\mathcal{A}} \left\{ r_t(s,a) + \gamma \int_{\mathcal{S}} V(s') p_t(\mathrm{d}s'|s,a) \right\}. \tag{1.1}$$

Remark 1.2. *In general, additional conditions on the MDP are required for the Bellman operator to be well-defined: for instance, since $\mathcal{T}_{t,\gamma} V$ involves an integration of V over $\mathcal{S}$ with respect to the transition kernel, we must ensure that we restrict ourselves to the space of* measurable *functions in $\mathcal{V}$, and that, after taking the supremum over $a \in \mathcal{A}$, the result remains measurable. Such technical considerations are discussed by [Put94, Section 2.3] and references therein.*

The Bellman operator can also be defined for action-value functions. Let $\mathcal{Q}$ be the space of bounded functions from $\mathcal{S} \times \mathcal{A}$ to $\mathbb{R}$, also equipped with the infinity norm $\|\cdot\|_\infty$ such that $\|f\|_\infty := \sup_{x,a\in\mathcal{S}\times\mathcal{A}} |f(x,a)|$. We define $\overline{\mathcal{T}}_{t,\gamma} : \mathcal{Q} \to \mathcal{Q}$ as

$$\left[\overline{\mathcal{T}}_{t,\gamma} Q\right](s,a) = r_t(s,a) + \gamma \int_{\mathcal{S}} \sup_{a'\in\mathcal{A}} Q(s',a') p_t(\mathrm{d}s'|s,a). \tag{1.2}$$

Finite-Horizon Criterion In this case, since $r_t > 0$ for $t > H$, the value function of any policy π satisfies $V_t^\pi = 0$ for $t > H$. Hence, the actions taken after $t > H$ are irrelevant and, for the purposes of utility maximization, we can focus on the set of Markov policies defined as a mapping from $\{1,\dots,H\} \times \mathcal{S}$ to $\Delta(\mathcal{A})$. Let $V_t^*(s) := \sup_{\pi\in\Pi_{\mathrm{M}}} V_t^\pi(s)$ and $Q_t^*(s,a) := \sup_{\pi\in\Pi_{\mathrm{M}}} Q_t^\pi(s,a)$ be the optimal value functions at time $t \in \{1,\dots,H\}$. It can be shown that $(V_t^*)_t$ satisfy the Bellman optimality equations [Put94, Theorem 4.3.2]:

$$\begin{aligned} &V_{H+1}^*(s) = 0, \quad \forall s \in \mathcal{S}, \\ &V_t^*(s) = \left[\mathcal{T}_{t,\gamma=1} V_{t+1}^*\right](s), \quad \forall s \in \mathcal{S}, \quad \forall t \in \{1,\dots,H\}. \end{aligned}$$

If the supremum in Equation (1.1) is attained for any state $s \in \mathcal{S}$ and any time $t \in \{1,\dots,H\}$, there exists a deterministic Markov policy $\pi^* \in \Pi_{\mathrm{MD}}$ such that

$$\pi^*(t,s) \in \operatorname*{argmax}_{a\in\mathcal{A}} \left\{ r_t(s,a) + \int_{\mathcal{S}} V_{t+1}^*(s') p_t(\mathrm{d}s'|s,a) \right\} \tag{1.3}$$

and $V_t^*(s) = V_t^{\pi^*}(s)$ for all (s,t). That is, π^* is an optimal policy.

Furthermore, the Bellman equations can be also be written for action-value functions as follows:

$$\begin{aligned} &Q_{H+1}^*(s,a) = 0, \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}, \\ &Q_t^*(s,a) = \left[\overline{\mathcal{T}}_{t,\gamma=1} Q_{t+1}^*\right](s,a), \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}, \quad \forall t \in \{1,\dots,H\}, \end{aligned}$$

where $(Q_t^*)_t$ are the optimal action-value functions, and we have $V_t^*(s) = \sup_{a\in\mathcal{A}} Q_t^*(s,a)$ for $t \in \{1,\dots,H\}$. If a policy $\pi^* \in \Pi_{\mathrm{MD}}$ satisfies $\pi^*(t,s) \in \operatorname{argmax}_{a\in\mathcal{A}} Q_t^*(s,a)$, provided that the maximum exists, then π^* is an optimal policy.

Discounted-Reward Criterion In this case, recall that the rewards and transitions are assumed to be time-invariant, which implies that: (i) the Bellman operators $\mathcal{T}_{t,\gamma}$ and $\overline{\mathcal{T}}_{t,\gamma}$ are time-invariant, and we denote them by $\mathcal{T}_\gamma$ and $\overline{\mathcal{T}}_\gamma$, respectively ; and (ii) the value functions V_t^*, Q_t^*, V_t^π and Q_t^π for any π are also time-invariant, and are denoted by V^*, Q^*, V^π and Q^π.

If the Bellman operator $\mathcal{T}_\gamma$ admits a fixed point $\overline{V} \in \mathcal{V}$, that is, $\mathcal{T}_\gamma \overline{V} = \overline{V}$, then $\overline{V}$ is unique and is the optimal value function $\overline{V} = V^*$ [Put94, Theorem 6.2.2]. Since $\mathcal{V}$ is a Banach space and the Bellman operator $\mathcal{T}_\gamma$ is a contraction mapping for $\gamma \in [0,1[$, a fixed point of $\mathcal{T}_\gamma$ is guaranteed to exist by Banach's fixed-point theorem.

Furthermore, if the supremum

$$\sup_{a\in\mathcal{A}} \left\{ r(s,a) + \gamma \int_{\mathcal{S}} V^*(s') p(\mathrm{d}s'|s,a) \right\}$$

is attained for all s, then a stationary deterministic Markov policy $\pi^* \in \Pi_{\mathrm{SD}}$ satisfying

$$\pi^*(s) \in \operatorname*{argmax}_{a\in\mathcal{A}} \left\{ r(s,a) + \gamma \int_{\mathcal{S}} V^*(s') p(\mathrm{d}s'|s,a) \right\}$$

is an optimal policy [Sze10, Theorem 2].

Similar results hold for the Bellman operator $\overline{\mathcal{T}}_\gamma$ for action-value functions: it admits a fixed-point $\overline{Q}$ such that $\overline{Q} = Q^*$. Also, if a policy $\pi^* \in \Pi_{\mathrm{SD}}$ satisfies $\pi^*(s) \in \operatorname{argmax}_{a\in\mathcal{A}} Q^*(s,a)$, then π^* is an optimal policy.

1.2.4 Value Iteration

In Section 1.2.3, we saw that optimal policies can be derived from optimal value functions, which can be defined either through recursive applications of the Bellman operator (in the finite-horizon setting) or as its fixed-point (in the discounted-reward setting).

The *Value Iteration algorithm* can be used to compute (or approximate) optimal value functions, and is defined as follows:

(i) set a number of iterations N and define an initial value function $Q_{N,N+1} := 0$;

(ii) apply the Bellman operator for N steps: $Q_{N,n} := \overline{\mathcal{T}}_{n,\gamma} Q_{N,n+1}$ for $n \in \{1,\dots,N\}$.

This algorithm is also called Q-Value Iteration (QVI), since it operates on action-value functions. In the finite-horizon setting, we have $Q_{N,n} = Q_n^*$ for $n \in \{1, \dots, H\}$ by setting $N = H$, and we can compute an optimal policy as $\pi^*(t, s) = \operatorname{argmax}_{a \in \mathcal{A}} Q_{N,t}(s, a)$ if the maximum exists.

In the discounted-reward setting, it can be shown that

$$\|Q_{N,1} - Q^*\|_\infty \leq \gamma^N \|Q^*\|_\infty \leq \frac{\gamma^N}{1-\gamma} r_{\max}.$$

For $N \geq \log_\gamma \frac{\varepsilon(1-\gamma)}{r_{\max}}$, the approximation error on the action-value function is bounded by ε and an approximate policy can be defined using $Q_{N,1}$ as $\pi^*_{\text{approx}}(s) \in \operatorname{argmax}_{a \in \mathcal{A}} Q_{N,1}(s, a)$ provided that the maximum exists. It can be shown that π^*_{approx} satisfies [SY94]

$$\|V^{\pi^*_{\text{approx}}} - V^*\|_\infty \leq \frac{2\varepsilon}{1-\gamma}.$$

Consequently, if the MDP is known, *i.e.*, we have access to the reward functions and the transition kernels, and if we can apply the Bellman operator in a *computationally* efficient way, the value iteration algorithm allows us to compute or approximate an optimal policy. If the set of states and actions are *finite*, the computational complexity of value iteration is $\Theta\left(N |\mathcal{S}|^2 |\mathcal{A}|\right)$, where N is the number of iterations. Hence, for very large or continuous state-action sets, we are in general required to use other approximation methods.

1.3 Sampling Models

In reinforcement learning, we do not have full knowledge of the MDP, *i.e.*, the reward functions and the transition kernels are not given *a priori*. The agent has only access to a *sampling model* of the environment, which represents its interaction with the world. Thus, a reinforcement learning algorithm must learn a good policy only by *sampling* from (a model of) the MDP. The two most common sampling models are the *generative model* and the *online model*, as defined below.

Generative Model The MDP is assumed to be time-invariant and the agent has access to a simulator that takes as input an arbitrary state-action pair (s, a) and returns R and S', where R is a random variable of mean $r(s, a)$ and S' is a random variable following the distribution $p(\cdot|s, a)$. Every output from simulator is independent of all the previous and future outputs.

Online Model At each time t, representing the amount of previous calls to the model, the model is at a state s_t. When the model receives an action a_t as input, it returns R and S', where $\mathbf{E}\left[R|S_1, A_1, \dots S_{t-1}, A_{t-1}, S_t\right] = r_t(S_t, A_t)$ and $S' \sim p_t(\cdot|S_t, A_t)$.

Notation We denote by

$$R, S' \leftarrow \texttt{GenerativeModel}(s, a)$$

the act of sampling a reward R and a transition S' from (s, a) using a generative model, and by

$$R, S' \leftarrow \texttt{OnlineModel}_t(a)$$

the act of sampling R and S' by taking action a in an online model at time t. When the reward functions are assumed to be known (Assumption 1.1), we only need to sample transitions S' from the models, which we denote by $S' \leftarrow \texttt{GenerativeModel}(s, a)$ and $S' \leftarrow \texttt{OnlineModel}_t(a)$.

Notice that the generative model is a stronger assumption since the agent can query any state-action pair at any time, whereas with an online model, the agent cannot query arbitrary states, as the state of the model is determined by its previous actions and by the transition dynamics of the MDP.

1.4 Evaluating Reinforcement Learning Algorithms

Reinforcement learning algorithms interact with the environment, represented by a generative or by an online model, and they either (i) interact with the model for an indefinite amount of time; or (ii) stop after a number τ of queries to the model (which is a possibly random stopping time). In the first case, we define an RL algorithm as a history-dependent policy $\boldsymbol{\pi}$ used to take actions in the environment.[2] In the second case, the algorithm eventually stops and outputs a policy, and we define it as a triple $(\boldsymbol{\pi}, \tau, \widehat{\pi}_\tau)$, where τ is a stopping time with respect to the filtration $(\mathcal{F}_t)_{t \in \mathbb{N}^*}$ and $\widehat{\pi}_\tau$ is a Markov policy given as the output.

An agent is evaluated either with respect to how may queries to the model are required before it is guaranteed to provide a good approximation of the optimal value function or the optimal policy, or with respect to the amount or the value of the "mistakes" it makes during any fixed number of interactions. These performance criteria are commonly formalized under two frameworks: the *sample complexity* framework, and the *regret* minimization framework. The precise definitions of such criteria may vary according to the MDP objective (finite-horizon, discounted reward, average reward, stochastic shortest path), and according to the RL algorithm (whether it outputs a value function or a policy, or if it runs for an indefinite number of time steps), and are introduced in the subsequent chapters when we analyze specific algorithms.

[2] Notice that, with a generative model, the agent can choose from which state the transition will be sampled, while a policy recommends a distribution over actions. This issue can be easily solved by slightly modifying the definition of a policy so that it outputs a state to be sampled, in addition to a probability on the action space.

1.5 Contributions

1.5.1 Outline

Planning with a Generative Model In Chapter 2, we analyze algorithms that use a generative model of the environment to define policies. In this case, it is possible to obtain algorithms whose sample complexities do not depend on the size of the state space $\mathcal{S}$, although, in general, they may have a non-polynomial dependence on $1/\varepsilon$, where ε is the approximation error of the output of the algorithm. Our contribution is a novel algorithm showing that, if we consider *regularized MDPs*, it is possible to obtain a sample complexity that is polynomial in $1/\varepsilon$ and, at the same time, independent of $\mathcal{S}$.

Online Interaction with Finite MDPs In Chapter 3, we study worst-case bounds on the sample complexity and regret of algorithms interacting with an MDP through an online model, in the particular case where the state-action set is finite. We provide unified, simple and complete proofs of the lower bounds in different settings. Since these lower bounds depend on the cardinality of the state-action sets, they show the need of structural assumptions when dealing with large or continuous MDPs. We also revisit the proof of a near-optimal algorithm for regret minimization in finite MDPs called `UCBVI` [AOM17], which we extend to continuous MDPs in Chapter 4.

A Kernel-Based Approach to Exploration in Continuous MDPs In Chapter 4, we tackle exploration in continuous MDPs. We assume that we have access to a distance function between states and actions, and that, if two states are close to each other, their rewards and transitions are similar. A kernel (or similarity) function is constructed based on this distance: state-action pairs that are close to each other are more similar than distant pairs. Through a kernel-based model of the environment, we show that it is possible to explore continuous MDPs by generalizing to any state the information gathered from previously visited states. We propose an algorithm called `Kernel-UCBVI` and prove a regret bound that depends on the covering dimension of the state-action space. By extending `Kernel-UCBVI` to use time-dependent kernels, we propose an algorithm called `KeRNS` that is able to handle *non-stationary environments*, where the agent interacts with a different MDP in each episode.

Exploration without Rewards & Applications to Deep Reinforcement Learning In Chapter 5, we review the `RF-UCRL` [Kau+21] and `RF-Express` [Mé+21a] algorithms that allow an agent to collect relevant data in finite MDPs without rewards and, inspired by those algorithms and borrowing ideas from `Kernel-UCBVI`, we propose a deep reinforcement learning method for reward-free exploration.

1.5.2 List of Publications

The list below contains the publications that I co-authored during my PhD.[3] Chapter 2 is based on [Gri+19] about planning in regularized MDPs; Chapter 3 is based on [Dom+21b] about minimax lower bounds for finite-horizon RL; Chapter 4 is based on [Dom+21d] and [Dom+21c] about kernel-based exploration in stationary and non-stationary MDPs; and Chapter 5 is based on [Kau+21] and [Mé+21a] about reward-free exploration in finite MDPs, and on [Dom+21e] about reward-free exploration in deep RL. I also worked on the development of the `rlberry` library in Python [Dom+21a] whose goal is to facilitate empirical research in RL, which I briefly discuss in Chapter 6.

Publications in international conferences with proceedings

- Jean-Bastien Grill*, Omar Darwiche Domingues*, Pierre Ménard, Rémi Munos, Michal Valko. **Planning in entropy-regularized Markov decision processes and games**. In *Advances in Neural Information Processing Systems 33* (NeurIPS), 2019 [Gri+19]. Presented in Chapter 2.
- Omar Darwiche Domingues, Pierre Ménard, Emilie Kaufmann, Michal Valko. **Episodic Reinforcement Learning in Finite MDPs: Minimax Lower Bounds Revisited**. In *32nd International Conference on Algorithmic Learning Theory* (ALT), 2021 [Dom+21b]. Presented in Chapter 3.
- Omar Darwiche Domingues, Pierre Ménard, Matteo Pirotta, Emilie Kaufmann, Michal Valko. **Kernel-Based Reinforcement Learning: A Finite-Time Analysis**. In *38th International Conference on Machine Learning* (ICML), 2021 [Dom+21d]. Presented in Chapter 4.
- Omar Darwiche Domingues, Pierre Ménard, Matteo Pirotta, Emilie Kaufmann, Michal Valko. **A Kernel-Based Approach to Non-Stationary Reinforcement Learning in Metric Spaces**. In *24th International Conference on Artificial Intelligence and Statistics* (AISTATS), 2021 [Dom+21c]. Presented in Chapter 4.
- Emilie Kaufmann, Pierre Ménard, Omar Darwiche Domingues, Anders Jonsson, Edouard Leurent, Michal Valko. **Adaptive reward-free exploration**. In *32nd International Conference on Algorithmic Learning Theory* (ALT), 2021 [Kau+21]. Discussed in Chapter 5.
- Pierre Ménard, Omar Darwiche Domingues, Anders Jonsson, Emilie Kaufmann, Edouard Leurent, Michal Valko. **Fast active learning for pure exploration in reinforcement learning**. In *38th International Conference on Machine Learning* (ICML), 2021 [Mé+21a]. Discussed in Chapter 5.

[3] Where * means equal contribution.

Workshop presentations in international conferences

- Omar Darwiche Domingues, Corentin Tallec, Rémi Munos, Michal Valko. **Density-Based Bonuses on Learned Representations for Reward-Free Exploration in Deep Reinforcement Learning**. In *Workshop on Unsupervised Reinforcement Learning at the 38th International Conference on Machine Learning* (ICML), 2021 [Dom+21e]. Presented in Chapter 5.

Software

- Omar Darwiche Domingues, Yannis Flet-Berliac, Edouard Leurent, Pierre Ménard, Xuedong Shang, Michal Valko. **`rlberry` - A reinforcement learning library for research and education**. GitHub repository, available at https://github.com/rlberry-py/rlberry [Dom+21a].

Collaborations not presented in this thesis

- Anders Jonsson, Emilie Kaufmann, Pierre Ménard, Omar Darwiche Domingues, Edouard Leurent, Michal Valko. **Planning in Markov Decision Processes with Gap-Dependent Sample Complexity**. In *Advances in Neural Information Processing Systems 34* (NeurIPS), 2020 [Jon+20].
- Pierre Ménard, Omar Darwiche Domingues, Xuedong Shang, Michal Valko. **UCB Momentum Q-learning: Correcting the bias without forgetting**. In *38th International Conference on Machine Learning* (ICML), 2021 [Mé+21b].
- Jean Tarbouriech, Omar Darwiche Domingues, Pierre Ménard, Matteo Pirotta, Michal Valko, Alessandro Lazaric. **Adaptive Multi-Goal Exploration**. Preprint, 2021 [Tar+21a].

Chapter 2

Planning with a Generative Model

In this chapter, we refer to *planning* as the task of approximating the optimal value function of an MDP, either for a fixed state $s \in \mathcal{S}$, or simultaneously for all states. We study the sample complexity of planning algorithms relying on a generative model, considering the discounted-reward criterion. We start from a model-based approach, whose complexity depends on the size of the MDP, then we move to sampling-based approaches for which we are able to guarantee sample complexities that are independent of the number of states in the environment.

The last part of this chapter is based on the paper [Gri+19], which I co-authored, about planning in regularized MDPs and two-player games.

Contents

2.1 Model-Based Q-Value Iteration **14**

2.2 `SparseSampling`: Planning in Arbitrary State Spaces **16**

2.3 `SmoothCruiser`: Planning in Regularized MDPs **21**

2.4 Discussion and Bibliographical Remarks **27**

2.1 Model-Based Q-Value Iteration

As explained in Section 1.2.4, if we have a finite MDP for which the transition kernel p and the reward function r are known, the Q-Value Iteration (QVI) algorithm can be used to approximate the optimal value function. By running $N = \left\lceil \log_\gamma \frac{\varepsilon(1-\gamma)}{r_{\max}} \right\rceil$ iterations, we obtain a value function V_N that is an ε-approximation of the optimal value function V^*, *i.e.* such that $\|V^* - V_N\|_\infty \leq \varepsilon$. The total runtime of QVI is of order $\Theta\left(NS^2A\right)$, where $S := |\mathcal{S}|$ and $A := |\mathcal{A}|$.

Hence, if we are given a generative model for a finite MDP, an intuitive strategy is to use the generative model to estimate r and p, and run QVI on the estimated model. Such approach is called Model-Based QVI, or `MBQVI`. The sample complexity of `MBQVI` has been initially studied by Kearns and Singh [KS99], and Azar et al. [AMK12] present a lower bound and an improved upper bound.

`MBQVI` is detailed in Algorithm 2.1. For each state-action pair (s, a), it samples n transitions from the generative model, and estimates the transition kernel and the reward function. Finally, it returns $\widehat{Q}_k$, a Q-function obtained by applying k times the Bellman operator defined with the estimated model.

Algorithm 2.1: `MBQVI`

input: `GenerativeModel` , $n \in \mathbb{N}^*$, $\gamma \in [0, 1[$
for $(s, a) \in \mathcal{S} \times \mathcal{A}$ **do**
  for $i \in \{1, \dots, n\}$ **do**
    $r_i, s'_i \leftarrow$ `GenerativeModel`(s, a)
  # estimate model
  $\widehat{p}(s'|s, a) \leftarrow \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{s'_i = s'\}$ for $s' \in \mathcal{S}$
  $\widehat{r}(s, a) \leftarrow \frac{1}{n} \sum_{i=1}^{n} r_i$
run value iteration
$\widehat{Q}_0 \leftarrow 0$
for $j \in \{1, \dots, k\}$ **do**
  # apply Bellman operator of the empirical model
  $\widehat{Q}_j(s, a) \leftarrow \widehat{r}(s, a) + \gamma \sum_{s'} \max_{a'} Q_{j-1}(s', a') \widehat{p}(s'|s, a)$ for $(s, a) \in \mathcal{S} \times \mathcal{A}$
return $\widehat{Q}_k$

Theorem 2.1 (Sample complexity of `MBQVI`). *Let $S := |\mathcal{S}|$ and $A := |\mathcal{A}|$. If $n = \left\lceil \frac{32 r_{\max}^2}{(1-\gamma)^4 \varepsilon^2} \log\left(\frac{4SA}{\delta}\right) \right\rceil$ and $k = \left\lceil \log_{1/\gamma}\left(\frac{2r_{\max}}{\varepsilon(1-\gamma)}\right) \right\rceil$, then, with probability at least $1 - \delta$, the output $\widehat{Q}_k$ of `MBQVI` satisfies $\|\widehat{Q}_k - Q^*\|_\infty \leq \varepsilon$. That is, with $\mathcal{O}\left(\frac{SAr_{\max}^2}{(1-\gamma)^4 \varepsilon^2} \log\left(\frac{4SA}{\delta}\right)\right)$ calls to the generative model, `MBQVI` produces an ε-approximation of the optimal Q-function with high probability.*

Theorem 2.1 shows that $\widetilde{\mathcal{O}}\left(SA(1-\gamma)^{-4}/\varepsilon^2\right)$ calls to the generative model are enough to guarantee that `MBQVI` outputs an ε approximation of the optimal Q-function. Although this scales

with $(1-\gamma)^{-4}$, it was shown by Azar et al. [AMK12] that the optimal dependence is $(1-\gamma)^{-3}$, and that it can be achieved by a shaper analysis using Bernstein concentration inequalities and Bellman-type equations for the variance of the value functions. We provide below the proof of Theorem 2.1, which is adapted from [KS99] and [AMK12].

Proof. (*of Theorem 2.1*) Consider the event $\mathcal{E} := \mathcal{E}_r \cap \mathcal{E}_p$, where

$$\mathcal{E}_r := \left\{ \forall (s,a) \in \mathcal{S} \times \mathcal{A},\ |\widehat{r}(s,a) - r(s,a)| \leq \sqrt{\frac{2r_{\max}^2}{n} \log\left(\frac{4SA}{\delta}\right)} \right\}, \quad \text{and}$$

$$\mathcal{E}_p := \left\{ \forall (s,a) \in \mathcal{S} \times \mathcal{A},\ \left| \sum_{s' \in \mathcal{S}} \left(\widehat{p}(s'|s,a) - p(s'|s,a)\right) V^*(s') \right| \leq \sqrt{\frac{2r_{\max}^2}{(1-\gamma)^2 n} \log\left(\frac{4SA}{\delta}\right)} \right\}.$$

By Hoeffding's inequality, we have $\mathbf{P}\left[\mathcal{E}\right] \geq 1-\delta$. Let $\widehat{Q}^*$ be the optimal Q-function in the MDP defined by the estimated rewards $\widehat{r}$ and transitions $\widehat{p}$. We decompose the error as

$$\|\widehat{Q}_k - Q^*\|_\infty \leq \|\widehat{Q}_k - \widehat{Q}^*\|_\infty + \|\widehat{Q}^* - Q^*\|_\infty.$$

By the contraction property of the Bellman operator, we have $\|\widehat{Q}_k - \widehat{Q}^*\|_\infty \leq \gamma^k r_{\max}/(1-\gamma)$. Let $\Delta := \|\widehat{Q}^* - Q^*\|_\infty$. On the event $\mathcal{E}$, we have

$$\begin{aligned}
\Delta &\leq \max_{s,a} \left[|\widehat{r}(s,a) - r(s,a)| + \gamma \left| \sum_{s' \in \mathcal{S}} \left(\widehat{p}(s'|s,a) - p(s'|s,a)\right) V^*(s') \right| \right] + \gamma\Delta \\
&\leq 2\sqrt{\frac{2r_{\max}^2}{(1-\gamma)^2 n} \log\left(\frac{4SA}{\delta}\right)} + \gamma\Delta.
\end{aligned}$$

Consequently,

$$\|\widehat{Q}_k - Q^*\|_\infty \leq \frac{\gamma^k r_{\max}}{1-\gamma} + \frac{2}{1-\gamma}\sqrt{\frac{2r_{\max}^2}{(1-\gamma)^2 n} \log\left(\frac{4SA}{\delta}\right)} \leq \varepsilon\,,$$

for $n = \left\lceil \frac{32 r_{\max}^2}{(1-\gamma)^4 \varepsilon^2} \log\left(\frac{4SA}{\delta}\right) \right\rceil$ and $k = \left\lceil \log_{1/\gamma}\left(\frac{2r_{\max}}{\varepsilon(1-\gamma)}\right) \right\rceil$. □

In the proof of Theorem 2.1, we see that estimating the quantities related to the transitions p (see event $\mathcal{E}_p$) requires more samples than estimating the reward function r (see event $\mathcal{E}_r$). In this case, the main "source" of sample complexity is the fact that the transitions are unknown to the agent. Hence, for simplicity, we assume from now on that the reward functions are known (Assumption 1.1), and generalizing the results in this thesis to the case of unknown rewards is straightforward.

2.2 `SparseSampling`: Planning in Arbitrary State Spaces

Theorem 2.1 and the lower bound by [AMK12] show that the sample complexity of `MBQVI` scales with the number of states S. Consequently, if the state space $\mathcal{S}$ is arbitrary and possibly continuous, `MBQVI` cannot be used. The `SparseSampling` algorithm introduced by Kearns et al. [KMN02], instead of estimating a value function for all states $s \in \mathcal{S}$, focuses on estimating $V^*(s)$ and $Q^*(s, \cdot)$ for a *fixed* state s. This is done purely by sampling from the generative model, and has a sample complexity that is independent of the size of $\mathcal{S}$. Then, using `SparseSampling` as subroutine, it is possible to implement a near-optimal policy (see Lemma 2.3): every time we need to select an action in a state s, we run `SparseSampling` to compute an approximation $\widehat{Q}^*(s, \cdot)$ of $Q^*(s, \cdot)$ and choose the action $a \in \operatorname{argmax}_a \widehat{Q}^*(s, a)$.

In this section, we study the sample complexity of `SparseSampling` (Algorithm 2.2), which is non-polynomial in $1/\varepsilon$. In the next section, we propose an algorithm called `SmoothCruiser` whose sample complexity is polynomial, provided that we regularize the MDP. For large values of ε, `SmoothCruiser` behaves like `SparseSampling`, hence we will present here a short proof of the sample complexity of `SparseSampling`, adapted from Kearns et al. [KMN02].

Algorithm 2.2: `SparseSampling`

global parameters: `GenerativeModel`, maximum depth $H \in \mathbb{N}^*$, width $C \in \mathbb{N}^*$, $\gamma \in [0, 1[$.
input: state $s \in \mathcal{S}$, depth $h \in \mathbb{N}^*$.
if $h = H$ **then**
  $\widehat{Q}_h(s, a) \leftarrow 0$ for $a \in \mathcal{A}$
else
  for $a \in \mathcal{A}$ **do**
    for $i \in \{1, \dots, C\}$ **do**
      $R^{(i)}_{s,a}, Z^{(i)}_{s,a} \leftarrow$ `GenerativeModel`(s, a)
      $\widehat{Q}_{h+1}(Z^{(i)}_{s,a}, \cdot) \leftarrow$ `SparseSampling`$(Z^{(i)}_{s,a}, h+1)$
      $\widehat{V}_{h+1}(Z^{(i)}_{s,a}) \leftarrow \max_{a'} \widehat{Q}_i(Z^{(i)}_{s,a}, a')$
    $\widehat{Q}_h(s, a) \leftarrow \frac{1}{C} \sum_{i=1}^{C} \left(R^{(i)}_{s,a} + \gamma \widehat{V}_{h+1}(Z^{(i)}_{s,a}) \right)$
return: $\widehat{Q}_h(s, \cdot)$

In order to estimate the Q-function in a state s, `SparseSampling` builds a look-ahead tree starting from s by sampling from the generative model. At the root of the tree, for each action a, it samples C rewards and next states, denoted by $R^{(i)}_{(s,a)}, Z^{(i)}_{(s,a)}$ for $1 \leq i \leq C$. Each next state $Z^{(i)}_{(s,a)}$ becomes a child of the root node. Then, it continues to sample starting from each $Z^{(i)}_{(s,a)}$, and stops when the tree reaches a depth H. By alternating between average and maximum operations on the Q functions of each node, it is possible to build an approximation of the optimal Q-function at the root. This procedure is detailed by Algorithm 2.2. The algorithm is called `SparseSampling` because it builds a tree with a finite number of nodes, although the

state space $\mathcal{S}$ may be infinite: the tree can be interpreted as a *sparse* version of the MDP, built through *sampling*.

Theorem 2.2 shows that `SparseSampling` builds an ε-approximation of the optimal Q-function at a fixed $s \in \mathcal{S}$ with probability at least $1-\delta$, *i.e.*, the algorithm is is (ε, δ)-correct. The number of calls it makes to the generative model is of order $\widetilde{\mathcal{O}}\left(\left(\varepsilon^{-1}\log\delta^{-1}\right)^{\mathcal{O}(\log(1/\varepsilon))}\right)$, which is non-polynomial in $1/\varepsilon$. The (ε,δ)-correctness is proven by recursively applying Hoeffding's inequality at each level h of the tree, which requires $\mathcal{O}\left(\varepsilon^{-1}\log\delta^{-1}\right)$ samples at each h, and a maximum depth $H = \mathcal{O}\left(\log(1/\varepsilon)\right)$. Hence, its total sample complexity is of order

$$\underbrace{\frac{1}{\varepsilon^2}\log\frac{1}{\delta} \times \cdots \times \frac{1}{\varepsilon^2}\log\frac{1}{\delta}}_{H=\mathcal{O}(\log(1/\varepsilon)) \text{ times}} = \left(\frac{1}{\varepsilon}\log\frac{1}{\delta}\right)^{\mathcal{O}\left(\log\left(\frac{1}{\varepsilon}\right)\right)},$$

which is formally proven below.

Theorem 2.2 (Sample complexity of `SparseSampling`)**.** *Assume that the number of actions is finite, that is, $A := |\mathcal{A}| < \infty$, and that $A \geq 2$. For $\varepsilon > 0$ and $\delta > 0$, let*

$$C := \left\lceil \frac{8\gamma^2 r_{\max}^2}{(1-\gamma)^4\varepsilon^2}\left(\log\frac{2}{\delta} + 2H\log\left(\frac{16AH\gamma^2 r_{\max}^2}{(1-\gamma)^4\varepsilon^2}\log\frac{2}{\delta}\right)\right)\right\rceil$$

and $H = 1 + \left\lceil \log_{1/\gamma}\left(\frac{2r_{\max}}{\varepsilon(1-\gamma)}\right)\right\rceil$. Then, for any fixed state $s \in \mathcal{S}$, the output of `SparseSampling` *at depth $h = 1$ satisfies*

$$\mathbf{P}\left[\forall a \in \mathcal{A}, \left|\widehat{Q}_1(s,a) - Q^*(s,a)\right| \leq \varepsilon\right] \geq 1-\delta.$$

Furthermore, let $n(\varepsilon,\delta)$ be the number of calls to the generative model made by `SparseSampling`*. Then,*

$$\begin{aligned} n(\varepsilon,\delta) \leq 2(AC)^{H-1} &= 2\left[\frac{c_1}{\varepsilon^2}\left(\log\frac{2}{\delta} + c_2\log\left(\frac{c_3}{\varepsilon}\right)\log\left(\frac{c_4}{\varepsilon}\log\left(\frac{c_5}{\varepsilon}\right)\log\frac{2}{\delta}\right)\right)\right]^{c_6\log\left(\frac{c_7}{\varepsilon}\right)} \\ &= \widetilde{\mathcal{O}}\left(\left(\frac{1}{\varepsilon}\log\frac{1}{\delta}\right)^{\mathcal{O}(\log(1/\varepsilon))}\right), \end{aligned}$$

where $c_1, c_2, c_3, c_4, c_5, c_6$, and c_7 are constants depending on $r_{\max}, \gamma$, and A.

Proof. We start with some useful definitions, then we recursively bound the error on the Q-function estimates at each depth h with high probability.

Definitions Let

$$\varepsilon' := \frac{\varepsilon(1-\gamma)}{2\gamma} \quad \text{and} \quad \delta' := 2\exp\left(-\frac{C\varepsilon'^2(1-\gamma)^2}{2r_{\max}^2}\right).$$

Let $(\varepsilon_h)_{h=1}^H$ and $(\delta_h)_{h=1}^H$ be two sequences such that ε_h and δ_h represent the accuracy and the confidence of the Q function estimation at depth h. We define

- $\varepsilon_H = r_{\max}/(1-\gamma)$ and $\varepsilon_h = \gamma(\varepsilon' + \varepsilon_{h+1})$ for $1 \leq h < H$;
- $\delta_H = 0$ and $\delta_h = A\delta' + AC\delta_{h+1}$ for $1 \leq h < H$.

We can verify by induction on h that

$$\delta_h = \delta' \sum_{i=1}^{H-h} A^j C^{j-1} \leq \delta'(AC)^{H-1}$$

if $A \geq 2$ and $C \geq 2$, and that

$$\varepsilon_h = \sum_{i=1}^{H-h} \gamma^i \varepsilon' + \frac{\gamma^{H-h} r_{\max}}{1-\gamma} \leq \frac{\gamma\varepsilon' + \gamma^{H-h} r_{\max}}{1-\gamma}.$$

Notice that $H = 1 + \left\lceil \log_{1/\gamma}\left(\frac{2r_{\max}}{\varepsilon(1-\gamma)}\right)\right\rceil$ and the definition of ε' imply $\varepsilon_1 \leq \varepsilon$.

High-probability events If $R_{s,a}^{(i)}, Z_{s,a}^{(i)} \leftarrow \texttt{GenerativeModel}(s,a)$, then $R_{s,a}^{(i)} = r(s,a) \leq r_{\max}$, by Assumption 1.1. For any state s, action a, and depth $h < H$, we define the event $\mathcal{G}(s,a,h)$ as

$$\mathcal{G}(s,a,h) := \left\{\left|\widehat{Q}_h(s,a) - Q^*(s,a)\right| \leq \varepsilon_h\right\} \bigcap \left\{\bigcap_{i=1}^{C} \mathcal{G}(Z_{s,a}^{(i)}, h+1)\right\}$$

where, for any state z,

$$\mathcal{G}(z,h) := \bigcap_{a \in \mathcal{A}} \mathcal{G}(z,a,h).$$

Let Ω be the whole sample space, and we define $\mathcal{G}(s,a,H) = \Omega$ for any (s,a).

We prove by induction on h that

$$\forall (s,h),\ \mathbf{P}\left[\mathcal{G}(s,h)\right] \geq 1 - \delta_h. \tag{2.1}$$

For $h = H$, we have $\varepsilon_H = r_{\max}/(1-\gamma)$, $\delta_H = 0$ and the claim in verified, since

$$\forall (s,a),\ |\widehat{Q}_H(s,a) - Q^*(s,a)| = |Q^*(s,a)| \leq r_{\max}/(1-\gamma)$$

with probability 1. Assume that (2.1) is true for $h+1$. Since the reward function is known,

$$\left|\widehat{Q}_h(s,a) - Q^*(s,a)\right| \leq \frac{\gamma}{C}\left|\sum_{i=1}^{C}\left(\widehat{V}_{h+1}(Z_{s,a}^{(i)}) - V^*(Z_{s,a}^{(i)})\right)\right| + \frac{\gamma}{C}\left|\sum_{i=1}^{C}\left(V^*(Z_{s,a}^{(i)}) - \mathbf{E}\left[V^*(Z_{s,a}^{(i)})\right]\right)\right|,$$

where we used the facts that $R_i = r(s,a)$ and $Q^*(s,a) = r(s,a) + \gamma \mathbf{E}\left[V^*(Z)\right]$ for $Z \sim p(\cdot|s,a)$, and the definition of $\widehat{Q}_h(s,a)$ in Line 11 of Algorithm 2.2.

On the event $\bigcap_{i=1}^{C} \mathcal{G}(Z_{s,a}^{(i)}, h+1)$, we have

$$\frac{\gamma}{C}\left|\sum_{i=1}^{C}\left(\widehat{V}_{h+1}(Z_{s,a}^{(i)}) - V^*(Z_{s,a}^{(i)})\right)\right| \leq \gamma\varepsilon_{h+1}.$$

Consequently,

$$\begin{aligned}
\mathbf{P}\left[\mathcal{G}(s,a,h)\right] &= \mathbf{P}\left[\bigcap_{i=1}^{C}\mathcal{G}(Z_{s,a}^{(i)}, h+1) \bigcap \left\{\left|\widehat{Q}_h(s,a) - Q^*(s,a)\right| \leq \varepsilon_h\right\}\right] \\
&\geq \mathbf{P}\left[\bigcap_{i=1}^{C}\mathcal{G}(Z_{s,a}^{(i)}, h+1) \bigcap \left\{\frac{\gamma}{C}\left|\sum_{i=1}^{C}\left(V^*(Z_{s,a}^{(i)}) - \mathbf{E}\left[V^*(Z_{s,a}^{(i)})\right]\right)\right| \leq \varepsilon_h - \gamma\varepsilon_{h+1}\right\}\right] \\
&\geq \mathbf{P}\left[\bigcap_{i=1}^{C}\mathcal{G}(Z_{s,a}^{(i)}, h+1) \bigcap \left\{\frac{1}{C}\left|\sum_{i=1}^{C}\left(V^*(Z_{s,a}^{(i)}) - \mathbf{E}\left[V^*(Z_{s,a}^{(i)})\right]\right)\right| < \varepsilon'\right\}\right].
\end{aligned}$$

By Hoeffding's inequality and the fact that $V^*(s) \in [-\frac{r_{\max}}{1-\gamma}, \frac{r_{\max}}{1-\gamma}]$ for any s, we have

$$\mathbf{P}\left[\frac{1}{C}\left|\sum_{i=1}^{C}\left(V^*(Z_{s,a}^{(i)}) - \mathbf{E}\left[V^*(Z_{s,a}^{(i)})\right]\right)\right| \geq \varepsilon'\right] \leq 2\exp\left(-\frac{C\varepsilon'^2(1-\gamma)^2}{2r_{\max}^2}\right) = \delta'.$$

Using the induction hypothesis that (2.1) holds for $h+1$, we obtain

$$\mathbf{P}\left[\mathcal{G}(s,h)^{\complement}\right] \leq \sum_{a\in\mathcal{A}}\mathbf{P}\left[\mathcal{G}(s,a,h)^{\complement}\right] \leq \sum_{a\in\mathcal{A}}\left(\delta' + \sum_{i=1}^{C}\mathbf{P}\left[\mathcal{G}(Z_{s,a}^{(i)}, h+1)\right]\right) \leq A\delta' + AC\delta_{h+1} = \delta_h.$$

which concludes the proof of (2.1) by induction.

At $h=1$, we obtain

$$\mathbf{P}\left[\forall a \in \mathcal{A}, \left|\widehat{Q}_1(s,a) - Q^*(s,a)\right| \leq \varepsilon\right] \geq \mathbf{P}\left[\mathcal{G}(s, h=1)\right] \geq 1 - \delta'(AC)^{H-1},$$

for any $s \in \mathcal{S}$.

Now, for a fixed $\delta > 0$, we need to find C such that

$$\delta'(AC)^{H-1} = 2\exp\left(-\frac{C\varepsilon'^2(1-\gamma)^2}{2r_{\max}^2}\right)(AC)^{H-1} \leq \delta.$$

By Lemma A.4, this condition is satisfied by a value C that is bounded as

$$\begin{aligned} C &\leq \frac{2r_{\max}^2}{(1-\gamma)^2\varepsilon'^2}\left(\log\frac{2}{\delta} + H\log\left(\frac{8Ar_{\max}^2}{(1-\gamma)^2\varepsilon'^2}\log\frac{2}{\delta} + \frac{4A^2H^2r_{\max}^4}{(1-\gamma)^4\varepsilon'^4}\right)\right) \\ &\leq \frac{2r_{\max}^2}{(1-\gamma)^2\varepsilon'^2}\left(\log\frac{2}{\delta} + 2H\log\left(\frac{4AHr_{\max}^2}{(1-\gamma)^2\varepsilon'^2}\log\frac{2}{\delta}\right)\right). \end{aligned}$$

Since $\varepsilon' := \frac{\varepsilon(1-\gamma)}{2\gamma}$, we take

$$C = \left\lceil \frac{8\gamma^2 r_{\max}^2}{(1-\gamma)^4\varepsilon^2}\left(\log\frac{2}{\delta} + 2H\log\left(\frac{16AH\gamma^2r_{\max}^2}{(1-\gamma)^4\varepsilon^2}\log\frac{2}{\delta}\right)\right)\right\rceil,$$

which gives us $\mathbf{P}\left[\forall a \in \mathcal{A}, \left|\widehat{Q}_1(s,a) - Q^*(s,a)\right| \leq \varepsilon\right] \geq 1-\delta$.

Finally, let $n_h(\varepsilon,\delta)$ be the number of calls made to the generative model by `SparseSampling` at depth h. We have $n_H(\varepsilon,\delta) = 0$ and $n_h(\varepsilon,\delta) = AC + ACn_{h+1}(\varepsilon,\delta)$ for $h < H$, which implies $n_h(\varepsilon,\delta) = \sum_{i=1}^{H-h}(AC)^i$. The total sample complexity is equal to $n_1(\varepsilon,\delta)$ and satisfies $n_1(\varepsilon,\delta) \leq 2(AC)^{H-1}$, which concludes the proof. □

Although `SparseSampling` has a sample complexity that is independent of the size of the state space $\mathcal{S}$, Theorem 2.2 only ensures that we are able to estimate the optimal Q-function at a *fixed* state s. The result below, adapted from Kearns et al. [KMN02], shows that it is possible to use `SparseSampling` as a subroutine to implement a near-optimal policy.

Lemma 2.3 (adapted from Lemma 5 by Kearns et al. [KMN02])**.** *Consider an algorithm that, at each state s, executes an action a_s by following the procedure below:*

- *Compute $\widehat{Q}_1(s,\cdot)$ such that $\mathbf{P}\left[\forall a \in \mathcal{A}, \left|\widehat{Q}_1(s,a) - Q^*(s,a)\right| \leq \varepsilon\right] \geq 1-\delta$;*
- *Choose the action $\widehat{a}_s \leftarrow \operatorname{argmax}_a \widehat{Q}_1(s,a)$.*

Since $\widehat{a}_s$ is a random variable, this algorithm implements a stochastic policy, that we denote by π.

Then, for any state s, we have

$$V^*(s) - V^\pi(s) \leq \frac{2\varepsilon}{1-\gamma} + \frac{2\delta r_{\max}}{(1-\gamma)^2}.$$

Proof. This result is a restatement of Lemma 5 by Kearns et al. [KMN02], which can be applied immediately by noticing that, with probability at least $1-\delta$,

$$V^*(s) - Q^*(s, \widehat{a}_s) \leq Q^*(s, \pi^*(s)) - \widehat{Q}_1(s, \pi^*(s)) + \widehat{Q}_1(s, \widehat{a}_s) - Q^*(s, \widehat{a}_s) \leq 2\varepsilon.$$

□

Finally, we restate below the theorem by Kearns et al. [KMN02] that provides a lower bound on the sample complexity of any algorithm that has access to a generative model that does not make assumptions on the size of the state space. It states that, for any algorithm that is able to implement an ε-optimal policy, at least $\Omega\left((1/\varepsilon)^{1/\log(1/\gamma)}\right)$ calls to the generative model are required.

Theorem 2.4 (Theorem 2 from Kearns et al. [KMN02]). *Consider an algorithm that is given access to a generative model for an MDP $\mathcal{M}$ and that implements a policy $\boldsymbol{\pi}$. If, for any input state $s \in \mathcal{S}$, $\boldsymbol{\pi}$ satisfies $|V^{\boldsymbol{\pi}}(s) - V^*(s)| \leq \varepsilon$, then there exists an MDP $\mathcal{M}$ on which this algorithm makes at least $\Omega\left((1/\varepsilon)^{1/\log(1/\gamma)}\right)$ calls to the generative model.*

2.3 SmoothCruiser: Planning in Regularized MDPs

In the previous section, we saw that the SparseSampling algorithm allows us to provide an (ε, δ)-correct approximation of the value function in a fixed state for any MDP with finite action set $\mathcal{A}$. However, its sample complexity is non-polynomial in $1/\varepsilon$, and there are no known algorithms with guaranteed polynomial complexity in the worst case. In this section, we propose an algorithm, called SmoothCruiser, that has a worst-case sample complexity of order $\mathcal{O}\left(1/\varepsilon^4\right)$ provided that the MDP is *regularized*.

2.3.1 Regularized MDPs

Consider an MDP $\mathcal{M}$ and let $\Delta(\mathcal{A})$ be the set of probability distributions on its action set $\mathcal{A}$. Let $\mathcal{R} : \Delta(\mathcal{A}) \to \mathbb{R}_+$ be a regularization functional and $\lambda > 0$ be a regularization factor. We assume that $\sup_{\mu \in \Delta(\mathcal{A})} \mathcal{R}(\mu) < \infty$. We say that $\mathcal{M}$ is regularized if its Bellman operator is defined as the operator $\mathcal{T}_{\gamma,\lambda}$ such that

$$\forall s \in \mathcal{S},\ [\mathcal{T}_{\gamma,\lambda}\overline{V}](s) = \max_{\pi(\cdot|s) \in \Delta(\mathcal{A})} \mathbf{E}\left[r(s,a) + \lambda \mathcal{R}(\pi(\cdot|s)) + \gamma \overline{V}(z)\right], \tag{2.2}$$

where the expectation is taken over $a \sim \pi(\cdot|s)$ and $z \sim p(\cdot|s,a)$.

Notice that, if $\lambda = 0$, $\mathcal{T}_{\gamma,\lambda}$ is equal to the Bellman operator $\mathcal{T}_\gamma$ for the discounted-reward criterion. The operator $\mathcal{T}_{\gamma,\lambda}$ is also a contraction, and we denote its fixed point by V. We refer to V as the

optimal regularized value function, and it satisfies [GSP19]

$$V(s) \geq \sup_{\pi \in \Pi_{\mathrm{M}}} \mathbf{E}_{\pi}\left[\sum_{t=1}^{\infty} \gamma^{t-1}\left[r(S_t, A_t) + \lambda \mathcal{R}(\pi(\cdot|S_t))\right] \middle| S_1 = s\right],$$

that is, it can be seen as the optimal value function in an MDP where a policy-dependent regularization term is added to the reward function.

We define $F : \mathbb{R}^A \to \mathbb{R}$ as

$$F(q) = \max_{\mu \in \Delta(\mathcal{A})} \mathbf{E}_{a \sim \mu}\left[q(a) + \lambda \mathcal{R}(\mu)\right], \tag{2.3}$$

such that the optimal regularized value function V satisfies

$$\forall s \in \mathcal{S}\,, V(s) = F(Q_s), \quad \text{where} \quad Q_s(a) = \mathbf{E}_{z \sim p(\cdot|s,a)}\left[r(s,a) + \gamma V(z)\right],$$

and we refer to Q_s as the optimal regularized Q-function.

Assumptions Without loss of generality, we assume in this section that the reward function satisfies $0 \leq r(s,a) \leq r_{\max} = 1$. By defining

$$M := F(\mathbf{0}) = \max_{\mu \in \Delta(\mathcal{A})} \lambda \mathcal{R}(\mu),$$

the optimal regularized value function is bounded by $(1+M)/(1-\gamma)$. Also, we make the following assumptions on the function F:

Assumption 2.5. *Consider the function $F : \mathbb{R}^A \to \mathbb{R}$ defined by (2.3). We assume that*

1. *F is differentiable;*
2. *(1-Lipschitz) $\forall q \in \mathbb{R}^A$, $0 < \|\nabla F(q)\|_1 \leq 1$;*
3. *(nonnegative gradient) $\forall q \in \mathbb{R}^A$, $\nabla F(q) \succeq 0$;*
4. *(L-smoothness) $\exists L \geq 0$ such that, for any $q, q' \in \mathbb{R}^A$,*

$$\left|F(q) - F(q') - (q - q')^{\intercal} \nabla F(q')\right| \leq L \left\|q - q'\right\|_2^2.$$

These assumptions are verified, for instance, if

$$\mathcal{R}(\mu) = -\sum_{a \in \mathcal{A}} \mu(a) \log \mu(a),$$

that is, the entropy of the probability distribution $\mu \in \Delta(\mathcal{A})$. In this case, we have $L = 1/\lambda$ and

$$F(q) = \lambda \log \sum_{i=1}^{A} \exp(q_i/\lambda),$$

which is a smooth approximation of the max function, defined as $\max(q) = \max_i q_i$. Indeed, we have $|\max(q) - \lambda \log \sum_{i=1}^{A} \exp(q_i/\lambda)| \leq \lambda \log A$, for any $q \in \mathbb{R}^A$.

Another example arises by setting $\mathcal{R}(\mu) = \sum_{a \in \mathcal{A}} \sqrt{\mu(a)}$, and the proof that it results in a function F satisfying Assumption 2.5 is given in [Gri+19, Appendix E].

2.3.2 Algorithm

We now describe our proposed algorithm, SmoothCruiser (Algorithm 2.3). Its building blocks are two procedures, sampleV (Algorithm 2.4) and estimateQ (Algorithm 2.5) that recursively call each other. The procedure sampleV returns a noisy estimate of $V(s)$ with a bias bounded by ε. The procedure estimateQ averages the outputs of several calls to sampleV to obtain an estimate $\widehat{Q}_s$ that is an approximation of Q_s with precision ε with high probability. Finally, SmoothCruiser calls estimateQ(s, ε) and outputs $\widehat{Q}_s$.

Algorithm 2.3: SmoothCruiser

input: $(s, \varepsilon, \delta') \in \mathcal{S} \times \mathbb{R}_+ \times \mathbb{R}_+$
$M \leftarrow F(\mathbf{0})$
$\kappa \leftarrow (1 - \sqrt{\gamma})/(AL)$
set δ', κ and M as global parameters
$\widehat{Q}_s \leftarrow$ estimateQ(s, ε)
return: $\widehat{Q}_s$

Algorithm 2.4: sampleV

input: $(s, \varepsilon) \in \mathcal{S} \times \mathbb{R}_+$
if $\varepsilon \geq (1 + M)/(1 - \gamma)$ **then**
    return: 0
else if $\varepsilon \geq \kappa$ **then**
    $\widehat{Q}_s \leftarrow$ estimateQ(s, ε)
    return: $F(\widehat{Q}_s)$
else if $\varepsilon < \kappa$ **then**
    $\widehat{Q}_s \leftarrow$ estimateQ$(s, \sqrt{\kappa\varepsilon})$
    $\widehat{A} \leftarrow$ action drawn from $\nabla F(\widehat{Q}_s)/\|\nabla F(\widehat{Q}_s)\|_1 \in \Delta(\mathcal{A})$
    $R, Z \leftarrow$ GenerativeModel$(s, \widehat{A})$
    $\widehat{V} \leftarrow$ sampleV$(Z, \varepsilon/\sqrt{\gamma})$
    return: $F(\widehat{Q}_s) - \widehat{Q}_s^{\intercal} \nabla F(\widehat{Q}_s) + (R + \gamma \widehat{V}) \|\nabla F(\widehat{Q}_s)\|_1$

Algorithm 2.5: `estimateQ`

input: $(s, \varepsilon) \in \mathcal{S} \times \mathbb{R}_+$
$N(\varepsilon) \leftarrow \left\lceil \frac{18(1+M)^2}{(1-\gamma)^4(1-\sqrt{\gamma})^2} \frac{\log(2A/\delta')}{\varepsilon^2} \right\rceil$
for $a \in \mathcal{A}$ **do**
  for $i \in \{1, \ldots, N(\varepsilon)\}$ **do**
    $(R, Z) \leftarrow \texttt{GenerativeModel}(s, a)$
    $\widehat{V} \leftarrow \texttt{sampleV}(Z, \varepsilon/\sqrt{\gamma})$
    $q_i \leftarrow R + \gamma \widehat{V}$
  $\widehat{Q}_s(a) \leftarrow \mathbf{mean}\,(q_1, \ldots, q_N)$
  # clip $\widehat{Q}_s(a)$ to $[0, (1+M)/(1-\gamma)]$
  $\widehat{Q}_s(a) \leftarrow \min\left(\max\left(\widehat{Q}_s(a), 0\right), (1+M)/(1-\gamma)\right)$
return: $\widehat{Q}_s \in \mathbb{R}^A$

The most important part of the algorithm is the procedure `sampleV`, that returns a low-bias estimate of the value function. Having the estimate of the value function, the procedure `estimateQ` averages the outputs of `sampleV` to obtain a good estimate of the Q function with high probability. The main idea of `sampleV` is to first compute an estimate of precision $\mathcal{O}(\sqrt{\varepsilon})$ of the value of each action $\{\widehat{Q}_s(a)\}_{a \in \mathcal{A}}$ to linearly approximate the function F around $\widehat{Q}_s$. The local approximation of F around $\widehat{Q}_s$ is subsequently used to estimate the value of s with a better precision, of order $\mathcal{O}(\varepsilon)$, which is possible due to the smoothness of F.

For a target accuracy ε at state s, `sampleV` distinguishes three cases, based on a reference threshold $\kappa := (1 - \sqrt{\gamma})/(AL)$, which is the maximum value of ε for which we can compute a good estimate of the value function using linear approximations of F.

- **First,** if $\varepsilon \geq (1+M)/(1-\gamma)$, then 0 is a valid output, since $V(s)$ is bounded by $(1 + M)/(1 - \gamma)$. This case furthermore ensures that our algorithm terminates, since the recursive calls are made with increasing values of ε.
- **Second,** if $\kappa \leq \varepsilon < (1+M)/(1-\gamma)$, we run F (`estimateQ`(s, ε)) in which for each action, both the generative model and `sampleV` are called $\mathcal{O}\left(1/\varepsilon^2\right)$ times in order to return $\widehat{V}(s)$ which is with high probability an ε-approximation of $V(s)$.
- **Finally,** if $\varepsilon < \kappa$, we take advantage of the smoothness of F to compute an ε-approximation of $V(s)$ in a more efficient way than calling `sampleV` (and the generative model) $\mathcal{O}\left(1/\varepsilon^2\right)$ times. We achieve it by calling `estimateQ` with a precision $\sqrt{\kappa\varepsilon}$ instead of ε, which requires $\mathcal{O}\left(1/\varepsilon\right)$ calls instead.

2.3.3 Sample Complexity of SmoothCruiser

In Theorem 2.6, we provide a bound on the sample complexity of SmoothCruiser. In Theorem 2.7, we provide a consistency result, stating that the output SmoothCruiser applied to a state $s \in \mathcal{S}$ is a good approximation of the optimal regularized value $V(s)$ with high probability.

Theorem 2.6. *Let $n(\varepsilon, \delta')$ be the number of calls to the generative model before* SmoothCruiser *terminates. For any state $s \in \mathcal{S}$ and $\varepsilon, \delta' > 0$,*

$$n(\varepsilon, \delta') \leq \frac{c_1}{\varepsilon^4} \log\left(\frac{c_2}{\delta'}\right) \left[c_3 \log\left(\frac{c_4}{\varepsilon}\right)\right]^{\log_2\left(c_5\left(\log\left(\frac{c_2}{\delta'}\right)\right)\right)} = \widetilde{\mathcal{O}}\left(\frac{1}{\varepsilon^4}\right),$$

where c_1, c_2, c_3, c_4, and c_5 are constants that depend only on A, L, and γ.

Theorem 2.7. *For any $s \in \mathcal{S}$, $\varepsilon > 0$, and $\delta > 0$, there exists a δ' that depends on ε and δ such that the output $\widehat{Q}_s$ of* SmoothCruiser*$(s, \varepsilon, \delta')$ satisfies*

$$\mathbf{P}\left[\forall a \in \mathcal{A},\ |\widehat{Q}_s(a) - Q_s(a)| \leq \varepsilon\right] \geq 1 - \delta$$

and such that $n(\varepsilon, \delta') = \mathcal{O}\left(1/\varepsilon^{4+c}\right)$ for any $c > 0$.

More precisely, in the proof of Theorem 2.7, we establish that

$$\mathbf{P}\left[\forall a \in \mathcal{A},\ |\widehat{Q}_s(a) - Q_s(a)| > \varepsilon\right] \leq \delta' n(\varepsilon, \delta').$$

Therefore, for any parameter δ' satisfying $\delta' n(\varepsilon, \delta') \leq \delta$, SmoothCruiser with parameters ε and δ' provides an approximation of Q_s which is (ε, δ) correct.

The proofs of Theorem 2.6 and Theorem 2.7 are given in Appendix A.1 and Appendix A.2, respectively. In the rest of this section, we explain the key ideas that allow us to exploit the smoothness of the Bellman operator to obtain a better sample complexity.

When $\varepsilon < \kappa$, the procedure `estimateQ` is called to obtain an estimate $\widehat{Q}_s$ such that

$$\|\widehat{Q}_s - Q_s\|_2 = \mathcal{O}\left(\sqrt{\varepsilon/L}\right).$$

The procedure `sampleV` then continues with computing a linear approximation of $F(Q_s)$ around $\widehat{Q}_s$. Using the L-smoothness of F, we guarantee the ε-approximation,

$$\left|F(Q_s) - \left\{F(\widehat{Q}_s) + (Q_s - \widehat{Q}_s)^\intercal \nabla F(\widehat{Q}_s)\right\}\right| \leq L\|\widehat{Q}_s - Q_s\|_2^2 = \mathcal{O}(\varepsilon).$$

We wish to output this linear approximation, but we need to handle the fact that the vector Q_s (the true Q-function at s) is unknown. Notice that the vector $\nabla F(\widehat{Q}_s)/\|\nabla F(\widehat{Q}_s)\|_1$ represents a probability distribution. The term $Q_s^\intercal \nabla F(\widehat{Q}_s)$ in the linear approximation of $F(Q_s)$ above can be expressed as

$$Q_s^\intercal \nabla F(\widehat{Q}_s) = \mathbf{E}\left[Q_s(\widehat{A})\|\nabla F(\widehat{Q}_s)\|_1 \middle| \widehat{Q}_s\right], \text{ with } \widehat{A} \sim \nabla F(\widehat{Q}_s)/\|\nabla F(\widehat{Q}_s)\|_1.$$

Therefore, we can build a low-bias estimate of $Q_s^\intercal \nabla F(\widehat{Q}_s)$ from estimating only $Q_s(\widehat{A})$:

- sample action $\widehat{A} \sim \nabla F(\widehat{Q}_s)/\|\nabla F(\widehat{Q}_s)\|_1$;
- sample a reward and a next state $R_{s,\widehat{A}}, Z_{s,\widehat{A}} \leftarrow \texttt{GenerativeModel}(s, \widehat{A})$;
- obtain an $\mathcal{O}(\varepsilon)$-approximation of $Q_s(\widehat{A})$: $\widetilde{Q}(\widehat{A}) = R_{s,\widehat{A}} + \gamma \texttt{sampleV}\left(Z_{s,\widehat{A}}, \varepsilon/\sqrt{\gamma}\right)$.

We show that $\widehat{V}(s)$ is, in expectation, an ε-approximation of the true value function $V(s)$. The benefit of such approach is that we can call `estimateQ` with a precision $\mathcal{O}(\sqrt{\varepsilon})$ instead of $\mathcal{O}(\varepsilon)$, which thanks to the smoothness of F, reduces the sample complexity. In particular, one call to $\texttt{sampleV}(s, \varepsilon)$ will make $\mathcal{O}(1/\varepsilon)$ recursive calls to $\texttt{sampleV}(s, \mathcal{O}(\sqrt{\varepsilon}))$, and the total number of calls to `sampleV` behaves as

$$\frac{1}{\varepsilon} \times \frac{1}{\varepsilon^{1/2}} \times \frac{1}{\varepsilon^{1/4}} \times \cdots \leq \frac{1}{\varepsilon^2}.$$

Therefore, the total number of calls to the generative model made by `sampleV` is $\mathcal{O}(1/\varepsilon^2)$, which implies that the total sample complexity is $\mathcal{O}(1/\varepsilon^4)$, since `SmoothCruiser` makes $\mathcal{O}(1/\varepsilon^2)$ calls to `sampleV`.

Impact of the smoothness L In Theorem 2.6 we did not make the dependence on L explicit to preserve simplicity. However, we can analyze the sample complexity in the two limits:

strong regularization $L \to 0$ and F is linear

no regularization $L \to \infty$ and F is not smooth

As $L \to 0$, the condition $\kappa = (1-\sqrt{\gamma})/(AL) \leq \varepsilon \leq (1+M)/(1-\gamma)$ will be met less and eventually the algorithm will sample $N = \mathcal{O}(1/\varepsilon^2)$ trajectories, which implies a sample complexity of order $\mathcal{O}(1/\varepsilon^2)$. On the other hand, as L goes to ∞, the condition $\varepsilon < \kappa$ will be met less and the algorithm eventually runs a sampling strategy similar to `SparseSampling`, which results in a sample complexity of order $\mathcal{O}\left((1/\varepsilon)^{\mathcal{O}(\log(1/\varepsilon))}\right)$, which is non-polynomial in $1/\varepsilon$. For a fixed L, `SmoothCruiser` can be seen as an interpolation of both cases, and results in a sample complexity of order $\mathcal{O}(1/\varepsilon^4)$.

Approximating the optimal value function Let $V^*(s)$ be the optimal value function without regularization. We can prove that $\sup_s |V(s) - V^*(s)| \leq F(\mathbf{0})/(1-\gamma) = \mathcal{O}(\lambda)$, where λ is the regularization factor. Thus, we can interpret the optimal regularized value function $V(s)$ as an approximate version of $V^*(s)$, which we can estimate faster.

Comparison to lower bound For non-regularized problems, Theorem 2.4 provides a sample complexity lower bound of $\Omega\left((1/\varepsilon)^{1/\log(1/\gamma)}\right)$, which is polynomial in $1/\varepsilon$, but its exponent grows as γ approaches 1. For regularized problems, Theorem 2.2 shows that the sample complexity is polynomial with an exponent that is *independent* of γ. Hence, when γ is close to 1, regularization gives us a better asymptotic behavior with respect to $1/\varepsilon$ than the lower bound for the non-regularized case, although we are not estimating the same value. To the best of our knowledge, there are still no lower bounds proved for planning in regularized MDPs.

2.4 Discussion and Bibliographical Remarks

In this chapter, we studied three algorithms that allow us to approximate the optimal value function of an MDP by using a generative model: `MBQVI`, `SparseSampling` and `SmoothCruiser`. We started with `MBQVI`, which applies only to finite MDPs, and saw that it requires $\widetilde{\mathcal{O}}\left(SA/\varepsilon^2\right)$ calls to the generative model to provide an ε-approximation of Q^*, where S and A are the number of states and actions in the MDP. Intuitively, this dependence on S comes from the fact that `MBQVI` outputs a Q-function $\widehat{Q}$ that approximates Q^* *for all states* $s \in \mathcal{S}$, *i.e.*, $\max_{s,a} |\widehat{Q}(s,a) - Q^*(s,a)| \leq \varepsilon$. By requiring an ε-approximation *only for a fixed state* s, `SparseSampling` has a sample complexity $\mathcal{O}\left((1/\varepsilon)^{\mathcal{O}(\log(1/\varepsilon))}\right)$, which does not depend on the size of the state space, but has a much worse dependence on $1/\varepsilon$, when compared to `MBQVI`. Although `SparseSampling` focuses on estimating Q^* only for a fixed s, we saw in Lemma 2.3 that it can be used to implement a near-optimal policy that runs `SparseSampling` from scratch every time we need to choose an action. Finally, we proposed `SmoothCruiser`, that focuses on approximating the optimal *regularized* value function at a fixed state s, and has a sample complexity of order $\mathcal{O}\left(1/\varepsilon^4\right)$, that is polynomial in $1/\varepsilon$, contrary to `SparseSampling`. As the regularization goes to zero, its sample complexity approaches that of `SparseSampling`, so that we can interpret `SmoothCruiser` as an acceleration of `SparseSampling` that applies when the MDP is regularized. A disadvantage of `SparseSampling` and `SmoothCruiser` is that both algorithms make a huge amount of recursive calls and we have no guarantees on their output if they are stopped before termination, which makes their implementation impractical in most situations. However, they allow us to obtain sample-complexity bounds that hold for MDPs with *arbitrary state spaces*, which is valuable from a theoretical point of view. In particular, regularization has been employed in several commonly used algorithms for RL, and `SmoothCruiser` shows how we can exploit the smoothness of the regularized Bellman operator to improve the sample complexity of planning.

