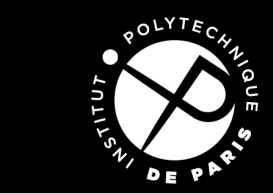
Constrained Exploration in Reinforcement Learning

Evrard Garcelon



INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2022IPPAG007

Thèse de doctorat

Constrained Exploration in Reinforcement Learning

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à l'École nationale de la statistique et de l'administration économique

École doctorale n°574 École doctorale de mathématiques Hadamard (EDMH)
Spécialité de doctorat : Mathématiques appliquées

Thèse présentée et soutenue à Palaiseau, le 08 Décembre 2022, par

EVRARD GARCELON

Composition du Jury :

Aurélien Garivier Professeur, ENS Lyon	Président, Rapporteur
Branislav Kveton Principal Scientist, Amazon	Rapporteur
Aurélien Bellet Chargé de recherche, Inria & Université de Lille	Examinateur
Azadeh Khaleghi Professeure, ENSAE (CREST)	Examinatrice
Vianney Perchet Professeur, ENSAE (CREST)	Directeur de thèse
Matteo Pirotta Research Scientist, Meta	Invité

Remerciements

First and foremost I wish to thank both my advisors Vianney Perchet and Matteo Pirotta. You gave me the opportunity to start to this 3-years life-changing adventure. Your scientific and human support significantly improved the quality of this thesis. I owe you both a lot. Vianney as I said during my defense since we met in 2017, under your mentorship I could grew up immensly technically but also as a human being and for that I want to sincerely thank you. Matteo, we met a bit later in 2019, you agreed to supervise my master's internship and later my PhD. The opportunity coupled with your help you gave me really is a positive EV event on my life and for that I can simply thank you sincerely.

I also want to thank the researchers that agreed to be part of my jury: Aurélien Garivier, Branislav Kveton, Azadeh Khaleghi and Aurélien Bellet. I am humbled that such researchers agreed to being part of my thesis jury, a very time-consuming task.

Over the last 3 years, I had the opportunity to collaborate with a lot of researchers from Meta AI and elsewhere. Every time it was a wonderful experience. I want to thank: Alessandro Lazaric, Ciara Pike-Burke, Michal Valko, Olivier Teytaud, Mohammad Ghavamzadeh, Remy Degenne, Paul Luyo, Kamalika Chaudhuri, Vashist Avadhanula, Yunchang Yang, Tianhao Wu, Han Zhong, Liwei Wang, Simon Shaolei Du, François Charton, Kristin Lauter, Mark Tygert, Cathy Li, Mingjie Chen and Emily Wenger.

During this PhD, I had the chance to meet some phenomenal people that made this PhD much easier. Either at ENSAE with Maria, Mathieu, Flore, Côme, Hugo, Lorenzo, Sasila, Reda, Corentin, Etienne, Firas. Going to ENSAE was sometimes long but always a pleasure to chat with each of you. Or at Meta, with Charlotte, Gautier, Lina, Louis, Leonard, Guillaume, Paul-Ambroise, Virginie, Hubert, Rui, Baptiste, Laurent. Again discussing with each of you was always a pleasure. I hope that someday I could cross paths with all of you again. Of course I can not finish this section without mentionning Jean-Baptiste, Pierre-Alexandre and Jean. You know why I thank each of you I already had the occasion to tell in person. Without you, those last few years would have been dramatically different.

Enfin je veux remercier ma famille ceux qui sont encore avec nous et ceux qu'on a malheureusement perdu au fil des années. En particulier, je veux remercier mes parents à qui je dois beaucoup si ce n'est quasiment tout. Simplement merci. Je tiens à terminer ces remerciements en remerciant Yanqi. Sans toi, je n'aurais tout simplement pas pu terminer ce doctorat.

Abstract

Today, most personalized services like recommendations are powered by supervised learning algorithms. But those algorithms use data collected thanks to the same applications and algorithms. This kind of feedback loop is reminiscent of an online learning problem. Surprisingly, online learning algorithms are not often used in applications like recommender system. Amongst online learning algorithms Reinforcement Learning (RL) and bandits algorithms, look the most well positioned to replace supervised learning algorithms (Netflix; Spotify; Li et al., 2016). Nonetheless, there exists a significant number of roadblocks before being able to fully replace current supervised methods. In this thesis, we focus on some of those constraints, from a theoretical point of view, to better understand some of the limitations of RL and bandit algorithms under those constraints and how it impacts their use in real-world applications. In this thesis, we consider three hurdles of deploying RL algorithms. But others exist such as model misspecification, non-stationarity and lack of real-time architectures.

The first is to guarantee that during the learning process of the newly-deployed algorithm, its performance does not fall significantly behind the performance of the already used method, called baseline policy. The latter is often suboptimal. In this case, it is desirable to deploy a Reinforcement Learning algorithm that interacts with the system to learn a better/optimal policy *under the constraint* that during the learning process, the performance is almost never worse than the performance of the baseline itself.

Recently, an other important problematic that emerged in personalized services is the privacy of the data used by the algorithms. For example, we can think to the leakage of information through recommended items in a recommender system scenario. In RL, it is common that user data contain sensitive information that needs to be protected from third parties. Motivated by this observation, we study privacy in the context of finite-horizon Markov Decision Processes (MDPs) by requiring information to be obfuscated on the user side. We formulate this notion of privacy for RL by leveraging the local differential privacy (LDP) framework. We establish a lower bound for regret minimization in finite-horizon MDPs with LDP guarantees which shows that guaranteeing privacy has a multiplicative effect on the regret. This result shows that while LDP is an appealing notion of privacy, it makes the learning problem significantly more complex.

At last, we consider the problem of security in Reinforcement Learning systems. One aspect is to understand how adversarial attacks can affect RL systems. In many domains, malicious agents may have incentives to force a bandit algorithm into a desired behavior. For instance, an unscrupulous ad publisher may try to increase their own revenue at the expense of the advertisers; a seller may want to increase the exposure of their products, or thwart a competitor's advertising campaign. We show that a malicious agent can force a bandit algorithm to recommend any desired items $T - o(T)$ times over a horizon of T steps, while applying adversarial modifications to either rewards or contexts with a cumulative cost only growing logarithmically. The second aspect of security in RL is to build an end-to-end encrypted RL system. Indeed, a critical aspect of bandit methods is that they require to observe contexts –i.e., individual or group-level data– and rewards in order to solve the sequential problem. The deployment in industrial applications has increased interest in methods that preserve the users' data security. We introduce a secure bandit framework based on homomorphic encryption which allows computations using encrypted data. The algorithm *only* observes encrypted information (contexts and rewards) and has no ability to decrypt it. The security of the data from the RL algorithm is then guaranteed.

Contents

Chapter 1

Introduction

Contents

1.1 Overview

Supervised Learning has powered a substantial amount of applications (Sharma et al., 2017; Wang et al., 2016; Ghazanfar and Prugel-Bennett, 2010) where some might argue that online decision-making algorithms may be better adapted (Sikka et al., 2012; Netflix). However, in the recent years there has been a lot of efforts to adapt and deploy online decision-making algorithms for applications like recommendation problems (Spotify; Ie et al., 2019). At the forefront of this push there is Reinforcement Learning (RL) and Bandit algorithms[1] (Shi et al., 2018; Zhao et al., 2019; Guo et al., 2020). Indeed, those learning paradigms allow practitioners to take into account the lifetime value of a customer (Wang et al., 2019a), or other objectives that are a function of the previous interactions of a given user with a product. Nonetheless, this change is not painless. One crucial reason for this, that we focus on in this thesis, between Supervised Learning based algorithms and RL ones is *exploration*.

To understand what is exploration in Bandit and RL algorithms, let us consider the classical example, described in (Lattimore and Szepesvári, 2020), where one faces two different slots machines with different winning probabilities. You can play one of the two machines 10 times with the goal to maximize the total number of wins after 10 pulls. Imagine that after 6 totally random pulls you observe that the first machine has a higher winning rate than the second one. How does one should allocate the remaining pulls? Should one commit to pulling only the first machine or try the second machine a few more times? The first example exploits the current results and makes a greedy decision based on those. Whereas taking the second option means that one is still exploring the potential actions maybe assuming the current results are due to randomness. The tension between optimizing the number of winning pulls and gathering more data on the winning rate of each machine is dubbed, in the literature, the *explore-exploit* tradeoff. When deploying a RL algorithm, exploration can be problematic for different reasons. For example, even though it may lead to better long-term results, exploration can lead to worse result in the short-term, as an example one may think about recommending a german-speaking (with no subtitles) movie to a solely french-speaking person. In the worst scenario, exploration may even be harmful when recommending a PG18-rated movie to a 13 years old. This motivates the need to constrain exploration when deploying RL algorithms.

This thesis is motivated by these questions around the exploration process for Reinforcement Learning and Bandits. We hope that through the different contributions presented here, we clarified some questions in order to apply RL algorithms in the real-world but also helped promising research directions to further facilitate the use of RL to emerge.

[1] In this thesis, we may sometimes use the term RL algorithms to design both Reinforcement Learning algorithms and Bandit algorithms

1.2 Contextual Bandit and Tabular Reinforcement Learning

Before delving into the contributions of this thesis, we provide an introduction to Tabular Reinforcement Learning and Contextual Bandits which are the two main settings studied in this thesis. We start by a brief introduction to Multi-Armed Bandit that is the basic setting to understand the interaction scenario between the learning algorithm and the rest of the world.

1.2.1 Multi-Armed Bandit

A Multi-Armed Bandit problem is a simple online decision-making problem. An algorithm (often called agent or learner) has to choose an action from a set of $K \in \mathbb{N}^\star$ possible actions for each time $t \geq 1$. Each action generates a reward, $r_{t,a}$ (that may be adversarial or stochastic) the objective of the learner is to maximize the cumulative rewards. In this thesis, we focus on stochastic bandit or Reinforcement Learning problem. That is to say, we assume that for each action $a \leq K$, the reward $r_a \sim \nu_a$ where ν_a is an *unknown* (for the learner) distribution on $\mathbb{R}$. The objective of the learner is then to maximize the cumulative reward, $\sum_t \mathbb{E}_{r\sim\nu_{a_t}}(r)$. Or equivalently to minimize the regret, $R(T) = \sum_{t=1}^T \mathbb{E}_{r\sim\nu_{a^\star}}(r) - \mathbb{E}_{r\sim\nu_{a_t}}(r)$ for all $T \geq 1$, where $a^\star := \arg\max_{a\in\{1,\dots,K\}} \mathbb{E}_{r\sim\nu_a}(r)$ is the best action to take with the knowledge of the distributions $(\nu_a)_a$.

As explained in the previous section, minimizing the regret requires the learner to balance exploring the arms and exploiting the current best arms. There exists a wide variety of assumptions that makes the multi-armed bandit problem more or less complex. For example, we can assume the reward distribution are sub-Gaussian or with heavy tails (Bubeck and Cesa-Bianchi, 2012; Zhuang and Sui, 2021). However, this provides a basic setting to understand the online nature of the problem studied in this thesis.

1.2.2 Linear Contextual Bandit

Contextual Bandit is a cardinal development of the standard bandit theory thanks to the addition of side-information called contexts that modify the reward-generating process enabling for example personalization in some application of bandit algorithms. In general in contextual bandit, the reward is a function of a feature given to the algorithm without any further assumption on the relationship between the reward and the feature (Beygelzimer et al., 2010, 2011; Bietti et al., 2018). In this work, we focus on the case where this relation is linear. The linear contextual bandit problem is one of the most studied version of the contextual bandit problem. There exists two slightly different formulation of the problem that are equivalent but are more comfortable depending on the situation.

Action Dependent Features Let's consider the standard contextual linear bandit setting with $K \in \mathbb{N}$ arms. At each time t, the agent observes a context $x_t \in \mathbb{R}^{d\times K}$, selects an action $a_t \in [\![1, K]\!]$ and observes a reward: $r_{t,a_t} = \langle \theta^\star, x_{t,a_t}\rangle + \eta^t_{a_t}$ where $\theta^\star \in \mathbb{R}^d$ is a feature vector and $\eta^t_{a_t}$ is a conditionally independent zero-mean, σ^2-subgaussian noise. No assumption is made on how the contexts are presented to the agent. That is to say they could be sampled stochastically or adversely. In most cases, the contexts are assumed to all have a norm upper-bounded by some known constant and the learner has access to an upper-bound on the norm of the unknown vector $\theta^\star$. The goal of the agent is to minimize the cumulative regret after T steps $R_T = \sum_{t=1}^T \langle \theta^\star, x_{t,a^\star_t}\rangle - \langle \theta^\star, x_{t,a_t}\rangle$, where $a^\star_t := \arg\max_a \langle \theta^\star, x_{t,a}\rangle$.

Action Independent Features In this formulation the features vector does depend on the action, that is to say at each time t, the agent receives a context $x_t \in \mathbb{R}^d$. The main difference with the formalism above is that when the agent selects an action $a_t \in [\![1, K]\!]$, it observes a reward: $r_{t,a_t} = \langle \theta_{a_t}, x_t\rangle + \eta^t_{a_t}$ where for each arm a, $\theta_a \in \mathbb{R}^d$ is a feature vector and $\eta^t_{a_t}$ is a conditionally independent zero-mean, σ^2-subgaussian noise. The regret can now be written as, $R_T = \sum_{t=1}^T \langle \theta_{a^\star_t}, x_t\rangle - \langle \theta_{a_t}, x_t\rangle$, where $a^\star_t := \arg\max_a \langle \theta_a, x_t\rangle$.

1.2.3 Reinforcement Learning

Reinforcement Learning is a generalization of the bandit setting by incorporating a notion of state, similar to contexts in contextual bandit. Over the years, different formulation of RL has been devloped. For example, most applied deep Reinforcement Learning research is taking place in the discounted infinite horizon setting (Mnih et al., 2013). In those work, we focus on two setting more studied in theoretical RL literature: finite-horizon (in Section 3.1 and Section 2.C.3) and Average-Reward Reinforcement Learning (in Section 2.3).

1.2.3.1 Finite-Horizon Reinforcement Learning

Let's consider a finite-horizon Markov Decision Process (Puterman, 1994, Chp. 4) $M = (\mathcal{S}, \mathcal{A}, p, r, H)$ with state space $\mathcal{S}$ and action space $\mathcal{A}$. Every state-action pair is characterized by a reward distribution with mean $r(s,a)$ and support in $[0,1]$ and a transition distribution $p(\cdot|s,a)$ over the next state. We denote by $S = |\mathcal{S}|$ and $A = |\mathcal{A}|$ the number of states and actions, and by H the horizon of an episode. A Markov randomized decision rule $d : \mathcal{S} \to P(\mathcal{A})$ maps states to distributions over actions. A policy π is a sequence of decision rules, i.e., $\pi = (d_1, d_2, \ldots, d_H)$. We denote by Π^{MR} (resp. Π^{MD}) the set of Markov randomized (resp. deterministic) policies. The value of a policy $\pi \in \Pi^{\text{MR}}$ is measured trough the value function

$$\forall t \in [H], \forall s \in \mathcal{S} \qquad V_t^\pi(s) = \mathbb{E}^\pi \left[\sum_{l=t}^{H} r_l(s_l, a_l) \mid s_t = s \right]$$

where the expectation is defined w.r.t. the model and policy (i.e., $a_l \sim d_l(s_l)$). This function gives the expected total reward that one could get by following policy π starting in state s, at time t. There exists an optimal policy $\pi^\star \in \Pi^{\text{MD}}$ (Puterman, 1994, Sec. 4.4) for which $V_t^\star = V_t^{\pi^\star}$ satisfies the *optimality equations*:

$$\forall t \in [H], \forall s \in \mathcal{S}, \qquad V_t^\star(s) = \max_{a \in \mathcal{A}} \left\{ r_t(s,a) + p(\cdot|s,a)^\mathsf{T} V_{t+1}^\star \right\} := L_t^\star V_t^\star \tag{1.1}$$

where $V_{H+1}^\star(s) = 0$ for any state $s \in \mathcal{S}$. The value function can be computed using backward induction (e.g., Puterman, 1994; Bertsekas, 1995) when the reward and transitions are known. Given a policy $\pi \in \Pi^{\text{MD}}$, the associated value function satisfies the *evaluation equations* $V_t^\pi(s) := L_t^\pi V_{t+1}^\pi(s) = r(s, d_t(s)) + p(\cdot|s, d_t(s))^\mathsf{T} V_{t+1}^\pi$. The optimal policy is thus defined as $\pi^\star = \arg\max_{\pi \in \Pi^{\text{MD}}} \{L_t^\pi V_t^\star\}$, $\forall t \in [H]$.

In the following we assume that the learning agent knows $\mathcal{S}$, $\mathcal{A}$ and $r_{\max}$, while the reward and dynamics are *unknown* and need to be estimated online. Given a finite number of episode K, we evaluate the performance of a learning algorithm $\mathfrak{A}$ by its cumulative regret

$$R(\mathfrak{A}, K) = \sum_{k=1}^{K} V_1^\star(s_{1,k}) - V_1^{\pi_k}(s_{1,k})$$

where π_k is the policy executed by the algorithm at episode k.

1.2.3.2 Average-Reward Reinforcement Learning

The Average-Reward setting in Reinforcement Learning is defined through a Markov Decision Process (Puterman, 1994, Sec. 8.3) $M = (\mathcal{S}, \mathcal{A}, p, r)$ with state space $\mathcal{S}$ and action space $\mathcal{A}$. Every state-action pair (s,a) is characterized by a reward distribution with mean $r(s,a)$ and support in $[0, r_{\max}]$, and a transition distribution $p(\cdot|s,a)$ over next states. We denote by $S = |\mathcal{S}|$ and $A = |\mathcal{A}|$ the number of states and actions. A stationary Markov randomized policy $\pi : \mathcal{S} \to P(\mathcal{A})$ maps states to distributions over actions. The set of stationary randomized (resp. deterministic) policies is denoted by Π^{SR} (resp. Π^{SD}). Any policy $\pi \in \Pi^{\text{SR}}$ has an associated *long-term average reward* (or gain) and a *bias function* defined as

$$g^\pi(s) := \lim_{T \to +\infty} \mathbb{E}_s^\pi \left[\frac{1}{T} \sum_{t=1}^{T} r(s_t, a_t) \right] \quad \text{and}$$

$$h^\pi(s) := \underset{T \to +\infty}{C\text{-}\lim} \, \mathbb{E}_s^\pi \left[\sum_{t=1}^{T} \left(r(s_t, a_t) - g^\pi(s_t) \right) \right],$$

where $\mathbb{E}_s^\pi$ denotes the expectation over trajectories generated starting from $s_1 = s$ with $a_t \sim \pi(s_t)$. The bias $h^\pi(s)$ measures the expected total difference between the reward and the stationary reward in *Cesaro-limit* (denoted by $C\text{-}\lim$). We denote by $sp(h^\pi) := \max_s h^\pi(s) - \min_s h^\pi(s)$ the *span* (or range) of the bias function.

Assumption 1. *The MDP M is* ergodic.

In ergodic MDPs, any policy $\pi \in \Pi^{\text{SR}}$ has *constant* gain, i.e., $g^\pi(s) = g^\pi$ for all $s \in \mathcal{S}$. There exists a policy $\pi^\star \in \arg\max_\pi g^\pi$ for which $(g^\star, h^\star) = (g^{\pi^\star}, h^{\pi^\star})$ satisfy the *optimality equations*,

$$h^\star(s) + g^\star = L h^\star(s) := \max_{a \in \mathcal{A}} \{ r(s,a) + p(\cdot|s,a)^\mathsf{T} h^\star \},$$

where L is the *optimal* Bellman operator. We use $D = \max_{s \neq s'} \min_{\pi \in \Pi^{\text{SD}}} \mathbb{E}[\tau_\pi(s'|s)]$ to denote the diameter of M, where $\tau_\pi(s'|s)$ is the hitting time of s' starting from s. We introduce the "worst-case" diameter

$$\Upsilon = \max_{s \neq s'} \max_{\pi \in \Pi^{\text{SD}}} \mathbb{E}\left[\tau_\pi(s'|s)\right], \tag{1.2}$$

which defines the worst-case time it takes for any policy π to move from any state s to s'. Asm. 1 guarantees that $D \leq \Upsilon < \infty$.

1.3 Outline and Contributions

In this thesis, we explore the theoretical implications of how one can reconciliate the exploration process in Reinforcement Learning and Bandits algorithms with practical constraints that at first glance would require to limit or even suppress exploration. In each of the following chapters, we explore one constraint showing its effect on the regret. The theoretical results are illustrated with experiments on either synthetic dataset and real-world datasets. The technical derivation are located in the Appendix at the end of each chapter.

Bandits with Noisy Evaluations and Conservative Exploration The first type of constraint, that is the focus of Section 2.1 of Chapter 2, can be seen as a performance type of constraint. In a standard recommendation ML pipeline it is not uncommon to have features being scored by a bunch of specialized models before being sent to the algorithm in charge of the final recommendation. The reason for this design are multiple. In some cases, it is simply due computational and latency constraints. For problems with millions of items to be recommended it is more efficient to get a rough ranking of all items and then computing more refined estimation of a reduced number of items. An other reason can be that the nature of objects to be ranked is heterogeneous that is to say the algorithm may have to rank images and text data. In which case, it is often recommended to use specialized scoring algorithms. The second performance constraint, studied in Sections 2.2 and 2.3 in Chapter 2 deals with the short-term reward loss due to exploration in RL and Bandits algorithms. As discussed above, due to exploration, RL algorithms takes action that are suboptimal in order to discard them quickly and focus on the other actions. However, oftentimes when deploying a RL algorithms there exists a trusted algorithm (which maybe relying on handcrafted rules) solving the same problem as the RL algorithm but potentially suboptimal. If this existing algorithm is indeed suboptimal, the RL algorithm will outperform it in the long run. In the worst case the gap in performance can be in serious disfavor of the RL algorithm for a long time. This situation is to be avoided at all cost and requires to control the exploration of the algorithm such that it is taking bad actions only if it has built a "budget" for it. That is to say, we want to introduce a parameter controlling the maximum difference between the performance of the existing algorithm and the RL one while the latter is performing worse than the former. This problem is called *conservative exploration*[2].

Privacy in Reinforcement Learning The second type of constraints we tackle is motivated by the recent push for more privacy in online services. Indeed, the question of privacy in machine learning algorithms has been a long standing question (Dwork et al., 2010a). Recently the notion of Differential Privacy has nonetheless been accepted as the default definition of privacy in the ML literature (Abadi et al., 2016). In Section 3.1 of Chapter 3, we examine how this definition has been adapted to the tabular RL setting. The main consequence of privacy in RL is a constraint on the sequence of actions that can be taken by the agent. We propose the first algorithm to enforce a stronger definition of privacy, a decentralized one (Bebensee, 2019), and show how this affect the regret by proving a lower bound and presenting an algorithm matching this lower-bound up to polynomial terms in the size of the state space and the size of the actions space. In addition, in Section 3.2 of Chapter 3 we show how to relate a recent advance in the theory of Differential Privacy (Feldman et al., 2020), which allows to achieve a trade-off between strong privacy and performance degradation, to a linear contextual bandit problem.

Security in Reinforcement Learning. Finally, the last type of constraint we consider in this thesis is a security constraint. In Chapter 4, we study two different aspects of security. First, in Section 4.1 we investigate how sensitive linear contextual bandits are to adversarial attacks. In those attacks, the attackers are allowed to modify the feedback sent to the algorithm. Their objective being to minimize the cumulative change in the feedback while ensuring that the bandit algorithm is unable to find the best actions. In Section 4.2, we explore a second aspect of security that

[2]For each type of constraints a more thorough related work discussion is provided in the corresponding chapter.

having all the feedback sent and received by the algorithm are encrypted end-to-end. The algorithm can not decrypt data but can perform computation on encrypted data. This encryption comes with serious computational drawbacks and limitations. That is to say, there is a limited number of additions or multiplications possible before making the data indecipherable. Despite this limitation, we show how to build a linear contextual bandit algorithm with a regret on par with standard linear contextual bandit algorithms.

List of Publications in International Conferences with Proceedings. The list below contains the publications I was involved in during this PhD. Chapter 2 is based on (**Garcelon** et al., 2022c, 2020a,b). Chapter 3 is based on (**Garcelon** et al., 2021, 2022b) and the last chapter, Chapter 4 is based on (**Garcelon** et al., 2020c, 2022c).

Publications presented in this thesis:

- **Evrard Garcelon**, Vashist Avadhanula, Alessandro Lazaric, and Matteo Pirotta. Top k ranking for multi-armed bandit with noisy evaluations. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 6242–6269. PMLR, 28–30 Mar 2022a. URL https://proceedings.mlr.press/v151/garcelon22b.html
- **Evrard Garcelon**, Mohammad Ghavamzadeh, Alessandro Lazaric, and Matteo Pirotta. Improved algorithms for conservative exploration in bandits. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04): 3962–3969, Apr. 2020a. doi: 10.1609/aaai.v34i04.5812. URL https://ojs.aaai.org/index.php/AAAI/article/view/5812
- **Evrard Garcelon**, Mohammad Ghavamzadeh, Alessandro Lazaric, and Matteo Pirotta. Conservative exploration in reinforcement learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1431–1441. PMLR, 26–28 Aug 2020b. URL https://proceedings.mlr.press/v108/garcelon20a.html
- **Evrard Garcelon**, Vianney Perchet, Ciara Pike-Burke, and Matteo Pirotta. Local differential privacy for regret minimization in reinforcement learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 10561–10573. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/
- **Evrard Garcelon**, Kamalika Chaudhuri, Vianney Perchet, and Matteo Pirotta. Privacy amplification via shuffling for linear contextual bandits. In Sanjoy Dasgupta and Nika Haghtalab, editors, *International Conference on Algorithmic Learning Theory, 29-1 April 2022, Paris, France*, volume 167 of *Proceedings of Machine Learning Research*, pages 381–407. PMLR, 2022b. URL https://proceedings.mlr.press/v167/garcelon22a.html
- **Evrard Garcelon**, Baptiste Roziere, Laurent Meunier, Jean Tarbouriech, Olivier Teytaud, Alessandro Lazaric, and Matteo Pirotta. Adversarial attacks on linear contextual bandits. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14362–14373. Curran Associates, Inc., 2020c. URL https://papers.nips.cc/paper/2020
- **Evrard Garcelon**, Matteo Pirotta, and Vianney Perchet. Encrypted linear contextual bandit. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 2519–2551. PMLR, 28–30 Mar 2022c. URL https://proceedings.mlr.press/v151/garcelon22a.html

Other publications not presented in this thesis:

- Rémy Degenne, **Evrard Garcelon**, and Vianney Perchet. Bandits with side observations: Bounded vs. logarithmic regret. In Amir Globerson and Ricardo Silva, editors, *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, pages 467–476. AUAI Press, 2018. URL http://auai.org/uai2018/proceedings/supplements/Supplementary-Paper182.pdf
- Jean Tarbouriech, **Evrard Garcelon**, Michal Valko, Matteo Pirotta, and Alessandro Lazaric. No-regret exploration in goal-oriented reinforcement learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9428–9437. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/tarbouriech20a.html

- Yunchang Yang, Tianhao Wu, Han Zhong, **Evrard Garcelon**, Matteo Pirotta, Alessandro Lazaric, Liwei Wang, and Simon Shaolei Du. A reduction-based framework for conservative bandits and reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=AcrlgZ9BKed

Chapter 2

Performance Constraint in Reinforcement Learning

In this chapter, we explore the impact of different empirical constraints on the exploration process in Reinforcement Learning and Bandit problems. There is a myriad of practical constraints that may hinder the exploration process in online learning. In this chapter, we explore two main constraints. The first constraint is when it is not possible for the learning algorithm to directly observe the input, x such that the reward is a function of x. That is to say the algorithm may only observe predictions of the reward from some prediction algorithms. For example, the learning algorithm only observes predictions from computer vision algorithms for image inputs, or NLP algorithms for text-based inputs. The second constraint we focus on is a performance constraint on the policy learned by the agent. That is to say, assuming the learning algorithm has access to a baseline policy (usually a policy based on expert knowledge) the agent is not allowed to deploy a policy in the environment that is performing significantly worse than the baseline policy. This problem is known in the bandit and RL literature as the *conservative exploration* problem.

Specifically, in this chapter we *formalize* for the first time the problem of *Bandits under Noisy Evaluations* where the agent choose arms based on noisy predictions provided by different evaluators. We define the notion of regret in this new setting and introduce two learning algorithms with sublinear regret. In the second part of this chapter, we revisit the problem of conservative exploration (Wu et al., 2016; Kazerouni et al., 2017). We introduce an algorithm with improved regret guarantees in the linear contextual bandit case. Finally, we formalize the conservative exploration problem in the Finite Horizon Reinforcement Learning but also in the Average Reward Reinforcement Learning setting.

This chapter is based on the three following articles:

- **Evrard Garcelon**, Vashist Avadhanula, Alessandro Lazaric, and Matteo Pirotta. Top k ranking for multi-armed bandit with noisy evaluations. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 6242–6269. PMLR, 28–30 Mar 2022a. URL https://proceedings.mlr.press/v151/garcelon22b.html
- **Evrard Garcelon**, Mohammad Ghavamzadeh, Alessandro Lazaric, and Matteo Pirotta. Improved algorithms for conservative exploration in bandits. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04): 3962–3969, Apr. 2020a. doi: 10.1609/aaai.v34i04.5812. URL https://ojs.aaai.org/index.php/AAAI/article/view/5812
- **Evrard Garcelon**, Mohammad Ghavamzadeh, Alessandro Lazaric, and Matteo Pirotta. Conservative exploration in reinforcement learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1431–1441. PMLR, 26–28 Aug 2020b. URL https://proceedings.mlr.press/v108/garcelon20a.html

Contents

2.1 Bandit With Noisy Evaluations

Consider an idealized content reviewing task in a large social media firm, where the objective is to identify harmful content that violates the platforms' community standards. Given the large volume of content generated on a daily basis, it may not be possible to ask human reviewers to provide a thorough assessment of each piece of content. For this reason, the platform may automatically assign a badness score for each piece of content depending on their estimated level of severity. For example, a hate speech related post may be assigned a higher badness score in comparison to a click bait post. The content with higher badness score may then be prioritized for human review, which eventually leads to what we can consider as a "ground-truth" evaluation of the severity of the content. The more accurate the badness score is in predicting the actual severity, the higher the chance that harmful content is passed for human review and properly identified. In practice, the badness score may be obtained by aggregating predictions returned by different automatic systems (e.g., rule-based, ML-based systems). For instance, the platform could rely on NLP-based classifiers for hostile speech detection, or CV-based classifiers for graphic images. As such, it is crucial to properly calibrate the predictions returned by each of these classifiers to ensure that the scores can be compared meaningfully and then return an aggregate and reliable badness score that correctly prioritizes the most harmful content for human review.

In this section, we consider the case where evaluators are noisy and possibly biased functions of the true reward of each arm. In particular, we consider two alternative settings, where evaluations are generated according to: **1)** a noisy generalized linear function of the true reward; **2)** a noisy linear function of the true reward. In both cases, we first define an "oracle" strategy that have prior knowledge of the evaluation function, including the noise distribution, and it is designed to maximize the rewards of the arms chosen at each round given the evaluations provided as input. We then devise the most suitable MAB strategies to approach the oracle's performance over time. In the first case, we show that one has to rely on an ϵ-greedy strategy to avoid dependencies between the evaluations observed over time and the decisions taken by the algorithm. This eventually leads to a regret w.r.t. the oracle of order $\widetilde{O}(T^{2/3})$ over T rounds. On the other hand, if the evaluation functions are linear and the variance of the additive noise is known, we show that a simple greedy strategy leveraging the specific structure of the problem is able to recover a $\widetilde{O}(\sqrt{T})$ regret. We then validate these results in a number of experiments. We first consider synthetic problems where we carefully design the MAB instances to support our theoretical findings and to compare to alternative approaches. Then we move to problems based on real data, where our assumptions may not be verified, to provide a more thorough evaluation of performance and robustness of our approach. Notably, we study a problem related to content review prioritization for integrity in social media platforms.

2.1.1 Definition of the Bandit Model

We consider a multi-armed bandit problem where, at each round t, the learner is provided with a set of $K_t > K \geq 1$ arms (e.g., the content to be reviewed at time t) characterized by a reward $r_{i,t} \in \mathbb{R}$ for each $i = 1, \dots, K_t$ (e.g., the badness score). While the true reward is unknown to the learner, J evaluators return noisy, possibly biased, evaluations $f_{i,t,j}$ for each arm (e.g., different rule-based and/or ML-classifiers). At the beginning of each round, the learner receives the evaluations $\{f_{i,t,j}\}$, it returns a set $\mathcal{A}_t \subseteq \{1, \dots, K_t\}$ of K arms (i.e., $|\mathcal{A}_t| = K$), and it observes their associated rewards $r_{i,t}$ for $i \in \mathcal{A}_t$.[1] The learner's objective is to accumulate as much reward as possible over T rounds by selecting the K arms with larger rewards.

Without any further assumption, this problem is not tractable since the rewards may change arbitrarily over time and the evaluations may not be predictive of the true rewards, thus making it impossible for any learner to achieve a satisfactory performance. Throughout the paper, we make a series of assumptions to make the problem solvable. We start from the rewards.

Assumption 2. *The rewards $r_{i,t}$ of each arm $i = 1, \dots, K_t$ at round $t = 1, \dots, T$ are drawn i.i.d. from a common distribution ν supported on $[0, C]$ with C a positive constant. The number of arms K_t at each round t is arbitrary and $K < K_t \leq K_{\max} < \infty$.*

While this assumption simplifies the treatment of the problem, it does not affect the objective of the learner, which is to select the top-K arms *at each round*, i.e., for the specific realizations $\{r_{i,t}\}$. For instance, for $K = 1$, the objective is to return the arm $i_t^\star = \arg\max_{i=1,\dots,K_t} r_{i,t}$. We do not assume that the learner has any prior knowledge of the distribution ν.

[1] For the sake of simplicity, we consider that the learner receives the exact reward $r_{i,t}$, but all our results can be adapted to the case of noisy feedback.

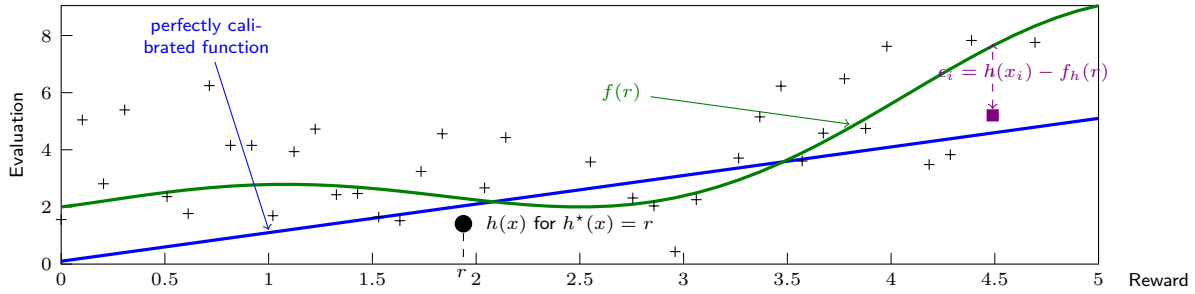


Figure 2.1: As an illustrative example, consider the case where at each round t each arm is associated to a context $x_{i,t}$ drawn from a context distribution ρ and there exists a function generating the true rewards as $r_{i,t} = h^\star(x_{i,t})$. The distribution ν is then defined by the distribution on rewards $r = h^\star(x)$ induced by $x \sim \rho$. We also denote by $\rho|h^\star(x) = r$ the conditional distribution over contexts associated with reward r. Consider then a neural network h, trained on past context-reward pairs, that returns an evaluation for arm i characterized by a context $x_{i,t}$ as $f_{i,t} = h(x_{i,t})$. The black crosses in the plot are the pairs $(h^\star(x_{i,t}), h(x_{i,t}))$. The evaluation function associated to the neural network h is then defined as $f_h(r) = \mathbb{E}_{x\sim\rho|h^\star(x)=r}\big[h(x)\big]$ (green line) and the noise ϵ is the deviation from $h(x)$ and $f(r)$ depending on the specific realization of x, i.e., $\epsilon_{i,t} = h(x_{i,t}) - f_h(r)$ for $r = h^\star(x_{i,t})$. The blue line illustrates the perfectly calibrated case, where $h^\star$ itself is used for prediction, in this case $f_{h^\star}(r) = r$.

In general, the evaluators may rely on some contextual information $x_{i,t}$ associated to each arm i (e.g., texts, images, meta-data related to the piece of content) to return their evaluation $f_{i,t,j}$ and their accuracy in predicting the true reward $r_{i,t}$ may vary depending on the evaluator and the specific context $x_{i,t}$. Nonetheless, we assume that the learner has no access to the context or the actual mechanism that generates the evaluations (e.g., the evaluators may be external services) and we rather rely on the following model to describe how the evaluations are generated

$$f_{i,t,j} = f_j(r_{i,t}) + \epsilon_{i,t,j}, \quad j = 1, \dots, J, \tag{2.1}$$

where $f_j : \mathbb{R} \to \mathbb{R}$ is the evaluation function, and $\epsilon_{i,t,j}$ is a stochastic error. This general formulation can be seen as the inverse of a calibration function, as it describes the intrinsic bias of each evaluator j and the noise associated to the evaluations of the true reward. See Fig. 2.1 for a qualitative illustration of this model. We assume the noise in the evaluations satisfy a rather mild assumption.[2]

Assumption 3. *Each error $\epsilon_{i,t,j}$ is generated i.i.d. from a sub-Gaussian distribution with zero mean and parameter σ_j, assumed to be known to the learner.*

As far as the evaluation function is concerned, we distinguish two settings.

Assumption 4 (Generalized linear setting)**.** *We assume that each evaluation function is a generalized linear model w.r.t. the true reward, i.e., $f_j(r) = g(\alpha_j \cdot r)$, for all $j \le J$, where $\alpha \in \mathbb{R}$ and g is a strictly increasing function and twice- differentiable with $\|g'\|_\infty \le L_g$ and $\|g''\|_\infty \le M_g$, and $c_g := \inf_{x,\theta} g'(x \cdot \theta) > 0$. The function g is known to the learner, while the evaluator-specific parameters α_j are unknown.*

Assumption 5 (Linear setting)**.** *We assume that each evaluation function is linear w.r.t. the true reward, i.e., $f_j(r) = \alpha_j r$, for all $j \le J$. While the shape of the function is known to the learner, the evaluator-specific parameters α_j are unknown.*

In the following we use $\alpha = (\alpha_1, \dots, \alpha_J) \in \mathbb{R}^J$ and $\sigma = (\sigma_1, \dots, \sigma_J) \in \mathbb{R}^J$. We use the standard notation $\|\cdot\|$ and $\|\cdot\|_\infty$ for the ℓ_2 and the maximum norm respectively, while for any two vectors $x, y \in \mathbb{R}^J$, $x \cdot y \in \mathbb{R}^J$ denotes the component-wise product.

We consider a setting where the bandit algorithm has only access to the predictors' evaluations of the true reward of an arm. This is a very generic scenario that encompasses the case where evaluations are a function of a context characterizing an arm. The generality of our framework allows us to deal with problems where the context is not directly observable (e.g., because it is kept private) or where it differs across evaluators. For example, similarly to bandits with expert advice, evaluators may use very different context sources (e.g., visual information, text, meta

[2] A similar assumption is used in (Yun et al., 2017), where the covariance matrix of the distribution generating the noisy features is assumed to be known to the learning.

data) to build their predictions, but these are unknown to the bandit algorithm (e.g., because evaluators are external services). The resulting model in Eq. 2.1 is then a calibration function which, in the case evaluations are function of a context, can be understood as explaining the connection between the true reward and the evaluations after averaging over the stochasticity in the (non-observable) context information (Fig. 2.1). Notice that if the context was available, the evaluations could be disregarded as the bandit could directly rely on the context to predict the rewards in the first place, as in standard contextual bandit.

We conclude by noticing, despite these additional assumptions, no learner can retrieve the best choice of top-K arms at each round (i.e., $\max_{i_1,\dots,i_K} \sum_{l=1}^{K} r_{i_l,t}$), since the only information available to the learner is from biased and noisy evaluators (see Lemma 7 in App. 2.A.1). As a result, instead of targeting the top-K arms, in the following we introduce *oracle* strategies that leverage the full knowledge of the problem (i.e., the evaluation function and the noise distribution) and use their performance as reference for the learner.

2.1.2 Other Related Bandit Models

The problem sketched before [3] can be seen as an instance of the multi-armed bandit (MAB) framework, where each piece of content is an arm and the objective of the bandit algorithm is to select arms/content with the higher reward (e.g., severity). The algorithm can rely on the estimations returned by a set of *evaluators* (e.g., a set of classifiers) to decide which arm to pull at each step (e.g., content to pass to human review). This setting can be formalized using a number of existing frameworks, such as MAB with expert advice, contextual bandit, bandit with side observation, and contextual bandit with noisy context. Before diving into how to solve the problem introduced here, we review alternative models that are related to our setting. Let consider the case with $K = 1$ (i.e., the learner returns one arm at each round). The most direct way to model our setting is MAB with expert advice (Auer et al., 2003), where the evaluators are *experts* and evaluations $\{f_{i,t,j}\}$ are the experts feedback. In this case, it is possible to derive algorithms with sublinear regret w.r.t. the best expert in hindsight (Beygelzimer et al., 2011). While this is a very general model, where no assumption is imposed either on the rewards or on the experts feedback (they could even be generated adversarially), algorithms designed for this setting tend to be over conservative in practice, as they have to be robust to any sort of data process. Furthermore, none of the evaluators may be very accurate (e.g., they all have very large variance) and targeting the performance of the best among them may not correspond to a satisfactory performance.

Alternatively, we can frame our problem as a contextual MAB problem (Agrawal and Goyal, 2013; Agarwal et al., 2014). We could aggregate all J evaluations for arm i into a context representation

$$\phi_{i,t} = (f_{i,t,1}, \dots, f_{i,t,j}, \dots, f_{i,t,J}) \in \mathbb{R}^J. \tag{2.2}$$

Unfortunately, there are two major issues using this model: **1)** in general, the reward $r_{i,t}$ may not be a simple function (e.g., linear) of $\phi_{i,t}$; **2)** the contextual features may be noisy realizations of some "true" features (e.g., due to the noise factor $\epsilon_{i,t,j}$ in Eq. (2.1)). In order to deal with the first issue, we could rely on Asm. 5, which would lead to a linear contextual problem. The second issue could be dealt by using the approach proposed by Yun et al. (2017) for linear contextual bandit with noisy features. While the setting in (Yun et al., 2017) bears some similarities (e.g., the noise distribution is assumed to be known, they consider a similar notion of relative regret and study a greedy algorithm), there remain some crucial differences: **1)** they consider unbiased features, which corresponds to a very specific instance of Asm. 5 with $\alpha_j = 1$; **2)** they provide guarantees only for Gaussian noise, while the algorithm designed to handle the general case has no regret guarantee.

Finally, alternative models of contextual bandit with non-deterministic features considered the case where the full distribution of the features is known (Yang et al., 2020) or part of the features are corrupted (Gajane et al., 2016; Bouneffouf, 2021). These settings do not match the use cases studied in this section. In addition, most other noise-in-the-features regression models such as Deming Regression (Walford and Deming, 1944) or Total Least Squares (Golub and Loan, 1980) does not take into account the relationship between the rewards and the features, that is to say the features are a function of the actual reward, this relationship creates dependence and correlation unaccounted for in the analysis of those regression models.

2.1.3 The Generalized Linear Model Case

We start by considering the case where the evaluator functions satisfy the generalized linear model in Asm. 4.

[3] Similar problems are patient prioritization in hospitals (Déry et al., 2019), credit scoring (Provenzano et al., 2020), and resume review (Li et al., 2020).

2.1.3.1 The Oracle Strategy

We first define an *oracle* strategy that, beside σ_j and g, has prior knowledge about the parameters α_j. At each round, the oracle receives as input the evaluations $\{f_{i,t,j}\}$ and has to select K arms. We focus on oracle strategies $\mathfrak{O}$ of the following form

1. The oracle $\mathfrak{O}$ first aggregates the evaluations into a reward estimation $\widehat{r}^{\mathfrak{O}}_{i,t}$ for each arm i using a weighted average scheme. Let $\phi_{i,t} \in \mathbb{R}^J$ the vector collecting all evaluations as in Eq. 2.2 and $w \in \mathbb{R}^J$ a weight vector, then we define[4]
$$\widehat{r}^{\mathfrak{O}}_{i,t} = \langle w, g^{-1}(\phi_{i,t})\rangle, \tag{2.3}$$
where g^{-1} is the inverse of the link function applied component-wise to $\phi_{i,t}$. The choice of the weights is fixed and independent from the actual evaluations, but it may depend on the evaluation function and the noise distribution.
2. The oracle $\mathfrak{O}$ then returns the top-K arms according to the estimates $\widehat{r}^{\mathfrak{O}}_{i,t}$, i.e.,
$$\mathcal{A}^{\mathfrak{O}}_t := \arg\max^{K}_{i} \langle w, g^{-1}(\phi_{i,t})\rangle \tag{2.4}$$

The crucial aspect is then to find the weighting scheme w that guarantees the best performance for the oracle. Let $i^\star_1, \dots, i^\star_K$ be the true top-K arms and $\mathcal{A}^{\mathfrak{O}}_t = \{i^{\mathfrak{O}}_1, \dots, i^{\mathfrak{O}}_K\}$ be the *estimated top-K arms* according to the estimated rewards $\widehat{r}^{\mathfrak{O}}_{i,t}$. Ideally, at each round t, we would like to find the oracle weights that minimize the suboptimality gap
$$\Delta^{\mathfrak{O}}_t = \sum_{l=1}^{K} r_{i^\star_l,t} - \sum_{l=1}^{K} r_{i^{\mathfrak{O}}_l,t}. \tag{2.5}$$

Since the rewards $r_{i,t}$ as well as the evaluations $\{f_{i,t,j}\}$ are random, it is not possible to minimize the previous expression for any possible realization using a fixed set of weights. Thus, we rather focus on minimizing a high-probability upper-bound of Eq. 2.5.

Lemma 1. *Under Asm. 3 and 4, with α_j being the parameter of the generalized linear model for each evaluator $j = 1, \dots, J$ and σ_j being the sub-Gaussian parameter of the noise $\epsilon_{i,t,j}$, let $\delta \in (0,1)$ be a desired confidence level, then the oracle strategy designed to minimize a $(1-\delta)$-upper bound of Eq. 2.5 is characterized by the weights solving the optimization problem*
$$\begin{aligned} &\min_{w \in \mathbb{R}^J} 2\sqrt{K^3 \sum_{j=1}^{J} (w_j\sigma_j)^2 \ell_\delta} + K\sqrt{J\sum_{j=1}^{J}(w_j\sigma_j)^2}, \\ &\text{s.t.} \quad \sum_{j=1}^{J} w_j\alpha_j = 1 \end{aligned} \tag{2.6}$$
where $\ell_\delta = \ln\left(\frac{K_{max}}{\delta}\right)$. The previous problem has a closed-form solution $w^+ \in \mathbb{R}^J$ such that
$$w^+_j = \frac{\alpha_j}{\sigma_j^2 \|\alpha \cdot \sigma^{-1}\|}. \tag{2.7}$$
The resulting oracle has suboptimality gap w.p. $1-\delta$
$$\Delta^+_t \le \frac{2K\sqrt{\ln\left(\frac{K_{max}e}{\delta}\right)} + K\sqrt{J}}{\|\alpha \cdot \sigma^{-1}\|}. \tag{2.8}$$

We first remark that the previous lemma does not use Asm. 2 and it holds for any realization of the rewards, where the probability $(1-\delta)$ is w.r.t. to noise in the evaluations. We notice that the optimal weight w^+_j is proportional to the ratio α_j/σ^2_j, which describes the amount of "signal" with respect to the noise for evaluator j. Indeed, the oracle gives less weight to evaluators that are noisy (large σ_j), while relying more on evaluators with strong "signal" (large α_j). Indeed, as α_j increases w.r.t. σ_j, the evaluations tend to be near deterministic and thus more reliable. Interestingly, the suboptimal gap in Eq. 2.8 shows that the oracle improves as the number of evaluators increases (the

[4] In the linear case for $\alpha_j = 1$ (i.e., the link function g is the identity function) and Gaussian noise, (Yun et al., 2017) showed the exact posterior over $r_{i,t}$ given the evaluations $\{f_{i,t,j}\}$ takes a weighted average form as in Eq. 2.3.

Algorithm 1: GLM-ε-GREEDY algorithm

Input: Noise parameters $\{\sigma_j\}_{j\leq J}$, confidence level δ
Parameters: exploration level ε; number of arms to pull K; regularization λ
Set $\mathcal{H}_0 = \emptyset$, $\widehat{\alpha} = 0$ and $w_0 = 0$
for $t = 1, \dots, T$ **do**
  Sample $Z_t \sim \text{Ber}(\varepsilon)$
  Observe evaluations for each arm $(\phi_{i,t})_{i\leq K_t}$
  if $Z_t = 1$ **then**
    Pull arms in $\mathcal{A}_t$ obtained by sampling K arms uniformly in $\{1, \dots, K_t\}$
    Observe rewards $r_{i,t}$ for all $i \in \mathcal{A}_t$
    Add sample to dataset $\mathcal{H}_t = \mathcal{H}_{t-1} \cup \big(\cup_{i\in\mathcal{A}_t} \{(\phi_{i,t}, r_{i,t})\}\big)$
    Update estimators $\widehat{\alpha}_{j,t}$ by solving

$$\sum_{\phi,r\in\mathcal{H}_t} r(g(\widehat{\alpha}_{t,j}\cdot r) - \phi_j) - \lambda\widehat{\alpha}_{t,j} = 0 \tag{2.9}$$

    Update weights $w_{t,j} = \widehat{\alpha}_{t,j}/\|\widehat{\alpha}_t \cdot \sigma^{-1}\|$
  else
    Select $\mathcal{A}_t = \arg\max_i^K \langle w_t, \phi_{i,t}\rangle$

term $\|\alpha\cdot\sigma^{-1}\|$ grows as $\sqrt{J}$) but $\Delta_t^{\mathfrak{O}}$ does not tend to zero even when $J \to \infty$. While this might be counterintuitive (the learner is provided with an infinite number of independent evaluations), the residual gap is due to the nonlinear nature of the generalized linear model, where the zero-mean noise added to the evaluations may be amplified or decreased through g^{-1} while reconstructing the unknown parameters α_j.

Based on the previous lemma, the oracle strategy for the generalized linear case is defined by the weights w^+ and we denote by $\widehat{r}_{i,t}^+$ and $\mathcal{A}_t^+$ the associated reward estimates and top-K arm selection rule.

2.1.3.2 The GLM-ε-greedy Algorithm

Building on the oracle strategy defined in the previous section, we now consider the learning problem when the link function g and the noise distribution are known, but the learner has no knowledge of the parameters α_j. As customary in MAB problems, at each round t, the learner observes only the rewards of the selected arms in $\mathcal{A}_t$. The main challenge in this setting is that a learner leveraging the evaluations $\{f_{i,t,j}\}$ is directly affected in its choices (i.e., the set $\mathcal{A}_t$) by the noise $\epsilon_{i,t,j}$ generated at the beginning of round t. This is radically different from the standard MAB setting, where the noise (in the reward) *follows* the arms played by the learner, which are then independent from any noise conditionally on the past (see App. 2.A.2). A way to circumvent this dependency is to rely on an ε-greedy strategy, where only the samples obtained in exploratory steps are actually used to build an estimator of the unknown parameters α_j.[5] The resulting algorithm is detailed in Alg. 1. The core of Alg. 1 is a *maximum likelihood estimation* step where the algorithm learn the parameter α. Coherently with the evaluation model in Eq. 2.1, the true rewards $(r_{i,t})_{i,t}$ serves as the input for the function f_j (in this case the GLM model), while the evaluations $(\phi_{i,t})_{i,t}$ work as the values we need to fit.

We now compare the performance of Alg. 1 to the oracle strategy and define the notion of *relative* regret

$$R_T = \sum_{t=1}^{T} \Big(\sum_{i\in\mathcal{A}_t^+} \widehat{r}_{i,t}^+ - \sum_{i\in\mathcal{A}_t} \widehat{r}_{i,t}^+ \Big), \tag{2.10}$$

where $w^+ \in \mathbb{R}^J$ is the oracle weight vector, $\widehat{r}_{i,t}^+$ are the associated reward estimates, and $\mathcal{A}_t^+$ is the oracle set of top-K arms. This notion of regret, first introduced by Yun et al. (2017) in noisy contextual bandit, is comparing the quality of the arms returned by the algorithm and the oracle according to the *estimated rewards*, for which the oracle is optimal (see App. 2.A.5 for further discussion). The following theorem shows that if ε is properly tuned, ε-greedy has sublinear regret.

[5] While we study an ε-greedy type of algorithm, any type of exploration method decorrelating the estimation procedure and the exploration, like an *Explore-Then-Commit* strategy, would achieve a similar regret.

Theorem 1. *For any $\delta \in (0,1)$, $T \geq 8$, $\lambda = J^{-1}$ and set $\varepsilon = T^{-1/3}$, let $\eta^2_{\nu,j} = \mathbb{E}_{r\sim\nu}(g(\alpha_j r)^2)$ and $\eta_{\nu,\min} = \min_j \eta_{\nu,j}$. Then under Asm. 2, 3, and 4 we have that with probability at least $1-\delta$ the regret of Alg. 1 is bounded as*

$$R_T \leq \widetilde{\mathcal{O}}\left(T^{2/3}\left(\frac{(1+\sqrt{J^{-1}}\|\alpha\|)\Phi S\|\sigma\|_\infty}{\sqrt{K}\eta_{\nu,\min}} + \frac{\|g\|^3_\infty}{K^2\eta^2_{\nu,\min}}\right)\right)$$

where $\|g\|_\infty = \max_{j,x\in[0,C]} g(\alpha_j x)$ and

$$\Phi = 2K\|\sigma\|_\infty\left(2\sqrt{J} + \sqrt{K\ln\left(\frac{eK_{\max}}{K\delta}\right)}\right) + K\|g\|_\infty$$

$$S = \left(\|\alpha\cdot\sigma^{-2}\|_2 + \|\sigma^{-2}\|_2\right)\frac{\|\sigma^{-2}\cdot\alpha\|_2}{\|\sigma^{-1}\cdot\alpha\|^4_2} + \left(\frac{\|\sigma^{-1}\|_4}{\|\sigma^{-1}\cdot\alpha\|_2}\right)^2.$$

As expected, the regret of GLM-ε-greedy increases as $\widetilde{O}(T^{2/3})$. While this shows that the algorithm is able to approach the performance of the oracle, it also illustrates the difficulty of this setting, where the strict decoupling between explorative and exploitative steps used to guarantee a consistent estimation process translates into a higher regret. Interestingly, this result matches the regret of Yun et al. (2017) for contextual bandit with noisy features.

In order to investigate the main terms appearing in the previous bound, we consider the case where all evaluators share the same parameters $\alpha_j = \alpha_0$ and $\sigma_j = \sigma_0$ for some $(\alpha_0, \sigma_0) \in \mathbb{R}^2_+$ (with $\alpha_0 \geq 1$). In this case, the regret bound of Thm. 1 reduces to

$$R_T \leq \widetilde{O}\left(T^{2/3}\sqrt{K}\left(1+\sqrt{\frac{K}{J}}\right)\frac{\sigma_0^2}{\alpha_0}\right)$$

We first notice that the bound scales as $\max\{\sqrt{K}, K\sqrt{J^{-1}}\}$. Similar to the suboptimality gap for the oracle (see Lemma 1), when $J \to \infty$, the bound improves but does not decrease down to 0 and rather converges to $\sqrt{K}$. The dependency on K may be surprising, since for $K = K_{\max}$ the regret is trivially 0 at each round (i.e., the algorithm returns all arms and it cannot make any error in the ranking). However, this dependency comes from the fact that in the regret analysis we bound the norm of the evaluations for the selected jobs, which scales linearly with K. We believe a more refined analysis could alleviate this dependency. Last, the regret depends inversely on the "signal-to-noise" ratio $\frac{\alpha_0}{\sigma_0^2}$.

2.1.4 Linear Case

We now consider the special case of linear evaluations and show how this relatively minor change to the problem has a major impact on how to approach the learning problem and the regret.

2.1.4.1 The Linear Oracle Strategy

Similar to the GLM case, we define the oracle strategy that defines an estimated reward $\widehat{r}^{\mathfrak{O}}_{i,t} = \langle w, \phi_{i,t}\rangle$. Then the oracle ranks arms according to $\widehat{r}^{\mathfrak{O}}_{i,t}$ and selects the set of top-K $\mathcal{A}^{\mathfrak{O}}_t$ accordingly. We then optimize weights to minimize a high-probability upper bound to the suboptimality gap $\Delta^{\mathfrak{O}}_t$.

Lemma 2. *Under Asm. 3 and 5, with α_j, being the parameter for each evaluator $j = 1,\dots,J$, σ_j being the sub-Gaussian parameter of the noise $\epsilon_{i,t,j}$, let $\delta \in (0,1)$ be a desired confidence level, then the oracle strategy designed to minimize a $(1-\delta)$-upper bound of Eq. 2.5 is characterized by the weights obtained as the solution of the optimization problem*

$$\min_{w\in\mathbb{R}^J} 2\sqrt{K^3\sum_{j=1}^J (w_j\sigma_j)^2\ell_\delta} \text{ s.t. } \sum_{j=1}^J w_j\alpha_j = 1 \quad , \tag{2.11}$$

where $\ell_\delta = \ln\left(\frac{K_{max}}{\delta}\right)$. The previous problem has a closed-form solution $w^+ \in \mathbb{R}^J$ such that

$$w^+_j = \frac{\alpha_j}{\sigma_j^2\|\alpha\cdot\sigma^{-1}\|^2}. \tag{2.12}$$

The resulting oracle has suboptimality gap w.p. $1-\delta$

$$\Delta^+_t \leq \frac{2K\sqrt{\ln\left(\frac{K_{max}e}{\delta}\right)}}{\|\alpha\cdot\sigma^{-1}\|} \tag{2.13}$$

Algorithm 2: The *Evaluation-Structure-Aware Greedy* (ESAG) algorithm for the linear case.

Input: Noise parameters $\{\sigma_j\}_{j\leq J}$, confidence level δ
Parameters: number of arms to pull K
Set $\widehat{\alpha}=0$, $w_0=0$ and $N_t=0$
for $t=1,\dots,T$ **do**
  Observe evaluations for each arm $(\phi_{i,t})_{i\leq K_t}$
  Select $\mathcal{A}_t=\arg\max_i^K \langle w_t,\phi_{i,t}\rangle$
  Observe rewards $r_{i,t}$ for all $i\in\mathcal{A}_t$
  Update $\widehat{\alpha}_t$ as:

$$\widehat{\alpha}_{t+1}=\widehat{\alpha}_t\frac{N_t+K_t-N_tK_t}{(N_t+K_t)N_t}+\frac{\sum_{i=1}^{K_t}\phi_{i,t}}{K_t} \tag{2.15}$$

  Update $N_{t+1}=N_t+K_t$ and w_t as

$$w_{t+1,j}=\frac{\widehat{\alpha}_{t+1,j}}{\sigma_j^2\,\|\widehat{\alpha}_{t+1,j}\cdot\sigma^{-1}\|^2} \tag{2.16}$$

The weights of the oracle have the same expression in the GLM case in Lemma 1. Nonetheless, the suboptimality gap is smaller than in Lemma 1 as the linear structure allows to concentrate the noise of the evaluations, unlike in the GLM setting where the potential non linearity of g forces us to study the worst-case scenario. Notably, in the linear case, we see that as J tends to infinity the suboptimality gap tends to zero.

2.1.4.2 Evaluation-Structure-Aware Greedy

Similarly to Sec. 2.1.3.2, the learner has no knowledge of the parameters α_j but only knows the noise distribution and the linear structure of the evaluations. The main estimation difficulty of the general case still applies to this setting, i.e., using samples obtained by selecting arms based on the evaluations may introduce a bias in the estimation process.

Instead of introducing explicit exploration steps to gather "unbiased" samples, as done in the GLM case, we exploit a more subtle property for this case. We notice for any evaluator j, the expected evaluation is

$$\mathbb{E}\big[f_{i,t,j}\big]=\mathbb{E}\big[\alpha_j r_{i,t}+\epsilon_{i,t,j}\big]=\alpha_j\overline{r}, \tag{2.14}$$

where $\overline{r}=\mathbb{E}[r_{i,t}]$ is the expectation of the reward distribution ν in Asm. 2. Consider an oracle strategy that is fed with the parameters $\alpha_j\overline{r}$, then the optimal weights in Eq. 2.12 become $\widetilde{w}^+=w^+/\overline{r}$. While this leads to estimates $\widetilde{r}^+_{i,t}$ that are biased w.r.t. the oracle estimates $\widehat{r}^+_{i,t}$, the factor $1/\overline{r}$ is evaluator- and arm-independent and it does not impact the ranking returned by this *biased* oracle, i.e., $\widetilde{\mathcal{A}}_t=\mathcal{A}_t^+$. This in striking contrast with the GLM where it is not possible to easily evaluate a vector proportional to α because of the potential non-linear behavior of g.