Regularization in RL Regularization has been shown to be useful in several RL algorithms. In the context of policy gradient algorithms, a common example is the `A3C` algorithm [Mni+16] that penalizes policies with low entropy to improve exploration, and the work of Neu et al. [NJG17] presents a theoretical framework for entropy regularization using the joint state-action distribution. Formulations with entropy-augmented rewards have been used to learn multi-modal policies to improve exploration and robustness [Haa+17; Haa+18] and can also be related to policy gradient methods [SCA17]. Furthermore, Geist et al. [GSP19] propose a theory of regularized MDPs, although they do not study the sample complexity of algorithms in this setting.

Planning from a fixed state s Walsh et al. [WGL10] provide an adaptive action-selection scheme for `SparseSampling`, but its sample complexity is still non-polynomial in $1/\varepsilon$. The `UCT` algorithm [KS06b], used for planning in MDPs and games, selects actions based on optimistic estimates of their values and has good empirical performance in several applications. However, the sample complexity of `UCT` can be worse than exponential in $1/\varepsilon$ for some environments, which is mainly due to exploration issues [CM07]. Algorithms with sample complexities of order $\mathcal{O}\left(1/\varepsilon^d\right)$, where d is a problem-dependent quantity, have been proposed for deterministic dynamics [HM08], and in an open-loop[1] setting [BM10; LM19; Bar+19], for bounded number of next states and a full MDP model is known [BM12], for bounded number of next states [SKM14], and for general MDPs [GVM16]. In general, when the state space is infinite and the transitions are stochastic, the problem-dependent quantity d can make the sample complexity guarantees non-polynomial. For a related setting, when rewards are only obtained in the leaves of a fixed tree, Kaufmann and Koolen [KK17] and Huang et al. [Hua+17] present algorithms to identify the optimal action in a game based on best-arm identification tools. In the *finite-horizon* setting with bounded number of next states, Feldman and Domshlak [FD14] and Jonsson et al. [Jon+20] provide planning algorithms (`BRUE` and `MDP-GapE`, respectively) that recommend an action $\widehat{a}$ for any fixed state s, whose sample complexities scale with the sub-optimality gaps $\Delta_a := V^*(s) - Q^*(s,a)$. Whereas Feldman and Domshlak [FD14] provide a bound for `BRUE` scaling with the inverse of the minimum gap $\Delta_{\min} = \min_{a \neq a^*} \Delta_a$, where $a^* = \operatorname{argmax}_a Q^*(s,a)$, the `MDP-GapE` algorithm by Jonsson et al. [Jon+20] has a sample complexity scaling with $\sum_a 1/\max(\Delta_a, \Delta_{\min}, \varepsilon)^2$ and has a better dependence with respect to the planning horizon H.[2]

Further reading Agarwal et al. [AKY20] provide a novel analysis of model-based planning with a generative model for finite MDPs, and show that `MBQVI` is also minimax optimal for obtaining an ε-optimal *policy*, whereas the analysis we provided in this chapter focuses on

[1] This means that the policy is seen as a function of time, not the states. The open-loop setting is particularly adapted to environments with deterministic transitions.

[2] Although the sample complexities of `BRUE` and `MDP-GapE` are polynomial with respect to $1/\Delta_{\min}$ or $1/\varepsilon$, they are exponential with respect to H. In the discounted-reward setting, an effective horizon H is defined as a function of $\log(1/\varepsilon)$, which, for instance, leads to the non-polynomial bound for `SparseSampling`.

obtaining an ε-optimal value function. Tarbouriech et al. [Tar+21b] extend `MBQVI` to stochastic shortest path problems. Du et al. [Du+20], Lattimore et al. [LSW20], and Weisz et al. [WAS21] analyze the sample complexity with a generative model when d-dimensional linear function approximation is used to represent the Q-functions, and study in which cases it is possible to obtain sample complexities that are polynomial in the quantities of interest (H, d, and $1/\varepsilon$), and in which cases the sample complexities are exponential.

Chapter 3

Online Interaction with Finite MDPs

In this chapter, we consider the finite-horizon criterion, and we assume that we have access to an online model of a finite MDP. We present three of the most common criteria used to evaluate RL algorithms in the online setting: the regret, the sample complexity of exploration, and the sample complexity for best-policy identification. Using a unified proof technique, we prove worst-case lower bounds for each of those criteria. Then, we present a simplified analysis of the `UCBVI` algorithm introduced by Azar et al. [AOM17], which is near-optimal for regret minimization in finite MDPs, and is generalized to continuous MDPs in Chapter 4.

The lower-bound analyses presented in this chapter are published in the paper [Dom+21b].

Contents

3.1 Performance Criteria 32
3.2 Lower Bounds: Key Ideas & Hard MDP Instances 34
3.3 Lower Bound on the Regret 38
3.4 Lower Bound on the Sample Complexity 41
3.5 Lower Bounds: Extensions 44
3.6 Upper Bound on the Regret of `UCBVI` 47
3.7 Discussion and Bibliographical Remarks 53

3.1 Performance Criteria

Consider the finite-horizon criterion and its probabilistic model introduced in Section 1.2, and assume that the agent has access to an online model of an MDP $\mathcal{M}$. In this setting, the agent interacts with the model in episodes of length H: in each stage $h \in \{1, \dots, H\}$ of an episode $t \in \mathbb{N}^*$, it is in a state $S_h^t \in \mathcal{S}$, it takes an action $A_h^t \in \mathcal{A}$ then observes the next state S_{h+1}^t sampled according to the transition kernel $p_h(\cdot|S_h^t, A_h^t)$, and receives a reward $r_h(S_h^t, A_h^t)$. The quality of an algorithm in the online setting can be measured with different performance metrics. We introduce below three of the main criteria studied in the literature.

Notation For any $n \in \mathbb{N}^*$, we define $[n] := \{1, \dots, n\}$. We recall some of the definitions introduced in Section 1.2.1. We denote by $B_H^t := (S_h^{t'}, A_h^{t'})_{h \in [H], 1 \le t' \le t}$, the transitions collected by the algorithm up to episode t, and by $\mathcal{F}_H^t = \boldsymbol{\sigma}(B_H^t)$ the σ-algebra generated by B_H^t. An RL algorithm is defined as a history-dependent policy $\boldsymbol{\pi}$ used to interact with the environment, and it might be equipped with a stopping time τ with respect to the filtration $(\mathcal{F}_H^t)_{t \in \mathbb{N}}$ and a policy recommendation $\widehat{\pi}_\tau$, such that the algorithm can stop at the end of episode τ and output $\widehat{\pi}_\tau$. We denote by $\boldsymbol{\pi}_H^{t-1}$ the restriction of $\boldsymbol{\pi}$ to the subset of histories starting with B_H^{t-1}. The number of states and actions in the MDP are denoted by S and A, respectively.

Regret The regret of an algorithm (Definition 3.1) after T episodes is defined as the sum over all episodes of the difference between the expected total reward gathered by an optimal policy and that of the algorithm, starting from a state that depends on the episode t. It quantifies the performance of an agent *during the learning period*, and how well it balances *exploration and exploitation*. That is, in order to minimize the regret, the agent must explore to learn the dynamics of the MDP, but the policy that it executes in each episode t must be close to the optimal policy as often as possible.

Definition 3.1 (Regret). *The regret of an algorithm $\boldsymbol{\pi}$ after T episodes is defined as*

$$\mathcal{R}_T := \sum_{t=1}^{T} \left(V_1^*(S_1^t) - V_1^{\boldsymbol{\pi}_H^{t-1}}(S_1^t) \right),$$

and its expected regret is $\overline{\mathcal{R}}_T := \mathbf{E}_{\boldsymbol{\pi}, \mathcal{M}}[\mathcal{R}_T]$. When not implied by the context, we explicit the dependence of $\mathcal{R}_T$ and $\overline{\mathcal{R}}_T$ on $\boldsymbol{\pi}$ and $\mathcal{M}$ by denoting those quantities by $\mathcal{R}_T(\boldsymbol{\pi}, \mathcal{M})$ and $\overline{\mathcal{R}}_T(\boldsymbol{\pi}, \mathcal{M})$.

Best-Policy Identification (BPI) The sample complexity for BPI (Definition 3.2) is defined in a Probably Approximately Correct (PAC) framework, and measures how many episodes τ are required for the agent to return a policy $\widehat{\pi}_\tau$ that is near-optimal with high probability. It

quantifies the performance *after the learning period*, hence it is a *pure-exploration objective*: the agent does not need to exploit during learning, since it is only evaluated at the end.

Definition 3.2 (PAC for best-policy identification)**.** *An algorithm $(\boldsymbol{\pi}, \tau, \widehat{\pi}_\tau)$ is (ε, δ)-PAC for best-policy identification in an MDP $\mathcal{M}$ if the policy $\widehat{\pi}_\tau$ returned after τ episodes satisfies*

$$\mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}\left[\mathbf{E}_{s\sim\mu}\left[V_1^*(s) - V_1^{\widehat{\pi}_\tau}(s)\right] \geq \varepsilon\right] \leq \delta.$$

where μ is a fixed distribution over the set of states $\mathcal{S}$. The sample complexity is defined as the number of episodes τ required for stopping.

PAC-MDP The *sample complexity of exploration* quantifies the number of episodes $\mathcal{N}^{\mathrm{PAC}}$ in which the agent executes a policy that is not ε-optimal. In this framework, an algorithm is said to be PAC for MDPs, PAC for exploration, or PAC-MDP [Kak03], if $\mathcal{N}^{\mathrm{PAC}}$ is bounded by a polynomial function with high probability (Definition 3.3). Like the regret, the PAC-MDP criterion measures the performance of the agent during learning, but the agent is penalized only by the *number of mistakes* it makes, and not by the *intensity* of those mistakes.

Definition 3.3 (PAC for exploration)**.** *An algorithm $\boldsymbol{\pi}$ is (ε, δ)-PAC for exploration in an MDP $\mathcal{M}$ (or PAC-MDP) if there exists a polynomial function $F_{\mathrm{PAC}}(H, 1/\varepsilon, \log(1/\delta))$, which may depend on the parameters of the MDP, such that its sample complexity*

$$\mathcal{N}^{\mathrm{PAC}} := \sum_{t=1}^{\infty} \mathbb{1}\left\{V_1^*(S_1^t) - V_1^{\boldsymbol{\pi}_H^{t-1}}(S_1^t) > \varepsilon\right\}$$

satisfies $\mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}\left[\mathcal{N}^{\mathrm{PAC}} > F_{\mathrm{PAC}}(H, 1/\varepsilon, \log(1/\delta))\right] \leq \delta.$

In the next sections, we prove worst-case lower bounds for the regret, the sample complexity of BPI, and for the sample complexity of PAC-MDP. These bounds state that, for any algorithm, there exists an MDP such that its regret is greater than a function of (S, A, H, T) and its sample complexity is greater than a function of $(S, A, H, \varepsilon, \delta)$. Lower bounds measure how "difficult" it is to optimize each criterion and allow us to assess the quality of the upper bounds provided for an algorithm.

Minimax optimality Worst-case lower bounds are also called *minimax lower bounds*, since they consider the best possible algorithm (*i.e.*, the algorithm with the *minimum* regret or sample complexity) in the worst possible environment for that algorithm (*i.e.*, with the *maximum* regret or sample complexity). An algorithm is said to be *minimax optimal* if its regret or sample complexity is equal to the lower bound, up to constant or logarithmic factors.

As discussed in Section 3.7, minimax lower bounds for RL have been analyzed in several previous works, and our main contributions are: (i) we provide unified proofs for lower bounds in the three criteria that we consider, in the sense that we use a single class of hard MDPs and the same information-theoretic tools for all our proofs; and (ii) our analyses are detailed, self-contained, and also consider *time-inhomogeneous* MDPs, that is, when the transitions p_h and rewards r_h may depend on the stage $h \in [H]$ within an episode.

3.2 Lower Bounds: Key Ideas & Hard MDP Instances

In this section, we present the two main ingredients for the proof of the minimax lower bounds. These bounds consider a class $\mathcal{C}$ of hard MDPs instances (on which the optimal policy is difficult to identify), that are typically close to each other, but for which the behavior of an algorithm is expected to be different (because they do not share the same optimal policy). The class $\mathcal{C}$ used to derive all the lower bounds is presented in Section 3.2.1. Then, lower bound proofs use a change of distribution between two well-chosen MDPs in $\mathcal{C}$ in order to obtain inequalities on the expected number of visits of certain state-action pairs in one of them. The information-theoretic tools that we use for these changes of distributions are gathered in Section 3.2.2.

3.2.1 Hard MDP Instances

From a high-level perspective, the family of MDPs that we use for our proofs behave like multi-armed bandits with $\Theta(HSA)$ arms. To gain some intuition about the construction, assume that $S = 4$ and consider the MDP in Figure 3.1. The agent starts in a *waiting state* s_{w} where it can take an action a_{w} to stay in s_{w} up to a stage $\overline{H} < H$, after which it has to leave s_{w}. From s_{w}, the agent can only transition to s_1, from which it can reach two absorbing states, a "good" state s_g and a "bad" state s_b. The state s_g is the only state where the agent can obtain a reward, which starts to be 1 at stage $\overline{H} + 2$. There is a single action a^* in state s_1 that increases by ε' the probability of arriving to the good state, and this action must be taken at a specific stage h^*. The intuition is that, in order to maximize the rewards, the agent must choose the right moment $h \in \left\{1, \dots, \overline{H}\right\}$ to leave s_{w}, and then choose the good action $a^* \in \{1, \dots, A\}$ in s_1. This results in a total of $\overline{H}A$ possible choices, or "arms", and the maximal reward is $\Theta(\overline{H})$. By analogy with the existing minimax regret bound for multi-armed bandits [Aue+02; LS20], the regret lower bound should be $\Omega(H\sqrt{HAT})$, by taking $\overline{H} = \Theta(H)$.

Inspired by the tree construction of Lattimore and Szepesvári [LS20] for the lower bound in the average-reward setting, we now generalize these MDPs to $S > 4$. Consider a family of MDPs described as follows and illustrated in Figure 3.2. First, we require the assumption below, which we relax in Section 3.5.1.

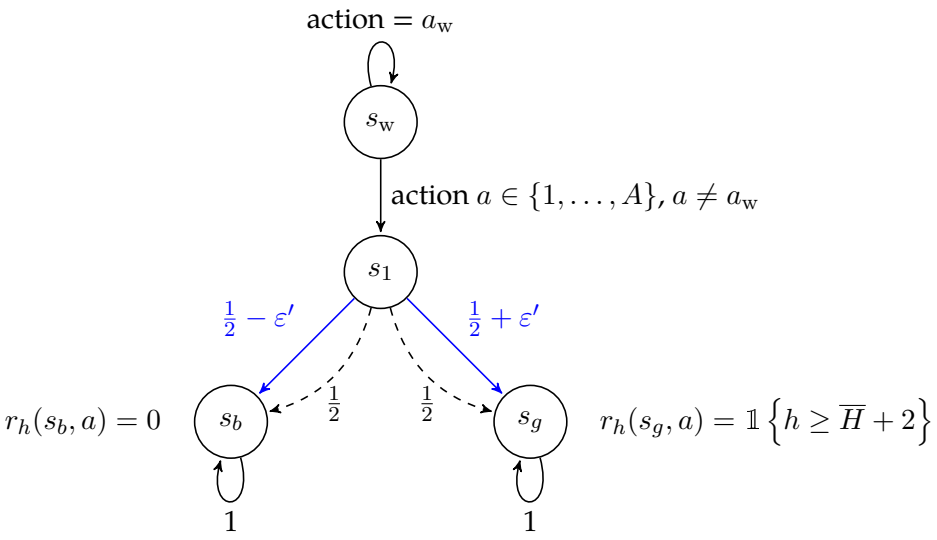


Figure 3.1 – Illustration of the class of hard MDPs for $S = 4$.

Assumption 3.4. *The number of states and actions satisfy $S \geq 6$, $A \geq 2$, and there exists an integer d such that $S = 3 + (A^d - 1)/(A - 1)$, which implies $d = \Theta(\log_A S)$. We further assume that $H \geq 3d$.*

As in the previous case, there are three special states: a "waiting" state s_{w} where the agent starts and can choose to stay up to a stage $\overline{H}$, a "good" state s_g that is absorbing and is the only state where the agent obtains rewards, and a "bad" state s_b that is absorbing and gives no reward. The other $S - 3$ states are arranged in a full A-ary tree of depth $d - 1$, which can be done since we assume there exists an integer d such that $S - 3 = \sum_{i=0}^{d-1} A^i$. The root of the tree is denoted by s_{root}, which can only be reached from s_{w}, and the states s_g and s_b can only be reached from the leaves of the tree.

Let $\overline{H} \leq H - d$ be an integer that will be a parameter of the class of MDPs. Letting $\mathcal{L} = \{s_1, s_2, \ldots, s_L\}$ be the set of L leaves of the tree, we define for each

$$(h^*, \ell^*, a^*) \in \left\{1 + d, \ldots, \overline{H} + d\right\} \times \mathcal{L} \times \mathcal{A},$$

an MDP $\mathcal{M}_{(h^*, \ell^*, a^*)}$ as follows. For any state in the tree, the transitions are deterministic: the a-th action in a node leads to the a-th child of that node. The transitions from s_{w} are given by

$$p_h(s_{\mathrm{w}} | s_{\mathrm{w}}, a) := \mathbb{1}\left\{a = a_{\mathrm{w}}, h \leq \overline{H}\right\} \quad \text{and} \quad p_h(s_{\mathrm{root}} | s_{\mathrm{w}}, a) := 1 - p_h(s_{\mathrm{w}} | s_{\mathrm{w}}, a).$$

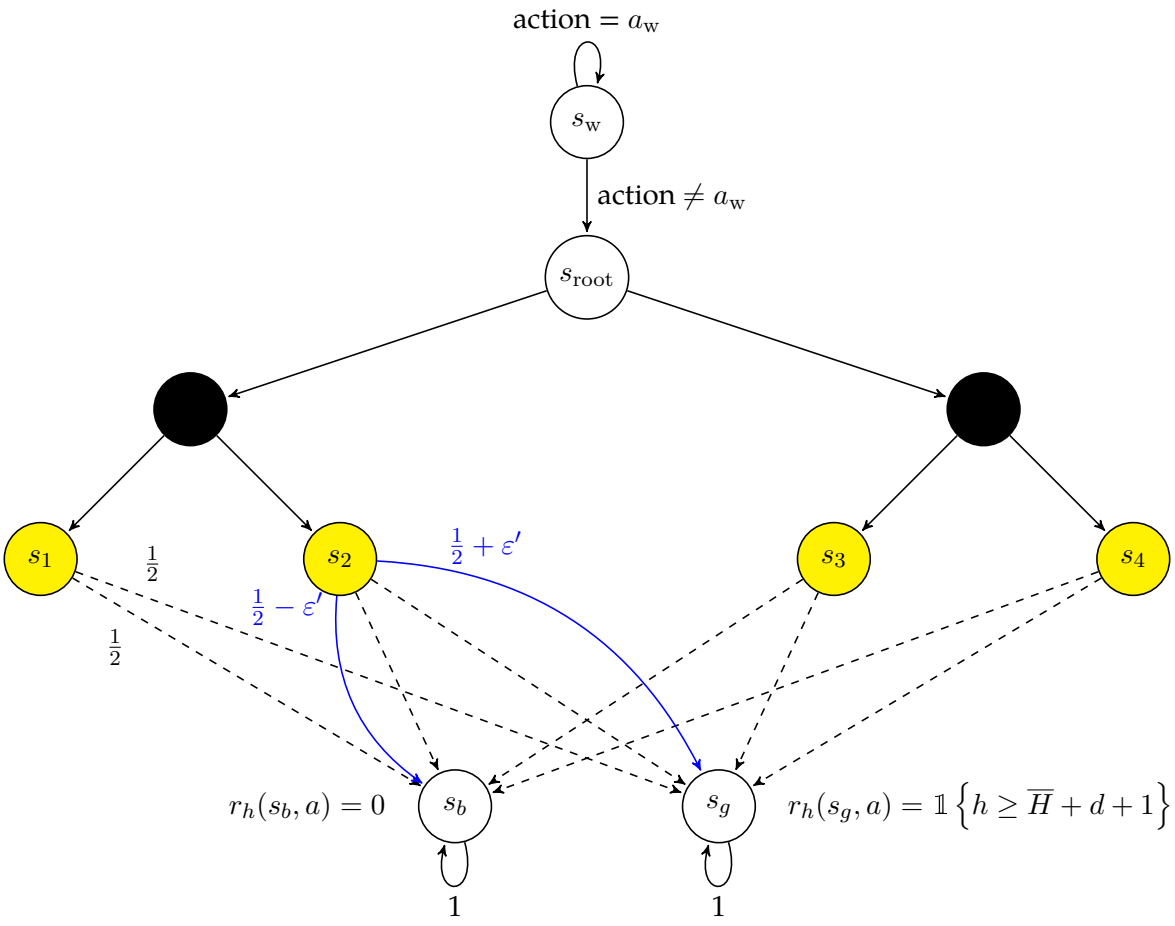


Figure 3.2 – Illustration of the class of hard MDPs used in the proofs of Theorems 3.8 and 3.9.

That is, there is an action a_{w} that allows the agent to stay at s_{w} up to a stage $\overline{H}$. After stage $\overline{H}$, the agent has to traverse the tree down to the leaves. The transitions from any leaf $s_i \in \mathcal{L}$ are given by

$$p_h(s_g|s_i, a) := \frac{1}{2} + \Delta_{(h^*,\ell^*,a^*)}(h, s_i, a) \quad \text{and} \quad p_h(s_b|s_i, a) := \frac{1}{2} - \Delta_{(h^*,\ell^*,a^*)}(h, s_i, a), \tag{3.1}$$

where $\Delta_{(h^*,\ell^*,a^*)}(h, s_i, a) := \mathbb{1}\{(h, s_i, a) = (h^*, s_{\ell^*}, a^*)\} \cdot \varepsilon'$, for some $\varepsilon' \in [0, 1/2]$ that is the second parameter of the class. This means that there is a single leaf ℓ^* where the agent can choose an action a^* at stage h^* that increases the probability of arriving to the good state s_g. Finally, the states s_g and s_b are absorbing, that is, for any action a, we have $p_h(s_b|s_b, a) := p_h(s_g|s_g, a) := 1$. The reward function depends only on the state and is defined as

$$\forall a \in \mathcal{A}, \quad r_h(s, a) := \mathbb{1}\left\{s = s_g, h \geq \overline{H} + d + 1\right\}$$

so that the agent does not miss any reward if it chooses to stay at s_{w} until stage $\overline{H}$.

We further define a reference MDP $\mathcal{M}_0$ which is an MDP of the above type but for which $\Delta_0(h, s_i, a) := 0$ for all (h, s_i, a). For every ε' and $\overline{H}$, we define the class $\mathcal{C}_{\overline{H},\varepsilon'}$ to be the set

$$\mathcal{C}_{\overline{H},\varepsilon'} := \{\mathcal{M}_0\} \bigcup \left\{\mathcal{M}_{(h^*,\ell^*,a^*)}\right\}_{(h^*,\ell^*,a^*)\in\left\{1+d,\dots,\overline{H}+d\right\}\times\mathcal{L}\times\mathcal{A}} .$$

3.2.2 Change of Distribution Tools

Recall from Section 1.2.1 that we denote by $\mathbf{P}_{\mathcal{M}}^{B_H^t}$ the pushforward measure of B_H^t, the history of states and actions up to episode t, under $\mathbf{P}_{\mathcal{M}}$.

Definition 3.5. *The Kullback-Leibler divergence between two distributions $\mathbf{P}_1$ and $\mathbf{P}_2$ on a measurable space $(\Omega, \mathcal{G})$ is defined as*

$$\mathrm{KL}(\mathbf{P}_1, \mathbf{P}_2) := \int_\Omega \log\left(\frac{\mathrm{d}\mathbf{P}_1}{\mathrm{d}\mathbf{P}_2}(\omega)\right) \mathrm{d}\mathbf{P}_1(\omega),$$

if $\mathbf{P}_1 \ll \mathbf{P}_2$ and $+\infty$ otherwise. For Bernoulli distributions, we define $\forall (p, q) \in [0, 1]^2$,

$$\mathrm{kl}(p, q) := \mathrm{KL}\left(\mathcal{B}(p), \mathcal{B}(q)\right) = p \log\left(\frac{p}{q}\right) + (1 - p) \log\left(\frac{1 - p}{1 - q}\right),$$

where $\mathcal{B}(p)$ denotes the Bernoulli distribution of parameter p.

Lemma 3.6 (proof in Appendix B.1)**.** *Let $\mathcal{M}$ and $\mathcal{M}'$ be two MDPs that are identical except for their transition probabilities, denoted by p_h and p'_h, respectively. Assume that we have $\forall (s, a)$, $p_h(\cdot|s, a) \ll p'_h(\cdot|s, a)$. Then, for any stopping time τ with respect to $(\mathcal{F}_H^t)_{t\geq 1}$ that satisfies $\mathbf{P}_{\mathcal{M}}[\tau < \infty] = 1$,*

$$\mathrm{KL}\left(\mathbf{P}_{\mathcal{M}}^{B_H^\tau}, \mathbf{P}_{\mathcal{M}'}^{B_H^\tau}\right) = \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \sum_{h\in[H-1]} \mathbf{E}_{\mathcal{M}}\left[N_{h,s,a}^\tau\right] \mathrm{KL}\left(p_h(\cdot|s, a), p'_h(\cdot|s, a)\right), \tag{3.2}$$

where $N_{h,s,a}^\tau := \sum_{t=1}^\tau \mathbb{1}\left\{(S_h^t, A_h^t) = (s, a)\right\}$ and B_H^τ is the random vector representing the history of states and actions up to episode τ.

Lemma 3.7 (Lemma 1 by Garivier et al. [GMS19])**.** *Consider a measurable space $(\Omega, \mathcal{F})$ equipped with two distributions $\mathbf{P}_1$ and $\mathbf{P}_2$. For any $\mathcal{F}$-measurable function $Z : \Omega \to [0, 1]$, we have*

$$\mathrm{KL}(\mathbf{P}_1, \mathbf{P}_2) \geq \mathrm{kl}(\mathbf{E}_1[Z], \mathbf{E}_2[Z]),$$

where $\mathbf{E}_1$ and $\mathbf{E}_2$ are the expectations under $\mathbf{P}_1$ and $\mathbf{P}_2$ respectively.

3.3 Lower Bound on the Regret

Using change of distributions between MDPs in a class $\mathcal{C}_{\overline{H},\varepsilon}$, we prove the following result, which gives a minimax lower bound on the expected regret of any algorithm.

Theorem 3.8. *Under Assumption 3.4, for any algorithm $\boldsymbol{\pi}$, there exists an MDP whose transitions depend on the stage h, such that, for $T \geq HSA$,*

$$\overline{\mathcal{R}}_T \geq \frac{1}{48\sqrt{6}}\sqrt{H^3SAT}\,.$$

Proof. Consider the class of MDPs $\mathcal{C}_{\overline{H},\varepsilon}$ introduced in Section 3.2.1, with $\overline{H}$ and ε to be chosen later. This class contains a reference MDP $\mathcal{M}_0$ and MDPs of the form $\mathcal{M}_{(h^*,\ell^*,a^*)}$ parameterized by

$$(h^*,\ell^*,a^*) \in \left\{1+d,\dots,\overline{H}+d\right\} \times \mathcal{L} \times \mathcal{A}$$

in which

$$\Delta_{(h^*,\ell^*,a^*)}(h,s_i,a) := \mathbb{1}\left\{(h,s_i,a) = (h^*,s_{\ell^*},a^*)\right\}\varepsilon.$$

As already mentioned, this family of MDPs behave like bandits, hence our proof follows the one for minimax lower bound in bandits (see, e.g., [BCB12]).

Regret of $\boldsymbol{\pi}$ in $\mathcal{M}_{(h^*,\ell^*,a^*)}$. The mean reward gathered by $\boldsymbol{\pi}$ in $\mathcal{M}_{(h^*,\ell^*,a^*)}$ is given by

$$\begin{aligned}\mathbf{E}_{(h^*,\ell^*,a^*)}\left[\sum_{t=1}^{T}\sum_{h=1}^{H} r_h(S_h^t,A_h^t)\right] &= \sum_{t=1}^{T}\mathbf{E}_{(h^*,\ell^*,a^*)}\left[\sum_{h=\overline{H}+d+1}^{H}\mathbb{1}\left\{S_h^t = s_g\right\}\right]\\ &= (H-\overline{H}-d)\sum_{t=1}^{T}\mathbf{P}_{(h^*,\ell^*,a^*)}\left[S_{\overline{H}+d+1}^t = s_g\right].\end{aligned}$$

For any $h \in \left\{1+d,\dots,\overline{H}+d\right\}$,

$$\begin{aligned}&\mathbf{P}_{(h^*,\ell^*,a^*)}\left[S_{h+1}^t = s_g\right]\\ &= \mathbf{P}_{(h^*,\ell^*,a^*)}\left[S_h^t = s_g\right] + \frac{1}{2}\mathbf{P}_{(h^*,\ell^*,a^*)}\left[S_h^t \in \mathcal{L}\right]\\ &+ \mathbb{1}\left\{h = h^*\right\}\mathbf{P}_{(h^*,\ell^*,a^*)}\left[S_h^t = s_{\ell^*}, A_h^t = a^*\right]\varepsilon. \qquad (3.3)\end{aligned}$$

Indeed, if $S_{h+1}^t = s_g$, we have either $S_h^t = s_g$ or $S_{h+1}^t \in \mathcal{L}$. In the latter case, the agent has $1/2$ probability of arriving at s_g, plus ε if the stage is h^*, the leaf is s_{ℓ^*} and the action is a^*.

Using the facts that $\mathbf{P}_{(h^*,\ell^*,a^*)}\left[S^t_{1+d} = s_g\right] = 0$ because the agent needs first to traverse the tree and $\sum_{h=1+d}^{\overline{H}+d} \mathbf{P}_{(h^*,\ell^*,a^*)}\left[S^t_h \in \mathcal{L}\right] = 1$ because the agent traverses the tree only once in one episode, we obtain from (3.3) that

$$\begin{aligned}
&\mathbf{P}_{(h^*,\ell^*,a^*)}\left[S^t_{\overline{H}+d+1} = s_g\right] \\
&= \sum_{h=1+d}^{\overline{H}+d} \frac{1}{2}\mathbf{P}_{(h^*,\ell^*,a^*)}\left[S^t_h \in \mathcal{L}\right] + \mathbb{1}\left\{h = h^*\right\}\mathbf{P}_{(h^*,\ell^*,a^*)}\left[S^t_h = s_{\ell^*}, A^t_h = a^*\right]\varepsilon \\
&= \frac{1}{2} + \varepsilon\mathbf{P}_{(h^*,\ell^*,a^*)}\left[S^t_{h^*} = s_{\ell^*}, A^t_{h^*} = a^*\right].
\end{aligned}$$

Hence, the optimal value in any of the MDPs is $\rho^* = (H - \overline{H} - d)(1/2 + \varepsilon)$, which is obtained by the policy that starts to traverse the tree at step $h^* - d$ then chooses to go to the leaf s_{ℓ^*} and performs action a^*. The expected regret of $\boldsymbol{\pi}$ in $\mathcal{M}_{(h^*,\ell^*,a^*)}$ is then

$$\overline{\mathcal{R}}_T\left(\boldsymbol{\pi}, \mathcal{M}_{(h^*,\ell^*,a^*)}\right) = T(H - \overline{H} - d)\varepsilon\left(1 - \frac{1}{T}\mathbf{E}_{(h^*,\ell^*,a^*)}\left[N^T_{(h^*,\ell^*,a^*)}\right]\right),$$

where $N^T_{(h^*,\ell^*,a^*)} = \sum_{t=1}^T \mathbb{1}\left\{S^t_{h^*} = s_{\ell^*}, A^t_{h^*} = a^*\right\}$.

Maximum expected regret of $\boldsymbol{\pi}$ over all possible $\mathcal{M}_{(h^*,\ell^*,a^*)}$. We first lower bound the maximum of the regret by the mean over all instances

$$\begin{aligned}
&\max_{(h^*,\ell^*,a^*)} \overline{\mathcal{R}}_T\left(\boldsymbol{\pi}, \mathcal{M}_{(h^*,\ell^*,a^*)}\right) \\
&\qquad \geq \frac{1}{\overline{H}LA}\sum_{(h^*,\ell^*,a^*)} \overline{\mathcal{R}}_T\left(\boldsymbol{\pi}, \mathcal{M}_{(h^*,\ell^*,a^*)}\right) \\
&\qquad \geq T(H - \overline{H} - d)\varepsilon\left(1 - \frac{1}{\overline{H}LAT}\sum_{(h^*,\ell^*,a^*)} \mathbf{E}_{(h^*,\ell^*,a^*)}\left[N^T_{(h^*,\ell^*,a^*)}\right]\right), \qquad (3.4)
\end{aligned}$$

so that, in order to lower bound the regret, we need an upper bound on the sum over all MDP instances (h^*, ℓ^*, a^*) of $\mathbf{E}_{(h^*,\ell^*,a^*)}\left[N^T_{(h^*,\ell^*,a^*)}\right]$. For this purpose, we will relate each expectation to the expectation of the same quantity under the reference MDP $\mathcal{M}_0$.

Upper bound on $\sum \mathbf{E}_{(h^*,\ell^*,a^*)}\left[N^T_{(h^*,\ell^*,a^*)}\right]$. Since $N^T_{(h^*,\ell^*,a^*)}/T \in [0,1]$, Lemma 3.7 gives us

$$\mathrm{kl}\left(\frac{1}{T}\mathbf{E}_0\left[N^T_{(h^*,\ell^*,a^*)}\right], \frac{1}{T}\mathbf{E}_{(h^*,\ell^*,a^*)}\left[N^T_{(h^*,\ell^*,a^*)}\right]\right) \leq \mathrm{KL}\left(\mathbf{P}_0^{B_H^T}, \mathbf{P}_{(h^*,\ell^*,a^*)}^{B_H^T}\right).$$

By Pinsker's inequality, $(p-q)^2 \leq (1/2)\,\mathrm{kl}(p,q)$, it implies

$$\frac{1}{T}\mathbf{E}_{(h^*,\ell^*,a^*)}\left[N^T_{(h^*,\ell^*,a^*)}\right] \leq \frac{1}{T}\mathbf{E}_0\left[N^T_{(h^*,\ell^*,a^*)}\right] + \sqrt{\frac{1}{2}\mathrm{KL}\left(\mathbf{P}_0^{B_H^T}, \mathbf{P}_{(h^*,\ell^*,a^*)}^{B_H^T}\right)}$$

and, by Lemma 3.6, we know that

$$\mathrm{KL}\left(\mathbf{P}_0^{B_H^T}, \mathbf{P}_{(h^*,\ell^*,a^*)}^{B_H^T}\right) = \mathbf{E}_0\left[N_{(h^*,\ell^*,a^*)}^T\right] \mathrm{kl}(1/2, 1/2+\varepsilon)$$

since $\mathcal{M}_0$ and $\mathcal{M}_{(h^*,\ell^*,a^*)}$ only differ at stage h^* when $(s,a) = (s_{\ell^*}, a^*)$. Assuming that $\varepsilon \leq 1/4$, we have $\mathrm{kl}(1/2, 1/2+\varepsilon) \leq 4\varepsilon^2$ by Lemma B.1, and, consequently

$$\frac{1}{T}\mathbf{E}_{(h^*,\ell^*,a^*)}\left[N_{(h^*,\ell^*,a^*)}^T\right] \leq \frac{1}{T}\mathbf{E}_0\left[N_{(h^*,\ell^*,a^*)}^T\right] + \sqrt{2}\varepsilon\sqrt{\mathbf{E}_0\left[N_{(h^*,\ell^*,a^*)}^T\right]}. \tag{3.5}$$

The sum of $N_{(h^*,\ell^*,a^*)}^T$ over all instances $(h^*,\ell^*,a^*) \in \left\{1+d,\dots,\overline{H}+d\right\} \times \mathcal{L} \times \mathcal{A}$ is

$$\sum_{(h^*,\ell^*,a^*)} N_{(h^*,\ell^*,a^*)}^T = \sum_{t=1}^{T}\sum_{h^*=1+d}^{\overline{H}+d} \mathbb{1}\left\{S_{h^*}^t \in \mathcal{L}\right\} = T \tag{3.6}$$

since for a single stage $h^* \in \left\{1+d,\dots,\overline{H}+d\right\}$, we have $S_{h^*}^t \in \mathcal{L}$ almost surely.

Summing (3.5) over all instances (h^*,ℓ^*,a^*) and using (3.6), we obtain using the Cauchy-Schwartz inequality that

$$\begin{aligned}\frac{1}{T}\sum_{(h^*,\ell^*,a^*)}\mathbf{E}_{(h^*,\ell^*,a^*)}\left[N_{(h^*,\ell^*,a^*)}^T\right] &\leq 1 + \sqrt{2}\varepsilon\sum_{(h^*,\ell^*,a^*)}\sqrt{\mathbf{E}_0\left[N_{(h^*,\ell^*,a^*)}^T\right]} \\ &\leq 1 + \sqrt{2}\varepsilon\sqrt{\overline{H}LAT}\,. \end{aligned} \tag{3.7}$$

Optimizing ε and choosing $\overline{H}$. Plugging (3.7) in (3.4), we obtain

$$\max_{(h^*,\ell^*,a^*)} \overline{\mathcal{R}}_T(\boldsymbol{\pi}, \mathcal{M}_{(h^*,\ell^*,a^*)}) \geq T(H-\overline{H}-d)\varepsilon\left(1 - \frac{1}{\overline{H}LA} - \frac{\sqrt{2}\varepsilon\sqrt{\overline{H}LAT}}{\overline{H}LA}\right).$$

The value of ε that maximizes the lower bound is $\varepsilon = \frac{1}{2\sqrt{2}}\left(1-\frac{1}{\overline{H}LA}\right)\sqrt{\frac{\overline{H}LA}{T}}$ which yields

$$\max_{(h^*,\ell^*,a^*)} \overline{\mathcal{R}}_T(\boldsymbol{\pi}, \mathcal{M}_{(h^*,\ell^*,a^*)}) \geq \frac{1}{4\sqrt{2}}\left(1-\frac{1}{\overline{H}LA}\right)(H-\overline{H}-d)\sqrt{\overline{H}LAT}. \tag{3.8}$$

The number of leaves is $L = (1-1/A)(S-3)+1/A \geq S/4$, since $A \geq 2$ and $S \geq 6$. We choose $\overline{H} = H/3$ and use the assumptions that $A \geq 2$ and $d \leq H/3$ to obtain

$$\max_{(h^*,\ell^*,a^*)} \overline{\mathcal{R}}_T(\boldsymbol{\pi}, \mathcal{M}_{(h^*,\ell^*,a^*)}) \geq \frac{1}{48\sqrt{6}}H\sqrt{HSAT}.$$

Finally, the assumption that $\varepsilon \leq 1/4$ is satisfied if $T \geq HSA$. □

3.4 Lower Bound on the Sample Complexity

Using again change of distributions between MDPs in a class $\mathcal{C}_{\overline{H},\varepsilon}$, we prove a minimax lower bound on the expected sample complexity of best-policy identification. We note that unlike existing sample complexity lower bounds which also construct "bandit-like" hard instances [SLL09; LH12; DB15], we do not refer to the bandit lower bound of Mannor and Tsitsiklis [MT04], but instead use explicit change of distribution arguments based on the tools given in Section 3.2.2. This allows us to provide BPI lower bounds for algorithms that output randomized policies and to have a self-contained proof. As a consequence of this result, we then derive a PAC-MDP lower bound in Corollary 3.10.

Theorem 3.9. *Let $(\boldsymbol{\pi}, \tau, \widehat{\pi}_\tau)$ be an algorithm that is (ε, δ)-PAC for best policy identification in any finite MDP. Then, under Assumption 3.4, there exists an MDP $\mathcal{M}$ and an initial distribution μ such that for $\varepsilon \leq H/24$, $H \geq 4$ and $\delta \leq 1/16$,*

$$\mathbf{E}_{\boldsymbol{\pi},\mathcal{M}}[\tau] \geq \frac{1}{3456}\frac{H^3SA}{\varepsilon^2}\log\left(\frac{1}{\delta}\right).$$

Corollary 3.10. *Let $\boldsymbol{\pi}$ be an algorithm that is (ε, δ)-PAC for exploration and that, in each episode t, plays a deterministic policy π_t. Then, under the assumptions of Theorem 3.9, there exists an MDP $\mathcal{M}$ such that*

$$\mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}\left[\mathcal{N}_\varepsilon^{\mathrm{PAC}} > \frac{1}{6912}\frac{H^3SA}{\varepsilon^2}\log\left(\frac{1}{\delta}\right) - 1\right] > \delta.$$

Proof. (*of Theorem 3.9*) Without loss of generality, we assume that for any $\mathcal{M}$, the algorithm satisfies $\mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}[\tau < \infty] = 1$. Otherwise, there exists an MDP with $\mathbf{E}_{\boldsymbol{\pi},\mathcal{M}}[\tau] = +\infty$ and the lower bound is trivial.

We will prove that the lower bound holds for the reference MDP $\mathcal{M}_0$ defined in Section 3.2.1, which has no optimal action. To do so, we will consider changes of distributions with other MDPs in the class $\mathcal{C}_{\overline{H},\widetilde{\varepsilon}}$ for $\overline{H}$ to be chosen later and $\widetilde{\varepsilon} := 2\varepsilon/(H - \overline{H} - d)$. As initial distribution μ on the set of states, we take the one that assigns probability 1 to the waiting state s_{w}: $\mu(s_{\mathrm{w}}) = 1$. These MDPs are of the form $\mathcal{M}_{(h^*,\ell^*,a^*)}$ with $(h^*, \ell^*, a^*) \in \left\{1 + d, \dots, \overline{H} + d\right\} \times \mathcal{L} \times \mathcal{A}$, for which

$$\Delta_{(h^*,\ell^*,a^*)}(h, s_i, a) = \mathbb{1}\{h = h^*, s_i = s_{\ell^*}, a = a^*\}\,\widetilde{\varepsilon}.$$

We recall that $d-1$ is the depth of the tree. We denote by $\mathbf{P}_{(h^*,\ell^*,a^*)} := \mathbf{P}_{\boldsymbol{\pi},\mathcal{M}_{(h^*,\ell^*,a^*)}}$ and $\mathbf{E}_{(h^*,\ell^*,a^*)} := \mathbf{E}_{\boldsymbol{\pi},\mathcal{M}_{(h^*,\ell^*,a^*)}}$ the probability measure and expectation in the MDP $\mathcal{M}_{(h^*,\ell^*,a^*)}$ by following $\boldsymbol{\pi}$ and by $\mathbf{P}_0$ and $\mathbf{E}_0$ the corresponding operators in the MDP $\mathcal{M}_0$.

Suboptimality gap of $\widehat{\pi}_\tau$. For any $(h^*, \ell^* a^*)$, let $\rho^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)}$ be the value of the recommended policy $\widehat{\pi}_\tau$ when the algorithm is run in the MDP $\mathcal{M}_{(h^*,\ell^* a^*)}$. We can show that the value of the optimal policy in any of the MDPs $\mathcal{M}_{(h^*,\ell^* a^*)}$ is $\rho^* := (H - \overline{H} - d)\left(\frac{1}{2} + \widetilde{\varepsilon}\right)$ and the value of the recommended policy $\widehat{\pi}_\tau$ is

$$\rho^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)} = (H - \overline{H} - d)\left(\frac{1}{2} + \widetilde{\varepsilon}\,\mathbf{P}^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)}\left[S_{h^*} = s_{\ell^*}, A_{h^*} = a^*\right]\right),$$

where $\mathbf{P}^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)}$ is the probability distribution over states and actions $(S_h, A_h)_{h\in[H]}$ following the Markov policy $\widehat{\pi}_\tau$ in the MDP $\mathcal{M}_{(h^*,\ell^* a^*)}$. Notice that $\rho^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)}$ is a random variable and $\mathbf{P}^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)}$ is a random measure that are $\mathcal{F}^\tau_H$-measurable. Hence,

$$\rho^* - \rho^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)} = 2\varepsilon\left(1 - \mathbf{P}^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)}\left[S_{h^*} = s_{\ell^*}, A_{h^*} = a^*\right]\right)$$

and

$$\rho^* - \rho^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)} < \varepsilon \iff \mathbf{P}^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)}\left[S_{h^*} = s_{\ell^*}, A_{h^*} = a^*\right] > \frac{1}{2}.$$

Definition of a "good" event $\mathcal{E}^\tau_{(h^*,\ell^*,a^*)}$ for $\mathcal{M}_{(h^*,\ell^* a^*)}$. The transitions of all MDPs are the same up to the stopping time $\eta = \min\{h \in [H] : S_h \in \mathcal{L}\}$ when a leaf is reached. Hence, η depends only on the policy that is followed, and not on the parameters of the MDP, which allows us to define the random measure $\mathbf{P}^{\widehat{\pi}_\tau}$ as

$$\begin{aligned}\mathbf{P}^{\widehat{\pi}_\tau}\left[S_{h^*} = s_{\ell^*}, A_{h^*} = a^*\right] &:= \mathbf{P}^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)}\left[S_\eta = s_{\ell^*}, A_\eta = a^*, \eta = h^*\right] \\ &= \mathbf{P}^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)}\left[S_{h^*} = s_{\ell^*}, A_{h^*} = a^*\right]\end{aligned} \tag{3.9}$$

since the probability distribution of (S_η, A_η, η) on the RHS of (3.9) does not depend on the parameters of the MDP (h^*, ℓ^*, a^*), given $\eta = h^*$. We define the event

$$\mathcal{E}^\tau_{(h^*,\ell^*,a^*)} := \left\{\mathbf{P}^{\widehat{\pi}_\tau}\left[S_{h^*} = s_{\ell^*}, A_{h^*} = a^*\right] > \frac{1}{2}\right\},$$

which is said to be "good" due to the fact that $\mathcal{E}^\tau_{(h^*,\ell^*,a^*)} = \left\{\rho^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)} > \rho^* - \varepsilon\right\}$. Since the algorithm is assumed to be (ε, δ)-PAC for any MDP, we have

$$\mathbf{P}_{(h^*,\ell^*,a^*)}\left[\mathcal{E}^\tau_{(h^*,\ell^*,a^*)}\right] = \mathbf{P}_{(h^*,\ell^*,a^*)}\left[\rho^{\widehat{\pi}_\tau}_{(h^*,\ell^*,a^*)} > \rho^* - \varepsilon\right] \geq 1 - \delta.$$

Lower bound on the expectation of τ in the reference MDP $\mathcal{M}_0$. Recall that

$$N^{\tau}_{(h^*,\ell^*,a^*)} = \sum_{t=1}^{\tau} \mathbb{1}\left\{S^t_{h^*} = s_{\ell^*}, A^t_{h^*} = a^*\right\} ,$$

such that $\sum_{(h^*,\ell^*,a^*)} N^{\tau}_{(h^*,\ell^*,a^*)} = \tau$. For any $\mathcal{F}^{\tau}_H$-measurable random variable Z taking values in $[0,1]$, we have

$$\begin{aligned}
&\mathbf{E}_0\left[N^{\tau}_{(h^*,\ell^*,a^*)}\right] \frac{16\varepsilon^2}{(H-\overline{H}-d)^2} \geq \mathbf{E}_0\left[N^{\tau}_{(h^*,\ell^*,a^*)}\right] \mathrm{kl}\left(\frac{1}{2}, \frac{1}{2}+\widetilde{\varepsilon}\right) \quad \text{by Lemma B.1} \\
&= \mathrm{KL}\left(\mathbf{P}_0^{B^{\tau}_H}, \mathbf{P}_{(h^*,\ell^*,a^*)}^{B^{\tau}_H}\right) \quad \text{by Lemma 3.6} \\
&\geq \mathrm{kl}\left(\mathbf{E}_0\left[Z\right], \mathbf{E}_{(h^*,\ell^*,a^*)}\left[Z\right]\right) \quad \text{by Lemma 3.7}
\end{aligned}$$

for any (h^*,ℓ^*,a^*), provided that $\widetilde{\varepsilon} \leq 1/4$. Letting $Z = \mathbb{1}\left\{\mathcal{E}^{\tau}_{(h^*,\ell^*,a^*)}\right\}$ yields

$$\begin{aligned}
&\mathrm{kl}\left(\mathbf{E}_0\left[Z\right], \mathbf{E}_{(h^*,\ell^*,a^*)}\left[Z\right]\right) = \mathrm{kl}\left(\mathbf{P}_0\left[\mathcal{E}^{\tau}_{(h^*,\ell^*,a^*)}\right], \mathbf{P}_{(h^*,\ell^*,a^*)}\left[\mathcal{E}^{\tau}_{(h^*,\ell^*,a^*)}\right]\right) \\
&\geq \left(1-\mathbf{P}_0\left[\mathcal{E}^{\tau}_{(h^*,\ell^*,a^*)}\right]\right)\log\left(\frac{1}{1-\mathbf{P}_{(h^*,\ell^*,a^*)}\left[\mathcal{E}^{\tau}_{(h^*,\ell^*,a^*)}\right]}\right) - \log(2) \quad \text{by Lemma B.2} \\
&\geq \left(1-\mathbf{P}_0\left[\mathcal{E}^{\tau}_{(h^*,\ell^*,a^*)}\right]\right)\log\left(\frac{1}{\delta}\right) - \log(2).
\end{aligned}$$

Consequently,

$$\mathbf{E}_0\left[N^{\tau}_{(h^*,\ell^*,a^*)}\right] \geq \frac{(H-\overline{H}-d)^2}{16\varepsilon^2}\left[\left(1-\mathbf{P}_0\left[\mathcal{E}^{\tau}_{(h^*,\ell^*,a^*)}\right]\right)\log\left(\frac{1}{\delta}\right) - \log(2)\right].$$

Summing over all MDP instances, we obtain

$$\begin{aligned}
\mathbf{E}_0\left[\tau\right] &\geq \sum_{(h^*,\ell^*,a^*)} \mathbf{E}_0\left[N^{\tau}_{(h^*,\ell^*,a^*)}\right] \\
&\geq \frac{(H-\overline{H}-d)^2}{16\varepsilon^2}\left[\left(\overline{H}LA - \sum_{(h^*,\ell^*,a^*)} \mathbf{P}_0\left[\mathcal{E}^{\tau}_{(h^*,\ell^*,a^*)}\right]\right)\log\left(\frac{1}{\delta}\right) - \overline{H}LA\log(2)\right]. \qquad (3.10)
\end{aligned}$$

Now, we have

$$\sum_{(h^*,\ell^*,a^*)} \mathbf{P}_0\left[\mathcal{E}^{\tau}_{(h^*,\ell^*,a^*)}\right] = \mathbf{E}_0\left[\sum_{(h^*,\ell^*,a^*)} \mathbb{1}\left\{\mathbf{P}^{\widehat{\pi}_\tau}\left[S_{h^*} = s_{\ell^*}, A_{h^*} = a^*\right] > \frac{1}{2}\right\}\right] \leq 1. \qquad (3.11)$$

Above we used the fact that

$$\sum_{(h^*,\ell^*,a^*)} \mathbf{P}^{\widehat{\pi}_\tau}\left[S_{h^*} = s_{\ell^*}, A_{h^*} = a^*\right] = \sum_{h^*} \mathbf{P}^{\widehat{\pi}_\tau}\left[S_{h^*} \in \mathcal{L}\right] = 1$$

since, at a single stage $h^* \in \left\{1+d, \overline{H}+d\right\}$, a leaf state will be reached almost surely. This implies that, if there exists (h^*, ℓ^*, a^*) such that $\mathbf{P}^{\widehat{\pi}_\tau}\left[S_{h^*} = s_{\ell^*}, A_{h^*} = a^*\right] > \frac{1}{2}$, then, for any other $(h', \ell', a') \neq (h^*, \ell^*, a^*)$, we have $\mathbf{P}^{\widehat{\pi}_\tau}\left[S_{h'} = s_{\ell'}, A_{h'} = a'\right] < \frac{1}{2}$, which proves (3.11).

Plugging (3.11) in (3.10) yields

$$\begin{aligned}\mathbf{E}_0\left[\tau\right] &\geq \frac{(H-\overline{H}-d)^2}{16\varepsilon^2}\left[\left(\overline{H}LA-1\right)\log\left(\frac{1}{\delta}\right) - \overline{H}LA\log(2)\right] \\ &\geq \overline{H}LA\frac{(H-\overline{H}-d)^2}{32\varepsilon^2}\log\left(\frac{1}{\delta}\right), \end{aligned} \tag{3.12}$$

where we used the assumption that $\delta \leq 1/16$. The number of leaves $L = (1-1/A)(S-3)+1/A$ satisfies $L \geq S/4$, since we assume $A \geq 2$, $S \geq 6$. Taking $\overline{H} = H/3$ and with the assumption $d \leq H/3$, we obtain

$$\mathbf{E}_0\left[\tau\right] \geq \frac{H^3SA}{3456\varepsilon^2}\log\left(\frac{1}{\delta}\right).$$

Finally, the condition $\varepsilon \leq H/24$ implies that $\widetilde{\varepsilon} \leq 1/4$, as required above.

□

3.5 Lower Bounds: Extensions

The lower bounds presented in the previous sections hold under Assumption 3.4, and assume that the rewards r_h and transitions p_h depend on the stage $h \in [H]$, *i.e.*, that we have *time-inhomogeneous* MDPs. In this section, we relax Assumption 3.4 and explain how to generalize the lower bounds to *time-homogeneous* MDPs, where $r_h := r$ and $p_h := p$ are independent of h.

3.5.1 Relaxing Assumption 3.4

In the proofs of Theorem 3.8 and Theorem 3.9, we use Assumption 3.4 stating that

- **(i)** there exists an integer d such that $S = 3 + (A^d - 1)/(A-1)$, and
- **(ii)** $H \geq 3d$,

which we discuss below.

Relaxing (i)

Assumption (i) makes the proof simpler by allowing us to consider a *full* A-ary tree with $S-3$ nodes, which implies that all the leaves are at the same level $d-1$ in the tree. The proof can be generalized to any $S \geq 6$ by arranging the states in a balanced, but not necessarily full, A-ary tree. In this case, there might be a subset of the leaves at a level $d-1$ and another subset at

a level $d-2$, which creates an asymmetry in the leaf nodes. To handle this, we proceed as follows:

- First, using $(S-3)/2$ states, we build a balanced A-ary tree of depth $d-1$;
- For each leaf at depth $d-2$, we add another state (taken among the remaining $(S-3)/2$ states) as its child.
- Any remaining state that was not added to the tree (and is not s_{w}, s_g or s_b), can be merged to the absorbing states s_g or s_b.

This construction ensures that we have a tree with at least $(S-3)/2$ and at most $(S-3)$ nodes, where all the leaves are at the same depth $d-1$, for

$$d = \lceil \log_A \left((S-3)(A-1)+1\right) \rceil \in [\log_A S - 1, \log_A S + 2]\,. \tag{3.13}$$

Lemma B.4 shows that the number of leaves L in this tree satisfies $S \geq L \geq (S-3)/8$. Hence, in the proofs of Theorem 3.8 (Eq. 3.8) and, Theorem 3.9 (Eq. 3.12) we take $L \geq (S-3)/8$ and obtain lower bounds of the same order.

Relaxing (ii)

Equation (3.13) implies that there exists a constant $c \in [-1, 2]$ such that $d = \log_A S + c$. Assumption (ii), stating that $H \geq 3d = 3\log_A S + 3c$ ensures that the horizon is large enough with respect to the size of the MDP for the agent to be able to traverse the tree down to the rewarding state. If this condition is not satisfied, that is, if $H < 3\log_A S + 3c$, we have $S \geq A^{\frac{H}{3}-2}$. In this case, we can build a tree using a subset of the state space containing $\left\lceil A^{\frac{H}{3}-2} \right\rceil$ states, and merge the remaining $S - \left\lceil A^{\frac{H}{3}-2} \right\rceil$ states to the absorbing states s_b or s_g. In this case, the resulting bounds will replace S by $\left\lceil A^{\frac{H}{3}-2} \right\rceil$, and become exponential in the horizon H,

$$\Omega\left(\sqrt{H^3 \left\lceil A^{\frac{H}{3}-2} \right\rceil T}\right) \quad \text{and} \quad \Omega\left(\frac{\left\lceil A^{\frac{H}{3}-2} \right\rceil AH^3}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right)$$

for regret and BPI, respectively.

The arguments above give us Theorem 3.11, Theorem 3.12, and Corollary 3.13 below, which state regret, BPI, and PAC-MDP lower bounds, respectively, without requiring Assumption 3.4.

Theorem 3.11. *If $S \geq 11$, $A \geq 4$ and $H \geq 6$, for any algorithm $\boldsymbol{\pi}$, there exists an MDP $\mathcal{M}_{\boldsymbol{\pi}}$ such that, for $T \geq HSA$*

$$\overline{\mathcal{R}}_T \geq c_3 \sqrt{\min\left(S, A^{\frac{H}{3}-2}\right)} \sqrt{H^3 AT}\,,$$

where c_3 is an absolute constant.

Proof. If $S \leq A^{\frac{H}{3}-2}$, then $H \geq 3d$, where d is given in Equation 3.13. In this case, we follow the proof of Theorem 3.8 up to Equation 3.8, where we take $L \geq (S-3)/8$ according to the arguments in Section 3.5.1. If $S > A^{\frac{H}{3}-2}$, then $H < 3d$ and we follow the arguments in Section 3.5.1. □

Theorem 3.12. *Let $(\boldsymbol{\pi}, \tau, \widehat{\pi}_\tau)$ be an algorithm that is (ε, δ)-PAC for best policy identification in any finite MDP. Then, if $S \geq 11$, $A \geq 4$ and $H \geq 6$, there exists an MDP $\mathcal{M}$ with stage-dependent transitions such that for $\varepsilon \leq H/24$ and $\delta \leq 1/16$,*

$$\mathbf{E}_{\boldsymbol{\pi},\mathcal{M}}[\tau] \geq c_1 \min\left(S, A^{\frac{H}{3}-2}\right) \frac{H^3 A}{\varepsilon^2} \log\left(\frac{1}{\delta}\right),$$

where c_1 is an absolute constant.

Proof. If $S \leq A^{\frac{H}{3}-2}$, then $H \geq 3d$, where d is given in Equation 3.13. In this case, we follow the proof of Theorem 3.9 up to Equation 3.12, where we take $L \geq (S-3)/8$ according to the arguments in Section 3.5.1. If $S > A^{\frac{H}{3}-2}$, then $H < 3d$ and we follow the arguments in Section 3.5.1. □

Corollary 3.13. *Let $\boldsymbol{\pi}$ be an algorithm that is (ε, δ)-PAC for exploration and that, in each episode t, plays a deterministic policy π_t. Then, under the conditions of Theorem 3.12, there exists an MDP $\mathcal{M}$ such that*

$$\mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}\left[\mathcal{N}_\varepsilon^{\mathrm{PAC}} > c_2 \min\left(S, A^{\frac{H}{3}-2}\right) \frac{H^3 A}{\varepsilon^2} \log\left(\frac{1}{\delta}\right) - 1\right] > \delta.$$

where c_2 is an absolute constant.

Proof. Analogous to the proof of Corollary 3.10, using Theorem 3.12 instead of Theorem 3.9. □

3.5.2 Lower Bounds for Time-Homogeneous MDPs

The proofs of Theorem 3.8 and Theorem 3.9 can be adapted to the case where the rewards r_h and transitions p_h do not depend on h. To do so, we need to have a set of hard MDPs with time-homogeneous transitions. For that, we remove the waiting state s_{w} and the agent starts at s_{root}, which roughly corresponds to setting $\overline{H} = 1$ in the proofs, and we take

$$\Delta_{(h^*,\ell^*,a^*)}(h, s_i, a) := \mathbb{1}\{(s_i, a) = (s_{\ell^*}, a^*)\}\,\varepsilon'$$

to be independent of h. The h-independent rewards are taken as

$$\forall a \in \mathcal{A}, \quad r_h(s, a) = \mathbb{1}\{s = s_g\}.$$

Since $\overline{H} = 1$ and no longer $H/3$, the regret bound becomes $\Omega(\sqrt{H^2SAT})$ and the BPI bound becomes $\Omega\left(\frac{SAH^2}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right)$.

3.6 Upper Bound on the Regret of UCBVI

In Section 3.1, we saw that both the regret and the PAC-MDP criteria allow us to measure how well an agent balances exploration and exploitation, whereas the BPI criterion is a pure-exploration objective. In this section and in the next chapter, we focus on algorithms for regret minimization. In Section 3.7 we present references to other works tackling the PAC-MDP and the BPI criteria.

Azar et al. [AOM17] introduced the UCBVI algorithm, meaning *upper confidence bound value iteration*, and proved regret upper bounds that match the lower bound up to logarithmic factors, provided that the number of episodes T is large enough. In this section, we provide a simplified analysis of UCBVI-CH, which is a version of UCBVI discussed below, that we generalize to continuous MDPs in Chapter 4. Our simplification relies on Lemma 3.16, and is discussed in Section 3.6.2.

Minimax optimality of UCBVI Jaksch et al. [JOA10] introduced the UCRL algorithm for regret minimization in the average-reward setting, which can be adapted to the finite-horizon setting, resulting in a regret upper bound of order $\widetilde{\mathcal{O}}\left(\sqrt{H^3S^2AT}\right)$ for time-homogeneous MDPs. Recall from Section 3.5.2 that the lower bound in this setting is $\Omega\left(\sqrt{H^2SAT}\right)$. Azar et al. [AOM17] also assume time-homogeneous MDPs, and provide two different versions of UCBVI: UCBVI-CH, based on Chernoff-Hoeffding's concentration inequality, and UCBVI-BF, based on Bernstein-Freedman's concentration inequalities. For T large enough, the regret of UCBVI-CH is $\mathcal{O}\left(\sqrt{H^3SAT}\right)$ and the regret of UCBVI-BF is $\mathcal{O}\left(\sqrt{H^2SAT}\right)$. That is, UCBVI-CH improves the regret of UCRL with respect to S, whereas UCBVI-BF brings an extra improvement with respect to H and matches the lower bound.

Time-homogeneous versus time-inhomogeneous MDPs In the analysis of UCBVI presented here, we consider the UCBVI-CH version in the *time-inhomogeneous* case, where the lower bound is $\Omega\left(\sqrt{H^3SAT}\right)$ (Theorem 3.8) and we prove a regret upper bound of order $\mathcal{O}\left(\sqrt{H^4SAT}\right)$, which is suboptimal only by a factor $\sqrt{H}$. Our choice for the time-inhomogeneous case is motivated by its generality and by mathematical convenience. The extension of our analysis to the time-homogeneous case is straightforward (*e.g.*, [Dom+21d, Appendix E]).

Notation and Assumptions We consider Assumption 1.1 stating that the reward functions are known, and we further assume that the rewards are nonnegative and bounded by $r_{\max} = 1$.

For any transition kernel p and any function $f : \mathcal{S} \to \mathbb{R}$, we denote

$$pf(s,a) := \sum_{z \in \mathcal{S}} p(z|s,a) f(z). \tag{3.14}$$

In the regret analysis, we omit logarithmic factors using the following notation:

$$A \lesssim B \iff A \leq B \times \text{polynomial}\left(\log(T), \log(1/\delta), \log(HSA)\right). \tag{3.15}$$

Also for the regret analysis, we denote by $\mathcal{F}_h^t$ the σ-algebra generated by all the state-action pairs observed up to time (t,h), that is, the h-th step of the t-th episode.

3.6.1 Description of the `UCBVI` Algorithm

In each episode $t \in [T]$, `UCBVI` (Algorithm 3.1) computes Q-functions Q_h^t for all $h \in [H]$ based on the data observed up to episode $t-1$, such that Q_h^t is an upper confidence bound on the optimal Q-function Q_h^*. That is, $Q_h^t \succeq Q_h^*$ with high probability, so that we refer to Q_h^t as an *optimistic Q-function*. Then, `UCBVI` executes the policy $\boldsymbol{\pi}^t$, defined as $\boldsymbol{\pi}^t(s,h) := \operatorname{argmax}_a Q_h^t(s,a)$.

In order to compute optimistic Q-functions Q_h^t, `UCBVI` builds *transition models* $(\widehat{p}_h^t)_h$ that approximate the true transitions $(p_h)_h$, and uses exploration bonuses $\mathtt{b}_h^t : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$ such that $\mathtt{b}_h^t(s,a)$ measures the estimation error of $\widehat{p}_h^t(\cdot|s,a)$ when compared to $p_h(\cdot|s,a)$.

Algorithm 3.1: `UCBVI`

for *episode* $t \in \{1, \dots, T\}$ **do**
  get initial state s_1^t
  # compute optimistic Q-functions
  compute $(Q_h^t)_{h \in [H]}$ according to Equation (3.18)
  for *stage* $h \in \{1, \dots, H\}$ **do**
    # select action
    $a_h^t \leftarrow \operatorname{argmax}_a Q_h^t(s_h^t, a)$
    # execute action
    $s_{h+1}^t \leftarrow \mathtt{OnlineModel}_{t,h}(a_h^t)$
    # update model
    compute $\widehat{p}_h^{t+1}$ using Equation (3.16)

Let $n_h^t(s,a)$ be the number of times that the state-action pair (s,a) has been visited at stage h before episode t, which is initialized to 1 if (s,a) has not been visited:

$$n_h^t(s,a) := 1 \vee \sum_{i=1}^{t-1} \mathbb{1}\left\{(s_h^i, a_h^i) = (s,a)\right\}.$$

where $\forall x, y \in \mathbb{R}, x \vee y := \max(x, y)$. The estimated transitions $\widehat{p}_h^t$ are defined as

$$\widehat{p}_h^t(z|s,a) := \frac{1}{n_h^t(s,a)} \sum_{i=1}^{t-1} \mathbb{1}\left\{(s_h^i, a_h^i) = (s,a)\right\} \delta_{s_{h+1}^i}(z), \tag{3.16}$$

where $\forall s \in \mathcal{S}$, δ_s is the Dirac measure at s.