Building on this evidence, we define an algorithm, the *Evaluation-Structure-Aware Greedy* (ESAG) in Alg. 2, that avoids the use of the observed rewards altogether, thus removing the statistical dependency between noise and decisions, and rather tries to estimate the expected evaluations in Eq. 2.14 (see Eq. 2.15 in Alg. 2) and use them to build estimates, as in the biased oracle. Since ESAG only relies on the evaluations available at round t, it does not need any *explicit* exploration strategy to collect useful information, and it executes greedy actions according to the current weights w_t at each round. We can derive the following regret guarantees.[6]

Theorem 2. *Under Asm. 2, 3, and 5 for any $\delta\in(0,1)$, $T\geq 1$ with probability at least $1-\delta$ the regret of Alg. 2 is bounded by:*

$$R_T\leq\mathbb{E}_{r\sim\nu}(r)\Phi' K\left[\left(\frac{\Phi'}{\mathbb{E}_{r\sim\nu}(r)\|\alpha\|_\infty\sqrt{K}}\right)^2+\frac{8\sqrt{T}\Phi' S}{\sqrt{\bar{K}_T}}\right]$$

where $\bar{K}_T=\frac{T}{\sum_{t=1}^T\frac{t-1}{\sum_{l=1}^{t-1}K_l}}\geq K$ is the harmonic average of the number of arms over T steps, S is the same as in Thm. 1 and

$$\Phi'=2\|\alpha\|C\ln\left(\frac{4}{\delta}\right)+2\|\sigma\|_\infty\left(2\sqrt{J}+\sqrt{K\ln\left(\frac{K_{max}}{K\delta}\right)}\right)$$

[6] While we derive Thm. 2 for a greedy algorithm, similar results hold for an optimistic exploration strategy.

The most interesting aspect of the previous theorem is that the regret is of order $\widetilde{O}(\sqrt{T})$, since ESAG does not pay for the sharp separation between exploration and exploitation steps as for GLM-ε-GREEDY.

2.1.5 Experiments

In order to study different aspects of the settings and algorithms introduced in the previous sections, we focus on both synthetic and real data experiments. We report further results in the supplementary material.

2.1.5.1 Synthetic Data

We first validate our algorithms on synthetic data. We consider $K_t = K_{\max} = 20$ arms where for each arm we get $J = 10$ evaluations. The reward distribution ν is a Gaussian distribution centered at 0 and truncated between $[0, 20]$. We consider both the logistic case with $g(x) = (1 + \exp(-x))^{-1}$ and the linear case where for all evaluators $f_j(x) = \alpha_j x$. At the beginning of each experiment, we draw the coefficients α_j and the parameters σ_j from uniform distributions in $[\alpha_0/2; 3\alpha_0/2]$ and $[\sigma_0/2; 3\sigma_0/2]$ respectively. As discussed in Sect. 2.1.3, a critical term characterizing the problem structure is the ratio α_j/σ_j. We then set $\alpha_0 = 1$ and consider three different values for σ_0 such that $\alpha_0/\sigma_0 \in \{0.1, 1, 10\}$. Finally, the noise in the evaluations are generated as $\epsilon_{i,t,j} \sim \mathcal{N}(0, \sigma_j^2)$ in the linear case whereas in the GLM case $\epsilon_{i,t,j}$ is drawn from a truncated centered gaussian distribution with variance $2\sigma_j^2$. We average all results over 80 runs and we report 95% confidence intervals. Additional details are reported in the supplement.

Oracle performance. Before investigating the performance of the learning algorithms, we compare the performance of the oracle strategy to a simple average strategy that defines $\widehat{r}_{i,t}^{\text{avg}} = 1/J \sum_{j=1}^{J} f_{i,t,j}$ and ranks arms accordingly. We also study how the suboptimality gap of the oracle changes as J increases for both the GLM and linear case. As shown in Fig. 2.2a, in both settings that oracle strategies outperform the simple average. Furthermore, as predicted by Lemma 1 and 2, the oracle suboptimality gap decreases as J increases, but it plateaus to a fixed value for GLM, while it tends to zero for the linear case.

2.1.5.2 The GLM Case

We consider different types of algorithms: GLM-ε-GREEDY (Alg. 1); GLM-ε-GREEDY-ALL, it has same structure as Alg. 1 but uses samples from both explorative and exploitative steps to build the estimator $\widehat{\alpha}$; GLM-EVALBASEDUCB, it uses all samples to build an estimator $\widehat{\alpha}$ and leverages a high-probability confidence interval on $\widehat{\alpha}$ to derive optimistic weights w and rank and select arms accordingly; GLM-LINUCB, the algorithm of Abbasi-Yadkori et al. (2011) using $g^{-1}(\phi_{i,t})$ as features; GLM-ESAG, Alg. 2 adapted for the GLM case; RAND, the fully random strategy; GLM-GREEDY, the greedy strategy using the MLE estimator $\widehat{\alpha}$; EXP4.P, the bandit with expert advice algorithm in Beygelzimer et al. (2011).

Estimation Bias. As discussed in Sect. 2.1.3.2, one of the critical aspects that motivated the use of an ε-greedy approach and led to the $\widetilde{O}(T^{2/3})$ is the fact that whenever the set of arms is chosen according to the noisy evaluations $\{f_{i,t,j}\}$, the dependency between $\epsilon_{i,t,j}$ and $\mathcal{A}_t$ may create a bias when estimating the parameters α_j using the samples $r_{i,t}$ observed after selecting $\mathcal{A}_t$. We illustrate this effect in Fig. 2.2b, where we report the error of the estimates $\widehat{\alpha}_t$ computed by GLM-ε-GREEDY and GLM-ε-GREEDY-ALL w.r.t. the true parameters α. While the error of the estimator computed by GLM-ε-GREEDY decreases over time, the error for GLM-ε-GREEDY-ALL has a residual bias due to the estimation procedure. Similar results can be shown for all the algorithms (e.g., GLM-ESAG) that rely on samples generated by selecting $\mathcal{A}_t$ based on the evaluations $\{f_{i,t,j}\}$ and they can be reproduced in the linear setting as well.

Regret Performance. We compare the performance of different learning algorithms in terms of relative regret w.r.t. the oracle defined in Sec. 2.1.3.We remove from Fig. 2.2c all algorithms (i.e., RAND, GREEDY, EXP4.P) that suffer large linear regret to avoid loosing resolution on the other algorithms. While GLM-ESAG and GLM-LINUCB have better regret, they both have a linear regret that keeps increasing over time, while GLM-ε-GREEDY has a sublinear regret.[7]

[7] Notice that we have not actively tried to find problem parameters that would make the linear regret of GLM-ESAG and GLM-LINUCB larger than GLM-ε-GREEDY in a shorter time. The linear regret of GLM-ESAG is due to the residual estimation bias illustrated in Fig. 2.2b.

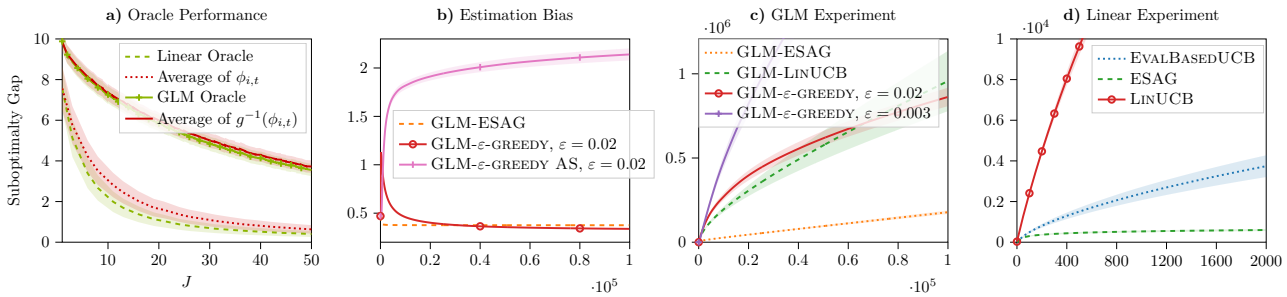


Figure 2.2: **a)** Average suboptimality gap $\Delta_t^{\mathfrak{D}}$ for the oracle strategy in the GLM and linear cases as a function of the number of evaluators J. **b)** Estimation error $\|\widehat{\alpha}_t - \alpha\|$. **c)** Regret w.r.t. to the oracle as defined in Sect. 2.1.3 for the GLM case. **d)** Regret w.r.t. to the oracle as defined in Sect. 2.1.3 for the linear case in the high noise regime (i.e., $\alpha_j = 0.1\sigma_j$).

Table 2.1: Cumulative badness over $T = 2000$ steps

Alg.	Badness
RAND	30076.18
GLM-ε-GREEDY	53403.6
GLM-ε-GREEDY-ALL	53419
GLM-EVALBASEDUCB	53328.3
GLM-LINUCB	30910.5
GLM-ESAG	53393.9
GLM-GREEDY	53371.9
EXP4.P	47766.7
EVALBASEDUCB	**90768.2**
LINUCB	**90332.2**
ESAG	**90790.5**
GREEDY	69956.3

2.1.5.3 The Linear Case

We also study the linear case and compare the linear versions of the algorithms described above. As illustrated in Fig. 2.2d, the crucial difference w.r.t. the GLM case is that now ESAG has sublinear regret, whereas other algorithms have linear regret. Interestingly, EVALBASEDUCB, despite the bias introduced by using samples obtained by selecting actions based on the noisy observation, is able to learn a good strategy and it outperforms LINUCB, but still suffers linear regret.

2.1.5.4 Content Review Prioritization

We now move to a real-world problem to investigate the performance of our algorithms when their assumptions are no longer verified.

Data description. We consider a small dataset of content shared on a large social media firm that has been reported for violating the platforms' community standards. In order to ensure that the most harmful content is prioritized for reviewing, the platform assigns badness score for each piece of content which increases with the severity of the content. We consider four different classifiers that provide badness estimates for the sampled content. Each of these classifiers are trained on the real data and follow different modeling architectures to predict the badness score. Our objective is to leverage scores from these four classifiers to identify the most harmful subset of content and flag them for the platform to review and action them.

Performance. We evaluate the cumulative badness of the content selected by the algorithms. The higher the score the better. The results are reported in Tab. 2.1. We first notice that all learning algorithms perform significantly better than the random strategy, thus indicating that the GLM and linear assumptions are accurate enough to return meaningful rankings. Nonetheless, we notice that GLM-based algorithms do not perform as well as linear ones, probably due to the choice of the logistic function, which, in this case, does not fit data accurately. On the other

hand, ESAG is the algorithm that performs best, followed by EvalBasedUCB and LinUCB. Notice that in the case of real data, we may even expect LinUCB to perform better, since it relies on somewhat less restrictive assumptions (no assumption is made on how the features are generated) and it relies on the true rewards to estimate parameters (this is also the case for EvalBasedUCB). This is in contrast with ESAG that exclusively builds on the evaluations, which clearly do not respect an exact linear model, to estimate the unknown parameters. This shows that, even in problems where the assumptions do not hold, ESAG is robust enough and it is competitive w.r.t. a large variety of algorithms.

2.1.6 Potential Extensions and Concluding Remarks

In this section, we studied a MAB problem where the learner is provided with noisy and biased evaluations of the true reward for each arm. We showed that under specific assumptions it is possible to design learning algorithms that are able to compete to oracle strategies both in theory and in practice. The empirical validation on real data also shows that this model and the associated algorithms are promising for solving challenging real-world problems.

Extensions. There is a number of directions that could be pursued to extend our current results.

- *Evaluation functions.* The GLM and linear assumptions are relatively strong. A natural venue of improvement is to generalize our results to richer function spaces, such as Gaussian processes.
- *Persistent arms.* While we assume that the set of arms is "refreshed" at each round, we can easily extend our setting to the case where all the arms that are not selected in $\mathcal{A}_t$ remain in the pool of arms available at the next round (e.g., content that has not been reviewed). All our results naturally extend to this case, except for ESAG, which would not be able to use K_t samples at each round, but would rather get K new samples corresponding to the arms replaced at each round.
- *Heteroschedastic noise.* In the model illustrated in Fig. 2.1, the noise $\epsilon_{i,t,j}$ is heteroschedastic, where the variance may depend on the reward value r. While our model leverages an upper-bound σ_j on the actual variance, better adapting to the reward-dependent variance may improve the performance.
- *Relaxing Asm. 2.* This assumption is used only in the linear case, while it is possible to remove any stochastic assumption on the rewards and replace it by a milder condition that requires that the (arbitrary) sequence of rewards is such that cumulative sum of the rewards observed over time is a growing function with T, i.e., $\sum_{t=1}^{T} r_{i_t,t} = \Omega(t)$. The results for GLM-$\varepsilon$-greedy could be easily extended to this case.

In the remainder of this chapter, we focus on a constraint related to the performance of the learning agent. In most potential applications of RL or bandit algorithms a trusted decision rule is already used and performs reasonably well. One obstacle to use directly RL algorithms is that while the algorithm will converge eventually to the optimal policy there is no guarantee that the algorithm will outperform the previously set decision rule in a time that is reasonable for the desired application.

2.2 Improved Conservative Exploration for Linear Contextual Bandits

Many problems in fields such as digital marketing, healthcare, finance, and robotics can be formulated as decision-making under uncertainty. Although many learning algorithms have been developed to find a good/optimal policy for these problems, a major obstacle in using them in real-world applications is the lack of guarantees for the actual performance of the policies they execute over time. Therefore, for the applicability of these algorithms, it is important that they execute policies that are guaranteed to perform at least as well as an existing *baseline*. We can think of the baseline either as a baseline value or the performance of a baseline policy. It is important to note that since the learning algorithms generate these polices from data, they are random variables, and thus, all the guarantees on their performance should be in high probability. This problem has been recently studied under the general title of *safety w.r.t. a baseline* in bandits and reinforcement learning (RL), in both *offline* Bottou et al. (2013); Thomas et al. (2015a,b); Swaminathan and Joachims (2015); Petrik et al. (2016) and *online* Mansour et al. (2015); Wu et al. (2016); Kazerouni et al. (2017); Katariya et al. (2019) settings.

In the online setting, which is the focus of this section, the learning algorithm updates its policy while interacting with the system. Although the algorithm eventually learns a good or an optimal policy, there is no guarantee on

the performance of the intermediate policies, especially at the very beginning, when the algorithm needs to heavily explore different options. Therefore, in order to make sure that at any point in time the (cumulative) performance of the policies generated by the algorithm is not worse than the baseline, it is important to control the exploration and make it more *conservative*. Consider a recommender system that runs our learning algorithm. Although we are confident that our algorithm will eventually learn a strategy that performs as well as the baseline, and possibly even better, we should control its exploration not to lose too many customers, as a result of providing them with unsatisfactory recommendations. This setting has been studied in multi-armed bandits Wu et al. (2016), contextual linear bandits Kazerouni et al. (2017), and stochastic combinatorial semi-bandits Katariya et al. (2019). These papers formulate the problem using a constraint defined based on the performance of the baseline policy (mean of the baseline arm in the multi-armed bandit case), and modify the corresponding UCB-type algorithm Auer et al. (2002a) to satisfy this constraint. At each round, the conservative bandit algorithm computes the action suggested by the corresponding UCB algorithm, if the action satisfies the constraint, it is taken, otherwise, the algorithm acts according to the baseline policy. Another algorithm in the online setting is by Mansour et al. (2015) that balances exploration and exploitation such that the actions taken are compatible with the agent's (customer's) incentive formulated as a Bayesian prior.

In this section, we focus on UCB-type algorithms and improve the design and empirical performance of the conservative algorithms in the contextual linear bandit setting. We first highlight the limitations of the existing conservative bandit algorithms Wu et al. (2016); Kazerouni et al. (2017) and show that simple modifications in constructing the conservative condition and the arm-selection strategy may significantly improve their performance. We show that our algorithm is formally correct by proving a regret bound, matching existing results and illustrate its practical advantage w.r.t. state-of-the-art algorithms in a number of synthetic and real-world environments. Finally, we consider the more realistic scenario where the conservative constraint is verified at predefined checkpoints (e.g., a manager may be interested in verifying the performance of the learning algorithm every few days). In this case, we prove a regret bound showing that as the checkpoints become less frequent, the conservative condition has less impact on the regret, which eventually reduces to the standard (unconstrained) one.

2.2.1 Conservative Contextual Linear Bandits

We consider the standard contextual linear bandit setting with arm dependent features, see Section 1.2.2 with $K > 1$ arms. At each time t, the agent selects an arm $a_t \in \mathcal{A}_t = \{1, \dots, K\}$ and observes a reward

$$r_a^t = \langle \theta^\star, x_{t,a} \rangle + \eta_a^t := \mu_a^t + \eta_a^t, \tag{2.17}$$

where $\theta^\star \in \mathbb{R}^d$ is a parameter vector, $x_{t,a} \in \mathbb{R}^d$ are the features of arm a at time t, and η_a^t is a zero-mean σ^2-subgaussian noise. When the features correspond to the canonical basis, this formulation reduces to multi-armed bandit (MAB) with d arms. In the more general case, the features may depend on a context s_t, so that $x_{t,a} = \phi(s_t, a)$ denotes the feature vector of a context-action pair (s_t, a) and (2.17) defines the so-called linear contextual bandit setting.

We rely on the following standard assumption on the features and the unknown parameter $\theta^\star$.

Assumption 6. *There exist $B, D \geq 0$, such that $\|\theta^\star\|_2 \leq B$, $\|x_{t,a}\| \leq D$, and $\langle \theta^\star, x_{t,a} \rangle \in [0, 1]$, for all t and a.*

Given a finite horizon n, the performance of the agent is measured by its (pseudo)-*regret*:

$$R(n) = \sum_{t=1}^{n} \langle \theta^\star, x_{t,a_t^\star} \rangle - \langle \theta^\star, x_{t,a_t} \rangle,$$

where $a_t^\star \in \arg\max_a \langle \theta^\star, x_{t,a} \rangle$ is the optimal action at time t. In the conservative setting, the objective is to minimize the regret under additional performance constraints w.r.t. a known baseline. We assume the agent has access to a *baseline policy*, which selects action b_t at time t.[8] The learning problem is constrained such that, at any time t, the difference in performance (i.e., expected cumulative reward) between the baseline and the agent should never fall below a predefined fraction of the baseline performance. Formally, the *conservative constraint* is given by

$$\forall t > 0, \qquad \sum_{i=1}^{t} \mu_{a_i}^i \geq (1 - \alpha) \sum_{i=1}^{t} \mu_{b_i}^i, \tag{2.18}$$

[8] In the non-contextual case, the baseline policy reduces to a single baseline action b.

where $\alpha \in (0,1)$ is the conservative level. As the LHS of (2.18) is a random variable depending on the agent's strategy, we require this constraint to be satisfied with high probability. Finally, in order to keep the presentation and analysis simple, we rely on the following assumption.

Assumption 7. *For any $t > 0$, the performance of the baseline strategy until t is known, i.e., $\sum_{i=1}^{t} \mu_{b_i}^i$ can be evaluated by the agent.*

This assumption is often reasonable since the baseline performance can be estimated from historical data (see Rem. 3 in Kazerouni et al. (2017)). Furthermore, as shown in Wu et al. (2016); Kazerouni et al. (2017), this knowledge can be removed and the algorithm can be modified to incorporate the estimation process (and preserve the same order of regret).

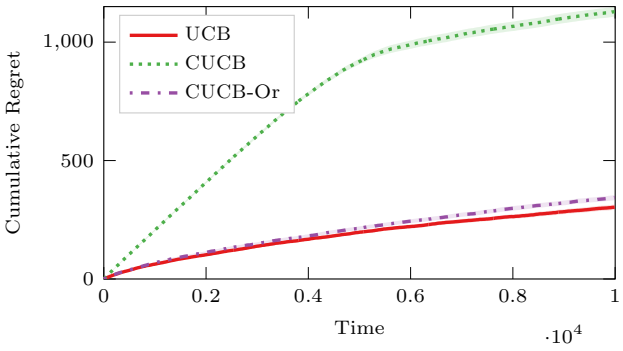


Figure 2.3: Comparison of the cumulative regret between UCB, its conservative variant (CUCB), and an *oracle* version of CUCB, where (2.18) can be evaluated exactly to decide whether to select the UCB arm or the baseline.

Conservative Exploration. Conservative exploration algorithms Wu et al. (2016); Kazerouni et al. (2017) are based on a two-step process to select the action to play. In the first step, they compute an optimistic action based on the optimism-in-the-face-of-uncertainty principle, i.e., using UCB Auer et al. (2002b) or LinUCB Li et al. (2010); Abbasi-Yadkori et al. (2011), which is effective in exploring and minimizing the regret over time. In the second step, they evaluate the conservative condition by replacing the unknown mean with a statistical lower bound. If the condition is verified, they play the optimistic action, otherwise, they act conservatively by selecting the baseline b_t. Playing the baseline over multiple steps contributes to "build a conservative budget", so that condition (2.18) is more likely to be satisfied by the UCB arm, and thus, allowing to execute explorative actions.

Formally, let S_t^b be the set of times up to t (included) where the agent played the baseline and $S_{t-1} = [t] \backslash S_t^b$ be the complementary set, i.e., when the agent played the UCB action. CLUCB uses the information collected when playing non-conservatively to build an estimator of $\theta^\star$ by solving a regularized least-square problem $\widehat{\theta}_t = (\Phi_t \Phi_t^\top + \lambda I)^{-1} \Phi_t Y_t$, where $\lambda > 0$, $\Phi_t = (x_{i,a_i})_{i \in S_{t-1}} \in \mathbb{R}^{d \times |S_{t-1}|}$ and $Y_t = (r_{a_i}^i)_{i \in S_{t-1}} \in \mathbb{R}^{|S_{t-1}|}$. Denote by $V_t = \lambda I + \Phi_t \Phi_t^\top$ the design matrix of the regularized least-square problem and by $\|x\|_V = \sqrt{x^\top V x}$ the weighted norm w.r.t. any positive matrix $V \in \mathbb{R}^{d \times d}$. We define the confidence set $\Theta_t = \{\theta \in \mathbb{R}^d : \|\theta - \widehat{\theta}_t\|_{V_t^{-1}} \leq \beta_t\}$ where

$$\beta_t = \sigma \sqrt{d \log\left(\frac{1 + D^2(1 + |S_{t-1}|)/\lambda}{\delta}\right)} + B\sqrt{\lambda}, \tag{2.19}$$

which guarantees that $\theta^\star \in \Theta_t$, for all $t > 0$, w.p. $1 - \delta$.

Similar to LinUCB, the optimistic action is computed as

$$a_t \in \arg\max_{a \in \mathcal{A}_t} \max_{\theta \in \Theta_t} \langle \theta, x_{t,a} \rangle,$$

and CLUCB decides if the action is "safe" by evaluating the following conservative condition:

$$\sum_{i \in S_{t-1}^b} \mu_{b_i}^i + \min_{\theta \in \Theta_t} \left\langle \theta, x_{t,a_t} + \sum_{i \in S_{t-1}} x_{i,a_i} \right\rangle \geq (1 - \alpha) \sum_{i=1}^{t} \mu_{b_i}^i. \tag{2.20}$$

The leftmost term in (2.20) represents the expected cumulative reward associated to baseline actions played up to time $t-1$. The second term –minimization problem– denotes the lower bound to the cumulative reward of optimistic

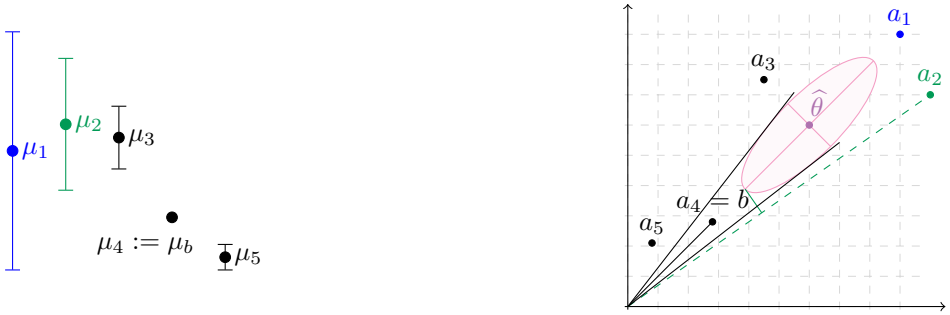


Figure 2.4: Examples of settings where the UCB arm (blue) does not satisfy the conservative condition but there is another "safe" arm to play (rather than the baseline).

actions, including the current optimistic action a_t. This lower bound is constructed using the most recent confidence set Θ_t. The rightmost term denotes the expected cumulative reward of playing the baseline policy at each step. This inequality is a surrogate for the conservative condition in (2.18).
If the condition is satisfied, then the optimistic action a_t is played, otherwise, the baseline strategy is selected and the corresponding action b_t is executed.

Limitations. While CLUCB enjoys strong regret guarantees (in the MAB setting, it is indeed near-optimal), its empirical behavior is often over-conservative, i.e., the baseline strategy is selected for a very long time to build enough *conservative budget* before the actual exploration takes place. We identify two main algorithmic causes for such behavior.

First, in building the conservative condition (2.20), CUCB and CLUCB rely on possibly loose statistical lower-bounds for the mean of the actions selected so far. This is well illustrated by the simulation in Fig. 2.3 in the MAB setting, where we report the performance of UCB, CUCB, and an *oracle* variant of CUCB, when the conservative condition (2.18) is evaluated exactly (i.e., no lower-bound is used). While the *oracle* version has almost the same regret as UCB, and thus, showing that the conservative condition itself does not have a major impact on the exploration of UCB, CUCB has a much higher regret. This shows that possibly loose estimates of the conservative condition have a significant impact on the regret. In fact, tightening the conservative condition would allow selecting the baseline strategy only when it is "strictly" needed, thus, reducing the conservative steps and improve the overall exploration performance.

Second, the two-step selection strategy of CUCB and CLUCB performs either an exploration step, when the UCB action is selected, or a conservative step, when the baseline is executed. Such sharp division between exploration and conservative steps may be unnecessary, as other actions may still be "safe" (i.e., satisfying the conservative condition), and thus, contribute to build "conservative budget", and, at the same time, be useful for exploration (e.g., optimistic), despite not being the UCB action. Exploiting such arms may lead to a better performance.

Finally, the conservative condition (2.18) itself is often too strict in practice. Instead of performing almost as well as the baseline *at every step*, it is more likely that recurrent "checkpoints" are set at which the agent is required to meet the condition. In this case, the agent may have extra time to perform exploratory actions and possibly recover from bad past choices when getting close to the conservative checkpoint.

In the next section, we address the two algorithmic limitations described above, while we later illustrate how a relaxed conservative condition may indeed allow an agent to achieve much smaller regret.

2.2.2 Improved Conservative Exploration

In this section we present Conservative Constrained LinUCB (CLUCB2) (Alg. 3 with $T = 1$), an improved conservative exploration algorithm for contextual linear bandit. All the proofs can be found in the extended version.

2.2.2.1 CLUCB2

The first improvement w.r.t. CLUCB is relative to the conservative condition (2.20). When evaluating the rewards accumulated by the agents so far, we rely on the fact that the sequence $(r^i_{a_i} - \mu^i_{a_i})_{i \in S_{t-1}}$ is a Martingale Difference

Sequence (MDS) with respect to the filtration $\mathcal{F}_{t-1} = \sigma\left((a_j, x_{j,a_j}, r^j_{a_j})_{j \in S_{t-1}}\right)$, i.e., the history strictly before time t. Indeed, the choice of arm a_i is $\mathcal{F}_i$-measurable and for all $i \in S_{t-1}$:

$$\mathbb{E}\left[r^i_{a_i} - \mu^i_{a_i} \mid \mathcal{F}_i\right] = \sum_{j=1}^{|\mathcal{A}_i|} \mathbb{1}_{(a_i=j)} \left(\mathbb{E}\left[r^i_j\right] - \mu^i_j\right) = 0$$

By using Freedman's inequality Freedman (1975) for martingales, with probability at least $1-\delta$ we have

$$\left|\sum_{i \in S_{t-1}} (r^i_{a_i} - \mu^i_{a_i})\right| \le \psi_L(t) := \sigma\sqrt{2|S_{t-1}|L^\delta_t} + \frac{2}{3}L^\delta_t \tag{2.21}$$

where $L^\delta_t := \log\left(3\left(|S_{t-1}| \vee 1\right)^2/\delta\right)$. Thus we replace (2.20) by

$$\sum_{i \in S^b_{t-1}} \mu^i_{b_i} + \sum_{i \in S_{t-1}} r^i_{a_i} - \psi_L(t) + \min_{\theta \in \Theta_t} \langle \theta, x_{t,a_t} \rangle \ge (1-\alpha) \sum_{i=1}^{t} \mu^i_{b_i}. \tag{2.22}$$

While it is not possible to prove that (2.22) is always tighter, in the next section we provide an extensive discussion on the potential improvements.

A second limitation of CLUCB is its two-step approach to action selection, in which either the optimistic action satisfies the conservative condition or the baseline b_t is selected. The idea behind this strategy is that it is necessary to select baseline actions before exploring any other action in order to build a "conservative budget", which allows performing effective exploration later on (once the conservative condition is met). In CLUCB2 we propose to combine the explorative and conservative requirements by selecting the most optimistic (i.e., useful for exploration) "safe" (i.e., satisfying the conservative condition) action. Formally, the algorithm computes the set $\mathcal{C}_t$ of "safe" arms such that[9]

$$\mathcal{C}_t = \left\{a \in \mathcal{A}_t \setminus \{b_t\} \mid \sum_{i \in S_{t-1}} r^i_{a_i} - \psi_L(t) + \sum_{i \in S^b_{t-1}} \mu^i_{b_i} + \max\left\{\min_{\theta \in \Theta_t} \langle \theta, x_{t,a} \rangle, 0\right\} \ge (1-\alpha) \sum_{i=1}^{t} \mu^i_{b_i}\right\}$$

where ψ_L is the Martingale bound given in Eq. 2.21. The algorithm plays the arm that solves the following constrained optimization problem:

$$a_t \in \arg\max\left\{r^t_{b_t}, \max_{a \in \mathcal{C}_t} \max_{\theta \in \Theta_t} \langle \theta, x_{t,a} \rangle\right\} \tag{2.23}$$

where, by definition, the max over an empty set is $-\infty$. The maximizer is either the baseline arm b_t or an arm in $\mathcal{C}_t$ that is optimistic w.r.t. the baseline. In order to illustrate the idea behind (2.23), consider the configuration illustrated in Fig. 2.4*(left)* for a MAB setting. If the algorithm has not built enough margin, the UCB arm (a_1) would not satisfy the conservative condition as its lower-confidence bound is well below the baseline. As a consequence, CUCB selects the baseline arm a_4. A direct improvement can be achieved by selecting arm a_3 (i.e., the one with the higher lower bound among the arms passing the conservative condition) as suggested in Wu et al. (2016). Nonetheless, while arm a_3 is indeed better than baseline and it allows building conservative budget faster, it may not be effective from an exploration point of view. In (2.23) we suggest arm a_2 would be a better choice as it does not give up on reducing the regret (i.e., it has a larger UCB than a_2). This may indeed result in a better tradeoff between building conservative budget and performing effective exploration. Finally, Fig. 2.4*(right)* shows that (2.23) may be effective even in the linear setting. The stretched out ellipsoid on one axis gives a precise estimate of some bad arms, while good arms would not be selected by choosing the arm maximizing the lower bound. Even though the arms are more correlated due to the linear structure, the interpretation of this case is as the one for stochastic MABs.