The exploration bonuses $\mathtt{b}_h^t(s,a)$ are defined as

$$\mathtt{b}_h^t(s,a) := \sqrt{\frac{2H^2}{n_h^t(s,a)} \log\left(\frac{6HSAn_h^t(s,a)(n_h^t(s,a)+1)}{\delta}\right)}. \tag{3.17}$$

The optimistic Q-functions Q_h^t are computed recursively as follows:

$$\forall h \in [H],\ Q_h^t(s,a) := r_h(s,a) + \widehat{p}_h^t V_{h+1}^t(s,a) + \mathtt{b}_h^t(s,a), \tag{3.18}$$

where, for all $s \in \mathcal{S}$, $V_{H+1}^t(s) := 0$ and $V_h^t(s) := \min(H-h+1, \max_a Q_h^t(s,a))$, and where we used the notation (3.14) for the expectation with respect to $\widehat{p}_h^t$.

3.6.2 Regret Analysis

Theorem 3.14 provides a high-probability upper bound on the regret of UCBVI. Its proof is split into three parts: (i) deriving confidence intervals for the transitions estimators $\widehat{p}_h^t$; (ii) proving that the algorithm is optimistic, *i.e.*, that $V_h^t(s) \geq V_h^*(s)$ for any (s,t,h) on a high-probability event $\mathcal{G}$; (iii) proving an upper bound on the regret by using the fact that $\mathcal{R}_T = \sum_t \left(V_1^*(s_1^t) - V_1^{\pi^t}(s_1^t)\right) \leq \sum_t \left(V_1^t(s_1^t) - V_1^{\pi^t}(s_1^t)\right)$ on the event $\mathcal{G}$.

Theorem 3.14. *With probability at least $1-\delta$, the regret of* UCBVI *satisfies*

$$\mathcal{R}_T \lesssim H^2\sqrt{SAT} + H^{\frac{3}{2}}\sqrt{T} + H^3S^2A,$$

where $\lesssim$ omits constant and logarithmic factors, as defined in (3.15).

According to Theorem 3.14, if $T \geq H^2S^3A$, the regret of UCBVI satisfies $\mathcal{R}_T \lesssim \sqrt{H^4SAT}$ with high probability, whereas the lower bound of Theorem 3.8 is of order $\Omega\left(\sqrt{H^3SAT}\right)$. As we will see in the analysis below, the extra factor $\sqrt{H}$ in the upper bound comes from the fact that we use Hoeffding's inequality to define the exploration bonus $\mathtt{b}_h^t$ (Equation 3.17). Azar et al. [AOM17] show that the upper bound can be improved by a factor $\sqrt{H}$ by using empirical Bernstein's inequality instead, matching the lower bound if T is large enough. Here, we do not study such improved version of UCBVI, since the dependence on H of Kernel-UCBVI— our

generalization of UCBVI beyond finite MDPs (Chapter 4) — comes from the term H^3S^2A, as discussed in Section 4.3.

Part 1: Confidence Intervals

Lemma 3.15 gives the confidence intervals used to define UCBVI's exploration bonuses (3.17).

Lemma 3.16 is used in the analysis to improve the regret by a factor of $\sqrt{S}$ when compared to UCRL. It is the simplification of a similar technique used by [AOM17] and can be generalized to continuous MDPs. This generalization would not be possible if we relied on the technique used by Azar et al. [AOM17], since they need to distinguish the set of states z where $\widehat{p}_h^t(z|s_h^t, a_h^t)n_h^t(s_h^t, a_h^t) \geq c$ and the set of states z where $\widehat{p}_h^t(z|s_h^t, a_h^t)n_h^t(s_h^t, a_h^t) < c$ for a positive constant c that depends on H. In continuous or arbitrary MDPs, the number of visits $n_h^t(s_h^t, a_h^t)$ is not well-defined, and $\widehat{p}_h^t$ can be a sum of Dirac measures, hence we can only handle *integration with respect to* $\widehat{p}_h^t$, and cannot treat $\widehat{p}_h^t(z|s_h^t, a_h^t)$ as a real number. Consequently, the advantage of Lemma 3.16 is that it only requires the integration of a function f with respect to $\widehat{p}_h^t$, for f in a given function space $\mathcal{V}$, and in Chapter 4 we propose a method that replaces $n_h^t(s_h^t, a_h^t)$ by a sum of weights that can be computed for arbitrary MDPs.

Lemma 3.15. *Let $\mathcal{G}_1$ be the event such that*

$$\mathcal{G}_1 := \left\{\forall t \geq 1,\ \forall (s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H],\ |(p_h - \widehat{p}_h^t)V_{h+1}^*(s,a)| \leq \mathtt{b}_h^t(s,a)\right\}.$$

Then, $\mathbf{P}\left[\mathcal{G}_1\right] \geq 1 - \delta/3$.

Proof. The proof is given in Appendix B.4.2, and is based on the Azuma-Hoeffding inequality. □

Lemma 3.16. *Let $\mathcal{V} = \left\{f \in \mathbb{R}^S : \|f\|_\infty \leq H, f \succeq 0\right\}$. and consider the event $\mathcal{G}_2$ defined as*

$$\mathcal{G}_2 := \left\{\forall (t,s,a,h,f),\ (p_h - \widehat{p}_h^t)f(s,a) \leq \frac{1}{H}p_h f(s,a) + \frac{55H^2S}{n_h^t(s,a)}\log\left(\frac{HSAn_h^t(s,a)}{\delta}\right)\right\}$$

where $t \geq 1$ and $(s,a,h,f) \in \mathcal{S} \times \mathcal{A} \times [H] \times \mathcal{V}$. Then, $\mathbf{P}\left[\mathcal{G}_2\right] \geq 1 - \delta/3$.

Proof. The proof is given in Appendix B.4.3, and is based on Bernstein's inequality and a union bound over a covering of the space $\mathcal{V}$. □

Part 2: Optimism

Lemma 3.17. *Consider the event $\mathcal{G}_1$ defined in Lemma 3.15. On $\mathcal{G}_1$, we have $V_h^t \succeq V_h^*$.*

Proof. We proceed by induction. For $h = H+1$, $V_h^t = V_h^* = 0$. Assume that the statement is true for $h+1$. Hence, for any (s,a),

$$\begin{aligned} Q_h^t(s,a) - Q_h^*(s,a) &= \widehat{p}_h^t V_{h+1}^t(s,a) - p_h V_{h+1}^*(s,a) + \mathtt{b}_h^t(s,a) \\ &= \underbrace{\widehat{p}_h^t(V_{h+1}^t - V_{h+1}^*)(s,a)}_{\geq 0 \text{ by the induction hypothesis}} + \underbrace{(\widehat{p}_h^t - p_h)V_{h+1}^*(s,a) + \mathtt{b}_h^t(s,a)}_{\geq 0 \text{ on the event } \mathcal{G}_1} \geq 0, \end{aligned}$$

which implies that $V_h^t(s) \geq V_h^*(s)$ for any s and concludes the proof. □

Part 3: Regret Bound

Consider the events $\mathcal{G}_1$ and $\mathcal{G}_2$ defined in lemmas 3.15 and 3.16, respectively. Let

$$\delta_h^t := V_h^t(s_h^t) - V_h^{\boldsymbol{\pi}^t}(s_h^t).$$

By Lemma 3.17, on the event $\mathcal{G}_1$, we have

$$\mathcal{R}_T = \sum_{t=1}^T \left(V_1^*(s_1^t) - V_1^{\boldsymbol{\pi}^t}(s_1^t)\right) \leq \sum_{t=1}^T \left(V_1^t(s_1^t) - V_1^{\boldsymbol{\pi}^t}(s_1^t)\right) = \sum_{t=1}^T \delta_1^t.$$

Since $\delta_{H+1}^t = 0$, we bound δ_h^t recursively in order to prove Theorem 3.14. At any stage h and episode t, we have

$$\begin{aligned} \delta_h^t &= V_h^t(s_h^t) - V_h^{\boldsymbol{\pi}^t}(s_h^t) \leq Q_h^t(s_h^t, a_h^t) - Q_h^{\boldsymbol{\pi}^t}(s_h^t, a_h^t) \\ &\leq (\widehat{p}_h^t - p_h)V_{h+1}^*(s_h^t, a_h^t) + (\widehat{p}_h^t - p_h)(V_{h+1}^t - V_{h+1}^*) + p_h(V_{h+1}^t - V_{h+1}^{\boldsymbol{\pi}^t})(s_h^t, a_h^t) + \mathtt{b}_h^t(s_h^t, a_h^t). \end{aligned}$$

On $\mathcal{G}_1$, $(\widehat{p}_h^t - p_h)V_{h+1}^*(s_h^t, a_h^t) \leq \mathtt{b}_h^t(s_h^t, a_h^t)$, which implies

$$\begin{aligned} \delta_h^t &\leq (\widehat{p}_h^t - p_h)(V_{h+1}^t - V_{h+1}^*)(s_h^t, a_h^t) + p_h(V_{h+1}^t - V_{h+1}^{\boldsymbol{\pi}^t})(s_h^t, a_h^t) + 2\mathtt{b}_h^t(s_h^t, a_h^t) \\ &= \delta_{h+1}^t + \xi_{h+1}^t + 2\mathtt{b}_h^t(s_h^t, a_h^t) + (\widehat{p}_h^t - p_h)(V_{h+1}^t - V_{h+1}^*)(s_h^t, a_h^t), \end{aligned}$$

where $\xi_{h+1}^t := p_h(V_{h+1}^t - V_{h+1}^{\boldsymbol{\pi}^t})(s_h^t, a_h^t) - \delta_{h+1}^t$. Notice that $(\xi_{h+1}^t)_{t,h}$ is a martingale difference sequence with respect to $(\mathcal{F}_h^t)_{t,h}$, that is, $\mathbf{E}\left[\xi_{h+1}^t | \mathcal{F}_h^t\right] = 0$.

Lemma 3.16 allows us to bound the term $(\widehat{p}_h^t - p_h)(V_{h+1}^t - V_{h+1}^*)(s_h^t, a_h^t)$ and gives us, on the event $\mathcal{G}_2$,

$$\delta_h^t \lesssim \delta_{h+1}^t + \xi_{h+1}^t + \mathtt{b}_h^t(s_h^t, a_h^t) + \frac{1}{H} p_h(V_{h+1}^t - V_{h+1}^*)(s_h^t, a_h^t) + \frac{H^2 S}{n_h^t(s_h^t, a_h^t)},$$

and recall that the notation $\lesssim$ omits constant and logarithmic factors.

Introducing $\overline{\xi}_{h+1}^t$ defined as

$$\overline{\xi}_{h+1}^t := \frac{1}{H} p_h(V_{h+1}^t - V_{h+1}^*)(s_h^t, a_h^t) - \frac{1}{H}\left(V_{h+1}^t(s_{h+1}^t) - V_{h+1}^*(s_{h+1}^t)\right),$$

and using the fact that $V_{h+1}^t(s_{h+1}^t) - V_{h+1}^*(s_{h+1}^t) \leq V_{h+1}^t(s_{h+1}^t) - V_{h+1}^{\pi^t}(s_{h+1}^t) = \delta_{h+1}^t$, we obtain

$$\delta_h^t \lesssim \left(1 + \frac{1}{H}\right)\delta_{h+1}^t + \xi_{h+1}^t + \overline{\xi}_{h+1}^t + \mathtt{b}_h^t(s_h^t, a_h^t) + \frac{H^2 S}{n_h^t(s_h^t, a_h^t)}.$$

Since $(1 + 1/H)^{H-h+1} \leq \exp(1)$, we can show by induction that

$$\mathcal{R}_T \lesssim \sum_{t=1}^{T}\sum_{h=1}^{H}\left[\left(\xi_{h+1}^t + \overline{\xi}_{h+1}^t\right) + \frac{H}{\sqrt{n_h^t(s_h^t, a_h^t)}} + \frac{H^2 S}{n_h^t(s_h^t, a_h^t)}\right] \tag{3.19}$$

on the event $\mathcal{G}_1 \cap \mathcal{G}_2$, where we used the definition of $\mathtt{b}_h^t$ given by Equation (3.17).

Now, we bound separately the terms in the sum in Equation (3.19).

Bounding the sum of $\left(\xi_{h+1}^t + \overline{\xi}_{h+1}^t\right)$. Let $\eta_{h+1}^t := \xi_{h+1}^t + \overline{\xi}_{h+1}^t$. Since $(\eta_{h+1}^t)_{h,t}$ is a martingale difference sequence with respect to $\mathcal{F}_h^t$ and $|\eta_{h+1}^t| \leq 4H$ almost surely, the Azuma-Hoeffding inequality (see Lemma B.5) implies that

$$\sum_{t=1}^{T}\sum_{h=1}^{H}\eta_{h+1}^t \lesssim H\sqrt{HT}$$

on an event $\mathcal{G}_3$ that satisfies $\mathbf{P}[\mathcal{G}_3] \geq 1 - \delta/3$.

Bounding the sum of $1/\sqrt{n_h^t(s_h^t, a_h^t)}$. In order to bound this term, we relate this sum to an integral, and we apply the Cauchy–Schwarz inequality. First, we define

$$\tau_h(s, a) := \min\left\{t : n_h^t(s, a) > 1\right\}.$$

Using the fact that $\sum_{s,a} n_h^{T+1}(s, a) = T$, we have:

$$\begin{aligned}
\sum_{t=1}^{T}\sum_{h=1}^{H}\frac{1}{\sqrt{n_h^t(s_h^t, a_h^t)}} &= \sum_{h=1}^{H}\sum_{s,a}\sum_{t=1}^{T}\frac{\mathbb{1}\left\{(s_h^t, a_h^t) = (s, a)\right\}}{\sqrt{n_h^t(s, a)}} \\
&\leq \sum_{h=1}^{H}\sum_{s,a}\left(1 + \sum_{t=\tau_h(s,a)}^{T}\frac{n_h^{t+1}(s, a) - n_h^t(s, a)}{\sqrt{n_h^t(s, a)}}\right) \\
&\leq \sum_{h=1}^{H}\sum_{s,a}\sum_{t=\tau_h(s,a)}^{T}\frac{n_h^{t+1}(s, a) - n_h^t(s, a)}{\sqrt{n_h^{t+1}(s, a) - 1}} + HSA
\end{aligned}$$

$$
\begin{aligned}
&\leq \sum_{h=1}^{H} \sum_{s,a} \int_{2}^{n_h^{T+1}(s,a)} \frac{\mathrm{d}z}{\sqrt{z-1}} + HSA \\
&\leq 2 \sum_{h=1}^{H} \sum_{s,a} \sqrt{n_h^{T+1}(s,a)} + HSA \\
&\leq 2 \sum_{h=1}^{H} \sqrt{SAT} + HSA \quad \text{by the Cauchy–Schwarz inequality} \\
&= 2H\sqrt{SAT} + HSA.
\end{aligned}
$$

Bounding the sum of $1/n_h^t(s_h^t, a_h^t)$. Analogously to the previous case,

$$
\begin{aligned}
&\sum_{t=1}^{T} \sum_{h=1}^{H} \frac{1}{n_h^t(s_h^t, a_h^t)} \leq \sum_{h=1}^{H} \sum_{s,a} \int_{2}^{n_h^{T+1}(s,a)} \frac{\mathrm{d}z}{z-1} + HSA \\
&\leq SA \sum_{h=1}^{H} \sum_{s,a} \frac{1}{SA} \log(n_h^{T+1}(s,a)) + HSA \\
&\leq SA \sum_{h=1}^{H} \log\left(\frac{T}{SA}\right) + HSA \quad \text{by Jensen's inequality} \\
&= HSA \log\left(\frac{T}{SA}\right) + HSA.
\end{aligned}
$$

Finally, we obtain that

$$
\mathcal{R}_T \lesssim H^2\sqrt{SAT} + H\sqrt{T} + H^2SA + H^3S^2A
$$

on the event $\mathcal{G} := \mathcal{G}_1 \cap \mathcal{G}_2 \cap \mathcal{G}_3$. Since $\mathbf{P}[\mathcal{G}] \geq 1 - \delta$, this concludes the proof.

3.7 Discussion and Bibliographical Remarks

In this chapter, we presented three of the main criteria used to evaluate reinforcement learning algorithms in the online setting: the regret, the BPI and the PAC-MDP criteria. In the regret and PAC-MDP framework, the agent is evaluated for its performance during learning, and, in the BPI framework, the agent is evaluated only by the policy it recommends at the end. For each of those criteria, we provided lower bounds, both for time-homogeneous and time-inhomogeneous MDPs. For the regret criterion in time-inhomogeneous MDPs, we presented a simplified analysis of the `UCBVI` algorithm, whose regret upper bound nearly matches the lower bound. We discuss below related works with respect to upper and lower bounds in different settings.

Sample complexity lower bounds Sample complexity has mostly been studied in the γ-discounted setting for PAC-MDP algorithms [Kak03]. State-of-the art lower bounds are a $\Omega\left(\frac{SA}{\varepsilon^2}\log\left(\frac{S}{\delta}\right)\right)$ bound by Strehl et al. [SLL09] and a $\Omega\left(\frac{SA}{(1-\gamma)^3\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right)$ bound by Lattimore and Hutter [LH12]. A lower bound of the same order is provided by Azar et al. [AMK12] for the number of steps algorithms that have access to a generative model need to identify an ε-optimal value function. PAC-MDP algorithms in the finite-horizon setting with time-homogeneous MDPs were later studied by Dann and Brunskill [DB15], who also provide a lower bound. Unlike the previous ones, they do not lower bound the number of ε-mistakes of the algorithm, but rather state that any algorithm that outputs a deterministic policy $\widehat{\pi}$ that is ε-optimal with probability at least $1-\delta$, there exists an MDP where the expected number of episodes before $\widehat{\pi}$ is returned must be at least $\Omega\left(\frac{SAH^2}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right)$. This lower bound therefore applies to the sample complexity of best-policy identification. The "hard MDP" instances used to prove this worse-case bound are inspired by the ones of Strehl et al. [SLL09] and consist of S multi-armed bandit (MAB) problems played in parallel. Jiang et al. [Jia+17], Dann et al. [DLB17], and Yin et al. [YBW21] show that the PAC lower bound has an extra factor H for time-inhomogeneous MDPs, and also rely on a construction of hard instances based on parallel MAB instances. In this chapter, we presented a lower bound that applies to algorithms that may output randomized policies after a random stopping time, which is more general than what has been previously shown. Unlike the prior lower bound constructions with parallel MAB instances, we designed a class of MDPs where each of them has stage-dependent transitions and behaves as *single* bandit instance with $\Theta(HSA)$ arms. In Theorem 3.9, we proved that in this class there exists an MDP for which the expected number of samples needed to identify an ε-optimal policy with probability $1-\delta$ is at least $\Omega\left(\frac{SAH^3}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right)$. Our construction avoids unnecessary assumptions without which prior analyses would not work.

Regret lower bounds In the average-reward setting, Jaksch et al. [JOA10] prove a regret lower bound of $\Omega(\sqrt{DSAT})$ where D is the diameter of the MDP and T is the total number of actions taken in the environment. In the finite-horizon setting, the total number of actions taken is HT, where T is now the number of episodes, and H is roughly the equivalent of the diameter D. [1] Hence, intuitively, the lower bound of Jaksch et al. [JOA10] should be translated to $\Omega(\sqrt{H^2SAT})$ for finite-horizon MDPs after T episodes. Yet, to the best of our knowledge, a precise proof of this claim had not been given previously in the literature. The proof of Jaksch et al. [JOA10] relies on building a set of hard MDPs with "bad" states (with zero reward) and "good" states (with reward 1), and can be adapted to finite-horizon MDPs by making the good states absorbing. However, this construction does not include MDPs whose transitions and rewards are allowed to change at every stage h, that is, that are time-inhomogeneous. In the

[1] The diameter D is the minimum average time to go from one state to another. In a finite-horizon MDP, if the agent can come back to the same initial state s_1 after H steps, the average time between any pair of states is bounded by $2H$, if we restrict the state set to the states that are reachable from s_1 in H steps.

case of time-inhomogeneous MDPs, Jin et al. [Jin+18] claim that the lower bound becomes $\Omega(\sqrt{H^3SAT})$, by using the construction of Jaksch et al. [JOA10] and a mixing-time argument, but they do not provide a complete proof. In Theorem 3.8, we provided a detailed proof of their statement, by relying on the same class of hard MDPs given for our sample complexity lower bound.

Algorithms matching the lower bounds Table 3.1 shows some of the algorithms whose upper bounds match the lower bounds[2] presented in this chapter for the regret, BPI, and PAC settings, both for time-homogeneous and time-inhomogeneous MDPs.[3,4] The last column indicates whether the algorithm was analyzed only for time-inhomogeneous MDPs.

Algorithm	Setting	time-inhomogeneous only
`UCBVI` [AOM17]	Regret	No[3]
Q-Learning+UCB [Jin+18]	Regret	Yes
`BPI-UCBVI` [Mé+21a]	BPI	Yes
`ORLC` [Dan+19][4]	BPI, PAC	No[3]

Table 3.1 – Algorithms matching the lower bounds in different settings.[2]

Further reading Algorithms relying on optimistic approaches, such as `UCBVI`, have been analyzed in different settings, for instance, `MBIE` by Strehl and Littman [SL08] for PAC-MDP, `UCRL` by Jaksch et al. [JOA10] for the regret in the average-reward setting, `BPI-UCRL` by Kaufmann et al. [Kau+21] and `BPI-UCBVI` by Ménard et al. [Mé+21a] for BPI. Although both the PAC-MDP and the regret frameworks measure how well an agent balances exploration and exploitation, Dann et al. [DLB17] show that there is no direct equivalence between the two criteria. They propose a new framework, called Uniform-PAC, that implies PAC-MDP and high-probability regret bounds, and introduce a new algorithm called `UBEV` for which they prove a Uniform-PAC bound, and also relies on an optimistic construction. Besides optimistic approaches for regret minimization, there are also algorithms based on *Thompson Sampling*, which is based on sampling in each episode an MDP from a posterior distribution, given a prior distribution, and executing the optimal policy in the sampled MDP [Str00; ORVR13].

[2] Up to constants and logarithmic terms, and assuming that either T is large enough (for the regret) or ε is small enough (for the sample complexity).

[3] `UCBVI` and `ORLC` have been proposed for MDPs with time-homogeneous transitions, but they can readily be used for time-inhomogeneous MDPs by viewing them as MDPs with HS states and h-independent transitions.

[4] Dann et al. [Dan+19] analyze the `ORLC` algorithm in a slightly different setting, proving that it outputs "Individual POlicy Certificates" (IPOC). `ORLC` can be converted to an (ε, δ)-PAC algorithm for BPI by setting the stopping rule to be the first time the optimality certificate is smaller than ε. Sample complexity guarantees for both BPI and PAC-MDP setting can be deduced from their analysis.

Chapter 4

A Kernel-Based Approach to Exploration in Continuous MDPs

In this chapter, we study a kernel-based algorithm for exploration-exploitation in large or continuous MDPs, called `Kernel-UCBVI`. A kernel function measures the similarity between any two state-action pairs, and the key idea of `Kernel-UCBVI` is that once a state-action pair (s, a) is visited, we also reduce the uncertainty about all the pairs (s', a') that are similar to (s, a), which allows the agent to explore large state-action spaces efficiently. We prove a regret bound for `Kernel-UCBVI` that depends on the *covering dimension* of the state-action space, instead of its cardinality. By extending `Kernel-UCBVI` to use time-dependent kernels, we introduce an algorithm called `KeRNS` that is able to handle *non-stationary environments*, where the agent may interact with a different MDP in each episode. Finally, we propose approximate versions of `Kernel-UCBVI` and `KeRNS` to reduce their computational complexity and analyze the impact of such approximations on the regret.

This chapter is based on the papers [Dom+21d; Dom+21c] about regret minimization with kernel-based reinforcement, both in stationary and non-stationary environments.

Contents

4.1 Kernel-Based Reinforcement Learning for Exploration **58**
4.2 Regret Analysis of `Kernel-UCBVI` **61**
4.3 Comparison to Lower Bounds & Related Work **66**
4.4 `KeRNS`: An Extension of `Kernel-UCBVI` to Non-Stationary MDPs **67**
4.5 Regret Analysis of `KeRNS` **70**
4.6 Reducing the Computational Complexity **73**
4.7 Experiments **77**
4.8 Discussion and Bibliographical Remarks **81**

4.1 Kernel-Based Reinforcement Learning for Exploration

Kernel-Based Reinforcement Learning (KBRL) was introduced by Ormoneit and Sen [OS02], in the setting where a generative model of the MDP is available. KBRL works by sampling independent transitions from the generative model, and uses a kernel function to build an approximate model of the transitions and rewards. Then, it computes a policy by running value iteration with the kernel-based model. Hence, it can be seen as a generalization to continuous MDPs of the `MBQVI` algorithm studied in Chapter 2. The main advantages of KBRL, as proposed by Ormoneit and Sen [OS02], are: (i) its stability, in the sense that it converges to a unique solution of approximate Bellman equations in the discounted-reward setting; (ii) its statistical consistency, as its output converges in probability to the optimal value function as the number of sampled transitions goes to infinity [OS02]; (iii) its flexibility in handling the bias-variance trade-off in RL, which can be controlled via kernel design. In this section, we propose an extension of KBRL to the *online* setting, where exploration is necessary. This extension is inspired by `UCBVI` and relies on a generalization of `UCBVI`'s exploration bonuses to continuous state-actions, and we name the resulting algorithm `Kernel-UCBVI`. In the following section, we analyze the regret of `Kernel-UCBVI`, assuming that the MDP satisfies certain regularity conditions and that the kernel function is designed based on a metric on the state-action space.

Notation & Assumptions We denote by $(s_h^t, a_h^t, s_{h+1}^t, \widetilde{r}_h^t)$ the state, the action, the next state and the reward observed by the algorithm at stage h of episode t. We do not assume that the reward function is known, but we assume that $\widetilde{r}_h^t \in [0,1]$ almost surely. If μ and $p(\cdot|s,a)$ are measures for any (s,a) and f is an arbitrary function, we denote $\mu f := \int f(y)\mathrm{d}\mu(y)$ and $pf(s,a) := \int f(y)\mathrm{d}p(y|s,a)$.

Algorithm 4.1: `Kernel-UCBVI`

for *episode* $t \in \{1,\dots,T\}$ **do**
  get initial state s_1^t
  # compute optimistic Q-functions
  compute $(Q_h^t)_{h\in[H]}$ according to Algorithm 4.2
  for *stage* $h \in \{1,\dots,H\}$ **do**
    # select action
    $a_h^t \leftarrow \mathrm{argmax}_a\, Q_h^t(s_h^t, a)$
    # execute action
    $\widetilde{r}_h^t, s_{h+1}^t \leftarrow \texttt{OnlineModel}_{t,h}(a_h^t)$
    # update model
    compute $\widehat{r}_h^{t+1}$ and $\widehat{p}_h^{t+1}$ using Equation (4.2)

`Kernel-UCBVI` (Algorithm 4.1) has the same algorithmic structure as `UCBVI`, except that it does not assume that the MDP is finite. We now describe how the Q-functions Q_h^t are computed.

Let $\Gamma : (\mathcal{S} \times \mathcal{A})^2 \to [0, 1]$ be a *kernel function*, where $\Gamma(u, v)$ represents the similarity between two state action pairs $u, v \in \mathcal{S} \times \mathcal{A}$. For any $(s, a) \in \mathcal{S} \times \mathcal{A}$, we define weights $w_h^i(s, a)$ and normalized weights $\widetilde{w}_h^{t,i}(s, a)$ for any (t, h) and for $i \in [t-1]$ as

$$w_h^i(s,a) := \Gamma\left((s,a),(s_h^i,a_h^i)\right) \quad \text{and} \quad \widetilde{w}_h^{t,i}(s,a) := \frac{w_h^i(s,a)}{\mathbf{C}_h^t(s,a)}, \tag{4.1}$$

where

$$\mathbf{C}_h^t(s,a) := \beta + \sum_{i=1}^{t-1} w_h^i(s,a)$$

and where $\beta > 0$ is a regularization parameter.

Before each episode t, `Kernel-UCBVI` uses those weights to build kernel-based estimators $\widehat{r}_h^t$ and $\widehat{p}_h^t$ of the reward function and of the transitions at time (t, h), respectively:

$$\widehat{r}_h^t(s,a) := \sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a)\widetilde{r}_h^i, \quad \widehat{p}_h^t(z|s,a) := \sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a)\delta_{s_{h+1}^i}(z), \tag{4.2}$$

where δ_s is the Dirac measure at $s \in \mathcal{S}$.

The weights $w_h^t(s, a)$ measure the influence that the transitions and rewards observed at time (t, h) will have on the estimators for the state-action pair (s, a). Their sum, $\mathbf{C}_h^t(s, a)$, is a generalization of the number of visits to (s, a). Indeed, if the MDP is finite, we can define a kernel as

$$\Gamma\left((s,a),(s',a')\right) = \mathbb{1}\left\{(s,a) = (s',a')\right\}$$

so that $\mathbf{C}_h^t(s, a) = \beta + n_h^t(s, a)$, where $n_h^t(s, a) = \sum_{i=1}^{t-1} \mathbb{1}\left\{(s, a) = (s_h^i, a_h^i)\right\}$. That is, $\mathbf{C}_h^t(s, a)$ is equal to β plus the number of visits to (s, a) at stage h before episode t. Recall that the exploration bonuses used by `UCBVI` are proportional to $H/\sqrt{n_h^t(s, a)}$, where H is the horizon, *i.e.*, the length of each episode. Hence, as a generalization of such bonuses to continuous MDPs, we propose the following

$$\mathtt{b}_h^t(s,a) = \frac{\kappa_1 H}{\sqrt{\mathbf{C}_h^t(s,a)}} + \frac{\kappa_2 \beta H}{\mathbf{C}_h^t(s,a)} + \kappa_3, \tag{4.3}$$

where κ_1, κ_2, and κ_3 are constants to be defined later. The first term in the sum is analogous to the bonus of `UCBVI`, and measures the uncertainty in the model estimation. The second term takes into account the bias introduced by the regularization constant β, and the third term κ_3 represents an extra bias term introduced by the kernel. `Kernel-UCBVI` defines the Q-functions

$Q_h^t(s,a)$ for any $(s,a) \in \mathcal{S} \times \mathcal{A}$ via backward induction and interpolation as follows:

$$\textbf{(backward induction)} \quad \widetilde{Q}_h^t(s,a) = \widehat{r}_h^t(s,a) + \widehat{p}_h^t V_{h+1}^t(s,a) + \mathtt{b}_h^t(s,a)$$
$$\textbf{(interpolation)} \quad Q_h^t(s,a) = \Lambda\left(s, a, \left\{\widetilde{Q}_h^t(s_h^i, a_h^i)\right\}_{i\in[t-1]}\right)$$

where V_h is defined as $V_h^t(s) = \min\left(H-h+1, \max_a Q_h^t(s,a)\right)$, and Λ is an interpolation function. In practice, we can skip the interpolation step and define $Q_h^t(s,a) = \widetilde{Q}_h^t(s,a)$ for any (s,a). However, to derive regret bounds, we need to define Λ as a linear interpolation of the values $\left\{\widetilde{Q}_h^t(x_h^i, a_h^i)\right\}_{i\in[t-1]}$ to control the complexity of the function class to which Q_h^t belongs.

Algorithm 4.2: Kernel Backward Induction

input: transitions $(s_h^i, a_h^i, s_{h+1}^i, \widetilde{r}_h^i)_{i=1}^{t-1}$ for all $h \in [H]$.
initialization: $V_h^t(s) \leftarrow 0$ for all $s \in \mathcal{S}$.
for $h = H, \dots, 1$ **do**
  for $i = 1, \dots, t-1$ **do**
    # Using weights in Equation (4.1) and bonuses in Equation (4.3), compute:
    $\widetilde{Q}_h^t(s_h^i, a_h^i) \leftarrow \sum_{j=1}^{t-1} \widetilde{w}_h^{t,j}(s_h^i, a_h^i)\left(\widetilde{r}_h^j + V_{h+1}^t(s_{h+1}^j)\right) + \mathtt{b}_h^t(s_h^i, a_h^i)$
  # Interpolated Q-function: defined, but not computed, for all $(s,a) \in \mathcal{S} \times \mathcal{A}$
  $Q_h^t(s,a) = \Lambda\left(s, a, \left\{\widetilde{Q}_h^t(s_h^i, a_h^i)\right\}_{i\in[t-1]}\right)$
  # Compute V_h^t for the observed states
  for $i = 1, \dots, t-1$ **do**
    $V_h^t(s_h^i) = \min(H-h+1, \max_a Q_h^t(s_h^i, a))$
return: $(Q_h^t)_{h\in[H]}$

Although `Kernel-UCBVI` defines $Q_h^t(s,a)$ for any (s,a), backward induction can be run in finite time, as detailed in Algorithm 4.2. Intuitively, backward induction with a kernel-based model in a continuous MDP is analogous to that in a finite MDP where the state set is composed of all previously observed states $s_{h'}^{t'}$ for $h' \in [H]$ and $t' \in [t-1]$. Consequently, one drawback of `Kernel-UCBVI` is that its runtime increases in each episode. In Section 4.6 we discuss approximation methods to decrease the runtime of the algorithm.

Time-homogeneous MDPs As mentioned in Section 3.6, we consider time-inhomogeneous MDPs, where the rewards and transitions depend on h, for mathematical convenience. The algorithm can be easily generalized to time-homogeneous MDPs by estimating the model as

$$\widehat{r}^t(s,a) := \frac{1}{\mathbf{C}^t(s,a)} \sum_{i=1}^{t-1}\sum_{h=1}^{H} w_h^i(s,a)\widetilde{r}_h^i, \quad \widehat{p}^t(z|s,a) := \frac{1}{\mathbf{C}^t(s,a)} \sum_{i=1}^{t-1}\sum_{h=1}^{H} w_h^i(s,a)\delta_{s_{h+1}^i}(z), \quad (4.4)$$

where $\mathbf{C}^t(s,a) := \beta + \sum_{i=1}^{t-1}\sum_{h=1}^{H} w_h^i(s,a)$, and by defining the bonuses as

$$\mathsf{b}_h^t(s,a) = \frac{\kappa_1 H}{\sqrt{\mathbf{C}^t(s,a)}} + \frac{\kappa_2 \beta H}{\mathbf{C}^t(s,a)} + \kappa_3. \tag{4.5}$$

4.2 Regret Analysis of Kernel-UCBVI

In this section, we provide a regret upper bound for Kernel-UCBVI. For that, we first need regularity assumptions on the MDP and on the kernel function, which are stated below.[1] Intuitively, our assumptions require similar state-action pairs to have similar transitions and rewards, which allow the algorithm to generalize and explore continuous MDPs.

Assumption 4.1. *The state-action space $\mathcal{S}\times\mathcal{A}$ is equipped with a metric $\rho : (\mathcal{S}\times\mathcal{A})^2 \to \mathbb{R}_+$, which is known. Also, we assume that there exists a metric $\rho_{\mathcal{S}}$ on $\mathcal{S}$ such that, for all (s,s',a), $\rho\left[(s,a),(s',a)\right] \le \rho_{\mathcal{S}}\left(s,s'\right)$.*

Assumption 4.2. *The (mean) reward functions are L_{r}-Lipschitz and the transition kernels are L_{p}-Lipschitz with respect to the 1-Wasserstein distance: $\forall(s,a,s',a')$ and $\forall h\in[H]$,*

$$\begin{aligned}\left|r_h(s,a) - r_h(s',a')\right| &\le L_{\mathrm{r}}\rho\left[(s,a),(s',a')\right], \text{ and}\\ \mathbb{W}_1\left(p_h(\cdot|s,a), p_h(\cdot|s',a')\right) &\le L_{\mathrm{p}}\rho\left[(s,a),(s',a')\right]\end{aligned}$$

where, for two measures μ and ν, we have $\mathbb{W}_1(\mu,\nu) := \sup_{f:\mathrm{Lip}(f)\le 1}\int_{\mathcal{S}} f(y)(\mathrm{d}\mu(y) - \mathrm{d}\nu(y))$ and where, for any Lipschitz function $f : \mathcal{S}\to\mathbb{R}$ with respect to $\rho_{\mathcal{S}}$, $\mathrm{Lip}(f)$ denotes its Lipschitz constant.

Assumption 4.3. *For any h, the optimal Q-function Q_h^* is L-Lipschitz with respect to ρ. Assumptions 4.1 and 4.2 imply that $L \le \sum_{h=1}^{H} L_{\mathrm{r}} {L_{\mathrm{p}}}^{H-h}$ (Lemma C.22 in the Appendix).*

For the regret analysis, we require the similarity function to be defined through a *base kernel function* and the distance ρ. Let $\sigma > 0$ be a kernel parameter. We assume that we have access to a base kernel function $\overline{\Gamma} : \mathbb{R}_+ \to [0,1]$ such, for any $u,v\in\mathcal{S}\times\mathcal{A}$, the kernel Γ is defined as

$$\Gamma(u,v) = \overline{\Gamma}\left(\frac{\rho\left[u,v\right]}{\sigma}\right).$$

[1] Regarding Assumption 4.1, if $(\mathcal{A},\rho_{\mathcal{A}})$ is also a metric space, we can take $\rho\left[(x,a),(x',a')\right] = \rho_{\mathcal{S}}\left(x,x'\right) + \rho_{\mathcal{A}}\left(a,a'\right)$, for instance. See Section 2.3 of [SBY19] for more examples and a discussion.

Assumption 4.4 (kernel properties). *We assume that $z \mapsto \overline{\Gamma}(z)$ is non-increasing and that $\overline{\Gamma}(4) > 0$. Additionally, we assume that there exists positive constants C_1, C_2, such that*

$$\textbf{(1) Fast decay:} \quad \forall z,\ \overline{\Gamma}(z) \leq C_1 \exp\left(-z^2/2\right),$$
$$\textbf{(2) Lipschitzness:} \quad \forall (y,z),\ \left|\overline{\Gamma}(y) - \overline{\Gamma}(z)\right| \leq C_2 \left|y - z\right|.$$

Condition **(1)** ensures that the bias due to kernel smoothing remains bounded by $\widetilde{\mathcal{O}}(\sigma)$; and **(2)** provides smoothness conditions that are needed to construct concentration inequalities for the rewards and transitions. The requirement $\overline{\Gamma}(4) > 0$ is mostly technical: it is used to ensure that $\mathbf{C}_h^t(s,a)$ is not too small in a $4\,\sigma$-neighborhood of (s,a).

Example 4.5. *As a simple example of an MDP satisfying assumptions 4.1 and 4.2, consider an MDP $\mathcal{M}$ with finite action set $\mathcal{A}$, a compact state space $\mathcal{S} \subset \mathbb{R}^d$ and deterministic transitions $y = f(x,a)$,* i.e., *$p_h(y|x,a) = \delta_{f(x,a)}(y)$. Let $\rho_{\mathcal{S}}$ be the Euclidean distance on $\mathbb{R}^d$ and $\rho_{\mathcal{A}}(a,a') = 0$ if $a = a'$ and $+\infty$ otherwise. Then, if for all $a \in \mathcal{A}$, $x \mapsto r_h(x,a)$ and $x \mapsto f(x,a)$ are Lipschitz continuous, then $\mathcal{M}$ satisfies our assumptions.*

Example 4.6. *As examples of kernels $\overline{\Gamma}$ satisfying Assumption 4.4, we have $\overline{\Gamma}(z) = \exp(-z^q/2)$ for $q \geq 2$, $\overline{\Gamma}(z) = \max(0, 1 - z/q)$ for $q > 4$, among other kernels that are Lipschitz continuous and have bounded support.*

Interpolation For the regret analysis, we consider the following interpolation function:

$$Q_h^t(s,a) = \Lambda\left(s, a, \left\{\widetilde{Q}_h^t(s_h^i, a_h^i)\right\}_{i\in[t-1]}\right) := \min_{i\in[t-1]}\left(\widetilde{Q}_h^t(s_h^i,a_h^i) + L\rho\left[(s,a),(s_h^i,a_h^i)\right]\right). \quad (4.6)$$

This ensures that the functions $(s,a) \mapsto Q_h^t(s,a)$ are L-Lipschitz for all (t,h), and allows us to prove the concentration inequalities on which the analysis relies.

Covering numbers & covering dimension The regret bounds for `Kernel-UCBVI` feature the σ-covering number and the covering dimension of the state-action space, which we now define. Let $(\mathcal{U},\rho)$ be a metric space. For any $u \in \mathcal{U}$, let $\mathcal{B}(u,\sigma) = \{v \in \mathcal{U} : \rho(u,v) \leq \sigma\}$. We say that a set $\mathcal{C}_\sigma$ is a σ-covering of $(\mathcal{U},\rho)$ if $\mathcal{U} \subset \cup_{u\in\mathcal{C}_\sigma}\mathcal{B}(u,\sigma)$. The σ-covering number of $(\mathcal{U},\rho)$ is

$$\mathcal{N}(\sigma,\mathcal{U},\rho) := \min\left\{|\mathcal{C}_\sigma| : \text{ is a } \sigma\text{-covering of } (\mathcal{U},\rho)\right\}.$$

That is, $\mathcal{N}(\sigma,\mathcal{U},\rho)$ is the minimum number of σ-radius balls required to cover the entire space. The covering dimension of $(\mathcal{U},\rho)$ is then defined as the smallest number d such that its σ-

covering number is proportional to σ^{-d}. For instance, the covering number of a ball in $\mathbb{R}^d$ equipped with the Euclidean distance is $\mathcal{O}(\sigma^{-d})$ and its covering dimension is d.[2]

We denote by $|\mathcal{C}_\sigma|$ and $|\mathcal{C}'_\sigma|$ the σ-covering numbers of $(\mathcal{S}\times\mathcal{A},\rho)$ and $(\mathcal{S},\rho_{\mathcal{S}})$, respectively, and by d_1 and d_2 their respective covering dimensions, where σ is the kernel parameter. Also, we define $d = \max(d_1, d_2)$. Theorem 4.7 provides a regret bound for Kernel-UCBVI under the assumptions above. We use the notation below when omitting constant and logarithmic factors:

$$A \lesssim B \iff A \leq B \times \text{polynomial}\left(d_1, d_2, \log(T), \log(1/\delta), \beta, 1/\beta, L_{\mathrm{r}}, L_{\mathrm{p}}, L\right).$$

Theorem 4.7. *With probability at least $1-\delta$, the regret of Kernel-UCBVI satisfies*

$$\mathcal{R}_T \lesssim H^2\sqrt{|\mathcal{C}_\sigma|\,T} + LHT\sigma + H^3\,|\mathcal{C}_\sigma|\,|\mathcal{C}'_\sigma|\,,$$

if the constants $(\kappa_i)_{i=1}^3$ defining the bonuses (4.3) are taken according to Definition C.1.

In the special case of finite MDPs, we can use the metric $\rho\left[(s,a),(s',a')\right] = 0$ if $(s,a)=(s',a')$ and ∞ otherwise. This allows us to take $\sigma = 0$ and results in $|\mathcal{C}_\sigma| = SA$ and $|\mathcal{C}'_\sigma| = S$, where $S = |\mathcal{S}|$ and $A = |\mathcal{A}|$. Hence, the regret of Kernel-UCBVI is equivalent in this case to that of UCBVI given in Theorem 3.14. Also, notice that $d = 0$ in finite MDPs. Corollary 4.8 provides a regret bound featuring the covering dimension d, in the case where $d > 0$.

Corollary 4.8. *Recall that d_1 and d_2 are the covering dimensions of the state-action and the state spaces, respectively, and that $d = \max(d_1, d_2)$. Assume that $d > 0$. By taking $\sigma = T^{-1/(2d+1)}$, the regret of Kernel-UCBVI satisfies, with probability at least $1-\delta$,*

$$\mathcal{R}_T \lesssim H^3 T^{\frac{2d}{2d+1}}.$$

Proof. The proof is an immediate consequence of Theorem 4.7 by noticing that $|\mathcal{C}_\sigma| \lesssim \sigma^{-d_1} \leq \sigma^{-d}$ and $|\mathcal{C}'_\sigma| \lesssim \sigma^{-d_2} \leq \sigma^{-d}$. □

Now, we provide a proof sketch of Theorem 4.7. The full proof is given in Appendix C.2. Similarly to UCBVI, the proof is split into three parts: (i) deriving confidence intervals for the kernel-based transitions estimators $\widehat{p}_h^t$; (ii) proving that the algorithm is optimistic, *i.e.*, that $V_h^t(s) \geq V_h^*(s)$ for any (s,t,h) on a high-probability event $\mathcal{G}$; (iii) proving an upper bound on the regret by using the fact that $\mathcal{R}_T = \sum_t \left(V_1^*(s_1^t) - V_1^{\boldsymbol{\pi}^t}(s_1^t)\right) \leq \sum_t \left(V_1^t(s_1^t) - V_1^{\boldsymbol{\pi}^t}(s_1^t)\right)$ on the event $\mathcal{G}$.

[2] For more details about covering numbers and covering dimension, see Section 3 of Kleinberg et al. [KSU19] and Section 2.2 of Sinclair et al. [SBY19].

4.2.1 Concentration

We focus on concentration inequalities for the transition kernels, as those for the rewards are similar. Since $\widehat{p}_h^t(\cdot|s,a)$ are weighted sums of Dirac measures, we cannot bound the distance between $p_h(\cdot|s,a)$ and $\widehat{p}_h^t(\cdot|s,a)$ directly. Instead, for V_{h+1}^*, the optimal value function at step $h+1$, we bound the difference:

$$|(\widehat{p}_h^t - p_h)V_{h+1}^*(s,a)| = \left|\sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a)V_{h+1}^*(s_{h+1}^i) - p_h V_{h+1}^*(s,a)\right|$$
$$\leq \left|\sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a)V_{h+1}^*(s_{h+1}^i) - p_h V_{h+1}^*(s_h^i,a_h^i)\right| + L_{\mathrm{p}} L \sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a)\rho\left[(s,a),(s_h^i,a_h^i)\right] + \frac{\beta H}{\mathbf{C}_h^t(s,a)}.$$

The first term above is a weighted sum of a martingale difference sequence. To control it, we use a Hoeffding-type inequality (Lemma C.2) that applies to weighted sums with random weights. The second term is a bias term that results from the fact that V_{h+1}^* is L-Lipschitz and that the transition kernel is L_{p}-Lipschitz, and this term is shown to be proportional to σ under Assumption 4.4 (Lemma C.20). The third term is the bias introduced by the regularization parameter β. Hence, for a fixed state-action pair (s,a), we show that, with high probability,

$$|(\widehat{p}_h^t - p_h)V_{h+1}^*(s,a)| \leq \frac{\kappa_1 H}{\sqrt{\mathbf{C}_h^t(s,a)}} + \frac{\kappa_2 \beta H}{\mathbf{C}_h^t(s,a)} + \kappa_3$$

for an appropriate choice of κ_1, κ_2, and κ_3. Then, we extend this bound to all (s,a) by leveraging the continuity of all terms involving (s,a) and a covering argument. This continuity is a consequence of kernel smoothing. Also, we use a Bernstein-type concentration inequality (Lemma C.3) that allows us to control the deviations of $(\widehat{p}_h^t - p_h)f(s,a)$ uniformly over a class of bounded Lipschitz functions f. This is similar to Lemma 3.16, which allowed us to gain a $\sqrt{S}$ factor in the regret bound of `UCBVI`, and here it allows us to gain a $\sqrt{|\mathcal{C}'_\sigma|}$ factor for `Kernel-UCBVI`. We define a favorable event $\mathcal{G}$, which has probability at least $1 - \delta/2$, on which our concentration inequalities hold.

4.2.2 Optimism

To prove that V_h^t is an upper bound on V_h^*, we proceed by induction and use the Q-functions. When $h = H+1$, we have $Q_{H+1}^t = Q_{H+1}^* = 0$, by definition. Assuming that $Q_{h+1}^t(s,a) \geq Q_{h+1}^*(s,a)$ for all (s,a), we have $V_{h+1}^t(s) \geq V_{h+1}^*(s)$ for all s. Then, the bonuses are defined so that $\widetilde{Q}_h^t(s,a) \geq Q_h^*(s,a)$ for all (s,a) on the event $\mathcal{G}$.

In particular, $\widetilde{Q}_h^t(s_h^i,a_h^i) \geq Q_h^*(s_h^i,a_h^i)$ for all $i \in [t-1]$, which gives us

$$\widetilde{Q}_h^t(s_h^i,a_h^i) + L\rho\left[(s,a),(s_h^i,a_h^i)\right] \geq Q_h^*(s_h^i,a_h^i) + L\rho\left[(s,a),(s_h^i,a_h^i)\right] \geq Q_h^*(s,a)$$

for all $i \in [t-1]$, since Q_h^* is L-Lipschitz. It follows from the definition of the interpolation function in Equation (4.6) that $Q_h^t(s,a) \geq Q_h^*(s,a)$ for all (s,a), which implies that, for all s, $V_h^t(s) \geq V_h^*(s)$ on $\mathcal{G}$.

4.2.3 Bounding the Regret

Let $\boldsymbol{\pi}^t$ be the policy executed by Kernel-UCBVI in episode t, and let $\delta_h^t := V_h^t(s_h^t) - V_h^{\boldsymbol{\pi}^t}(s_h^t)$. On the event $\mathcal{G}$, we have $V_h^t \geq V_h^*$, which implies that $\mathcal{R}_T \leq \sum_{t=1}^T \delta_1^t$. Let $(\widetilde{s}_h^t, \widetilde{a}_h^t)$ be the state-action pair that is the closest to (s_h^t, a_h^t) among the transitions observed before episode t , that is

$$(\widetilde{s}_h^t, \widetilde{a}_h^t) := \operatorname*{argmin}_{(s_h^i, a_h^i): i<t} \rho\left[(s_h^t, a_h^t), (s_h^i, a_h^i)\right],$$

and we define $\rho_h^t := \rho\left[(s_h^t, a_h^t), (\widetilde{s}_h^t, \widetilde{a}_h^t)\right]$. We bound δ_h^t using the following decomposition

$$\begin{aligned}
\delta_h^t &\leq Q_h^t(s_h^t, a_h^t) - Q_h^{\boldsymbol{\pi}^t}(s_h^t, a_h^t) \leq \widetilde{Q}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) - Q_h^{\boldsymbol{\pi}^t}(s_h^t, a_h^t) + L\rho_h^t \\
&\leq 2\, \mathtt{b}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) + (L + L_{\mathrm{p}} L + L_{\mathrm{r}})\rho_h^t \\
&\quad + \underbrace{(\widehat{p}_h^t - p_h) V_{h+1}^*(\widetilde{s}_h^t, \widetilde{a}_h^t)}_{(\mathbf{A})} + \underbrace{p_h(V_{h+1}^t - V_{h+1}^{\boldsymbol{\pi}^t})(s_h^t, a_h^t)}_{(\mathbf{B})} + \underbrace{(\widehat{p}_h^t - p_h)(V_{h+1}^t - V_{h+1}^*)(\widetilde{s}_h^t, \widetilde{a}_h^t)}_{(\mathbf{C})}.
\end{aligned}$$

The term $(\mathbf{A})$ is shown to be smaller than $\mathtt{b}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)$, by definition of the bonus. The term $(\mathbf{B})$ can be written as δ_{h+1}^t plus a term ξ_{h+1}^t, where $(\xi_{h+1}^t)_{t,h}$ is a martingale difference sequence. Using the fact that $V_{h+1}^t - V_{h+1}^*$ is $2L$-Lipschitz and the uniform deviation inequalities that hold on the event $\mathcal{G}$, we prove that

$$(\mathbf{C}) \lesssim \frac{1}{H}(\delta_{h+1}^t + \xi_{h+1}^t) + \frac{H^2\,|\mathcal{C}'_\sigma|}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)} + L\rho_h^t + L\sigma.$$

When $\rho_h^t > 2\sigma$, we bound δ_h^t by H and we verify that $\sum_{h=1}^H \sum_{t=1}^T \mathbb{1}\{\rho_h^t > 2\sigma\} \leq H^2\,|\mathcal{C}_\sigma|$ by a pigeonhole argument. Hence, we can focus on the case where $\rho_h^t \leq 2\sigma$, and add $H^2\,|\mathcal{C}_\sigma|$ to the regret bound to take into account the steps (t,h) where $\rho_h^t > 2\sigma$. The sum $\sum_{t,h} \xi_{h+1}^t$ is bounded by $\widetilde{\mathcal{O}}\left(H^{3/2}\sqrt{T}\right)$ by Hoeffding-Azuma's inequality on an event $\mathcal{G}'$ of probability larger than $1 - \delta/2$. Now, we focus on the case where $\rho_h^t \leq 2\sigma$ and omit the terms involving ξ_{h+1}^t. Using the definition of the bonus, we obtain

$$\delta_h^t \lesssim \left(1 + \frac{1}{H}\right)\delta_{h+1}^t + \frac{H}{\sqrt{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}} + \frac{H^2\,|\mathcal{C}'_\sigma|}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)} + L\sigma.$$

Using the fact that $(1 + 1/H)^H \leq e$, we have, on the event $\mathcal{G} \cap \mathcal{G}'$ of probability at least $1 - \delta$,

$$\mathcal{R}_T \lesssim \sum_{h=1}^H \sum_{t=1}^T \left(\frac{H}{\sqrt{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}} + \frac{H^2\,|\mathcal{C}'_\sigma|}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}\right) + LHT\sigma.$$

Finally, we show that

$$\sum_{h=1}^{H}\sum_{t=1}^{T}\frac{1}{\sqrt{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}} \lesssim H\sqrt{|\mathcal{C}_\sigma|\,T} \quad \text{and} \quad \sum_{h=1}^{H}\sum_{t=1}^{T}\frac{1}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)} \lesssim H\,|\mathcal{C}_\sigma|\log T,$$

which gives us the final bound on the event $\mathcal{G}\cap\mathcal{G}'$:

$$\mathcal{R}_T \lesssim H^2\sqrt{|\mathcal{C}_\sigma|\,T} + LHT\sigma + H^3\,|\mathcal{C}_\sigma|\,|\mathcal{C}'_\sigma| + H^2\,|\mathcal{C}_\sigma|\,,$$

where the term $H^2\,|\mathcal{C}_\sigma|$ takes into account the time steps (t,h) such that $\rho_h^t > 2\sigma$.

4.3 Comparison to Lower Bounds & Related Work

To the best of our knowledge, the regret bound we proved for `Kernel-UCBVI` is the first regret bound for kernel-based RL using smoothing kernels, and we present below further discussions on this result, regarding lower bounds and related work.

Comparison to lower bound for Lipschitz MDPs In terms of the number of episodes T and the dimension d, the lower bound for Lipschitz MDPs is of order $T^{\frac{d+1}{d+2}}$, which is a consequence of the lower bounds for Lipschitz multi-armed bandits [Bub+11; Sli14]. In terms of H, the optimal dependence can be conjectured to be $H^{\frac{3}{2}}$, which is the case for finite MDPs, as we saw in Chapter 3 (Theorem 3.8). For $d=1$, our bound for `Kernel-UCBVI` has an optimal dependence on T, leading to a regret of order $\widetilde{\mathcal{O}}\left(H^3T^{\frac{2}{3}}\right)$.

Comparison to other upper bounds for Lipschitz MDPs The best available upper bound in this setting, in terms of T and d, is $\widetilde{\mathcal{O}}\left(H^{\frac{5}{2}}T^{\frac{d+1}{d+2}}\right)$, which is achieved by model-free algorithms performing either uniform or adaptive discretization of the state-action space [SS19; SBY19; TTB20].

Relevance of kernel-based algorithms Although our upper bound does not match the lower bound for Lipschitz MDPs, kernel-based RL (KBRL) can be a very useful tool in practice to handle the bias-variance trade-off in RL. It allows us to easily provide expert knowledge to the algorithm through kernel design, which can be seen as introducing more bias to reduce the variance of the algorithm and, consequently, improve the learning speed. Furthermore, kernels can be defined on arbitrary types of objects, such as graphs, sets, strings (*i.e.*, sequences of symbols) etc., which might make KBRL applicable to a wider range of tasks than discretization-based algorithms. In addition, Badia et al. [Bad+20b] have shown that kernel-based exploration bonuses similar to the ones derived for `Kernel-UCBVI` can improve exploration in Atari games.

Model-free versus model-based approaches An interesting observation comes from the comparison between Kernel-UCBVI and model-free approaches in continuous MDPs [SS19; SBY19; TTB20]. These algorithms are based on optimistic Q-learning [Jin+18], to which we refer as OptQL, and achieve a regret of order $\widetilde{\mathcal{O}}\left(H^{\frac{5}{2}}T^{\frac{d+1}{d+2}}\right)$, which has an optimal dependence on T and d. While we achieve the same $\widetilde{\mathcal{O}}\left(T^{\frac{2}{3}}\right)$ regret when $d = 1$, our bound is slightly worse for $d > 1$. To understand this gap, it is useful to look at the regret bound for finite MDPs. Since our algorithm is inspired by UCBVI [AOM17] with Chernoff-Hoeffding bonus, we compare it to OptQL, which is used by [SS19; SBY19; TTB20], with the same kind of exploration bonus. Consider a time-inhomogeneous MDP (where the transitions depend on h) with S states and A actions. UCBVI has a regret bound of $\widetilde{\mathcal{O}}\left(\sqrt{H^4SAT} + H^3S^2A\right)$ while OptQL has $\widetilde{\mathcal{O}}\left(\sqrt{H^5SAT} + H^2SA\right)$. As we can see, OptQL is a $\sqrt{H}$-factor worse than UCBVI when comparing the first-order term (*i.e.*, the term scaling with $\sqrt{T}$), but it is HS times better in the second-order term (*i.e.*, the term that does not depend on T). For large values of T, second-order terms can be neglected in the comparison of the algorithms in finite MDPs, since they do not depend on T. However, they play an important role in continuous MDPs, where S and A are replaced by the σ-covering number of the state-action space, which is roughly $1/\sigma^d$. In this case, the algorithms define the granularity σ of the representation of the state-action space based on the number of episodes T, connecting the number of states S with T. For example, in [SS19] the ε-net used by the algorithm is tuned such that $\varepsilon = (HT)^{-\frac{1}{d+2}}$ (see also [OR12; LOR15; Jia+19]). Similarly, for Kernel-UCBVI we have that $\sigma = T^{-\frac{1}{2d+1}}$ (Corollary 4.8). For this reason, the second-order term in UCBVI becomes the dominant term in our analysis, leading to a worse dependence on d compared to model-free algorithms. For similar reasons, Kernel-UCBVI has an additional $\sqrt{H}$ factor compared to model-free algorithms based on OptQL. However, as observed in Section 4.7, model-based algorithms seem to enjoy a better empirical performance.

4.4 KeRNS: An Extension of Kernel-UCBVI to Non-Stationary MDPs

4.4.1 Non-Stationary Environments and Dynamic Regret

During the online interaction of an agent with an environment, it is possible that the behavior of the environment changes from one episode to another. For instance, imagine that the goal of the agent is to control the heating system of a building, in order to keep its temperature at a fixed value and minimize the total cost of electricity that is consumed. The optimal control strategy might depend on several external factors, such as the number of people currently inside the building, the weather outside, the time of the year, and the price of electricity. Some of those factors are unknown to the agent, and thus cannot be included as a part of the state variables $s \in \mathcal{S}$. Consequently, from the agent's perspective, the dynamics of the environment

are *non-stationary*: taking an action a at a state s in an episode t might have a different outcome than if a is taken at s in another episode t'.

Typically, reinforcement learning algorithms build their policy $\boldsymbol{\pi}^t$ to be executed in episode t based on the transitions observed up to episode $t-1$. For instance, the model $(\widehat{r}_h^t, \widehat{p}_h^t)_{h\in[H]}$ estimated by `Kernel-UCBVI` to compute its policy $\boldsymbol{\pi}^t$ is built with $(s_h^{t'}, a_h^{t'}, s_{h+1}^{t'}, \widetilde{r}_h^{t'})_{h\in[H]}$, for $t' \leq t-1$. Hence, if the transitions of the MDP change from one episode to another, those estimators are *biased* due to non-stationarity. If nothing is done to handle such bias, algorithms will suffer a linear regret [OGA19]. To deal with this issue, different approaches have been proposed for finite MDPs: Gajane et al. [GOA18] and Cheung et al. [CSLZ20] use sliding windows to compute estimators that use only the most recently observed transitions, whereas Ortner et al. [OGA19] restart the algorithm periodically and, after each restart, new estimators are build and past data are discarded. In the multi-armed bandit literature, in addition to sliding windows, exponential discounting has also been used as a mean to give more importance to recent data [KS06a; GM11; RVC19]. In this section, we show that `Kernel-UCBVI` can be adapted to handle the bias due to non-stationarity by using *time-dependent kernels*, which generalize the approaches based on sliding windows and exponential discounting.

Non-stationary MDPs We model the environment as a finite-horizon non-stationary MDP, where $\{r_h^t\}_{t,h}$ and $\{p_h^t\}_{t,h}$ are the sets of reward functions and transition kernels, respectively. More precisely, when taking action a in state s at time (t,h), the agent observes a random reward $\widetilde{r}_h^t \in [0,1]$ with mean $r_h^t(s,a)$ and makes a transition to the next state according to the probability measure $p_h^t(\cdot|s,a)$. In this case, the value functions also depend on the episode t. For a deterministic Markov policy $\pi : [H] \times \mathcal{S} \to \mathcal{A}$, its Q-function is defined as

$$Q_{t,h}^{\pi}(s,a) := \mathbf{E}\left[\sum_{h'=h}^{H} r_{h'}^t(s_{h'}, a_{h'}) \middle| s_h = s, a_h = a\right]$$

where $s_{h'+1} \sim p_{h'}^t(\cdot|s_{h'}, a_{h'})$, $a_{h'} = \pi(h', s)$, and its value function is defined by $V_{t,h}^{\pi}(s) = Q_{t,h}^{\pi}(s, \pi(h,s))$. The optimal value functions in episode t, $V_{t,h}^*(s) := \sup_\pi V_{t,h}^{\pi}(s)$ satisfy the Bellman equations [Put94],

$$V_{t,h}^*(s) = \max_{a\in\mathcal{A}} Q_{t,h}^*(s,a), \text{ where } Q_{t,h}^*(s,a) := r_h^t(s,a) + p_h^t V_{t,h+1}^*(s,a),$$

where $V_{t,H+1}^* = 0$ by definition.

Dynamic regret In non-stationary environments, we measure the performance of an agent by its *dynamic regret* $\mathcal{R}_T^{\text{dyn}}$ after T episodes, as defined below. When compared to the usual regret definition, the difference is that, in each episode t, the dynamic regret compares the performance of the agent to $V_{t,1}^*$, the optimal value function *in episode* t, whereas the regret

compares the agent to a *fixed* optimal value function V_1^*. The dynamic regret is defined as

$$\mathcal{R}_T^{\mathrm{dyn}} := \sum_{t=1}^{T} \left(V_{t,1}^*(s_1^t) - V_{t,1}^{\pi^t}(s_1^t) \right),$$

where s_1^t is the starting state in each episode.

4.4.2 Algorithm

To handle non-stationarity, we propose a modification of the Kernel-UCBVI introduced in Section 4.1, that we call KeRNS.[3] It has exactly the same structure as Kernel-UCBVI, as described in Algorithm 4.1, except that the weights are computed using a time-dependent kernel. Let $\Gamma : \mathbb{N} \times (\mathcal{S} \times \mathcal{A})^2 \to [0,1]$ be a *non-stationary kernel function*, where $\Gamma(n,u,v)$ represents the similarity between two state action pairs u, v in $\mathcal{S} \times \mathcal{A}$ separated by a time interval n. KeRNS is defined by a modification of the weights and normalized weights in Equation (4.1), which are now defined as

$$w_h^{t,i}(s,a) := \Gamma\left(t-i-1, (s,a), (s_h^i, a_h^i)\right) \quad \text{and} \quad \widetilde{w}_h^{t,i}(s,a) := \frac{w_h^{t,i}(s,a)}{\mathbf{C}_h^t(s,a)} \tag{4.7}$$

where $\mathbf{C}_h^t(s,a) := \beta + \sum_{i=1}^{t-1} w_h^{t,i}(s,a)$.

That is, the difference with respect to Kernel-UCBVI is that the weights $w_h^{t,i}(s,a)$ also depend on the episode t: the definition of the model estimator (4.2), exploration bonuses (4.3) and algorithmic structure (Algorithm 4.1) remain the same.

The weights $w_h^{t,i}(s,a)$ measure the influence that the transitions and rewards observed at time (i,h) will have on the estimators for the state-action pair (s,a) at time (t,h). Intuitively, the kernel function Γ must be designed so that $w_h^{t,i}(s,a)$ is small when $t-i-1$ is large, which means that the sample (s_h^i, a_h^i) was collected too far in the past and should have a small impact on the estimators at time t: this allows the agent to control the bias due to non-stationarity.

For the theoretical analysis of KeRNS, we need the assumptions below on the kernel function Γ:

Assumption 4.9 (non-stationary kernel properties)**.** *Let $\sigma > 0$, $\lambda \in]0,1[$ and $W \in \mathbb{N}$ be the kernel parameters. For each set of parameters, we assume that we have access to a base kernel function $\overline{\Gamma}_{(\lambda,W)} : \mathbb{N} \times \mathbb{R}_+ \to [0,1]$ and we define, for any $n, u, v \in \mathbb{N}^* \times \mathcal{S} \times \mathcal{A}$,*

$$\Gamma(n,u,v) = \overline{\Gamma}_{(\lambda,W)}\left(n, \rho\left[u,v\right]/\sigma\right).$$

[3] meaning Kernel-based Reinforcement Learning in Non-Stationary environments.