From a computational perspective, the cost of an update for both CLUCB and CLUCB2 is $O(Ad^3)$. This complexity comes from the maximization over action and the construction of the confidence intervals. Compared to CLUCB, CLUCB2 has to evaluate the conservative condition for each arm instead of only for the UCB arm. However, the cost of this operation is dominated by the arm selection procedure.

Finally, we notice that following the same construction as in Kazerouni et al. (2017), CLUCB2 can be easily adapted to the case when Asm. 7 does not hold and the baseline performance needs to be estimated online.

[9] S_{t-1} contains all steps when the baseline is *not* selected, and now it may include arms different from the UCB arm.

Algorithm 3: CLUCB2 ($T = 1$) and CLUCB2T

Input: α, δ, T
Set $S_0 = S_0^b = \emptyset$, $k = 0$
for $t = 1, \dots, n$ **do**
  Compute "safe" set C_t as in Eq. 2.23 or Eq. 2.26
  Compute a_t by solving (2.23)
  Pull arm a_t and observe $r_{a_t}^t$
  if $a_t \neq b_t$ **then**
    Set $S_t = S_{t-1} \cup \{t\}$, $S_t^b = S_{t-1}^b$
    Compute new confidence set Θ_{t+1}
  else
    Set $S_t = S_{t-1}$, $S_t^b = S_{t-1}^b$, $\Theta_{t+1} = \Theta_t$
  if $t \bmod T = 0$ **then**
    $k = k + 1$

2.2.2.2 Theoretical Analysis

Let $\Delta_a^t = \mu_{a_t^\star}^t - \mu_a^t$ be the action gap at time t. As in Kazerouni et al. (2017), we rely on the following assumption.

Assumption 8. *There exists* $0 \leq \Delta_l \leq \Delta_h$ *and* $0 < \mu_l \leq \mu_h$ *such that for every* t *:*

$$\Delta_l \leq \Delta_{b_t}^t \leq \Delta_h \qquad \text{and} \qquad \mu_l \leq \mu_{b_t}^t \leq \mu_h$$

Asm. 8 ensures that the baseline policy has a minimum level of performance, which is reasonable since the baseline policy is the strategy currently used by default. Note that in MABs and linear bandits $\mu_l = \mu_h = \mu_b$ since the performance does not depend on a system context. The terms Δ_l and μ_h are not critical quantities in the regret bound and it is possible to take $\Delta_l = 0$, $\mu_h = 1$. We are now ready to state the following result for CLUCB2.

Theorem 3. *For any contextual linear bandit problem, under Asm. 6, 7, and 8,* CLUCB2 *satisfies the conservative condition with probability* $1 - \delta$ *and its regret can be bounded for any* $n > 0$ *with probability at least* $1 - \delta$ *by*

$$R_{\text{CLUCB2}}(n) \leq O\left(\sigma d \log\left(\frac{nD^2}{\lambda d}\right)\sqrt{n} + \frac{\Delta_h d^2}{(\alpha\mu_l)^2}\left(\sqrt{\lambda}B + \sigma\right)^2 \log\left(\frac{d^2\left(\sqrt{\lambda}B + \sigma\right)^2\sqrt{D_0}}{\sqrt{\delta}\alpha\mu_l}\right)^2\right)$$

where $D_0 := \max\left\{2D^2/\lambda, 3\right\}$.

This regret is of the same order as the bound for CLUCB.[10] While this shows that the changes made to CLUCB are "safe", we cannot prove a direct improvement to the regret performance, apart from better constants (the regret of CLUCB is at least half of the one of CLUCB2, see appendix). Notice also that in the MAB case, this is not even possible in general, as CUCB is already proved to match the lower bound (in a worst-case sense). However, a worst-case argument may be misleading in the ranking of the algorithms. The empirical validation reported in the experimental section will provide a more direct evidence of the improvement of CLUCB2 over CLUCB. In the rest of the section we analyze the parts of the regret that are most directly impacted by the two algorithmic changes in CLUCB2.

Discussion on martingale bound.

An interpretation for the $\sqrt{d}$-improvement comes from comparing (2.20) and (2.22). The minimization in (2.20) has a closed form solution given by $\langle\widehat{\theta}_t, x\rangle - \beta_t\|x\|_{V_t^{-1}}$, with $x := \sum_{i\in S_{t-1}} x_{i,a_i}$.[11] While $\langle\widehat{\theta}_t, x\rangle \approx \sum_{i\in S_{t-1}} r_{a_i}^i$ since $\widehat{\theta}_t$ solves the associated regularized least-square problem, $\beta_t\|x\|_{V_t^{-1}} = \widetilde{O}(\sigma\sqrt{d|S_{t-1}|})$, which is larger than the martingale term, which is of order $\widetilde{O}(\sigma\sqrt{|S_{t-1}|})$. The advantage of the martingale argument is that it avoids to explicitly use the linear structure of the reward by building a concentration for the sum of scalar values. As shown above, this allows to derive a bound independent from the dimensionality of the linear parametrization. Nonetheless, notice that in evaluating the quality of the next arm, a minimization over θ is still needed in (2.22), which brings

[10] Notice that there is a typo in Thm.6 in Kazerouni et al. (2017), as the denominator in the log term of K should be 1.
[11] We remove the contribution of the optimistic arm a_t since it is the same when using martingale or self-normalizing bound.

back the dependency on $\sqrt{d}$ (but on a much smaller term) in the regret analysis, which eventually prevents us from proving an explicit advantage in the final bound. A similar reasoning can be derived for the MAB case (see appendix).

Another interpretation for this $\sqrt{d}$-improvement can be seen when looking at the regret. We start bounding the regret as:

$$R_{\text{CLUCB2}}(n) \leq \sum_{t \in S_n} \left(\mu^t_{a^\star} - \mu^t_{a_t}\right) + |S^b_n|\Delta_h$$

In Kazerouni et al. (2017), the first term is upper-bounded by the standard regret of LinUCB, while the key step is to bound the regret $|S^b_n|\Delta_h$ incurred while playing the baseline. By exploiting the martingale bound, we can provide a tighter bound for $|S^b_n|$ compared to CLUCB. In Kazerouni et al. (2017) (after Eq. 19), the authors shows that $\alpha\mu_l|S^b_n| \lesssim 114d^2c_1^2/(\alpha\mu_l)$ where $c_1 = \sigma + \sqrt{\lambda}B$ (ignoring logarithmic terms). By exploiting the martingale bound, we can show that $\alpha\mu_l|S^b_n| \lesssim 10(\sigma + dc_1)/\sqrt{\alpha\mu_l} + 32(\sigma + dc_1)^2/(\alpha\mu_l)$ (see Eq. 2.126). This already shows that we have a linear term in d depending only on $1/\sqrt{\alpha}$ and a term quadratic in d as in CLUCB with a much smaller constant. This is a big improvement compared to CLUCB that shows that the martingale indeed provides a $\sqrt{d}$-improvement (and also in $1/\sqrt{\alpha}$). Finally, if we take a very loose upper bound we obtain a term that is smaller by a factor at least 2 compared to the one of CLUCB (formally we obtain that $\alpha\mu_l|S^b_n| \lesssim 48d^2c_1^2/(\alpha\mu_l)$).

Discussion on action selection. The second difference in CLUCB2 is the action selection process. Denote by $\widehat{\mu}_i$ the empirical mean of arm i, let $\mathcal{A}^{\text{UCB}}_t := \arg\max_{i\in[K]}\{\widehat{\mu}_i + \psi^{\text{UCB}}_t(i)\}$ be the set of UCB optimal arms, $\mathcal{A}^+_t := \{i \in [K] : \widehat{\mu}_i + \psi^{\text{UCB}}_t(i) \geq \mu^\star\}$ the set of optimistic arms and $\mathcal{C}_t$ the set of arms satisfying the conservative condition. At any time t we can define three events: $E_{1,t} = \{a_t \in \mathcal{A}^{\text{UCB}}_t \wedge a_t \in \mathcal{C}_t\}$, $E_{2,t} = \{a_t \neq b \wedge a_t \notin \mathcal{A}^{\text{UCB}}_t \wedge a_t \in \mathcal{C}_t\}$ and $E_{3,t} = \{a_t = b\}$. Following the two-step selection process of CUCB, only $E_{1,t}$ and $E_{3,t}$ can happen. In case $E_{1,t}$, the algorithm behaves like UCB, thus performing exploration that contributes to reduce the regret over time. In case $E_{3,t}$, the regret is equal to Δ_b and no "progress" is made on the exploration side, but it serves in building conservative budget for later steps. In CLUCB2, event $E_{2,t}$ happens when the UCB arm is not "safe" to play (i.e., $\mathcal{A}^{\text{UCB}}_t \notin \mathcal{C}_t$) but there are other arms that are *safe w.r.t. the baseline*. Interestingly, in this case a_t is indeed *optimistic* w.r.t. the baseline, i.e., $a_t \in \mathcal{C}_t$ and $\widehat{\mu}_i + \psi^{\text{UCB}}_t(i) \geq \mu^t_b$. In analogy with the OFU principle for $a^\star$, this is a good strategy for performing efficient exploration w.r.t. the baseline policy. One source of improvement comes when a_t is optimistic despite not being the UCB arm (i.e., $a_t \in \mathcal{A}^+_t$ and $a_t \notin \mathcal{A}^{\text{UCB}}_t$). In this case, the time step can be analyzed as in UCB, thus reducing the number of pulls $T_b(n)$ to the baseline arm and its impact to the regret. This is likely to happen in earlier phases of the learning process, where the UCB of all arms tend to be optimistic (i.e., $a_t \in \mathcal{A}^+_t$). Even when a_t is not optimistic w.r.t. $\mu^\star$, it may happen that $\mu_{a_t} > \mu_b$. In this case, while this step can be still considered as a "conservative", it would contribute for less than Δ_b regret. Unfortunately, it is difficult to provide theoretical evidence that such events happen often enough to provably reduce the regret. Nonetheless, the fact that the arm is optimistic w.r.t. the baseline (i.e., $\widehat{\mu}_i + \psi^{\text{UCB}}_t(i) \geq \mu_b$) is sufficient to guarantee that the new action selection strategy is never worse than CUCB.

We can provide an intuition of the impact of the new arm selection through a simple *example*. Consider the situation of $N > 3$ arms, such that $\mu_i > \mu_{i+1}$, for any i. Assume we know the variance of the arms and we use Bernstein inequality in building confidence intervals. Let $\sigma_2 = 0$ and $(\mu_i, \sigma_i)_{i>3}$ such that the probability of being safe and better than arm 2 is negligible. Then the regret can be decomposed by the normal LinUCB term (due to event E_1), the pull of an arm that is optimistic w.r.t. the baseline (i.e., event E_2) and the conservative play. The number of conservative play is thus further reduced due to event E_2, leading to a smaller contribution to the regret. Now, in this specific case, the arm played during event E_2 is w.h.p. always arm 2. Since it is better than the baseline, we have a further improvement to the regret. To conclude, in this example, CLUCB2 will have a regret strictly better than CLUCB.

2.2.3 Checkpoint-based Conservative Exploration

CLUCB2 is designed to get a tighter proxy for (2.18), but it is still required to be conservative at any time t. This requirement is often too strict in practice, where the conservative condition may be verified only at some "checkpoints" over time. We study the case where the checkpoints are equally spaced every T steps. We still assume that Asm. 8 holds and we redefine the conservative condition such that for some $\alpha \in [0, 1]$ and $T \in \mathbb{N}^\star$ a learning agent must satisfy

$$\forall k > 0, \qquad \sum_{t=1}^{kT} \mu^t_{a_t} \geq (1-\alpha) \sum_{t=1}^{kT} \mu^t_{b_t}, \tag{2.24}$$

which reduces to (2.18) for $T = 1$. Knowing that the conservative condition is checked every T steps provides the agent with a leeway that can be used to perform more exploration and possibly converge faster towards the optimal policy.

We first derive a conservative condition that can be evaluated at any time $t \in [kT + 1, (k + 1)T]$ of a phase $k \in \mathbb{N}$ in order to determine whether action a_t is safe. We build this condition such that when selecting an action a_t, we want to ensure that by playing the baseline arm until the next checkpoint (i.e., until $(k+1)T$) , the algorithm would meet the condition (2.24). Formally, at any step t, we replace (2.24) with

$$\sum_{i \in S_{t-1}} \mu^i_{a_i} + \sum_{i \in S^b_{t-1}} \mu^i_{b_i} + \mu^t_{a_t} + \alpha((k+1)T - t)\, \mu_l \geq (1-\alpha) \sum_{i=1}^{t} \mu^i_{b_i}, \tag{2.25}$$

where μ_l is as in Asm. 8 i.e a lower bound on the average reward of the baseline strategy.

We now modify CLUCB2 to satisfy this constraint. Changing the conservative condition impacts how the algorithm evaluates whether an action is "safe" or not (i.e., if selecting a specific action is compatible with the conservative condition), however the algorithm can still use the bound in (2.21) to lower bound the sum of the values of actions selected so far, and compute the conservative set at time t as

$$\begin{aligned} \mathcal{C}_t := \Bigg\{ a \in \mathcal{A}_t \setminus \{b_t\} \mid \max\Bigg\{ \sum_{i \in S_{t-1}} r^i_{a_i} - \psi_L(t), 0 \Bigg\} \\ + \sum_{i \in S^C_{t-1}} \mu^i_{b_i} + \max\Big\{ \min_{\theta \in \mathcal{C}_t} \langle \theta, x_{t,a} \rangle, 0 \Big\} + \alpha \left(((k+1)T - t)\, \mu_l \geq (1-\alpha) \sum_{i=1}^{t} \mu^i_{b_i} \right) \Bigg\}, \end{aligned}$$

and the arm to pull is obtained by solving the constrained problem (2.23). We now proceed by analyzing how this different conservative condition impacts the final regret of CLUCB2, which we rename CLUCB2T to stress the checkpoint-based conservative condition.

Theorem 4. *For any $\delta > 0$, with probability as least $1 - \delta$* CLUCB2T *satisfies condition (2.24) at every checkpoint. Furthermore, let $\tilde{T}^\alpha_\delta := \frac{\alpha\mu_l}{(1-\alpha)\mu_h + \alpha\mu_l} T$, then* CLUCB2T *suffers a regret*

- *If $\tilde{T}^\alpha_\delta \geq C_b(\alpha, \mu_l, \delta)$*

$$R(n) \leq O\left(\sigma d \log\left(\frac{nD^2}{\lambda d} \right) \sqrt{n} + \frac{\Delta_h}{\alpha\mu_l} \max\left\{ d\big(\sqrt{\lambda}B + \sigma\big) \sqrt{\tilde{T}^\alpha_\delta \log\left(\frac{\tilde{T}^\alpha_\delta D^2}{\lambda\delta} \right)} + \mu_h - \frac{\alpha\mu_l}{2} \tilde{T}^\alpha_\delta, 0 \right\} \right),$$

- *otherwise*

$$R(n) \leq O\left(\sigma d \log\left(\frac{nD^2}{\lambda d} \right) \sqrt{n} + \frac{\Delta_h d^2}{(\alpha\mu_l)^2} \big(\sqrt{\lambda}B + \sigma\big)^2 \log\left(\frac{d^2\sqrt{D_0}}{\sqrt{\delta}\alpha\mu_l} \big(\sqrt{\lambda}B + \sigma\big)^2 \right)^2 \right),$$

where

$$C_b(\alpha, \mu_l, \delta) := 28d^2 \left(\frac{2/3 + \sqrt{\lambda}B + 2\sigma}{\alpha\mu_l} \right)^2 \ln\left(\frac{696 d^2 D_0}{\delta\, (\alpha\mu_b)^2} \left(2/3 + \sqrt{\lambda}B + 2\sigma \right)^2 \right)^2,$$

with $D_0 := \max\{3, 2D^2/\lambda\}$.

This bound illustrate how the length T of each phase may significantly simplify the problem. As T gets larger, satisfying condition (2.24) becomes easier, the baseline is selected less often and more time is spent in exploring different actions, thus leading to smaller regret. Interestingly, when T is large enough (i.e., $T \geq C_b(\alpha, \mu_l, \delta)$), the conservative contribution has a smaller and smaller impact onto the regret, to the point that the *max* in the second term can become 0, thus reducing the regret to the standard regret bound of LINUCB, with no impact from the conservative constraint.

Alternative checkpoint schemes. While we assumed T to be fixed, it is possible to generalize this result along different lines. If the time between any two checkpoints is known to be lower-bounded by $T_{\min}$, the same analysis could hold by replacing T with $T_{\min}$. Similarly, if T is random from a known distribution, then it is possible to compute the $1-\delta/2$ quantile of T to recover high-probability guarantees on the conservative properties of the algorithm. Finally, if the checkpoints are completely arbitrary (or even adversarially chosen) then in order to guarantee that (2.24) is verified at *all* checkpoints, the agent needs to be conservative at every step, thus reducing to condition (2.18).

2.2.4 Experiments

In this section we provide empirical evidence of the advantage of the Martingale lower-bound and the action selection process in synthetic and real-data problems.

2.2.4.1 Synthetic Environments

We consider a MAB with $K = 10$ Bernoulli arms with means drawn from a uniform distribution, $\mu_i \sim \text{Uniform}([0.25, 0.75])$. The conservative level α is set to 0.05, the horizon n to 10^6 ($T = 1$) and $\delta = 0.01$. We generated 70 different Bernoulli bandit problems (i.e., values of μ_i) and we performed 40 simulations for each. In each problem, we selected the 4^{th} best arm as baseline. Out of the 70 problems, we report the regret curves for the instance where the advantage of CUCB2 w.r.t. CUCB in terms of the average regret at n is the smallest. This provides an estimated worst-case scenario for our comparison (see Appendix for further details and results). We report the performance of UCB and an oracle variant of CUCB (CUCB-Or) where the conservative condition (2.18) is checked exactly. Furthermore, we test CUCB and a variant of CUCB (CUCB-L) using the action selection process suggested in Wu et al. (2016), which returns the safe arm with the largest lower bound . Finally, we report an ablation study for CUCB2, where we consider the Martingale lower bound (CUCB-M) and the constrained action selection process (2.23) (CUCB-S) separately beside the full algorithm (CUCB2). Fig. 2.5(*top*) shows that the MDS bound alone provides a significant improvement, where the regret is reduced by 43% w.r.t. CUCB's. Interestingly, the action selection process (CUCB-S) is much more effective than CUCB-L and it reduces the regret of CUCB by 12%. Finally, the combination of the two elements (CUCB2) leads to a reduction of the original regret of more than 51%, with a performance which gets much closer to CUCB-Or.

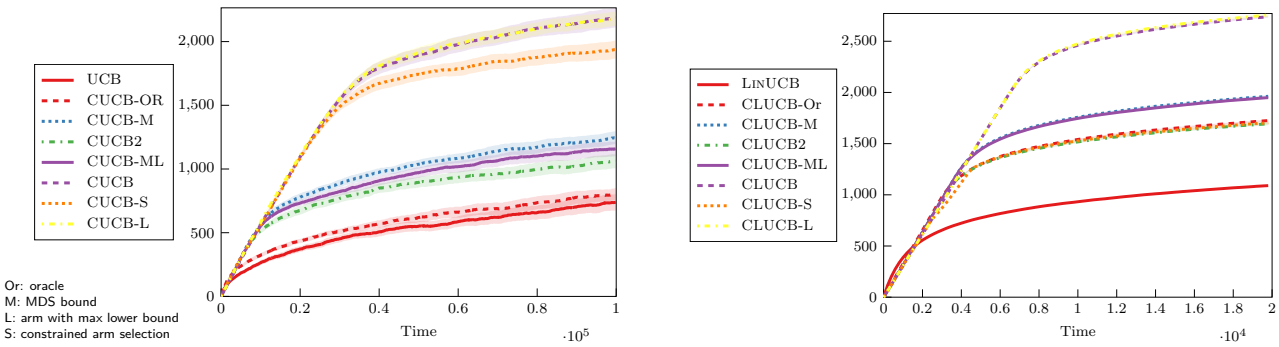


Figure 2.5: Cumulative regret in synthetic models. *Left:* Bernoulli arms. *Right:* linear bandits.

We also evaluated CLUCB2 in the linear setting. We considered a non-contextual case with 30 actions, each defined by a 100-dimensional feature vector. The features and $\theta^\star$ are drawn randomly in the unit ball such that the mean reward of each arm is in $[0, 1]$. The reward noise is drawn from $\mathcal{N}(0, 0.1^2)$ and the baseline arm is the 6^{th} best action. We set $\lambda = 0.5$, $\delta = 0.01$ and $\alpha = 0.05$. We generated 70 models and for each model we averaged the results over 40 runs. As in the MAB case, we report the results for the model with the smallest advantage for CLUCB2 w.r.t. CLUCB (Fig. 2.5(*bottom*)). Contrary to the MAB setting, the main improvement is obtained by CLUCB-S, whose performance matches CLUCB2 and even the oracle variant, corresponding to an improvement of 38% compared to CLUCB. As reported in the appendix in the average case (over models), we observe similar behaviors and performance improvements as in the MAB setting. Finally, to provide an idea of how many times event E_2 occurs, we performed tests in synthetic linear setting (see experiment section) with baseline being the third-best arm. On average over multiple models, the percentage of time event E_2 happened in the first 5000 steps (25% of

overall time) is 38.7% ($\pm 9\%$). This shows that potentially we have played something better than baseline and for sure we have gained information (in contrast to playing the baseline).

Checkpoint-based Condition. We compare the effect of the checkpoint T on the regret of the algorithms. We report the results for Bernoulli arms, the linear experiments can be found in Appendix. In this case, the horizon is set to $n = 20000$, all the other parameters are unchanged. We generated 15 (integer) checkpoint values logarithmic space between 1 and n. Fig. 2.6 shows the difference in the regret between $\mathrm{CUCB2T}$ and UCB as a function of T. As expected, the difference decreases as T increases since the condition becomes less strong. Note that even for $T = n$, $\mathrm{CUCB2T}$ and UCB are different since $\mathrm{CUCB2T}$ might discard UCB optimal arms in order to be safe. In order to have the same behavior, T should be put sufficiently large in order to overcome the pessimistic estimate used in the condition. Finally, the improvement provided by the new condition is proportional to the quality of the baseline. The stronger the baseline, the less is the margin for performing better exploration than playing the baseline itself.

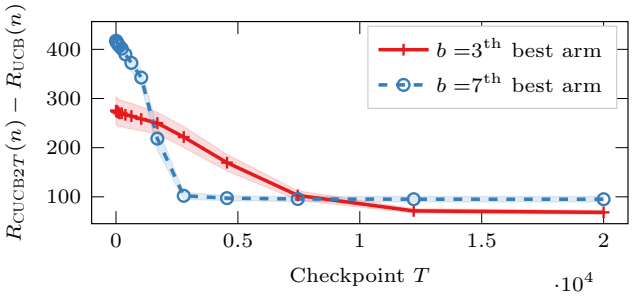


Figure 2.6: Relative performance between $\mathrm{CUCB2T}$ and UCB in synthetic MAB setting.

2.2.4.2 Dataset-based Environments

Fig. 2.7 reports the results using the Jester Dataset Goldberg et al. (2001) that consists of joke ratings in a continuous scale from -10 to 10 for 100 jokes from a total of 73421 users. We consider the cold start problem: a new user arrives and we need to learn her preferences (i.e., $\theta^\star$). We use the features extracted via a low-rank matrix factorization ($d = 35$) to represents the actions (i.e., the jokes). We consider a complete subset of 40 jokes and 19181 users rating all the 40 jokes. The preference of the new user is randomly selected from the 19181 users and mean rewards are normalized in $[0, 1]$. The reward noise is $\mathcal{N}(0, 0.1^2)$, the horizon is $T = 10^5$, $\alpha = 0.01$, $\delta = 0.01$ and $\lambda = 0.5$ (see App. 2.B.2). We report the results averaged over 100 randomly selected users and for each user we performed 5 runs. The baseline is the 10^{th} best arm. We also report the regret of $\mathrm{CLUCB2T}$ with a checkpoint horizon T equal to $5\%, 10\%$ or 12% of the horizon n. This experiment confirms that $\mathrm{CLUCB2}$ performs best, with a regret that is less than half of CLUCB. Furthermore, the results confirm that as the checkpoints become sparser, the performance of $\mathrm{CLUCB2}$ approaches the one of $\textsc{LinUCB}$.

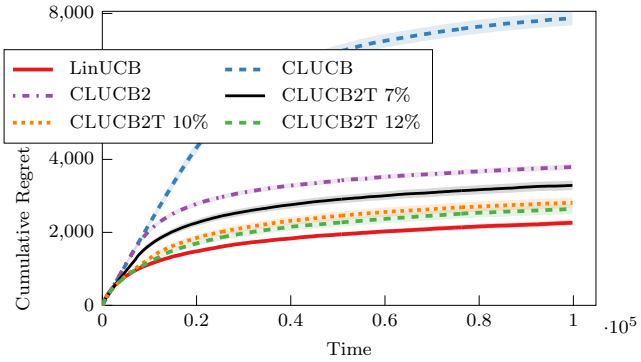


Figure 2.7: Average regret over multiple users of the Jester dataset

2.2.5 Potential Extensions and Concluding Remarks

We introduced CLUCB2, a novel conservative exploration algorithm for linear bandit that matches existing regret bound and outperforms state-of-the-art algorithms in a number of empirical tests. In this section, we also proposed a first direction to relax the conservative condition towards a more realistic scenario.

Important directions for future work are: identify alternative conservative exploration constraints that are directly motivated by specific applications, extend the current algorithms beyond linear bandit towards the more challenging reinforcement learning setting.

So far conservative exploration has been defined in the bandit setting, however in the next section we show that the definition can be extended to the Reinforcement Learning setting both in the average reward and finite-horizon setting.

2.3 Conservative Exploration in Reinforcement Learning

While Reinforcement Learning (RL) has achieved tremendous successes in simulated domains, its use in real system is still rare. A major obstacle is the lack of guarantees on the learning process, that makes difficult its application in domains where hard constraints (e.g., on safety or performance) are present. Examples of such domains are digital marketing, healthcare, finance, and robotics. For a vast number of domains, it is common to have a known and reliable *baseline policy* that is potentially suboptimal but satisfactory. Therefore, for applications of RL algorithms, it is important that are guaranteed to perform at least as well as the existing baseline.

In the offline setting, this problem has been studied under the name of *safety w.r.t. a baseline* (Bottou et al., 2013; Thomas et al., 2015a,b; Swaminathan and Joachims, 2015; Petrik et al., 2016; Laroche et al., 2019; Simão and Spaan, 2019). Given a set of trajectories collected with the baseline policy, these approaches aim to learn a policy –without knowing or interacting with the MDP– that is guaranteed (e.g., w.h.p.) to perform at least as good as the baseline. This requires that the set of trajectories is sufficiently reach in order to allow to perform counterfactual reasoning with it. This often implies strong requirements on the ability of exploration of the baseline policy. These approaches can be extended to a semi-batch settings where phases of offline learning are alternated with the executing of the improved policy. This is the idea behind conservative policy iteration (e.g., Kakade and Langford, 2002; Pirotta et al., 2013b) where the goal is to guarantee a monotonic policy improvement in order to overcome the policy oscillation phenomena Bertsekas (2011). These approaches has been successively extended to function approximation preserving theoretical guarantees (e.g., Pirotta et al., 2013a; Achiam et al., 2017). A related problem studied in RL is the one of safety, where the algorithm is forced to satisfy a set of constraints, potentially not directly connected with the performance of a policy (e.g., Altman, 1999; Berkenkamp et al., 2017; Chow et al., 2018).

In the online setting, which is the focus of this paper, the learning agent needs to trade-off exploration and exploitation while interacting with the MDP. Opposite to offline learning, the agent has direct control over exploration. Exploration means that the agent is willing to give up rewards for policies improving his knowledge of the environment. Therefore, there is no guarantee on the performance of policies generated by the algorithm, especially in the initial phase where the uncertainty about the MDP is maximal and the algorithm has to explore multiple options (almost randomly). To increase the application of exploration algorithm, it is thus important that the policies selected by the algorithm are (cumulatively) guaranteed to perform as well as the baseline by making exploration more *conservative*. This setting has been studied in multi-armed bandits (Wu et al., 2016), contextual linear bandits (Kazerouni et al., 2017), and stochastic combinatorial semi-bandits (Katariya et al., 2019). These papers formulate the problem using a constraint defined based on the performance of the baseline policy (mean of the baseline arm in the multi-armed bandit case), and modify the corresponding UCB-type algorithm (Auer et al., 2002b) to satisfy this constraint. Another algorithm in the online setting is by (Mansour et al., 2015) that balances exploration and exploitation such that the actions taken are compatible with the agent's (customer's) incentive formulated as a Bayesian prior.

While the conservative exploration problem is well-understood in bandits, little is known about this setting to RL, where the actions taken by the learning agent affect the system state. This dynamic component makes the definition of the conservative condition much less obvious in RL. While in the bandit case it is sufficient to look at (an estimate of) the immediate reward to perform a conservative decision, in MDPs acting greedily may not be sufficient since an action can be "safe" in a single step but lead to a potentially dangerous state space where it will not be possible to satisfy the conservative constraint. Moreover, after t steps, the action followed by the learning agent may lead to a state that is possibly different from the one observed by following the baseline. This dynamical aspect is not captured by the bandit problem and should be explicitly taken into account by the learning agent in order to perform

a meaningful decision. This, together with the problem of counterfactual reasoning in an unknown MDP, make the conservative exploration problem is much more difficult (and interesting) in RL than in bandits.