We assume that $z \mapsto \overline{\Gamma}_{(\lambda,W)}(n,z)$ is non-increasing for any $n \in \mathbb{N}$. Additionally, we assume that there exists positive constants C_1, C_2, a constant $C_3 \geq 0$ and an arbitrary function $G : \mathbb{R} \to \mathbb{R}_{\geq 0}$ that satisfies $G(4) > 0$ such that

(1) Fast decay: $\forall (n,z),\ \overline{\Gamma}(n,z) \leq C_1 \exp\left(-z^2/2\right),$

(2) Lipschitzness: $\forall (n,y,z),\ \left|\overline{\Gamma}(n,y) - \overline{\Gamma}(n,z)\right| \leq C_2 \left|y - z\right|,$

(3) Forget old data: $\forall z,\ \overline{\Gamma}_{(\lambda,W)}(n,z) \leq C_3 \lambda^n, \quad$ *for all* $n \geq W,$

(4) Remember recent data: $\forall z,\ \overline{\Gamma}_{(\lambda,W)}(n,z) \geq G(z)\lambda^n, \quad$ *for all* $n < W.$

We now provide some justification for these conditions. **(1)** and **(2)** are the same as required by Assumption 4.4 for `Kernel-UCBVI`, and ensure smoothness conditions to construct confidence intervals and to control the bias introduced by σ. **(3)** and **(4)** allow us to control the bias and the variance due to non-stationarity, respectively. Intuitively, **(3)** says the algorithm should forget data further than W episodes in the past (to reduce the bias), and **(4)** says that recent data in the W most recent episodes must have a minimum weight (to reduce the variance). In the next section, we analyze regret bounds for `KeRNS`, and we see that an appropriate choice of the kernel parameters (λ, W) allows us to balance this bias-variance trade-off.

The kernels in the example below satisfy our conditions, and show that they indeed generalize sliding-window and exponential discounting approaches.

Example 4.10 (sliding-window and exponential discount)**.** *Let $q \geq 2$. The kernels*

$$\overline{\Gamma}_{(\lambda,W)}(n,z) = \mathbb{1}\{n < W\} \exp(-z^q/2) \quad \textbf{(sliding-window)}$$
$$\overline{\Gamma}_{(\lambda,W)}(n,z) = \lambda^n \exp(-z^q/2) \quad \textbf{(exponential discount)}$$

satisfy Assumption 4.9.

The conditions in Assumption 4.9 are needed to prove regret bounds for `KeRNS`. However, if one has further knowledge about the MDP and its changes, this information can also be integrated to the kernel function Γ. For example, if the MDP only changes in a certain region of the state-action space, the kernel can be designed to forget past data only in that region.

4.5 Regret Analysis of `KeRNS`

To provide dynamic regret bounds for `KeRNS`, we rely on Assumption 4.9 and we also consider that assumptions 4.1, 4.2, and 4.3 hold for the rewards $(r_h^t)_h$, the transitions $(p_h^t)_h$, and the value functions $(Q_{t,h}^*)_h$, in all episodes t. That is, we assume that the metric ρ does not depend on t,

and that the rewards, transitions and value functions are Lipschitz continuous with respect to ρ, and that their Lipschitz constants do not depend on t.

The dynamic regret bound that we provide for KeRNS depends on a quantity Δ, which is the total variation of the MDP in T episodes:

Definition 4.11 (MDP variation)**.** *We define* $\Delta = \Delta^{\mathrm{r}} + L\Delta^{\mathrm{p}}$*, where*

$$\Delta^{\mathrm{r}} := \sum_{i=1}^{T}\sum_{h=1}^{H}\sup_{s,a}\left|r_h^i(s,a) - r_h^{i+1}(s,a)\right|, \quad \textit{and} \quad \Delta^{\mathrm{p}} := \sum_{i=1}^{T}\sum_{h=1}^{H}\sup_{s,a}\mathbb{W}_1\left(p_h^i(\cdot|s,a), p_h^{i+1}(\cdot|s,a)\right).$$

A similar notion has been introduced, for instance, by Ortner et al. [OGA19] and Li and Li [LL19] for MDPs and by Besbes et al. [BGZ14] for multi-armed bandits. Here, the difference is that we use the Wasserstein distance to define the variation of the transitions, instead of the total variation (TV) distance $\|p_h^i(\cdot|s,a) - p_h^{i+1}(\cdot|s,a)\|_1$. This choice was made in order to take into account the metric ρ when measuring changes in the environment, and the results would be analogous if we had chosen the TV distance.

We provide two regret bounds for KeRNS, which are given in the theorem below.

Theorem 4.12. *Let* (σ, λ, W) *be the kernel parameters and*

$$\mathcal{R}_T^1 := H^2T\sqrt{\log\frac{1}{\lambda}}\sqrt{|\mathcal{C}'_\sigma|\,|\mathcal{C}_\sigma|} + H^2\,|\mathcal{C}_\sigma|\,T\log\frac{1}{\lambda};$$
$$\mathcal{R}_T^2 := H^2T\sqrt{\log\frac{1}{\lambda}}\sqrt{|\mathcal{C}_\sigma|} + H^3\,|\mathcal{C}_\sigma|\,|\mathcal{C}'_\sigma|\,T\log\frac{1}{\lambda}; \quad \textit{and}$$
$$\mathbf{bias}(\sigma,\lambda,W,\Delta,T) := W\Delta H + \frac{\lambda^W}{1-\lambda}H^3T + LHT\sigma.$$

Then, with probability at least $1-\delta$*, the dynamic regret of* KeRNS *satisfies*

$$\mathcal{R}_T^{\mathrm{dyn}} \lesssim \min\left(\mathcal{R}_T^1, \mathcal{R}_T^2\right) + \mathbf{bias}(\sigma,\lambda,W,\Delta,T).$$

Proof The proof of Theorem 4.12 is detailed in our paper [Dom+21c]. It is based on the regret analysis of Kernel-UCBVI and borrows techniques used to prove regret bounds for non-stationarity MDPs with finite state-action sets [OGA19] and for non-stationarity multi-armed bandits [GM11; RVC19]. In Appendix C.3, we provide a proof sketch.

Before discussing the regret bounds for KeRNS, we present the corollary below, which gives the bounds resulting from optimizing the kernel parameters (σ, λ, W), that is, from optimizing the bias-variance trade-off in the regret bound.

Corollary 4.13. *Recall that d_1 and d_2 are the covering dimensions of the state-action and the state spaces, respectively, and that $d = \max(d_1, d_2)$. By optimizing the kernel parameters, we obtain the regret bounds in Table 4.1, which also presents the conditions on the variation Δ that are required for a sub-linear regret bound.*

Proof. Since $|\mathcal{C}_\sigma|$ and $|\mathcal{C}'_\sigma|$ are $\mathcal{O}\left(1/\sigma^d\right)$, the bounds follow from Theorem 4.12. □

Table 4.1 – Regret bound for KeRNS with optimized kernel parameters, for $W = \log_\lambda \frac{(1-\lambda)}{T}$.

	σ	$\log\left(\frac{1}{\lambda}\right)$	condition	bound	regret
$d = 0$	0	$\Delta^{\frac{2}{3}} T^{-\frac{2}{3}}$	$\Delta < T$	$\mathcal{R}_T^1$	$H^2 S\sqrt{A}\Delta^{\frac{1}{3}} T^{\frac{2}{3}}$
	0	$\Delta^{\frac{2}{3}} T^{-\frac{2}{3}}$	$\Delta < T$	$\mathcal{R}_T^2$	$H^2\sqrt{SA}\Delta^{\frac{1}{3}} T^{\frac{2}{3}} + H^3 S^2 A \Delta^{\frac{2}{3}} T^{\frac{1}{3}}$
$d > 0$	$\left(\frac{1}{T}\right)^{\frac{1}{2d+3}}$	$\Delta^{\frac{2}{3}} T^{-\frac{2d+2}{2d+3}}$	$\Delta < T^{\frac{3}{2d+3}}$	$\mathcal{R}_T^1$	$H^2 \Delta^{\frac{1}{3}} T^{\frac{2d+2}{2d+3}}$
	$\left(\frac{1}{T}\right)^{\frac{1}{2d+2}}$	$\frac{\Delta^{\frac{1}{2}}}{H} T^{-\frac{2d+1}{2d+2}}$	$\Delta < T^{\frac{1}{d+1}}$	$\mathcal{R}_T^2$	$H^2 \Delta^{\frac{1}{2}} T^{\frac{2d+1}{2d+2}} + H^{\frac{3}{2}} \Delta^{\frac{1}{4}} T^{\frac{3}{4}}$

We see that, after optimizing the kernel parameters (Table 4.1), the bound $\mathcal{R}_T^1$ has a worse dependence on T, and a better dependence on Δ. On the other hand, $\mathcal{R}_T^2$ is better with respect to T, but worse in Δ. The difference comes from how we handle the concentration inequalities for the transition probabilities in the analysis. To obtain $\mathcal{R}_T^1$, we use concentration inequalities on the term $|(\hat{p}_h^t - p_h^t) f|$ for *all* functions f that are bounded and Lipschitz continuous. To obtain $\mathcal{R}_2$, the concentration is done only for $f = V_{t,h+1}^*$, but this results in a worse dependence on $|\mathcal{C}_\sigma|\,|\mathcal{C}'_\sigma|$ in the regret bound (Theorem 4.12).

We now discuss the regret bounds according to the covering dimension d. We consider two cases: the finite MDP case, where $d = 0$, and the continuous case, where $d > 0$.

Finite case Let $S = |\mathcal{S}|$ and $A = |\mathcal{A}|$. By taking $\sigma = 0$, we have $|\mathcal{C}'_\sigma| = S$ and $|\mathcal{C}_\sigma| = SA$. As shown in Table 4.1, the $\mathcal{R}_T^1$ bound states that the regret of KeRNS is $\widetilde{\mathcal{O}}\left(H^2 S\sqrt{A}\Delta^{\frac{1}{3}} T^{\frac{2}{3}}\right)$. This bound matches the one proved by Ortner et al. [OGA19] for the average-reward setting using restarts, up to a factor of $H^{\frac{2}{3}}$ coming from our finite-horizon setting, where the transitions p_h^t depend on h. The $\mathcal{R}_T^2$ bound states that the regret of KeRNS can be improved to $\widetilde{\mathcal{O}}\left(H^2\sqrt{SA}\Delta^{\frac{1}{3}} T^{\frac{2}{3}}\right)$, up to second-order terms (*i.e.*, the terms scaling with $T^{1/3}$). In the multi-armed bandit case ($H = 1$), these bounds are *optimal* in terms of T and Δ, according to the lower bound by Besbes et al. [BGZ14].

Continuous case For $d > 0$, we proved the first dynamic regret bounds in our setting, which are of order $H^2\Delta^{\frac{1}{3}} T^{\frac{2d+2}{2d+3}}$ (better in Δ) or $H^2\Delta^{\frac{1}{2}} T^{\frac{2d+1}{2d+2}}$ (better in T) for two different tunings of

the kernel. Deriving a lower bound in the non-stationary case for $d > 0$ is an open problem, even for multi-armed bandits. As a sanity-check, we note that in stationary MDPs, for which $\Delta = 0$, we recover the regret bound of Kernel-UCBVI[4] of $H^3 T^{\frac{2d}{2d+1}}$ from the bound $\mathcal{R}_2$ with $\log(1/\lambda) = 1/K$, $W \to \infty$ and $\sigma = T^{-\frac{1}{2d+1}}$, which is optimal for $d = 1$ in the (stationary) bandit case, according to the lower bound by Bubeck et al. [Bub+11].

Knowledge of Δ To optimally choose the kernel parameters, KeRNS requires an upper bound on the variation Δ. Other works have started to tackle this issue in bandit algorithms [Che+19; AGO19], and finite MDPs using sliding windows [CSLZ20]. For instance, [CSLZ20] use a multi-armed bandit algorithm to adaptively tune the size of the sliding window, and avoid the need of knowing Δ. A similar technique could be combined with KeRNS to adaptively choose the parameters (λ, W) in each episode.

4.6 Reducing the Computational Complexity

In order to analyze the computational complexity of Kernel-UCBVI and KeRNS, we first assume that the action set is finite and has cardinality $A := |\mathcal{A}|$. This is due to the fact that both algorithms require computations of $\operatorname{argmax}_a Q_h^t(s,a)$, which is usually not trivial for infinite action sets. Since Kernel-UCBVI and KeRNS use non-parametric kernel estimators, their computational complexity scales with the number of observed transitions. Their total space complexity is $\mathcal{O}(HT)$ and their time complexity per episode t is $\mathcal{O}(Ht^2 + HAt)$, resulting in a total runtime of $\mathcal{O}(HT^3 + HAT^2)$ for T episodes. This runtime is very prohibitive in practice, especially in non-stationary environments, where we might need to run the algorithm for a very long time. In this section, we study two methods to reduce their computational complexity: real-time dynamic programming [BBS95; Efr+19] that reduces the complexity of backward induction, and the use of representative states [KT12; BPP16], that reduces the number of states on which backward induction is executed.

4.6.1 Real-Time Dynamic Programming

Algorithm 4.3 describes the Kernel-UCBVI algorithm using real-time dynamic programming (RTDP). At time (t,h) Kernel-UCBVI+RTDP computes $\widetilde{Q}_h^t(s,a)$ for all actions a, but *only for the current state* $s = s_h^t$, whereas Kernel-UCBVI computes it for $s \in \{s_h^1, \dots, s_h^{t-1}\}$: that is why it is called a *real-time* algorithm.

Then, Kernel-UCBVI+RTDP executes the greedy action $a_h^t = \operatorname{argmax}_{a \in \mathcal{A}} \widetilde{Q}_h^t(s_h^t, a)$. As a next step, it computes $\widetilde{V}_h^t(s_h^t) = \widetilde{Q}_h^t(s_h^t, a_h^t)$ and refines the previous L-Lipschitz upper confidence

[4] Another choice of λ might allow us to avoid the dependence on H^3 of Kernel-UCBVI and get H^2 instead.

Algorithm 4.3: `Kernel-UCBVI+RTDP`

initialization: $V_h^1(s) \leftarrow H - h + 1$ for all $s \in \mathcal{S}$ and all $h \in [H]$.
for *episode* $t \in \{1, \dots, T\}$ **do**
  get initial state s_1^t
  # compute optimistic Q-functions
  for *stage* $h \in \{1, \dots, H\}$ **do**
    for $a \in \mathcal{A}$ **do**
      $\widetilde{Q}_h^t(s_h^t, a) \leftarrow \sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s_h^t, a) \left(\widetilde{r}_h^i + V_{h+1}^t(s_{h+1}^i)\right) + \mathtt{b}_h^t(s_h^t, a)$
    # select action
    $a_h^t \leftarrow \operatorname{argmax}_a \widetilde{Q}_h^t(s_h^t, a)$
    # execute action
    $\widetilde{r}_h^t, s_{h+1}^t \leftarrow \mathtt{OnlineModel}_{t,h}(a_h^t)$
    $\widetilde{V}_h^t(s_h^t) \leftarrow \min\left(H - h + 1, \max_a \widetilde{Q}_h^t(s_h^t, a)\right)$
    # interpolate
    for $i = 1, \dots, t$ **do**
      $s \leftarrow s_h^i$
      $V_h^{t+1}(s) \leftarrow \min\left(V_h^t(s), \widetilde{V}_h^t(s_h^t) + L\rho_{\mathcal{S}}\left(s, s_h^t\right)\right)$

bound on the value function, by defining

$$\forall s,\ V_h^{t+1}(s) = \min\left(V_h^t(s), \widetilde{V}_h^t(s_h^t) + L\rho_{\mathcal{S}}\left(s, s_h^t\right)\right). \tag{4.8}$$

Notice that, although V_h^{t+1} is defined for all s, the algorithm only needs to compute it for the previously observed states. Since $\widetilde{Q}_h^t(s, a)$ is only computed for $s = s_h^t$, the runtime of `Kernel-UCBVI+RTDP` is $\mathcal{O}(HAt)$ per episode t, that is, $\mathcal{O}(t)$ times faster than `Kernel-UCBVI`. Also, notice that `Kernel-UCBVI+RTDP` also requires a metric $\rho_{\mathcal{S}}$ on the state space for the interpolation step (although the interpolation can be skipped in practical implementations, as mentioned in Section 4.1).

Theorem 4.14 shows that the regret bound of `Kernel-UCBVI+RTDP` is of the same order as the one of `Kernel-UCBVI`. Its proof follows the analysis of RTDP as proposed by Efroni et al. [Efr+19] for finite MDPs.

Theorem 4.14. *With probability at least* $1 - \delta$*, the regret of* `Kernel-UCBVI+RTDP` *satisfies*

$$\mathcal{R}_T \lesssim \mathcal{R}_T^{\texttt{Kernel-UCBVI}} + H^2 \left|\mathcal{C}_\sigma'\right|,$$

where $\mathcal{R}_T^{\texttt{Kernel-UCBVI}}$ *is the regret bound for* `Kernel-UCBVI` *from Theorem 4.7.*

Proof. The proof is given in Appendix C.4, and the key properties for proving this regret bound are (i) $V_h^t \geq V_h^*$ with high probability, and (ii) the fact that $V_h^{t+1} \leq V_h^t$; where V_h^t is defined in Equation (4.8). □

4.6.2 Representative States and Actions

The RTDP technique used to speed up `Kernel-UCBVI` cannot be applied to `KeRNS`. This is due to the fact that `Kernel-UCBVI+RTDP` builds upper bounds V_h^t that are point-wise *non-increasing* with respect to t, whereas the value functions V_h^t computed by `KeRNS` increases for states that were not visited recently. This property of `KeRNS` is necessary to promote extra exploration and adapt to possible changes in the environment. Additionally, even with the RTDP-based acceleration, `Kernel-UCBVI+RTDP` still has a time complexity that increases with the time t, which can be a considerable limitation in non-stationary environments where the algorithm needs to be run for a long time. In this section, we propose an alternative to run `KeRNS` in *constant* time per episode, while controlling the impact of this speed-up on the regret.

As proposed by [KT12] and [BPP16], we take an approach based on using *representative states* to construct an algorithm called `RS-KeRNS` (for `KeRNS` on Representative States). In each episode t, `RS-KeRNS` keeps and updates sets of representative states $\bar{\mathcal{S}}_h^t$, actions $\bar{\mathcal{A}}_h^t$ and next-states $\bar{\mathcal{Y}}_h^t$, for each h, whose cardinalities are denoted by $\bar{S}_h^t$, $\bar{A}_h^t$ and $\bar{Y}_h^t$, respectively. Every time a new transition $\left\{s_h^t, a_h^t, s_{h+1}^t, \tilde{r}_h^t\right\}$ is observed, the representative sets are updated using Algorithm 4.4, which ensures that any two representative state-action pairs are at a distance greater than ε from each other. Similarly, it ensures that any pair of representative next-states are at a distance greater than $\varepsilon_{\mathcal{X}}$ from each other. Then, (s_h^t, a_h^t) and s_{h+1}^t are mapped to their nearest neighbors in $\bar{\mathcal{S}}_h \times \bar{\mathcal{A}}_h$ and $\bar{\mathcal{Y}}_h$, respectively, and the estimators of the rewards and transitions are updated. Consequently, we build a finite MDP, denoted by $\widetilde{\mathcal{M}}_t$, with $\bar{S}_h^t$ states, $\bar{A}_h^t$ actions and $\bar{Y}_h^t$ next-states, *per stage* h. The rewards and transitions of $\widetilde{\mathcal{M}}_t$ can be stored in arrays of size $\bar{S}_h^t \bar{A}_h^t$ and $\bar{S}_h^t \bar{A}_h^t \bar{Y}_h^t$, for each h.

Remark 4.15. *The technique of using representative states also applies to* `Kernel-UCBVI`*, since it is equivalent to* `KeRNS` *when using stationary kernels. In that case, we name the resulting algorithm* `RS-Kernel-UCBVI`.

Algorithm 4.4: Update Representative Sets

input: Input: t, h, $\bar{\mathcal{S}}_h^t$, $\bar{\mathcal{A}}_h^t$, $\bar{\mathcal{Y}}_h^t$, $\left\{s_h^t, a_h^t, s_{h+1}^t\right\}$, ε, $\varepsilon_{\mathcal{X}}$.
if $\min_{(\bar{s},\bar{a})\in\bar{\mathcal{S}}_h\times\bar{\mathcal{A}}_h} \rho\left[(\bar{s},\bar{a}),(s_h^t,a_h^t)\right] > \varepsilon$ **then**
  $\bar{\mathcal{S}}_h^{t+1} \leftarrow \bar{\mathcal{S}}_h^t \cup \{s_h^t\}$, $\bar{\mathcal{A}}_h^{t+1} \leftarrow \bar{\mathcal{A}}_h^t \cup \{a_h^t\}$
if $\min_{\bar{y}\in\bar{\mathcal{Y}}_h} \rho_{\mathcal{S}}\left(\bar{s}, s_{h+1}^t\right) > \varepsilon_{\mathcal{X}}$ **then**
  $\bar{\mathcal{Y}}_h^{t+1} \leftarrow \bar{\mathcal{Y}}_h \cup \left\{s_{h+1}^t\right\}$

RS-KeRNS is described precisely in Appendix C.5 (Algorithm C.1). It computes a Q-function for all $(\overline{s}, \overline{a}) \in \cup_h \bar{\mathcal{S}}_h^t \times \bar{\mathcal{A}}_h^t$ by running backward induction in $\widetilde{\mathcal{M}}_t$, which is then extended to any $(s, a) \in \mathcal{S} \times \mathcal{A}$ by performing an interpolation step, as in KeRNS. In Appendix C.5, we also explain how the rewards and transitions estimators of $\widetilde{\mathcal{M}}_k$ can be updated online. Below, we provide regret and runtime guarantees for this efficient implementation.

Theorem 4.16. *Let $\chi : \mathbb{N} \to [0, 1]$, $u, v \in \mathcal{S} \times \mathcal{A}$, and assume that we use the kernel*

$$\Gamma(n, u, v) = \chi(n) \exp\left(-\rho\,[u, v]^2 / (2\sigma^2)\right)$$

assumed to satisfy Assumption 4.9. In this case, the dynamic regret of RS-KeRNS satisfies

$$\mathcal{R}_T^{\mathrm{dyn}} \lesssim \mathcal{R}_T^{\texttt{KeRNS}} + L(\varepsilon + \varepsilon_{\mathcal{X}})H^2T + \frac{\varepsilon}{\sigma}H^3T$$

with probability at least $1 - \delta$, where $\mathcal{R}_T^{\texttt{KeRNS}}$ is regret bound of KeRNS given in Theorem 4.12.

Proof. A detailed proof is given in our paper [Dom+21c] (Theorem 2). A proof sketch is given in Appendix C.6. □

Theorem 4.16 shows that RS-KeRNS enjoys the same regret bounds as KeRNS plus a bias term that can be controlled by ε and $\varepsilon_{\mathcal{X}}$, as long as we use a kernel Γ that is the product between a temporal kernel $\chi(n)$ and a Gaussian kernel $\exp\left(-\rho\,[u, v]^2 / (2\sigma^2)\right)$.

The lemma below shows that the per-episode runtime of RS-KeRNS is bounded by a constant.

Lemma 4.17 (runtime of RS-KeRNS)**.** *Consider the kernel defined in Theorem 4.16, and let $\lambda \in]0, 1]$. If we take $\chi(n) = \lambda^n$, the runtime of RS-KeRNS in each episode t is bounded by*

$$\mathcal{O}\left(H \min\left(t^2, |\mathcal{C}_\varepsilon||\mathcal{C}'_{\varepsilon_{\mathcal{X}}}|\right) + H \min\left(t, |\mathcal{C}'_{\varepsilon_{\mathcal{X}}}|\right) A\right),$$

where $|\mathcal{C}_\varepsilon|$ is the ε-covering number of $(\mathcal{S} \times \mathcal{A}, \rho)$, $|\mathcal{C}'_{\varepsilon_{\mathcal{X}}}|$ is the $\varepsilon_{\mathcal{X}}$-covering number of $(\mathcal{S}, \rho)$.

Lemma 4.17 bounds the runtime of RS-KeRNS for a temporal kernel $\chi(n)$ satisfying a specific structure: $\chi(n) = \lambda^n$. In particular, we can take $\lambda < 1$, which gives an exponential-discount strategy for handling non-stationarity, or set $\lambda = 1$ if the environment is stationary.

Consequently, the constants ε and $\varepsilon_{\mathcal{X}}$ provide a trade-off between regret and computational complexity. Since $|\mathcal{C}_\varepsilon| = \mathcal{O}\left(\varepsilon^{-d_1}\right)$ and $|\mathcal{C}'_{\varepsilon_{\mathcal{X}}}| = \mathcal{O}\left(\varepsilon_{\mathcal{X}}{}^{-d_2}\right)$, increasing $(\varepsilon, \varepsilon_{\mathcal{X}})$ may reduce *exponentially* the runtime of RS-KeRNS, while having only a *linear* increase in its regret.

Related work on representative states for accelerating KBRL Kveton and Theocharous [KT12] and Barreto et al. [BPP16] propose the use of representative states to accelerate KBRL, and we provided the first regret bounds in this setting. More precisely, our result complements previous work in the following aspects: (i) [KT12] and [BPP16] do not tackle exploration and do not have finite-time analyses: they provide approximate versions of the KBRL algorithm of [OS02] which has asymptotic guarantees assuming that transitions are generated from *independent* samples; (ii) the error bounds of [KT12] scale with $\exp(1/\sigma^2)$. In our online setting, σ can be chosen as a function of the number of episodes T, and their bound could result in an error that scales exponentially with T, instead of linearly. Our result comes from an improved analysis of the smoothness of kernel estimators, that leverages the regularization constant β; (iii) [BPP16] propose an algorithm that also builds a set of representative states in an online manner. However, their theoretical guarantees only hold when this set is *fixed*, *i.e.* cannot be updated during exploration, whereas our bounds hold in this case.

4.7 Experiments

In this section, we illustrate the empirical behavior of `Kernel-UCBVI` and `KeRNS` on simple environments, compared to baselines. For the implementation of both algorithms, we used representative states to decrease their runtime. We refer to `RS-Kernel-UCBVI+RTDP` as the version of `Kernel-UCBVI` using both real-time dynamic programming (RTDP) and representative states. The environments we considered are such that $\mathcal{S} \subset \mathbb{R}^2$, thus we used the Euclidean distance on $\mathbb{R}^2$ to define the metric,

$$\rho\left[(s,a),(s',a')\right] = \left\|s - s'\right\|_2^2 \text{ if } a = a', \text{ and } +\infty \text{ otherwise,}$$

which was combined with the Gaussian kernel $\exp(-z^2/2)$ to compute the weights. We set $\sigma = 0.025$, $\varepsilon = \varepsilon_{\mathcal{X}} = 0.05$, and $\beta = 0.01$ for all experiments. Some baselines require a uniform discretization of the state space, in which case we chose the granularity of the discretization to match the value of ε used to define representative states for `Kernel-UCBVI`.

Also, we consider environments such that the rewards and transition probabilities do not depend on h (however, for non-stationary environments, they depend on t). Hence, for the variants of `Kernel-UCBVI` and `KeRNS`, the model estimators were built based on Equation (4.4), and we used the following simplified exploration bonuses

$$\mathfrak{b}_h^t(s,a) = \frac{1}{\sqrt{\mathbf{C}^t(s,a)}} + \frac{H-h+1}{\mathbf{C}^t(s,a)}. \tag{4.9}$$

That is, we used bonuses and models for h-independent MDPs, and considered simplified constants κ_1, κ_2 and κ_3 for the bonuses.

4.7.1 Stationary Environment

To illustrate experimentally the properties of `Kernel-UCBVI`, we consider a Grid-World environment with continuous states. This Grid-World has a state space $\mathcal{S} \subset [0,2] \times [0,1] \subset \mathbb{R}^2$, and is composed of two rooms separated by a wall of width 0.1, as illustrated by Figure 4.1. There are four actions: left, right, up, and down, each one resulting to a displacement of 0.1 in the corresponding direction. A two-dimensional Gaussian noise is added to the transitions, and, in each room, there is a single region with non-zero reward. The agent has 0.5 probability of starting in each of the rooms, and the starting position is at the room's bottom left corner.

We compare `RS-Kernel-UCBVI` and `RS-Kernel-UCBVI+RTDP` to the following algorithms:

- `UCBVI` [AOM17] using a uniform discretization of the state-space with 20 bins in each coordinate;
- `OptQL` [Jin+18], also on a uniform discretization;
- `AdaptiveQL` [SBY19] that uses an adaptive discretization of the state space.

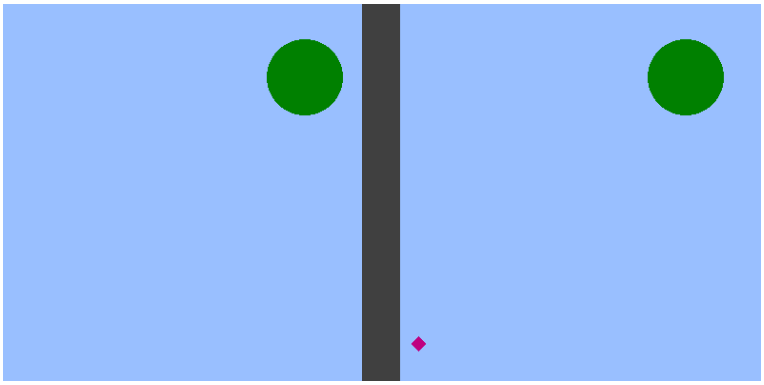


Figure 4.1 – Continuous Grid-World with two rooms separated by a wall. The circles represent the regions with non-zero rewards.

The baselines require a discretization of the state space, and we denote by $I(s_h^t)$ the index of the discrete state corresponding to the continuous state s_h^t. For the baselines, we used the same simplified bonuses as for `Kernel-UCBVI` (4.9), except that $\mathbf{C}^t(s,a)$ is replaced by

$$\mathbf{N}_h^t(I(s),a) = \max\left(1, \sum_{i=1}^{t-1} \mathbb{1}\left\{I(s_h^i) = I(s), a_h^i = a\right\}\right),$$

for `OptQL` and `AdaptiveQL` (since those algorithms require h-dependent bonuses *even if the MDP is time-homogeneous*) and, for `UCBVI`, we used

$$\mathbf{N}^t(I(s),a) = \max\left(1, \sum_{h=1}^{H}\sum_{i=1}^{t-1} \mathbb{1}\left\{I(s_h^i) = I(s), a_h^i = a\right\}\right).$$

We also implemented a version of RS-Kernel-UCBVI using a prior domain knowledge that the two rooms are equivalent under translation, by using a metric that is invariant with respect to the change of rooms. More precisely, before computing the Euclidean distance between two states s and s', both states are mapped to their corresponding positions in the left room.

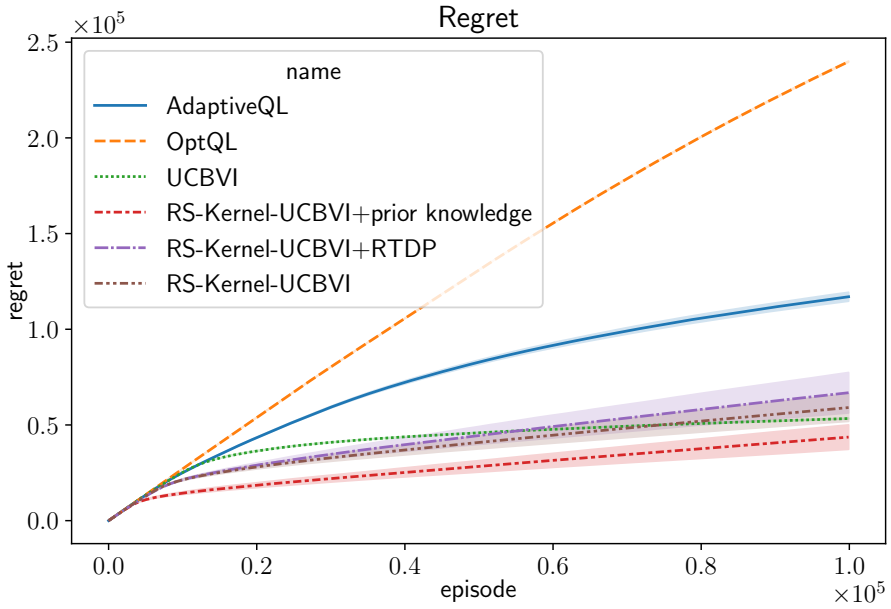


Figure 4.2 – Regret of different versions of Kernel-UCBVI compared to baselines (**smaller is better**). To estimate the optimal value function for the regret computation, we used the best policy among all agents at the final episode. Average over 16 independent runs.

We ran the algorithms for 10^5 episodes with $H = 20$, and Figure 4.2 shows the regret incurred by each of them. We see that, *at the beginning*, Kernel-UCBVI has a smaller regret than the baselines, and that the use of expert knowledge in the kernel design can increase the learning speed. Also, we see that model-based algorithms (Kernel-UCBVI and UCBVI) learn faster than the model-free baselines (OptQL and AdaptiveQL), despite the fact that those baselines have a better regret with respect to the dimension d [SS19; SBY19], as discussed in Section 4.3. As the number of episodes T increases, the extra bias introduced by the kernel might make Kernel-UCBVI converge to a worse policy when compared, for instance, to UCBVI using a uniform discretization. The kernel bandwidth and the discretization width are comparable, but the Gaussian kernel introduces more bias by assigning a non-zero similarity between states that may be in disjunct discretization bins.[5] On the other hand, we see that introducing more bias can improve the learning speed at the beginning, especially when domain knowledge is used for kernel design. This flexibility in handling the bias-variance trade-off is one of the strengths of kernel-based approaches: for the baselines used in this experiment, the use of arbitrary

[5]Notice that the bias can be controlled by changing the bandwidth σ, or by using kernels with faster decay or bounded support.

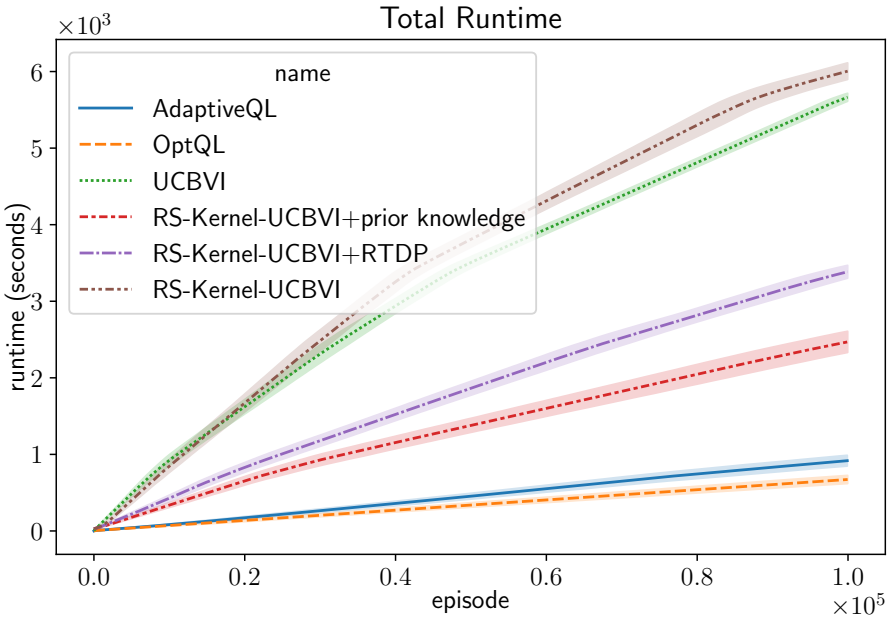


Figure 4.3 – Total runtime of different algorithms in a continuous Grid-World versus the number of episodes (**smaller is better**). Average over 16 runs.

custom metrics on the state space is not straightforward. Nonetheless, kernel-based algorithms might be sensitive to kernel design.

Another important point to consider is the runtime of the algorithms. Figure 4.3 shows the total runtime of each algorithm as function of the number of episodes. When comparing `RS-Kernel-UCBVI` to `RS-Kernel-UCBVI+RTDP`, we observe that using real-time dynamic programming results in a considerable speed-up. When combined with prior knowledge, `RS-Kernel-UCBVI` has a smaller number of representative states in the Grid-World environment, which explains why it is faster than its version without such knowledge. Also, we see that the model-free algorithms have a much smaller runtime than the model-based ones.

4.7.2 Non-Stationary Environment

To illustrate the behavior of `RS-KeRNS`, we consider another continuous Grid-World whose state-space is $\mathcal{S} = [0, 1]^2$ with four actions, representing a move to the right, left, up or down. The agent starts at $(0.5, 0.5)$. Let $c_p^t \in \{0, 0.1, 0.2, 0.5, 1\}$. We consider the following mean reward function which depends on the episode t:

$$r_h^t(s, a) = \sum_{p \in \{0.1, 0.9\}^2} c_p^t \exp\left(-\frac{\|s - p\|_2^2}{2 \times 0.1^2}\right),$$

where the vectors $p \in \{(0.1, 0.1), (0.1, 0.9), (0.9, 0.1), (0.9, 0.9)\}$ represent the positions where the rewards are centered. Every 2.5×10^4 episodes, the coefficients c_p^t are changed according to Table 4.2, which impact the optimal policy.

We used the kernel defined in Theorem 4.16 with $\chi(n) = \lambda^n$ for $\lambda = 0.9999$ and ran the algorithm for 10^5 episodes. RS-KeRNS was compared to two baselines: (i) RS-Kernel-UCBVI, which does not adapt to non-stationarity and corresponds to RS-KeRNS when we set $\lambda = 1$; (ii) a restart-based algorithm, called RestartBaseline which is implemented as RS-Kernel-UCBVI, but it uses *information about when the environment changes*, and, at every change, it restarts its reward estimator and bonuses, forcing the agent to re-explore the environment and discover possible changes. Notice that RS-KeRNS does not require such information.

Table 4.2 – Value of c_p^t for each to p in episode t.

episode / p	$(0.9, 0.9)$	$(9.0, 0.1)$	$(0.1, 0.1)$	$(0.1, 0.9)$
$t \in [1, 25 \times 10^3]$	0.1	0.0	0.0	0.0
$t \in]25 \times 10^3, 50 \times 10^3]$	0.1	0.2	0.0	0.0
$t \in]50 \times 10^3, 75 \times 10^3]$	0.1	0.2	0.5	0.0
$t \in]75 \times 10^3, 100 \times 10^3]$	0.1	0.2	0.5	1.0

In Figure 4.4, we can see that, as expected, RS-KeRNS gathers more rewards than RS-Kernel-UCBVI, which was not designed for non-stationary environments, and that RS-KeRNS is able to track the behavior of RestartBaseline.

4.8 Discussion and Bibliographical Remarks

In this chapter, we introduced and analyzed Kernel-UCBVI and KeRNS, which are kernel-based algorithms for RL that learn by interacting online with an MDP. Algorithmically, they only require a similarity function on the state-action space $\mathcal{S} \times \mathcal{A}$ to be implemented. Hence, they can be applied to *very general MDPs*, whose states can be, for instance, real vectors, graphs, sets, strings etc. Under certain regularity assumptions, we proved regret bounds that depend on the covering numbers or the covering dimension d of $\mathcal{S} \times \mathcal{A}$. Furthermore, we proposed approximate versions of both algorithms that can be run in constant time per episode, and we analyzed their regret. Nonetheless, the generality of Kernel-UCBVI and KeRNS comes at a cost: they suffer from the curse of dimensionality, meaning that their regret becomes close to linear as the dimension d increases, which is a direct consequence of the fact that we only make weak assumptions on the MDP.

Approaches such as linear or low-rank MDPs [Jin+20b] and RKHS approximations [Yan+20; CO20; CG19] avoid the curse of dimensionality and achieve regret bounds scaling with $\sqrt{T}$, but they either require much stronger assumptions on the MDP, such as the closedness of the

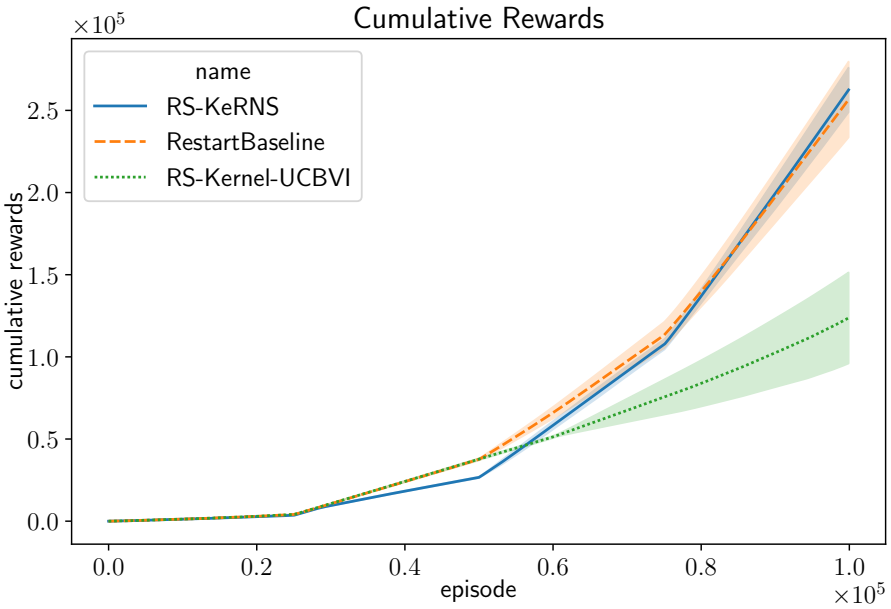


Figure 4.4 – Cumulative rewards of `RS-KeRNS` versus `RS-Kernel-UCBVI` and `RestartBaseline` in a non-stationary environment (**larger is better**). The environment changes every 2.5×10^4 episodes. Average over 16 independent runs.

Bellman operator in the function class used to represent optimistic Q-functions, or might be computationally intractable. Ren et al. [Ren+21] show that, under a noise assumption on the transition probabilities, the MDP is linear (see [Jin+20b]) in an infinite-dimensional feature space, and provide algorithms with regret bounds in this setting. However, since their results rely on noisy transitions, they do not apply to deterministic MDPs. Interestingly, Barreto et al. [BPP16] show how KBRL can be approximated by low-rank MDPs.

Further reading on continuous MDPs For MDPs with continuous state-action space, the sample complexity [KKL03; KS02; LHS+13; PP13] or regret have been studied under structural assumptions. Regarding regret minimization, a standard assumption is that rewards and transitions are Lipschitz continuous. Ortner and Ryabko [OR12] studied this problem in the average-reward setting. They combined the ideas of `UCRL2` [JOA10] and uniform discretization, proving a regret bound of $\widetilde{\mathcal{O}}\left(T^{\frac{2d+1}{2d+2}}\right)$ for a learning horizon T in d-dimensional state spaces. This work was later extended by Lakshmanan et al. [LOR15] to use a kernel density estimator instead of a frequency estimator for each region of the fixed discretization. For each *discrete* region $I(s)$, the density $p(\cdot|I(s), a)$ of the transition kernel is computed through kernel density estimation. The granularity of the discretization is selected in advance based on the properties of the MDP and the learning horizon T. As a result, they improve upon the bound of Ortner and Ryabko [OR12], but require the transition kernels to have densities that are κ times

differentiable.[6] However, these two algorithms rely on an intractable optimization problem for finding an optimistic MDP. Jian et al. [Jia+19] solve this issue by providing an algorithm that uses exploration bonuses, but they still rely on a uniform discretization of the state space. Ok et al. [OPT18] studied the asymptotic regret in Lipschitz MDPs with *finite* state and action spaces, providing a nearly asymptotically optimal algorithm. Their algorithm leverages ideas from asymptotic optimal algorithms in structured bandits [CMP17] and tabular RL [BK97], but does not scale to continuous state-action spaces. Regarding exploration for finite-horizon MDP with continuous state-action space, Ni et al. [NYW19] present an algorithm for deterministic MDPs with Lipschitz transitions. Assuming that the Q-function is Lipschitz continuous, Song and Sun [SS19] provided a model-free algorithm by combining the ideas of tabular optimistic Q-learning [Jin+18] with uniform discretization, showing a regret bound of $\mathcal{O}\left(H^{\frac{5}{2}}T^{\frac{d+1}{d+2}}\right)$ where d is the covering dimension of the state-action space. This approach was extended by Sinclair et al. [SBY19] and Touati et al. [TTB20] to use adaptive partitioning of the state-action space, achieving the same regret bound. Osband and Van Roy [OVR14] prove a *Bayesian* regret bound in terms of the eluder and Kolmogorov dimension, assuming access to an approximate MDP planner. Sinclair et al. [Sin+20] provide a model-based algorithm for continuous MDPs relying on adaptive discretization, with a regret bound that has a better dependence on H and d than `Kernel-UCBVI`, but requires extra assumptions on the metric space to construct packings and coverings. Cao and Krishnamurthy [CK20] prove a regret bound for the `AdaptiveQL` algorithm of [SBY19] that scales with the zooming dimension, which is a problem-dependent quantity that may be smaller than the covering dimension.

Further reading on non-stationary MDPs In this chapter, we considered dynamic regret bounds for non-stationary RL. In finite MDPs, this type of bound has been been studied by Gajane et al. [GOA18], Ortner et al. [OGA19], Cheung et al. [CSLZ20], and Mao et al. [Mao+21], whereas Touati and Vincent [TV20] propose an algorithm using linear function approximation. A related approach consists in comparing the performance of the learner to the best stationary policy in hindsight, *e.g.* [EDKM09; YM09; Neu+13; DGS14], which is less suited to non-stationary environments, since the performance of any fixed policy might not be satisfactory. Non-stationary RL has also been studied outside the regret minimization framework, without, however, tackling the issue of exploration. For instance, Choi et al. [CYZ00] propose a model where the MDP varies according to a sequence of tasks whose changes form a Markov chain. Szita et al. [STL02] and Csáji and Monostori [CM08] study the convergence of Q-learning when the environment changes but remain close to a fixed MDP. Assuming full knowledge of the MDP at each time step, but with unknown evolution, Lecarpentier and Rachelson [LR19] introduce a risk-averse approach to planning in slowly changing environments. In a related

[6] For instance, when $d = 1$ and $\kappa \to \infty$, their bound approaches $T^{\frac{2}{3}}$, improving the previous bound of $T^{\frac{3}{4}}$.

setting, Lykouris et al. [Lyk+19] study episodic RL problems where the MDP can be corrupted by an adversary and provide regret bounds in this case.

Chapter 5

Exploration without Rewards & Applications to Deep RL

In the previous chapters, we studied algorithms for regret minimization in finite or continuous MDPs and we saw that adding exploration bonuses to the rewards is an effective method to balance exploration and exploitation. In some environments, the reward function is *sparse*: for most states $s \in \mathcal{S}$, we have $r(s, \cdot) = 0$, so that the agent must be able to explore even in the absence of rewards from the environment. Additionally, it might be useful for some applications to design exploratory agents whose goal goes beyond reward maximization. Hence, we investigate in this chapter algorithms for *reward-free exploration*. We start from finite MDPs, and review the `RF-UCRL` [Kau+21] and the `RF-Express` [Mé+21a] algorithms showing that bonuses that decay as the number of visits $n_h^t(s, a)$ grows — similar to those used by `UCBVI`— also allow an agent to explore efficiently in the reward-free case. Then, we propose a deep reinforcement learning approach to reward-free exploration, based on a learned kernel combined with an approximate version of the exploration bonuses used by `Kernel-UCBVI`.

This chapter is based on the papers [Kau+21; Mé+21a] about reward-free exploration in finite MDPs, in which I participated as a collaborator, and on the workshop paper [Dom+21e] about reward-free exploration for deep reinforcement learning.

Contents

5.1 Reward-Free Exploration in Finite MDPs 86
5.2 Kernel-Based Bonuses for Exploration in Deep RL 90
5.3 Related Work 94
5.4 Experiments 96
5.5 Discussion and Bibliographical Remarks 104

5.1 Reward-Free Exploration in Finite MDPs

In several situations, a reinforcement learning agent needs to explore the environment in the absence of rewards. For instance, in sparse-reward environments, where the reward function is such that $r(s,\cdot) = 0$ for most states $s \in \mathcal{S}$, the agent might spend most of its time exploring without any reward feedback from the environment. Another example is when there is not a single reward function that is specified before the interaction with the environment, so that the agent is required to learn a transition model of the MDP, which can be latter used by planning algorithms to find near-optimal policies for any possible reward function.

Without a reward feedback, a key question is to decide what the agent's objective should be. Here, we consider the finite-horizon reward-free exploration (RFE) problem as proposed by Jin et al. [Jin+20a]. An algorithm for RFE should be able to generate a dataset $\mathcal{D}_\tau$ containing τ reward-free trajectories of horizon H,

$$\mathcal{D}_\tau := \left\{ (s_h^t, a_h^t, s_{h+1}^t)_{t=1}^{\tau}, \quad \text{for } h \in \{1, \dots, H\} \right\},$$

with τ being as small as possible, such that the empirical transition model $\widehat{p}^{\tau}$ built with the trajectories $\mathcal{D}_\tau$ can be used to compute near-optimal policies for any reward function with high probability. Intuitively, this requires the agent to explore the environment aiming to collect data that is as *diverse* as possible, covering all reachable states and enabling it to estimate an accurate transition model everywhere. More precisely, let $\widehat{\pi}_{\tau,r}$ be the optimal policy in the MDP with transitions $\widehat{p}^{\tau}$ and reward function r. Without loss of generality, assume that the initial state in each episode is fixed, *i.e.*, $s_1^t = s_1 \in \mathcal{S}$.[1] An algorithm is said to be (ε, δ)-PAC for RFE [Jin+20a; Kau+21] if

$$\mathbf{P}\left[\text{for all reward function } r, \ \left| V_1^*(s_1, r) - V_1^{\widehat{\pi}_{\tau,r}}(s_1; r) \right| \le \varepsilon \right] \ge 1 - \delta,$$

where $V_1^{\pi}(\cdot; r)$ is the value function of a policy π with the reward r, and $V_1^*(\cdot; r)$ is the optimal value function with respect to the reward function r.

Recall that proving regret bounds for the `UCBVI` algorithm (Section 3.6) does not require any assumption regarding the sparsity of the reward function. Hence, `UCBVI` is able to explore even in environments with sparse rewards, and can only start exploiting once states with non-zero rewards are encountered. `UCBVI`'s ability to explore comes from its exploration bonuses: thus, we might wonder if the same kind of bonuses can be used for reward-free exploration. This is indeed the case: Kaufmann et al. [Kau+21] introduce `RF-UCRL`, an algorithm similar to `UCRL` [JOA10] and `UCBVI` [AOM17] —relying on value iteration with exploration bonuses of the form $1/\sqrt{n}$— and show that it is (ε, δ)-PAC for RFE.

[1] As observed by Fiechter [Fie94], this is easily generalized to any initial state distribution $s_1 \sim \mu$ by considering a fixed initial state s_0 with a single action a_0 such that the reward at (s_0, a_0) is zero and $p_0(\cdot|s_0, a_0) = \mu$.

RF-UCRL is described in Algorithm 5.1 and is based on upper bounds $E_h^t(s,a)$ on the estimation error of the value of any policy for any reward function, which we now define. For any policy π, reward function r and time (t,h), let

$$e_h^{t,\pi}(s,a;r) := \left|\widehat{Q}_h^{t,\pi}(s,a;r) - Q_h^{\pi}(s,a;r)\right|,$$

where $Q_h^{\pi}(s,a;r)$ is the Q-function of a policy π in the MDP with the reward function r and the true transitions p_h, and where $\widehat{Q}_h^{\pi}(s,a;r)$ is the Q-function of π in the MDP with reward r and the estimated transitions $\widehat{p}_h^t$:

$$\widehat{p}_h^t(z|s,a) = \frac{n_h^t(s,a,z)}{n_h^t(s,a)}, \text{ if } n_h^t(s,a) > 0, \text{ and } \widehat{p}_h^t(z|s,a) = \frac{1}{S} \text{ otherwise,}$$

where $n_h^t(s,a,z) = \sum_{i=1}^{t-1} \mathbb{1}\left\{(s,a,z) = (s_h^i, a_h^i, s_{h+1}^i)\right\}$ and $n_h^t(s,a) = \sum_{z\in\mathcal{S}} n_h^t(s,a,z)$.

We define $E_h^t(s,a)$ for any (s,a,h,t) as

$$E_h^t(s,a) = \min\left(H, \mathtt{b}_h^t(s,a) + \sum_z \widehat{p}_h^t(z|s,a) \max_b E_{h+1}^t(z,b)\right), \tag{5.1}$$

where $E_{H+1}^t := 0$ and $\mathtt{b}_h^t$ is an exploration bonus given by

$$\mathtt{b}_h^t(s,a) := \sqrt{\frac{2H^2\beta\left(n_h^t(s,a),\delta\right)}{n_h^t(s,a)}}$$

for a threshold function $\beta(n,\delta)$ which is given as input to the algorithm.

Algorithm 5.1: RF-UCRL

initialize dataset $\mathcal{D} \leftarrow \emptyset$, $t \leftarrow 1$
while true do
  get initial state s_1^t
  # compute upper bounds on the error
  compute $(E_h^t)_{h\in[H]}$ according to Equation (5.1)
  # check stopping time
  if $\max_a E_1^t(s_1^t,a) \le \varepsilon/2$ **then**
    $\tau \leftarrow t$, $\mathcal{D}_\tau \leftarrow \mathcal{D}$,
    return: dataset $\mathcal{D}_\tau$
  for *stage* $h \in \{1,\dots,H\}$ **do**
    # select and execute action
    $a_h^t \leftarrow \operatorname{argmax}_a E_h^t(s_h^t,a)$, $s_{h+1}^t \leftarrow \mathtt{OnlineModel}_{t,h}(a_h^t)$
    # store data
    $\mathcal{D} \leftarrow \mathcal{D} \cup (s_h^t, a_h^t, s_{h+1}^t)$
  $t \leftarrow t+1$

It can be shown that $\sup_{r,\pi} e_h^{t,\pi}(s,a;r) \leq E_h^t(s,a)$ *for any* (t,h,s,a) with probability at least $1-\delta$ [Kau+21, Lemma 3]. That is, E_h^t provides an upper bound on the estimation error on the value function of any policy for any reward function. Then, at any state s at time (t,h), `RF-UCRL` chooses the action a_h^t with the highest estimation error upper bound: $a_h^t \in \operatorname{argmax}_a E_h^t(s_h^t, a)$. Intuitively, this action-selection strategy allows the algorithm to uniformly reduce the estimation of all policies for any possible reward function. Then, the algorithm stops when this estimation error at step $h=1$ is smaller than $\varepsilon/2$. Theorem 5.1, proved in [Kau+21], shows that `RF-UCRL` is (ε,δ)-PAC for reward-free exploration and provides a high-probability bound on the number of episodes τ executed by the algorithm before it stops.

Theorem 5.1 (Theorem 5 by [Kau+21])**.** `RF-UCRL` *using the threshold function*

$$\beta(n,\delta) = \log\left(\frac{2HSA}{\delta}\right) + (S-1)\log\left(e\left(1+\frac{n}{S-1}\right)\right)$$

is (ε,δ)*-PAC for reward-free exploration. Moreover, with probability* $1-\delta$*, the number* τ *of episodes before the algorithm stops satisfies*

$$\tau \lesssim \left(\frac{H^4SA}{\varepsilon^2}\right)\left(S+\log\left(\frac{1}{\delta}\right)\right),$$

where $\lesssim$ *omits factors depending on* $\log(1/\varepsilon)$ *and* $\log\log(1/\delta)$*.*

Jin et al. [Jin+20a] show that the lower bound on the number of episodes τ required for an algorithm to be (ε,δ)-PAC for RFE is $\Omega\left(H^2S^2A/\varepsilon^2\right)$, considering time-homogeneous MDPs. Kaufmann et al. [Kau+21] show that the upper bound on τ given in Theorem 5.1 can be improved by a factor of H if the transitions are time-homogeneous, matching the lower bound of [Jin+20a] up to a factor H.

For time-inhomogeneous MDPs, Ménard et al. [Mé+21a] propose another algorithm for reward-free exploration, called `RF-Express`, which is similar to `RF-UCRL`, but uses exploration bonuses of the form $1/n$ (instead of $1/\sqrt{n}$) and a different stopping time τ. By providing a novel empirical Bernstein inequality, Ménard et al. [Mé+21a] prove that `RF-Express` improves the upper bound of `RF-UCRL` by a factor of H. A similar technique had been previously used to improve the regret of `UCBVI` [AOM17] using a Bellman-type equation for the variances of value functions (which depend on r), but extending such technique to RFE is considerably more challenging, since the agent does not receive rewards and, consequently, cannot compute empirical variances.

Algorithmically, `RF-Express` proceeds as `RF-UCRL`, but replaces the error upper bound E_h^t by another quantity W_h^t that can be used to bound the estimation errors, defined as

$$W_h^t(s,a) = \min\left(H, \mathtt{b}_h^t(s,a) + \left(1+\frac{1}{H}\right)\sum_z \widehat{p}_h^t(z|s,a)\max_b E_{h+1}^t(z,b)\right),$$

where the bonus is given by

$$\mathtt{b}_h^t(s,a) := \frac{15H^2\beta\left(n_h^t(s,a),\delta\right)}{n_h^t(s,a)}.$$

At every time (t,h), `RF-Express` chooses the action $a_h^t \in \operatorname{argmax}_a W_h^t(s_h^t,a)$, and stops at the episode t if the condition $3\exp(1)\sqrt{\max_a W_1^t(s_1,a)} + \max_a W_1^t(s_1,a) \le \varepsilon/2$ is met. Such stopping condition is justified by the fact that the estimation error $e_h^{t,\pi}$ is bounded as

$$\sup_{r,\pi} e_h^{t,\pi}(s_1,\pi_1(s_1);r) \le 3\exp(1)\sqrt{\max_a W_1^t(s_1,a)} + \max_a W_1^t(s_1,a)$$

with high probability, as proved in [Mé+21a, Lemma 1]. For completeness, we restate below the theorem by Ménard et al. [Mé+21a] showing that `RF-Express` is (ε,δ)-PAC for RFE, and that it improves the sample complexity of `RF-UCRL` by a factor of H for time-inhomogeneous MDPs.

Theorem 5.2 (Theorem 1 by [Mé+21a])**.** `RF-Express` *using the threshold function*

$$\beta(n,\delta) = \log\left(\frac{2HSA}{\delta}\right) + S\log\left(8e(n+1)\right)$$

is (ε,δ)-PAC for reward-free exploration. Moreover, with probability $1-\delta$, the number τ of episodes before the algorithm stops satisfies

$$\tau \lesssim \left(\frac{H^3SA}{\varepsilon^2}\right)\left(S + \log\left(\frac{1}{\delta}\right)\right),$$

where $\lesssim$ omits factors depending on $\log(1/\varepsilon)$ and $\log\log(1/\delta)$.

At first, the $1/n$ bonuses used by `RF-Express` might be surprising, when compared to the $1/\sqrt{n}$ bonuses used by `UCBVI` and `RF-UCRL`, for instance. Indeed, in order to estimate the mean μ of a random variable X by an estimator $\widehat{\mu}_n$ computed with n i.i.d. samples fom X, the error $|\mu-\widehat{\mu}_n|$ scales with $1/\sqrt{n}$ by Hoeffding's inequality, which explains the bonuses used by other algorithms. However, the analysis of `RF-Express` is based on concentration inequalities for the Kullback-Leibler divergence, and bound the term $(\mu-\widehat{\mu}_n)^2$, which scales with $1/n$.

5.2 Kernel-Based Bonuses for Exploration in Deep RL

In the previous section, we saw that using exploration bonuses depending on $n_t(s,a)$ allows an agent to explore finite MDPs even in a reward-free situation, where $n_t(s,a)$ is the number of visits to (s,a) up to time t. In Chapter 4, we proposed the `Kernel-UCBVI` algorithm that generalizes the counts $n_t(s,a)$ to continuous MDPs by using a kernel function, and analyzed its regret bound. Hence, it is reasonable to expect that, at least empirically, `Kernel-UCBVI` is also able to explore continuous MDPs in a reward-free situation. Indeed, in finite MDPs with an appropriate choice of the kernel function, `Kernel-UCBVI` becomes identical to `UCBVI`, which is similar to `RF-UCRL` in the sense that it is based on value iteration combined with $1/\sqrt{n}$ exploration bonuses.

Thus, in principle, `Kernel-UCBVI` is a very general algorithm for exploration: it makes very weak assumptions on the MDP and can be implemented as long as we have an online model of the MDP and a kernel function. Nevertheless, it has two main practical limitations: (i) its computational complexity that either increases with the time t in its exact version, or scales with the covering number of the state space in its approximate version with representative states, and (ii) the fact that it requires a well-designed kernel function.

In this section, in order to tackle those limitations of `Kernel-UCBVI`, we propose a method that we call `AKBX`, meaning Approximate Kernel-Based eXploration. `AKBX` is a method that (i) replaces `Kernel-UCBVI`'s backward induction by an off-policy RL algorithm using function approximation; (ii) learns a representation that is used to define a kernel; and (iii) based on the learned representation, computes an approximation of `Kernel-UCBVI`'s exploration bonuses. We remark that there are several works aiming to generalize $1/\sqrt{n}$ bonuses to deep RL, some of which are discussed and compared to `AKBX` in Section 5.3.

Algorithm 5.2: Simplified structure of RL algorithms based on exploration bonuses

initialize data buffer $\mathcal{D} \leftarrow \emptyset$
get initial state s_1
for $t = 1, \dots, T$ **do**
    Define exploration bonuses $\mathtt{b}_t(s,a)$ based on data $\mathcal{D}$
    Using $\mathtt{b}_t$ and past data $\mathcal{D}$, compute a policy π_t
    Execute the action $a_t \sim \pi_t(s_t)$
    Observe next state s_{t+1} and reward r_t (if reward-free, $r_t = 0$)
    Update data $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, s_{t+1}, r_t)$

Consider Algorithm 5.2, which describes in a simplified manner the structure of the RL algorithms based on exploration bonuses that we have studied so far. `AKBX` follows this structure, and we assume that the computation of the policy π_t, given the past data and the bonuses that are added to the rewards, is done by any off-policy RL algorithm with function approximation,

such as Fitted Q-Learning [EGW05; Rie05; MS08] and Deep Q-Learning [Mni+13]. In fact, any offline RL algorithm could be used, see Levine et al. [Lev+20] for a survey on offline RL. Thus, we are left with the problem of defining a kernel and approximating the bonuses of `Kernel-UCBVI`.

Kernel design through representation learning In cases where we do not have enough domain knowledge to design a good kernel function, we can use representation learning methods that, for each (s, a), learn a representation function f that maps (s, a) to a real-valued vector $f(s, a) \in \mathbb{R}^d$. Different representation learning algorithms have been proposed for RL, designed to encourage certain properties in the function f, such as learning representations that ignore uncontrollable features of the environment, that are useful to build transition models, or that allow generalization across tasks [Pat+17; Aza+19; Guo+20; Bad+20b]. Given a learned representation function f, we can use the Euclidean distance in the representation space to define a metric ρ on $\mathcal{S} \times \mathcal{A}$ as $\rho\left[(s, a), (s', a')\right] = \left\|f(s, a) - f(s', a')\right\|_2$, and define a kernel function $\Gamma_f((s, a), (s', a'))$ as

$$\Gamma_f((s, a), (s', a')) = \psi\left(\left\|f(s, a) - f(s', a')\right\|_2\right) \tag{5.2}$$

where $\psi : \mathbb{R}_+ \to \mathbb{R}_+$ is a non-increasing function. For instance, a Gaussian kernel is obtained by taking $\psi(x) = \exp(-x^2/2)$.

Approximation of exploration bonuses Given the kernel Γ, our goal now is to approximate the kernel-based bonuses introduced in Section 4.1. We start with the following simplification of those bonuses:

$$\mathtt{b}^{t+1}(s, a) = \left(\frac{1}{\beta + \sum_{i=1}^{t} \Gamma_f((s, a), (s_i, a_i))}\right)^{\alpha}, \quad \alpha \in \left\{\frac{1}{2}, 1\right\}, \tag{5.3}$$

where β is a regularization constant, and α can be either $1/2$, as in the `Kernel-UCBVI` bonuses in Section 4.1, or $\alpha = 1$, which generalizes the $1/n$ bonuses of `RF-Express`. It is necessary to approximate $\mathtt{b}^{t+1}(s, a)$ because its computation requires $\mathcal{O}(t)$ time, and we would like an algorithm whose time complexity does not grow with the number of transitions t. We propose an approximation of the function $\mathtt{b}^{t+1}$ by a function g_θ that is updated at every time t by taking gradient steps to minimize a loss function. In order to do so, we first rewrite $\mathtt{b}^{t+1}$ as:

$$\mathtt{b}^{t+1}(s, a) = \frac{1}{t^\alpha}\left(\frac{1}{\beta/t + \mathbf{E}_B\left[\frac{1}{|B|}\sum_{i \in B} \Gamma_f((s, a), (s_i, a_i))\right]}\right)^{\alpha},$$

where B is a batch of indices sampled uniformly at random (with replacement) from $\{1, \ldots, t\}$.

Now, we consider the following loss function:

$$\ell_t(s,a,f,B,\theta) := \left(g_\theta(s,a) - \left(\frac{1}{\varepsilon_t + \frac{1}{|B|}\sum_{i\in B} \Gamma_f((s,a),(s_i,a_i))} \right)^\alpha \right)^2, \tag{5.4}$$

where ε_t is a regularization parameter, and define the expected loss over a distribution ν_t on $\mathcal{S}\times\mathcal{A}$ and the batch B:

$$\ell_t(\theta) := \mathbf{E}_{(s,a)\sim\nu_t}\left[\mathbf{E}_B\left[\ell_t(s,a,f,B,\theta)\right]\right]. \tag{5.5}$$

In practice, we take the distribution ν_t such that (s,a) is sampled uniformly at random from $(s_i,a_i)_{i=1}^t$, the data available at time t. The parameters θ defining the function g_θ are then updated at time t by taking stochastic gradient descent steps to minimize the loss $\ell_t(\theta)$.

The intuition behind the loss in Equation (5.4) is the following: it is a regression problem where $g_\theta(s,a)$ is optimized to predict the expected value of $\left(\varepsilon_t + \frac{1}{|B|}\sum_{i\in B}\Gamma_f((s,a),(s_i,a_i))\right)^{-\alpha}$, which satisfies, by Jensen's inequality:

$$\begin{aligned}
&\mathbf{E}_B\left[\left(\frac{1}{\varepsilon_t + \frac{1}{|B|}\sum_{i\in B}\Gamma_f((s,a),(s_i,a_i))}\right)^\alpha\right] \\
&\geq \left(\frac{1}{\varepsilon_t + \mathbf{E}_B\left[\frac{1}{|B|}\sum_{i\in B}\Gamma_f((s,a),(s_i,a_i))\right]}\right)^\alpha = t^\alpha \mathfrak{b}^{t+1}(s,a),
\end{aligned} \tag{5.6}$$

if we take $\varepsilon_t := \beta/t$. Hence, $g_\theta(s,a)$ is trained to predict a term that is proportional to an upper bound on the true bonuses $\mathfrak{b}^{t+1}(s,a)$.

Finally, we propose the following approximate bonuses for AKBX, which we explain next:

$$\widetilde{\mathfrak{b}}_\theta^{t+1}(s,a) = \lambda_{1,t}\, g_\theta(s,a) + \lambda_{2,t}\sqrt{\ell_t(s,a,f,B_t,\theta)}, \tag{5.7}$$

where ℓ_t is the loss defined in Equation (5.4), B_t is a batch of indices sampled uniformly at random from $\{1,\dots,t\}$, and $(\lambda_{1,t},\lambda_{2,t})$ are positive factors that may depend on t.

Algorithm 5.3 summarizes the AKBX method, which trains a representation function f, constructs a kernel based on f, defines approximate kernel-based bonuses as in Equation (5.7), and trains a policy π_t using an off-policy RL algorithm in order to maximize the sum of bonuses and rewards (which are 0 in the reward-free case). Below, we provide further explanation regarding the bonuses that we propose in Equation (5.7).

First term of approximate bonuses (5.7) The term $\lambda_{1,t}\ g_\theta(s,a)$ comes from the fact that $g_\theta(s,a)$ is trained to predict a term that is *proportional* to an upper bound on the true bonuses

Algorithm 5.3: AKBX: Approximate Kernel-Based Exploration

initialize data buffer $\mathcal{D} \leftarrow \emptyset$
get initial state s_1
for $t = 1, \dots, T$ **do**
  Sample two batches B_{t-1} and B'_{t-1} from $\{1, \dots, t-1\}$
  Update the representation function f using the batch $(s_i, a_i, s_{i+1})_{i \in B_t}$
  Update the bonus parameters θ using $\nabla_\theta \ell_t(\theta)$, where

$$\ell_t(\theta) := \frac{1}{|B'_{t-1}|} \sum_{j \in B'_{t-1}} \ell_t(s_j, a_j, f, B_{t-1}, \theta)$$

  Define exploration bonuses $\widetilde{\mathtt{b}}^t_\theta$ as in Equation (5.7)
  Using $\widetilde{\mathtt{b}}^t_\theta$ and past data $\mathcal{D}$, compute a policy π_t that optimizes bonuses + rewards
  Execute the action $a_t \sim \pi_t(s_t)$
  Observe next state s_{t+1} and reward r_t (if reward-free, $r_t = 0$)
  Update data $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, s_{t+1}, r_t)$

(5.6). The time-dependent value $\lambda_{1,t}$ can be chosen in order to control *how quickly the bonuses decay to* 0. For instance, the choice $\lambda_{1,t} = 1/t^\alpha$ will make the term $\lambda_{1,t}\, g_\theta(s,a)$ approximate an upper bound on the exact bonuses from Equation (5.3). However, the off-policy RL algorithm used to compute the policy π_t (which aims to maximize the sum of bonuses and rewards) is usually trained by taking gradient steps: consequently, if the bonuses decay too quickly, the policy might not have been updated enough in order to be able to visit states where the bonuses were high. Choosing a $\lambda_{1,t}$ that decays slower than $1/t^\alpha$ can be useful to adjust the time scales between bonus computation and policy optimization. In some cases, we can even choose $\lambda_{1,t}$ to be constant (that is, independent of t): this is especially relevant in reward-free situations, where the scale of the bonuses is not important, as long as the bonuses are high in the least visited states. Notice that, if we ignore the clipping to H in Equation (5.1), which defines the error upper bounds $E^t_h(s,a)$ used to define RF-UCRL's policy, we can replace $\mathtt{b}^t_h$ by $\lambda_{1,t}\mathtt{b}^t_h$ for any $\lambda_{1,t}$, and the policy $\pi^t_h(s) = \operatorname{argmax}_a E^t_h(s,a)$ remains unchanged. Additionally, ignoring such clipping in RF-UCRL has been observed to improve its empirical performance [Kau+21].

Second term of approximate bonuses (5.7) We add to the bonuses the following term, which is proportional to the absolute error between the targets $\left(\varepsilon_t + \frac{1}{|B|}\sum_{i \in B} \Gamma_f((s,a),(s_i,a_i))\right)^{-\alpha}$ and the prediction $g_\theta(s,a)$:

$$\lambda_{2,t} \sqrt{\ell_t(s,a,f,B_t,\theta)},$$

where ℓ_t is defined in Equation (5.4). This is motivated by the fact that, if the loss ℓ_t is high, the approximation made by g_θ might not be good enough to encourage the agent to explore. Hence, we add to the bonuses a term proportional to the loss, so that the agent is able to explore even before g_θ has converged to a meaningful function: that is, a function whose values are large

for the least visited state-action pairs. Notice that the loss $\sqrt{\ell_t(s, a, f, B_t, \theta)}$ should be higher if states similar to (s, a) have not been visited before. Also, since the parameters θ are trained to minimize the expectation of $\ell_t(s, a, f, B_t, \theta)$, the loss should decrease in expectation as g_θ is trained, and the first term of the bonus $(\lambda_{1,t}\ g_\theta(s, a))$ becomes the dominant term.

Using nearest neighbors to improve bonus approximation In practice, we observed that replacing the targets

$$\left(\frac{1}{\varepsilon_t + \frac{1}{|B|}\sum_{i\in B} \Gamma_f((s, a), (s_i, a_i))}\right)^\alpha$$

in the loss (5.4) by

$$\left(\frac{1}{\varepsilon_t + \frac{1}{k}\sum_{(s',a')\in \mathrm{NN}_k(s,a,B)} \Gamma_f((s, a), (s', a'))}\right)^\alpha$$

where $\mathrm{NN}_k(s, a, B)$ is the subset of $(s_i, a_i)_{i\in B}$ containing the k-nearest neighbors of (s, a) with respect to the distance $\rho\left[(s, a), (s', a')\right] = \|f(s, a) - f(s', a')\|_2$, results in a more robust algorithm with respect to the hyperparameters. This is inspired by the kernel-based bonuses used by the `Never-Give-Up` algorithm by Badia et al. [Bad+20b], which we discuss in Section 5.3. One possible explanation for such improvement is that using k-nearest neighbors might reduce the impact of outliers, which is important since the representation f is being learned at the same time as the approximate bonuses.

5.3 Related Work

Several techniques for exploration in deep reinforcement learning take inspiration from the $1/\sqrt{n}$-bonuses used by near-optimal algorithms in finite MDPs. For instance, Bellemare et al. [Bel+16] propose a method to compute pseudo-counts approximating $n_t(s, a)$ using density estimation on images, and Tang et al. [Tan+17] use locality-sensitive hashing to map continuous states to discrete representations, where explicit counts are computed. A common property across these techniques is that the more a state-action pair (s, a) is visited, the smaller the bonus at (s, a). Such property is also satisfied by algorithms such as Random Network Distillation (`RND`) by Burda et al. [Bur+19] and `Never-Give-Up` by Badia et al. [Bad+20b]. `Never-Give-Up` relies on kernel-based exploration bonuses and, among the related work, we believe it is the closest to `AKBX`. Since `Never-Give-Up` also uses `RND`, we briefly explain below how both algorithms construct their exploration bonuses.

Random Network Distillation The exploration bonuses used by `RND` are based on functions u_θ and $u_{\theta'}$, implemented by two neural networks with *identical architecture*, that take as input a state $s \in \mathcal{S}$ and output a real-valued vector. The network parameters θ, θ' are initialized randomly and, whereas θ' is kept fixed, θ is slowly updated with gradient steps that minimize

the loss

$$\ell_{\texttt{RND}}(\theta) := \mathbf{E}_{s\sim\nu_t}\left[\|u_\theta(s) - u_{\theta'}(s)\|_2^2\right], \tag{5.8}$$

where ν_t is taken as a uniform distribution on the previously visited states $(s_i)_{i=1}^t$. Then, the bonuses used by RND are

$$\mathtt{b}^{t+1}_{\texttt{RND},\theta}(s) := \|u_\theta(s) - u_{\theta'}(s)\|_2^2. \tag{5.9}$$

Intuitively, the RND loss will be higher for the least visited states, and the bonus will thus encourage the agent to visit new states. As we will see in the experiments of Section 5.4, a disadvantage of RND in reward-free exploration is that, at some point, its bonuses tend to zero[2], which makes the agent stop exploring. Never-Give-Up, which we explain below, avoids this issue by using non-vanishing kernel-based bonuses.

Never-Give-Up The Never-Give-Up algorithm also relies on a kernel function defined on top of a learned representation function f. Let $\mathbf{clip}_{[a,b]}(x) := \min(b, \max(a, x))$, and c and L be positive constants. For a state s_t visited at time t, the Never-Give-Up exploration bonus is defined as

$$\mathtt{b}^{t+1}_{\texttt{NGU}}(s_t) := \mathbf{clip}_{[1,L]}\left(1 + \mathtt{b}^{t+1}_{\texttt{RND},\theta}(s_t)\right) \times \frac{1}{\sqrt{\sum_{s'\in \mathrm{NN}_k^{\text{episodic}}(s_t,B)} \Gamma_f(s_t, s')} + c},$$

where $\mathrm{NN}_k^{\text{episodic}}(s_t, B)$ is the set of k-nearest neighbors of s_t among the states $(s_i)_{i=1}^{t-1}$ that are *in the same episode as* s_t, and $\Gamma_f : \mathcal{S}\times\mathcal{S} \to \mathbb{R}_+$ is a kernel function on the state-space, computed using the distance $\|f(s) - f(s')\|_2$ based on a state-representation function f.

The first factor in the Never-Give-Up bonus is based on RND, and encourages the agent to visit new states. However, as the RND bonus tend to zero, the first factor becomes approximately 1, and the agent is left with the kernel-based bonus in the second factor, which gives high bonuses to the agent if it visits different states *in each single episode*. Since the kernel-based factor does not tend to zero, an agent using the Never-Give-Up bonuses can keep exploring even after the RND bonuses vanish, which was shown by Badia et al. [Bad+20b] to greatly improve an agent's performance in Atari games that are difficult in terms of exploration.

Relation to AKBX We see that the kernel-based bonus of AKBX in Equation (5.7), which is an approximate version of Kernel-UCBVI's bonus, is similar in principle to Never-Give-Up, in the sense that it encourages the agent to visit novel states with respect to a learned representation. However, the kernel-based factor in the Never-Give-Up bonus is computed using data only *from a single episode* and, algorithmically, it requires us to keep a buffer containing data from the

[2]The minimum loss is indeed zero, since both RND networks have the same architecture.

current episode. AKBX is simpler to implement, since it relies only on sampling batches (and does not need to keep track of all data generated in the current episode), and can be directly related to Kernel-UCBVI, a theoretically-grounded algorithm. A possible advantage of the intra-episode bonuses of Never-Give-Up is that they might be more adapted to environments with very long episodes, which is the case of some Atari games, where explicitly requiring diversity in the states visited in a single episode might be beneficial for exploration. Futhermore, the loss term added to the AKBX bonuses has a similar interpretation to RND, since the loss at the least visited states is higher, and the agent receives high bonuses for visiting those states.

5.4 Experiments

In this section, we provide experiments that serve as a proof of concept of AKBX. The AKBX approach involves the interaction between three learning procedures: (i) learning a representation function f; (ii) learning approximate exploration bonuses by minimizing the loss (5.5); and (iii) training a reinforcement learning algorithm to maximize the sum of reward and exploration bonuses (or only the bonuses in a reward-free case). Each procedure depends on the others: in order to learn a good representation, the agent needs to explore well the MDP; and, to explore, the agent needs a good representation and a reinforcement learning algorithm that is able to optimize the sum of exploration bonuses, which are non-stationary (since the bonuses are being constantly updated). Hence, we designed experiments that verify that AKBX is able to explore even in a scenario with such complex learning interactions.

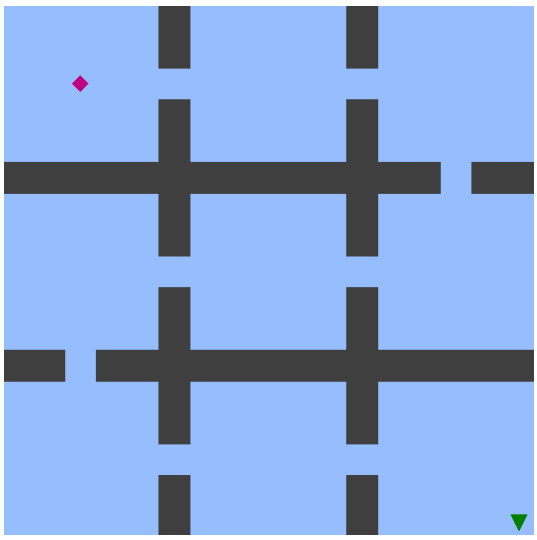


Figure 5.1 – Grid-World with 9 rooms. The number of states is $S = 233$ (not counting the walls), and the agent's observations are one-hot encodings of the discrete states.

We consider the Grid-World environment illustrated in Figure 5.1. The state space is discrete, but AKBX's assumes continuous observations, as it observes one-hot encodings of the discrete

states. That is, for a state $s \in \mathcal{S} := \{1, \dots, S\}$, the AKBX agent observes a vector $s_{\text{obs}} \in \mathbb{R}^S$ such that $s_{\text{obs}}(i) = \mathbb{1}\{i = s\}$, where $s_{\text{obs}}(i)$ denotes the i-th component of the vector s_{obs}. There are four actions available, corresponding to movements in the four possible directions (up, down, left, right). The agent starts at the top-left room and when choosing an action, the agent moves in the corresponding direction with probability 0.95, and moves in a random direction with probability 0.05. In a reward-free situation, the goal of the agent is to constantly explore all the states. When rewards from the environment are enabled, the agent receives a reward equal to 1 in the state in the bottom-right corner.

Before illustrating AKBX's performance in a reward-free situation (Section 5.4.3) and with rewards (Section 5.4.4), we present in Section 5.4.1 a simple representation learning method inspired by manifold learning, and show in Section 5.4.2 that the approximate bonuses used by AKBX (5.7) are able to mimic the $1/\sqrt{n}$ and $1/n$ bonuses in tabular MDPs.

State-dependent representation and bonuses As usually done in related work on exploration for deep RL (*e.g.*, [Bel+16; Bad+20b]), we consider that the representation function f is a function of states only (instead of state-action pairs) and, similarly, the kernel function Γ_f depends only on pairs of states, that is

$$\Gamma_f((s,a),(s',a')) = \Gamma_f(s,s') = \psi\left(\left\|f(s) - f(s')\right\|_2\right).$$

The intuition is that the agent is encouraged to visit novel states (instead of novel state-action pairs), and the exploration among the possible actions is done by introducing randomness in the exploration policy: for instance, by ε-greedy exploration (in Q-learning algorithms) or by entropy regularization (in actor-critic algorithms).

Notation Throughout this section, we denote by ν_t the distribution on $\mathcal{S}$ such that $s \sim \nu_t$ is sampled uniformly at random (with replacement) from $(s_i)_{i=1}^t$, the data available at time t.

5.4.1 Representation Learning

Since the agent receives one-hot encodings of discrete states, its inputs are vectors of dimension $S = 233$ without any structure (that is, the Euclidean distance between any pair of observations is the same). In order to learn a meaningful representation function f, we minimize the following loss:

$$\begin{aligned}\ell_t^{\text{repr}}(f) :=& \mathbf{E}_{s_i \sim \nu_t}\left[\ell_{\text{Huber}}(f(s_i) - f(s_{i+1}))\right] \\ &+ c_1 \mathbf{E}_{s\sim\nu_t}\left[f(s)\right] + c_2 \mathbf{E}_{s\sim\nu_t}\left[\|f(s)\|_2^2\right] - c_3 \mathbf{E}_{s,s'\sim\nu_t}\left[\ell_{\text{Huber}}(f(s) - f(s'))\right] \qquad (5.10)\end{aligned}$$

where c_1, c_2 and c_3 are positive constants, and ℓ_{Huber} is a pseudo-Huber loss with parameters $\delta > 0$ and $q > 0$ defined as

$$\ell_{\text{Huber}}(x) := (\delta^q + \|x\|_2^q)^{1/q} . \tag{5.11}$$

The first term in the loss $\ell_t^{\text{repr}}(f)$ forces the distance between consecutive states s_t and s_{t+1} to be small in the representation space. The term proportional to c_1 means that f should be orthogonal to a constant function, and avoids constant solutions, whereas the term proportional to c_2 penalizes functions f with large norm. These first three terms are analogous to the Laplacian Eigenmaps method for manifold learning [VMS16; BN03] given an adjacency graph. The analogy in our case is that the graph adjacency matrix is defined by connecting consecutive states s_t and s_{t+1}. The last term, proportional to c_3, is a constrastive loss that forces a separation in the embedding space between states s and s' that are sampled randomly. Besides its relation to Laplacian Eigenmaps, the loss $\ell_t^{\text{repr}}(f)$ and the use of the pseudo-Huber loss (5.11) is inspired by the adjacency regularization loss proposed by Guo et al. [Guo+21].

Table 5.1 – Parameters used for representation learning

Parameter	Value
Representation function class	`MLP` $(128, 64)$
Pseudo-Huber loss parameters	$q = 4, \delta = 1.0$
Representation loss parameters	$c_1 = c_2 = 10^{-9}, c_3 = 10^{-4}$
Optimizer (`Adam`) learning rate	10^{-4}
Batch dimensions	$(8, 32)$

To train a representation f, we parameterized f using a multilayer perceptron (`MLP`) with a hidden layer of size 128 and an output layer of size 64, meaning that the state embeddings are 64-dimensional vectors. We used batches of dimension $(8, 32)$, meaning that each batch has 8 sub-trajectories containing 32 time steps, and minimized the loss (5.10) using the `Adam` optimizer [KB14] and the `PyTorch` library [Pas+19]. Other parameters for the loss computation are given in Table 5.1.

In order to validate the representation learning method used here, we sampled 5×10^4 transitions from the Grid-World environment (Figure 5.1), corresponding to 5×10^2 trajectories of horizon $H = 100$. In order to remove exploration issues and validate *only the representation learning method*, the starting state of each trajectory was sampled uniformly at random among the centers of each room. Figure 5.2 shows the ground truth positions of each state in the Grid-World from Figure 5.1, and the 2-dimensional projection of the 64-dimensional embeddings $f(s)$ before and after minimizing the representation loss (5.10). The 2D projections were computed using the `t-SNE` algorithm [MH08] implemented in the `scikit-learn` library [Ped+11]. We can see that the representation learning method used here is able to capture the adjacency structure of the states of the Grid-World.

Figure 5.2 – **Left:** true positions of each state in the Grid-World with 9 rooms, each state is associated to a unique color. **Middle:** 2-dimensional projection of the 64-dimensional embeddings before minimizing the loss (5.10). **Right:** 2-dimensional projection of the embeddings after 15×10^3 optimization steps to minimize the representation loss (5.10).

5.4.2 Bonus Computation

Now that we have validated our representation learning method, we study AKBX's exploration bonuses, which are trained at the same time as the representation function f. We approximate the function g_θ, used to define the bonus (5.7), by an MLP with a hidden layer of size 128. The other parameters used to defined the bonuses are given in Table 5.2.

Table 5.2 – Parameters used to compute AKBX's bonuses

Parameter	Value
Function class for g_θ (5.7)	MLP $(128, 1)$
Base kernel function (5.2)	$\psi(x) = 1/(1 + x^2)$
Regularization constant	$\beta = 0$
Bonus parameters (5.7)	$\lambda_{1,t} = \lambda_{2,t} = 1$
Optimizer (Adam) learning rate	10^{-4}
Batch dimensions	$(8, 32)$

Figure 5.3 compares the number of state visits to the bonuses (5.7) learned by minimizing the loss in Equation (5.5) for $\alpha = 1/2$ and $\alpha = 1$, based on 5×10^2 trajectories of horizon $H = 100$ collected by playing a random policy starting from the top-left room. We ran 15×10^3 optimization steps to learn the representation f and 15×10^3 optimization steps to minimize the bonus loss Equation (5.5). Both the representation and the bonuses where trained together: after each block of 500 optimization steps for the representation, we ran 500 optimization steps for the bonuses. We observe that AKBX is able to estimate bonuses that behave either as $1/\sqrt{n}$ (for $\alpha = 1/2$) or $1/n$ (for $\alpha = 1$), even if bonuses and representation are trained simultaneously.

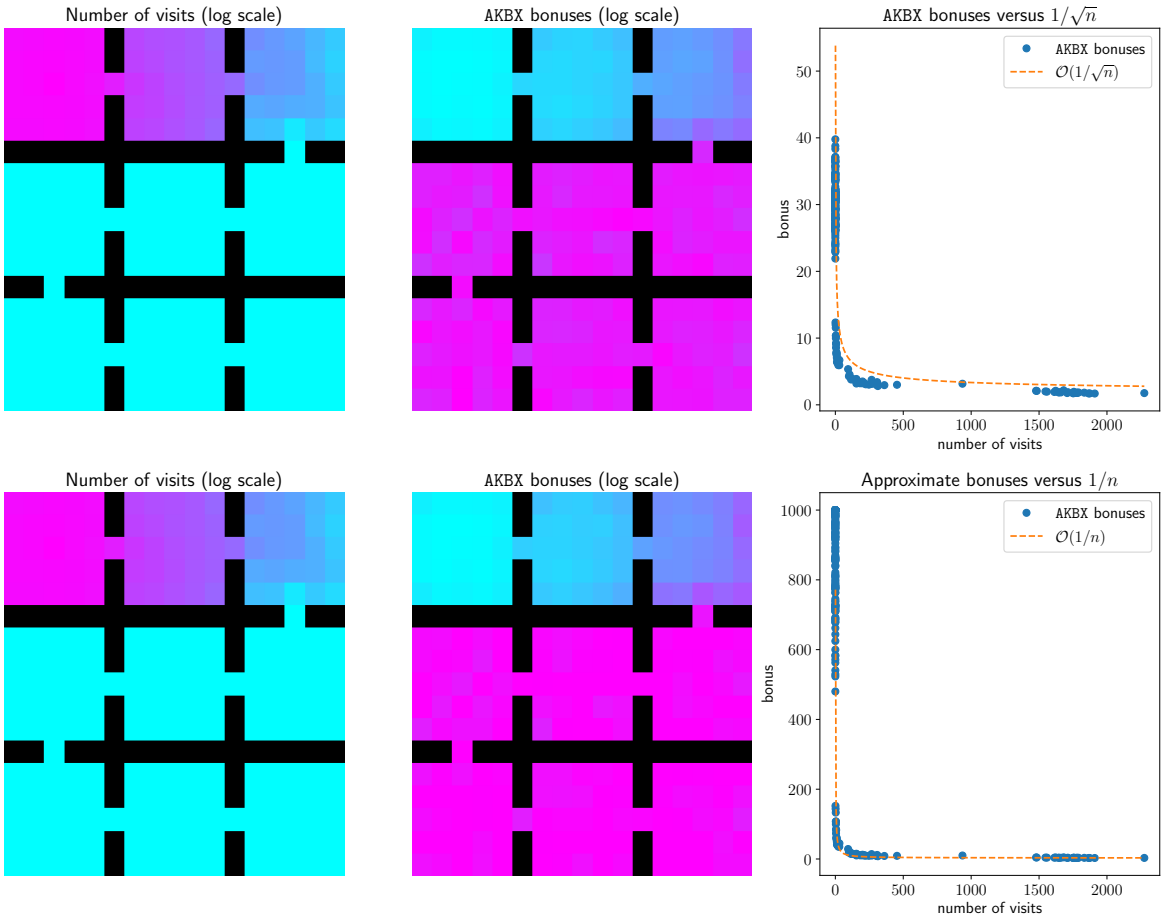


Figure 5.3 – Visualization of AKBX's bonuses for $\alpha = 1/2$ (**top**) and $\alpha = 1$ (**bottom**). **Left:** Visualization of number of visits of each state in the GridWorld, when random trajectories are sampled starting from the center of the top-left room. Higher values are shown in magenta and smaller values are shown in blue. **Middle:** Bonuses (5.7) estimated by AKBX based on the learned representation function f. **Right:** Comparison between AKBX's bonuses and $1/n^{\alpha}$, where n is the number of state visits.

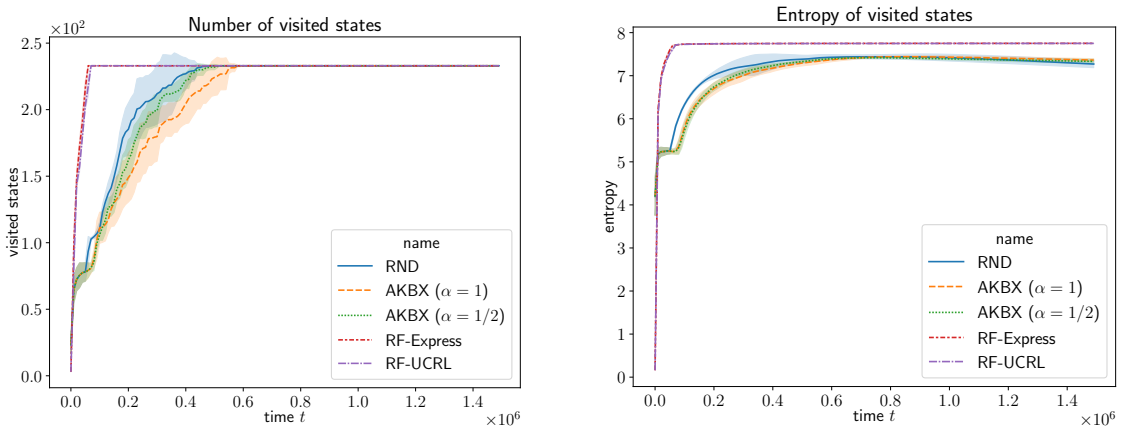
5.4.3 Reward-Free Exploration

In this section, we evaluate the performance of AKBX in a problem of reward-free exploration, and we compare it to RF-UCRL, RF-Express and RND. We use the Grid-World environment from Figure 5.1 and evaluate the algorithms by

(i) the number of visited states, which shows how quickly they are able to explore the environment; and

(ii) the entropy of the empirical state-visit distribution[3], which illustrates the diversity of the states visited by the exploration policy.

[3]That is, the entropy of the distribution $n_t(s)/t$, where $n_t(s)$ is the number of visits to the state s up to time t.



(**a**) Number of visited states.

(**b**) Entropy of the empirical state-visit distribution.

Figure 5.4 – Evaluation of reward-free exploration: number of visits and entropy of state-visit distribution in the 9-room Grid-World for `AKBX`, `RND`, `RF-UCRL` and `RF-Express`. Average over 4 independent runs.

As off-policy RL algorithm, used both by `AKBX` and `RND`, we implemented an actor-critic method called Advantage-Weighted Regression (`AWR`), introduced by Peng et al. [Pen+19]. We chose `AWR` due to its higher stability when compared to algorithms based on Q-learning. Table 5.3 shows the hyperparameters used in our `AWR` implementation. For `RND`, we used an `MLP` with two hidden layers of size 256 to implement the functions u_θ and $u_{\theta'}$ used to define its bonus (5.9), and the loss (5.8) was minimized with the `Adam` optimizer and a learning rate of 10^{-5}.

For `AKBX` and `RND`, the representation, the bonuses and `AWR` are trained in the following manner: every T_{update} time steps in the environment, we perform:

- N_{repr} optimization steps of representation learning;
- N_{bonus} optimization steps of minimization of the bonus loss function;
- N_{critic} optimization steps of `AWR`'s value function updates; and
- N_{actor} optimization steps of `AWR`'s policy updates.

In our implementation, we chose the values $T_{\text{update}} = 2048$, $N_{\text{critic}} = 200$, $N_{\text{actor}} = 1000$, as suggested for `AWR` by [Pen+19], and $N_{\text{repr}} = N_{\text{bonus}} = 500$.

Figure 5.4 shows the number of state visits and the entropy of state-visit distribution for `AKBX`, `RND`, `RF-UCRL` and `RF-Express`. We can see that `RF-UCRL` and `RF-Express` quickly discover all the states and maintain a high state-visit entropy. Also, both `AKBX` and `RND` are able to discover all the states and, apparently, maintain a reasonable level exploration if we look at their state-visit entropy. However, only looking at the entropy might be misleading, since it does not allow us to evaluate whether the agents keep consistently exploring all the rooms. Figure 5.5 shows for each algorithm the number of visits per state and for different time intervals. At

Table 5.3 – Parameters used in AWR implementation

Parameter	Value
Function class for policy (actor)	MLP (128, 64, $\lvert\mathcal{A}\rvert$)
Function class for value function (critic)	MLP (128, 64, 1)
Policy optimizer (Adam) learning rate	10^{-4}
Value optimizer (Adam) learning rate	10^{-2}
TD(λ) parameter	$\lambda = 0.95$
Replay buffer size (number of transitions)	3×10^5
Advantage weight parameter	$\beta_{\text{adv}} = 0.5$
Maximum advantage weight	40.0
Batch size	Full trajectory of length H

the beginning, we see that RND is able to explore all the rooms but, as the time t increases, its exploration bonuses become close to zero and the RND agent stops exploring the last rooms at the bottom. By setting $\lambda_{1,t} = 1$ in Equation (5.7), AKBX's bonuses do not tend to 0, and the agent is able to keep exploring all the rooms in the Grid-World.

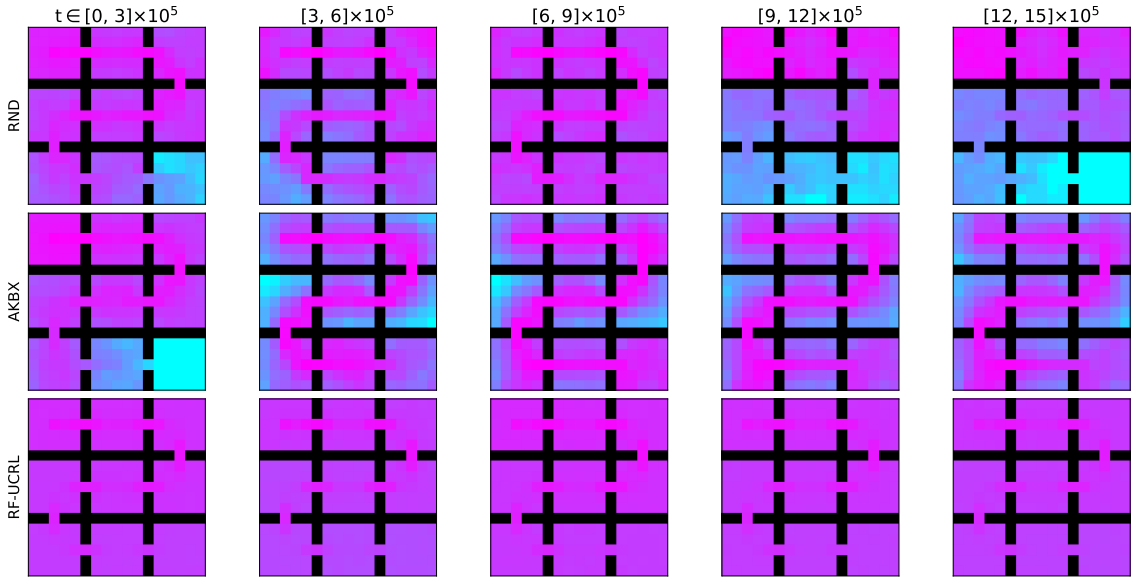


Figure 5.5 – Number of state visits (in logarithmic scale) for RND (top), AKBX for $\alpha = 1/2$ (middle) and RF-UCRL (bottom) in different time intervals. Each column corresponds to the number of visits in a given time interval. For instance, the first column shows state visits for between time $t = 0$ and $t = 3 \times 10^5$, whereas the last column shows the visits between $t = 12 \times 10^5$ and $t = 15 \times 10^5$. The colors are normalized per column, higher values are shown in magenta and smaller values are shown in blue. Results for RF-Express and AKBX with $\alpha = 1$ were omitted from this plot since they are very similar to those of RF-UCRL and AKBX with $\alpha = 1/2$, respectively.

5.4.4 Exploration with Rewards

To conclude this experimental section, we now evaluate AKBX in the same Grid-World environment (Figure 5.1) in the case where the agent receives a reward when it reaches a state in the bottom-right room. We consider $\alpha = 1/2$, since this is no longer a reward-free situation. In order to reach the rewarding state, the agent has to traverse the 9 rooms, that is, it needs to explore. Figure 5.6 compares the values of the policies π_t learned by AKBX and RND at time t to the value of an optimal policy. The values correspond to the expected sum of rewards in a horizon $H = 100$, starting from the central state in the top-left room, and are estimated by Monte-Carlo policy evaluation.

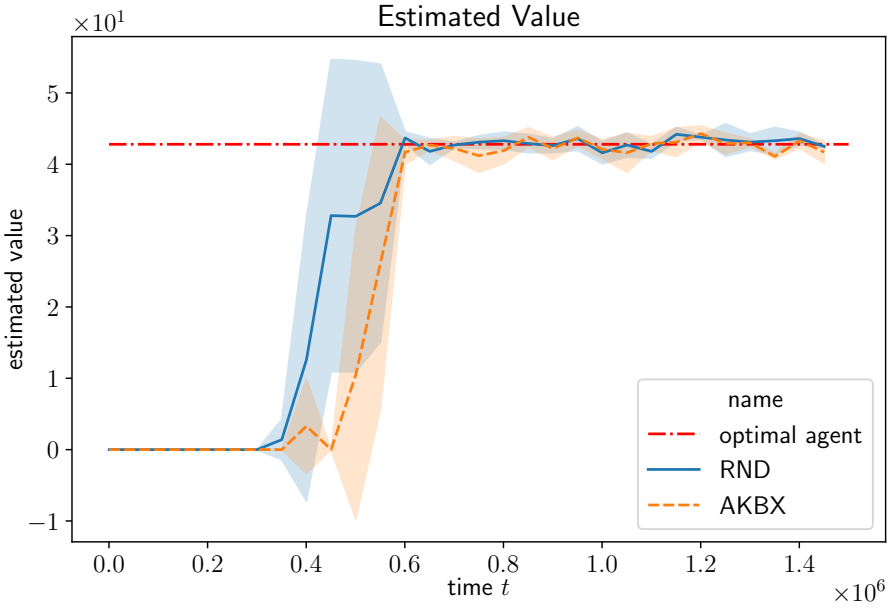


Figure 5.6 – Comparison between the values of the policies π_t learned by AKBX with $\alpha = 1/2$ and RND at time t to the value of an optimal agent. The values correspond to the expected sum of rewards in a horizon $H = 100$, starting from the top-left room of the Grid-World, and are estimated by Monte-Carlo policy evaluation. The optimal policy was computed using value iteration. Average over 4 runs.

We see that both AKBX and RND are able to compute near optimal policies after an initial period of exploration. When rewards are available, though, using a time-independent $\lambda_{1,t}$ for AKBX's bonus (5.7), such that the bonuses do not decay to zero, might prevent the agent from *exploiting*. Indeed, in order to exploit, the bonuses must decay so that the agent is able to maximize its rewards. In our example, setting $\lambda_{1,t} = 1$ did not prevent exploitation, since, in our Grid-World and given the scale of the bonuses, the policy that maximizes the sum of bonuses and rewards should be the same as the policy that maximizes rewards only. However, this is not the case in general, and a well-chosen schedule for decaying $\lambda_{1,t}$ with t may be crucial to obtain a good policy at the end. Notice that the Never-Give-Up agent mentioned in Section 5.3 also has non-vanishing bonuses and, consequently, might have issues with exploitation in some

environments. Never-Give-Up was later extended by Badia et al. [Bad+20a] who introduced Agent57, an agent that adaptively chooses an exploration-exploitation parameter at each time t —analogous to $\lambda_{1,t}$— based on a non-stationary multi-armed bandit algorithm. Thus, a similar technique could be applied when using AKBX's bonuses.

5.5 Discussion and Bibliographical Remarks

In this chapter, we saw that algorithms based on $1/\sqrt{n}$ exploration bonuses are also able to explore an MDP even without rewards, by collecting diverse data that cover all reachable states. Based on the Kernel-UCBVI algorithm introduced in Chapter 4, that generalizes $1/\sqrt{n}$ bonuses to continuous MDPs, we proposed a method called AKBX, which can be seen as a *scalable* version of Kernel-UCBVI with *a learned metric*, that defines a distance and a kernel based on learned representations and explores with approximate kernel-based bonuses. The relevance of collecting diverse data from an MDP is that they can be used, for instance, to learn a transition model, that can be latter used in offline planning for any reward function, or simply to simulate the environment, among other applications, such as learning state representations that can be used for multiple tasks. We related AKBX to Never-Give-Up [Bad+20b], a deep RL algorithm that has been successful at solving hard-exploration Atari games, and we believe that the link between AKBX and Kernel-UCBVI might help us understand why and when approaches like Never-Give-Up work. For instance, as we have seen in Chapter 4, Kernel-UCBVI suffers from the curse of dimensionality: consequently, we might expect algorithms such as AKBX and Never-Give-Up to work only when it is possible to learn a state representation function f such that the covering dimension of the state space with respect to the metric induced by f is small.

Further reading on reward-free exploration In this chapter, we focused on the reward-free exploration (RFE) problem as proposed by Jin et al. [Jin+20a] where the goal is to collect a diverse dataset of transitions. Zhang et al. [ZMS20] study a related setting, called Task-Agnostic Exploration, where an agent has an initial reward-free exploration phase, after which it needs to be able to compute near-optimal policies for N arbitrary tasks. Another common approach to RFE is to frame exploration as a problem of searching for a policy that induces a state-visitation distribution that has maximum entropy, as proposed by Hazan et al. [Haz+19] for finite MDPs. Guo et al. [Guo+21] extend this approach to continuous MDPs by using a geometry-aware Shannon entropy that relies on a kernel function that can also be learned. A fundamental difference between entropy maximization and the dataset approach discussed in this chapter is that a maximum-entropy policy may fail to cover some regions of the state space, whereas increasing the diversity of a dataset usually requires the agent to constantly update its exploration policy in order to visit the least visited states in the previous episodes. Other reward-free goals include the autonomous exploration task of learning policies that are able to visit any state that is reachable in a given radius of a fixed initial state [LA12; Tar+20a;

Tar+21a]. Beyond the finite-horizon criterion, reward-free exploration has also been studied for stochastic shortest path problems [Tar+20b], and in the context of active model estimation [Tar+20c].

Further reading on exploration for deep reinforcement learning Different kinds of exploration bonuses, intrinsic motivation or curiosity methods have been proposed to enhance exploration of RL agents. A non-exhaustive list includes [Sch91; SP12; MJR15; Osb+16; Bel+16; Pat+17; Tan+17; FCRL17; Ost+17; Bur+19; FCL18; OAC18; MBB20], which mostly involve algorithms based on uncertainty quantification, such as $1/\sqrt{n}$-inspired bonuses, prediction errors, density estimation, and bootstrapping. Flet-Berliac et al. [FB+21] propose an actor-critic algorithm for exploration based on constraining the policy at time t to be different from the policy of an adversary trained to predict the actions from previous policies.

Chapter 6

Conclusion

6.1 Main Contributions & Directions for Future Work

Reinforcement learning (RL) is a powerful framework to design algorithms that learn to make decisions and to interact with the world. It allows us to model problems ranging from music recommendation [LSS15] to scheduling viral tests to control an epidemic [Mei+21], from robotics [KBP13] to the design of clinical trials [ZKZ09; Yu+21].

Algorithms for RL can be classified as *offline* or *online*. In the offline case, the agent is given a fixed dataset of transitions, based on which it needs to compute a good policy. Learning policies from offline data is challenging, mostly due to the fact that the algorithm is required to learn a near-optimal policy from data that is often collected by sub-optimal policies. In this thesis, we focused on the *online* case, where the agent needs to collect its own data in order to learn a good policy. In other words, it needs to efficiently *explore* the environment. I believe that studying online reinforcement is extremely relevant, since it provides a framework in which an agent can learn through real-time interactions, adapt to changes in the world, and actively gather new data in order to improve its learning process. Furthermore, progress in online RL can allow us to design agents that are able to quickly build datasets with sufficient coverage of the environment so that they can be used later for offline RL: indeed, this is the goal of the reward-free exploration framework proposed by [Jin+20a] that we studied in Chapter 5.

This thesis presented theoretical and practical contributions to online reinforcement learning. In Chapter 3, we provided unified proofs for worst-case lower bounds in finite MDPs for three common performance criteria: the regret, the sample complexity of best-policy identification (BPI), and the sample complexity of exploration (PAC-MDP). We expect this contribution to be useful to understand the links between the different criteria, and to help the derivation of lower bounds for other settings. Since the lower bounds in finite MDPs scale with the number of states S, we need *regularity conditions* in order to efficiently explore large or continuous MDPs.

In Chapter 4, we studied exploration under weak regularity conditions, and proposed two kernel-based RL algorithms: `Kernel-UCBVI` and `KeRNS`, for which we proved regret bounds in stationary and non-stationary MDPs, and for which we provided approximations with improved runtime. The two main advantages of kernel-based approaches for exploration are (i) they can be applied to general state spaces, since kernels can be defined for several kinds of sets (discrete sets, Euclidean spaces, sets of graphs, etc.); and (ii) they allow us to include prior knowledge about the environment (*e.g.*, symmetries in the state space) through kernel design, which can accelerate exploration. Finally, in Chapter 5 we studied algorithms for reward-free exploration, where an agent can collect data efficiently even without reward feedback, and we introduced `AKBX`, an approximation of `Kernel-UCBVI` that can be used in deep RL, and is based on a kernel function that is learned in real time by any representation learning method. Furthermore, we saw how `AKBX` is related to other approaches for exploration in deep RL [Bad+20b; Bad+20a].

Establishing connections between algorithms with theoretical guarantees, such as `Kernel-UCBVI`, and empirically successful methods for deep RL, such as `Never-Give-Up` [Bad+20b], is important both to better understand current deep RL algorithms and to find possible directions for future research. For instance, since the regret of `Kernel-UCBVI` suffers from the curse of dimensionality, and the exploration of `Never-Give-Up` relies on similar exploration bonuses, we might expect algorithms such as `AKBX` and `Never-Give-Up` to work only for environments where it is possible to learn low-dimensional state representations. Consequently, a key question for future research to improve large-scale exploration might be how to use stronger regularity assumptions in deep RL, which are necessary to improve the regret bounds in theory, as in [Jin+20b; Yan+20; CO20] for instance.

In addition, we saw that several algorithms for regret minimization rely on dynamic programming (DP) combined with exploration bonuses, which are non-stationary and added to the rewards. Hence, besides finding effective ways to compute bonuses in deep RL, an important research direction is the improvement and development of approximate DP algorithms that are able to use off-policy data and that are robust to non-stationary rewards.

An active area of research in online RL is exploration under safety constraints [MA12; Sui+15; TBK16; Ber+17; Wac+18; Din+21]. Although it was not studied in this thesis, safety is extremely important in applications related to robotics and medicine, for instance, and is a crucial point to be considered in future work.

6.2 Software for Reinforcement Learning Research

Another challenging aspect of reinforcement learning research is how to properly organize experiments and compare agents. This is due to the facts that:

- There are several performance criteria used to design agents, which may lead to different implementations. Nevertheless, in a practical application, we might wish to compare agents that were designed for different criteria. For instance, consider the finite-horizon and the infinite-horizon with discounted rewards criteria. In the finite-horizon case, the optimal policy depends on the step $h \in [H]$ within an episode, whereas in the infinite-horizon case the optimal policy depends only on the state. However, a finite-horizon agent can be run in infinite-horizon problems by always playing the policy for $h = 1$, or by seeing an infinite horizon problem as a sequence of finite-horizon ones, for example.
- Different kinds of agents are implemented differently, which can make it more difficult to compare them. For instance, a deep RL algorithm might have its implementation centered around collecting batches of transitions and minimizing a loss function, whereas this is not the case for algorithms such as `UCBVI`.
- To debug algorithms for online RL, it might be useful to compare them to algorithms relying on a generative model, since using such type of model removes the problem of exploration and might provide us a strong baseline policy or a "ground truth". However, it might not be trivial to code and compare algorithms both for online RL and for RL with a generative model using a single interface or codebase.

This diversity in code implementations makes it difficult to compare agents developed in different cases and written by different people. This comparison may be essential for research: consider, for instance, the task of comparing `RF-Express`, an algorithm for finite-horizon reward-free exploration in tabular MDPs, to `AKBX`, an algorithm for reward-free exploration in deep RL. Their code implementations are considerably distinct, but their comparison is important to evaluate how `AKBX` explores when compared to a theoretically-grounded algorithm.

To facilitate empirical research in RL, we developed the `rlberry` library in Python[1] [Dom+21a]. It allows us to compare and debug different agents, create and visualize custom environments, organize experiments, optimize hyperparameters, run agents on remote servers, among other features. The library is based on the principle that the interface used to define an agent must be *simple* enough to include any RL algorithm, such as value iteration, algorithms for finite MDPs, deep RL, MCTS, etc. For that, `rlberry` only requires an agent to implement two methods: a `fit()` method that runs the agent for a certain amount of time, given a computational or sample complexity budget, and an `eval()` method that evaluates the performance of the agent in the environment. Once the agent implements this interface, we can use all the functionalities of the library, which also allows us to train several instances of the algorithms in parallel and handles automatically all the seeds for the random number generators, which is important for reproducibility. The `rlberry` library is compatible with other libraries, such as `Stable-Baselines3` [Raf+21] that implements several deep RL agents. All the experiments in this

[1] Available at https://github.com/rlberry-py/rlberry.

thesis were run using the `rlberry` library. In particular, in the experiments of Chapter 5, `rlberry` was used to compare `RF-UCRL` and `RF-Express`, two algorithms for exploration in finite MDPs, to `RND` and `AKBX`, two deep RL algorithms, and allowed us to easily generate plots as in Figure 5.4. It was also used in the experiments of some of the papers that I co-authored, such as [Dom+21d; Mé+21a; Mé+21b; Tar+21a].

We expect that the development of `rlberry` will be continued, and that it will be useful to develop, debug, and compare RL algorithms created by different research groups. More generally, we hope it will facilitate code sharing and reproducibility in RL.

Appendix A

Complements on Chapter 2

A.1 Proof of Theorem 2.6: Sample Complexity of SmoothCruiser

To bound the sample complexity of SmoothCruiser, we take the following steps:

- In Lemma A.1, we bound the number of recursive calls of sampleV in the uniform sampling phase ($\varepsilon \geq \kappa$), which is similar to the proof of the sample complexity of SparseSampling.
- In Lemma A.2, we bound the number of recursive calls of sampleV when $\varepsilon < \kappa$.
- By noticing that the number of recursive calls of sampleV is equal to the number of oracle calls, we bound the sample complexity of SmoothCruiser, we conclude the proof.

Let $n_{\texttt{sampleV}}(s, \varepsilon, \delta')$ be the total number of recursive calls to sampleV after an initial call with parameters (s, ε), and including the initial call. Since this number does not depend on the state s, we denote it by $n_{\texttt{sampleV}}(\varepsilon, \delta')$.

Lemma A.1. *Let $\varepsilon \geq \kappa$. For all $h \in \mathbb{N}$ and $\forall \varepsilon$ such that $\frac{(1+M)\sqrt{\gamma}^h}{1-\gamma} \leq \varepsilon \leq \frac{1+M}{1-\gamma}$, we have*

$$n_{\texttt{sampleV}}(\varepsilon, \delta') \leq \gamma^{\frac{1}{2}H(\varepsilon)(H(\varepsilon)-1)} \left(\frac{2\alpha(\delta')}{\varepsilon^2}\right)^{H(\varepsilon)} \leq \gamma^{\frac{1}{2}H(\kappa)(H(\kappa)-1)} \left(\frac{2\alpha(\delta')}{\kappa^2}\right)^{H(\kappa)}$$

where

$$H(\varepsilon) = \left\lceil 2\log_\gamma\left(\frac{\varepsilon(1-\gamma)}{1+M}\right)\right\rceil \quad \textit{and} \quad \alpha(\delta') = \frac{18(1+M)^2 A}{(1-\gamma)^4(1-\sqrt{\gamma})^2}\log\left(\frac{2A}{\delta'}\right).$$

Proof. We want to prove that $n_{\texttt{sampleV}}(\varepsilon, \delta') \leq G(\varepsilon)$, where

$$G(\varepsilon) = \gamma^{\frac{1}{2}H(\varepsilon)(H(\varepsilon)-1)} \left(\frac{2\alpha(\delta')}{\varepsilon^2}\right)^{H(\varepsilon)}$$

We proceed by induction on h.

Base case Let $h = 0$. We have $\varepsilon = \frac{1+M}{1-\gamma}$, which implies $n_{\texttt{sampleV}}(\varepsilon, \delta') = 1$ and $G(\varepsilon) = 1$ (since $H(\varepsilon) = 0$). Hence, the claim is true for $h = 0$.

Induction step Assume that the result holds for h. Let $\varepsilon \geq \frac{(1+M)\sqrt{\gamma}^{h+1}}{1-\gamma}$. Since $\frac{\varepsilon}{\sqrt{\gamma}} \geq \frac{(1+M)\sqrt{\gamma}^{h}}{1-\gamma}$, we use the induction hypothesis to obtain

$$
\begin{aligned}
n_{\texttt{sampleV}}(\varepsilon, \delta') &= \underbrace{1}_{\text{current call}} + \underbrace{AN(\varepsilon) n_{\texttt{sampleV}}\left(\frac{\varepsilon}{\sqrt{\gamma}}, \delta'\right)}_{\text{calls in } \texttt{estimateQ}} \\
&\leq \frac{2\alpha(\delta')}{\varepsilon^2} n_{\texttt{sampleV}}\left(\frac{\varepsilon}{\sqrt{\gamma}}, \delta'\right) \\
&\leq \frac{2\alpha(\delta')}{\varepsilon^2} \gamma^{\frac{1}{2}(H(\varepsilon)-1)(H(\varepsilon)-2)} \left(\frac{\gamma 2\alpha(\delta')}{\varepsilon^2}\right)^{H(\varepsilon)-1}, \quad \text{since } H\left(\frac{\varepsilon}{\sqrt{\gamma}}\right) = H(\varepsilon) - 1 \\
&= \gamma^{\frac{1}{2}H(\varepsilon)(H(\varepsilon)-1)} \left(\frac{2\alpha(\delta')}{\varepsilon^2}\right)^{H(\varepsilon)},
\end{aligned}
$$

which completes the proof. □

Lemma A.2. *Let $\varepsilon \leq \kappa$. For all $h \in \mathbb{N}$, $\forall \varepsilon \geq \kappa\sqrt{\gamma}^h$, we have*

$$
n_{\texttt{sampleV}}(\varepsilon, \delta') \leq \eta_1 \left[\log_{\frac{1}{\gamma}}\left(\frac{\kappa/\gamma}{\varepsilon}\right)\right]^{\eta_2(\delta')} \frac{1}{\varepsilon^2}
$$

where

$$
\begin{aligned}
\kappa &= \frac{1-\sqrt{\gamma}}{AL} \\
\eta_1 &= \kappa^2 n_{\texttt{sampleV}}(\kappa, \delta') \\
\eta_2(\delta') &= \log_2\left(\frac{\gamma}{1-\gamma}\frac{2\beta(\delta')}{\kappa}\right) \\
\beta(\delta') &= \frac{18(1+M)^2A^2L}{(1-\gamma)^4(1-\sqrt{\gamma})^3} \log\left(\frac{2A}{\delta'}\right)
\end{aligned}
$$

under the condition that

$$
\log_2\left(\frac{\gamma}{1-\gamma}\frac{2\beta(\delta')}{\kappa}\right) \geq 0, \quad \textit{i.e.,} \quad \beta(\delta') \geq \frac{(1-\gamma)(1-\sqrt{\gamma})}{2\gamma AL} \tag{A.1}
$$

which is satisfied by choosing δ' small enough.

Proof. First, let us define some auxiliary quantities,

$$B_1(\varepsilon) := \left[\log_{\frac{1}{\gamma}}\left(\frac{\kappa/\gamma}{\varepsilon}\right)\right]^{\eta_2(\delta')}, \tag{A.2}$$

$$B_2(\varepsilon) := \frac{\eta_1}{\varepsilon^2}, \quad \text{and} \tag{A.3}$$

$$B(\varepsilon) := B_1(\varepsilon)B_2(\varepsilon) \tag{A.4}$$

We want to prove that $n_{\texttt{sampleV}}(\varepsilon, \delta') \leq B(\varepsilon)$ and we proceed by induction on h.

Base case For $h = 0$, we have $\varepsilon \geq \kappa$ and, by assumption, $\varepsilon \leq \kappa$. Therefore, $\varepsilon = \kappa$. It can be easily verified that $B(\kappa) = n_{\texttt{sampleV}}(\kappa, \delta')$, hence the lemma is true for $h = 0$.

Induction hypothesis Assume that the lemma is true for h.

Induction step Let $\varepsilon \geq \kappa\sqrt{\gamma}^{h+1}$. We have that

$$\begin{aligned} n_{\texttt{sampleV}}(\varepsilon, \delta') &= \underbrace{1}_{\text{current call}} + \underbrace{n_{\texttt{sampleV}}\left(\frac{\varepsilon}{\sqrt{\gamma}}, \delta'\right)}_{\text{call in line 11 of } \texttt{sampleV}} + \underbrace{AN(\sqrt{\kappa\varepsilon}) n_{\texttt{sampleV}}\left(\sqrt{\frac{\kappa\varepsilon}{\gamma}}, \delta'\right)}_{\text{calls in } \texttt{estimateQ}} \\ &= 1 + n_{\texttt{sampleV}}\left(\frac{\varepsilon}{\sqrt{\gamma}}, \delta'\right) + \frac{\beta(\delta')}{\varepsilon} n_{\texttt{sampleV}}\left(\sqrt{\frac{\kappa\varepsilon}{\gamma}}, \delta'\right) \\ &\leq n_{\texttt{sampleV}}\left(\frac{\varepsilon}{\sqrt{\gamma}}, \delta'\right) + \frac{2\beta(\delta')}{\varepsilon} n_{\texttt{sampleV}}\left(\sqrt{\frac{\kappa\varepsilon}{\gamma}}, \delta'\right) \end{aligned}$$

Since $\varepsilon \geq \kappa\sqrt{\gamma}^{h+1}$ and $\varepsilon \leq \kappa$, we have $\sqrt{\frac{\kappa\varepsilon}{\gamma}} \geq \frac{\varepsilon}{\sqrt{\gamma}} \geq \kappa\sqrt{\gamma}^{h}$. This allows us to use our induction hypothesis to get

$$n_{\texttt{sampleV}}(\varepsilon, \delta') \leq B\left(\frac{\varepsilon}{\sqrt{\gamma}}\right) + \frac{2\beta(\delta')}{\varepsilon} B\left(\sqrt{\frac{\kappa\varepsilon}{\gamma}}\right).$$

Below, we will use the fact that:

$$\log\left(\frac{\kappa}{\sqrt{\frac{\kappa\varepsilon}{\gamma}}\gamma}\right) = \frac{1}{2}\log\left(\frac{\kappa/\gamma}{\varepsilon}\right) \tag{A.5}$$

We have that

$$\frac{B\left(\frac{\varepsilon}{\sqrt{\gamma}}\right)}{B(\varepsilon)} = \frac{B_1\left(\frac{\varepsilon}{\sqrt{\gamma}}\right)}{B_1(\varepsilon)} \frac{B_2\left(\frac{\varepsilon}{\sqrt{\gamma}}\right)}{B_2(\varepsilon)} = \gamma \underbrace{\left[\frac{\log\left(\frac{\kappa/\gamma}{\varepsilon}\right) - \frac{1}{2}\log\frac{1}{\gamma}}{\log\left(\frac{\kappa/\gamma}{\varepsilon}\right)}\right]^{\eta_2(\delta')}}_{<1} \leq \gamma,$$

where we used the assumption that $\eta_2(\delta') \geq 0$.

Also,

$$\frac{B\left(\sqrt{\frac{\kappa\varepsilon}{\gamma}}\right)}{B(\varepsilon)} = \frac{\varepsilon\gamma}{\kappa}\frac{B_1\left(\sqrt{\frac{\kappa\varepsilon}{\gamma}}\right)}{B_1(\varepsilon)} = \frac{\varepsilon\gamma}{\kappa}\left[\frac{\log_{\frac{1}{\gamma}}\left(\frac{\kappa}{\sqrt{\frac{\kappa\varepsilon}{\gamma}}\gamma}\right)}{\log_{\frac{1}{\gamma}}\left(\frac{\kappa/\gamma}{\varepsilon}\right)}\right]^{\eta_2(\delta')} = \frac{\varepsilon\gamma}{\kappa}\left[\frac{\frac{1}{2}\log_{\frac{1}{\gamma}}\left(\frac{\kappa/\gamma}{\varepsilon}\right)}{\log_{\frac{1}{\gamma}}\left(\frac{\kappa/\gamma}{\varepsilon}\right)}\right]^{\eta_2(\delta')}$$
$$= \frac{\varepsilon\gamma}{\kappa}\left(\frac{1}{2}\right)^{\eta_2(\delta')} = \frac{\varepsilon\gamma}{\kappa}\frac{(1-\gamma)}{\gamma}\frac{\kappa}{2\beta(\delta')} = \frac{(1-\gamma)\varepsilon}{2\beta(\delta')}.$$

Finally, we obtain

$$n_{\texttt{sampleV}}(\varepsilon,\delta') \leq B\left(\frac{\varepsilon}{\sqrt{\gamma}}\right) + \frac{2\beta(\delta')}{\varepsilon}B\left(\sqrt{\frac{\kappa\varepsilon}{\gamma}}\right) \leq \gamma B(\varepsilon) + \frac{2\beta(\delta')}{\varepsilon}\frac{(1-\gamma)\varepsilon}{2\beta(\delta')}B(\varepsilon) = B(\varepsilon),$$

which proves the lemma. □

Now, we can conclude the proof of Theorem 2.6. Notice that the number of calls to the generative model is smaller than the total number of calls to `sampleV`. `SmoothCruiser` makes one call to `estimateQ`, which makes $N(\varepsilon)$ calls to `sampleV`. If $\varepsilon \geq \kappa$, Lemma A.1 shows that the sample complexity is bounded by a constant. Lemma A.2 bounds the sample complexity for $\varepsilon \leq \kappa$, and we use it to bound $n(\varepsilon,\delta')$:

$$n(\varepsilon,\delta') = N(\varepsilon)n_{\texttt{sampleV}}(\varepsilon,\delta') \leq N(\varepsilon)\eta_1\left[\log_{\frac{1}{\gamma}}\left(\frac{\kappa/\gamma}{\varepsilon}\right)\right]^{\eta_2(\delta')}\frac{1}{\varepsilon^2}$$
$$\leq \frac{c_1}{\varepsilon^4}\log\left(\frac{c_2}{\delta'}\right)\left[c_3\log\left(\frac{c_4}{\varepsilon}\right)\right]^{\log_2\left(c_5\left(\log\left(\frac{c_2}{\delta'}\right)\right)\right)} = \tilde{\mathcal{O}}\left(\frac{1}{\varepsilon^4}\right)$$

by using the definition of $N(\varepsilon)$ for $\varepsilon \leq \kappa$ and the definition of $\eta_2(\delta')$ in Lemma A.2. The constants are given by:

$$c_1 = \frac{18(1+M)^2 n_{\texttt{sampleV}}(\kappa,\delta')}{A^2L^2(1-\gamma)^4}; \quad c_2 = 2A; \quad c_3 = [\log(1/\gamma)]^{-1};$$
$$c_4 = (1-\sqrt{\gamma})/(\gamma AL); \quad c_5 = \frac{36(1+M)^2\gamma A^3L^2}{(1-\gamma)^5(1-\sqrt{\gamma})^4}.$$

A.2 Proof of Theorem 2.7: Consistency of SmoothCruiser

To prove that SmoothCruiser outputs a good estimate of the optimal regularized value function with high probability, we proceed as follows:

- In Lemma A.3, we prove that the output of sampleV, conditioned on an event $\mathcal{G}$, is a low-bias estimate of the true value function, and that $\mathcal{G}$ happens with high probability;
- Given Lemma A.3, the proof of Theorem 2.7 is straightforward.

Throughout the proof, we will make distinctions between two cases:

- **Case 1:** $\kappa \leq \varepsilon < \frac{1+M}{1-\gamma}$;
- **Case 2:** $\varepsilon < \kappa$.

Useful definitions We define the function $\zeta(\varepsilon)$ as

$$\zeta(\varepsilon) = \begin{cases} \varepsilon, & \text{if } \kappa \leq \varepsilon < \frac{1+M}{1-\gamma}, \\ \sqrt{\kappa\varepsilon}, & \text{if } \varepsilon < \kappa, \\ \infty, & \text{otherwise.} \end{cases}$$

We define $\mathbf{params}(s, \varepsilon)$ as the (random) set of parameters used to call sampleV after a call to sampleV(s, ε), that is

$$\mathbf{params}(s, \varepsilon) = \left\{ \left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}} \right) \text{ for } k = 1, \ldots, N(\varepsilon); a \in \mathcal{A} \right\} \tag{A.6}$$

in case 1 and

$$\mathbf{params}(s, \varepsilon) = \left\{ \left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}} \right) \text{ for } k = 1, \ldots, N(\varepsilon); a \in \mathcal{A} \right\} \bigcup \left\{ \left(Z_{s,\widehat{A}}, \frac{\varepsilon}{\sqrt{\gamma}} \right) \right\} \tag{A.7}$$

in case 2, where $Z_{s,a}^{(k)}$ are the next states sampled in estimateQ and $Z_{s,\widehat{A}}$ is the next state sampled in sampleV(s, ε).

A call to sampleV(s, ε) makes one call to estimateQ. Denote the output of this call to estimateQ by $\widehat{Q}_s^{\varepsilon}$. We define the event $\mathcal{G}(s, \varepsilon)$ as follows:

$$\mathcal{G}(s, \varepsilon) = \begin{cases} \left\{ \|\widehat{Q}_s^{\varepsilon} - Q_s\|_\infty \leq \zeta(\varepsilon) \right\} \bigcap \mathcal{B}(s, \varepsilon), & \text{if } 0 < \varepsilon < \frac{1+M}{1-\gamma}, \\ \Omega, & \text{if } \varepsilon \geq \frac{1+M}{1-\gamma}. \end{cases}$$

where Ω is the whole sample space and

$$\mathcal{B}(s,\varepsilon) = \bigcap_{(z,e)\in\mathbf{params}(s,\varepsilon)} \mathcal{G}(z,e).$$

Finally, we define C_γ as:

$$C_\gamma = \frac{3(1+M)}{(1-\gamma)^2}$$

Lemma A.3. *Let $\widehat{V}_\varepsilon(s) = \texttt{sampleV}(s,\varepsilon)$. For all $h \in \mathbb{N}, s \in \mathcal{S}, \varepsilon \geq \frac{(1+M)\sqrt{\gamma}^h}{1-\gamma}$, we have:*

$$\begin{aligned}
&(\mathbf{i}) \quad |\mathbf{E}\left[\widehat{V}_\varepsilon(s)\middle|\mathcal{G}(s,\varepsilon)\right] - V(s)| \leq \varepsilon,\\
&(\mathbf{ii}) \quad \mathbf{P}\left[|\widehat{V}_\varepsilon(s)| \leq C_\gamma\middle|\mathcal{G}(s,\varepsilon)\right] = 1, \text{ and}\\
&(\mathbf{iii}) \quad \mathbf{P}\left[\mathcal{G}(s,\varepsilon)\right] \geq 1 - \delta' n_{\texttt{sampleV}}(\varepsilon,\delta').
\end{aligned}$$

where

$$n_{\texttt{sampleV}}(\varepsilon,\delta') = 1 + \sum_{(z,e)\in\mathbf{params}(s,\varepsilon)} n_{\texttt{sampleV}}(e,\delta')$$

is the total number of recursive calls to `sampleV` *after an initial call with parameters (s,ε).*

Proof. We proceed by induction on h.

(1) Base case If $h = 0$, $\varepsilon \geq \frac{1+M}{1-\gamma}$ and $\mathcal{G}(s,\varepsilon) = \Omega$. The output is then $\widehat{V}_\varepsilon(s) = 0$. Point **(i)** is verified by using the fact that $|V(s)| \leq \frac{1+M}{1-\gamma} \leq \varepsilon$; points **(ii)** and **(iii)** are trivially verified.

(2) Induction hypothesis Assume that **(i)**, **(ii)** and **(iii)** are true for h.

(3) Induction step Let $\varepsilon \geq \frac{(1+M)\sqrt{\gamma}^{h+1}}{1-\gamma}$. This implies that $\varepsilon/\sqrt{\gamma}$ and $\zeta(\varepsilon)/\sqrt{\gamma}$ are both greater than $\frac{(1+M)\sqrt{\gamma}^h}{1-\gamma}$, which will allow us to use the induction hypothesis.

We start by proving **(iii)**, that is, that the event $\mathcal{G}(s,\varepsilon)$ has high probability.