This paper aims to provide the first analysis of conservative exploration in RL. In Sec. 2.3.2 we explain the design choices that lead to the definition of the conservative condition for RL (both in average reward and finite horizon settings), and discuss all the issues introduced by the dynamical nature of the problem. Then, we provide the first algorithm for efficient conservative exploration in average reward and analyze its regret guarantees. The variant for finite-horizon problems is postponed to the appendix. We conclude the paper with synthetic experiments.

2.3.1 Average Reward Reinforcement Learning

In this section, we consider the Average Reward Reinforcement Learning setting defined in Section 1.2.3.2 $M = (\mathcal{S}, \mathcal{A}, p, r)$ with state space $\mathcal{S}$ and action space $\mathcal{A}$.

Exploration in RL. Let $M^\star$ be the true *unknown* MDP. We consider the learning problem where $\mathcal{S}$, $\mathcal{A}$ and $r_{\max}$ are *known*, while rewards r and transition probabilities p are *unknown* and need to be estimated online. We evaluate the performance of a learning algorithm $\mathfrak{A}$ after T time steps by its cumulative *regret*

$$R(\mathfrak{A}, T) = Tg^\star - \sum_{t=1}^{T} r_t(s_t, a_t). \tag{2.26}$$

The exploration-exploitation dilemma is a well-known problem in RL and (nearly optimal) solutions have been proposed in the literature both base on optimism-in-the-face-of-uncertainty (OFU, e.g., Jaksch et al., 2010a; Bartlett and Tewari, 2009; Fruit et al., 2018a) and Thompson sampling (TS, e.g., Gopalan and Mannor, 2015; Osband and Roy, 2016). Refer to (Lazaric et al., 2019) for more details.

2.3.2 Definition of Conservative Exploration in Average Reward RL

In conservative exploration, a learning agent is expected to perform as well as the optimal policy over time (i.e., regret minimization) under the *constraint* that at no point in time its performance is significantly worse than a known baseline policy $\pi_b \in \Pi^{\mathrm{SR}}$. This problem has been studied in the bandit literature (Wu et al., 2016; Kazerouni et al., 2017), where the conservative constraint compares the cumulative *expected* reward obtained by the actions $a_1, a_2, \dots, a_t$ selected by the algorithm to the one of the baseline action a_b,

$$\forall t > 0, \quad \sum_{i=1}^{t} r(a_i) \geq (1-\alpha)\, t\, r(a_b), \tag{2.27}$$

where $r(a)$ is the expected reward of action a. At any time t, conservative exploration algorithms first query a standard regret minimization algorithm (e.g., UCB) and decide whether to play the proposed action $\widetilde{a}_t$ or the baseline a_b based on the accumulated budget (i.e., past rewards) and whether the estimated performance of $\widetilde{a}_t$ is sufficient to guarantee that the conservative constraint is satisfied at $t+1$ after $\widetilde{a}_t$ is executed. While (2.27) effectively formalizes the objective of constraining an algorithm to never perform much worse than the baseline, in RL it is less obvious how to define such constraint. In the following we review three possible directions, we point out their limitations, and we finally propose a conservative condition for RL for which we derive an algorithm in the next section.

Gain-based condition. Instead of actions, RL exploration algorithms (e.g., $\mathrm{UCRL2}$), first select a policy and then execute the corresponding actions. As a result, a direct way to obtain a conservative condition is to translate the reward of each action in (2.27) to the *gain* associated to the policies selected over time, i.e.,

$$\forall t > 0, \quad \sum_{i=1}^{t} g^{\pi_i} \geq (1-\alpha)\, t\, g^{\pi_b}. \tag{2.28}$$

The main drawback of this formulation is that the gain g^{π_i} is the expected *asymptotic* average reward of a policy and it may be very far from the *actual* reward accumulated while executing π_i in the specific state s_i achieved at time i. The same reasoning applies to the baseline policy, whose cumulative reward up to time t may significantly different from t times its gain. As a result, an algorithm that is conservative in the sense of Eq. (2.28) may still perform quite poorly in practice depending on t, the initial state, and the actual trajectories observed over time.

Reward-based condition. In order to address the concerns about the gain-based condition, we could define the stronger condition

$$\forall t > 0, \quad \sum_{i=1}^{t} r_i \geq (1-\alpha) \sum_{i=1}^{t} r_i^b, \tag{2.29}$$

where r_i is the sequence of rewards obtained while executing the algorithm and r_i^b is the reward obtained by the baseline. While this condition may be desirable in principle (the learning algorithm never performs worse than baseline), it is impossible to achieve. In fact, even if the optimal policy $\pi^\star$ is executed for all t steps, the condition may still be violated because of an unlucky realization of transitions and rewards. If we wanted to accounting for the effect of randomness, we would need to introduce an additional slack of order $O(\sqrt{t})$ (i.e., the cumulative deviation due to the randomness in the environment), which would make the condition looser and looser over time.

Condition in expectation. The previous remarks could be solved by taking the expectation of both sides

$$\forall t > 0, \ \mathbb{E}_{\mathfrak{A}} \left[\sum_{i=1}^{t} r_i(s_i, a_i) \Big| s_1 = s \right] \geq (1-\alpha) \mathbb{E} \left[\sum_{i=1}^{t} r_i(s_i, a_i) \Big| s_1 = s, \pi_b \right], \tag{2.30}$$

where $\mathbb{E}_{\mathfrak{A}}$ denotes the expectation w.r.t. the trajectory of states and actions generated by the learning algorithm $\mathfrak{A}$, while the RHS is simply the expected reward obtain by running the baseline for t steps. Condition (2.30) effectively captures the nature of the RL problem w.r.t. the bandit case. In fact, after t steps, the actions followed by the learning algorithm may lead to a state that is possibly very different from the one we would have reached by playing only the baseline policy from the beginning. This deviation in the state dynamics needs to be taken into account when deciding if an exploratory policy is safe to play in the future. In the bandit case, selecting the baseline action contributes to *build a conservative budget* that can be *spent* to play explorative actions later on (i.e., by selecting a_b, the LHS of (2.27) in increased by $r(a_b)$, while only a fraction $1-\alpha$ is added to the RHS, thus increasing the margin that may allow playing alternative actions later). In the RL case, selecting policy π_b at time t may not immediately contribute to increasing the conservative budget. In fact, the state s_t where π_b is applied may significantly differ from the state that π_b *would have achieved* had we selected it from the beginning. As a result, a conservative RL algorithm should be extra-cautious when selecting policies different from π_b since their execution may lead to unfavorable states, where it is difficult to recover good performance, even when selecting the baseline policy.

While this may seem a reasonable requirement, unfortunately it is impossible to build an empirical estimate of (2.30) that a conservative exploration algorithm could use to guide the choice of policies to execute. In fact, the LHS *averages* the performance of the algorithm over multiple executions, while in practice we have only access to a single realization of the algorithm's process. This prevents from constructing accurate estimates of such expectation directly from the data observed up to time t. A possible approach would be to construct an estimate of the MDP and use it to *replay* the algorithm itself for t steps. Beside prohibitive computational complexity, the resulting estimate of the expected cumulative reward of $\mathfrak{A}$ would suffer from an error that increases with t, thus making it a poor proxy for (2.30).[12]

Condition with conditional expectation. Let t be a generic time and $\mu_t = (\pi_1, \pi_2, \dots, \pi_t)$, the non-stationary policy executed up to t. We require the algorithm to satisfy the following *conditional* conservative condition

$$\forall t > 0, \ \mathbb{E} \left[\sum_{i=1}^{t} r_i(s_i, a_i) | s_1 = s, \mu_t \right] \geq (1-\alpha) \mathbb{E} \left[\sum_{i=1}^{t} r_i(s_i, a_i) | s_1 = s, \pi_b \right]. \tag{2.31}$$

where the expectations are taken w.r.t. the trajectories generated by a *fixed* non-stationary policy μ_t (i.e., we ignore how rewards affect μ_t). Notice that this condition is now stochastic, as μ_t itself is a random variable and thus we require to satisfy (2.31) with *high probability*. This formulation can be seen as relying on a *pseudo*-performance evaluation of the algorithm instead of the actual expectation as in (2.30)[13] and it is similar to (2.27), which takes the expected performance of each of the (random) actions, thus ignoring their correlation with the rewards. This formulation has several advantages w.r.t. the conditions proposed above: **1)** it considers the sum of rewards rather than the gain as (2.28), thus capturing the dynamical nature of RL, **2)** it contains expected values, so as to avoid penalizing the algorithm by unlucky noisy realizations as (2.29), **3)** as shown in the next section, it can be verified using the samples observed by the algorithm unlike (2.30).

[12] More precisely, let $\widehat{M}_t$ be an estimate of $M^\star$ and ϵ_t be the largest error in estimating its dynamics at time t. Estimating the expected cumulative reward by running (an infinite number of) simulations of $\mathfrak{A}$ in $\widehat{M}_t$ would suffer from an error scaling as $t\epsilon_t$. For any regret minimization algorithm, ϵ_t cannot decrease linearly with t and thus the estimation of $\mathbb{E}_{\mathfrak{A}}$ would have an error increasing with t.

[13] We use *pseudo*-performance to stress the link the *pseudo*-regret formulations used in bandit (e.g., Auer et al., 2002b)

Algorithm 4: CUCRL2 algorithm.

Input: $\pi_b \in \Pi^{\mathrm{SR}}$, $\delta \in (0,1)$, $r_{\max}$, $\mathcal{S}$, $\mathcal{A}$, $\alpha \in (0,1)$
Set $S_0 = S_0^b = \emptyset$, $k = 0$
for *episodes* $k = 1, 2, \ldots$ **do**
  Set $t_k = t$ and episode counters $\nu_k(s,a) = 0$.
  Compute estimates $\widehat{p}_k(s'|s,a)$, $\widehat{r}_k(s,a)$ and a confidence set $\mathcal{M}_k$.
  Compute an $r_{\max}/\sqrt{t_k}$-approximation $\widetilde{\pi}_k$ of the optimistic planning problem $\max_{M \in \mathcal{M}_k, \pi \in \Pi^{\mathrm{SD}}} \{g^\pi(M)\}$.
  Compute $(g_k^-, h_k^-) = \mathrm{EVI}(\mathcal{L}_k^{\widetilde{\pi}_k}, r_{\max}/\sqrt{t_k})$, see Eq. 2.35.
  Compute an $r_{\max}/\sqrt{t_k}$-approximation $\widetilde{\pi}_k$ of the optimistic planning problem $\max_{M \in \mathcal{M}_k, \pi \in \Pi^{\mathrm{SD}}} \{g^\pi(M)\}$.
  if *Eq. 2.39 is true* **then**
    $\pi_k = \widetilde{\pi}_k$ **else** $\pi_k = \pi_b$
  Sample action $a_t \sim \pi_k(\cdot|s_t)$.
  while $\nu_k(s_t, a_t) \le N_k^+(s_t, a_t) \wedge t \le t_k + T_{k-1}$ **do**
    Execute a_t, obtain reward r_t, and observe s_{t+1}.
    Set $\nu_k(s_t, a_t) = \nu_k(s_t, a_t) + 1$.
    Sample action $a_{t+1} \sim \pi_k(\cdot|s_{t+1})$ and set $t = t + 1$.
  Set $N_{k+1}(s,a) = N_k(s,a) + \nu_k(s,a)$, $\Lambda_k = \Lambda_{k-1} \cup \{k\} \cdot \mathbb{1}_{(Eq.\ 2.39)}$ and $\Lambda_k^c = \Lambda_{k-1}^c \cup \{k\} \cdot \mathbb{1}_{(\neg Eq.\ 2.39)}$

The finite-horizon case. We conclude the section, by reformulating (2.31) in the finite-horizon case. In this setting, the learning agent interacts with the environment in episodes of fixed length H. Let $\bar{s}$ be the initial state, π_j be the policy proposed at episode j and let $t = (k-1)H + 1$ be beginning of the k-th episode. Then μ_t is a sequence of policies π_j, each executed for H steps. In this case, condition (2.31) can be conveniently written as

$$(1-\alpha)kV_1^{\pi_b}(\bar{s}) \le \mathbb{E}\left[\sum_{i=1}^{t} r_i(s_i,a_i)|s_1 = s, \mu_t\right] = \sum_{j=1}^{k} \mathbb{E}\left[\sum_{i=1}^{H} r_i^j(s_i^j, a_i^j)|s_1 = s, \pi_j\right] = \sum_{j=1}^{k} V_1^{\pi_j}(\bar{s}) \tag{2.32}$$

where V_1^π is the H step value function of π at the first stage. In this formulation, the conservative condition has a direct interpretation, as it directly mimics the bandit case (2.27). In fact, the performance of the algorithm up to episode k is simply measured by the sum of the value functions of the policies executed over time (each for H steps) and it is compared to the value function of the baseline itself. Note that this definition is compatible with the regret: $R_{\mathrm{FH}}(\mathfrak{A}, K) = \sum_{k=1}^{K} V^\star(\bar{s}) - V^{\pi_k}(\bar{s})$. Indeed, the regret defined in expectation w.r.t. the stochasticity of the model but not w.r.t. the algorithm, there is no expectation w.r.t. the possible sequence of policies generated by $\mathfrak{A}$.

2.3.3 Conservative UCRL

In this section, we introduce *conservative upper-confidence bound for reinforcement learning* (CUCRL2), an efficient algorithm for exploration-exploitation in average reward that both minimize the regret (2.26) and satisfy condition (2.31).

CUCRL2 builds on UCRL2 in order to perform efficient *conservative* exploration. At each episode k, CUCRL2 builds a bounded parameter MDP $\mathcal{M}_k = \{M = (\mathcal{S}, \mathcal{A}, r, p), r(s,a) \in B_r^k(s,a), p(\cdot|s,a) \in B_p^k(s,a)$, where $B_r^k(s,a) \in [0, r_{\max}]$ and $B_p^k(s,a) \in \Delta_S$ are high-probability confidence intervals on the rewards and transition probabilities such that $M^\star \in \mathcal{M}_k$ w.h.p. and Δ_S is the S-dimensional simplex. This confidence intervals can be built using Hoeffding or empirical Bernstein inequalities by using the samples available at episode k (e.g., Jaksch et al., 2010a; Fruit et al., 2018b). CUCRL2 computes an optimistic policy $\widetilde{\pi}_k$ in the same way as UCRL2: $(\widetilde{M}_k, \widetilde{\pi}_k) \in \arg\max_{M \in \mathcal{M}_k, \pi \in \Pi^{\mathrm{SD}}} \{g^\pi(M)\}$. This problem can be solved using EVI (see Fig. 8 in appendix) on the optimistic optimal Bellman operator $\mathcal{L}_k^+$ of $\mathcal{M}_k$ (Jaksch et al., 2010a).[14] Then, it needs to decide whether policy π_k is "safe" to play by checking a conservative condition $f_c(\mathcal{H}_k)$ (see Sec. 2.3.3.1) where $\mathcal{H}_k$ contains all the information (samples and chosen policies) available at the beginning of episode k, including the optimistic policy π_k. If $f_c(\mathcal{H}_k) \ge 0$, the UCRL2 policy $\widetilde{\pi}_k$ is "safe" to play and CUCRL2 plays $\pi_k = \widetilde{\pi}_k$ until the end of the episode. Otherwise, CUCRL2 executes the baseline π_b, i.e., $\pi_k = \pi_b$. We denote by Λ_k the set of episodes (k included) where

[14] $\mathcal{L}_k^+ v(s) = \max_a \{\max_{r \in B_k^r(s,a)} \{r\} + \max_{p \in B_k^p(s,a)} p^\mathsf{T} v\}$.

UCRL2 executed an optimistic policy and by $\Lambda_k^c = \{1, \dots, k\} \setminus \Lambda_k$ its complement. Formally, if $f_c(\mathcal{H}_k) \geq 0$ we set $\Lambda_k = \Lambda_{k-1} \cup \{k\}$ else $\Lambda_k = \Lambda_{k-1}$. The pseudocode of CUCRL2 is reported in Alg. 4.

Note that, contrary to what happens in conservative (linear) bandits, the statistics of the algorithm are updated continuously, i.e., using also the samples collected by running the baseline policy. This is possible since UCRL2 is a model-based algorithm and any off-policy sample can be used to update the estimates of the model. To have a better estimate of the conservative condition, it is possible to use the model available at episode k to re-evaluate the policies $(\pi_l)_{l<k}$ at previous episodes (change line 3 in Alg. 4). This will improve the empirical performance of CUCRL2 but breaks the regret analysis.

2.3.3.1 Algorithmic Conservative Condition

We now derive a *checkable* conservative condition that can be incorporated in the UCRL2 structure illustrated in the previous section. In the bandit setting, it is relatively straightforward to turn (2.27) into a condition that can be checked at any time t using estimates and confidence intervals build from the data collected so far. On the other hand, while condition (2.31) effectively formalizes the requirement that the learning algorithm should constantly perform almost as well as the baseline policy, we need to consider the specific RL structure to obtain a condition that can be verified *during the execution* of the algorithm itself. In order to simplify the derivation, we rely on the following assumption.

Assumption 9. *The gain and bias function (g^{π_b}, h^{π_b}) of the baseline policy are known.*

As explained in (Kazerouni et al., 2017), this is a reasonable assumption since the baseline policy is assumed to be the policy currently executed by the company and for which historical data are available. We will mention how to relax this assumption in Sec. 2.3.3.1.

We follow two main steps in deriving a checkable condition. **1)** We need to estimate the cumulative reward obtained by each of the policies played by the learning algorithm directly from the samples observed so far. We do this by relating the cumulative reward to the gain and bias of each policy and then building their estimates. **2)** It is necessary to evaluate whether the policy proposed by UCRL2 is safe to play w.r.t. the conservative condition, before actually executing it. While this is simple in bandit, as each action is executed for only one step. In RL, policies cannot be switched at each step and need to be played for a *whole* episode. Nonetheless, the length of a UCRL2 episode is not known in advance and this requires predicting for how long the explorative policy could be executed in order to check its performance.

Step 1: Estimating the conditional conservative condition from data. In order to evaluate (2.31) from data, one may be tempted to first replace the sum of rewards obtained by each policy π_j in μ_t on the lhs side by its gain g^{π_j}, similar to the gain-based condition in (2.28). Indeed, under Asm. 1 any stationary policy π receives *asymptotically* an expected reward g^π at each step. Unfortunately, in our case $\mathbb{E}\left[\sum_{i=1}^t r_i | \mu_t\right] \neq \sum_{j=1}^k T_j g^{\pi_j}$. In fact, when evaluating a policy for a finite number of steps, we need to account for the time required to reach the steady regime (i.e., mixing time) and, as such, the influence of the state at which the policy is started. The notion of reward collected during the transient regime is captured by the bias function.[15] In particular, for any stationary (unichain) policy $\pi \in \Pi^{\mathrm{SR}}$ with gain g^π and gain function h^π executed for t steps, we have that:

$$\mathbb{E}\left[\sum_{i=1}^t r_i | s_1 = s, \pi\right] = t\, g^\pi + h^\pi(s) - P_\pi^t(\cdot|s)^\mathsf{T} h^\pi. \tag{2.33}$$

As a result, we have the bounds

$$t\, g^\pi - sp(h^\pi) \leq \mathbb{E}\left[\sum_{i=1}^t r_i | s_1 = s, \pi\right] \leq t\, g^\pi + sp(h^\pi).$$

Leveraging prior knowledge of the gain and bias of the baseline, we can use the second inequality to directly upper bound the baseline performance as

$$\mathbb{E}\left[\sum_{i=1}^t r_i | s_1 = s, \pi_b,\right] \leq sp(h^{\pi_b}) + t\, g^{\pi_b}. \tag{2.34}$$

[15] Puterman (1994, Sec. 8.2.1) refers to the gain as "stationary" reward while to the bias as "transient" reward.

On the other hand, for a generic policy π, the gain and bias cannot be directly computed since $M^\star$ is unknown. To estimate the cumulative reward of the algorithm we resort to the estimate of the true MDP build by UCRL2 to construct a pessimistic estimate of the cumulative reward for any policy π_j (i.e., to perform counterfactual reasoning).

Given a policy π and the bounded-parameter MDP $\mathcal{M}_k$, we are intersted in finding $\underline{g}^\pi$ where $\underline{g}^\pi := \min_{M\in\mathcal{M}_k}\{g^\pi(M)\}$. Define the Bellman operator $\mathcal{L}_k^\pi$ associated to $\mathcal{M}_k$ as: $\forall v \in \mathbb{R}^S, \forall s \in \mathcal{S}$

$$\mathcal{L}_k^\pi v(s) := \min_{r\in B_r^k(s,a)} r + \min_{p\in B_p^k(s,a)}\{p^\mathsf{T} v\} \tag{2.35}$$

Then, there exists $(\underline{g}^\pi, \underline{h}^\pi)$ such that, $\forall s \in \mathcal{S}$, $\underline{g}^\pi e + \underline{h}^\pi = \mathcal{L}_k^\pi \underline{h}^\pi$ where $e = (1,\dots,1)$ (see Lem. 15.*1* in App. 2.C.1). Similarly to what is done by UCRL2, we can use EVI with $\mathcal{L}_k^\pi$ to build an ϵ_k-approximate solution of the Bellman equations. Let $(g_n, v_n) = \mathrm{EVI}(\mathcal{L}_k^\pi, \varepsilon_k)$, then $g_n - \varepsilon_k \le \underline{g}^\pi \le g^\pi(M^\star)$. The values computed by the pessimistic policy evaluation can be then used to bound the cumulative reward of any stationary policy.

Lemma 3. *Consider a bounded parameter MDP $\mathcal{M}$ such that $M^\star \in \mathcal{M}$ w.h.p., a policy π and let $(g_n, v_n) = \mathrm{EVI}(\mathcal{L}^\pi, \varepsilon)$. Then, under Asm. 1 for any state $s \in \mathcal{S}$:*

$$\mathbb{E}\left[\sum_{i=1}^t r_i | s_1 = s, \pi\right] \ge t(g_n - \varepsilon) - sp(v_n).$$

Step 2: Test safety of optimistic policy. Let t_k be the time when episode k starts. Policies $\pi_1,\dots,\pi_{k-1}$ have been executed until $t_k - 1$ and UCRL2 computed an optimistic policy $\widetilde{\pi}_k$. In order to guarantee that (2.31) is verified the algorithm needs to anticipate how well $\widetilde{\pi}_k$ may perform if executed for the next episode. For any policy $(\pi_j)_{j<k} \cup \{\widetilde{\pi}_k\}$, we first compute $(g_j^-, h_j^-) = \mathrm{EVI}(\mathcal{L}_j^{\pi_j}, \varepsilon_j)$.[16] If $\pi_j = \pi^b$ (i.e., the baseline was executed at episode j), we let $(g_j^-, h_j^-) = (g^{\pi_b}, h^{\pi_b})$ and $\varepsilon_j = 0$. Then

$$\mathbb{E}\left[\sum_{i=1}^t r_i \middle| s_1 = s, \mu_t\right] = \sum_{j=1}^k \sum_{y\in\mathcal{S}} \mathbb{P}\left(s_{t_j} = y \middle| s, \mu_t\right) \cdot \mathbb{E}\left[\sum_{i=1}^{T_j} r_i \mid y, \pi_j\right] \ge \sum_{j=1}^k T_j(g_j^- - \varepsilon_j) - sp(h_j^-) \tag{2.36}$$

where t_j is time at which episode j started, $\mathbb{P}\left(s_{t_j} = y \middle| s_1 = s, \mu_t\right)$ is the probability of reaching state y after t_j steps starting from state s following policy μ_t. The inequality follows from Lem. 3. By lower bounding the LHS of (2.31) by (2.36) and upper bounding the RHS by (2.34), the conservative condition becomes:

$$\sum_{j=1}^{k-1}\left(T_j(g_j^- - \varepsilon_j - g^{\pi_b}) - sp(h_j^-)\right) - sp(h^{\pi_b}) + T_k(g_k^- - \varepsilon_k - (1-\alpha)g^{\pi_b}) \ge 0 \tag{2.37}$$

Note that the algorithm should check this condition at the beginning of episode k in order to understand if the policy $\widetilde{\pi}_k$ is safe or if it should resort to playing policy π_b. In many OFU algorithms, including UCRL2, the length of episode k (i.e., T_k) is not known at the beginning of the episode. As a consequence, condition (2.37) is not directly computable. To overcome this limitation, we consider the dynamic episode condition introduced by (Ouyang et al., 2017). This stopping condition provides an upper-bound on the length of each episode as $T_k \le T_{k-1} + 1$, without affecting the regret bound of UCRL2 (up to constants). This condition can be used to further lower-bound the last term in (2.37) by

$$T_k(g_k^- - \varepsilon_k - (1-\alpha)g^{\pi_b}) \ge (T_{k-1}+1)(g_k^- - \varepsilon_k - (1-\alpha)g^{\pi_b}) \cdot \mathbb{1}_{\left((1-\alpha)g^{\pi_b} \ge g_k^- - \varepsilon_k\right)}. \tag{2.38}$$

Plugging this lower bound into (2.37) gives the final conservative condition

$$\sum_{j=1}^{k-1}\left(T_j(g_j^- - \varepsilon_j - g^{\pi_b}) - sp(h_j^-)\right) - sp(h^{\pi_b}) + (T_{k-1}+1)(g_k^- - \varepsilon_k - (1-\alpha)g^{\pi_b}) \cdot \mathbb{1}_{\left((1-\alpha)g^{\pi_b} \ge g_k^- - \varepsilon_k\right)} \ge 0, \tag{2.39}$$

[16] The subscript j in the operator $\mathcal{L}_j^{\pi_j}$ denotes the fact that it is computed using the samples observed up to t_j. For each episode, we need to compute the estimate only for the new UCRL2 policy. In order to have a tighter estimate of the conservative condition it possible to recompute the gain and bias of the past policies at every episode (or periodically) by using all the available samples (i.e., using $\mathcal{L}_k^\pi$). However, this will break the current regret proof.

tested by CUCRL2 at the beginning of each episode. *Unknown* (g^{π_b}, h^{π_b}). If the gain and bias of the baseline are unknown, we can use EVI on $\mathcal{L}_l^{+,\pi_b}$ (Eq. 2.35 with max instead of min) to compute an optimistic estimate of the cumulative reward of the baseline up to time $t_l + T_{l-1} + 1$. While this account for the RHS of Eq. 2.31, we simply define $(g_l^-, h_l^-) = \text{EVI}(\mathcal{L}_l^{\pi_b}, \epsilon_l)$ for every episode $l \in \Lambda_{k-1}^c$ to compute a lower bound to the cumulative reward obtained by the algorithm by playing the baseline in episode before k. Clearly, this approach is very pessimistic and it may be possible to design better strategies for this case.

The finite-horizon case. We conclude this section with a remark on the finite horizon case. This case is much simpler and resemble the bandit setting. We can directly build a lower bound $\underline{v}_{l,1}$ to the value function $V_1^{\pi_l}$ by using the model estimate and its uncertainty at episode l. This estimate can be computed via extended backward induction –see (Azar et al., 2017a, Alg. 2)– simply subtracting the exploration bonus, see Lem. 20 in App. 2.C.3. The same approach can be used to construct an optimistic and pessimistic estimate of $V_1^{\pi_b}$ when it is unknown. This values can be directly plugged in (2.32) to define a checkable condition for the algorithm.

2.3.3.2 Regret Guarantees

We start providing an upper-bound to the regret of CUCRL2 showing the dependence on UCRL2 and on the baseline π_b. Since the set Λ_k is updated at the end of the episode, we denote by $\Lambda_T = \Lambda_{k_T} \cup \{k_T\} \cdot \mathbb{1}_{(Eq.\ (2.39))}$ the set containing all the episodes where CUCRL2 played an optimistic policy. The set Λ_T^c is its complement.

Lemma 4. *Under Asm. 1 and 9, for any T and any conservative level α, there exists a numerical constant $\beta > 0$ such that the regret of* CUCRL2 *is upper-bounded as*

$$R(\text{CUCRL2}, T) \le \beta \cdot \left(R_{\text{UCRL2}}(T|\Lambda_T) + (g^\star - g^{\pi_b}) \sum_{l \in \Lambda_T^c} T_l + sp(h^{\pi_b}) \sqrt{SAT \ln(T/\delta)} \right),$$

and the conservative condition (2.31) is met at every step $t = 1, \dots, T$ with probability at least $1 - \frac{2\delta}{5}$.[17]

$R_{\text{UCRL2}}(T|\Lambda_T)$ denotes the regret of UCRL2 over an horizon T conditioned on the fact that the UCRL2 policy is executed only at episodes $i \in \Lambda_T$. During the other episodes, the internal statistics of UCRL2 are updated using the samples collected by the baseline policy π_b. This does not pose any major technical challenge and, as shown in App. 2.C.2, the UCRL2 regret can be bounded as follows.

Lemma 5 (Jaksch et al. (2010a))**.** *Let $L_T = \ln\left(\frac{5T}{\delta}\right)$, for any T, there exists a numerical constant $\beta > 0$ such that, with probability at least $1 - \frac{2\delta}{5}$,*

$$R_{\text{UCRL2}}(T|\Lambda_T) \le \beta DS\sqrt{ATL_T} + \beta DS^2 AL_T$$

The second term in Lem. 4 represents the regret incurred by the algorithm when playing the baseline policy π_b. The following lemma shows that the total time spent executing conservative actions is sublinear in time (see Lem. 18 in App. 2.C.2 for details).

Lemma 6. *For any $T > 0$ and any conservative level α, with probability at least $1 - \frac{2\delta}{5}$, the total number of play of conservative actions is bounded by:*

$$\sum_{l \in \Lambda_T^c} T_l \le 2\sqrt{SAT \ln(T)} + \frac{112 SAL_T}{(\alpha g^{\pi_b})^2}(1 + S(D + \Upsilon)^2) + \frac{16\sqrt{TL_T}}{\alpha g^{\pi_b}} \left[(D + \Upsilon)\sqrt{SA} + r_{\max} + \sqrt{SA} sp(h^{\pi_b}) \right]$$

where $L_T = \ln\left(\frac{5SAT}{\delta}\right)$ and $\Upsilon < \infty$ as in Eq. 1.2

Proof. Let τ be the last episode played conservatively: $\tau = \sup\{k > 0 : k \in \Lambda_k^c\}$. This means that at the beginning of episode τ the conservative condition was not verified. By rearranging the terms in Eq. 2.39 and using simple bounds, we can write that:

$$\Delta_t + 4k_T \left(sp(g^{\pi_b}) + \Upsilon + (1 - \alpha) r_{\max} \right) \ge \alpha \sum_{l=1}^{\tau - 1} T_l g^{\pi_b}$$

[17] The probability refers to both events: the regret bound and the conservative condition.

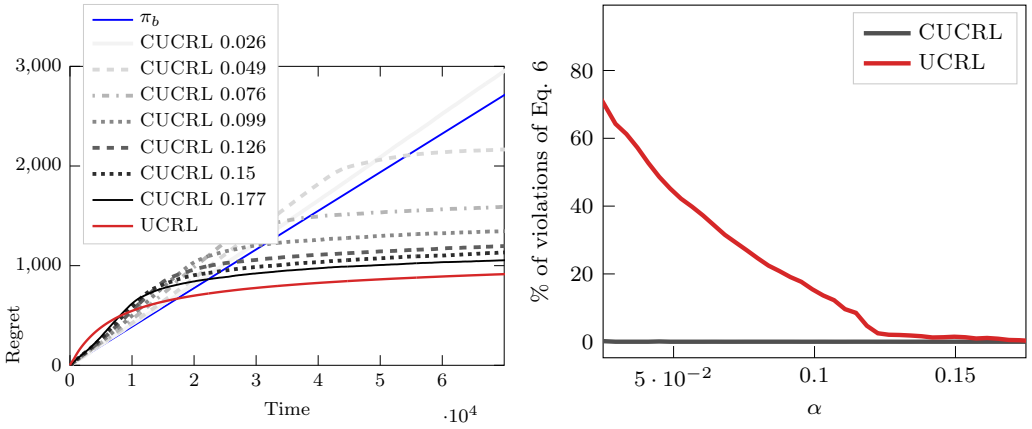


Figure 2.8: Inventory control problem.

where $\Delta_\tau := \sum_{l \in \Lambda_{t-1}} T_l(\widetilde{g}_l - \underline{g}_l)$ and $(\widetilde{g}_l, \underline{g}_l)$ are optimistic and pessimistic gain of policy $\pi_l = \widetilde{\pi}_l$. Note that both satisfies the Bellman equation: $\widetilde{g}_l + \widetilde{h}_l = \mathcal{L}_l^+ \widetilde{h}_l$ (see footnote 14) and $\underline{g}_l + \underline{h}_l = \mathcal{L}_l^{\pi_l} \underline{h}_l$ (see Eq. 2.35). At this point, the important terms in upper-bounding Δ_τ are similar to the one analysed in UCRL2. In particular, we have a term depending on the confidence intervals $\Delta_{ci}^{l,t} := 2\beta_r^l(s_t, a_t) + \beta_p^l(s_t, a_t)\big(sp(\widetilde{h}_l) + sp(\underline{h}^{\pi_l})\big)$ and one depending on the transitions $\Delta_p^{l,t} := p^\star(\cdot|s_t, a_t)^\intercal \left(\widetilde{h}_l + \underline{h}^{\pi_l}\right) - (\widetilde{h}_l(s_{t+1}) - \underline{h}^{\pi_l}(s_{t+1}))$. Let $X = \sum_{l=1}^{\tau-1} T_l$. By using the definition of the confidence intervals, it is easy to show that $\sum_{l \in \Lambda_{\tau-1}} \sum_{t=t_l}^{t_{l+1}-1} \Delta_{ci}^{l,t} \lesssim \sqrt{SATX} + (D + \Upsilon)\sqrt{S^2AX}$. Define the σ-algebra based on past history at t: $\mathcal{F}_t = \sigma(s_1, a_1, r_1, \dots, s_t, a_t, r_t, s_{t+1})$. The sequence $(\Delta_p^{l,t}, \mathcal{F}_t)_{l,t}$ is an MDS. Thus, using Azuma inequality we have that $\sum_{l \in \Lambda_{\tau-1}} \sum_{t=t_l}^{t_{l+1}-1} \Delta_{ci}^{l,t} \lesssim (D + \Upsilon)\sqrt{T}$. Putting everything together we have a quadratic form in X and solving it we can write that $\alpha g^{\pi_b} X \lesssim b_T + S^2 A L_T^\delta (D + \Upsilon)^2 / (\alpha g^{\pi_b})$ where $b_T = \widetilde{O}((D + \Upsilon)\sqrt{SAT})$ (see App. 2.C.2). The result follows noticing that $\sum_{l \in \Lambda_\tau^c} T_l \le \sum_{l=1}^{\tau-1} T_l + T_\tau$. □

Combining the results of Lem. 5 and Lem. 6 into Lem. 4 leads to an overall regret of order $\widetilde{O}(\sqrt{T})$, which matches the regret of UCRL2. This shows that CUCRL2 is able to satisfy the conservative condition without compromising the learning performance. Nonetheless, the bound in Lem. 6 shows how conservative exploration is more challenging in RL compared to the bandit setting. While the dependency on the conservative level α is the same, the number of steps the baseline policy is executed can be as large as $\widetilde{O}(\sqrt{T})$ instead of constant as in CUCB (Wu et al., 2016). Furthermore, Lem. 4 relies on an ergodicity assumption instead of the much milder communicating assumption needed by UCRL2 to satisfy Lem. 5. Asm. 1 translates into the bound through the "worst-case" diameter Υ, which in general is much larger than the diameter D. This dependency is due to the need of computing a lower bound to the reward accumulated by policies π_j in the past (see Lem. 3). In fact, UCRL2 only needs to compute *upper bounds* on the gain and the value function returned by EVI by applying the optimistic Bellman operator $\mathcal{L}_k^+$ has span bounded by the diameter D. This is no longer the case for computing pessimistic estimates of the value of a policy. Whether Asm. 1 and the worst-case diameter Υ are the unavoidable price to pay for conservative exploration in infinite horizon RL remains as an open question.

The finite-horizon case. App. 2.C.3 shows how to modify UCB-VI (Azar et al., 2017a) to satisfy the conservative condition in Eq. 2.32. In this setting, it is possible to show (see Prop. 5 in App. 2.C.3) that the number of conservative episodes is simply logarithmic in $T = KH$. Formally, $|\Lambda_T^c| = O(H^5 S^2 A \ln(T/\delta)/(\alpha r_b(\underline{\Delta}_b + \alpha r_b))$ where $0 < r_b \le r(s,a)$, for all (s,a), and $\underline{\Delta}_b = \min_s\{V_1^\star(s) - V_1^{\pi_b}(s)\}$ is the optimality gap. This problem dependent terms resemble the one in the bandit analysis. The regret of conservative UCB-VI is bounded by $\widetilde{O}(H\sqrt{SAT} + 1/(\alpha r_b(\underline{\Delta}_b + \alpha r_b)))$.

2.3.4 Experiments

In this subsection, we report results in the inventory control problem to illustrate the performance of CUCRL2 compared to unconstrained UCRL2 and how it varies with the conservative level. See App. 2.A.6 for additional

experiments for both average reward and finite horizon. In order to have a better estimate of the budget, we re-evaluate past policies at each episode. We start considering the *stochastic inventory control* problem (Puterman, 1994, Sec. 3.2.1) with capacity $M = 6$ and uniform demand. At the beginning of a month t, the manager has to decide the number of items to order in order to satisfy the random demand, taking into account the cost of ordering and maintainance of the inventory (see App. 2.A.6). Since the optimal policy is a threshold policy, as baseline we consider a (σ, Σ) policy (Puterman, 1994, Sec. 3.2.1) with target stock $\Sigma = 4$ and capacity threshold $\sigma = 4$. Note that $g^\star = 0.603$ and $(g^{\pi_b}, sp(h^{\pi_b})) = (0.565, 0.651)$. We use this domain to perform an ablation study w.r.t. the conservative level α. We have taken $T = 70000$ and the results are averaged over 100 realizations.

Fig. 2.8(*left*) shows that the regret of $\mathrm{CUCRL2}$ grows at the same speed as the one of the baseline policy π_b at the beginning (the conservative phase), because during this phase $\mathrm{CUCRL2}$ is constrained to follow π_b to make sure that constraint (2.31) is satisfied. Clearly, the duration of this conservative phase is proportional to the conservative level α. As soon as $\mathrm{CUCRL2}$ has built margin, it starts interleaving exploratory (optimistic) policies with the baseline. After this phase, $\mathrm{CUCRL2}$ has learn enough about the system and has a sufficient margin to behave as $\mathrm{UCRL2}$. As expected, Fig. 2.8(*left*) confirms that the convergence to the $\mathrm{UCRL2}$ behavior happens more quickly for larger values of α, i.e., when the conservative condition is relaxed and $\mathrm{CUCRL2}$ can explore more freely. On the other hand, $\mathrm{UCRL2}$ converges faster since it is agnostic to the safety constraint and may explore very poor policies in the initial phase. To better understand this condition, Fig. 2.8(*right*) shows the percentage of time the constraint was violated in the first 15000 steps (about 20% of the overall time T). $\mathrm{CUCRL2}$ always satisfies the constraint for all values of α while $\mathrm{UCRL2}$ fails a significant number of times, especially when the conditions is tight (small values of α).

2.3.5 Future Extensions

We presented algorithms for conservative exploration for both finite horizon and average reward problems with $O(\sqrt{T})$ regret. We have shown that the non-episodic nature of average reward problems makes the definition of the conservative condition much harder than in finite horizon problems. In both cases, we used a model-based approach to perform counterfactual reasoning required by the conservative condition. Recent papers have focused on model-free exploration in tabular settings or linear function approximation (Jin et al., 2018; Yang and Wang, 2019; Jin et al., 2019), thus a question is if it is possible for model-free algorithms to be conservative and still achieve $\widetilde{O}(\sqrt{T})$ regret.

2.4 Conclusion

In this chapter, we studied different constraints that may be encountered during the deployment of a Reinforcement Learning system. Those constraints are different by nature. One is about pure performance in terms of reward whereas the other shows how challenging it is to deploy a bandit system as a part of whole recommendation pipeline. In the following of this thesis, we focus on the privacy and security aspect of RL. Two questions that have gained a lot in popularity recently. We will show how those two concepts can be seen as a constraint on the exploration process and which solutions we can use to solve those issues.

Appendix

2.A Appendix for Bandit with Noisy Evaluations

2.A.1 Lower Bound

In this appendix, we prove why no algorithm can compute the top K arms at every time step. More, precisely we prove the following Lemma.

Lemma 7. *Let consider the linear case in Asm. 5 and parameters $K_t = K_{\max}$, $K = 1$, $\alpha_j = 1$ for all $j = 1, \dots, J$ and a noise distributed as $\mathcal{N}(0, \sigma_j)$ with $\sigma_j = \sigma_0$ for all $j = 1, \dots, J$. The learner $\mathfrak{A}$ receives as input the evaluations $f_{i,t,j} = r_{i,t} + \epsilon_{i,t,j}$ and we denote by $\mathfrak{A}(\{f_{i,t,j}\})$ the arm returned by $\mathfrak{A}$. For all arms $i \leq K_t$ the reward $r_{i,t}$ is sampled from a Bernoulli distribution $Ber(1/2)$. At every step, let's define $I_t^\star = \{i \leq K_t \mid r_{i,t} = 1\}$ the set of optimal arms. Then any learner $\mathfrak{A}$ has a fixed non-zero probability to return the wrong ranking at each step, i.e.,*

$$\forall t, \forall \mathfrak{A}, \exists \delta > 0, \mathbb{P}_{Ber, \{\epsilon_{i,t,j}\}} \left[\mathfrak{A}(\{f_{i,t,j}\}) \notin I_t^\star \right] \geq \delta. \tag{2.40}$$

Proof. We reason by contradiction and assume that there exists a deterministic algorithm $\mathfrak{A}$ and a time step t such that

$$\forall \delta > 0, \qquad \mathbb{P}_{\mathsf{Ber}, \{\epsilon_{i,t,j}\}} \left[\mathfrak{A}(\{f_{i,t,j}\}) \notin I_t^\star \right] \leq \delta. \tag{2.41}$$

Now given the rewards $(r_{i,t})_{i \leq K_t}$ the distribution of the evaluations $(f_{i,t,j})_j$ is distributed as $\mathcal{N}\left((r_{i,t})_{j \leq J}, \sigma_0 I_J\right)$ for each i. Now, let's consider the set of sub optimal arms, $I_t^- := \{i \leq K_t \mid r_{i,t} = 0\}$ because the rewards are independent then with probability at least $\frac{1}{4}$ the set I_t^+ and I_t^- are both non-empty. For an index $i_0 \leq K_t$, let $X^{0,1} \in \mathbb{R}^{K_t \times J}$ and $X^{1,0} \in \mathbb{R}^{K_t \times J}$ be two random variables distributed sampled from the same distributions as the $(f_{i,t,j})_{i,j} \mid I_t^\star = \{i_0\}, I_t^- = \{i_{-0}\}$ and $(f_{i,t,j})_{i,j} \mid I_t^- = \{i_0\}, I_t^\star = \{i_{-0}\}$ respectively. We then have that:

$$\mathbb{P}\left(X^{0,1} = X^{1,0}\right) > 0 \tag{2.42}$$

This is a direct consequence of the independence of the noise and rewards. Indeed,

$$\mathbb{P}\left(X^{0,1} = X^{1,0}\right) = \mathbb{P}\left(\forall (i,j) \in \{1, \cdots, K_t\} \times \{1, \dots, J\} \; X_{i,j}^{0,1} = X_{i,j}^{1,0}\right) \tag{2.43}$$

$$= \prod_{i,j} \mathbb{P}(X_{i,j}^{0,1} = X_{i,j}^{1,0}) \tag{2.44}$$

But for a given couple $(i,j) \in \{1, \dots, K_t\} \times \{1, \dots, J\}$, $X_{i,j}^{0,1} \sim \mathcal{N}(\mathbb{1}_{\{i=i_0\}}, \sigma_0^2)$, $X_{i,j}^{1,0} \sim \mathcal{N}(\mathbb{1}_{\{i \neq i_0\}}, \sigma_0^2)$ and are independent. Therefore, the probability of those random variables being equal is:

$$\mathbb{P}\left(X_{i,j}^{0,1} = X_{i,j}^{1,0}\right) = \frac{1}{2\pi\sigma_0^2} \int_{-\infty}^{+\infty} \exp\left(- \frac{\left[\left(x - \mathbb{1}_{\{i=i_0\}}\right)^2 + \left(x - \mathbb{1}_{\{i \neq i_0\}}\right)^2 \right]}{2\sigma_0^2} \right) dx > 0 \tag{2.45}$$

Let's note $\gamma > 0$ such that $\gamma \leq \mathbb{P}\left(X_{i,j}^{0,1} = X_{i,j}^{1,0}\right)$ for all i then we have that: $\mathbb{P}\left(X^{0,1} = X^{1,0}\right) \geq \gamma^{K_t J} > 0$. This implies that there exists a δ_0 such that $\mathbb{P}_{\mathsf{Ber}, \{\epsilon_{i,t,j}\}} \left[\mathfrak{A}(X^{0,1}) \in I_t^\star, X^{0,1} = X^{1,0}, I_t^\star = \{i_0\}, I_t^- = i_{-0} \right] \geq \delta_0 > 0$,

thanks to Eq. (2.41) but then on this event we have that:

$$\delta_0 \leq \mathbb{P}_{\mathsf{Ber},\{\epsilon_{i,t,j}\}}\Big[\mathfrak{A}(X^{0,1}) \in I_t^\star, X^{0,1} = X^{1,0}, I_t^\star = \{i_0\}, I_t^- = i_{-0}\Big] = \mathbb{P}_{\mathsf{Ber},\{\epsilon_{i,t,j}\}}\Big[\mathfrak{A}(X^{1,0}) = i_0, I_t^\star = \{i_0\}, I_t^- = i_{-0}\Big] \tag{2.46}$$

$$\leq \mathbb{P}_{\mathsf{Ber},\{\epsilon_{i,t,j}\}}\Big[\mathfrak{A}(X^{1,0}) = i_0\Big] \tag{2.47}$$

$$\leq \mathbb{P}_{\mathsf{Ber},\{\epsilon_{i,t,j}\}}\Big[\mathfrak{A}(X^{1,0}) \notin I_t^\star\Big] \leq \frac{\delta_0}{2} \tag{2.48}$$

thanks to Eq. 2.41. This is not possible therefore by contradiction we have the result. □

Lem. 7 shows that there exists a instance of the problem studied in this paper where it is not possible to retrieve systematically the K actions maximiziing the reward $(r_{i,t})_{i\leq K_t}$. Therefore defining the regret with respect to the true top K actions.

2.A.2 Noise Correlation Issue

The standard result on GLM are based theorems with the same structure as the following proposition (Filippi et al., 2010):

Proposition 1. *Let $(\mathcal{F}_k)_{k\geq 0}$ be a filtration, $(m_k)_{k\geq 0}$ be an $\mathbb{R}^d$-valued stochastic process adapted to $(\mathcal{F}_k)_k$, $(\eta_k)_k$ be a real-valued martingale difference process adapted to $(\mathcal{F}_k)_k$. Assume that η_k is conditionnally sub-Gaussian in the sense that there exists some $R > 0$ such that for any $\gamma \geq 0$, $k \geq 1$,*

$$\mathbb{E}(\exp(\gamma\eta_k) \mid \mathcal{F}_{k-1}) \leq \exp\left(\frac{\gamma^2 R^2}{2}\right) \text{ a.s} \tag{2.49}$$

Consider the martingale $\xi_t = \sum_{k=1}^t m_{k-1}\eta_k$ and the process $M_t = \sum_{k=1}^t m_{k-1}m_{k-1}^\intercal$. Assume that with probability one the smallest of M_d is lower bounded by some positive constant λ_0 and that $\|m_k\|_2 \leq c_m$ holds a.s. for any $k \geq 0$. The following hold true: Let

$$\kappa = \sqrt{3 + 2\log(1 + 2c_m^2/\lambda_0)}$$

For any $x \in \mathbb{R}^d$, $0 < \delta \leq e^{-1}$, $t \geq \max(d, 2)$, with probability at least $1 - \delta$,

$$|\langle x, \xi_t\rangle \leq \kappa R\sqrt{2\log(t)\log(1/\delta)}\|x\|_{M_t} \tag{2.50}$$

Further, for any $0 < \delta < \min\{1, d/e\}$, $t \geq \max\{\ d, 2\}$, with probability at least $1 - \delta$,

$$\|\xi_t\|_{M_t^{-1}} \leq \kappa R\sqrt{2d\log(t)\log(d/\delta)} \tag{2.51}$$

In our setting, we can not use the type of results presented above because estimators like ridge regression or maximum likelihood estimation rely on the assumption that the noise associated to the data is zero-mean. However, in our setting the learner[18] takes a decision $\mathcal{A}_t$ based on the noisy evaluations $(\phi_{i,t})_i$ and this introduces a statistical dependence that biases the distribution of the noise affecting the evaluations. For simplicity, assume $K = 1$ then the action a_t is a function of the noise in the observations $(\phi_{i,t})$ and, in general, we have that $\mathbb{E}\left(\epsilon_{a_t,t,j}\right) \neq 0$ for all evaluators $j \leq J$ (where $(\epsilon_{i,t,j})$ is the noise in the evaluation $f_{i,t,j}$) although for any fixed non random action $a \leq K_t$, $\mathbb{E}(\epsilon_{a,t,j}) = 0$.

More formally, let consider the GLM setting of Sec. 2.1.3 assuming $K = 1$ and a MLE estimator using all the data $\mathcal{H}_t := \{(f_{a_t,t,j})_j, r_{a_t,t}\ \}$ to estimate the parameter α. Let also assume that the action a_t is chosen as $a_t = \arg\max_i\langle w_t, \phi_{i,t}\rangle$ where for any t, w_t is a vector adapted to the σ-algebra $\sigma(\mathcal{H}_{t-1})$. For each evaluator j we compute the MLE, $\widehat{\alpha}_j$ as the solution to

$$\sum_{l=1}^{t-1}(f_{a_t,t,j} - g(\widehat{\alpha}_j r_{i_t,t}))r_{i_t,t} = 0 \tag{2.52}$$

[18] Notice that the following discussion holds also for any non-learning algorithm, e.g., an oracle algorithm.

with a_t is the arm selected by the algorithm and $r_{a_t,t}$ is the associated reward observed at the end of the round. In order to evaluate the accuracy of this estimator $\widehat{\alpha}_j$, following the proof of Filippi et al. (2010) or Li et al. (2017), we eventually need to control the term

$$\frac{\left|\sum_{l=1}^{t} r_{a_l,l}\epsilon_{a_l,l,j}\right|}{\sqrt{\sum_{l=1}^{t} r_{a_l,l}^2}}, \tag{2.53}$$

where $\epsilon_{a_l,l,j}$ is the noise associated with the evaluation j for arm a_l at round l. One may be tempted to apply Prop. 1 with $m_{t-1} = r_{a_t,t}$ and $\eta_t = \epsilon_{a_t,t,j}$. The issue is that $a_t = \arg\max_i \langle w_{t-1}, g^{-1}(\phi_{i,t})\rangle$ where w_t is a measurable function for of the past (in addition, note that this action selection process is used in most optimistic algorithms for GLM). As a result,

$$\mathbb{E}\left(\epsilon_{a_t,t,j} \mid \mathcal{H}_{t-1}, (r_{i,t})_{i\leq K_t}\right) \neq 0, \tag{2.54}$$

(2.54) can be further simplified when g is the identity function, $J = 1$, $\alpha_j = 0$, $\sigma_j = \sigma$ and $w_t \neq 0$ then (2.54) reduces to

$$\mathbb{E}\left(\epsilon_{a_t,t,j} \mid \mathcal{H}_{t-1}, (r_{i,t})_{i\leq K_t}\right) = \mathbb{E}\left(\epsilon_{i,t}\mathbb{1}_{\{i=\arg\max w_t\epsilon_{i,t}\}} \mid \mathcal{H}_{t-1}\right) \approx \sigma, \tag{2.55}$$

thus showing that $\epsilon_{a_t,t}$ is no longer a zero-mean variable. A way to address this issue would be to take a union bound over all the arms chosen over time $a_1, \ldots, a_t, \ldots, a_T$. Unfortunately, this would lead to take a union bound over K^T elements, which would eventually lead to an additional $\sqrt{T}$ factor in Prop 1 and ultimately a linear regret bound. We further confirm this effect in several empirical validations, where algorithms relying on data generated based on the evaluations introduce a bias in the estimation of the parameters α.

2.A.3 Generalized Linear Model

In this subsection, we present how to defined the oracle and the analysis of the regret in the case of a GLM model.

2.A.3.1 Oracle in the GLM Model (Proof of Lemma 1)

We consider an oracle $\mathfrak{O}$ with access to the link function g, the parameters $(\alpha_j)_{j\leq J}$ of the evaluation function, and the parameters $(\sigma_j)_{j\leq J}$ of the noise distribution. Given the vector $\phi_{i,t} \in \mathbb{R}^J$ obtained by aggregating all the evaluations $(f_{i,t,.})_j$ for each arm $i \leq K_t$, we recall that an oracle defines a set of weights $(w_j)_{j\leq J}$ and predicts the reward of arm i as

$$r_{i,t}^{\mathfrak{O}} = \langle w, g^{-1}(\phi_{i,t})\rangle \tag{2.56}$$

We aim at minimizing the gap between reward of the true top-K arms $i_1^\star, \ldots, i_K^\star$ and the reward of the *estimated top-K arms* $i_1^{\mathfrak{O}}, \ldots, i_K^{\mathfrak{O}}$ according to the estimated rewards $r_{i,t}^{\mathfrak{O}}$

$$(a) = \sum_{l=1}^{K} r_{i_l^\star,t} - \sum_{l=1}^{K} r_{i_l^{\mathfrak{O}},t}. \tag{2.57}$$

By leveraging the definition of estimated top-K arms, we can rewrite the previous expression as

$$\begin{aligned}
(a) &= \sum_{l=1}^{K} r_{i_l^\star,t} - \sum_{l=1}^{K} \widehat{r}_{i_l^\star,t}^{\mathfrak{O}} + \sum_{l=1}^{K} \widehat{r}_{i_l^\star,t}^{\mathfrak{O}} - \sum_{l=1}^{K} \widehat{r}_{i_l^{\mathfrak{O}},t}^{\mathfrak{O}} + \sum_{l=1}^{K} \widehat{r}_{i_l^{\mathfrak{O}},t}^{\mathfrak{O}} - \sum_{l=1}^{K} r_{i_l^{\mathfrak{O}},t} \\
&\leq \sum_{l=1}^{K} r_{i_l^\star,t} - \sum_{l=1}^{K} \widehat{r}_{i_l^\star,t}^{\mathfrak{O}} + \sum_{l=1}^{K} \widehat{r}_{i_l^{\mathfrak{O}},t}^{\mathfrak{O}} - \sum_{l=1}^{K} r_{i_l^{\mathfrak{O}},t} \\
&= \sum_{l=1}^{K} \left(r_{i_l^\star,t} - \widehat{r}_{i_l^\star,t}^{\mathfrak{O}}\right) + \sum_{l=1}^{K} \left(\widehat{r}_{i_l^{\mathfrak{O}},t}^{\mathfrak{O}} - r_{i_l^{\mathfrak{O}},t}\right) \\
&\leq 2 \max_{i_1,\ldots,i_K} \sum_{l=1}^{K} \underbrace{\left|\widehat{r}_{i_l,t}^{\mathfrak{O}} - r_{i_l,t}\right|}_{(b)}.
\end{aligned}$$

Using the GLM model of Asm. 4, we have that for any $i = 1, \dots, K_t$, the estimated reward can be written as

$$\widehat{r}_{i,t}^{\mathfrak{O}} = \sum_{j=1}^{J} w_j g^{-1} \left(g(\alpha_j \cdot r_{i,t}) + \epsilon_{i,t,j} \right). \tag{2.58}$$

We can then measure the deviation between true reward and estimated reward as

$$(b) = \sum_{j=1}^{J} w_j \left[g^{-1} \left(g(\alpha_j \cdot r_{i,t}) + \epsilon_{i,t,j} \right) - g^{-1}(g(\alpha_j r_{i,t})) \right] + \Big(\sum_{j=1}^{J} w_j \alpha_j - 1 \Big) r_{i,t} \tag{2.59}$$

$$\leq c_g^{-1} \sum_{j=1}^{J} |w_j| \cdot |\epsilon_{i,t,j}| + \Big(\sum_{j=1}^{J} w_j \alpha_j - 1 \Big) r_{i,t}, \tag{2.60}$$

where in the first equality we introduce the term $\sum_{j=1}^{K} w_j \alpha_j = \sum_{j=1}^{K} w_j g^{-1}(g(\alpha_j r_{i,t}))$ and in the second step we leverage the fact that g^{-1} is c_g^{-1}-Lipschitz. The last expression above contains two random realizations, i.e., $\epsilon_{i,t,j}$ and $r_{i,t}$, thus preventing from computing a fixed set of weights to minimize the performance gap (a). Furthermore, while we assume the oracle to have prior knowledge about the parameters, we would like to avoid leveraging any knowledge on the true rewards $(r_{i,t})$. In order to remove any dependency on the rewards, we impose a constrain on the weights, such that $\sum_{j=1}^{J} w_j \alpha_j = 1$, thus removing the last term, and thus the rewards, in the previous expression. We can then focus on minimizing a high-probability upper bound of (a)

$$(a) \leq 2 c_g^{-1} \max_{i_1, \dots, i_K} \sum_{l=1}^{K} \sum_{j=1}^{J} |w_j| \cdot |\epsilon_{i,t,j}| \tag{2.61}$$

where we leverage the fact that the noise $\epsilon_{i,t,j}$ are sub-Gaussian then $|\epsilon_{i,t,j}| - \mathbb{E}(|\epsilon_{i,t,j}|)$ is also sub-Gaussian with the same parameter and by Jansen inequality its mean is $\mathbb{E}(|\epsilon_{i,t,j}|) \leq \sqrt{\mathbb{E}(\epsilon_{i,t,j}^2)} = \sigma_j$. Therefore for any time t and any set $U \subset \{1, \cdots, K_t\}$ of size K , we have by Chernoff inequality, for any $x > 0$

$$\mathbb{P}\left(\left| \sum_{u \in U} \sum_{j=1}^{J} |w_j \epsilon_{u,t,j}| \ - \mathbb{E}(|w_j \epsilon_{u,t,j}|) \right| \geq x \right) \leq 2 \exp\left(- \frac{x^2}{2K \sum_{j=1}^{J} (w_j \sigma_j)^2} \right) \tag{2.62}$$

Therefore we have taking a union over all the set of size K,

$$\mathbb{P}\left(\max_{i_1, \cdots, i_K} \sum_{l=1}^{K} \sum_{j=1}^{J} |w_j \epsilon_{i_l,t,j}| - \mathbb{E}(|w_j \epsilon_{i_l,t,j}|) \geq x \right) \leq \sum_{U, |U|=K} \mathbb{P}\left(\sum_{u \in U} \sum_{j=1}^{J} |w_j \epsilon_{u,t,j}| - \mathbb{E}(|w_j \epsilon_{u,t,j}|) \geq x \right) \tag{2.63}$$

$$\leq \sum_{U, |U|=K} 2 \exp\left(- \frac{x^2}{2 \sum_{j=1}^{J} (w_j \sigma_j)^2} \right) \tag{2.64}$$

$$= 2 \binom{K_t}{K} \exp\left(- \frac{x^2}{2K^2 \sum_{j=1}^{J} (w_j \sigma_j)^2} \right) \tag{2.65}$$

Therefore for any $\delta \in (0, 1)$ we have that with probability at least $1 - \delta$:

$$\max_{i_1, \dots, i_K} \sum_{l=1}^{K} \sum_{j=1}^{J} |w_j| \cdot |\epsilon_{i,t,j}| - \mathbb{E}(|w_j| \cdot |\epsilon_{i,t,j}|) \leq K \sqrt{2 \sum_{j=1}^{J} (w_j \sigma_j)^2 \ln\left(\frac{2 \binom{K_t}{K}}{\delta} \right)} \leq K \sqrt{2K \sum_{j=1}^{J} (w_j \sigma_j)^2 \ln\left(\frac{2e K_{\max}}{K \delta} \right)} \tag{2.66}$$

using the standard inequality $\binom{K_t}{K} \leq \left(\frac{e K_t}{K} \right)^K$ and $K_t \ \leq K_{\max}$. Therefore,

$$(a) \leq \frac{2K}{c_g} \sqrt{K \sum_{j=1}^{J} w_j^2 \sigma_j^2 \ln\left(\frac{K_t e}{\delta} \right)} + \frac{K}{c_g} \sum_{j=1}^{J} |w_j| \sigma_j \leq \frac{2K}{c_g} \sqrt{K \sum_{j=1}^{J} w_j^2 \sigma_j^2 \ln\left(\frac{K_{\max} e}{\delta} \right)} + \frac{K}{c_g} \sqrt{J \sum_{j=1}^{J} (w_j \sigma_j)^2}, \tag{2.67}$$

Finally, this leads to the optimization problem in Eq. 2.7. By plugging the optimal solution back into the optimization problem, we also obtain the stated upper bound on the suboptimality gap.