Let $\widehat{Q}_s^\varepsilon = \texttt{estimateQ}(s,\zeta(\varepsilon))$. Let the reward $R_{s,a}^{(k)}$ and state $Z_{s,a}^{(k)}$ be the random variables associated to the k-th call to the generative model used to compute $\widehat{Q}_s$ in `estimateQ`, for $k \in \{1,\cdots,N(\varepsilon)\}$. Let

$$q_s^k(a) := R_{s,a}^{(k)} + \gamma\texttt{sampleV}\left(Z_{s,a}^{(k)}, \zeta(\varepsilon)/\sqrt{\gamma}\right)$$

and let

$$\overline{Q}_s^{\varepsilon}(a) = \frac{1}{N(\varepsilon)} \sum_{k=1}^{N(\varepsilon)} q_s^k(a) \tag{A.8}$$

so that:

$$\widehat{Q}_s^{\varepsilon} = \mathbf{clip}_{(1+M)(1-\gamma)^{-1}} \left(\overline{Q}_s^{\varepsilon}(a) \right)$$

where, for any $c \geq 0$ and $x \in \mathbb{R}^A$, $\mathbf{clip}_c(x) = \min(\max(x, 0), c)$.

we have:

$$\begin{aligned} \left| \widehat{Q}_s^{\varepsilon}(a) - Q_s(a) \right| &\leq \left| \overline{Q}_s^{\varepsilon}(a) - Q_s(a) \right| \\ &\leq \underbrace{\left| \overline{Q}_s^{\varepsilon}(a) - \mathbf{E}\left[\overline{Q}_s^{\varepsilon}(a) | \mathcal{B}(s,\varepsilon) \right] \right|}_{\textbf{(I)}} + \underbrace{\left| \mathbf{E}\left[\overline{Q}_s^{\varepsilon}(a) | \mathcal{B}(s,\varepsilon) \right] - Q_s(a) \right|}_{\textbf{(II)}}. \end{aligned}$$

We'd like to use Hoeffding's inequality to bound **(I)** with high probability. For that, we need to verify that the random variables $\{q_s^k(a)\}_{k=1}^{N(\varepsilon)}$ are bounded and independent conditionally on $\mathcal{B}(s,\varepsilon)$.

Boundedness. By the induction hypothesis **(ii)**, in the event $\mathcal{B}(s,\varepsilon)$, the random variables

$$\texttt{sampleV}\left(Z_{s,a}^{(k)}, \zeta(\varepsilon)/\sqrt{\gamma} \right),$$

for all k, are bounded by C_γ. Using the fact that the rewards are in $[0,1]$, we obtain that $q_s^k(a)$ is also bounded by C_γ.

Independence. Let $E_k = \mathcal{G}\left(Z_{s,a}^k, \zeta(\varepsilon)/\sqrt{\gamma} \right)$. For any $t \in \mathbb{R}^{N(\varepsilon)}$, the characteristic function of the random vector $\{q_s^k(a)\}_{k=1}^{N(\varepsilon)}$ conditionally on $\mathcal{B}(s,\varepsilon)$ is given by

$$\begin{aligned} &\mathbf{E}\left[\exp\left(\mathrm{i} \sum_k t_k q_s^k(a) \right) \middle| \mathcal{B}(s,\varepsilon) \right] \\ &\overset{\textbf{(a)}}{=} \mathbf{E}\left[\exp\left(\mathrm{i} \sum_k t_k q_s^k(a) \right) \middle| \bigcap_k E_k \right] = \frac{\mathbf{E}\left[\exp\left(\mathrm{i} \sum_k t_k q_s^k(a) \right) \prod_k \mathbb{1}\{E_k\} \right]}{\mathbf{E}\left[\prod_k \mathbb{1}\{E_k\} \right]} \\ &= \frac{\mathbf{E}\left[\prod_k \exp\left(\mathrm{i} t_k q_s^k(a) \right) \mathbb{1}\{E_k\} \right]}{\mathbf{E}\left[\prod_k \mathbb{1}\{E_k\} \right]} \overset{\textbf{(b)}}{=} \frac{\prod_k \mathbf{E}\left[\exp\left(\mathrm{i} t_k q_s^k(a) \right) \mathbb{1}\{E_k\} \right]}{\prod_k \mathbf{E}\left[\mathbb{1}\{E_k\} \right]} \\ &= \prod_k \mathbf{E}\left[\exp\left(\mathrm{i} t_k q_s^k(a) \right) \middle| E_k \right] \overset{\textbf{(c)}}{=} \prod_k \mathbf{E}\left[\exp\left(\mathrm{i} t_k q_s^k(a) \right) \middle| \mathcal{B}(s,\varepsilon) \right] \end{aligned}$$

which is justified by

- **(a)** Definition of $\mathcal{B}(s,\varepsilon)$ and the fact that $\{q_s^k(a)\}_{k=1}^{N(\varepsilon)}$ are independent of $\mathcal{G}\left(Z_{s,\widehat{A}}, \frac{\varepsilon}{\sqrt{\gamma}}\right)$;
- **(b)** The random variables $\{q_s^k(a)\}_{k=1}^{N(\varepsilon)}$ are independent and the events $\{E_k\}_{i=1}^{N(\varepsilon)}$ are also independent;
- **(c)** The random variable $q_s^k(a)$ is independent of every E_j for $j \neq k$.

Since the characteristic function of $\{q_s^k(a)\}_{k=1}^{N(\varepsilon)}$ is the product of their characteristic functions, these random variables are independent given $\mathcal{B}(s,\varepsilon)$.

Now we can use Hoeffding's inequality:

$$\begin{aligned}
&\mathbf{P}\left[\left|\overline{Q}_s^{\varepsilon}(a) - \mathbf{E}\left[\overline{Q}_s^{\varepsilon}(a)\middle|\mathcal{B}(s,\varepsilon)\right]\right| \geq (1-\sqrt{\gamma})\zeta(\varepsilon)\middle|\mathcal{B}(s,\varepsilon)\right] \\
&= \mathbf{P}\left[\left|\frac{1}{N(\varepsilon)}\sum_{k=1}^{N(\varepsilon)} q_s^k(a) - \mathbf{E}\left[q_s^k(a)\middle|\mathcal{B}(s,\varepsilon)\right]\right| \geq (1-\sqrt{\gamma})\zeta(\varepsilon)\middle|\mathcal{B}(s,\varepsilon)\right] \\
&\leq 2\exp\left(-\frac{N(\varepsilon)(1-\sqrt{\gamma})^2\zeta(\varepsilon)^2}{2C_\gamma^2}\right) \\
&\leq \frac{\delta'}{A}
\end{aligned}$$

And **(II)** is bounded by using the induction hypothesis **(i)**:

$$\begin{aligned}
&\left|\mathbf{E}\left[q_s^k(a)\middle|\mathcal{B}(s,\varepsilon)\right] - Q_s(a)\right| \\
&\overset{(a)}{=} \gamma\left|\mathbf{E}\left[\texttt{sampleV}\left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}}\right)\middle|\mathcal{B}(s,\varepsilon)\right] - \mathbf{E}\left[V(Z_{s,a}^{(k)})\middle|\mathcal{B}(s,\varepsilon)\right]\right| \\
&\overset{(b)}{=} \gamma\left|\mathbf{E}\left[\texttt{sampleV}\left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}}\right)\middle|\mathcal{G}\left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}}\right)\right] - \mathbf{E}\left[V(Z_{s,a}^{(k)})\middle|\mathcal{G}\left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}}\right)\right]\right| \\
&\overset{(c)}{=} \gamma\left|\mathbf{E}\left[\mathbf{E}\left[\texttt{sampleV}\left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}}\right)\middle|Z_{s,a}^{(k)}, \mathcal{G}\left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}}\right)\right] - V(Z_{s,a}^{(k)})\middle|\mathcal{G}\left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}}\right)\right]\right| \\
&\overset{(d)}{\leq} \gamma\frac{\zeta(\varepsilon)}{\sqrt{\gamma}} = \sqrt{\gamma}\zeta(\varepsilon)
\end{aligned}$$

which is justified by the following:

- **(a)** $\mathbf{E}\left[R_{s,a}^{(k)}\middle|\mathcal{B}(s,\varepsilon)\right] = \mathbf{E}\left[R_{s,a}^{(k)}\right]$, since the reward depends only on s, a;
- **(b)** The term $\left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}}\right)$ depends on $\mathcal{B}(s,\varepsilon)$ only through $\mathcal{G}\left(Z_{s,a}^{(k)}, \frac{\zeta(\varepsilon)}{\sqrt{\gamma}}\right)$;
- **(c)** Law of total expectation;
- **(d)** Consequence of induction hypothesis **(i)**.

Putting together the bounds for **(I)** and **(II)** and doing an union bound over all actions, we obtain:

$$\mathbf{P}\left[\|\widehat{Q}_s^\varepsilon - Q_s\|_\infty \geq \zeta(\varepsilon)\middle|\mathcal{B}(s,\varepsilon)\right] \leq \delta'.$$

We can now give a lower bound to the probability of the event $\mathcal{G}(s,\varepsilon)$. Let

$$\mathcal{E} = \left\{\|\widehat{Q}_s^\varepsilon - Q_s\|_\infty < \zeta(\varepsilon)\right\}$$

We have:

$$\begin{aligned}\mathbf{P}\left[\mathcal{G}(s,\varepsilon)\right] &\geq \mathbf{P}\left[\mathcal{E}\cap\mathcal{B}(s,\varepsilon)\right] = \mathbf{P}\left[\mathcal{E}\middle|\mathcal{B}(s,\varepsilon)\right]\mathbf{P}\left[\mathcal{B}(s,\varepsilon)\right] \\ &= \left(1-\mathbf{P}\left[\mathcal{E}^\complement\middle|\mathcal{B}(s,\varepsilon)\right]\right)\mathbf{P}\left[\mathcal{B}(s,\varepsilon)\right] \geq \mathbf{P}\left[\mathcal{B}(s,\varepsilon)\right] - \delta' \geq 1-\delta' n_{\texttt{sampleV}}(\varepsilon,\delta')\end{aligned}$$

since

$$\begin{aligned}\mathbf{P}\left[\mathcal{B}(s,\varepsilon)\right] &= 1-\mathbf{P}\left[\mathcal{B}(s,\varepsilon)^\complement\right] = 1-\mathbf{P}\left[\bigcup_{(z,e)\in\mathbf{params}(s,\varepsilon)}\mathcal{G}(z,e)^\complement\right] \\ &\geq 1-\sum_{(z,e)\in\mathbf{params}(s,\varepsilon)}\mathbf{P}\left[\mathcal{G}(z,e)^\complement\right] \\ &\geq 1-\delta'\sum_{(z,e)\in\mathbf{params}(s,\varepsilon)} n_{\texttt{sampleV}}(e,\delta') \quad \text{by induction hypothesis (iii)} \\ &= 1-\delta'(n_{\texttt{sampleV}}(\varepsilon,\delta')-1)\end{aligned}$$

This proves the point **(iii)** of Lemma A.3.

Now, let's prove **(i)**, which bounds the bias of the output of `sampleV`. For any event $\mathcal{E}$, we write

$$\mathbf{E}_{\mathcal{E}}\left[\cdot\right] := \mathbf{E}\left[\cdot|\mathcal{E}\right].$$

Case 1 We start with case 1, $\kappa \leq \varepsilon < \frac{1+M}{1-\gamma}$, where $\zeta(\varepsilon)=\varepsilon$ and $\widehat{V}_\varepsilon(s) = F(\widehat{Q}_s^\varepsilon)$. We have:

$$\begin{aligned}&\left|\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\widehat{V}_\varepsilon(s)\right] - V(s)\right| = \left|\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[F(\widehat{Q}_s^\varepsilon) - F(Q_s)\right]\right| \\ &\leq \mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\left|F(\widehat{Q}_s^\varepsilon) - F(Q_s)\right|\right] \leq \mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\|\widehat{Q}_s^\varepsilon(a) - Q_s(a)\|_\infty\right] \leq \zeta(\varepsilon) = \varepsilon\end{aligned}$$

which proves **(i)** in case 1.

Case 2 Consider now the case 2, $\varepsilon < \kappa$, where $\zeta(\varepsilon) = \sqrt{\kappa\varepsilon}$. Let $\widehat{A}$ be the action following the distribution $\frac{\nabla F\left(\widehat{Q}_s^\varepsilon\right)}{\left\|\nabla F\left(\widehat{Q}_s^\varepsilon\right)\right\|_1}$, and let the reward $R_{s,\widehat{A}}$ and the state $Z_{s,\widehat{A}}$ be the random variables associated to the call to the generative model with parameters $(s,\widehat{A})$. Let $\widehat{V} = \texttt{sampleV}\left(Z_{s,\widehat{A}}, \varepsilon/\sqrt{\gamma}\right)$.

The output in this case is given by

$$\widehat{V}_\varepsilon(s) = F\left(\widehat{Q}_s^\varepsilon\right) - (\widehat{Q}_s^\varepsilon)^\intercal \nabla F\left(\widehat{Q}_s^\varepsilon\right) + (R + \gamma\widehat{V})\left\|\nabla F\left(\widehat{Q}_s^\varepsilon\right)\right\|_1$$

Let

$$Q_s(\widehat{A}) = \mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[R_{s,\widehat{A}} + \gamma V(Z_{s,\widehat{A}})|\widehat{A}, \widehat{Q}_s^\varepsilon\right] = \mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[R_{s,\widehat{A}} + \gamma V(Z_{s,\widehat{A}})|\widehat{A}\right]$$

and let

$$\widetilde{V}(s) = \mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[F\left(\widehat{Q}_s^\varepsilon\right) - (\widehat{Q}_s^\varepsilon)^\intercal \nabla F\left(\widehat{Q}_s^\varepsilon\right) + Q_s(\widehat{A})\left\|\nabla F\left(\widehat{Q}_s^\varepsilon\right)\right\|_1\right].$$

We have

$$\begin{aligned}
&\left|\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\widehat{V}_\varepsilon(s)\right] - \widetilde{V}(s)\right| \\
&\overset{(\mathbf{a})}{=} \gamma\left|\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\texttt{sampleV}\left(Z_{s,\widehat{A}}, \frac{\varepsilon}{\sqrt{\gamma}}\right) - V(Z_{s,\widehat{A}})\middle|\widehat{A}, \widehat{Q}_s^\varepsilon, Z_{s,\widehat{A}}\right]\left\|\nabla F\left(\widehat{Q}_s^\varepsilon\right)\right\|_1\right]\right| \\
&\overset{(\mathbf{b})}{=} \gamma\left|\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\left(\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\texttt{sampleV}\left(Z_{s,\widehat{A}}, \frac{\varepsilon}{\sqrt{\gamma}}\right)\middle|\widehat{A}, \widehat{Q}_s^\varepsilon, Z_{s,\widehat{A}}\right] - V(Z_{s,\widehat{A}})\right)\right]\left\|\nabla F\left(\widehat{Q}_s^\varepsilon\right)\right\|_1\right| \\
&\overset{(\mathbf{c})}{\leq} \gamma\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\left|\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\texttt{sampleV}\left(Z_{s,\widehat{A}}, \frac{\varepsilon}{\sqrt{\gamma}}\right)\middle|\widehat{A}, \widehat{Q}_s^\varepsilon, Z_{s,\widehat{A}}\right] - V(Z_{s,\widehat{A}})\right|\right] \\
&\overset{(\mathbf{d})}{=} \gamma\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\left|\mathbf{E}_{\mathcal{G}(Z_{s,\widehat{A}},\varepsilon/\sqrt{\gamma})}\left[\texttt{sampleV}\left(Z_{s,\widehat{A}}, \frac{\varepsilon}{\sqrt{\gamma}}\right)\middle|Z_{s,\widehat{A}}\right] - V(Z_{s,\widehat{A}})\right|\right] \\
&\overset{(\mathbf{e})}{\leq} \gamma\frac{\varepsilon}{\sqrt{\gamma}} = \sqrt{\gamma}\varepsilon.
\end{aligned}$$

which is justified by the following points:

- **(a)** The reward depends only on s, a and law of total expectation;
- **(b)** $V(Z_{s,\widehat{A}})$ is a function of $Z_{s,\widehat{A}}$ and no other random variable;
- **(c)** Jensen's inequality and the fact that $\|\nabla F\left(\widehat{Q}_s^\varepsilon\right)\|_1 \leq 1$;
- **(d)** Given $Z_{s,\widehat{A}}$, the term $\texttt{sampleV}\left(Z_{s,\widehat{A}}, \frac{\varepsilon}{\sqrt{\gamma}}\right)$ depends on $\mathcal{G}(s,\varepsilon)$ only through $\mathcal{G}(Z_{s,\widehat{A}}, \frac{\varepsilon}{\sqrt{\gamma}})$;
- **(e)** Induction hypothesis **(i)**.

Now, $\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[Q_s(\widehat{A})\|\nabla F(\widehat{Q}_s^\varepsilon)\|_1\right]$ can be written as

$$\begin{aligned}
\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[Q_s(\widehat{A})\|\nabla F(\widehat{Q}_s^\varepsilon)\|_1\right] &= \mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[Q_s(\widehat{A})\middle|\widehat{Q}_s^\varepsilon\right]\|\nabla F(\widehat{Q}_s^\varepsilon)\|_1\right] \\
&= \mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[Q_s^\intercal \nabla F(\widehat{Q}_s^\varepsilon)\right]
\end{aligned}$$

so that $\widetilde{V}(s)$ is given by

$$\widetilde{V}(s) = \mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[F(\widehat{Q}_s^\varepsilon) + (Q_s - \widehat{Q}_s^\varepsilon)^\intercal \nabla F(\widehat{Q}_s^\varepsilon)\right].$$

Finally, we bound the difference between $\widetilde{V}(s)$ and $V(s)$:

$$\begin{aligned}
|\widetilde{V}(s) - V(s)| &\leq \mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\left|F\left(\widehat{Q}_s^\varepsilon\right) + (Q_s - \widehat{Q}_s^\varepsilon)^\intercal \nabla F\left(\widehat{Q}_s^\varepsilon\right) - V(s)\right|\right] \\
&\leq L\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\|Q_s - \widehat{Q}_s^\varepsilon\|_2^2\right] \overset{(a)}{\leq} AL\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\|Q_s - \widehat{Q}_s^\varepsilon\|_\infty^2\right] \leq AL\zeta(\varepsilon)^2 \\
&= AL\kappa\varepsilon = (1-\sqrt{\gamma})\varepsilon
\end{aligned}$$

by using the fact that we are on $\mathcal{G}(s,\varepsilon)$ and (a) uses the fact that for all $x \in \mathbb{R}^A$, $\|x\|_2^2 \leq A\|x\|_\infty^2$.

We now conclude the proof of **(i)** for case 2:

$$\begin{aligned}
\left|\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\widehat{V}_\varepsilon(s)\right] - V(s)\right| &\leq \left|\mathbf{E}_{\mathcal{G}(s,\varepsilon)}\left[\widehat{V}_\varepsilon(s)\right] - \widetilde{V}(s)\right| + \left|\widetilde{V}(s) - V(s)\right| \\
&\leq \sqrt{\gamma}\varepsilon + (1-\sqrt{\gamma})\varepsilon = \varepsilon.
\end{aligned}$$

Finally, let's prove **(ii)**, stating that $\widehat{V}_\varepsilon(s)$ is bounded by C_γ on the event $\mathcal{G}(s,\varepsilon)$.

Case 1 In this case, $\widehat{V}_\varepsilon(s) = F\widehat{Q}_s^\varepsilon)$ with $\|\widehat{Q}_s^\varepsilon\|_\infty \leq (1+M)/(1-\gamma)$, since each component of $\widehat{Q}_s^\varepsilon$ is clipped and lie in the interval $\left[0, \frac{1+M}{1-\gamma}\right]$. The assumptions on F imply that $\left|\widehat{V}_\varepsilon(s)\right| \leq \frac{1+M}{1-\gamma} \leq C_\gamma$.

Case 2. In this case, we have:

$$\begin{aligned}
|\widehat{V}_\varepsilon(s)| &\leq \left|F(\widehat{Q}_s^\varepsilon) - (\widehat{Q}_s^\varepsilon)^\intercal \nabla F(\widehat{Q}_s^\varepsilon)\right| + |R + \gamma\widehat{V}|\|\nabla F(\widehat{Q}_s^\varepsilon)\|_1 \\
&\leq 2\|\widehat{Q}_s^\varepsilon\|_\infty + M + 1 + \gamma C_\gamma \leq \frac{2(1+M)}{1-\gamma} + M + 1 + \gamma C_\gamma \leq C_\gamma,
\end{aligned}$$

since $|\widehat{V}| \leq C_\gamma$ by induction hypothesis **(ii)**.

This proves **(ii)** for case 2:

$$\mathbf{P}\left[|\widehat{V}(s)| \leq C_\gamma \middle| \mathcal{G}(s,\varepsilon)\right] = 1$$

and concludes the proof of Lemma A.3. □

Proof of Theorem 2.7 Now, we can prove Theorem 2.7 using Lemma A.3.

Let $\widehat{Q}_s = \texttt{estimateQ}(s, \varepsilon)$. We have $\widehat{V}(s) = Fs(\widehat{Q}_s)$. As in the proof of Lemma A.3, let the reward $R_{s,a}^{(k)}$ and state $Z_{s,a}^{(k)}$ be the random variables associated to the k-th call to the generative model used to compute $\widehat{Q}_s(a)$ in `estimateQ`, for $k \in \{1, \cdots, N(\varepsilon)\}$.

We have:

$$\widehat{Q}_s(a) = \frac{1}{N(\varepsilon)} \sum_{k=1}^{N(\varepsilon)} R_{s,a}^{(k)} + \gamma \texttt{sampleV}\left(Z_{s,a}^{(k)}, \varepsilon/\sqrt{\gamma}\right).$$

Consider the event $\mathcal{E}$ defined as

$$\mathcal{E} := \bigcap_{k=1}^{N(\varepsilon)} \mathcal{G}\left(Z_{s,a}^{(k)}, \frac{\varepsilon}{\sqrt{\gamma}}\right).$$

By the same arguments as in the proof of Lemma A.3,

- In $\mathcal{E}$, we have $\|\widehat{Q}_s - Q_s\|_\infty \leq \varepsilon$;
- $\mathbf{P}[\mathcal{E}] \geq 1 - \delta' N(\varepsilon) n_{\texttt{sampleV}}(\varepsilon, \delta') = 1 - \delta' n(\varepsilon, \delta')$.

Hence,

$$\mathbf{P}\left[\forall a \in \mathcal{A},\ |\widehat{Q}_s(a) - Q_s(a)| > \varepsilon\right] \leq \delta' n(\varepsilon, \delta').$$

To conclude the proof, for every $\varepsilon > 0$ and every $\delta > 0$, we need to be able to find a value of δ' such that $\delta' n(\varepsilon, \delta') \leq \delta$. That is, given ε and δ, we need to find δ' such that

$$\delta' \frac{c_1}{\varepsilon^4} \log\left(\frac{c_2}{\delta'}\right) \left[c_3 \log\left(\frac{c_4}{\varepsilon}\right)\right]^{\log_2\left(c_5\left(\log\left(\frac{c_2}{\delta'}\right)\right)\right)} \leq \delta.$$

Such value exists, since the term on the LHS tends to 0 as $\delta' \to 0$, and it depends on ε. We will show that this dependence is polynomial when $\varepsilon \to 0$.

Let $\delta' = \varepsilon^5$. There exists a value $\widetilde{\varepsilon}$ that depends on δ such that

$$\forall \varepsilon \leq \widetilde{\varepsilon}, \quad \varepsilon^5 \frac{c_1}{\varepsilon^4} \log\left(\frac{c_2}{\varepsilon^5}\right) \left[c_3 \log\left(\frac{c_4}{\varepsilon}\right)\right]^{\log_2\left(c_5\left(\log\left(\frac{c_2}{\varepsilon^5}\right)\right)\right)} \leq \delta.$$

since the term on the LHS tends to 0 as $\varepsilon \to 0$, as a consequence of Lemma A.5.

Putting it all together, we can choose δ' as follows:

$$\delta' = \begin{cases} \widetilde{\delta} \text{ such that } \widetilde{\delta} \frac{c_1}{\widetilde{\varepsilon}^4} \log\left(\frac{c_2}{\widetilde{\delta}}\right) \left[c_3 \log\left(\frac{c_4}{\widetilde{\varepsilon}}\right)\right]^{\log_2\left(c_5\left(\log\left(\frac{c_2}{\widetilde{\delta}}\right)\right)\right)} \leq \delta, & \text{if } \varepsilon > \widetilde{\varepsilon}, \\ \varepsilon^5, & \text{if } \varepsilon \leq \widetilde{\varepsilon} \end{cases}$$

which is $\delta' = \mathcal{O}(\varepsilon^5)$.

Lemma A.6 implies that, for this choice of δ', the sample complexity is of order $\mathcal{O}(1/\varepsilon^{4+c})$ for any $c > 0$.

A.3 Technical Lemmas

Lemma A.4. *Let a, b, c be constants in $[1, +\infty[$ and δ be a constant in $]0, 1]$. Let*

$$x^* = \min\{x \geq 1 : \exp(-ax)(bx)^c \leq \delta\}.$$

Then,

$$x^* \leq \frac{1}{a}\left(\log\frac{1}{\delta} + c\log\left(\frac{4b}{a}\log\frac{1}{\delta} + \frac{b^2c^2}{a^2}\right)\right).$$

Proof. Let $\bar{x} := \frac{4}{a}\log\frac{1}{\delta} + \frac{bc^2}{a^2}$. For any $x \geq \frac{4}{a}\log\frac{1}{\delta} + \frac{bc^2}{a^2}$, we have

$$ax \geq \log\frac{1}{\delta} + c\sqrt{bx} \geq \log\frac{1}{\delta} + c\log(bx) = \log\left(\frac{(bx)^c}{\delta}\right)$$

and $\exp(-ax)(bx)^c \leq \delta$. This implies that $\exp(-a\bar{x})(b\bar{x})^c \leq \delta$ and $x^* \leq \bar{x}$.

Now, let

$$x' := \frac{1}{a}\left(\log\frac{1}{\delta} + c\log(b\bar{x})\right) = \frac{1}{a}\left(\log\frac{1}{\delta} + c\log\left(\frac{4b}{a}\log\frac{1}{\delta} + \frac{b^2c^2}{a^2}\right)\right)$$

such that $\exp(-ax')(b\bar{x})^c \leq \delta$.

Now, we claim that $x^* \leq x'$. If $x' > \bar{x}$, then we have immediately $x^* \leq \bar{x} < x'$. Otherwise, if $x' \leq \bar{x}$:

$$\exp(-ax')(bx')^c \leq \exp(-ax')(b\bar{x})^c \leq \delta$$

which implies that $x^* \leq x'$ by the definition of x^*. □

Lemma A.5. $\forall a, b, c > 0\ \lim_{x\to\infty} \frac{1}{x^c}\exp\left(a[\log\log(x^b)]^2\right) = 0.$

Proof. We have

$$\begin{aligned}\frac{1}{x^c}\exp\left(a[\log\log(x^b)]^2\right) &= \exp\left(a[\log\log(x^b)]^2 - c\log x\right)\\ &= \exp\left(a[\log u]^2 - \frac{c}{b}u\right), \quad \text{by setting } u = \log(x^b)\end{aligned}$$

And, for any $k > 0$, we have $\lim_{u\to\infty} \log^2 u - ku = -\infty$, which allows us to conclude. □

For the following results, we denote by $n(\varepsilon, \delta')$ the number of calls to the generative model before `SmoothCruiser` terminates, when called with parameters (ε, δ').

Lemma A.6. *If we set $\delta' = \delta'(\varepsilon) = \varepsilon^5$, we have:*

$$n(\varepsilon, \delta'(\varepsilon)) = \mathcal{O}\left(\frac{1}{\varepsilon^{4+c}}\right), \quad \forall c > 0$$

Proof. We have, from Lemma A.2,

$$n_{\texttt{sampleV}}(\varepsilon, \delta'(\varepsilon)) \leq \eta_1 \left[\log_{\frac{1}{\gamma}}\left(\frac{\bar{\varepsilon}/\gamma}{\varepsilon}\right)\right]^{\eta_2(\varepsilon^5)} \frac{1}{\varepsilon^2} = \underbrace{\left[\log_{\frac{1}{\gamma}}\left(\frac{\bar{\varepsilon}/\gamma}{\varepsilon}\right)\right]^{\log_2\left(k\log\left(\frac{2A}{\varepsilon^5}\right)\right)}}_{\text{(A)}} \frac{1}{\varepsilon^2}$$

where k is a constant that does not depend on ε. The term (A) can be rewritten as:

$$\begin{aligned}\left[\log_{\frac{1}{\gamma}}\left(\frac{\bar{\varepsilon}/\gamma}{\varepsilon}\right)\right]^{\log_2\left(k\log\left(\frac{2A}{\varepsilon^5}\right)\right)} &= \left[c_1 \log\left(\frac{c_2}{\varepsilon}\right)\right]^{c_3 \log\left[k\log\left(\frac{c_4}{\varepsilon^5}\right)\right]} \\ &= \exp\left\{c_3 \log\left[k\log\left(\frac{c_4}{\varepsilon^5}\right)\right] \log\left(c_1 \log\left(\frac{c_2}{\varepsilon}\right)\right)\right\}\end{aligned}$$

which can be shown to be $\mathcal{O}\left(\frac{1}{\varepsilon^c}\right)$ for any $c > 0$ by applying Lemma A.5 after some algebraic manipulations.

Hence,

$$n_{\texttt{sampleV}}(\varepsilon, \delta'(\varepsilon)) = \frac{1}{\varepsilon^2}\mathcal{O}\left(\frac{1}{\varepsilon^c}\right) = \mathcal{O}\left(\frac{1}{\varepsilon^{2+c}}\right), \quad \forall c > 0.$$

The fact that $n(\varepsilon, \delta') = N(\varepsilon) n_{\texttt{sampleV}}(\varepsilon, \delta')$, where $N(\varepsilon) = \widetilde{\mathcal{O}}\left(1/\varepsilon^2\right)$, concludes the proof. □

Corollary A.7. *If we set $\delta' = \delta'(\varepsilon) = \varepsilon^5$, we have:*

$$\lim_{\varepsilon\to 0} \delta'(\varepsilon) n(\varepsilon, \delta'(\varepsilon)) = 0$$

Proof. It is an immediate consequence of Lemma A.6 by taking $c \in]0, 1[$. □

Appendix B

Complements on Chapter 3

B.1 Change of Distribution: Proof of Lemma 3.6

Let

$$b_h^t := (s_1^1, a_1^1, \dots, s_H^1, \dots, s_1^t, a_1^t, \dots, s_h^t)$$

be a sequence of state actions up to stage h of episode t. The pushforward measure of $\mathbf{P}_{\mathcal{M}}$ under B_H^τ is given by

$$\forall T,\ \forall b_H^T, \quad \mathbf{P}_{\mathcal{M}}^{B_H^\tau}\left[b_H^T\right] = \mathbf{P}_{\mathcal{M}}^{B_H^T}\left[\tau = T, b_H^T\right] = \mathbf{P}_{\mathcal{M}}\left[\tau = T \middle| B_H^T = b_H^T\right] \mathbf{P}_{\mathcal{M}}^{B_H^T}\left[b_H^T\right].$$

If $\mathbf{P}_{\mathcal{M}'}\left[\tau = T \middle| B_H^T = b_H^T\right] > 0$ and $\mathbf{P}_{\mathcal{M}'}^{B_H^T}\left[b_H^T\right] > 0$, we have

$$\frac{\mathbf{P}_{\mathcal{M}}^{B_H^\tau}\left[b_H^T\right]}{\mathbf{P}_{\mathcal{M}'}^{B_H^\tau}\left[b_H^T\right]} = \frac{\mathbf{P}_{\mathcal{M}}\left[\tau = T \middle| B_H^T = b_H^T\right] \mathbf{P}_{\mathcal{M}}^{B_H^T}\left[b_H^T\right]}{\mathbf{P}_{\mathcal{M}'}\left[\tau = T \middle| B_H^T = b_H^T\right] \mathbf{P}_{\mathcal{M}'}^{B_H^T}\left[b_H^T\right]} = \frac{\mathbf{P}_{\mathcal{M}}^{B_H^T}\left[b_H^T\right]}{\mathbf{P}_{\mathcal{M}'}^{B_H^T}\left[b_H^T\right]}$$

where we use the fact that $\mathbf{P}_{\mathcal{M}}\left[\tau = T \middle| B_H^T = b_H^T\right] = \mathbf{P}_{\mathcal{M}'}\left[\tau = T \middle| B_H^T = b_H^T\right]$ since the event $\{\tau = T\}$ depends only on B_H^T. This implies that

$$\mathbf{P}_{\mathcal{M}}^{B_H^\tau}\left[b_H^T\right] \log\left(\frac{\mathbf{P}_{\mathcal{M}}^{B_H^\tau}\left[b_H^T\right]}{\mathbf{P}_{\mathcal{M}'}^{B_H^\tau}\left[b_H^T\right]}\right) = \mathbf{P}_{\mathcal{M}}\left[\tau = T \middle| B_H^T = b_H^T\right] \mathbf{P}_{\mathcal{M}}^{B_H^T}\left[b_H^T\right] \log\left(\frac{\mathbf{P}_{\mathcal{M}}^{B_H^T}\left[b_H^T\right]}{\mathbf{P}_{\mathcal{M}'}^{B_H^T}\left[b_H^T\right]}\right)$$

under the convention that $0 \log(0/0) = 0$. Hence,

$$\mathrm{KL}\left(\mathbf{P}_{\mathcal{M}}^{B_H^\tau}, \mathbf{P}_{\mathcal{M}'}^{B_H^\tau}\right) = \sum_{T=1}^{\infty} \sum_{b_H^T} \mathbf{P}_{\mathcal{M}}^{B_H^\tau}\left[b_H^T\right] \log\left(\frac{\mathbf{P}_{\mathcal{M}}^{B_H^\tau}\left[b_H^T\right]}{\mathbf{P}_{\mathcal{M}'}^{B_H^\tau}\left[b_H^T\right]}\right)$$

$$
\begin{aligned}
&= \sum_{T=1}^{\infty} \sum_{b_H^T} \mathbf{P}_{\mathcal{M}}\left[\tau = T \middle| B_H^T = b_H^T\right] \mathbf{P}_{\mathcal{M}}^{B_H^T}\left[b_H^T\right] \log\left(\frac{\mathbf{P}_{\mathcal{M}}^{B_H^T}\left[b_H^T\right]}{\mathbf{P}_{\mathcal{M}'}^{B_H^T}\left[b_H^T\right]}\right) \\
&= \sum_{T=1}^{\infty} \sum_{b_H^T} \mathbf{P}_{\mathcal{M}}\left[\tau = T \middle| B_H^T = b_H^T\right] \mathbf{P}_{\mathcal{M}}^{B_H^T}\left[b_H^T\right] \sum_{t=1}^{T} \sum_{h=1}^{H-1} \log\left(\frac{p_h(s_{h+1}^t|s_h^t,a_h^t)}{p'_h(s_{h+1}^t|s_h^t,a_h^t)}\right) \\
&= \sum_{T=1}^{\infty} \mathbf{E}_{\mathcal{M}}\left[\mathbb{1}\{\tau = T\} \sum_{t=1}^{T} \sum_{h=1}^{H-1} \log\left(\frac{p_h(S_{h+1}^t|S_h^t,A_h^t)}{p'_h(S_{h+1}^t|S_h^t,A_h^t)}\right)\right] \\
&= \mathbf{E}_{\mathcal{M}}\left[\sum_{t=1}^{\tau} \sum_{h=1}^{H-1} \log\left(\frac{p_h(S_{h+1}^t|S_h^t,A_h^t)}{p'_h(S_{h+1}^t|S_h^t,A_h^t)}\right)\right].
\end{aligned}
$$

Now, we apply Lemma B.3 by taking $X_t = \sum_{h=1}^{H-1} \log\left(\frac{p_h(S_{h+1}^t|S_h^t,A_h^t)}{p'_h(S_{h+1}^t|S_h^t,A_h^t)}\right)$ and $\mathcal{F}_t = \mathcal{F}_H^t$. Notice that X_t is bounded almost surely, since when $p_h(S_{h+1}^t|S_h^t,A_h^t) = p'_h(S_{h+1}^t|S_h^t,A_h^t) = 0$, the trajectory containing $(S_h^t, A_h^t, S_{h+1}^t)$ has zero probability. Lemma B.3 and the Markov property give us

$$
\begin{aligned}
&\mathrm{KL}\left(\mathbf{P}_{\mathcal{M}}^{B_H^\tau}, \mathbf{P}_{\mathcal{M}'}^{B_H^\tau}\right) = \mathbf{E}_{\mathcal{M}}\left[\sum_{t=1}^{\tau} \sum_{h=1}^{H-1} \mathbf{E}_{\mathcal{M}}\left[\log\left(\frac{p_h(S_{h+1}^t|S_h^t,A_h^t)}{p'_h(S_{h+1}^t|S_h^t,A_h^t)}\right) \middle| S_h^t, A_h^t\right]\right] \\
&= \mathbf{E}_{\mathcal{M}}\left[\sum_{t=1}^{\tau} \sum_{h=1}^{H-1} \mathrm{KL}\left(p_h(\cdot|S_h^t,A_h^t), p'_h(\cdot|S_h^t,A_h^t)\right)\right] = \sum_{s,a,h} \mathbf{E}_{\mathcal{M}}\left[N_{h,s,a}^\tau\right] \mathrm{KL}\left(p_h(\cdot|s,a), p'_h(\cdot|s,a)\right).
\end{aligned}
$$

B.2 PAC-MDP Lower Bound: Proof of Corollary 3.10

Recall that $\mathcal{N}_\varepsilon^{\mathrm{PAC}} = \sum_{t=1}^{\infty} \mathbb{1}\{\rho^* - \rho^{\pi_t} > \varepsilon\}$ and let

$$
T(\varepsilon, \delta) := \frac{1}{6912} \frac{H^3 SA}{\varepsilon^2} \log\left(\frac{1}{\delta}\right) - 1.
$$

We proceed by contradiction and assume that the claim in Corollary 3.10 is false. Then we have

$$
\text{for all MDP } \mathcal{M}, \quad \mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}\left[\mathcal{N}_\varepsilon^{\mathrm{PAC}} \leq T(\varepsilon,\delta)\right] \geq 1 - \delta, \tag{B.1}
$$

that is, the algorithm satisfies Definition 3.3 with $F_{\mathrm{PAC}}(S, A, H, 1/\varepsilon, \log(1/\delta)) = T(\varepsilon, \delta)$. In particular, (B.1) holds for any MDP in the class $\mathcal{C}_{\overline{H},\widetilde{\varepsilon}}$ used to prove Theorem 3.9, for which $\overline{H} = H/3$ and $\widetilde{\varepsilon} = 2\varepsilon/(H - \overline{H} - d)$.

This allows us to build from $\boldsymbol{\pi}$ a best-policy identification algorithm that outputs an ε-optimal policy with probability larger than $1 - \delta$ for every MDP in $\mathcal{C}_{H/3,\widetilde{\varepsilon}}$. We proceed as follows: the sampling rule is that of the algorithm $\boldsymbol{\pi}$ while the stopping rule is deterministic and set to $\tau := 2T(\varepsilon, \delta) + 1$. Letting $N_t(\pi)$ be the number of times that the algorithm plays a deterministic policy π up to episode t, we let the recommendation rule be $\widehat{\pi}_\tau = \mathrm{argmax}_\pi N_\tau(\pi)$.

For every $\mathcal{M} \in \mathcal{C}_{H/3,\widetilde{\varepsilon}}$, the event $\left\{\mathcal{N}_\varepsilon^{\mathrm{PAC}} \leq T(\varepsilon,\delta)\right\}$ implies $\widehat{\pi}_\tau = \pi^*$. This is trivial for $\mathcal{M}_0$, where any policy is optimal, and this holds for any other $\mathcal{M}_{(h^*,\ell^*a^*)} \in \mathcal{C}_{H/3,\widetilde{\varepsilon}}$ since there is a unique optimal policy π^* and it satisfies $(\rho^{\pi^*} - \rho^{\pi}) = 2\varepsilon > \varepsilon$ in $\mathcal{M}_{(h^*,\ell^*a^*)}$ for any other deterministic policy π. Hence, if $\widehat{\pi}_\tau \neq \pi^*$, the number of mistakes $\mathcal{N}_\varepsilon^{\mathrm{PAC}}$ would be larger than $T(\varepsilon,\delta)$. Thus we proved that the BPI algorithm that we defined satisfies

$$\forall \mathcal{M} \in \mathcal{C}_{H/3,\widetilde{\varepsilon}}, \quad \mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}\left[\widehat{\pi}_\tau = \pi^*\right] \geq \mathbf{P}_{\boldsymbol{\pi},\mathcal{M}}\left[\mathcal{N}_\varepsilon^{\mathrm{PAC}} \leq T(\varepsilon,\delta)\right] \geq 1-\delta.$$

Under these conditions, we established in the proof of Theorem 3.9 that, for $\mathcal{M}_0 \in \mathcal{C}_{H/3,\widetilde{\varepsilon}}$,

$$\tau = \mathbb{E}_{\mathcal{M}_0}[\tau] \geq \frac{1}{3456}\frac{H^3SA}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)$$

which yields

$$2T(\varepsilon,\delta) + 1 \geq \frac{1}{3456}\frac{H^3SA}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)$$

and contradicts the definition of $T(\varepsilon,\delta)$.

B.3 Technical Lemmas for Lower-Bound Proofs

Lemma B.1. *If* $\varepsilon \in [0,1/4]$, *then* $\mathrm{kl}(1/2, 1/2+\varepsilon) \leq 4\varepsilon^2$.

Proof. Using the inequality $-\log(1-x) \leq 1/(1-x) - 1$ for any $0 < x < 1$, we obtain

$$\mathrm{kl}(1/2, 1/2+\varepsilon) = -\frac{1}{2}\log(1-4\varepsilon^2) \leq \frac{1}{2}\left(\frac{1}{1-4\varepsilon^2} - 1\right) = \frac{2\varepsilon^2}{1-4\varepsilon^2}.$$

If $\varepsilon \leq 1/4$, then $1 - 4\varepsilon^2 \geq 3/4 > 1/2$, which implies the result. □

Lemma B.2. *For any* $p, q \in [0,1]$,

$$\mathrm{kl}(p,q) \geq (1-p)\log\left(\frac{1}{1-q}\right) - \log(2).$$

Proof. If follows from the definition of $\mathrm{kl}(p,q)$ and the fact that the entropy $H(p) := p\log(1/p) + (1-p)\log(1/(1-p))$ satisfies $H(p) \leq \log(2)$:

$$\begin{aligned}
\mathrm{kl}(p,q) &= p\log\left(\frac{p}{q}\right) + (1-p)\log\left(\frac{1-p}{1-q}\right) \\
&= (1-p)\log\left(\frac{1}{1-q}\right) + (1-p)\log\left(\frac{1}{1-q}\right) + p\log\left(\frac{1}{q}\right) - H(p) \\
&\geq (1-p)\log\left(\frac{1}{1-q}\right) - \log(2).
\end{aligned}$$

□

Lemma B.3. *Let $(X_t)_{t\geq 1}$ be a stochastic process adapted to the filtration $(\mathcal{F}_t)_{t\geq 1}$. Let τ be a stopping time with respect to $(\mathcal{F}_t)_{t\geq 1}$ such that $\tau < \infty$ with probability 1. If there exists a constant c such that $\sup_t |X_t| \leq c$ almost surely, then*

$$\mathbf{E}\left[\sum_{t=1}^{\tau} X_t\right] = \mathbf{E}\left[\sum_{t=1}^{\tau} \mathbf{E}\left[X_t|\mathcal{F}_{t-1}\right]\right].$$

Proof. Let $M_n := \sum_{t=1}^{n} \left(X_t - \mathbf{E}\left[X_t|\mathcal{F}_{t-1}\right]\right)$. Then, M_n is a martingale and, by Doob's optional stopping theorem, $\mathbf{E}\left[M_\tau\right] = \mathbf{E}\left[M_0\right] = 0$. □

Lemma B.4. *Let L be the number of leaves in a balanced A-ary tree with S nodes and $A \geq 2$. Then, $L \geq S/4$.*

Proof. Let d be the depth of the tree. There exists an integer R such that $0 < R \leq A^d$ such that

$$S = \frac{A^d - 1}{A - 1} + R.$$

The number of leaves is given by $L = R + A^{d-1} - \left\lceil \frac{R}{A} \right\rceil$. We consider two cases: either $\frac{A^d-1}{A-1} \leq \frac{S}{2}$ or $\frac{A^d-1}{A-1} > \frac{S}{2}$. If $\frac{A^d-1}{A-1} \leq \frac{S}{2}$, we have $R \geq S/2$ which implies $L \geq S/2 > S/4$. If $\frac{A^d-1}{A-1} > \frac{S}{2}$, we have $L \geq A^{d-1} > \frac{1}{A} + \frac{S}{2}\left(1 - \frac{1}{A}\right) \geq S/4$. □

B.4 Complements on the proof of Theorem 3.14 (Regret of `UCBVI`)

B.4.1 Concentration Inequalities

Lemma B.5. *Let $(X_t)_t$ be a stochastic process adapted to the filtration $(\mathcal{F}_t)_t$ for $t \in \mathbb{N}$ such that, for a given $\sigma > 0$, we have $\mathbf{E}\left[\exp(\lambda X_t)|\mathcal{F}_{t-1}\right] \leq \exp(\lambda^2\sigma^2/2)$ almost surely for all $\lambda \in \mathbb{R}$. Then,*

$$\mathbf{P}\left[\exists n : \frac{1}{n}\sum_{t=1}^{n} X_t \geq \sqrt{\frac{2\sigma^2}{n}\log\left(\frac{n(n+1)}{\delta}\right)}\right] \leq \delta$$

Proof. For a fixed n, we have

$$\mathbf{P}\left[\frac{1}{n}\sum_{t=1}^{n} X_t \geq \sqrt{\frac{2\sigma^2}{n}\log\left(\frac{n(n+1)}{\delta}\right)}\right] \leq \frac{\delta}{n(n+1)}$$

by the Azuma-Hoeffding inequality. A union bound over n and the fact that $\sum_{n=1}^{\infty} 1/(n(n+1)) = 1$ gives the result. □

Lemma B.6 (Lemma 3 by [Dom+21d]). *Let $(Y_t)_{t\in\mathbb{N}^*}$ and $(w_t)_{t\in\mathbb{N}^*}$ be two sequences of random variables adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$. Assume that (i) $w_t \in [0,1]$; (ii) w_t is $\mathcal{F}_{t-1}$-measurable; (iii) $|Y_t| \le b$ almost surely; and (iv) $\mathbf{E}[Y_t|\mathcal{F}_{t-1}] = 0$. Let*

$$S_t := \sum_{i=1}^{t} w_i Y_i, \quad V_t := \sum_{i=1}^{t} w_i^2 \cdot \mathbf{E}\left[Y_i^2|\mathcal{F}_{i-1}\right], \quad \text{and} \quad W_t := \sum_{i=1}^{t} w_i$$

and let $g(x) := (x+1)\log(x+1) - x$. For all $\delta > 0$, we have

$$\mathbf{P}\left[\exists t \ge 1,\ |S_t| \le \sqrt{2V_t \log\left(\frac{4e(2t+1)}{\delta}\right)} + 3b\log\left(\frac{4e(2t+1)}{\delta}\right)\right] \ge 1-\delta.$$

B.4.2 Proof of Lemma 3.15

To simplify the notations, let $f := V^*_{h+1}$. Since the rewards are in $[0,1]$, $V^*_{h+1}(s) \in [0,H]$ for any s. We have

$$(p_h - \widehat{p}^t_h)V^*_{h+1}(s,a) = \frac{1}{n^t_h(s,a)} \sum_{i=1}^{t-1} \mathbb{1}\left\{(s^i_h, a^i_h) = (s,a)\right\}\left(p_h f(s,a) - f(s^i_{h+1})\right).$$

Consider the stopping times $(\tau^j_h(s,a))_j$ defined as

$$\begin{aligned}
\tau^1_h(s,a) &:= \min\left\{k : (s,a) = (s^k_h, a^k_h)\right\}, \\
\tau^j_h(s,a) &:= \min\left\{k > \tau^{j-1}_h(s,a) : (s,a) = (s^k_h, a^k_h)\right\}, \text{ for } j > 1,
\end{aligned} \tag{B.2}$$

such that $\tau^j_h(s,a)$ represents the episode where (s,a) was visited for the j-th time at stage h. Let

$$W^i_h(s,a,f) := \mathbb{1}\left\{(s^i_h, a^i_h) = (s,a)\right\}\left(p_h f(s,a) - f(s^i_{h+1})\right),$$

that satisfies $|W^i_h(s,a,f)| \le H$ almost surely and $\mathbf{E}\left[W^i_h(s,a,f)|\mathcal{F}^i_h\right] = 0$. Now, let

$$\overline{W}^j_h(s,a,f) = W^{\tau^j_h}_h(s,a,f) \quad \text{and} \quad \overline{\mathcal{F}}^j_h = \mathcal{F}^{\tau^j_h}_h \quad \text{for} \quad j \ge 1. \tag{B.3}$$

By Lemma B.7, we have $\mathbf{E}\left[\overline{W}^j_h(s,a,f)|\overline{\mathcal{F}}^j_h\right] = 0$. Notice that

$$(p_h - \widehat{p}^t_h)V^*_{h+1}(s,a) = \frac{1}{n^t_h(s,a)} \sum_{j=1}^{n^t_h(s,a)} \overline{W}^j_h(s,a,f),$$

which implies, by Lemma B.5, that

$$\mathbf{P}\left[\exists t \geq 1,\ (p_h - \widehat{p}_h^t)V_{h+1}^*(s,a) \geq \sqrt{\frac{2H^2}{n_h^t(s,a)}\log\left(\frac{6HSAn_h^t(s,a)(n_h^t(s,a)+1)}{\delta}\right)}\right] \leq \frac{\delta}{3HSA}.$$

The definition of the bonus $\mathfrak{b}_h^t(s,a)$ in Equation (3.17) and a union bound over $(h,s,a) \in [H] \times \mathcal{S} \times \mathcal{A}$ concludes the proof.

B.4.3 Proof of Lemma 3.16

Consider the sequence of random variables $(\overline{W}_h^j(s,a,f))_j$ and the filtration $(\overline{\mathcal{F}}_h^j)_j$ defined in Equation (B.3) in the proof of Lemma 3.15. We have $|\overline{W}_h^i(s,a,f)| \leq H$ almost surely, $\mathbf{E}\left[\overline{W}_h^i(s,a,f)|\overline{\mathcal{F}}_h^i\right] = 0$, and

$$(p_h - \widehat{p}_h^t)f(s,a) = \frac{1}{n_h^t(s,a)}\sum_{j=1}^{n_h^t(s,a)} \overline{W}_h^j(s,a,f).$$

Let $\tau = \tau_h^j(s,a)$, as defined in Equation (B.2), representing the episode where (s,a) was visited for the j-time at stage h. The conditional variance of $\overline{W}_h^j(s,a,f)$ given $\overline{\mathcal{F}}_h^i$ is bounded as follows:

$$\begin{aligned}\mathbf{E}\left[\overline{W}_h^j(s,a,f)^2|\overline{\mathcal{F}}_h^j\right] &= \mathbf{E}\left[\left(p_h f(s,a) - f(s_{h+1}^\tau)\right)^2|\overline{\mathcal{F}}_h^j\right] \leq \mathbf{E}\left[f(s_{h+1}^\tau)^2|\overline{\mathcal{F}}_h^j\right] \\ &\leq \|f\|_\infty \mathbf{E}\left[f(x_{h+1}^\tau)|\overline{\mathcal{F}}_h^j\right] \leq Hp_h f(s,a).\end{aligned}$$

For fixed (t,s,a,h,f), Lemma B.6 implies

$$\mathbf{P}\left[\frac{1}{t}\sum_{j=1}^{t}\overline{W}_h^j(s,a,f) \geq \sqrt{2Hp_h f(s,a)\frac{\beta(t,\delta)}{t}} + 3H\frac{\beta(t,\delta)}{t}\right] \leq \frac{\delta}{t(t+1)}.$$

where $\beta(t,\delta) := \log\left(66t^2(t+1)/\delta\right)$, which implies

$$\mathbf{P}\left[\frac{1}{t}\sum_{j=1}^{t}\overline{W}_h^j(s,a,f) \geq \frac{1}{H}p_h f(s,a) + \frac{(H^2/2+3H)\beta(t,\delta)}{t}\right] \leq \frac{\delta}{t(t+1)}. \tag{B.4}$$

since $2\sqrt{xy} \leq (x+y)$ for any $x,y \geq 0$.

Now, we extend the inequality above so that it holds for any $f \in \mathcal{V}$ and any $t \geq 1$. In order to do so, let $\mathcal{C}_{1/t} := \left\{f_1, \ldots, f_{|\mathcal{C}_{1/t}|}\right\}$ be a $1/t$-covering of $(\mathcal{V}, \|\cdot\|_\infty)$ such that $|\mathcal{C}_{1/t}| = (Ht)^S$ and, for any $f \in \mathcal{V}$, there exists a $f_i \in \mathcal{C}_{1/t}$ satisfying $\|f - f_i\|_\infty \leq 1/t$. Let

$$\widetilde{\beta}(t,\delta) := \beta\left(t, \frac{\delta}{3HSA(Ht)^S}\right).$$

We have

$$\begin{aligned}
&\mathbf{P}\left[\exists t \geq 1, \exists f \in \mathcal{V},\ \frac{1}{t}\sum_{j=1}^{t} \overline{W}_h^j(s,a,f) \geq \frac{1}{H} p_h f(s,a) + \frac{(H^2/2+3H)\widetilde{\beta}(t,\delta)}{t} + \frac{3}{t}\right] \\
&\leq \mathbf{P}\left[\exists t \geq 1, \exists f \in \mathcal{C}_{1/t},\ \frac{1}{t}\sum_{j=1}^{t} \overline{W}_h^j(s,a,f) \geq \frac{1}{H} p_h f(s,a) + \frac{(H^2/2+3H)\widetilde{\beta}(t,\delta)}{t}\right] \\
&\leq \sum_{t=1}^{\infty} \sum_{f \in \mathcal{C}_{1/t}} \mathbf{P}\left[\frac{1}{t}\sum_{j=1}^{t} \overline{W}_h^j(s,a,f) \geq \frac{1}{H} p_h f(s,a) + \frac{(H^2/2+3H)\widetilde{\beta}(t,\delta)}{t}\right] \\
&\leq \sum_{t=1}^{\infty} \sum_{f \in \mathcal{C}_{1/t}} \frac{\delta}{3HSA|\mathcal{C}_{1/t}|t(t+1)} = \frac{\delta}{3HSA}, \quad \text{using Equation (B.4).}
\end{aligned}$$

A union bound over $(h,s,a) \in [H] \times \mathcal{S} \times \mathcal{A}$ concludes the proof.

B.4.4 Technical Lemmas

Lemma B.7. *Let $(X_t)_t$ be a stochastic process adapted to the filtration $(\mathcal{F}_t)_t$ such that $\mathbf{E}[X_t|\mathcal{F}_t] = 0$ almost surely for all t. Let τ be a stopping time. Then, if X_t is bounded almost surely for all t and if $X_\tau \mathbb{1}\{\tau = \infty\} = 0$, we have $\mathbf{E}[X_\tau|\mathcal{F}_\tau] = 0$ almost surely.*

Proof. Let $A \in \mathcal{F}_\tau$. Then,

$$\begin{aligned}
\mathbf{E}[\mathbb{1}_A X_\tau] = \mathbf{E}[\mathbb{1}_A \mathbf{E}[X_\tau|\mathcal{F}_\tau]] &= \mathbf{E}\left[\mathbb{1}_A \sum_{t=1}^{\infty} X_t \mathbb{1}\{\tau = t\}\right] \\
&= \sum_{t=1}^{\infty} \mathbf{E}[\mathbb{1}_A X_t \mathbb{1}\{\tau = t\}] \quad \text{by the dominated convergence theorem} \\
&= \sum_{t=1}^{\infty} \mathbf{E}\left[\mathbb{1}_{A\cap\{\tau=t\}} \mathbf{E}[X_t|\mathcal{F}_t]\right] \quad \text{since } A \cap \{\tau = t\} \in \mathcal{F}_t \\
&= 0.
\end{aligned}$$

Notice that, since $X_t \leq M$ for all t for some $M > 0$, we have

$$\mathbf{E}\left[\sum_{t=1}^{n} \mathbb{1}_{A\cap\{\tau=t\}} X_t\right] \leq M\mathbf{P}[A \cap \{\tau \leq n\}] \leq M,$$

which justifies the use of the dominated convergence theorem. □

Appendix C

Complements on Chapter 4

C.1 Definitions

Definition C.1 (exploration bonuses)**.** *The exploration bonus at* (s,a) *at time* (t,h) *is defined as*

$$\begin{aligned}
&\mathtt{b}_h^t(s,a) := {}^{r}\mathtt{b}_h^t(s,a) + {}^{p}\mathtt{b}_h^t(s,a), \quad \textit{where}\\
&{}^{r}\mathtt{b}_h^t(s,a) := \sqrt{\frac{2\boldsymbol{\vartheta}_1^{\mathrm{r}}(T,\delta/6)}{\mathbf{C}_h^t(s,a)}} + \frac{\beta}{\mathbf{C}_h^t(s,a)} + \mathbf{b}_{\mathrm{r}}(T,\delta/6)\sigma, \quad \textit{and}\\
&{}^{p}\mathtt{b}_h^t(s,a) := \sqrt{\frac{2H^2\boldsymbol{\vartheta}_1^{\mathrm{p}}(T,\delta/6)}{\mathbf{C}_h^t(s,a)}} + \frac{\beta H}{\mathbf{C}_h^t(s,a)} + \mathbf{b}_{\mathrm{p}}(T,\delta/6)\sigma
\end{aligned}$$

where

$$\begin{aligned}
&\boldsymbol{\vartheta}_1^{\mathrm{r}}(t,\delta) = \widetilde{\mathcal{O}}\left(d_1\right) = \log\left(\frac{\mathcal{N}\left(\sigma^2/T, \mathcal{S}\times\mathcal{A}, \rho\right)\sqrt{1+t/\beta}}{\delta}\right)\\
&\mathbf{b}_{\mathrm{r}}(t,\delta) = \widetilde{\mathcal{O}}\left(L+\sqrt{d_1}\right) = \left(\frac{C_2}{2\beta^{3/2}}\sqrt{2\boldsymbol{\vartheta}_1^{\mathrm{r}}(t,\delta)} + \frac{4C_2}{\beta}\right) + 2L_{\mathrm{r}}L\left(1+\sqrt{\log^+(C_1 t/\beta)}\right)\\
&\boldsymbol{\vartheta}_1^{\mathrm{p}}(t,\delta) = \widetilde{\mathcal{O}}\left(d_1\right) = \log\left(\frac{H\mathcal{N}\left(\sigma^2/HT, \mathcal{S}\times\mathcal{A}, \rho\right)\sqrt{1+t/\beta}}{\delta}\right)\\
&\mathbf{b}_{\mathrm{p}}(t,\delta) = \widetilde{\mathcal{O}}\left(L+\sqrt{d_1}\right) = \left(\frac{C_2}{2\beta^{3/2}}\sqrt{2\boldsymbol{\vartheta}_1^{\mathrm{p}}(t,\delta)} + \frac{4C_2}{\beta}\right) + 2L_{\mathrm{p}}L\left(1+\sqrt{\log^+(C_1 t/\beta)}\right).
\end{aligned}$$

and where d_1 *is the covering dimension of* $(\mathcal{S}\times\mathcal{A},\rho)$ *and, for any* $z\in\mathbb{R}$, $\log^+(z) = \log(z+e)$.

That is, we define the constants κ_1, κ_2, *and* κ_3 *in Equation* (4.3) *as:*

$$\kappa_1 := \sqrt{2\boldsymbol{\vartheta}_1^{\mathrm{r}}(T,\delta/8)}/H + \sqrt{\boldsymbol{\vartheta}_1^{\mathrm{p}}(T,\delta/6)}; \quad \kappa_2 := 1+\frac{1}{\beta}; \quad \kappa_3 := \left(\mathbf{b}_{\mathrm{r}}(T,\delta/6) + \mathbf{b}_{\mathrm{p}}(T,\delta/6)\right)\sigma.$$

C.2 Proof of Theorem 4.7

Notation We denote by $\mathcal{F}_h^t$ the σ-algebra generated by all the state-action pairs observed up to time (t,h), that is, the h-th step of the t-th episode. For a metric space $\mathcal{U}, \rho$, we denote by $\mathcal{N}(\sigma, \mathcal{U}, \rho)$ is σ-covering number. We define $\log^+(z) := \log(z+e)$ for any $z \geq 0$.

C.2.1 Concentration

In this section, we provide the confidence intervals that will be used to prove the regret bound of `Kernel-UCBVI`. The main concentration results are presented in Lemma C.7, which defines an event $\mathcal{G}$ where all the confidence intervals hold, and we show that $\mathbf{P}[\mathcal{G}] \geq 1 - \delta/2$.

Concentration inequalities for weighted sums

We reproduce below the concentration inequalities for weighted sums that we proved in [Dom+21d].

Lemma C.2 (Hoeffding-type inequality [Dom+21d])**.** *Consider the sequences of random variables $(w_t)_{t\in\mathbb{N}^*}$ and $(Y_t)_{t\in\mathbb{N}^*}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$. Assume that, for all $t \geq 1$, w_t is $\mathcal{F}_{t-1}$ measurable and $\mathbf{E}[\exp(\lambda Y_t)|\mathcal{F}_{t-1}] \leq \exp(\lambda^2 c^2/2)$ for all $\lambda > 0$. Let $S_t := \sum_{s=1}^t w_s Y_s$ and $V_t := \sum_{s=1}^t w_s^2$, and assume $w_s \leq 1$ almost surely for all s. Then,for any $\beta > 0$, with probability at least $1-\delta$, for all $t \geq 1$,*

$$\frac{|S_t|}{\sum_{s=1}^t w_s + \beta} \leq \sqrt{2c^2 \log\left(\frac{\sqrt{1+t/\beta}}{\delta}\right) \frac{1}{\sum_{s=1}^t w_s + \beta}}\,.$$

Proof. See Lemma 2 of [Dom+21d]. □

Lemma C.3 (Bernstein-type inequality [Dom+21d])**.** *Consider the sequences of random variables $(w_t)_{t\in\mathbb{N}^*}$ and $(Y_t)_{t\in\mathbb{N}^*}$ adapted to a filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$. Let*

$$S_t := \sum_{s=1}^t w_s Y_s, \quad V_t := \sum_{s=1}^t w_s^2 \mathbf{E}\left[Y_s^2 \middle| \mathcal{F}_{s-1}\right] \quad \text{and} \quad W_t := \sum_{s=1}^t w_s\,,$$

Assume that, for all $t \geq 1$, (i) w_t is $\mathcal{F}_{t-1}$ measurable, (ii) $\mathbf{E}[Y_t|\mathcal{F}_{t-1}] = 0$, (iii) $w_t \in [0,1]$ almost surely, (iv) there exists $b > 0$ such that $|Y_t| \leq b$ almost surely. Then, for all $\beta > 0$, with probability at least $1-\delta$, for all $t \geq 1$,

$$\frac{|S_t|}{\beta + \sum_{s=1}^t w_s} \leq \sqrt{2\log\left(4e(2t+1)/\delta\right) \frac{V_t + b^2}{\left(\beta + \sum_{s=1}^t w_s\right)^2}} + \frac{2b}{3} \frac{\log(4e(2t+1)/\delta)}{\beta + \sum_{s=1}^t w_s}\,.$$

Proof. See Lemma 3 of [Dom+21d]. □

Hoeffding-type concentration inequalities

Lemma C.4. *With probability at least $1-\delta$, for all $(s,a,t,h) \in \mathcal{S}\times\mathcal{A}\times[T]\times[H]$, we have*

$$\left|(\widehat{p}_h^t - p_h)V_{h+1}^*(s,a)\right| \le \sqrt{\frac{2H^2\boldsymbol{\vartheta}_1^{\mathrm{p}}(t,\delta)}{\mathbf{C}_h^t(s,a)}} + \frac{\beta H}{\mathbf{C}_h^t(s,a)} + \mathbf{b}_{\mathrm{p}}(t,\delta)\sigma$$

where

$$\boldsymbol{\vartheta}_1^{\mathrm{p}}(t,\delta) = \widetilde{\mathcal{O}}\left(d_1\right) = \log\left(\frac{HT\mathcal{N}\left(\sigma^2/(HT),\mathcal{S}\times\mathcal{A},\rho\right)\sqrt{1+t/\beta}}{\delta}\right)$$

$$\mathbf{b}_{\mathrm{p}}(t,\delta) = \widetilde{\mathcal{O}}\left(L+\sqrt{d_1}\right) = \left(\frac{C_2}{2\beta^{3/2}}\sqrt{2\boldsymbol{\vartheta}_1^{\mathrm{p}}(k,\delta)} + \frac{4C_2}{\beta}\right) + 2L_{\mathrm{p}}L\left(1+\sqrt{\log(C_1t/\beta)}\right)$$

and where d_1 is the covering dimension of $(\mathcal{S}\times\mathcal{A},\rho)$.

Proof. Let $V = V_{h+1}^*$. For fixed (s,a,h), we have

$$\begin{aligned}
&|(\widehat{p}_h^t - p_h)V_{h+1}^*(s,a)| \\
&= \left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)\left(V(s_{h+1}^i) - \int_{\mathcal{S}}V(y)\mathrm{d}p_h(y|s,a)\right) - \frac{\beta}{\mathbf{C}_h^t(s,a)}\int_{\mathcal{S}}V(y)\mathrm{d}p_h(y|s,a)\right| \\
&\le \underbrace{\left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)\left(V(s_{h+1}^i) - \int_{\mathcal{S}}V(y)\mathrm{d}p_h(y|s_h^i,a_h^i)\right)\right|}_{①} \\
&\quad + \underbrace{\left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)\left(\int_{\mathcal{S}}V(y)\mathrm{d}p_h(y|s_h^i,a_h^i) - \int_{\mathcal{S}}V(y)\mathrm{d}p_h(y|s,a)\right)\right|}_{②} + \frac{\beta H}{\mathbf{C}_h^t(s,a)}.
\end{aligned}$$

Bounding ① (martingale term) Let $Y_i = V(s_{h+1}^i) - p_hV(s_h^i,a_h^i)$. Since $(Y_i)_i$ is a martingale difference sequence with respect to $(\mathcal{F}_h^i)_i$, we obtain from Lemma C.2 that, for a fixed tuple (s,a,t,h),

$$① = \left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)Y_i\right| \le \sqrt{2H^2\log\left(\frac{\sqrt{1+t/\beta}}{\delta}\right)\frac{1}{\mathbf{C}_h^t(s,a)}}$$

with probability at least $1-\delta$.