2.A.3.2 Regret of ϵ-greedy Algorithm for GLM Model (Proof of Thm. 1)

We now move on proving the regret upper bound of Thm. 1. The first step is to bound the norm of the noisy evaluations at every step. That is the object of the following lemma.

Lemma 8. *Let $\phi_{i,t} = (f_{i,t,1}, \dots, f_{i,t,j}, \dots, f_{i,t,J}) \in \mathbb{R}^J$ the vector summarizing all the evaluations observed at round t. Then for any $\delta \in (0,1)$ with probability $1-\delta$ for any $t \leq T$, for any set $\mathcal{A}_t$ of K arms chosen in $\{1, \dots, K_{\max}\}$ possibly adaptive to the evaluations, it holds that*

$$\Big\| \sum_{i \in \mathcal{A}_t} \phi_{i,t} \Big\|_2 \leq \Phi := 2K\|\sigma\|_\infty \left(2\sqrt{J} + \sqrt{K \ln\left(\frac{2eK_{max}T}{K\delta} \right)} \right) + 2KJ\|g\|_\infty$$

where $\|\sigma\|_\infty = \max_{j \leq J} \sigma_j$ and $\|g\|_\infty = \max_{j \leq J, x \in [0,C]} |g(\alpha_j x)|$

Proof. For a time $t \leq T$ and set of size K $\mathcal{A}_t \subset \{1, \cdots, K_t\}$ and an evaluator $j \leq J$ the sum of features can be decomposed as,

$$\left(\sum_{i \in \mathcal{A}_t} \phi_{i,t,j} \right)^2 = \left(\sum_{i \in \mathcal{A}_t} g(\alpha_j r_{i,t}) + \sum_{i \in \mathcal{A}_t} \epsilon_{i,t,j} \right)^2 \leq 2 \left(\sum_{i \in \mathcal{A}_t} g(\alpha_j r_{i,t}) \right)^2 + 2 \left(\sum_{i \in \mathcal{A}_t} \epsilon_{i,t,j} \right)^2 \tag{2.68}$$

$$\leq 2K^2 \|g\|_\infty^2 + 2 \left(\sum_{i \in \mathcal{A}_t} \epsilon_{i,t,j} \right)^2 \tag{2.69}$$

Therefore, we just need to bound $\big\| \sum_{i \in \mathcal{A}_t} \epsilon_{i,t} \big\|_2$ to finish the proof. But thanks to the subGaussian assumption on the noise ϵ, we have that:

$$\left\| \sum_{i \in \mathcal{A}_t} \epsilon_{i,t} \right\|_2 \leq 4K \sqrt{J \max_{j \leq J} \sigma_j^2} + 2K \max_{j \leq J} \sigma_j \sqrt{\log\left(\frac{1}{\delta} \right)} \tag{2.70}$$

because the vector $\sum_{i \in \mathcal{A}_t} \epsilon_{i,t}$ is $K \max_j \sigma_j$ sub-Gaussian. The last step is to take a union bound over steps $t \leq T$ and subset of size K. □

The first step to analyze the regret of Alg. 1 is to decompose the regret according to the steps where the algorithm selected a totally random set of arms or when it played greedy. Noting, as in Alg. 1, Z_t the Bernoulli random variable used by the algorithm to distinguish between exploratory and exploitative steps, the regret is

$$R_T = \underbrace{\sum_{t=1}^T Z_t \langle w^+, \sum_{i \in \mathcal{A}_t^+} \phi_{t,i} - \sum_{i \in \mathcal{A}_t} \phi_{i,t} \rangle}_{:= R_{T,1}} + \underbrace{\sum_{t=1}^T (1 - Z_t) \langle w^+, \sum_{i \in \mathcal{A}_t^+} \phi_{i,t} - \sum_{i \in \mathcal{A}_t} \phi_{i,t} \rangle}_{:= R_{T,2}} \tag{2.71}$$

In the following, we bound each term $R_{T,1}$ and $R_{T,2}$.

Bounding $R_{T,1}$. The term we analyze is the regret due to the random steps in the algorithm. Centering the variable Z_t we get that $R_{T,1}$ scales as:

$$\begin{aligned} R_{T,1} &= \sum_{t=1}^T (Z_t - \epsilon) \langle w^+, \sum_{i \in \mathcal{A}_t^+} \phi_{t,i} - \sum_{i \in \mathcal{A}_t} \phi_{i,t} \rangle + \epsilon \sum_{t=1}^T \langle w^+, \sum_{i \in \mathcal{A}_t^+} \phi_{t,i} - \sum_{i \in \mathcal{A}_t} \phi_{i,t} \rangle \\ &\leq \sum_{t=1}^T (Z_t - \epsilon) \langle w^+, \sum_{i \in \mathcal{A}_t^+} \phi_{t,i} - \sum_{i \in \mathcal{A}_t} \phi_{i,t} \rangle + \epsilon T \|w^+\| \left(\Big\| \sum_{i \in \mathcal{A}_t^+} \phi_{i,t} \Big\|_2 + \Big\| \sum_{i \in \mathcal{A}_t} \phi_{i,t} \Big\|_2 \right) \\ &\leq \sum_{t=1}^T (Z_t - \epsilon) \langle w^+, \sum_{i \in \mathcal{A}_t^+} \phi_{t,i} - \sum_{i \in \mathcal{A}_t} \phi_{i,t} \rangle + 2\Phi\epsilon T \|w^+\| \end{aligned} \tag{2.72}$$

thanks to Lemma 8. Now, the first term can be bounded thanks to a standard Azuma-Hoeffding inequality as for every time $t \leq T$, Z_t is independent from the evaluations $(\phi_{i,t})_i$ and the set of arms $\mathcal{A}_t$ hence considering the filtration $(\mathcal{F}_t)_t$ being the history up until time t, we have,

$$\forall t \leq T, \quad \left| \sum_{t=1}^{T} (Z_t - \epsilon) \langle w^+, \sum_{i \in \mathcal{A}_t^+} \phi_{t,i} - \sum_{i \in \mathcal{A}_t} \phi_{i,t} \rangle \right| \leq 4\Phi \|w^+\| \sqrt{2T \ln\left(\frac{4T}{\delta}\right)} \text{ w.p at least } 1 - \delta/2 \tag{2.73}$$

Putting everything toghether we have that for any $\delta \in (0,1)$,

$$R_{T,1} \leq 4\Phi \|w^+\| \left(\sqrt{2T \ln\left(\frac{4T}{\delta}\right)} + \epsilon T \right) \tag{2.74}$$

with probability at least $1 - \delta/2$.

Bounding $R_{T,2}$. Next, we bound the regret coming due to the estimation error in the exploitative steps, that is to say when $Z_t = 0$. We begin by using the greedy behavior of the algorithm to enough to study the error $\|w_t - w^+\|$ to bound $R_{T,2}$,

$$R_{T,2} \leq \sum_{t=1}^{T} (1 - Z_t) \left[\langle w^+ - w_t, \sum_{i \in \mathcal{A}_t^+} \phi_{i,t} \rangle + \langle w_t, \sum_{i \in \mathcal{A}_t^+} \phi_{i,t} - \sum_{i \in \mathcal{A}_t} \phi_{i,t} \rangle + \langle w_t - w^+, \sum_{i \in \mathcal{A}_t} \phi_{i,t} \rangle \right] \tag{2.75}$$

But because $\mathcal{A}_t$ is the maximizer of the estimated reward with respect to w_t we have that

$$(1 - Z_t) \left\langle w_t, \sum_{i \in \mathcal{A}_t^+} \phi_{i,t} - \sum_{i \in \mathcal{A}_t} \phi_{i,t} \right\rangle \leq 0. \tag{2.76}$$

Furthermore, using Lemma 8 we have that with probability at least $1 - \delta/4$

$$\max \left\{ \left\langle w^+ - w_t, \sum_{i \in \mathcal{A}_t^+} \phi_{i,t} \right\rangle, \left\langle w_t - w^+, \sum_{i \in \mathcal{A}_t} \phi_{i,t} \right\rangle \right\} \leq \Phi \|w^+ - w_t\|_2. \tag{2.77}$$

As a result, the remaining step to obtain a bound on the regret is to build a concentration inequality for the weights w_t w.r.t. w^+ and properly tune the exploration factor ϵ.

We start by studying the concentration of the MLE estimator $\widehat{\alpha}_t \in \mathbb{R}^J$ to the true parameter $\alpha \in \mathbb{R}^J$. Thanks to the random choice of $\mathcal{A}_t$ in the explorative steps, all the data in $\mathcal{H}_t$ are i.i.d. and we can directly apply the standard concentration inequality for MLE in GLM (Li et al., 2017).

Lemma 9. *For any $\delta \in (0,1)$ with probability at least $1 - \delta/8$ for all $j \leq J$ and $t \leq T$:*

$$|\widehat{\alpha}_{t,j} - \alpha_j| \leq \beta_{t,j} := \frac{\sqrt{\lambda}\|\alpha\| + \sigma_j \sqrt{\frac{1}{2} \log\left(1 + \frac{2T}{\lambda}\right) + \log\left(\frac{8JT}{\delta}\right)}}{c_g \sqrt{\sum_{l, Z_l = 1} \sum_{i \in \mathcal{A}_t} r_{l,i}}}. \tag{2.78}$$

In other words,

$$\|\widehat{\alpha}_t - \alpha_r\|_\infty \leq \beta_t := \max_j \beta_{t,j} = \frac{\sqrt{\lambda}\|\alpha\| + \|\sigma\|_\infty \sqrt{\frac{1}{2} \log\left(1 + \frac{2T}{\lambda}\right) + \log\left(\frac{8JT}{\delta}\right)}}{c_g \sqrt{\sum_{l, Z_l = 1} \sum_{i \in \mathcal{A}_t} r_{l,i}}} \tag{2.79}$$

We leverage Lem. 9 to concentrate $\|w^\star - w_t\|$. But first, recall that w_t is given by:

$$w_t = \frac{\sigma^{-1} \cdot \widehat{\alpha}_t}{\|\sigma^{-1} \cdot \widehat{\alpha}_t\|_2^2} \text{ and } w^+ = \frac{\sigma^{-1} \cdot \alpha}{\|\sigma^{-1} \cdot \alpha\|_2^2}$$

where $\Sigma = \text{diag}(\sigma_1, \cdots, \sigma_J)$. Therefore, the error between w^+ and w_t can be written as:

$$\|w_t - w^+\|_2 = \left\| \frac{\sigma^{-2} \cdot (\widehat{\alpha}_t - \alpha)}{\|\sigma^{-1} \cdot \widehat{\alpha}_t\|^2} + \sigma^{-2} \cdot \alpha \left(\frac{1}{\|\sigma^{-1} \cdot \widehat{\alpha}_t\|^2} - \frac{1}{\|\sigma^{-1} \cdot \alpha\|^2} \right) \right\| \tag{2.80}$$

$$\leq \left\| \sigma^{-2} \cdot \alpha \left(\frac{1}{\|\sigma^{-1} \cdot \widehat{\alpha}_t\|^2} - \frac{1}{\|\sigma^{-1} \cdot \alpha\|^2} \right) \right\|_2 + \left\| \frac{\sigma^{-2} \cdot (\widehat{\alpha}_t - \alpha)}{\|\sigma^{-1} \cdot \widehat{\alpha}_t\|^2} \right\|_2 \tag{2.81}$$

Proposition 2. *For any time $t \leq T$, assuming that $\|\widehat{\alpha}_t - \alpha\|_\infty \leq \beta_t$ and $\|\alpha\|_\infty \geq 8\beta_t$ then:*

$$\frac{\|\sigma^{-2} \cdot (\widehat{\alpha}_t - \alpha)\|_2}{\langle \widehat{\alpha}_t, \sigma^{-2} \cdot \widehat{\alpha}_t \rangle} \leq \frac{4\beta_t \sqrt{\sum_{j=1}^J \frac{1}{\sigma_j^4}}}{\sum_{j=1}^J \frac{\alpha_j^2 \mathbb{1}_{\{|\alpha_j| \geq 8\beta_t\}}}{\sigma_j^2}} = 4\beta_t \left(\frac{\|\sigma^{-1}\|_4}{\|\sigma^{-1} \cdot \alpha \mathbb{1}_{\{|\alpha| \geq 8\beta_t\}}\|_2} \right)^2 \tag{2.82}$$

Proof. Using the definition of $\sigma^{-2}\cdot$ we have,

$$\frac{\|\sigma^{-2} \cdot (\widehat{\alpha}_t - \alpha)\|_2}{\langle \widehat{\alpha}_t, \sigma^{-2} \cdot \widehat{\alpha}_t \rangle} = \frac{\sqrt{\sum_{j=1}^J \frac{(\widehat{\alpha}_{t,j} - \alpha_j)^2}{\sigma_j^4}}}{\sum_{r=1}^J \frac{\widehat{\alpha}_{t,j}^2}{\sigma_j^2}} \tag{2.83}$$

However, we have that for any $j \leq J$:

$$\begin{aligned} \widehat{\alpha}_{t,j}^2 = \alpha_j^2 + (\widehat{\alpha}_{t,j} - \alpha_j)^2 + 2(\widehat{\alpha}_{t,j} - \alpha_j)\alpha_j &\geq \alpha_j^2 + 2(\widehat{\alpha}_{t,j} - \alpha_j)\alpha_j \\ &\geq \alpha_j^2 - 2\beta_t |\alpha_j| \\ &\geq \frac{\alpha_j^2}{2} \text{ when } |\alpha_j| \geq 8\beta_t \end{aligned}$$

using that $\|\widehat{\alpha}_t - \alpha\|_\infty \leq \beta_t$. In addition, using again that $\|\widehat{\alpha}_t - \alpha\|_\infty \leq \beta_t$,

$$\sum_{j=1}^J \frac{(\widehat{\alpha}_{t,j} - \alpha_j)^2}{\sigma_j^4} \leq \sum_{j=1}^J \frac{\beta_t^2}{\sigma_j^4} \tag{2.84}$$

□

Using the same reasonning as in Prop. 2, we can show the following which bound the second term in (2.80).

Proposition 3. *For any time $t \leq T$, assuming that $\|\widehat{\alpha}_t - \alpha\|_\infty \leq \beta_t$ and $\|\alpha\|_\infty \geq 8\beta_t$ then:*

$$\left\| \left(\frac{\|\sigma^{-1} \cdot \alpha\|^2 - \|\sigma^{-1} \cdot \widehat{\alpha}_t\|^2}{\|\sigma^{-1} \cdot \widehat{\alpha}_t\|^2} \right) \frac{\sigma^{-2} \cdot \alpha}{\|\sigma^{-1} \cdot \alpha\|^2} \right\| \leq \frac{4\beta_t \left(\|\sigma^{-2} \cdot \alpha\|_2 + \beta_t \|\sigma^{-2}\| \right) \|\sigma^{-2} \cdot \alpha\|}{\|\sigma^{-1} \cdot \alpha \mathbb{1}_{\{|\alpha| \geq 8\beta_t\}}\|^2 \|\sigma^{-1} \cdot \alpha\|^2} \tag{2.85}$$

Therefore, using Prop. 2 and Prop. 3 on the event that the confidence intervals in Lem. 9 holds then with probability at least $1 - \delta/4$:

$$\|w_t - w^+\|_2 \leq \frac{4\beta_t \left(\|\sigma^{-2} \cdot \alpha\|_2 + \beta_t \|\sigma^{-2}\| \right) \|\sigma^{-2} \cdot \alpha\|}{\|\sigma^{-1} \cdot \alpha \mathbb{1}_{\{|\alpha| \geq 8\beta_t\}}\|^2 \|\sigma^{-1} \cdot \alpha\|^2} + 4\beta_t \left(\frac{\|\sigma^{-1}\|_4}{\|\sigma^{-1} \cdot \alpha \mathbb{1}_{\{|\alpha| \geq 8\beta_t\}}\|_2} \right)^2 \tag{2.86}$$

In order to complete the proof of the regret upper bound, we need to control how fast the confidence intervals width β_t decreases with t.

Lemma 10. *For any $\delta \in (0,1)$ and $\lambda > 0$ we have with probability at least $1 - \delta/4$,*

$$\forall t \in \left\{ \frac{2C^4 \ln\left(\frac{8T}{\delta}\right)}{\epsilon^2 K^2 \eta_\nu^4}, \ldots, T \right\}, \ \beta_t \leq \frac{\|\sigma\|_\infty \sqrt{\ln\left(1 + \frac{2T}{\lambda}\right) + \ln\left(\frac{8TJ}{\delta}\right)} + \sqrt{\lambda}\|\alpha\|}{c_g \sqrt{\epsilon t K \eta_\nu^2 - C^2 \sqrt{2t \ln\left(\frac{8T}{\delta}\right)}}} \tag{2.87}$$

where for all $j \leq J$, $\eta_\nu^2 = \mathbb{E}_{r \sim \nu}(r^2)$

Proof. First, thanks to a standard Hoeffding inequality we have that with probability at least $1-\delta/4$,

$$\forall t \leq T, \quad \left| \sum_{l \leq t, Z_l = 1} \sum_{i \in A_t} r_{l,i}^2 - \mathbb{E}\left(\sum_{l \leq t, Z_l = 1} \sum_{i \in A_t} r_{l,i}^2 \right) \right| \leq C^2 \sqrt{2t \ln\left(\frac{8T}{\delta}\right)}$$

And $\mathbb{E}\left(\sum_{l \leq t, Z_l = 1} \sum_{i \in A_t} r_{l,i}^2\right) = \epsilon t K \eta_\nu^2$ where $\eta_\nu^2 := \mathbb{E}_{r \sim \nu}(r^2)$. Therefore, using the definition of β_t

$$\beta_t \leq \frac{\|\sigma\|_\infty \sqrt{\ln\left(1+\frac{2T}{\lambda}\right) + \ln\left(\frac{8TJ}{\delta}\right)} + \sqrt{\lambda}\|\alpha\|}{c_g \sqrt{\epsilon t K \eta_\nu^2 - C^2 \sqrt{2t \ln\left(\frac{8T}{\delta}\right)}}} \tag{2.88}$$

□

As a consequence, for $t \geq \tau_0 := \frac{8\ln\left(\frac{8T}{\delta}\right) C^4}{\left(\epsilon K \eta_\nu^2\right)^2}$, $\beta_t \leq \mathcal{O}\left(\frac{1}{\sqrt{\epsilon t}}\right)$. Therefore, using Lem. 10 with probability at least $1-\delta/4$:

$$\begin{aligned} \sum_{t=\tau_0+1}^{T} \beta_t &\leq \sum_{t=\tau_0+1}^{T} \frac{\sqrt{2}\|\sigma\|_\infty \sqrt{\ln\left(1+\frac{2T}{\lambda}\right) + \ln\left(\frac{8TJ}{\delta}\right)} + \sqrt{\lambda}\|\alpha\|}{c_g \sqrt{\epsilon t K \eta_\nu^2}} \\ &\leq \frac{3\sqrt{2T}\|\sigma\|_\infty}{c_g \sqrt{\epsilon K \eta_\nu^2}} \left(\sqrt{\ln\left(1+\frac{2T}{\lambda}\right) + \ln\left(\frac{8TJ}{\delta}\right)} + \sqrt{\lambda}\|\alpha\| \right) \end{aligned} \tag{2.89}$$

We can finally bound $R_{T,1}$ with probability at least $1-\delta/2$ by:

$$\begin{aligned} R_{T,1} \leq \tau + \frac{24\Phi\sqrt{2T}\|\sigma\|_\infty}{c_g \sqrt{\epsilon K \eta_\nu^2}} \left(\sqrt{\ln\left(1+\frac{2T}{\lambda}\right) + \ln\left(\frac{8TJ}{\delta}\right)} + \sqrt{\lambda}\|\alpha\| \right) \left(\frac{\left(\|\sigma^{-2}\cdot\alpha\|_2 + \|\sigma^{-2}\|\right)\|\sigma^{-2}\cdot\alpha\|}{\|\sigma^{-1}\cdot\alpha \mathbb{1}_{\{|\alpha| \geq 8\beta_t\}}\|^2 \|\sigma^{-1}\cdot\alpha\|^2} \right. \\ \left. + \left(\frac{\|\sigma^{-1}\|_4}{\|\sigma^{-1}\cdot\alpha \mathbb{1}_{\{|\alpha| \geq 8\beta_t\}}\|_2} \right)^2 \right) \end{aligned} \tag{2.90}$$

where $\tau := \max\left\{ \frac{128\|\sigma\|_\infty \left(\ln\left(1+\frac{2T}{\lambda}\right) + \ln\left(\frac{8TJ}{\delta}\right)\right) + \lambda\|\alpha\|^2}{\min_j |\alpha_r|^2 \epsilon K \eta_\nu^2}, \frac{8\ln\left(\frac{8T}{\delta}\right) C^4}{\left(\epsilon K \eta_\nu^2\right)^2} \right\}$

Finally, the regret is bounded with probability at least $1-\delta$ by:

$$\begin{aligned} R_T \leq \frac{4\Phi}{\|\sigma^{-1}\cdot\alpha\|} \left(\sqrt{2T \ln\left(\frac{4T}{\delta}\right)} + \epsilon T \right) + \max\left\{ \frac{128\|\sigma\|_\infty \left(\ln\left(1+\frac{2T}{\lambda}\right) + \ln\left(\frac{8TJ}{\delta}\right)\right) + \lambda\|\alpha\|^2}{\min_{j,\alpha_j \neq 0} |\alpha_j|^2 \epsilon K \eta_\nu^2}, \frac{8\ln\left(\frac{8T}{\delta}\right) C^4}{\left(\epsilon K \eta_\nu^2\right)^2} \right\} \\ + \frac{24\Phi\sqrt{2T}\|\sigma\|_\infty}{c_g \sqrt{\epsilon K \eta_\nu^2}} \left(\sqrt{\ln\left(1+\frac{2T}{\lambda}\right) + \ln\left(\frac{8TJ}{\delta}\right)} + \sqrt{\lambda}\|\alpha\| \right) \left(\frac{\left(\|\sigma^{-2}\cdot\alpha\|_2 + \|\sigma^{-2}\|\right)\|\sigma^{-2}\cdot\alpha\|}{\|\sigma^{-1}\cdot\alpha\|^2 \|\sigma^{-1}\cdot\alpha\|^2} + \left(\frac{\|\sigma^{-1}\|_4}{\|\sigma^{-1}\cdot\alpha\|_2} \right)^2 \right) \end{aligned} \tag{2.91}$$

Remark. Compared to the bound reported in Thm. 1, we corrected an error in the analysis which removed the dependency on the norm $\|g\|_\infty$ and replaceed it with the upper bound on the distribution of the reward C. In addition, here we report a bound depends on $\min_{j \leq J, \alpha_j \neq 0} |\alpha_j|^2$ however because of the condition $\mathbb{1}_{\{|\alpha_j| \geq 8\beta_t\}}$ it is possible to present a regret upper bound that scales inversly with $\max_j |\alpha_j|$ but at the cost of a much bigger problem-dependent quantity S. Fortunately, the difference between the bound report in Thm 1 and the one in (2.91) has no impact on the discussion around the depency on J. Simplifying the bound in (2.91), we get that the regret is bounded with high probability by,

$$R_T \leq \widetilde{\mathcal{O}}\left(T^{2/3} \left(\frac{(1+\sqrt{J^{-1}}\|\alpha\|)\Phi S \|\sigma\|_\infty}{c_g \sqrt{K}\eta_\nu} + \max\left\{ \frac{\|\sigma\|_\infty}{\min_{j,\alpha_j \neq 0} |\alpha_j|^2 K \eta_\nu^2}, \frac{C^4}{\left(K\eta_\nu^2\right)^2} \right\} \right) \right) \tag{2.92}$$

when choosing $\epsilon = \frac{1}{T^{1/3}}$, $\lambda = J^{-1}$ and setting $S := \frac{\left(\|\sigma^{-2}\cdot\alpha\|_2 + \|\sigma^{-2}\|\right)\|\sigma^{-2}\cdot\alpha\|}{\|\sigma^{-1}\cdot\alpha\|^2 \|\sigma^{-1}\cdot\alpha\|^2} + \left(\frac{\|\sigma^{-1}\|_4}{\|\sigma^{-1}\cdot\alpha\|_2} \right)^2$.

2.A.4 Linear Model

In this subsection, we show how the definition of the oracle changes with the linear setting and the proof of Thm. 2, that is to the regret of Alg. 2

2.A.4.1 Oracle in the Linear Model (Proof of Lem. 2)

Just as in Section 2.A.3.1, we consider an oracle that has access to the parameters $(\alpha_j)_{j\leq J}$ but also knows the noise $(\sigma_j)_{j\leq J}$ for each evaluation functions. Given a vector of evalutations $\phi_{i,t}$ for some $t \leq T$ and $i \leq K_t$, an orcale is defined through a set of weights $(w_j)_{j\leq J}$ and predicts the reward, $r^{\mathfrak{O}}_{i,t} = \langle w, \phi_{i,t}\rangle$. Following the reasonning in Section 2.A.3.1, the difference between the true top-K arms, $i^+_1, \cdots, i^+_K$, and the top K according to the oracle, $i^{\mathfrak{O}}_1, \cdots, i^{\mathfrak{O}}_K$ is bounbed by

$$\sum_{l=1}^{K} r_{i^+_l,t} - r_{i^{\mathfrak{O}}_l,t} \leq 2 \max_{i_1,\cdots,i_K} \sum_{l=1}^{K} \left|\widehat{r}^{\mathfrak{O}}_{i_l,t} - r_{i_l,t}\right| \tag{2.93}$$

Again following the same reasonning as in App. 2.A.3.1, the difference between the estimated reward and the actual reward for any arm $i \leq K_t$ is bounded by

$$\left|r_{i,t} - \widehat{r}^{\mathfrak{O}}_{i,t}\right| \leq \left|r_{i,t}\left(\sum_{j=1}^{J} w_j\alpha_j - 1\right)\right| + \left|\sum_{j=1}^{J} w_j\epsilon_{i,t,j}\right| \leq C\left|\left(\sum_{j=1}^{J} w_j\alpha_j - 1\right)\right| + \left|\sum_{j=1}^{J} w_j\epsilon_{i,t,j}\right| \tag{2.94}$$

The main difference compared to the proof of Lem. 1 is that thanks to the linear strucutre contorlling the noise can be done direclty without usingany Lipschitz property. Therefore, in order to remove the dependency on the reward, we focus on weights w such that $\langle w, \alpha\rangle = 1$. Hence using Chernoff inequality the error is bounded for any $\delta \in (0,1)$ by

$$\left|r_{i,t} - r^{\mathfrak{O}}_{i,t}\right| \leq 2K\sqrt{K\sum_{j=1}^{J} w_j^2\sigma_j^2 \ln\left(\frac{K_{\mathsf{max}}e}{\delta}\right)} \tag{2.95}$$

with probability at least $1-\delta$. This conlcudes the proof of Lem. 1

2.A.4.2 Regret of ESAG (Proof of Thm. 2)

The first step to analyze the regret of Alg. 2 is to compute an upper bound on the deviation of the estimator $\widehat{\alpha}_t$.

Lemma 11. *For any $\delta \in (0,1)$ and $t \leq T$, the error between $\widehat{\alpha}_t$ and $\mathbb{E}_{r\sim\nu}(r)\alpha$ is bounded with probability at least $1-\delta/4$ by,*

$$\|\widehat{\alpha}_t - \mathbb{E}_{r\sim\nu}(r)\alpha\| \leq \beta^L_t := \frac{C\sqrt{2\ln\left(\frac{16JT}{\delta}\right)}}{\sqrt{\sum_{l=1}^{t-1} K_l}} + \frac{4\|\sigma\|_\infty\sqrt{J}}{\sqrt{\sum_{l=1}^{t-1} K_l}} + \frac{2\|\sigma\|_\infty\sqrt{\ln\left(\frac{8TJ}{\delta}\right)}}{\sqrt{\sum_{l=1}^{t-1} K_l}} \tag{2.96}$$

Proof. For any $t \leq T$, $\widehat{\alpha}_t = \frac{\sum_{l=1}^{t-1}\sum_{i=1}^{K_l}\phi_{i,l}}{\sum_{l=1}^{t-1} K_l}$ but using the strucuture of the evaluations $\phi_{i,t}$ for every $j \leq J$,

$$\widehat{\alpha}_{t,j} = \frac{1}{\sum_{l=1}^{t-1} K_l}\left(\sum_{l=1}^{t-1}\sum_{i=1}^{K_l}\alpha_j r_{i,l} + \sum_{l=1}^{t-1}\sum_{i=1}^{K_l}\epsilon_{i,t,j}\right) \tag{2.97}$$

But the first term converges toward $\alpha_j\mathbb{E}_{r\sim\nu}(r)$ because at every time step the reward are independtly sampled from the distribution ν (see Assumption 2). More precisely, with probability at least $1-\delta/8$, for any $t \leq T$ and $j \leq J$

$$\left|\frac{1}{\sum_{l=1}^{t-1} K_l}\sum_{l=1}^{t-1}\sum_{i=1}^{K_l} r_{i,l} - \mathbb{E}_{r\sim\nu}(r)\right| \leq \frac{C\sqrt{2\ln\left(\frac{16JT}{\delta}\right)}}{\sqrt{\sum_{l=1}^{t-1} K_l}} \tag{2.98}$$

In addition, the second term in Eq. (2.97) can be bounded with probability at least $1-\delta/8$ by

$$\begin{aligned}\left\| \frac{1}{\sum_{l=1}^{t-1} K_l} \sum_{l=1}^{t-1} \sum_{i=1}^{K_l} \epsilon_{i,l} \right\| &\leq \frac{4\sqrt{J \max_j \sigma_j^2 \sum_{l=1}^{t-1} K_l} + 2\sqrt{\max_j \sigma_j^2 \sum_{l=1}^{t-1} K_l \ \ln\left(\frac{8TJ}{\delta}\right)}}{\sum_{l=1}^{t-1} K_l} \\ &= \frac{4\|\sigma\|_\infty \sqrt{J}}{\sqrt{\sum_{l=1}^{t-1} K_l}} + \frac{2\|\sigma\|_\infty \sqrt{\ln\left(\frac{8TJ}{\delta}\right)}}{\sqrt{\sum_{l=1}^{t-1} K_l}}\end{aligned} \tag{2.99}$$

Finally, to finish the proof, we simply need to note that

$$\|\widehat{\alpha}_t - \mathbb{E}_{r\sim\nu}(r)\alpha\| \leq \left\| \frac{\alpha}{\sum_{l=1}^{t-1} K_l} \sum_{l=1}^{t-1} \sum_{i=1}^{K_l} r_{i,l} - \alpha \mathbb{E}_{r\sim\nu}(r) \right\| + \left\| \frac{1}{\sum_{l=1}^{t-1} K_l} \sum_{l=1}^{t-1} \sum_{i=1}^{K_l} \epsilon_{i,l} \right\| \tag{2.100}$$

$$\leq \|\alpha\| \left| \frac{1}{\sum_{l=1}^{t-1} K_l} \sum_{l=1}^{t-1} \sum_{i=1}^{K_l} r_{i,l} - \mathbb{E}_{r\sim\nu}(r) \right| + \frac{4\|\sigma\|_\infty \sqrt{J}}{\sqrt{\sum_{l=1}^{t-1} K_l}} + \frac{2\|\sigma\|_\infty \sqrt{\ln\left(\frac{8TJ}{\delta}\right)}}{\sqrt{\sum_{l=1}^{t-1} K_l}} \tag{2.101}$$

□

Once ESAG has computed the weigths $\widehat{\alpha}_t$ it then computes a set of weights thanks to this estimator and based on the shape of the optimal weigths of the oracle. That is to say the weights w used by the algorithm are:

$$w_t = \frac{\sigma^{-2} \cdot \widehat{\alpha}_t}{\|\sigma^{-1} \cdot \widehat{\alpha}_t\|_2^2} \tag{2.102}$$

The optimal weights for the parameter $\mathbb{E}_{r\sim\nu}(r)\alpha$ is $\frac{w^+}{\mathbb{E}_{r\sim\nu}(r)}$. Therefore, in the following we bound the error between the weights w_t and the rescaled optimal weights $\frac{w^+}{\mathbb{E}_{r\sim\nu}(r)}$. Following the same reasonning as in Prop. 2 and Prop. 3, we can show the following proposition

Proposition 4. *For any $t \leq T$ let's assume $\mathbb{E}_{r\sim\nu}(r)\|\alpha\|_\infty \geq 8\beta_t^L$ with β_t^L defined in Eq. (2.97). Then for any $\delta \in (0,1)$ with probability at least $1-\delta/4$*

$$\left\| w_t - \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)} \right\|_2 \leq \frac{4\beta_t^L \left(\mathbb{E}_{r\sim\nu}(r)\|\sigma^{-2}\cdot\alpha\|_2 + \beta_t^L \|\sigma^{-2}\|\right) \|\sigma^{-2}\cdot\alpha\|}{\mathbb{E}_{r\sim\nu}(r)^3 \|\sigma^{-1}\cdot\alpha \mathbb{1}_{\{\mathbb{E}_{r\sim\nu}(r)|\alpha|\geq 8\beta_t^L\}}\|^2 \|\sigma^{-1}\cdot\alpha\|^2} + \frac{4\beta_t^L}{\mathbb{E}_{r\sim\nu}(r)^2} \left(\frac{\|\sigma^{-1}\|_4}{\|\sigma^{-1}\cdot\alpha \mathbb{1}_{\{\mathbb{E}_{r\sim\nu}(r)|\alpha|\geq 8\beta_t^L\}}\|_2} \right)^2 \tag{2.103}$$

Thanks to Lemma 11 and Prop. 4, we can now analyze the regret of ESAG. Recall the regret is defined as follows,

$$R_T = \sum_{t=1}^{T} \sum_{i\in\mathcal{A}_t^+} \langle w^+, \phi_{i,t}\rangle - \sum_{i\in\mathcal{A}_t} \langle w^+, \phi_{i,t}\rangle \tag{2.104}$$

where $\mathcal{A}_t^+ := \arg\max_i^K \langle w^+, \phi_{i,t}\rangle$ and $\mathcal{A}_t$ is the set of jobs selected by the learner at time t.