By Lemma C.21, the functions $(s,a)\mapsto\sqrt{1/\mathbf{C}_h^t(s,a)}$ and $(s,a)\mapsto\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)Y_i$ are Lipschitz continuous, with Lipschitz constants bounded by $C_2t/(2\sigma\beta^{3/2})$ and $4HC_2t/(\beta\sigma)$, respectively. Let $\mathcal{C}_{\mathcal{S}\times\mathcal{A}}\left(\sigma^2/HT\right)$ be a (σ^2/HT)-covering of $\mathcal{S}\times\mathcal{A}$. Using the Lipschitz continuity of the functions above and a union bound over $\mathcal{C}_{\mathcal{S}\times\mathcal{A}}\left(\sigma^2/(HT)\right)$ and over $(t,h)\in[T]\times[H]$, we

have

$$① = \left|\sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a) Y_i\right| \leq \sqrt{2H^2 \log\left(\frac{\sqrt{1+t/\beta}}{\delta}\right) \frac{1}{\mathbf{C}_h^t(s,a)}} + \left(\frac{C_2 t}{2\sigma\beta^{3/2}} \sqrt{2H^2 \log\left(\frac{\sqrt{1+t/\beta}}{\delta}\right)} + \frac{4HC_2 t}{\beta\sigma}\right) \frac{\sigma^2}{HT}$$

for all (s,a,t,h) with probability at least $1 - \delta HT\mathcal{N}\left(\sigma^2/(HT), \mathcal{S}\times\mathcal{A}, \rho\right)$.

Bounding ② (spatial bias term) We have

$$\begin{aligned}
② &= \left|\sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a) \left(\int_{\mathcal{S}} V(y) \mathrm{d}p_h(y|s_h^i, a_h^i) - \int_{\mathcal{S}} V(y) \mathrm{d}p_h(y|s,a)\right)\right| \\
&\leq L \sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a) \mathbb{W}_1\left(p_h(\cdot|s_h^i, a_h^i), p_h(\cdot|s,a)\right) \quad \text{by the definition of } \mathbb{W}_1(\cdot,\cdot) \\
&\leq L_{\mathrm{p}} L \sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a) \rho\left[(s_h^i, a_h^i), (s,a)\right] \quad \text{by Assumption 4.2} \\
&\leq 2\sigma L_{\mathrm{p}} L \left(1 + \sqrt{\log^+(C_1 t/\beta)}\right) \quad \text{by Lemma C.20.}
\end{aligned}$$

Putting together the bounds for ① and ② concludes the proof. □

Lemma C.5. *With probability at least $1-\delta$, for all $(s,a,t,h) \in \mathcal{S}\times\mathcal{A}\times[T]\times[H]$, we have*

$$|\widehat{r}_h^t(s,a) - r_h^t(s,a)| \leq \sqrt{\frac{2\vartheta_1^{\mathrm{r}}(t,\delta)}{\mathbf{C}_h^t(s,a)}} + \frac{\beta}{\mathbf{C}_h^t(s,a)} + \mathbf{b}_{\mathrm{r}}(t,\delta)\sigma$$

where

$$\vartheta_1^{\mathrm{r}}(t,\delta) = \widetilde{\mathcal{O}}(d_1) = \log\left(\frac{\mathcal{N}\left(\sigma^2/T, \mathcal{S}\times\mathcal{A}, \rho\right)\sqrt{1+t/\beta}}{\delta}\right)$$

$$\mathbf{b}_{\mathrm{r}}(t,\delta) = \widetilde{\mathcal{O}}\left(L + \sqrt{d_1}\right) = \left(\frac{C_2}{2\beta^{3/2}}\sqrt{2\vartheta_1^{\mathrm{r}}(t,\delta)} + \frac{4C_2}{\beta}\right) + 2L_{\mathrm{r}} L\left(1 + \sqrt{\log\left(\frac{C_1 t}{\beta}\right)}\right)$$

Proof. Almost identical to the proof of Lemma C.4, except for the fact that the rewards are bounded by 1 instead of H. □

C.2.2 Bernstein-type concentration inequality

Lemma C.6. *Let $\mathcal{L}(2L, 2H)$ be the class of $2L$-Lipschitz functions from $\mathcal{S}$ to $\mathbb{R}$ bounded by $2H$. With probability at least $1-\delta$, for all $(s,a,t,h) \in \mathcal{S}\times\mathcal{A}\times[T]\times[H]$ and for all $f \in \mathcal{L}(2L,2H)$, we have*

$$\left|(\widehat{p}_h^t - p_h)f(s,a)\right| \le \frac{1}{H}p_h|f|(s,a) + \frac{14H^2C_2\boldsymbol{\vartheta}_2(t,\delta)+2\beta H}{\mathbf{C}_h^t(s,a)} + \theta_{\mathrm{b}}^1(t,\delta)\sigma^{1+d_2} + \theta_{\mathrm{b}}^2(t,\delta)\sigma$$

where d_1 is the covering dimension of $(\mathcal{S}\times\mathcal{A},\rho)$, d_2 is the covering dimension of $(\mathcal{S},\rho_{\mathcal{S}})$ and

$$\boldsymbol{\vartheta}_2(t,\delta) = \widetilde{\mathcal{O}}\left(|\mathcal{C}'_\sigma| + d_1d_2\right) = \log\left(\frac{4e(2t+1)}{\delta}HT\mathcal{N}\left(\frac{\sigma^{2+d_2}}{H^2T},\mathcal{S}\times\mathcal{A},\rho\right)\left(\frac{2H}{L\sigma}\right)^{\mathcal{N}(\sigma,\mathcal{S},\rho_{\mathcal{S}})}\right)$$

$$\theta_{\mathrm{b}}^1(t,\delta) = \widetilde{\mathcal{O}}\left(|\mathcal{C}'_\sigma| + d_1d_2 + L\sigma\right) = \frac{2L_{\mathrm{p}}L\sigma}{H^2T} + \frac{4C_2}{H\beta} + \frac{14\boldsymbol{\vartheta}_2(t,\delta)C_2}{\beta^2}$$

$$\theta_{\mathrm{b}}^2(t,\delta) = \widetilde{\mathcal{O}}\left(L\right) = 32L + 6L_{\mathrm{p}}L\left(1+\sqrt{\log^+(C_1t/\beta)}\right)$$

where $|\mathcal{C}'_\sigma| = \mathcal{O}\left(1/\sigma^{d_2}\right)$ is the σ-covering number of $(\mathcal{S},\rho_{\mathcal{S}})$.

Proof. We have

$$\begin{aligned}
&\left|(\widehat{p}_h^t - p_h)f(s,a)\right| \\
&= \left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)\left(f(s_{h+1}^i) - p_hf(s,a)\right) - \frac{\beta p_hf(s,a)}{\mathbf{C}_h^t(s,a)}\right| \\
&\le \underbrace{\left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)\left(f(s_{h+1}^i) - \int_{\mathcal{S}}f(y)\mathrm{d}P_h(y|s_h^i,a_h^i)\right)\right|}_{①} \\
&\quad + \underbrace{\left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)\left(\int_{\mathcal{S}}f(y)\mathrm{d}p_h(y|s_h^i,a_h^i) - \int_{\mathcal{S}}f(y)\mathrm{d}p_h(y|s,a)\right)\right|}_{②} + \frac{2\beta H}{\mathbf{C}_h^t(s,a)}.
\end{aligned}$$

Bounding ② (spatial bias term) As in the proof of Lemma C.4, we can show that

$$\begin{aligned}
② &= \left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)\left(\int_{\mathcal{S}}f(y)\mathrm{d}p_h(y|s_h^i,a_h^i) - \int_{\mathcal{S}}f(y)\mathrm{d}p_h(y|s,a)\right)\right| \\
&\le 4\sigma L_{\mathrm{p}}L\left(1+\sqrt{\log^+(C_1t/\beta)}\right).
\end{aligned}$$

Bounding the martingale term (①) with a Bernstein-type inequality Notice that $(s,a) \mapsto \int_{\mathcal{S}} f(y)\mathrm{d}p_h(y|s,a)$ is bounded by $2H$ and

$$\mathbf{E}\left[f(s_{h+1}^i)|\mathcal{F}_h^i\right] = \int_{\mathcal{S}} f(y)\mathrm{d}p_h(y|s_h^i,a_h^i).$$

The conditional variance of $f(s_{h+1}^i)$ is bounded as follows

$$\begin{aligned}\mathbf{V}\left[f(s_{h+1}^i)|\mathcal{F}_h^i\right] &= \mathbf{E}\left[f(s_{h+1}^i)^2|\mathcal{F}_h^i\right] - \left(\int_{\mathcal{S}} f(y)\mathrm{d}p_h(y|s_h^i,a_h^i)\right)^2 \\ &\leq 2H\mathbf{E}\left[\left|f(s_{h+1}^i)\right||\mathcal{F}_h^i\right] \\ &= 2H\int_{\mathcal{S}} |f(y)|\,\mathrm{d}p_h(y|s_h^i,a_h^i)\end{aligned}$$

which we use to bound its weighted average

$$\begin{aligned}&\frac{1}{\mathbf{C}_h^t(s,a)}\sum_{i=1}^{t-1} w_h^{t,i}(s,a)^2\mathbf{V}\left[f(s_{h+1}^i)|\mathcal{F}_h^i\right] \\ &\leq \frac{1}{\mathbf{C}_h^t(s,a)}\sum_{i=1}^{t-1} w_h^{t,i}(s,a)\mathbf{V}\left[f(s_{h+1}^i)|\mathcal{F}_h^i\right] \\ &\leq \frac{2H}{\mathbf{C}_h^t(s,a)}\sum_{i=1}^{t-1} w_h^{t,i}(s,a)\int_{\mathcal{S}} |f(y)|\,\mathrm{d}p_h(y|s_h^i,a_h^i) \\ &= \frac{2H}{\mathbf{C}_h^t(s,a)}\sum_{i=1}^{t-1} w_h^{t,i}(s,a)p_h\left|f\right|(s,a) + \frac{2H}{\mathbf{C}_h^t(s,a)}\sum_{i=1}^{t-1} w_h^{t,i}(s,a)\left(p_h\left|f\right|(s_h^i,a_h^i) - p_h\left|f\right|(s,a)\right) \\ &\leq 2H\left(p_h\left|f\right|(s,a) - \frac{\beta p_h\left|f\right|(s,a)}{\mathbf{C}_h^t(s,a)}\right) + \frac{4HL_{\mathrm{p}}L}{\mathbf{C}_h^t(s,a)}\sum_{i=1}^{t-1} w_h^{t,i}(s,a)\rho\left[(s_h^i,a_h^i),(s,a)\right] \\ &\leq 2Hp_h\left|f\right|(s,a) + 8HL_{\mathrm{p}}L\sigma\left(1+\sqrt{\log^+(C_1t/\beta)}\right)\end{aligned}$$

where, in the last inequality, we used Lemma C.20.

Let $\triangle(t,\delta) = \log\left(4e(2t+1)/\delta\right)$. Let $Y_i(f) = f(s_{h+1}^i) - p_h f(s_h^i,a_h^i)$. By Lemma C.3, we have

$$① = \left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)Y_i(f)\right| \leq \sqrt{2\triangle(t,\delta)\frac{\sum_{i=1}^{t-1} w_h^{t,i}(s,a)^2\mathbf{V}\left[f(s_{h+1}^i)|\mathcal{F}_h^i\right]}{\mathbf{C}_h^t(s,a)^2}} + \frac{10H\triangle(t,\delta)}{\mathbf{C}_h^t(s,a)}$$

with probability at least $1-\delta$, since, for a fixed f, $(Y_i(f))_i$ is a martingale difference sequence with respect to $(\mathcal{F}_h^i)_i$. Using the fact that $\sqrt{uv} \leq (u+v)/2$ for all $u,v>0$,

$$\left|\sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)Y_i(f)\right| \leq \frac{4H^2\triangle(t,\delta)}{\mathbf{C}_h^t(s,a)} + \frac{1}{4H^2}\frac{\sum_{i=1}^{t-1} w_h^{t,i}(s,a)^2\mathbf{V}\left[f(s_{h+1}^i)|\mathcal{F}_h^i\right]}{\mathbf{C}_h^t(s,a)} + \frac{10H\triangle(t,\delta)}{\mathbf{C}_h^t(s,a)}$$

$$\leq \frac{1}{H}\int_{\mathcal{S}} |f(y)|\,\mathrm{d}p_h(y|s,a) + \frac{(4H^2+10H)\triangle(t,\delta)}{\mathbf{C}_h^t(s,a)} + \frac{2L_{\mathrm{p}}L\sigma}{H}\left(1+\sqrt{\log^+(C_1t/\beta)}\right)$$

with probability $1-\delta$.

Covering of $\mathcal{S}\times\mathcal{A}$ As a consequence of Assumption 4.2, the function

$$(s,a) \mapsto \frac{1}{H} p_h\,|f(y)|\,(s,a)$$

is $2L_{\mathrm{p}}L$-Lipschitz . Also, the functions

$$(s,a) \mapsto \left|\sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a) Y_i(f)\right| \quad \text{and} \quad (s,a) \mapsto \frac{1}{\mathbf{C}_h^t(s,a)}$$

are $4HC_2t/(\beta\sigma)$-Lipschitz and $C_2t/(\beta^2\sigma)$, respectively, by Lemma C.21. Consequently, a union bound over a $(\sigma^{2+d_2}/(H^2T))$-covering of $(\mathcal{S}\times\mathcal{A},\rho)$ and over $(t,h)\in[T]\times[H]$ gives us

$$\begin{aligned}\left|\sum_{i=1}^{t-1} \widetilde{w}_h^{t,i}(s,a) Y_i(f)\right| \leq{}& \frac{1}{H} p_h\,|f(y)|\,(s,a) + \frac{(4H^2+10H)\triangle(t,\delta)}{\mathbf{C}_h^t(s,a)} \\ &+ \frac{2L_{\mathrm{p}}L\sigma}{H}\left(1+\sqrt{\log^+(C_1t/\beta)}\right) \\ &+ \left(2L_{\mathrm{p}}L + \frac{4HC_2t}{\beta\sigma} + \frac{(4H^2+10H)\triangle(t,\delta)C_2t}{\beta^2\sigma}\right)\frac{\sigma^{2+d_2}}{H^2T}\end{aligned}$$

for all (s,a,t,h) with probability at least $1-\delta HT\mathcal{N}\left(\frac{\sigma^{2+d_2}}{H^2T}, \mathcal{S}\times\mathcal{A}, \rho\right)$.

Covering of $\mathcal{L}(2L,2H)$ The bounds for ① and ② give us

$$\begin{aligned}\left|(\widehat{p}_h^t - p_h)f(s,a)\right| \leq{}& \frac{1}{H} p_h\,|f(y)|\,(s,a) + \frac{(4H^2+10H)\triangle(t,\delta)}{\mathbf{C}_h^t(s,a)} \\ &+ \left(2L_{\mathrm{p}}L + \frac{4HC_2t}{\beta\sigma} + \frac{(4H^2+10H)\triangle(t,\delta)C_2t}{\beta^2\sigma}\right)\frac{\sigma^{2+d_2}}{H^2T} \\ &+ 6\sigma L_{\mathrm{p}}L\left(1+\sqrt{\log^+(C_1t/\beta)}\right) + \frac{2\beta H}{\mathbf{C}_h^t(s,a)}\,.\end{aligned}$$

The $8L\sigma$-covering number of $\mathcal{L}(2L,2H)$ with respect to the infinity norm is bounded by $(2H/(L\sigma))^{\mathcal{N}(\sigma,\mathcal{S},\rho_{\mathcal{S}})}$, by Lemma 5 of [Dom+21d]. The functions $f\mapsto|(p_h-\widehat{p}_h^t)f(s,a)|$ and $f\mapsto\frac{1}{H}\int_{\mathcal{S}}|f(y)|\,\mathrm{d}p_h(y|s,a)$ are 2-Lipschitz with respect to $\|\cdot\|_\infty$. Consequently, with probability at least

$$1-\delta HT\mathcal{N}\left(\frac{\sigma^{2+d_2}}{H^2T}, \mathcal{S}\times\mathcal{A}, \rho\right)\left(\frac{2H}{L\sigma}\right)^{\mathcal{N}(\sigma,\mathcal{S},\rho_{\mathcal{S}})},$$

for all $\mathcal{L}(2L, 2H)$ and for all (s, a, t, h), we have

$$\begin{aligned}\left|(\widehat{p}_h^t - p_h)f(s,a)\right| \leq \frac{1}{H} p_h \left|f(y)\right|(s,a) + \frac{(4H^2+10H)\triangle(t,\delta)}{\mathbf{C}_h^t(s,a)} \\ + \left(2L_{\mathrm{p}}L + \frac{4HC_2t}{\beta\sigma} + \frac{(4H^2+10H)\triangle(t,\delta)C_2t}{\beta^2\sigma}\right)\frac{\sigma^{2+d_2}}{H^2T} \\ + 6\sigma L_{\mathrm{p}}L\left(1+\sqrt{\log^+(C_1t/\beta)}\right) + \frac{2\beta H}{\mathbf{C}_h^t(s,a)} + 32L\sigma\end{aligned}$$

which concludes the proof. □

Lemma C.7. *Let $\mathcal{G} = \mathcal{G}_1 \cap \mathcal{G}_2 \cap \mathcal{G}_3$, where*

$$\mathcal{G}_1 := \left\{\forall(s,a,t,h),\ \left|\widehat{r}_h^t(s,a) - r_h(s,a)\right| \leq \sqrt{\frac{2\boldsymbol{\vartheta}_1^{\mathrm{r}}(t,\delta/6)}{\mathbf{C}_h^t(s,a)}} + \frac{\beta}{\mathbf{C}_h^t(s,a)} + \mathbf{b}_{\mathrm{r}}(t,\delta/6)\sigma\right\}$$

$$\mathcal{G}_2 := \left\{\forall(s,a,t,h),\ \left|(\widehat{p}_h^t - p_h)V_{h+1}^*(s,a)\right| \leq \sqrt{\frac{2H^2\boldsymbol{\vartheta}_1^{\mathrm{p}}(t,\delta/6)}{\mathbf{C}_h^t(s,a)}} + \frac{\beta H}{\mathbf{C}_h^t(s,a)} + \mathbf{b}_{\mathrm{p}}(t,\delta/6)\sigma\right\}$$

$$\begin{aligned}\mathcal{G}_3 := \Bigg\{\forall(s,a,t,h,f),\ \left|(\widehat{p}_h^t - p_h)f(s,a)\right| \leq \frac{1}{H}p_h\left|f(y)\right|(s,a) + \frac{14H^2C_2\boldsymbol{\vartheta}_2(t,\delta/6) + 2\beta H}{\mathbf{C}_h^t(s,a)} \\ + \theta_{\mathrm{b}}^1(t,\delta/6)\sigma^{1+d_2} + \theta_{\mathrm{b}}^2(t,\delta/6)\sigma\Bigg\}\end{aligned}$$

for $(s,a,t,h) \in \mathcal{S}\times\mathcal{A}\times[T]\times[H]$ and $f \in \mathcal{L}(2L, 2H)$, and where

$$\boldsymbol{\vartheta}_1^{\mathrm{p}}(t,\delta) = \widetilde{\mathcal{O}}(d_1),\quad \mathbf{b}_{\mathrm{p}}(t,\delta) = \widetilde{\mathcal{O}}\left(L+\sqrt{d_1}\right),\quad \boldsymbol{\vartheta}_1^{\mathrm{r}}(t,\delta) = \widetilde{\mathcal{O}}(d_1),\quad \mathbf{b}_{\mathrm{r}}(t,\delta) = \widetilde{\mathcal{O}}\left(L+\sqrt{d_1}\right)$$
$$\boldsymbol{\vartheta}_2(t,\delta) = \widetilde{\mathcal{O}}\left(|\mathcal{C}'_\sigma| + d_1d_2\right),\quad \theta_{\mathrm{b}}^1(t,\delta) = \widetilde{\mathcal{O}}\left(|\mathcal{C}'_\sigma| + d_1d_2 + L\sigma\right),\quad \theta_{\mathrm{b}}^2(t,\delta) = \widetilde{\mathcal{O}}(L)$$

are defined in Lemmas C.4, C.5, and C.6, respectively. Then,

$$\mathbf{P}[\mathcal{G}] \geq 1 - \delta/2.$$

Proof. Immediate consequence of lemmas C.4, C.5, and C.6. □

C.2.3 Regret bound in terms of the sum of exploration bonuses

Lemma C.8 (Optimism)**.** *On the event $\mathcal{G}$, we have*

$$\forall (s,a,t,h), Q_h^t(s,a) \geq Q_h^*(s,a).$$

Proof. We proceed by induction. When $h = H+1$, we have $Q_{H+1}^t = Q_{H+1}^* = 0$, by definition. Assuming that $Q_{h+1}^t(s,a) \geq Q_{h+1}^*(s,a)$ for all (s,a), we have $V_{h+1}^t(s) \geq V_{h+1}^*(s)$ for all s. Then, by the definition of the bonuses and the event $\mathcal{G}$, we have for all (s,a)

$$\begin{aligned}\widetilde{Q}_h^t(s,a) - Q_h^*(s,a) &= \underbrace{\widehat{r}_h^t(s,a) - r_h(s,a) + (\widehat{p}_h^t - p_h)V_{h+1}^*(s,a) + \mathtt{b}_h^t(s,a)}_{\geq 0 \text{ on } \mathcal{G}} \\ &\quad + \underbrace{\widehat{p}_h(V_{h+1}^t - V_{h+1}^*)}_{\geq 0 \text{ by induction hypothesis}} \\ &\geq 0.\end{aligned}$$

In particular, $\widetilde{Q}_h^t(s_h^i, a_h^i) \geq Q_h^*(s_h^i, a_h^i)$ for all $i \in [t-1]$, which gives us

$$\widetilde{Q}_h^t(s_h^i, a_h^i) + L\rho\left[(s,a),(s_h^i,a_h^i)\right] \geq Q_h^*(s_h^i, a_h^i) + L\rho\left[(s,a),(s_h^i,a_h^i)\right] \geq Q_h^*(s,a)$$

for all $i \in [t-1]$, since Q_h^* is L-Lipschitz. It follows from the definition of the interpolation function in Equation (4.6) that $Q_h^t(s,a) \geq Q_h^*(s,a)$ for all (s,a), which implies that, for all s, $V_h^t(s) \geq V_h^*(s)$ on $\mathcal{G}$. □

Corollary C.9. *Let $\delta_h^t := V_h^t(s_h^t) - V_h^{\pi^t}(s_h^t)$. Then, the regret of* `Kernel-UCBVI` *satisfies $\mathcal{R}_T \leq \sum_{t=1}^T \delta_1^t$ on the event $\mathcal{G}$.*

Proof. As a consequence of Lemma C.8, we have

$$\begin{aligned}\mathcal{R}_T &= \sum_{t=1}^T \left(V_1^*(s_1^t) - V_1^{\pi^t}(s_1^t)\right) = \sum_{t=1}^T \left(\max_a Q_1^*(s_1^t,a) - V_1^{\pi^t}(s_1^t)\right) \\ &\leq \sum_{t=1}^T \left(\min\left[H-h+1, \max_a Q_1^t(s_1^t,a)\right] - V_1^{\pi^t}(s_1^t)\right) = \sum_{t=1}^T \left(V_1^t(s_1^t,a) - V_1^{\pi^t}(s_1^t)\right).\end{aligned}$$

□

Definition C.10. *For any (t,h), let $(\widetilde{s}_h^t, \widetilde{a}_h^t)$ be the state-action pair that is the closest to (s_h^t, a_h^t) among the transitions observed before episode t , that is*

$$(\widetilde{s}_h^t, \widetilde{a}_h^t) := \operatorname*{argmin}_{(s_h^i, a_h^i): i<t} \rho\left[(s_h^t, a_h^t), (s_h^i, a_h^i)\right].$$

Furthermore, we define $\rho_h^t := \rho\left[(s_h^t, a_h^t), (\widetilde{s}_h^t, \widetilde{a}_h^t)\right]$.

Lemma C.11. *On the event* $\mathcal{G}$*, the regret of* `Kernel-UCBVI` *is bounded by*

$$\mathcal{R}_T \lesssim \sum_{t=1}^{T}\sum_{h=1}^{H}\left(\frac{H}{\sqrt{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}} + \frac{H^2\,|\mathcal{C}'_\sigma|}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}\right)\mathbb{1}\left\{\rho_h^t \le 2\sigma\right\} + H^2\,|\mathcal{C}_\sigma| + \sum_{t=1}^{T}\sum_{h=1}^{H}\left(1+\frac{1}{H}\right)^h \widetilde{\xi}_{h+1}^t + LHT\sigma$$

where $|\mathcal{C}_\sigma|$ *is the* σ*-covering number of* $(\mathcal{S}\times\mathcal{A}, \rho)$*,* $|\mathcal{C}'_\sigma|$ *is the* σ*-covering number of* $(\mathcal{S}, \rho_{\mathcal{S}})$ *and* $(\widetilde{\xi}_{h+1}^t)_{t,h}$ *is a martingale difference sequence with respect to* $(\mathcal{F}_h^t)_{t,h}$ *bounded by* $4H$*.*

Proof. **Regret decomposition** On $\mathcal{G}$, we upper bound δ_h^t using the following decomposition:

$$\begin{aligned}
\delta_h^t &= V_h^t(s_h^t) - V_h^{\pi^t}(s_h^t)\\
&\le Q_h^t(s_h^t, a_h^t) - Q_h^{\pi^t}(s_h^t, a_h^t)\\
&\le Q_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) - Q_h^{\pi^t}(s_h^t, a_h^t) + L\rho\left[(\widetilde{s}_h^t, \widetilde{a}_h^t), (s_h^t, a_h^t)\right], \quad \text{since } Q_h^t \text{ is } L\text{-Lipschitz}\\
&\le \widetilde{Q}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) - Q_h^{\pi_t}(s_h^t, a_h^t) + L\rho\left[(\widetilde{s}_h^t, \widetilde{a}_h^t), (s_h^t, a_h^t)\right], \quad \text{since } Q_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) \le \widetilde{Q}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)\\
&= \widehat{r}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) - r_h(s_h^t, a_h^t) + \widehat{p}_h^t V_{h+1}^t(\widetilde{s}_h^t, \widetilde{a}_h^t) - p_h V_{h+1}^{\pi^t}(s_h^t, a_h^t) + \mathtt{b}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) + L\rho_h^t\\
&= \underbrace{\widehat{r}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) - r_h(s_h^t, a_h^t)}_{(\mathbf{A})} + \underbrace{\left[\widehat{p}_h^t - p_h\right] V_{h+1}^*(\widetilde{s}_h^t, \widetilde{a}_h^t)}_{(\mathbf{B})} + \underbrace{\left[\widehat{p}_h^t - p_h\right]\left(V_{h+1}^t - V_{h+1}^*\right)(\widetilde{s}_h^t, \widetilde{a}_h^t)}_{(\mathbf{C})}\\
&+ \underbrace{p_h V_{h+1}^t(\widetilde{s}_h^t, \widetilde{a}_h^t) - p_h V_{h+1}^{\pi^t}(s_h^t, a_h^t)}_{(\mathbf{D})} + \mathtt{b}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) + 2L\rho\left[(\widetilde{s}_h^t, \widetilde{a}_h^t), (s_h^t, a_h^t)\right].
\end{aligned}$$

Now, we bound each term $(\mathbf{A})$-$(\mathbf{D})$ separately, using the definition of $\mathcal{G}$ (Lemma C.7).

Term $(\mathbf{A})$:

$$\begin{aligned}
(\mathbf{A}) &= \widehat{r}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) - r_h(\widetilde{s}_h^t, \widetilde{a}_h^t) + r_h(\widetilde{s}_h^t, \widetilde{a}_h^t) - r_h(s_h^t, a_h^t)\\
&\le \widehat{r}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) - r_h(\widetilde{s}_h^t, \widetilde{a}_h^t) + L_{\mathrm{r}}\rho\left[(\widetilde{s}_h^t, \widetilde{a}_h^t), (s_h^t, a_h^t)\right]\\
&\le {}^{r}\mathtt{b}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t) + L_{\mathrm{r}}\rho\left[(\widetilde{s}_h^t, \widetilde{a}_h^t), (s_h^t, a_h^t)\right].
\end{aligned}$$

Term $(\mathbf{B})$:

$$(\mathbf{B}) = \left[\widehat{p}_h^t - p_h\right] V_{h+1}^*(\widetilde{s}_h^t, \widetilde{a}_h^t) \le {}^{p}\mathtt{b}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t).$$

Term (**C**): Since $V^t_{h+1} \geq V^*_{h+1}$ on $\mathcal{G}$ and $V^*_{h+1} \geq V^{\boldsymbol{\pi}}_{h+1}$:

$$\begin{aligned}
(\mathbf{C}) &= \left[\widehat{p}^t_h - p_h\right]\left(V^t_{h+1} - V^*_{h+1}\right)(\widetilde{s}^t_h, \widetilde{a}^t_h) \\
&\lesssim \frac{1}{H} p_h \left(V^t_{h+1} - V^*_{h+1}\right)(\widetilde{s}^t_h, \widetilde{a}^t_h) + \frac{H^2 \left|\mathcal{C}'_\sigma\right|}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)} + L\sigma \\
&\leq \frac{1}{H} p_h \left(V^t_{h+1} - V^*_{h+1}\right)(s^t_h, a^t_h) + 2L_{\mathrm{p}} L\rho\left[(\widetilde{s}^t_h, \widetilde{a}^t_h), (s^t_h, a^t_h)\right] + \frac{H^2 \left|\mathcal{C}'_\sigma\right|}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)} + L\sigma \\
&\lesssim \frac{1}{H} p_h \left(V^t_{h+1} - V^{\boldsymbol{\pi}^t}_{h+1}\right)(s^t_h, a^t_h) + L\rho\left[(\widetilde{s}^t_h, \widetilde{a}^t_h), (s^t_h, a^t_h)\right] + \frac{H^2 \left|\mathcal{C}'_\sigma\right|}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)} + L\sigma \\
&= \frac{1}{H}\left(\delta^t_{h+1} + \xi^t_{h+1}\right) + L\rho\left[(\widetilde{s}^t_h, \widetilde{a}^t_h), (s^t_h, a^t_h)\right] + \frac{H^2 \left|\mathcal{C}'_\sigma\right|}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)} + L\sigma
\end{aligned}$$

where

$$\xi^t_{h+1} = p_h\left(V^t_{h+1} - V^{\boldsymbol{\pi}^t}_{h+1}\right)(s^t_h, a^t_h) - \delta^t_{h+1}$$

is a martingale difference sequence with respect to $(\mathcal{F}^t_h)_{t,h}$ bounded by $4H$.

Term (**D**): We have

$$\begin{aligned}
(\mathbf{D}) &= p_h V^t_{h+1}(\widetilde{s}^t_h, \widetilde{a}^t_h) - p_h V^{\boldsymbol{\pi}^t}_{h+1}(s^t_h, a^t_h) \\
&\leq L_{\mathrm{p}} L\rho\left[(\widetilde{s}^t_h, \widetilde{a}^t_h), (s^t_h, a^t_h)\right] + p_h V^t_{h+1}(s^t_h, a^t_h) - p_h V^{\boldsymbol{\pi}^t}_{h+1}(s^t_h, a^t_h) \\
&= \delta^t_{h+1} + \xi^t_{h+1} + L_{\mathrm{p}} L\rho\left[(\widetilde{s}^t_h, \widetilde{a}^t_h), (s^t_h, a^t_h)\right].
\end{aligned}$$

Putting together the bounds above, we obtain

$$\delta^t_h \lesssim \left(1 + \frac{1}{H}\right)\left(\delta^t_{h+1} + \xi^t_{h+1}\right) + L\rho\left[(\widetilde{s}^t_h, \widetilde{a}^t_h), (s^t_h, a^t_h)\right] + \sqrt{\frac{H^2}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)}} + \frac{H^2 \left|\mathcal{C}'_\sigma\right|}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)} + L\sigma\,,$$

where the constant in front of δ^t_{h+1} is exact (not hidden by $\lesssim$).

Now, consider the event $E^t_h := \{\rho\left[(\widetilde{s}^t_h, \widetilde{a}^t_h), (s^t_h, a^t_h)\right] \leq 2\sigma\}$ and let $\overline{E}^t_h$ be its complement. Using the fact that $\delta^t_{h+1} \geq 0$ on $\mathcal{G}$, the inequality above implies

$$\begin{aligned}
&\mathbb{1}\left\{E^t_h\right\}\delta^t_h \\
&\lesssim \mathbb{1}\left\{E^t_h\right\}\left(1 + \frac{1}{H}\right)\left(\delta^t_{h+1} + \xi^t_{h+1}\right) + L\sigma + \mathbb{1}\left\{E^t_h\right\}\left(\sqrt{\frac{H^2}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)}} + \frac{H^2 \left|\mathcal{C}'_\sigma\right|}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)}\right) \\
&\lesssim \left(1 + \frac{1}{H}\right)\left(\delta^t_{h+1} + \mathbb{1}\left\{E^t_h\right\}\xi^t_{h+1}\right) + L\sigma + \mathbb{1}\left\{E^t_h\right\}\left(\sqrt{\frac{H^2}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)}} + \frac{H^2 \left|\mathcal{C}'_\sigma\right|}{\mathbf{C}^t_h(\widetilde{s}^t_h, \widetilde{a}^t_h)}\right) \qquad \text{(C.1)}
\end{aligned}$$

Now, using the fact that $\delta_h^t \leq H$, we obtain

$$\begin{aligned}\delta_h^t &= \mathbb{1}\left\{E_h^t\right\}\delta_h^t + \mathbb{1}\left\{\overline{E}_h^t\right\}\delta_h^t \\ &\leq \mathbb{1}\left\{E_h^t\right\}\delta_h^t + H\mathbb{1}\left\{\overline{E}_h^t\right\} \\ &\lesssim H\mathbb{1}\left\{\overline{E}_h^t\right\} + \left(1+\frac{1}{H}\right)\left(\delta_{h+1}^t + \mathbb{1}\left\{E_h^t\right\}\xi_{h+1}^t\right) + L\sigma \\ &\quad + \mathbb{1}\left\{E_h^t\right\}\left(\sqrt{\frac{H^2}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}} + \frac{H^2\left|\mathcal{C}'_\sigma\right|}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}\right).\end{aligned} \tag{C.2}$$

This yields

$$\begin{aligned}\delta_1^t \lesssim \sum_{h=1}^H \mathbb{1}\left\{E_h^t\right\}\left(\sqrt{\frac{H^2}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}} + \frac{H^2\left|\mathcal{C}'_\sigma\right|}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}\right) \\ + \sum_{h=1}^H\left(1+\frac{1}{H}\right)^h \mathbb{1}\left\{E_h^t\right\}\xi_{h+1}^t + LH\sigma + H\sum_{h=1}^H \mathbb{1}\left\{\overline{E}_h^t\right\}.\end{aligned}$$

Let $\widetilde{\xi}_{h+1}^t := \mathbb{1}\left\{E_h^t\right\}\xi_{h+1}^t$. We can verify that $\widetilde{\xi}_{h+1}^t$ is a martingale difference sequence with respect to $(\mathcal{F}_h^t)_{t,h}$ bounded by $4H$.

Applying Corollary C.9, we obtain:

$$\begin{aligned}\mathcal{R}_T \leq \sum_{t=1}^T \delta_1^t \lesssim \sum_{t=1}^T\sum_{h=1}^H \mathbb{1}\left\{E_h^t\right\}\left(\sqrt{\frac{H^2}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}} + \frac{H^2\left|\mathcal{C}'_\sigma\right|}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}\right) \\ + \sum_{t=1}^T\sum_{h=1}^H\left(1+\frac{1}{H}\right)^h \widetilde{\xi}_{h+1}^t + LHT\sigma + H\sum_{t=1}^T\sum_{h=1}^H \mathbb{1}\left\{\overline{E}_h^t\right\}.\end{aligned}$$

Finally, we bound the sum

$$H\sum_{t=1}^T\sum_{h=1}^H \mathbb{1}\left\{\overline{E}_h^t\right\} = H\sum_{h=1}^H\sum_{t=1}^T \mathbb{1}\left\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right] > 2\sigma\right\} \leq H^2\left|\mathcal{C}_\sigma\right|$$

since, for each h, the number of episodes where the event $\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right] > 2\sigma\}$ occurs is bounded by $|\mathcal{C}_\sigma|$. Recalling the definition $E_h^t := \{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right] \leq 2\sigma\}$, this concludes the proof. □

C.2.4 Bounding the sum of exploration bonuses

Lemma C.12. *We have*

$$\sum_{t=1}^{T}\sum_{h=1}^{H}\frac{1}{\sqrt{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}}\mathbb{1}\left\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right]\leq 2\sigma\right\}\lesssim H\left|\mathcal{C}_\sigma\right|+H\sqrt{\left|\mathcal{C}_\sigma\right|T}$$

and

$$\sum_{t=1}^{T}\sum_{h=1}^{H}\frac{1}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}\mathbb{1}\left\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right]\leq 2\sigma\right\}\lesssim H\left|\mathcal{C}_\sigma\right|.$$

Proof. First, we will need some definitions. Let $\mathcal{C}_\sigma=\{(s_j,a_j)\in\mathcal{S}\times\mathcal{A}, j=1,\dots,|\mathcal{C}_\sigma|\}$ be a σ-covering of $(\mathcal{S}\times\mathcal{A},\rho)$. We define a partition $\{B_j\}_{j=1}^{|\mathcal{C}_\sigma|}$ of $\mathcal{S}\times\mathcal{A}$ as follows:

$$B_j=\left\{(s,a)\in\mathcal{S}\times\mathcal{A}:(s_j,a_j)=\operatorname*{argmin}_{(s_i,a_i)\in\mathcal{C}_\sigma}\rho\left[(s,a),(s_i,a_i)\right]\right\}$$

where ties in the argmin are broken arbitrarily.

We define the number of visits to each set B_j as $\mathbf{N}_h^t(B_j):=\sum_{i=1}^{t-1}\mathbb{1}\left\{(s_h^i,a_h^i)\in B_j\right\}$.

Now, assume that $(s_h^t,a_h^t)\in B_j$. If, in addition, $\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right]\leq 2\sigma$, we obtain

$$\begin{aligned}\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)&=\beta+\sum_{i=1}^{t-1}\Gamma((\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^i,a_h^i))=\beta+\sum_{i=1}^{t-1}\overline{\Gamma}\left(\frac{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^i,a_h^i)\right]}{\sigma}\right)\\&\geq\beta+\sum_{i=1}^{t-1}\overline{\Gamma}\left(\frac{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^i,a_h^i)\right]}{\sigma}\right)\mathbb{1}\left\{(s_h^i,a_h^i)\in B_j\right\}\\&\geq\beta+\overline{\Gamma}(4)\sum_{i=1}^{t-1}\mathbb{1}\left\{(s_h^i,a_h^i)\in B_j\right\}=\beta\left(1+\overline{\Gamma}(4)\beta^{-1}\mathbf{N}_h^t(B_j)\right)\end{aligned}$$

since, if $(s_h^i,a_h^i)\in B_j$, we have $\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^i,a_h^i)\right]\leq 4\sigma$ and we use the fact that $\overline{\Gamma}$ is non-increasing by assumption.

We are now ready to bound the sums involving $1/\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)$. We will use the fact that $\overline{\Gamma}(4)>0$ by Assumption 4.4.

Bounding the sum of $1/\sqrt{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}$

$$\sum_{t=1}^{T}\sum_{h=1}^{H}\sqrt{\frac{1}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}}\mathbb{1}\left\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right]\leq 2\sigma\right\}$$

$$
\begin{aligned}
&= \sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\sqrt{\frac{1}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}}\mathbb{1}\left\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right]\le 2\sigma\right\}\mathbb{1}\left\{(s_h^t,a_h^t)\in B_j\right\}\\
&\le \beta^{-1/2}\sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\frac{1}{\sqrt{1+\overline{\Gamma}(4)\beta^{-1}\mathbf{N}_h^t(B_j)}}\mathbb{1}\left\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right]\le 2\sigma\right\}\mathbb{1}\left\{(s_h^t,a_h^t)\in B_j\right\}\\
&\le \beta^{-1/2}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\sum_{t=1}^{T}\frac{\mathbb{1}\left\{(s_h^t,a_h^t)\in B_j\right\}}{\sqrt{1+\overline{\Gamma}(4)\beta^{-1}\mathbf{N}_h^t(B_j)}}\\
&\le \beta^{-1/2}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\left(1+\int_0^{\mathbf{N}_h^{T+1}(B_j)}\frac{\mathrm{d}z}{\sqrt{1+\overline{\Gamma}(4)\beta^{-1}z}}\right)\quad\text{by Lemma C.19}\\
&\le \beta^{-1/2}H\,|\mathcal{C}_\sigma| + \frac{2\beta^{1/2}}{\overline{\Gamma}(4)}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\sqrt{1+\overline{\Gamma}(4)\beta^{-1}\mathbf{N}_h^{T+1}(B_j)}\\
&\le \beta^{-1/2}H\,|\mathcal{C}_\sigma| + \frac{2\beta^{1/2}}{\overline{\Gamma}(4)}\sum_{h=1}^{H}\sqrt{|\mathcal{C}_\sigma|}\sqrt{|\mathcal{C}_\sigma|+\overline{\Gamma}(4)\beta^{-1}T}\quad\text{by Cauchy-Schwarz inequality}\\
&\le H\left(\beta^{-1/2}+\frac{2\beta^{1/2}}{\overline{\Gamma}(4)}\right)|\mathcal{C}_\sigma| + \frac{2H}{\overline{\Gamma}(4)}\sqrt{\overline{\Gamma}(4)\,|\mathcal{C}_\sigma|\,T}\lesssim H\,|\mathcal{C}_\sigma| + H\sqrt{|\mathcal{C}_\sigma|\,T}\,.
\end{aligned}
$$

Bounding the sum of $1/\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)$

$$
\begin{aligned}
&\sum_{t=1}^{T}\sum_{h=1}^{H}\frac{1}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}\mathbb{1}\left\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right]\le 2\sigma\right\}\\
&= \sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\frac{1}{\mathbf{C}_h^t(\widetilde{s}_h^t,\widetilde{a}_h^t)}\mathbb{1}\left\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right]\le 2\sigma\right\}\mathbb{1}\left\{(s_h^t,a_h^t)\in B_j\right\}\\
&\le \beta^{-1}\sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\frac{1}{1+\overline{\Gamma}(4)\beta^{-1}\mathbf{N}_h^t(B_j)}\mathbb{1}\left\{\rho\left[(\widetilde{s}_h^t,\widetilde{a}_h^t),(s_h^t,a_h^t)\right]\le 2\sigma\right\}\mathbb{1}\left\{(s_h^t,a_h^t)\in B_j\right\}\\
&\le \beta^{-1}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\sum_{t=1}^{T}\frac{\mathbb{1}\left\{(s_h^t,a_h^t)\in B_j\right\}}{1+\overline{\Gamma}(4)\beta^{-1}\mathbf{N}_h^t(B_j)}\\
&\le \beta^{-1}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\left(1+\int_0^{\mathbf{N}_h^{T+1}(B_j)}\frac{\mathrm{d}z}{1+\overline{\Gamma}(4)\beta^{-1}z}\right)\quad\text{by Lemma C.19}\\
&\le \beta^{-1}H\,|\mathcal{C}_\sigma| + \frac{1}{\overline{\Gamma}(4)}\sum_{h=1}^{H}\sum_{j=1}^{|\mathcal{C}_\sigma|}\log\left(1+\overline{\Gamma}(4)\beta^{-1}\mathbf{N}_h^{t+1}(B_j)\right)\\
&\le \beta^{-1}H\,|\mathcal{C}_\sigma| + \frac{1}{\overline{\Gamma}(4)}\sum_{h=1}^{H}|\mathcal{C}_\sigma|\log\left(\frac{\sum_{j=1}^{|\mathcal{C}_\sigma|}\left(1+\overline{\Gamma}(4)\beta^{-1}\mathbf{N}_h^{T+1}(B_j)\right)}{|\mathcal{C}_\sigma|}\right)\quad\text{by Jensen's inequality}\\
&\le \beta^{-1}H\,|\mathcal{C}_\sigma| + \frac{1}{\overline{\Gamma}(4)}H\,|\mathcal{C}_\sigma|\log\left(1+\frac{1+\overline{\Gamma}(4)\beta^{-1}T}{|\mathcal{C}_\sigma|}\right)\lesssim H\,|\mathcal{C}_\sigma|\,.
\end{aligned}
$$

□

C.2.5 Final regret bound

We are now ready to conclude the proof of Theorem 4.7. By Hoeffding-Azuma's inequality, the sum $\sum_{t=1}^{T}\sum_{h=1}^{H}(1+1/H)^H\widetilde{\xi}_{h+1}^t$ is bounded by $\left(\sqrt{8e^2H^2\log(2/\delta)}\right)\sqrt{HT}$ on an event $\mathcal{G}'$ such that $\mathbf{P}\left[\mathcal{G}'\right] \geq 1-\delta/2$.

Hence, by lemmas C.11 and C.12, we obtain

$$\mathcal{R}_T \lesssim H^2\sqrt{|\mathcal{C}_\sigma|\,T} + H^3\,|\mathcal{C}_\sigma|\,|\mathcal{C}'_\sigma| + H^{3/2}\sqrt{T} + LHT\sigma + H^2\,|\mathcal{C}_\sigma|\,,$$

on the event $\mathcal{G}\cap\mathcal{G}'$, that satisfies $\mathbf{P}\left[\mathcal{G}\cap\mathcal{G}'\right] \geq 1-\delta$.

C.3 Proof Sketch for Theorem 4.12: Regret of KeRNS

We now outline the proof of Theorem 4.12 assuming, for simplicity, that the rewards are known. The full proof is given in our paper [Dom+21c].

Bias due to non-stationarity To bound the bias, we introduce an average MDP with transitions $\overline{p}_h^t$:

$$\overline{p}_h^t(y|s,a) := \sum_{i=1}^{t-1}\widetilde{w}_h^{t,i}(s,a)p_h^i(y|s,a) + \frac{\beta\,p_h^t(y|s,a)}{\mathbf{C}_h^t(s,a)},$$

where $(p_h^i)_{i,h}$ are the true transitions at time (i,h). We prove that, for any L-Lipschitz function f bounded by H:

$$\left|\left(p_h^t - \overline{p}_h^t\right)f(s,a)\right| \leq \mathbf{bias_p}(t,h),$$

where the term $\mathbf{bias_p}(t,h)$ is defined as

$$\mathbf{bias_p}(t,h) := L\sum_{i=1\vee(t-W)}^{t-1}\sup_{s,a}\mathbb{W}_1\left(p_h^i(\cdot|s,a), p_h^{i+1}(\cdot|s,a)\right) + \frac{2C_3H}{\beta}\frac{\lambda^W}{1-\lambda}\,.$$

Concentration Using concentration inequalities for weighted sums, we prove that $\widehat{p}_h^t$ is close to the average transition $\overline{p}_h^t$ using Hoeffding- and Bernstein-type inequalities, and define an event $\mathcal{G}$ where our confidence sets hold, such that $\mathbf{P}\left[\mathcal{G}\right] \geq 1-\delta/2$. For instance, as for KernelUCBVI, we have

$$\left|(\widehat{p}_h^t - \overline{p}_h^t)V_{t,h+1}^*(s,a)\right| \lesssim \sqrt{\frac{H^2}{\mathbf{C}_h^t(s,a)}} + \frac{\beta H}{\mathbf{C}_h^t(s,a)} + L\sigma\,,$$

which explains the form of the exploration bonuses.

Upper bound on the true value function On the event $\mathcal{G}$, we show that:

$$Q_h^t(s,a) + \sum_{h'=h}^{H} \mathbf{bias}(t,h) \geq Q_{t,h}^*(s,a)$$

where the term $\mathbf{bias}(t,h)$ is the sum of $\mathbf{bias_p}(t,h)$ defined above, and a similar term representing the bias in the reward estimation.

Regret bounds Let $(\widetilde{s}_h^t, \widetilde{a}_h^t)$ be the state-action pair among the previously visited ones that is the closest to (s_h^t, a_h^t):

$$(\widetilde{s}_h^t, \widetilde{a}_h^t) := \underset{(s_h^i, a_h^i): i<t}{\operatorname{argmin}} \rho\left[(s_h^t, a_h^t), (s_h^i, a_h^i)\right].$$

We show that:

$$H \sum_{t=1}^{T} \sum_{h=1}^{H} \mathbb{1}\left\{\rho\left[(s_h^t, a_h^t), (\widetilde{s}_h^t, \widetilde{a}_h^t)\right] > 2\sigma\right\} \leq H^2 \left|\mathcal{C}_\sigma\right|.$$

Thus, to simplify the outline, for all (t,h), we assume that $\rho\left[(s_h^t, a_h^t), (\widetilde{s}_h^t, \widetilde{a}_h^t)\right] \leq 2\sigma$ and add $H^2 \left|\mathcal{C}_\sigma\right|$ to the final regret bound. On the event $\mathcal{G}$, we prove that the regret of KeRNS is bounded by:

$$\mathcal{R}_T^{\mathrm{dyn}} \lesssim \sum_{t=1}^{T} \sum_{h=1}^{H} \left(\frac{H}{\sqrt{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}} + \frac{\beta H}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)} \right) + \sum_{t=1}^{T} \sum_{h=1}^{H} \mathbf{bias}(t,h) + LHT\sigma + H^2 \left|\mathcal{C}_\sigma\right|$$

where we omitted factors involving $|\mathcal{C}_\sigma|$ and $|\mathcal{C}'_\sigma|$ (which depend on the type of bound considered, $\mathcal{R}_T^1$ or $\mathcal{R}_T^2$), and martingale terms (which are bounded by $\approx H^{3/2}\sqrt{T}$ with probability at least $1 - \delta/2$).

Using the properties of the kernel Γ (Assumption 4.9), we prove that:

$$\sum_{t=1}^{T} \sum_{h=1}^{H} \frac{1}{\sqrt{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}} \lesssim HT \log \frac{1}{\lambda} \left(\left|\mathcal{C}_\sigma\right| + \sqrt{\frac{\left|\mathcal{C}_\sigma\right|}{\log(1/\lambda)}} \right)$$

$$\sum_{t=1}^{T} \sum_{h=1}^{H} \frac{1}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)} \lesssim H \left|\mathcal{C}_\sigma\right| T \log \frac{1}{\lambda}$$

Finally, we bound the sum of biases as

$$\sum_{t=1}^{T}\sum_{h=1}^{H} \mathbf{bias}(t,h) \leq 2W\left(\Delta^{\mathrm{r}} + L\Delta^{\mathrm{p}}\right) + \frac{2C_3(H+1)HT}{\beta}\frac{\lambda^W}{1-\lambda}.$$

Putting these bounds together, we conclude the proof of Theorem 4.12.

C.4 Proof of Theorem 4.14: Regret of Kernel-UCBVI+RTDP

In this Appendix, following Efroni et al. [Efr+19], we show that, by modifying Kernel-UCBVI so that we apply the optimistic Bellman operator (C.3) only once instead of doing a complete value iteration, we obtain almost the same guaranties as for Kernel-UCBVI, but with a large improvement in computational complexity: the time complexity of each episode t is reduced from $\mathcal{O}\left(t^2\right)$ to $\mathcal{O}\left(t\right)$. We begin by recalling how Kernel-UCBVI+RTDP (Algorithm 4.3) proceeds.

Assume we are at episode t at step h at state s_h^t. To compute the next action we will apply the optimistic Bellman operator to the previous value function. That is, for all $a \in \mathcal{A}$ we compute the upper bounds on the Q-value based on a kernel estimator:

$$\widetilde{Q}_h^t(s_h^t, a) = \widehat{r}_h^t(s,a) + \widehat{p}_h^t V_{h+1}^t(s,a) + \mathtt{b}_h^t(s,a). \tag{C.3}$$

The action a_h^t is then computed as

$$a_h^t = \operatorname*{argmax}_{a\in\mathcal{A}} \widetilde{Q}_h^t(s_h^t, a)\,,$$

and we define an optimistic value $\widetilde{V}_h^t(s_h^t) = \min\left(H-h+1, \widetilde{Q}_h^t(s_h^t, a_h^t)\right)$ for the value function at the state s_h^t. Then, we build an optimistic function V_h^{t+1} by interpolating the previous optimistic value function V_h^t and the value $\widetilde{V}_h^t(s_h^t)$:

$$\forall s,\ V_h^{t+1}(s) = \min\left(V_h^t(s), \widetilde{V}_h^t(s_h^t) + L\rho_{\mathcal{S}}\left(s, s_h^t\right)\right).$$

Lemma C.13 (Optimism). *On the event $\mathcal{G}$, whose probability is at least $1-\delta$, we have*

$$\forall (s,t,h),\ V_h^t(s) \geq V_h^*(s) \text{ and } V_h^t(s) \geq V_h^{t+1}(s)\,.$$

Proof. The fact that $V_h^t(s) \geq V_h^{t+1}(s)$ is immediate by the definition of V_h^t:

$$\forall s,\ V_h^{t+1}(s) := \min\left(V_h^t(s), \widetilde{V}_h^t(s_h^t) + L\rho_{\mathcal{S}}\left(s, s_h^t\right)\right) \leq V_h^t(s).$$

To show that $V_h^t(s) \geq V_h^*(s)$, we proceed by induction on t. For $t = 1$, $V_h^t(s) = H - h \geq V_h^*(s)$ for all (s, h).

Now, consider $t > 1$ and assume that $V_h^{t-1} \geq V_h^*$ for all h. As in the proof of Lemma C.8, we prove that $V_h^t \geq V_h^*$ for all h by induction on h. For $h = H+1$, $V_h^t(s) = V_h^*(s) = 0$ for all s. Now, assume that $V_{h+1}^t(s) \geq V_{h+1}^*(s)$ for all s. We have, for all (s, a),

$$\begin{aligned}
\widetilde{Q}_h^t(s,a) &= \widehat{r}_h^t(s,a) + \widehat{p}_h^t V_{h+1}^t(s,a) + \mathtt{b}_h^t(s,a) \\
&\geq \widehat{r}_h^t(s,a) + \widehat{p}_h^t V_{h+1}^*(s,a) + \mathtt{b}_h^t(s,a) \quad \text{by induction hypothesis on } h \\
&\geq r_h(s,a) + p_h V_{h+1}^*(s,a) = Q_h^*(s,a) \quad \text{on } \mathcal{G}
\end{aligned}$$

which implies that $\widetilde{V}_h^t(s_h^t) \geq V_h^*(s_h^t)$ and, consequently,

$$\begin{aligned}
&\widetilde{V}_h^t(s_h^t) + L\rho_{\mathcal{S}}\left(s, s_h^t\right) \geq V_h^*(s_h^t) + L\rho_{\mathcal{S}}\left(s, s_h^t\right) \geq V_h^*(s) \\
&\implies V_h^t(s) = \min\left(V_h^{t-1}(s), \widetilde{V}_h^t(s_h^t) + L\rho_{\mathcal{S}}\left(s, s_h^t\right)\right) \geq V_h^*(x) \text{ by induction hypothesis on } t,
\end{aligned}$$

where we used the fact that V_h^* is L-Lipschitz. □

Now, we prove that, with probability at least $1 - \delta$, the regret of `Kernel-UCBVI+RTDP` satisfies

$$\mathcal{R}_T \lesssim H^2\sqrt{|\mathcal{C}_\sigma|\, T} + H^3\,|\mathcal{C}_\sigma|\,|\mathcal{C}'_\sigma| + H^{3/2}\sqrt{T} + LHT\sigma + H^2\,|\mathcal{C}_\sigma| + H^2\,|\mathcal{C}'_\sigma|\,,$$

which implies Theorem 4.14.

On $\mathcal{G}$, we have

$$\begin{aligned}
\widetilde{\delta}_h^t &:= V_h^{t+1}(s_h^t) - V_h^{\boldsymbol{\pi}^t}(s_h^t) \leq V_h^t(s_h^t) - V_h^{\boldsymbol{\pi}^t}(s_h^t) \\
&\leq \widetilde{V}_h^t(s_h^t) - V_h^{\boldsymbol{\pi}^t}(s_h^t) \leq \widetilde{Q}_h^t(s_h^t, a_h^t) - Q_h^{\boldsymbol{\pi}^t}(s_h^t, a_h^t).
\end{aligned}$$

Recall that $\delta_h^t := V_h^t(s_h^t) - V_h^{\boldsymbol{\pi}^t}(s_h^t)$. From this point, we follow the proof of Lemma C.11 to obtain a bound on $\widetilde{\delta}_h^t$:

$$\begin{aligned}
\widetilde{\delta}_h^t &\lesssim \left(1 + \frac{1}{H}\right)\left(\delta_{h+1}^t + \xi_{h+1}^t\right) + L\rho\left[(\widetilde{s}_h^t, \widetilde{a}_h^t), (s_h^t, a_h^t)\right] + \sqrt{\frac{H^2}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}} + \frac{H^2\,|\mathcal{C}'_\sigma|}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)} + L\sigma \\
&\lesssim \left(1 + \frac{1}{H}\right)\left(\widetilde{\delta}_{h+1}^t + \left(V_{h+1}^t - V_{h+1}^{t+1}\right)(s_{h+1}^t) + \xi_{h+1}^t\right) + L\rho\left[(\widetilde{s}_h^t, \widetilde{a}_h^t), (s_h^t, a_h^t)\right] \\
&\quad + \sqrt{\frac{H^2}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}} + \frac{H^2\,|\mathcal{C}'_\sigma|}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)} + L\sigma.
\end{aligned}$$

On $\mathcal{G}$, using that $V_h^* \leq V_h^{t+1}$ and the same arguments as in equations (C.1) and (C.2) in Lemma C.11 (which can be used since $V_{h+1}^t \geq V_{h+1}^{t+1}$), we obtain

$$\begin{aligned}
\mathcal{R}_T &\leq \sum_{t=1}^{T} \widetilde{\delta}_1^t \\
&\lesssim H^2 \left|\mathcal{C}_\sigma\right| + LHT\sigma + \sum_{t=1}^{T}\sum_{h=1}^{H} \left(1+\frac{1}{H}\right)^h \xi_{h+1}^t \\
&+ \sum_{t=1}^{T}\sum_{h=1}^{H} \left(\frac{H}{\sqrt{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)}} + \frac{H^2 \left|\mathcal{C}'_\sigma\right|}{\mathbf{C}_h^t(\widetilde{s}_h^t, \widetilde{a}_h^t)} \right) \mathbb{1}\left\{ \rho\left[(\widetilde{s}_h^t, \widetilde{a}_h^t), (s_h^t, a_h^t) \right] \leq 2\sigma \right\} \\
&+ \sum_{t=1}^{T}\sum_{h=1}^{H} \left(1+\frac{1}{H}\right)^h \left(V_{h+1}^t - V_{h+1}^{t+1}\right)(s_{h+1}^t) \qquad \text{(C.4)}
\end{aligned}$$

This bound differs only by the additive term (C.4) from the bound given in Lemma C.11. Thus we just need to handle this sum and rely on the previous analysis to upper bound the other terms. We consider the following partition of the state space:

Definition C.14. *Let $\mathcal{C}'_\sigma$ be a σ-covering of $\mathcal{S}$. We write $\mathcal{C}'_\sigma := \{s_j, j \in [|\mathcal{C}'_\sigma|]\}$. For each $s_j \in \mathcal{C}'_\sigma$, we define the set $B_j \subset \mathcal{S}$ as the set of points in $\mathcal{S}$ whose nearest neighbor in $\mathcal{C}'_\sigma$ is s_j, with ties broken arbitrarily, such that $\{B_j\}_{j\in[|\mathcal{C}'_\sigma|]}$ form a partition of $\mathcal{S}$.*

Using the fact that the V_h^t are point-wise non-increasing we can write the sum (C.4) as a telescopic sum:

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{h=1}^{H} \left(V_{h+1}^t - V_{h+1}^{t+1}\right)(s_{h+1}^t) \leq \sum_{t=1}^{T}\sum_{h=1}^{H} \left(V_{h+1}^t - V_{h+1}^{t+1}\right)(s_{h+1}^t) \\
&\leq \sum_{j=1}^{|\mathcal{C}'_\sigma|}\sum_{t=1}^{T}\sum_{h=1}^{H} \left(V_{h+1}^t - V_{h+1}^{t+1}\right)(s_{h+1}^t) \mathbb{1}\left\{s_{h+1}^t \in B_j\right\} \\
&\leq \sum_{j=1}^{|\mathcal{C}'_\sigma|}\sum_{t=1}^{T}\sum_{h=1}^{H} \left(V_{h+1}^t - V_{h+1}^{t+1}\right)(s_j) \mathbb{1}\left\{s_{h+1}^t \in B_j\right\} + 2L\rho_{\mathcal{S}}\left(s_j, s_{h+1}^t\right) \mathbb{1}\left\{s_{h+1}^t \in B_j\right\} \\
&\leq \sum_{j=1}^{|\mathcal{C}'_\sigma|}\sum_{h=1}^{H}\sum_{t=1}^{T} \left(V_{h+1}^t - V_{h+1}^{t+1}\right)(s_j) + T\sum_{h=1}^{H} 2L\sigma \\
&\leq H^2 \left|\mathcal{C}'_\sigma\right| + 2\sigma LHT\,,
\end{aligned}$$

where in the third inequality, we used the fact that the function $V_{h+1}^t - V_{h+1}^{t+1}$ is $2L$-Lipschitz. Combining the previous inequalities and the proof of Theorem 4.7, as explained above, allows us to conclude.

C.5 Detailed Description of RS-KeRNS

RS-KeRNS is described in Algorithm C.1, which uses a backward induction on representative states (Algorithm C.2) and updates the model online (algorithms 4.4 and C.3). In this section, we introduce the main definitions used by RS-KeRNS, and we analyze its runtime.

C.5.1 Definitions

In each episode t and for each h, RS-KeRNS keeps and updates sets of representative states $\bar{\mathcal{S}}_h^t$, actions $\bar{\mathcal{A}}_h^t$, and next-states $\bar{\mathcal{Y}}_h^t$, with cardinalities $\bar{S}_h^t$, $\bar{A}_h^t$ and $\bar{Y}_h^t$, respectively.

These sets are built using the data observed up to episode $t-1$. We define the following projections:

$$\zeta_h^{t+1}(s,a) := \operatorname*{argmin}_{(\bar{s},\bar{a})\in\bar{\mathcal{S}}_h^t\times\bar{\mathcal{A}}_h^t} \rho\left[(s,a),(\bar{s},\bar{a})\right], \quad \bar{\zeta}_h^{t+1}(y) := \operatorname*{argmin}_{\bar{y}\in\bar{\mathcal{Y}}_h^t} \rho_{\mathcal{S}}\left(y,\bar{y}\right).$$

where we also assume to have access to the metric $\rho_{\mathcal{S}}$. The definitions below introduce the kernel function and the estimated MDP used by RS-KeRNS.

Definition C.15 (kernel function for RS-KeRNS)**.** *Let $\lambda \in]0,1]$. RS-KeRNS uses a kernel of the form $\Gamma(n,u,v) = \chi(n)\phi(u,v)$, where*

$$\chi(n) := \lambda^n, \quad \textit{and} \quad \phi(u,v) := \exp\left(-\rho\left[u,v\right]^2/(2\sigma^2)\right).$$

Definition C.16 (empirical MDP for RS-KeRNS)**.** *Let*

$$\breve{W}_h^{t+1}(s,a) = \sum_{i=1}^{t} \chi(t-i)\phi\left(\zeta_h^{t+1}(s,a), \zeta_h^{i+1}(s_h^i,a_h^i)\right).$$

In episode $t+1$, RS-KeRNS uses the following estimate of the reward function

$$\breve{r}_h^{t+1}(s,a) = \frac{1}{\beta + \breve{W}_h^{t+1}(s,a)} \sum_{i=1}^{t} \chi(t-i)\phi\left(\zeta_h^{t+1}(s,a), \zeta_h^{i+1}(s_h^i,a_h^i)\right)\tilde{r}_h^i$$

and the follow estimate of the transitions

$$\breve{p}_h^{t+1}(y|s,a) = \frac{1}{\beta + \breve{W}_h^{t+1}(s,a)} \sum_{i=1}^{t} \chi(t-i)\phi\left(\zeta_h^{t+1}(s,a), \zeta_h^{i+1}(s_h^i,a_h^i)\right)\delta_{\bar{\zeta}_h^{i+1}(s_{h+1}^i)}(y).$$

Also, its exploration bonuses are computed as

$$\breve{\mathtt{b}}_h^{t+1}(s,a) := \widetilde{\mathcal{O}}\left(\frac{H}{\sqrt{\beta + \breve{W}_h^{t+1}(s,a)}} + \frac{\beta H}{\beta + \breve{W}_h^{t+1}(s,a)} + L\sigma\right)$$

where the factors hidden by $\widetilde{\mathcal{O}}(\cdot)$ are given in Definition C.1.

RS-KeRNS needs to store the quantities in Definition C.16 *only for the representatives (s,a) in $\bar{\mathcal{S}}_h^{t+1} \times \bar{\mathcal{A}}_h^{t+1}$ and $y \in \bar{\mathcal{Y}}_h^{t+1}$*. We will show that, using the auxiliary quantities defined below, the values of $\breve{W}_h^t$, $\breve{r}_h^t$ and $\breve{p}_h^t$ can be updated online in $\mathcal{O}\left(\sum_h \bar{S}_h^t \bar{A}_h^t \bar{Y}_h^t\right)$ time per episode t.

Definition C.17 (auxiliary quantities for online updates)**.** *For any (h,s,a,y), we define*

$$\breve{N}_h^{t+1}(s,a,y) := \sum_{i=1}^{t} \chi(t-i)\mathbb{1}\left\{\zeta_h^{i+1}(s_h^i,a_h^i) = (s,a)\right\}\delta_{\bar{\zeta}_h^{i+1}(s_{h+1}^i)}(y)$$

$$\breve{N}_h^{t+1}(s,a) := \sum_{i=1}^{t} \chi(t-i)\mathbb{1}\left\{\zeta_h^{i+1}(s_h^i,a_h^i) = (s,a)\right\}$$

$$\breve{S}_h^{t+1}(s,a) := \sum_{i=1}^{t} \chi(t-i)\mathbb{1}\left\{\zeta_h^{i+1}(s_h^i,a_h^i) = (s,a)\right\}\widetilde{r}_h^i.$$

Notice that, if $(s,a) \notin \bar{\mathcal{S}}_h^{t+1} \times \bar{\mathcal{A}}_h^{t+1}$, the quantities above are equal to zero.

The following lemma will be necessary in order to derive online updates.

Lemma C.18. *The empirical MDP used by RS-KeRNS can be computed as*

$$\breve{r}_h^{t+1}(s,a) = \frac{\sum_{(\bar{s},\bar{a})} \phi\left(\zeta_h^{t+1}(s,a),(\bar{s},\bar{a})\right)\breve{S}_h^{t+1}(\bar{s},\bar{a})}{\beta + \sum_{(\bar{s},\bar{a})} \phi\left(\zeta_h^{t+1}(s,a),(\bar{s},\bar{a})\right)\breve{N}_h^{t+1}(\bar{s},\bar{a})} \tag{C.5}$$

$$\breve{p}_h^{t+1}(y|s,a) = \frac{\sum_{(\bar{s},\bar{a})} \phi\left(\zeta_h^{t+1}(s,a),(\bar{s},\bar{a})\right)\breve{N}_h^{t+1}(\bar{s},\bar{a},y)}{\beta + \sum_{(\bar{s},\bar{a})} \phi\left(\zeta_h^{t+1}(s,a),(\bar{s},\bar{a})\right)\breve{N}_h^{t+1}(\bar{s},\bar{a})} \tag{C.6}$$

$$\breve{W}_h^{t+1}(s,a) = \sum_{(\bar{s},\bar{a})} \phi\left(\zeta_h^{t+1}(s,a),(\bar{s},\bar{a})\right)\breve{N}_h^{t+1}(\bar{s},\bar{a}) \tag{C.7}$$

where the sums are over $(\bar{s},\bar{a}) \in \bar{\mathcal{S}}_h^{t+1} \times \bar{\mathcal{A}}_h^{t+1}$.

Proof. It is an immediate consequence of the definitions. For instance,

$$\breve{W}_h^{t+1}(s,a) = \sum_{i=1}^{t} \chi(t-i)\phi\left(\zeta_h^{t+1}(s,a),\zeta_h^{i+1}(s_h^i,a_h^i)\right)$$

$$= \sum_{i=1}^{t} \chi(t-i)\phi\left(\zeta_h^{t+1}(s,a),\zeta_h^{i+1}(s_h^i,a_h^i)\right) \sum_{(\bar{s},\bar{a}) \in \bar{\mathcal{S}}_h^{t+1} \times \bar{\mathcal{A}}_h^{t+1}} \mathbb{1}\left\{\zeta_h^{i+1}(s_h^i,a_h^i) = (\bar{s},\bar{a})\right\}$$

$$= \sum_{(\overline{s},\overline{a})} \sum_{i=1}^{t} \chi(t-i)\phi\left(\zeta_h^{t+1}(s,a),(\overline{s},\overline{a})\right) \mathbb{1}\left\{\zeta_h^{i+1}(s_h^i,a_h^i) = (\overline{s},\overline{a})\right\}$$

$$= \sum_{(\overline{s},\overline{a})} \phi\left(\zeta_h^{t+1}(s,a),(\overline{s},\overline{a})\right) \sum_{i=1}^{t} \chi(t-i) \mathbb{1}\left\{\zeta_h^{i+1}(s_h^i,a_h^i) = (\overline{s},\overline{a})\right\}$$

$$= \sum_{(\overline{s},\overline{a})} \phi\left(\zeta_h^{t+1}(s,a),(\overline{s},\overline{a})\right) \breve{N}_h^{t+1}(\overline{s},\overline{a}).$$

□

Algorithm C.1: RS-KeRNS

initialization: $\bar{\mathcal{S}}_h = \emptyset$, $\bar{\mathcal{A}}_h \leftarrow \emptyset$, $\bar{\mathcal{Y}}_h \leftarrow \emptyset$, for $h \in [H]$.
for *episode* $t = 1, \dots, T$ **do**
  get initial state s_1^t
  compute $(\breve{Q}_h^t)_h$ using Algorithm C.2
  for $h = 1, \dots, H$ **do**
    # select action
    $a_h^t \leftarrow \operatorname{argmax}_a \breve{Q}_h^t(s_h^t, a)$
    # execute action
    $\widetilde{r}_h^t, s_{h+1}^t \leftarrow \texttt{OnlineModel}_{t,h}(a_h^t)$
    update $\bar{\mathcal{S}}_h, \bar{\mathcal{A}}_h, \bar{\mathcal{Y}}_h$ using $\left\{s_h^t, a_h^t, s_{h+1}^t\right\}$ with Algorithm 4.4
    update model using $s_h^t, a_h^t, s_{h+1}^t, \widetilde{r}_h^t$ with Algorithm C.3

Algorithm C.2: Kernel Backward Induction on Representative States

input: $\breve{r}_h^t(\overline{s},\overline{a})$, $\breve{p}_h^t(\overline{y}|\overline{s},\overline{a})$, $\breve{\mathfrak{b}}_h^t(\overline{s},\overline{a})$ for all $(\overline{s},\overline{a},\overline{y}) \in \bar{\mathcal{S}}_h^t \times \bar{\mathcal{A}}_h^t \times \bar{\mathcal{Y}}_h^t$ and all $h \in [H]$.
initialization: $\breve{V}_{H+1}(s) \leftarrow 0$ for all $s \in \mathcal{S}$
for $h = H, \dots, 1$ **do**
  for $(\overline{s},\overline{a}) \in \bar{\mathcal{S}}_h^t \times \bar{\mathcal{A}}_h^t$ **do**
    $\widetilde{Q}_{h,\zeta}^t(\overline{s},\overline{a}) \leftarrow \breve{r}_h^t(\overline{s},\overline{a}) + \breve{p}_h^t \breve{V}_{h+1}(\overline{s},\overline{a}) + \breve{\mathfrak{b}}_h^t(\overline{s},\overline{a})$
  # Interpolated Q-function. Defined, but not computed for all (s,a)
  $\breve{Q}_h^t(s,a) = \min_{(\overline{s},\overline{a}) \in \bar{\mathcal{S}}_h^t \times \bar{\mathcal{A}}_h^t} \left(\widetilde{Q}_{h,\zeta}^t(\overline{s},\overline{a}) + L\rho\left[(s,a),(\overline{s},\overline{a})\right]\right)$
  if $h > 1$ **then**
    # Compute value function at the next states for the stage $h-1$
    for $\overline{y} \in \bar{\mathcal{Y}}_{h-1}^t$ **do**
      $\breve{V}_h^t(\overline{y}) = \min\left(H - h + 1, \max_a \breve{Q}_h^t(\overline{y}, a)\right)$
return: $(\breve{Q}_h^t)_{h \in [H]}$

Algorithm C.3: Online Update of RS-KeRNS Model

input: $t, h, s_h^t, a_h^t, s_{h+1}^t, \widetilde{r}_h^t$.
Map to representatives
Map $(\widetilde{s}, \widetilde{a}) = \zeta_h^{t+1}(s_h^t, a_h^t)$ and $\widetilde{y} = \overline{\zeta}_h^{t+1}(s_{h+1}^t)$
Update auxiliary quantities
$\breve{N}_h^{t+1}(\widetilde{s}, \widetilde{a}, \widetilde{y}) = 1 + \lambda \breve{N}_h^t(\widetilde{s}, \widetilde{a}, \widetilde{y})$
$\breve{N}_h^{t+1}(\widetilde{s}, \widetilde{a}) = 1 + \lambda \breve{N}_h^t(\widetilde{s}, \widetilde{a})$
$\breve{S}_h^{t+1}(\widetilde{s}, \widetilde{a}) = \widetilde{r}_h^t + \lambda \breve{S}_h^t(\widetilde{s}, \widetilde{a})$
# Update empirical MDP
for $(\overline{s}, \overline{a}) \in \bar{\mathcal{S}}_h^{t+1} \times \bar{\mathcal{A}}_h^{t+1}$ **do**
  if $(\overline{s}, \overline{a}) \in \bar{\mathcal{S}}_h^t \times \bar{\mathcal{A}}_h^t$ **then**
    # $(\overline{s}, \overline{a})$ was added before episode t
    $\breve{W}_h^{t+1}(\overline{s}, \overline{a}) = \phi\left((\overline{s}, \overline{a}), (\widetilde{s}, \widetilde{a})\right) + \lambda \breve{W}_h^t(\overline{s}, \overline{a})$
    $\breve{r}_h^{t+1}(\overline{s}, \overline{a}) = \frac{\phi\left((\overline{s}, \overline{a}), (\widetilde{s}, \widetilde{a})\right)}{\beta + \breve{W}_h^{t+1}(\overline{s}, \overline{a})} \widetilde{r}_h^t + \lambda \left(\frac{\beta + \breve{W}_h^t(\overline{s}, \overline{a})}{\beta + \breve{W}_h^{t+1}(\overline{s}, \overline{a})} \right) \breve{r}_h^t(\overline{s}, \overline{a})$
    for $y \in \bar{\mathcal{Y}}_h^{t+1}$ **do**
      $\breve{p}_h^{t+1}(y|\overline{s}, \overline{a}) = \frac{\phi\left((\overline{s}, \overline{a}), (\widetilde{s}, \widetilde{a})\right)}{\beta + \breve{W}_h^{t+1}(\overline{s}, \overline{a})} \delta_{\widetilde{y}}(y) + \lambda \left(\frac{\beta + \breve{W}_h^t(\overline{s}, \overline{a})}{\beta + \breve{W}_h^{t+1}(\overline{s}, \overline{a})} \right) \breve{p}_h^t(y|\overline{s}, \overline{a})$
  else
    # $(\overline{s}, \overline{a})$ was added in episode k
    Initialize $\breve{r}_h^{t+1}(\overline{s}, \overline{a}), \breve{p}_h^{t+1}(\cdot|\overline{s}, \overline{a}), \breve{W}_h^{t+1}(\overline{s}, \overline{a})$ using equations (C.5), (C.6) and (C.7)
return:

C.5.2 Online updates & runtime

Assume that we observed a transition $\left\{s_h^t, a_h^t, s_{h+1}^t, \widetilde{r}_h^t\right\}$ at time (t, h), updated the representative sets, and mapped the transition to the representatives $(\widetilde{s}, \widetilde{a}, \widetilde{y}) \in \bar{\mathcal{S}}_h^{t+1} \times \bar{\mathcal{A}}_h^{t+1} \times \bar{\mathcal{Y}}_h^{t+1}$. We wish to update the estimated MDP given in Definition C.16, which, at step h, are only stored for (s, a) in $\bar{\mathcal{S}}_h^{t+1} \times \bar{\mathcal{A}}_h^{t+1}$ and $y \in \bar{\mathcal{Y}}_h^{t+1}$.

The auxiliary quantities (Definition C.17) are updated as:

$$\begin{aligned}
\breve{N}_h^{t+1}(\widetilde{s}, \widetilde{a}, \widetilde{y}) &= 1 + \lambda \breve{N}_h^t(\widetilde{s}, \widetilde{a}, \widetilde{y}) \\
\breve{N}_h^{t+1}(\widetilde{s}, \widetilde{a}) &= 1 + \lambda \breve{N}_h^t(\widetilde{s}, \widetilde{a}) \\
\breve{S}_h^{t+1}(\widetilde{s}, \widetilde{a}) &= \widetilde{r}_h^t + \lambda \breve{S}_h^t(\widetilde{s}, \widetilde{a}).
\end{aligned}$$

We need to update $\breve{W}_h^t$, $\breve{r}_h^t$ and $\breve{p}_h^t$ for all $(\overline{s}, \overline{a}, \overline{y}) \in \bar{\mathcal{S}}_h^{t+1} \times \bar{\mathcal{A}}_h^{t+1} \times \bar{\mathcal{Y}}_h^{t+1}$. The update rule will depend on whether the $(\overline{s}, \overline{a})$ is a *new* representative state-action pair (included in episode t) or it was *visited before episode* t. These two cases are studied below.

Case 1: $(\bar{s},\bar{a}) \in \bar{\mathcal{S}}_h^{t+1} \times \bar{\mathcal{A}}_h^{t+1}$ **and** $(\bar{s},\bar{a}) \notin \bar{\mathcal{S}}_h^{t} \times \bar{\mathcal{A}}_h^{t}$ This means that the representative state-action pair $(\bar{s},\bar{a})$ was added at time (t,h). In this case, for all $y \in \bar{\mathcal{Y}}_h^{t+1}$, the quantities $\breve{r}_h^{t+1}(\bar{s},\bar{a})$, $\breve{p}_h^{t+1}(y|\bar{s},\bar{a})$ and $\breve{W}_h^{t+1}(\bar{s},\bar{a})$ can be initialized using equations (C.5), (C.6) and (C.7). This is done in $\mathcal{O}\left(\bar{S}_h^{t+1}\bar{A}_h^{t+1}\bar{Y}_h^{t+1}\right)$ time and can happen, at most, for one pair $(\bar{s},\bar{a})$: the one that was newly added. Therefore, we have a total per-episode runtime of $\mathcal{O}\left(\sum_{h=1}^{H}\bar{S}_h^{t+1}\bar{A}_h^{t+1}\bar{Y}_h^{t+1}\right)$ taking this case into account.

Case 2: $(\bar{s},\bar{a}) \in \bar{\mathcal{S}}_h^{t} \times \bar{\mathcal{A}}_h^{t}$ This means that the representative state-action pair $(\bar{s},\bar{a})$ was added *before* episode t, which implies that $\zeta_h^{t+1}(\bar{s},\bar{a}) = \zeta_h^{t}(\bar{s},\bar{a}) = (\bar{s},\bar{a})$. Hence,

$$\begin{aligned}
\breve{W}_h^{t+1}(\bar{s},\bar{a}) &= \sum_{i=1}^{t}\chi(t-i)\phi\left(\zeta_h^{t+1}(\bar{s},\bar{a}),\zeta_h^{i+1}(s_h^i,a_h^i)\right)\\
&= \phi\left(\zeta_h^{t+1}(\bar{s},\bar{a}),\zeta_h^{t+1}(s_h^t,a_h^t)\right) + \sum_{i=1}^{t-1}\lambda^{t-i}\phi\left(\zeta_h^{t}(\bar{s},\bar{a}),\zeta_h^{i+1}(s_h^i,a_h^i)\right)\\
&= \phi\left((\bar{s},\bar{a}),\zeta_h^{t+1}(s_h^t,a_h^t)\right) + \lambda\sum_{i=1}^{t-1}\lambda^{t-i-1}\phi\left(\zeta_h^{t}(\bar{s},\bar{a}),\zeta_h^{i+1}(s_h^i,a_h^i)\right)\\
&= \phi\left((\bar{s},\bar{a}),\zeta_h^{t+1}(s_h^t,a_h^t)\right) + \lambda\breve{W}_h^{t}(\bar{s},\bar{a}),
\end{aligned}$$

This implies that, for a fixed $(\bar{s},\bar{a})$, the quantity $\breve{W}_h^{t+1}(\bar{s},\bar{a})$ can be updated in $\mathcal{O}(1)$ time, assuming that the mapping $\zeta_h^{t+1}(s_h^t,a_h^t)$ was previously computed (this mapping is only computed *once* for all the updates, and takes $\mathcal{O}\left(\bar{S}_h^{t+1}\times\bar{A}_h^{t+1}\right)$ time).

Now, notice that

$$\begin{aligned}
\breve{r}_h^{t+1}(\bar{s},\bar{a}) &= \frac{\sum_{i=1}^{t}\chi(t-i)\phi\left(\zeta_h^{t+1}(\bar{s},\bar{a}),\zeta_h^{i+1}(s_h^i,a_h^i)\right)\tilde{r}_h^i}{\beta+\breve{W}_h^{t+1}(\bar{s},\bar{a})}\\
&= \frac{\phi\left((\bar{s},\bar{a}),\zeta_h^{t+1}(s_h^t,a_h^t)\right)}{\beta+\breve{W}_h^{t+1}(\bar{s},\bar{a})}\tilde{r}_h^t + \lambda\left(\frac{\beta+\breve{W}_h^{t}(\bar{s},\bar{a})}{\beta+\breve{W}_h^{t+1}(\bar{s},\bar{a})}\right)\breve{r}_h^t(\bar{s},\bar{a})
\end{aligned}$$

where we used again the fact that, in this case, $\zeta_h^{t+1}(\bar{s},\bar{a}) = \zeta_h^{t}(\bar{s},\bar{a})$. Hence, similarly to $\breve{W}_h^{t+1}(\bar{s},\bar{a})$, the quantity $\breve{r}_h^{t+1}(\bar{s},\bar{a})$ can be updated in $\mathcal{O}(1)$ time. A similar reasoning shows that $\breve{p}_h^{t+1}(y,\bar{s},\bar{a})$ can be updated, for all $y \in \bar{\mathcal{Y}}_h^{t+1}$, in $\mathcal{O}\left(\bar{Y}_h^{t+1}\right)$ time:

$$\breve{p}_h^{t+1}(y|\bar{s},\bar{a}) = \frac{\phi\left((\bar{s},\bar{a}),\zeta_h^{t+1}(s_h^t,a_h^t)\right)}{\beta+\breve{W}_h^{t+1}(\bar{s},\bar{a})}\delta_{\bar{\zeta}_h^{t+1}(s_{h+1}^t)}(y) + \lambda\left(\frac{\beta+\breve{W}_h^{t}(\bar{s},\bar{a})}{\beta+\breve{W}_h^{t+1}(\bar{s},\bar{a})}\right)\breve{p}_h^t(y|\bar{s},\bar{a}).$$

Summary Every time a new transition is observed at time (t,h), the estimators for all $(s,a,y) \in \bar{\mathcal{S}}_h^{t+1}\times\bar{\mathcal{A}}_h^{t+1}\times\bar{\mathcal{Y}}_h^{t+1}$ must be updated. For a given representative (s,a), the updates can be done

in $\mathcal{O}\left(\bar{Y}_h^{t+1}\right)$ time if it has been observed before episode t (case 2). This results in a total runtime, per episode, of $\mathcal{O}\left(\sum_h \bar{S}_h^{t+1}\bar{A}_h^{t+1}\bar{Y}_h^{t+1}\right)$ for all the representatives observed before episode t. If the representative (s,a) has not been observed before episode t (case 1), the updates require $\mathcal{O}\left(\bar{S}_h^{t+1}\bar{A}_h^{t+1}\bar{Y}_h^{t+1}\right)$ time, and this can happen, at most, for one state-action pair at each time (t,h). Hence, the total runtime required for the updates is $\mathcal{O}\left(\sum_h \bar{S}_h^{t+1}\bar{A}_h^{t+1}\bar{Y}_h^{t+1}\right)$ per episode.

C.6 Proof Sketch for Theorem 4.16: Regret of RS-KeRNS

To prove the regret bound in Theorem 4.16 for RS-KeRNS, we consider the kernel:

$$\Gamma(n,u,v) = \chi(n)\phi\left(u,v\right), \text{ where } \quad \phi\left(u,v\right) := \exp\left(-\rho\left[u,v\right]^2/(2\sigma^2)\right),$$

for a given function $\chi : \mathbb{N} \to [0,1]$. In each episode t, RS-KeRNS has build representative sets of states $\bar{\mathcal{S}}_h^t$, actions $\bar{\mathcal{A}}_h^t$ and next states $\bar{\mathcal{Y}}_h^t$, for each $h \in [H]$. We recall the definition of the projections

$$\zeta_h^t(s,a) := \operatorname*{argmin}_{(\bar{s},\bar{a})\in\bar{\mathcal{S}}_h^t\times\bar{\mathcal{A}}_h^t} \rho\left[(s,a),(\bar{s},\bar{a})\right], \quad \bar{\zeta}_h^t(y) := \operatorname*{argmin}_{\bar{y}\in\bar{\mathcal{Y}}_h^t} \rho_{\mathcal{S}}\left(y,\bar{y}\right).$$

from any (s,a,y) to their representatives.

Let $\breve{W}_h^{t+1}(s,a) = \sum_{i=1}^t \chi(t-i)\phi\left(\zeta_h^{t+1}(s,a), \zeta_h^{i+1}(s_h^i,a_h^i)\right)$. In episode $t+1$, RS-KeRNS computes the following estimate of the rewards

$$\breve{r}_h^{t+1}(s,a) = \frac{1}{\beta + \breve{W}_h^{t+1}(s,a)} \sum_{i=1}^t \chi(t-i)\phi\left(\zeta_h^{t+1}(s,a), \zeta_h^{i+1}(s_h^i,a_h^i)\right)\tilde{r}_h^i$$

and the following estimate of the transitions

$$\breve{p}_h^{t+1}(y|s,a) = \frac{1}{\beta + \breve{W}_h^{t+1}(s,a)} \sum_{i=1}^t \chi(t-i)\phi\left(\zeta_h^{t+1}(s,a), \zeta_h^{i+1}(s_h^i,a_h^i)\right)\delta_{\bar{\zeta}_h^{i+1}(s_{h+1}^i)}(y).$$

which are similar to the estimates that would be computed by KeRNS, but using the projections ζ and $\bar{\zeta}$ to the representative states and actions. The values of $\breve{r}_h^{t+1}(s,a)$ and $\breve{p}_h^{t+1}(y|s,a)$ are defined for all $(s,a,y) \in \mathcal{S}\times\mathcal{A}\times\mathcal{S}$, but they only need to be stored for $(s,a,y) \in \bar{\mathcal{S}}_h^t \times \bar{\mathcal{A}}_h^t \times \bar{\mathcal{Y}}_h^t$, which corresponds to storing a *finite* representation of the MDP. The exploration bonuses of RS-KeRNS are defined similarly:

$$\breve{\mathfrak{b}}_h^{t+1}(s,a) := \widetilde{\mathcal{O}}\left(\frac{H}{\sqrt{\beta + \breve{W}_h^{t+1}(s,a)}} + \frac{\beta H}{\beta + \breve{W}_h^{t+1}(s,a)} + L\sigma\right)$$

We prove that the estimates used by RS-KeRNS are close to the ones used by KeRNS up to bias terms. Then, this result is used to prove that the regret bound of RS-KeRNS is the same as KeRNS, but adding a bias term multiplied by the number of episodes T. For any (s_h^i, a_h^i) with $i < t$ and $h \in [H]$, we show that:

$$\left|\left(\widehat{p}_h^t - \breve{p}_h^t\right) V(s_h^i, a_h^i)\right| \lesssim L\varepsilon_{\mathcal{X}} + 8H\frac{\varepsilon}{\sigma}$$

and similar bounds are obtained for the rewards $\breve{r}_h^t(s,a)$ and the exploration bonuses $\breve{\mathsf{b}}_h^t$. This allows us to prove that the dynamic regret of RS-KeRNS is bounded as

$$\mathcal{R}_T^{\text{RS-KeRNS}} \lesssim \mathcal{R}_T^{\text{KeRNS}} + L(\varepsilon + \varepsilon_{\mathcal{X}})H^2T + \frac{\varepsilon}{\sigma}H^3T.$$

C.7 Technical Lemmas

Lemma C.19. *Consider a sequence $\{a_n\}_{n\geq 1}$ of non-negative numbers such that $a_m \leq c$ for some constant $c > 0$. Let $A_t = \sum_{n=1}^{t-1} a_n$. Then, for any $b > 0$ and any $p > 0$,*

$$\sum_{t=1}^{T} \frac{a_t}{(1+bA_t)^p} \leq c + \int_0^{A_{T+1}-c} \frac{\mathrm{d}z}{(1+bz)^p}.$$

Proof. Let $n := \max\{t : a_1 + \ldots + a_{t-1} \leq c\}$. We have

$$\sum_{t=1}^{n-1} \frac{a_t}{(1+bA_t)^p} \leq \sum_{t=1}^{n-1} a_t \leq c$$

and, consequently,

$$\begin{aligned}
\sum_{t=1}^{T} \frac{a_t}{(1+bA_t)^p} &\leq c + \sum_{t=n}^{T} \frac{a_t}{(1+bA_t)^p} = c + \sum_{t=n}^{T} \frac{A_{t+1}-A_t}{(1+bA_t)^p} \\
&= c + \sum_{t=n}^{T} \frac{A_{t+1}-A_t}{(1+bA_{t+1}-ba_t)^p} \leq c + \sum_{t=n}^{T} \frac{A_{t+1}-A_t}{(1+b(A_{t+1}-c))^p} \\
&= c + \sum_{t=n}^{T} \int_{A_t}^{A_{t+1}} \frac{1}{(1+b(A_{t+1}-c))^p}\mathrm{d}z \leq c + \sum_{t=n}^{T} \int_{A_t}^{A_{t+1}} \frac{1}{(1+b(z-c))^p}\mathrm{d}z \\
&= c + \int_{A_n}^{A_{T+1}} \frac{1}{(1+b(z-c))^p}\mathrm{d}z \leq c + \int_{c}^{A_{T+1}} \frac{1}{(1+b(z-c))^p}\mathrm{d}z\,.
\end{aligned}$$

□

Lemma C.20. *Consider a sequence of non-negative real numbers* $\{z_s\}_{s=1}^t$ *and let* $\overline{\Gamma} : \mathbb{R}_+ \to [0,1]$ *satisfy Assumption 4.4. Let*

$$w_s := \overline{\Gamma}\left(\frac{z_s}{\sigma}\right) \quad \text{and} \quad \widetilde{w}_s := \frac{w_s}{\beta + \sum_{s'=1}^t w_{s'}}$$

for $\beta > 0$*. Then, we have*

$$\sum_{s=1}^t \widetilde{w}_s z_s \leq 2\sigma\left(1 + \sqrt{\log(C_1 t/\beta + e)}\right).$$

Proof. We split the sum into two terms:

$$\sum_{s=1}^t \widetilde{w}_s z_s = \sum_{s:z_s<c} \widetilde{w}_s z_s + \sum_{s:z_s\geq c} \widetilde{w}_s z_s \leq c + \sum_{s:z_s\geq c} \widetilde{w}_s\,.$$

From Assumption 4.4, we have $w_s \leq C_1 \exp\left(-z_s^2/(2\sigma^2)\right)$. Hence, $\widetilde{w}_s \leq (C_1/\beta)\exp\left(-z_s^2/(2\sigma^2)\right)$, since $\beta + \sum_{s'=1}^t w_{s'} \geq \beta$.

We want to find c such that:

$$z_s \geq c \implies \frac{C_1}{\beta}\exp\left(-\frac{z_s^2}{2\sigma^2}\right) \leq \frac{1}{t}\frac{2\sigma^2}{z_s^2}$$

which implies, for $z_s \geq c$, that $\widetilde{w}_s \leq \frac{1}{t}\frac{2\sigma^2}{z_s^2}$.

Let $x = z_s^2/2\sigma^2$. Reformulating, we want to find a value c' such that $C_1\exp(-x) \leq \beta/(xt)$ for all $x \geq c'$. Let $c' = 2\log(C_1 t/\beta + e)$. If $x \geq c'$, we have:

$$\frac{x}{2} \geq \log\left(\frac{C_1 t}{\beta} + e\right) \implies x \geq \frac{x}{2} + \log\left(\frac{C_1 t}{\beta} + e\right) \implies x \geq \log x + \log(C_1 t/\beta + e)$$
$$\implies (C_1/\beta)\exp(-x) \leq 1/(xt)$$

as we wanted.

Now, $x \geq c'$ is equivalent to $z_s \geq \sqrt{2\sigma^2 c'} = 2\sigma\sqrt{\log(C_1 t/\beta + e)}$. Therefore, we take $c = 2\sigma\sqrt{\log(C_1 t/\beta + e)}$, which gives us

$$\sum_{s:z_s\geq c} \widetilde{w}_s z_s \leq \sum_{s:z_s\geq c} \frac{1}{t}\frac{2\sigma^2}{z_s^2} z_s \leq \frac{2\sigma^2}{t}\sum_{s:z_s\geq c}\frac{1}{z_s} \leq \frac{2\sigma^2}{c}\frac{|\{s : z_s \geq c\}|}{t} \leq \frac{2\sigma^2}{c}\,.$$

Finally, we obtain:

$$\sum_{s=1}^t \widetilde{w}_s z_s \leq c + \sum_{s:z_s\geq c} \widetilde{w}_s z_s \leq c + \frac{2\sigma^2}{c}$$

$$= 2\sigma\sqrt{\log(C_1 t/\beta + e)} + \frac{\sigma}{\sqrt{\log(C_1 t/\beta + e)}} \leq 2\sigma\left(1 + \sqrt{\log(C_1 t/\beta)}\right).$$

□

Lemma C.21. *Let $\overline{\Gamma} : \mathbb{R}_+ \to [0,1]$ be a kernel that satisfies Assumption 4.4. Let $a \in \mathbb{R}_+^t$ and f_1, f_2, f_3 be functions from $\mathbb{R}_+^t$ to $\mathbb{R}$ defined as*

$$f_1(z) = \frac{\sum_{s=1}^t \overline{\Gamma}(z_s/\sigma)\, a_s}{\beta + \sum_{s=1}^t \overline{\Gamma}(z_s/\sigma)},$$
$$f_2(z) = \sqrt{\frac{1}{\beta + \sum_{s=1}^t \overline{\Gamma}(z_s/\sigma)}},$$
$$f_3(z) = \frac{1}{\beta + \sum_{s=1}^t \overline{\Gamma}(z_s/\sigma)}$$

Then, for any $y, z \in \mathbb{R}_+$, we have

$$|f_1(z) - f_1(y)| \leq \frac{2C_2 \|a\|_\infty t}{\beta\sigma} \|z - y\|_\infty$$
$$|f_2(z) - f_2(y)| \leq \frac{C_2 t}{2\beta^{3/2}\sigma} \|z - y\|_\infty$$
$$|f_3(z) - f_3(y)| \leq \frac{C_2 t}{\beta^2\sigma} \|z - y\|_\infty$$

Proof. From Assumption 4.4, the function $z \mapsto \overline{\Gamma}(z)$ is C_2-Lipschitz , which yields

$$\begin{aligned}
&|f_1(z) - f_1(y)| \\
&\leq \left| \frac{\sum_{s=1}^t \left(\overline{\Gamma}(z_s/\sigma) - \overline{\Gamma}(y_s/\sigma)\right) a_s}{\beta + \sum_{s=1}^t \overline{\Gamma}(z_s/\sigma)} \right| \\
&+ \left| \frac{\sum_{s=1}^t \overline{\Gamma}(y_s/\sigma)\, a_s}{\beta + \sum_{s=1}^t \overline{\Gamma}_{(\lambda,W)}(t-s-1, y_s/\sigma)} \right| \left| \frac{\sum_{s=1}^t \left(\overline{\Gamma}(z_s/\sigma) - \overline{\Gamma}(y_s/\sigma)\right)}{\beta + \sum_{s=1}^t \overline{\Gamma}(z_s/\sigma)} \right| \\
&\leq \frac{C_2 \sum_{s=1}^t (1/\sigma)\, |z_s - y_s|\, a_s}{\beta + \sum_{s=1}^t \overline{\Gamma}(z_s/\sigma)} + \|a\|_\infty \frac{C_2 \sum_{s=1}^t (1/\sigma)\, |z_s - y_s|}{\beta + \sum_{s=1}^t \overline{\Gamma}(z_s/\sigma)} \\
&\leq \frac{2C_2 \|a\|_\infty t}{\beta\sigma} \|z - y\|_\infty .
\end{aligned}$$

The proofs for f_2 and f_3 are analogous. For f_2, we also use the fact that the function $x \mapsto (1/\sqrt{\beta + x})$ is $1/(2\beta^{3/2})$-Lipschitz . □

Lemma C.22 (value functions are Lipschitz continuous)**.** *Under Assumptions 4.1 and 4.2, for all (t, h), the functions $V_{t,h}^*$ and $Q_{t,h}^*$ are L_h-Lipschitz , where $L_h := \sum_{h'=h}^H L_{\mathrm{r}} {L_{\mathrm{p}}}^{H-h'}$.*

Proof. This fact is proved in Lemma 4 of [Dom+21d] and also in Proposition 2.5 of [SBY19]. For completeness, we also present a proof here.

We proceed by induction. For $h = H$, $Q_H^*(s,a) = r_H(s,a)$ which is L_r-Lipschitz by Assumption 4.2. Also,

$$V_H^*(x) - V_H^*(y) = \max_a Q_H^*(x,a) - \max_a Q_H^*(y,a) \leq \max_a \left(Q_H^*(x,a) - Q_H^*(y,a)\right) \tag{C.8}$$

$$\leq \max_a L_H \rho\left[(x,a),(y,a)\right] \leq L_H \rho_{\mathcal{S}}\left(x,y\right), \quad \text{by Assumption 4.1} \tag{C.9}$$

which verifies the induction hypothesis for $h = H$, since we can invert the roles of x and y to obtain $|V_H^*(x) - V_H^*(y)| \leq L_H \rho_{\mathcal{S}}\left(x,y\right)$.

Now, assume that the hypothesis is true for $h+1$, *i.e.*, that V_{h+1}^* and Q_{h+1}^* are L_{h+1}-Lipschitz . We have

$$\begin{aligned}
Q_h^*(x,a) - Q_h^*(x',a') &\leq L_\mathrm{r}\rho\left[(x,a),(x',a')\right] + \int_{\mathcal{S}} V_{h+1}^*(y)(P_h(\mathrm{d}y|x,a) - P_h(\mathrm{d}y|x',a')) \\
&\leq L_\mathrm{r}\rho\left[(x,a),(x',a')\right] + L_{h+1}\int_{\mathcal{S}} \frac{V_{h+1}^*(y)}{L_{h+1}}(P_h(\mathrm{d}y|x,a) - P_h(\mathrm{d}y|x',a')) \\
&\leq \left[L_\mathrm{r} + L_\mathrm{p}\sum_{h'=h+1}^{H} L_\mathrm{r} {L_\mathrm{p}}^{H-h'}\right]\rho\left[(x,a),(x',a')\right] \\
&= \sum_{h'=h}^{H} L_\mathrm{r} {L_\mathrm{p}}^{H-h'}\rho\left[(x,a),(x',a')\right]
\end{aligned}$$

where, in last inequality, we use fact that V_{h+1}^*/L_{h+1} is 1-Lipschitz, the definition of the 1-Wasserstein distance and Assumption 4.2. The same argument used in Equation (C.8) shows that $|V_h^*(x) - V_h^*(y)| \leq L_h \rho_{\mathcal{S}}\left(x,y\right)$, which concludes the proof. □

List of Figures

3.1 Illustration of the class of hard MDPs for $S = 4$. 35
3.2 Illustration of the class of hard MDPs used in the proofs of Theorems 3.8 and 3.9. 36

4.1 Continuous Grid-World with two rooms separated by a wall. The circles represent the regions with non-zero rewards. 78
4.2 Regret of different versions of `Kernel-UCBVI` compared to baselines (**smaller is better**). To estimate the optimal value function for the regret computation, we used the best policy among all agents at the final episode. Average over 16 independent runs. 79
4.3 Total runtime of different algorithms in a continuous Grid-World versus the number of episodes (**smaller is better**). Average over 16 runs. 80
4.4 Cumulative rewards of `RS-KeRNS` versus `RS-Kernel-UCBVI` and `RestartBaseline` in a non-stationary environment (**larger is better**). The environment changes every 2.5×10^4 episodes. Average over 16 independent runs. 82

5.1 Grid-World with 9 rooms. The number of states is $S = 233$ (not counting the walls), and the agent's observations are one-hot encodings of the discrete states. 96
5.2 **Left:** true positions of each state in the Grid-World with 9 rooms, each state is associated to a unique color. **Middle:** 2-dimensional projection of the 64-dimensional embeddings before minimizing the loss (5.10). **Right:** 2-dimensional projection of the embeddings after 15×10^3 optimization steps to minimize the representation loss (5.10). 99
5.3 Visualization of `AKBX`'s bonuses for $\alpha = 1/2$ (**top**) and $\alpha = 1$ (**bottom**). **Left:** Visualization of number of visits of each state in the GridWorld, when random trajectories are sampled starting from the center of the top-left room. Higher values are shown in magenta and smaller values are shown in blue. **Middle:** Bonuses (5.7) estimated by `AKBX` based on the learned representation function f. **Right:** Comparison between `AKBX`'s bonuses and $1/n^\alpha$, where n is the number of state visits. 100

5.4 Evaluation of reward-free exploration: number of visits and entropy of state-visit distribution in the 9-room Grid-World for AKBX, RND, RF-UCRL and RF-Express. Average over 4 independent runs. 101

5.5 Number of state visits (in logarithmic scale) for RND (top), AKBX for $\alpha = 1/2$ (middle) and RF-UCRL (bottom) in different time intervals. Each column corresponds to the number of visits in a given time interval. For instance, the first column shows state visits for between time $t = 0$ and $t = 3 \times 10^5$, whereas the last column shows the visits between $t = 12 \times 10^5$ and $t = 15 \times 10^5$. The colors are normalized per column, higher values are shown in magenta and smaller values are shown in blue. Results for RF-Express and AKBX with $\alpha = 1$ were omitted from this plot since they are very similar to those of RF-UCRL and AKBX with $\alpha = 1/2$, respectively. 102

5.6 Comparison between the values of the policies π_t learned by AKBX with $\alpha = 1/2$ and RND at time t to the value of an optimal agent. The values correspond to the expected sum of rewards in a horizon $H = 100$, starting from the top-left room of the Grid-World, and are estimated by Monte-Carlo policy evaluation. The optimal policy was computed using value iteration. Average over 4 runs. 103

List of Algorithms

2.1 MBQVI 14
2.2 SparseSampling 16
2.3 SmoothCruiser 23
2.4 sampleV 23
2.5 estimateQ 24

3.1 UCBVI 48

4.1 Kernel-UCBVI 58
4.2 Kernel Backward Induction 60
4.3 Kernel-UCBVI+RTDP 74
4.4 Update Representative Sets 75

5.1 RF-UCRL 87
5.2 Simplified structure of RL algorithms based on exploration bonuses 90
5.3 AKBX: Approximate Kernel-Based Exploration 93

C.1 RS-KeRNS 154
C.2 Kernel Backward Induction on Representative States 154
C.3 Online Update of RS-KeRNS Model 155

List of Tables

3.1 Algorithms matching the lower bounds in different settings. 55

4.1 Regret bound for KeRNS with optimized kernel parameters, for $W = \log_\lambda \frac{(1-\lambda)}{T}$. 72
4.2 Value of c_p^t for each to p in episode t. 81

5.1 Parameters used for representation learning 98
5.2 Parameters used to compute AKBX's bonuses 99
5.3 Parameters used in AWR implementation 102

References

[AKY20] Alekh Agarwal, Sham Kakade, and Lin F Yang. Model-based reinforcement learning with a generative model is minimax optimal. In *Conference on Learning Theory*. 2020.

[Aue+02] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing* (2002).

[AGO19] Peter Auer, Pratik Gajane, and Ronald Ortner. Adaptively tracking the best bandit arm with an unknown number of distribution changes. In *Conference on Learning Theory*. 2019.

[AMK12] Mohammad Gheshlaghi Azar, Rémi Munos, and Hilbert J Kappen. On the Sample Complexity of Reinforcement Learning with a Generative Model. In 2012.

[AOM17] Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *International Conference on Machine Learning*. 2017.

[Aza+19] Mohammad Gheshlaghi Azar, Bilal Piot, Bernardo Avila Pires, Jean-Bastien Grill, Florent Altché, and Rémi Munos. World discovery models. *arXiv preprint arXiv:1902.07685* (2019).

[Bad+20a] Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo Sprechmann, Alex Vitvitskyi, Zhaohan Daniel Guo, et al. Agent57: Outperforming the atari human benchmark. In *International Conference on Machine Learning*. 2020.

[Bad+20b] Adrià Puigdomènech Badia, Pablo Sprechmann, Alex Vitvitskyi, Daniel Guo, Bilal Piot, Steven Kapturowski, et al. Never Give Up: Learning Directed Exploration Strategies. In *International Conference on Learning Representations*. 2020.

[BPP16] André MS Barreto, Doina Precup, and Joelle Pineau. Practical kernel-based reinforcement learning. *The Journal of Machine Learning Research* 17.1 (2016), pp. 2372–2441.

[Bar+19] Peter L Bartlett, Victor Gabillon, Jennifer Healey, and Michal Valko. Scale-free adaptive planning for deterministic dynamics & discounted rewards. In *International Conference on Machine Learning*. 2019.

[BBS95] Andrew G Barto, Steven J Bradtke, and Satinder P Singh. Learning to act using real-time dynamic programming. *Artificial intelligence* 72.1-2 (1995), pp. 81–138.

[BN03] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation* 15.6 (2003), pp. 1373–1396.

[Bel+16] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in Neural Information Processing Systems* (2016).

[Ber+17] Felix Berkenkamp, Matteo Turchetta, Angela P. Schoellig, and Andreas Krause. Safe Model-Based Reinforcement Learning with Stability Guarantees. In *Advances in Neural Information Processing Systems*. 2017.

[Ber11] Dimitri P Bertsekas. Dynamic programming and optimal control 3rd edition, volume II. *Belmont, MA: Athena Scientific* (2011).

[BGZ14] Omar Besbes, Yonatan Gur, and Assaf Zeevi. Stochastic multi-armed-bandit problem with non-stationary rewards. In *Advances in Neural Information Processing Systems*. 2014.

[Bub+11] Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvári. X-armed bandits. *Journal of Machine Learning Research* 12 (2011), pp. 1587–1627.

[BCB12] Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning* (2012).

[BM10] Sébastien Bubeck and Rémi Munos. Open-loop optimistic planning. In *Conference on Learning Theory*. 2010.

[Bur+19] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*. 2019.

[BK97] Apostolos N Burnetas and Michaël N Katehakis. Optimal adaptive policies for Markov decision processes. *Mathematics of Operations Research* (1997).

[BM12] Lucian Buşoniu and Rémi Munos. Optimistic planning for Markov decision processes. In *International Conference on Artificial Intelligence and Statistics*. 2012.

[CK20] Tongyi Cao and Akshay Krishnamurthy. Provably adaptive reinforcement learning in metric spaces. In *Advances in Neural Information Processing Systems*. 2020.

[Che+19] Yifang Chen, Chung-Wei Lee, Haipeng Luo, and Chen-Yu Wei. A new algorithm for non-stationary contextual bandits: Efficient, optimal and parameter-free. In *Conference on Learning Theory*. 2019.

[CSLZ20] Wang Chi Cheung, David Simchi-Levi, and Ruihao Zhu. Reinforcement Learning for Non-Stationary Markov Decision Processes: The Blessing of (More) Optimism. In *International Conference on Machine Learning*. 2020.

[CYZ00] Samuel PM Choi, Dit-Yan Yeung, and Nevin L Zhang. Hidden-mode markov decision processes for nonstationary sequential decision making. In *Sequence Learning*. Springer, 2000, pp. 264–287.

[CG19] Sayak Ray Chowdhury and Aditya Gopalan. Online Learning in Kernelized Markov Decision Processes. In *International Conference on Artificial Intelligence and Statistics*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. 2019.

[CO20] Sayak Ray Chowdhury and Rafael Oliveira. No-Regret Reinforcement Learning with Value Function Approximation: a Kernel Embedding Approach. *arXiv preprint arXiv:2011.07881* (2020).

[CMP17] Richard Combes, Stefan Magureanu, and Alexandre Proutiere. Minimal exploration in structured stochastic bandits. In *Advances in Neural Information Processing Systems*. 2017.

[CM07] Pierre-Arnaud Coquelin and Rémi Munos. Bandit algorithms for tree search. In *Uncertainty in Artificial Intelligence*. 2007.

[CM08] Balázs Csanád Csáji and László Monostori. Value function based reinforcement learning in changing Markovian environments. *Journal of Machine Learning Research* 9.Aug (2008), pp. 1679–1709.

[DB15] Christoph Dann and Emma Brunskill. Sample complexity of episodic fixed-horizon reinforcement learning. In *Advances in Neural Information Processing Systems*. 2015.

[DLB17] Christoph Dann, Tor Lattimore, and Emma Brunskill. Unifying PAC and regret: Uniform PAC bounds for episodic reinforcement learning. In *Advances in Neural Information Processing Systems*. 2017.

[Dan+19] Christoph Dann, Lihong Li, Wei Wei, and Emma Brunskill. Policy Certificates: Towards Accountable Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning*. 2019.

[DGS14] Travis Dick, Andras Gyorgy, and Csaba Szepesvari. Online learning in markov decision processes with changing cost sequences. In *International Conference on Machine Learning*. PMLR. 2014, pp. 512–520.

[Din+21] Dongsheng Ding, Xiaohan Wei, Zhuoran Yang, Zhaoran Wang, and Mihailo Jovanovic. Provably Efficient Safe Exploration via Primal-Dual Policy Optimization. In *International Conference on Artificial Intelligence and Statistics*. 2021.

[Dom+21a] Omar Darwiche Domingues, Yannis Flet-Berliac, Edouard Leurent, Pierre Ménard, Xuedong Shang, and Michal Valko. *rlberry - A Reinforcement Learning Library for Research and Education*. 2021.

[Dom+21b] Omar Darwiche Domingues, Pierre Ménard, Emilie Kaufmann, and Michal Valko. Episodic Reinforcement Learning in Finite MDPs: Minimax Lower Bounds Revisited. In *International Conference on Algorithmic Learning Theory*. 2021.

[Dom+21c] Omar Darwiche Domingues, Pierre Ménard, Matteo Pirotta, Emilie Kaufmann, and Michal Valko. A Kernel-Based Approach to Non-Stationary Reinforcement Learning in Metric Spaces. In *International Conference on Artificial Intelligence and Statistics*. 2021.

[Dom+21d] Omar Darwiche Domingues, Pierre Ménard, Matteo Pirotta, Emilie Kaufmann, and Michal Valko. Kernel-Based Reinforcement Learning: A Finite-Time Analysis. In *International Conference on Machine Learning*. 2021.

[Dom+21e] Omar Darwiche Domingues, Corentin Tallec, Rémi Munos, and Michal Valko. Density-Based Bonuses on Learned Representations for Reward-Free Exploration in Deep Reinforcement Learning. In *ICML 2021 Workshop on Unsupervised Reinforcement Learning*. 2021.

[Du+20] Simon S. Du, Sham M. Kakade, Ruosong Wang, and Lin F. Yang. Is a Good Representation Sufficient for Sample Efficient Reinforcement Learning? In *International Conference on Learning Representations*. 2020.

[Efr+19] Yonathan Efroni, Nadav Merlis, Mohammad Ghavamzadeh, and Shie Mannor. Tight regret bounds for model-based reinforcement learning with greedy policies. In *Advances in Neural Information Processing Systems*. 2019.

[EGW05] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research* (2005).

[EDKM09] Eyal Even-Dar, Sham M Kakade, and Yishay Mansour. Online Markov decision processes. *Mathematics of Operations Research* 34.3 (2009), pp. 726–736.

[FD14] Zohar Feldman and Carmel Domshlak. Simple regret optimization in online planning for Markov decision processes. *Journal of Artificial Intelligence Research* (2014).

[Fie94] Claude-Nicolas Fiechter. Efficient Reinforcement Learning. In *Conference on Computational Learning Theory*. 1994.

[FB+21] Yannis Flet-Berliac, Johan Ferret, Olivier Pietquin, Philippe Preux, and Matthieu Geist. Adversarially Guided Actor-Critic. In *International Conference on Learning Representations*. 2021.

[FCL18] Lior Fox, Leshem Choshen, and Yonatan Loewenstein. DORA The Explorer: Directed Outreaching Reinforcement Action-Selection. In *International Conference on Learning Representations*. 2018.

[FCRL17] Justin Fu, John Co-Reyes, and Sergey Levine. EX2: Exploration with Exemplar Models for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems*. 2017.

[GOA18] Pratik Gajane, Ronald Ortner, and Peter Auer. A sliding-window algorithm for markov decision processes with arbitrarily changing rewards and transitions. *arXiv preprint arXiv:1805.10066* (2018).

[GM11] A. Garivier and E. Moulines. On Upper-Confidence Bound Policies For Switching Bandit Problems. In *International Conference on Algorithmic Learning Theory*. 2011.

[GMS19] Aurélien Garivier, Pierre Ménard, and Gilles Stoltz. Explore first, exploit next: The true shape of regret in bandit problems. *Mathematics of Operations Research* (2019).

[GSP19] Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. In *International Conference on Machine Learning*. 2019.

[Gri+19] Jean-Bastien Grill, Omar Darwiche Domingues, Pierre Ménard, Rémi Munos, and Michal Valko. Planning in entropy-regularized Markov decision processes and games. In *Neural Information Processing Systems*. 2019.

[GVM16] Jean-Bastien Grill, Michal Valko, and Rémi Munos. Blazing the trails before beating the path: Sample-efficient Monte-Carlo planning. In *Neural Information Processing Systems*. 2016.

[Guo+21] Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Alaa Saade, Shantanu Thakoor, Bilal Piot, Bernardo Avila Pires, et al. Geometric entropic exploration. *arXiv preprint arXiv:2101.02055* (2021).

[Guo+20] Zhaohan Daniel Guo, Bernardo Avila Pires, Bilal Piot, Jean-Bastien Grill, Florent Altché, Rémi Munos, et al. Bootstrap latent-predictive representations for multitask reinforcement learning. In *International Conference on Machine Learning*. 2020.

[Haa+17] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning*. 2017.

[Haa+18] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*. 2018.

[Haz+19] Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *International Conference on Machine Learning*. PMLR. 2019, pp. 2681–2691.

[HM08] Jean-Francois Hren and Rémi Munos. Optimistic planning of deterministic systems. In *European Workshop on Reinforcement Learning*. 2008.

[Hua+17] Ruitong Huang, Mohammad M. Ajallooeian, Csaba Szepesvári, and Martin Müller. Structured best-arm identification with fixed confidence. In *Algorithmic Learning Theory*. 2017.

[JOA10] Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research* (2010).

[Jia+19] Qian Jian, Ronan Fruit, Matteo Pirotta, and Alessandro Lazaric. Exploration Bonus for Regret Minimization in Discrete and Continuous Average Reward MDPs. In *Advances in Neural Information Processing Systems*. 2019.

[Jia+17] Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E. Schapire. Contextual Decision Processes with low Bellman rank are PAC-Learnable. In *International Conference on Machine Learning*. 2017.

[Jin+18] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is Q-Learning Provably Efficient? In *Advances in Neural Information Processing Systems*. 2018.

[Jin+20a] Chi Jin, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu. Reward-Free Exploration for Reinforcement Learning. In *International Conference on Machine Learning*. 2020.

[Jin+20b] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I. Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*. 2020.

[Jon+20] Anders Jonsson, Emilie Kaufmann, Pierre Menard, Omar Darwiche Domingues, Edouard Leurent, and Michal Valko. Planning in Markov Decision Processes with Gap-Dependent Sample Complexity. In *Advances in Neural Information Processing Systems*. 2020.

[Kak03] Sham Kakade. On the Sample Complexity of Reinforcement Learning. PhD thesis. University College London, 2003.

[KKL03] Sham Kakade, Michael J Kearns, and John Langford. Exploration in metric state spaces. In *International Conference on Machine Learning*. 2003.

[KK17] Emilie Kaufmann and Wouter M Koolen. Monte-carlo tree search by best-arm identification. In *Neural Information Processing Systems*. 2017.

[Kau+21] Emilie Kaufmann, Pierre Ménard, Omar Darwiche Domingues, Anders Jonsson, Edouard Leurent, and Michal Valko. Adaptive Reward-Free Exploration. In *International Conference on Algorithmic Learning Theory*. 2021.

[KMN02] Michael Kearns, Yishay Mansour, and Andrew Y Ng. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine learning* 49.2 (2002), pp. 193–208.

[KS99] Michael Kearns and Satinder Singh. Finite-Sample Convergence Rates for Q-Learning and Indirect Algorithms. In *Advances in Neural Information Processing Systems*. 1999.

[KS02] Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning* 49.2-3 (2002), pp. 209–232.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[KSU19] Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Bandits and experts in metric spaces. *Journal of the ACM (JACM)* 66.4 (2019), pp. 1–77.

[KBP13] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* (2013).

[KS06a] L. Kocsis and C. Szepesvári. Discounted UCB. In *2nd PASCAL Challenges Workshop*. 2006.

[KS06b] Levente Kocsis and Csaba Szepesvári. Bandit-based Monte-Carlo planning. In *European Conference on Machine Learning*. 2006.

[KT12] Branislav Kveton and Georgios Theocharous. Kernel-based reinforcement learning on representative states. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*. 2012.

[LOR15] Kailasam Lakshmanan, Ronald Ortner, and Daniil Ryabko. Improved regret bounds for undiscounted continuous reinforcement learning. In *International Conference on Machine Learning*. 2015.

[LH12] Tor Lattimore and Marcus Hutter. PAC bounds for discounted MDPs. In *International Conference on Algorithmic Learning Theory*. 2012.

[LHS+13] Tor Lattimore, Marcus Hutter, Peter Sunehag, et al. The sample-complexity of general reinforcement learning. In *Proceedings of the 30th International Conference on Machine Learning*. Journal of Machine Learning Research. 2013.

[LS20] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[LSW20] Tor Lattimore, Csaba Szepesvári, and Gellert Weisz. Learning with good feature representations in bandits and in rl with a generative model. In *International Conference on Machine Learning*. 2020.

[LR19] Erwan Lecarpentier and Emmanuel Rachelson. Non-Stationary Markov Decision Processes, a Worst-Case Approach using Model-Based Reinforcement Learning. In *Advances in Neural Information Processing Systems*. 2019.

[LM19] Edouard Leurent and Odalric-Ambrym Maillard. Practical open-loop pptimistic planning. In *European Conference on Machine Learning*. 2019.

[Lev+20] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643* (2020).

[LL19] Yingying Li and Na Li. Online Learning for Markov Decision Processes in Non-stationary Environments: A Dynamic Regret Analysis. In *2019 American Control Conference (ACC)*. IEEE. 2019, pp. 1232–1237.

[LSS15] Elad Liebman, Maytal Saar-Tsechansky, and Peter Stone. DJ-MC: A Reinforcement-Learning Agent for Music Playlist Recommendation. In *International Conference on Autonomous Agents and Multiagent Systems*. 2015.

[LA12] Shiau Hong Lim and Peter Auer. Autonomous Exploration For Navigating In MDPs. In *Conference on Learning Theory*. 2012.

[Lyk+19] Thodoris Lykouris, Max Simchowitz, Aleksandrs Slivkins, and Wen Sun. Corruption robust exploration in episodic reinforcement learning. *arXiv preprint arXiv:1911.08689* (2019).

[MH08] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research* 9.11 (2008).

[MBB20] Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. Count-Based Exploration with the Successor Representation. *Proceedings of the AAAI Conference on Artificial Intelligence* (2020).

[MT04] Shie Mannor and John N Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research* (2004).

[Mao+21] Weichao Mao, Kaiqing Zhang, Ruihao Zhu, David Simchi-Levi, and Tamer Basar. Near-Optimal Model-Free Reinforcement Learning in Non-Stationary Episodic MDPs. In *International Conference on Machine Learning*. 2021.

[Mei+21] Eli Meirom, Haggai Maron, Shie Mannor, and Gal Chechik. Controlling Graph Dynamics with Reinforcement Learning and Graph Neural Networks. In *International Conference on Machine Learning*. 2021.

[Mni+16] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, et al. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*. 2016.

[Mni+13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, et al. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[MJR15] Shakir Mohamed and Danilo Jimenez Rezende. Variational Information Maximisation for Intrinsically Motivated Reinforcement Learning. In *Advances in Neural Information Processing Systems*. 2015.

[MA12] Teodor Mihai Moldovan and Pieter Abbeel. Safe Exploration in Markov Decision Processes. In *International Coference on International Conference on Machine Learning*. 2012.

[MS08] Rémi Munos and Csaba Szepesvári. Finite-Time Bounds for Fitted Value Iteration. *Journal of Machine Learning Research* 9.5 (2008).

[Mé+21a] Pierre Ménard, Omar Darwiche Domingues, Anders Jonsson, Emilie Kaufmann, Edouard Leurent, and Michal Valko. Fast active learning for pure exploration in reinforcement learning. In *International Conference on Machine Learning*. 2021.

[Mé+21b] Pierre Ménard, Omar Darwiche Domingues, Xuedong Shang, and Michal Valko. UCB Momentum Q-learning: Correcting the bias without forgetting. In *International Conference on Machine Learning*. 2021.

[Neu+13] Gergely Neu, András György, Csaba Szepesvari, and Andras Antos. Online markov decision processes under bandit feedback. *IEEE Transactions on Automatic Control* 59.3 (2013), pp. 676–691.

[NJG17] Gergely Neu, Anders Jonsson, and Vicenç Gómez. A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798* (2017).

[NYW19] Chengzhuo Ni, Lin F Yang, and Mengdi Wang. Learning to Control in Metric Space with Optimal Regret. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE. 2019, pp. 726–733.

[OPT18] Jungseul Ok, Alexandre Proutiere, and Damianos Tranos. Exploration in structured reinforcement learning. In *Advances in Neural Information Processing Systems*. 2018.

[OS02] Dirk Ormoneit and Śaunak Sen. Kernel-Based Reinforcement Learning. *Machine Learning* (2002).

[OGA19] Ronald Ortner, Pratik Gajane, and Peter Auer. Variational Regret Bounds for Reinforcement Learning. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*. 2019.

[OR12] Ronald Ortner and Daniil Ryabko. Online regret bounds for undiscounted continuous reinforcement learning. In *Advances in Neural Information Processing Systems*. 2012, pp. 1763–1771.

[OAC18] Ian Osband, John Aslanides, and Albin Cassirer. Randomized Prior Functions for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems*. 2018.

[Osb+16] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep Exploration via Bootstrapped DQN. In *Advances in Neural Information Processing Systems*. 2016.

[ORVR13] Ian Osband, Daniel Russo, and Benjamin Van Roy. (More) efficient reinforcement learning via posterior sampling. In *Advances in Neural Information Processing Systems*. 2013.

[OVR14] Ian Osband and Benjamin Van Roy. Model-based reinforcement learning and the eluder dimension. In *Advances in Neural Information Processing Systems*. 2014.

[Ost+17] Georg Ostrovski, Marc G Bellemare, Aäron Oord, and Rémi Munos. Count-based exploration with neural density models. In *International Conference on Machine Learning*. 2017.

[Pas+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*. 2019.

[Pat+17] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*. 2017.

[PP13] Jason Pazis and Ronald Parr. PAC optimal exploration in continuous space Markov decision processes. In *AAAI Conference on Artificial Intelligence*. 2013.

[Ped+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[Pen+19] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177* (2019).

[Put94] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994.

[Raf+21] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research* (2021).

[Ren+21] Tongzheng Ren, Tianjun Zhang, Csaba Szepesvári, and Bo Dai. A Free Lunch from the Noise: Provable and Practical Exploration for Representation Learning. *arXiv e-prints* (2021).

[Rie05] Martin Riedmiller. Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*. 2005.

[RVC19] Yoan Russac, Claire Vernade, and Olivier Cappé. Weighted Linear Bandits for Non-Stationary Environments. In *Advances in Neural Information Processing Systems*. 2019.

[Sch91] Jürgen Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*. 1991, pp. 222–227.

[SCA17] John Schulman, Xi Chen, and Pieter Abbeel. Equivalence between policy gradients and soft Q-learning. In *arXiv:1704.06440*. 2017.

[Sin+20] Sean Sinclair, Tianyu Wang, Gauri Jain, Siddhartha Banerjee, and Christina Yu. Adaptive Discretization for Model-Based Reinforcement Learning. In *Advances in Neural Information Processing Systems*. 2020.

[SBY19] Sean R Sinclair, Siddhartha Banerjee, and Christina Lee Yu. Adaptive Discretization for Episodic Reinforcement Learning in Metric Spaces. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* (2019).

[SY94] Satinder P Singh and Richard C Yee. An upper bound on the loss from approximate optimal-value functions. *Machine Learning* 16.3 (1994), pp. 227–233.

[Sli14] Aleksandrs Slivkins. Contextual bandits with similarity information. *Journal of Machine Learning Research* 15.1 (2014), pp. 2533–2568.

[SS19] Zhao Song and Wen Sun. Efficient model-free reinforcement learning in metric spaces. *arXiv preprint arXiv:1905.00475* (2019).

[SP12] Susanne Still and Doina Precup. An information-theoretic approach to curiosity-driven reinforcement learning. *Theory in Biosciences* 131.3 (2012), pp. 139–148.

[SLL09] Alexander L. Strehl, Lihong Li, and Michael L. Littman. Reinforcement Learning in Finite MDPs: PAC Analysis. *Journal of Machine Learning Research* (2009).

[SL08] Alexander L. Strehl and Michael L. Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences* (2008).

[Str00] Malcolm Strens. A Bayesian framework for reinforcement learning. In *International Conference on Machine Learning*. 2000.

[Sui+15] Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe Exploration for Optimization with Gaussian Processes. In *International Conference on Machine Learning*. 2015.

[Sze10] Csaba Szepesvári. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning* 4.1 (2010), pp. 1–103.

[STL02] István Szita, Bálint Takács, and András Lörincz. ε-MDPs: Learning in varying environments. *Journal of Machine Learning Research* 3.Aug (2002), pp. 145–174.

[SKM14] Balázs Szörényi, Gunnar Kedenburg, and Rémi Munos. Optimistic planning in Markov decision processes using a generative model. In *Neural Information Processing Systems*. 2014.

[Tan+17] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, et al. # Exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in Neural Information Processing Systems*. 2017.

[Tar+21a] Jean Tarbouriech, Omar Darwiche Domingues, Pierre Ménard, Matteo Pirotta, Michal Valko, and Alessandro Lazaric. Adaptive Multi-Goal Exploration. *arXiv preprint arXiv:2111.12045* (2021).

[Tar+20a] Jean Tarbouriech, Matteo Pirotta, Michal Valko, and Alessandro Lazaric. Improved Sample Complexity for Incremental Autonomous Exploration in MDPs. In *Advances in Neural Information Processing Systems*. 2020.

[Tar+20b] Jean Tarbouriech, Matteo Pirotta, Michal Valko, and Alessandro Lazaric. Reward-free exploration beyond finite-horizon. In *ICML 2020 Workshop on Theoretical Foundations of Reinforcement Learning*. 2020.

[Tar+21b] Jean Tarbouriech, Matteo Pirotta, Michal Valko, and Alessandro Lazaric. Sample Complexity Bounds for Stochastic Shortest Path with a Generative Model. In *International Conference on Algorithmic Learning Theory*. 2021.

[Tar+20c] Jean Tarbouriech, Shubhanshu Shekhar, Matteo Pirotta, Mohammad Ghavamzadeh, and Alessandro Lazaric. Active Model Estimation in Markov Decision Processes. In *Conference on Uncertainty in Artificial Intelligence* (*UAI*). 2020.

[TTB20] Ahmed Touati, Adrien Ali Taiga, and Marc G. Bellemare. Zooming for Efficient Model-Free Reinforcement Learning in Metric Spaces. *arXiv e-prints* (2020).

[TV20] Ahmed Touati and Pascal Vincent. Efficient learning in non-stationary linear Markov decision processes. *arXiv preprint arXiv:2010.12870* (2020).

[TBK16] Matteo Turchetta, Felix Berkenkamp, and Andreas Krause. Safe exploration in finite markov decision processes with gaussian processes. *Advances in Neural Information Processing Systems* (2016).

[VMS16] René Vidal, Yi Ma, and S Shankar Sastry. *Generalized principal component analysis*. Vol. 5. Springer, 2016.

[Wac+18] Akifumi Wachi, Yanan Sui, Yisong Yue, and Masahiro Ono. Safe exploration and optimization of constrained mdps using gaussian processes. In *AAAI Conference on Artificial Intelligence*. 2018.

[WGL10] Thomas J Walsh, Sergiu Goschin, and Michael L Littman. Integrating sample-based planning and model-based reinforcement learning. *AAAI Conference on Artificial Intelligence* (2010).

[WAS21] Gellert Weisz, Philip Amortila, and Csaba Szepesvári. Exponential lower bounds for planning in mdps with linearly-realizable optimal action-value functions. In *Algorithmic Learning Theory*. 2021.

[Yan+20] Zhuoran Yang, Chi Jin, Zhaoran Wang, Mengdi Wang, and Michael Jordan. Provably Efficient Reinforcement Learning with Kernel and Neural Function Approximations. *Advances in Neural Information Processing Systems* (2020).

[YBW21] Ming Yin, Yu Bai, and Yu-Xiang Wang. Near-Optimal Provable Uniform Convergence in Offline Policy Evaluation for Reinforcement Learning. In *International Conference on Artificial Intelligence and Statistics*. 2021.

[Yu+21] Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys* (*CSUR*) (2021).

[YM09] Jia Yuan Yu and Shie Mannor. Online learning in Markov decision processes with arbitrarily changing rewards and transitions. In *International conference on game theory for networks*. IEEE. 2009.

[ZMS20] Xuezhou Zhang, Yuzhe Ma, and Adish Singla. Task-agnostic Exploration in Reinforcement Learning. In *Advances in Neural Information Processing Systems*. 2020.

[ZKZ09] Yufan Zhao, Michael R Kosorok, and Donglin Zeng. Reinforcement learning design for cancer clinical trials. *Statistics in medicine* (2009).