However, the set $\mathcal{A}_t^+$ is only defined as the arg max of some values, it is invariant by multiplicating the weights w^+ by some constant independent of the arms. In particular, we have that for any time $t \leq T$

$$\mathcal{A}_t^+ = \arg\max_i^K \langle w^+, \phi_{i,t}\rangle = \arg\max_i^K \left\langle \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)}, \phi_{i,t} \right\rangle \tag{2.105}$$

Hence the regret can rewritten as,

$$\begin{aligned} R_T &= \sum_{t=1}^{T} \mathbb{E}_{r\sim\nu}(r) \left(\sum_{i\in\mathcal{A}_t^+} \left\langle \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)}, \phi_{i,t} \right\rangle - \sum_{i\in\mathcal{A}_t} \left\langle \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)}, \phi_{i,t} \right\rangle \right) \\ &= \sum_{t=1}^{T} \mathbb{E}_{r\sim\nu}(r) \left(\sum_{i\in\mathcal{A}_t^+} \left\langle \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)} - w_t, \phi_{i,t} \right\rangle + \sum_{i\in\mathcal{A}_t^+} \langle w_t, \phi_{i,t}\rangle - \sum_{i\in\mathcal{A}_t} \langle w_t, \phi_{i,t}\rangle + \sum_{i\in\mathcal{A}_t} \left\langle w_t - \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)}, \phi_{i,t} \right\rangle \right) \end{aligned}$$

But for any $t \leq T$, because of the greedy arm selection process of ESAG:

$$\sum_{i \in \mathcal{A}_t^+} \langle w_t, \phi_{i,t} \rangle - \sum_{i \in \mathcal{A}_t} \langle w_t, \phi_{i,t} \rangle \leq 0 \tag{2.106}$$

But using Lem. 8 with probability at least $1 - \delta/4$

$$\max \left\{ \left\langle \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)} - w_t, \sum_{i \in \mathcal{A}_t^+} \phi_{i,t} \right\rangle, \left\langle w_t - \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)}, \sum_{i \in \mathcal{A}_t} \phi_{i,t} \right\rangle \right\} \leq \Phi \| \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)} - w_t \|_2. \tag{2.107}$$

Thanks to Lem.11 and that for all $t \leq T$, $K_t \geq K$ therefore for

$$t \geq t_0 := 1 + 4K^{-1} \left(C^2 \ln \left(\frac{16JT}{\delta} \right) + 2\|\sigma\|_\infty \left(4J + \ln \left(\frac{8JT}{\delta} \right) \right) \right) \max \left\{ \frac{64}{\mathbb{E}_{r\sim\nu}(r)^2 \min_{j,\alpha_j \neq 0} \alpha_j^2}, 1 \right\}$$

,

$$\left\| \frac{w^+}{\mathbb{E}_{r\sim\nu}(r)} - w_t \right\| \leq \frac{4\beta_t^L \left(\mathbb{E}_{r\sim\nu}(r) \|\sigma^{-2} \cdot \alpha\|_2 + \|\sigma^{-2}\| \right) \|\sigma^{-2} \cdot \alpha\|}{\mathbb{E}_{r\sim\nu}(r)^3 \|\sigma^{-1} \cdot \alpha\|^2 \|\sigma^{-1} \cdot \alpha\|^2} + \frac{4\beta_t^L}{\mathbb{E}_{r\sim\nu}(r)^2} \left(\frac{\|\sigma^{-1}\|_4}{\|\sigma^{-1} \cdot \alpha\|_2} \right)^2 \tag{2.108}$$

But using the definition of β_t^L:

$$\begin{aligned} \sum_{t=t_0+1}^{T} \beta_t^L &\leq \left(C\sqrt{2\ln\left(\frac{16JT}{\delta}\right)} + 4\|\sigma\|_\infty \sqrt{J} + 2\|\sigma\|_\infty \sqrt{\ln\left(\frac{8TJ}{\delta}\right)} \right) \sum_{t=t_0+1}^{T} \frac{1}{\sqrt{\sum_{l=1}^{t-1} K_l}} \\ &\leq \left(C\sqrt{2\ln\left(\frac{16JT}{\delta}\right)} + 4\|\sigma\|_\infty \sqrt{J} + 2\|\sigma\|_\infty \sqrt{\ln\left(\frac{8TJ}{\delta}\right)} \right) \sqrt{\frac{T \sum_{t=1}^{T-1} \frac{1}{t}}{\bar{K}_T}} \\ &\leq \left(C\sqrt{2\ln\left(\frac{16JT}{\delta}\right)} + 4\|\sigma\|_\infty \sqrt{J} + 2\|\sigma\|_\infty \sqrt{\ln\left(\frac{8TJ}{\delta}\right)} \right) \sqrt{\frac{T \ln(T)}{\bar{K}_T}} \end{aligned} \tag{2.109}$$

where $\bar{K}_T := \frac{T}{\sum_{t=t_0}^{T-1} \frac{t-1}{\sum_{l=1}^{t-1} K_l}}$ is the average number of arms presented over the T steps, $K_T \geq K$. Therefore the regret is bounded with probability at least $1 - \delta$

$$R_T \leq \frac{2}{\|\sigma^{-1} \cdot \alpha\|} \left(1 + \frac{2L^2}{K} \max \left\{ \frac{64}{\mathbb{E}_{r\sim\nu}(r)^2 \min_{j,\alpha_j \neq 0} \alpha_j^2}, 1 \right\} \right) + \frac{16LS\Phi}{\mathbb{E}_{r\sim\nu}(r)} \max \left\{ 1, \frac{1}{\mathbb{E}_{r\sim\nu}(r)} \right\} \sqrt{\frac{T \ln(T)}{\bar{K}_T}} \tag{2.110}$$

where $S = \left(\|\alpha \cdot \sigma^{-2}\|_2 + \|\sigma^{-2}\|_2 \right) \frac{\|\sigma^{-2} \cdot \alpha\|_2}{\|\sigma^{-1} \cdot \alpha\|_2^4} + \left(\frac{\|\sigma^{-1}\|_4}{\|\sigma^{-1} \cdot \alpha\|_2} \right)^2$ and $L = C\sqrt{2\ln\left(\frac{16JT}{\delta}\right)} + 8\|\sigma\|_\infty \sqrt{J} + 4\|\sigma\|_\infty \sqrt{\ln\left(\frac{8TJ}{\delta}\right)}$

2.A.5 Regret Analysis

In Section 2.3.1, the regret has been defined as the difference between the top-K arms according to the oracle and the K arms selected by a learner, see (2.10). However one may argue that a more interesting notion of regret is to compare the actions of the oracle to the action of the learner with respect to the true reward of each arm. That is to say, to define the notion of *absolute* regret,

$$R_T^{\text{abs}} = \sum_{t=1}^{T} \sum_{i \in \mathcal{A}_t^+} r_{i,t} - \sum_{i \in \mathcal{A}_t} r_{i,t} \tag{2.111}$$

The two notions of regret, that is to say the *relative* and *absolute* regret are related,

$$R_T - R_T^{\text{abs}} = \sum_{t=1}^{T} \sum_{i \in \mathcal{A}_t^+} \langle w^+, g^{-1}(\phi_{i,t}) - g^{-1}(g(\alpha \cdot r_{i,t})) \rangle - \sum_{i \in \mathcal{A}_t} \langle w^+, g^{-1}(\phi_{i,t}) - g^{-1}(g(\alpha \cdot r_{i,t})) \rangle \tag{2.112}$$

However controlling the deviation here is not possible through standard concentration inequalities. Indeed the set of arms, $\mathcal{A}_t^+$ is computed by taking the argmax over the noise therefore $(\epsilon_{i,t})_{i\in\mathcal{A}_t^+}$ are not centered anymore, see App. 2.A.2. Nonetheless, one can still provide some guarantees on the regret $\tilde{R}_T^{\text{abs}}$.

Theorem 5. *For every $t \leq T$ let $(\phi_{(i),t})_{i\leq K_t}$ be the ordered set of arms with respect to the oracle $\widehat{\mathfrak{O}}$, that is to say $\langle w^\star, g^{-1}(\phi_{t,(1)})\rangle \geq \langle w^\star, g^{-1}(\phi_{t,(2)})\rangle \geq \ldots \geq \langle w^\star, g^{-1}(\phi_{t,(K_t)})\rangle$ let's now define the gap between the top-K according the oracle and the rest of the arms, $\Delta_{t,K} := \langle w^+, g^{-1}(\phi_{(K),t}) - g^{-1}(\phi_{(K+1),t})\rangle$. For any weights $w \in \mathbb{R}^J$ such that:*

$$\forall i \leq K_t, \qquad \left|\langle w^+ - w, g^{-1}(\phi_{i,t})\rangle\right| \leq \frac{\Delta_{t,K}}{4} \tag{2.113}$$

Then the top-K arms, $\mathcal{A} := \arg\max_i^K \langle w, \phi_{i,t}\rangle$ is exactly the ranking of the oracle, that is to say $\mathcal{A} = \mathcal{A}_t^+$.

Proof. For an arm $i \in \mathcal{A}_t^+$,

$$\begin{aligned}\langle w_t, g^{-1}(\phi_{i,t})\rangle \geq \langle w^\star, g^{-1}(\phi_{i,t})\rangle - \frac{\Delta_{t,K}}{4} &\geq \langle w^\star, g^{-1}(\phi_{(K),t})\rangle - \frac{\Delta_{t,K}}{4} \\ &\geq \langle w^\star, g^{-1}(\phi_{(K+1),t}))\rangle + \frac{3\Delta_{t,K}}{4} \\ &\geq \langle w_t, g^{-1}(\phi_{(K+1),t}))\rangle + \frac{\Delta_{t,K}}{2} > \langle w_t, g^{-1}(\phi_{(K+1),t}))\rangle\end{aligned}$$

Therefore $\mathcal{A}_t^+ \subset \mathcal{A}$. On the other hand for any $i \notin \mathcal{A}_t^+$, we have that:

$$\begin{aligned}\langle w_t, g^{-1}(\phi_{i,t})\rangle \leq \langle w^\star, g^{-1}(\phi_{(K+1),t}))\rangle + \frac{\Delta_{t,K}}{4} &\leq \langle w^\star, g^{-1}(\phi_{(K),t})\rangle - \frac{3\Delta_{t,K}}{4} \\ &\leq \langle w_t, g^{-1}(\phi_{(K),t})\rangle - \frac{\Delta_{t,K}}{2} < \langle w_t, g^{-1}(\phi_{(K),t})\rangle\end{aligned}$$

Therefore, the set $\mathcal{A}_t^+ \subset \mathcal{A}$ but because both are of size K, we have that $\mathcal{A} = \mathcal{A}_t^+$. □

The main implication of Thm. 5 is that we can now provide a bound on the absolute regret of GLM-ε-GREEDY. Indeed, thanks to the bound of Lem. 9, for $t \geq \max\left\{\left(\frac{2C^2\sqrt{2\ln\left(\frac{8T}{\delta}\right)}}{\varepsilon K\eta_\nu^2}\right)^2, \frac{32\left(\sqrt{\lambda}\|\alpha\|+\|\sigma\|_\infty\left(\sqrt{\ln\left(1+\frac{2T}{\lambda}\right)+\ln\left(\frac{8JT}{\delta}\right)}\right)\right)^2}{\varepsilon K\eta_\nu^2 \min_{t\leq T}\Delta_{t,K}^2}\right\}$, the error between the estimated weights w_t and the optimal oracle weights w^+ is bounded by $\Delta_{t,K}/4$, therefore the absolute regret of Alg. 1 is bounded by

$$R_T^{\text{abs}} \leq C\max\left\{\left(\frac{2C^2\sqrt{2\ln\left(\frac{8T}{\delta}\right)}}{\varepsilon K\eta_\nu^2}\right)^2, \frac{32\left(\sqrt{\lambda}\|\alpha\| + \|\sigma\|_\infty\left(\sqrt{\ln\left(1+\frac{2T}{\lambda}\right)+\ln\left(\frac{8JT}{\delta}\right)}\right)\right)^2}{\varepsilon K\eta_\nu^2 \min_{t\leq T}\Delta_{t,K}^2}\right\} \tag{2.114}$$

$$+\frac{4\Phi}{\|\alpha\cdot\sigma^{-1}\|}\left(\sqrt{2T\ln\left(\frac{4T}{\delta}\right)} + \varepsilon T\right) \tag{2.115}$$

Choosing $\varepsilon = \min\left\{T^{-1/3}, \min_{t\leq T}\Delta_{t,K}^2, \frac{1}{\sqrt{T}\min_{t\leq T}\Delta_{t,K}}\right\}$ yields an absolute regret of:

$$R_T^{\text{abs}} = \widetilde{\mathcal{O}}\left(T^{2/3}, \frac{\sqrt{T}}{\min_{t\leq T}\Delta_{t,K}}, \frac{1}{\min_{t\leq T}\Delta_{t,K}^4}\right)$$

. Therefore, when $\min_{t\leq T}\Delta_{t,K} \geq T^{-1/6}$ the absolute regret and relative regret of Alg. 1 scales similarly with a relative and absolute regret of order $\widetilde{\mathcal{O}}(T^{2/3})$.

2.A.6 Additional Experiments

In this subsection, we present in details the algorithms used in Sec. 4.1.5, the experimental protocol and additionnal experimental results.

Algorithm 5: GLM-ε-GREEDY-ALL SAMPLES algorithm

Input: Noise parameters $\{\sigma_j\}_{j\leq J}$, confidence level δ
Parameters: exploration level ε; number of arms to pull K; regularization λ
Set $\mathcal{H}_0 = \emptyset$, $\widehat{\alpha} = 0$ and $w_0 = 0$
for $t = 1, \dots, T$ **do**
  Sample $Z_t \sim \text{Ber}(\varepsilon)$
  Observe evaluations for each arm $(\phi_{i,t})_{i\leq K_t}$
  if $Z_t = 1$ **then**
    Pull arms in $\mathcal{A}_t$ obtained by sampling K arms uniformly in $\{1, \dots, K_t\}$
    Observe rewards $r_{i,t}$ for all $i \in \mathcal{A}_t$
  else
    Select $\mathcal{A}_t = \arg\max_i^K \langle w_t, g^{-1}(\phi_{i,t})\rangle$
  Add sample to dataset $\mathcal{H}_t = \mathcal{H}_{t-1} \cup \left(\cup_{i\in\mathcal{A}_t} \{(\phi_{i,t}, r_{i,t})\}\right)$
  Update estimators $\widehat{\alpha}_{j,t}$ by solving

$$\sum_{\phi,r\in\mathcal{H}_t} r(g(\widehat{\alpha}_{t,j}\cdot r) - \phi_j) - \lambda\widehat{\alpha}_{t,j} = 0 \tag{2.116}$$

  Update weights $w_{t,j} = \widehat{\alpha}_{t,j}/(\sigma_j^2)\|\widehat{\alpha}_t \cdot \sigma^{-1}\|^2$

2.A.6.1 Baseline Algorithms

In Sec. 4.1.5, we compare our algorithms, GLM-ε-GREEDY (Alg. 1) and ESAG (Alg. 2), to different baselines.

GLM-ε-greedy-ALL (Alg. 5). This algorithm is the s Alg. 1 and illustrate the problem with the noise correlation of App. 2.A.2. This algorithm thus updates its own MLE estimator at every time step instead of step with a random exploration.

GLM-EvalBasedUCB (Alg. 6). This algorithm does not rely on an ε-greedy type of exploration but on an optimistic approach. Similar to other algorithms, it first computes an MLE estimator of α using all samples collected over time. Then it build a "confidence" set around the estimated $\widehat{\alpha}$ and it computes weights w_t as those resulting from a worst-case choice of a parameter $\widetilde{\alpha}$ in terms of error. Notice that the confidence sets are not theoretically justified because they are designed by ignoring the correlation between decisions and evaluations. As such, this algorithm has not theoretical guarantee and it should be considered as a heuristic. We provide further details in Sect. 2.A.6.2.

RAND (Alg. 7). This baseline algorithm is simply selecting one random evaluators every step and computing a ranking of the arms based on this random evaluators.

GLM-Greedy. This algorithm is a variant of GLM-ε-GREEDY-ALL with $\varepsilon = 0$.

2.A.6.2 Derivation of GLM-EvalBasedUCB

Here, we breifly present how we derive the GLM-EVALBASEDUCB algorithm. This algorithm takes a *optimistic approach* w.r.t. the set of plausible α values and compute a set of weights w_t as the set of oracle weights but for a different value of α. The first step is to compute an estimator $\widetilde{\alpha}_t$ such that:

$$\tilde{\alpha}_t = \arg \max_{z\in\mathbb{R}^J : \|\hat{\alpha}_t - z\| \leq \beta_t^E} \min_{w\in\mathbb{R}^J, \langle w,z\rangle = 1} \sum_{j=1}^J w_j^2\sigma_j^2 \tag{2.118}$$

where β_t^E is the width of the confidence intervals defined in Eq. (2.96) Solving the minimization problem in the previous equation implies that $\widetilde{\alpha}_t$ is solution to the following equivalent optimization problem,

$$\min_{\|z-\hat{\alpha}_t\|_2\leq\beta_t^E} \|\sigma^{-1}\cdot z\|_2^2 = \min_{\|u\|\leq 1} \|\sigma^{-1}\cdot(\hat{\alpha}_t + \beta_t^E u)\|_2^2 \tag{2.119}$$

with $\widehat{\alpha}_t = \dfrac{\sum_{l=1}^{t-1}\sum_{i\in\mathcal{A}_l} g^{-1}(\phi_{i,l})}{\sum_{l=1}^{t-1}\sum_{i\in\mathcal{A}_l} r_{i,l}}$ the average features observed until time t. Eq. (2.119) is a convex problem and using KKT conditions, we distinguish two different situations:

Algorithm 6: GLM-EvalBasedUCB algorithm

Input: Noise parameters $\{\sigma_j\}_{j\leq J}$, confidence level δ
Set $\mathcal{H}_0 = \emptyset$, $\widehat{\alpha} = 0$ and $w_0 = 0$
for $t = 1, \dots, T$ **do**
  Observe evaluations for each arm $(\phi_{i,t})_{i\leq K_t}$
  Compute $\widehat{\alpha}_t = \frac{\sum_{l=1}^{t-1}\sum_{i\in\mathcal{A}_l} g^{-1}(\phi_{i,l})}{\sum_{l=1}^{t-1}\sum_{i\in\mathcal{A}_l} r_{i,l}}$ and

$$\beta_t^E = \frac{\frac{2\sqrt{2J\ln(2/\delta)}}{3} + \sqrt{(t-1)K\sum_{j=1}^J \sigma_j^2 \ln(2/\delta)}}{\sum_{l=1}^{t-1}\sum_{i\in\mathcal{A}_t} r_{i,l}} \tag{2.117}$$

  if $\|\widehat{\alpha}_t\|_2 \leq \beta_t^E$ **then**
    Compute weights $w_t = \frac{\sigma^{-2}\hat{\alpha}_t}{\|\sigma^{-1}\hat{\alpha}_t\|^2}$
  else
    Compute $\lambda_t > 0$ solving $\sum_{j=1,\sigma_j>0}^J \left(\frac{\widehat{\alpha}_{t,j}}{\lambda_t\sigma_j^2 + (\beta_t^E)^2}\right)^2 = \frac{1}{\beta_t^2}$
    Compute $\widetilde{\alpha}_t = \left(I_J - \beta_t^2\left(\lambda_t I_J + (\beta_t^E)^2\sigma^{-2}\right)^{-1}\sigma^{-2}\right)\widehat{\alpha}_t$ and $w_t = \frac{\sigma^{-2}\widetilde{\alpha}_t}{\|\sigma^{-1}\widetilde{\alpha}_t\|_2}$
  Select $\mathcal{A}_t = \arg\max_i^K \langle w_t, g^{-1}(\phi_{i,t})\rangle$ and observe rewards $r_{i,t}$ for all $i \in \mathcal{A}_t$

Algorithm 7: Rand algorithm

Parameters: number of arms to pull K
for $t = 1, \dots, T$ **do**
  Randomly select $j \leq J$
  Observe evaluations for each arm $(\phi_{i,t})_{i\leq K_t}$
  Select $\mathcal{A}_t = \arg\max_i^K g^{-1}(\phi_{i,t,j})$

1. If $\|\hat{\alpha}_t\| \leq \beta_t^E$ then $\min_{\|z-\hat{\alpha}_t\|_2\leq\beta_t^E} \|\sigma^{-1}\cdot z\|_2^2 = 0$ thus the problem is ill-defined and the weights are also not defined (A necessary condition for this condition to be fulfilled is that $\|\alpha\| \leq \beta_t^E(1+\sqrt{1+(\beta_t^E)^2})$)
2. If $\|\hat{\alpha}_t\| > \beta_t^E$ then the solution is such that $u = -\beta_t^E\left(\lambda_t I_J + (\beta_t^E)^2\text{Diag}(\sigma^{-2})\right)^{-1}\sigma^{-2}\cdot\hat{\alpha}_t$ where $\lambda_t > 0$ is solution to[19]:

$$\sum_{j=1,\sigma_j>0}^J \left(\frac{\hat{\alpha}_{t,j}}{\lambda_t\sigma_j^2 + (\beta_t^E)^2}\right)^2 = \frac{1}{(\beta_t^E)^2} \tag{2.120}$$

   Overall, $\widetilde{\alpha}_t$ is defined as:

$$\widetilde{\alpha}_t = \left(I_J - (\beta_t^E)^2\left(\lambda_t I_J + (\beta_t^E)^2\text{Diag}(\sigma^{-2})\right)^{-1}\sigma^{-2}\right)\hat{\alpha}_t, \tag{2.121}$$

   that is to say the weights used by the algorithm is $w_t = \frac{\Sigma^{-2}\tilde{\alpha}_t}{\|\Sigma^{-1}\tilde{\alpha}_t\|_2^2}$

2.A.6.3 Additional Synthetic Experiments

We report here the regret and estimation bias for the in the logisitic and linear setting for the three different regimes of $\frac{\alpha_0}{\sigma_0} \in \{0.1, 1, 10\}$ as reported in Section 4.1.5. For both the linear and logistic case we choose $K = 10$ and $K_t = 60$. For GLM-ε-Greedy and GLM-ε-greedy-ALL SAMPLES, we use different values of $\varepsilon \in \{0.1T^{-1/3}, 0.1T^{-1/2}\}$. The rewards are sampled from a centered unit variance normal distribution truncated between $[0, 20]$.

[19] A solution λ_t always exist when there is at least one evaluator j such that $\sigma_j > 0$ because the function $f : \lambda \mapsto \sum_{j=1,\sigma_j>0}\left(\frac{\hat{\alpha}_{t,j}}{\lambda\sigma_j^2+(\beta_t^E)^2}\right)^2$ is strictly non-increasing over $\mathbb{R}^+$ and for $\lambda = 0$, we have $f(0) = \|\hat{\alpha}_t\|^2/(\beta_t^E)^4 > (\beta_t^E)^{-2}$ and $\lim_{\lambda\to+\infty} f(\lambda) = 0$ so by continuity there exists a unique solution $\lambda_t > 0$.

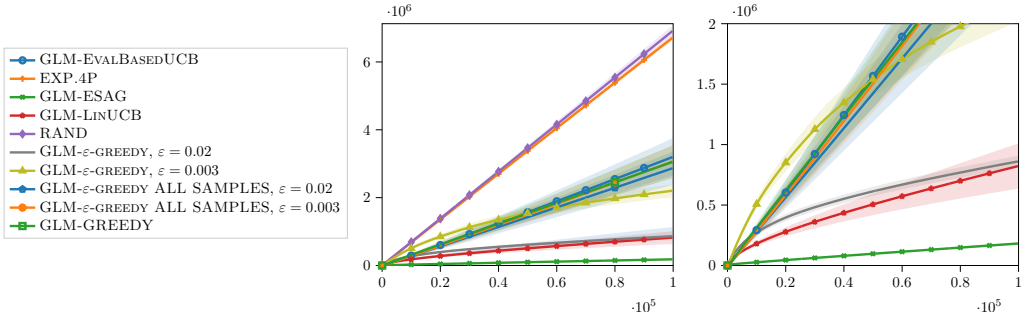


Figure 2.A.1: $\frac{\alpha_0}{\sigma_0} = 0.1$

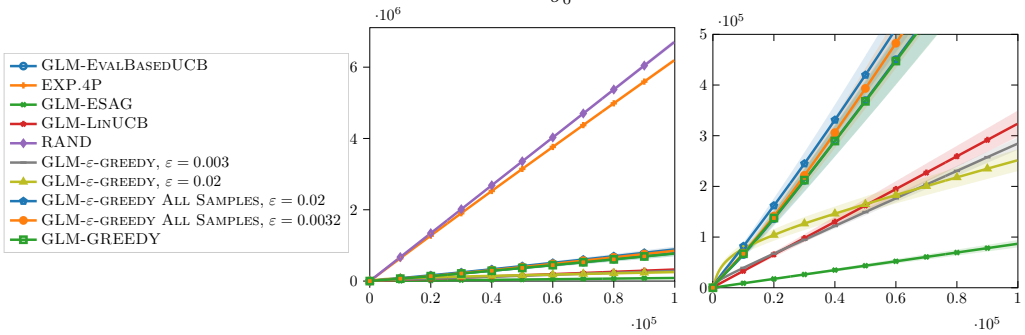


Figure 2.A.2: $\frac{\alpha_0}{\sigma_0} = 1$

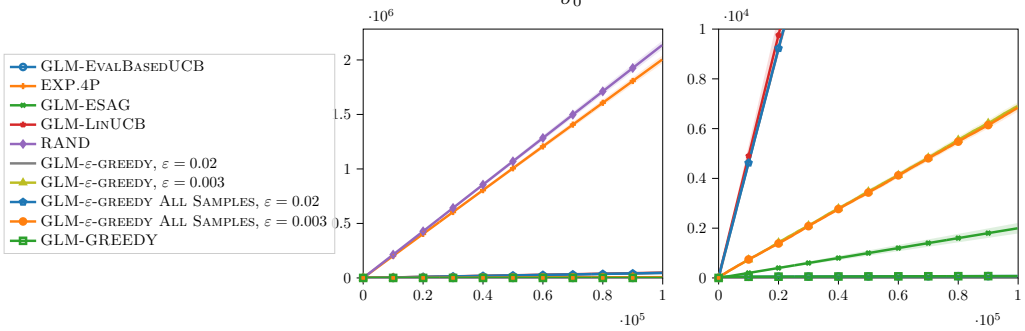


Figure 2.A.3: $\frac{\alpha_0}{\sigma_0} = 10$

Figure 2.A.4: Regret (*Left*) and zoomed in regret (*Right*, removing RAND and EXP4.P algorithms) wrt to the oracle $\mathfrak{O}$ in Section 2.1.3 for different values of $\frac{\alpha_0}{\sigma_0}$

Logistic and Linear Experiments. As highlighted in Section 4.1.5, for a small ratio $\frac{\alpha_0}{\sigma_0}$, the algorithms GLM-$\varepsilon$-GREEDY-ALL SAMPLES or GLM-EvalBasedUCB that use all samples suffer from a linear regret compared to our algorithm Alg. 1 for $\varepsilon = \mathcal{O}(T^{-1/3})$. In addition, we observe that as the ratio $\frac{\alpha_0}{\sigma_0}$ increases the correlation effect highlighted in App. 2.A.2 becomes less and less relevant and GLM-EvalBasedUCB performs particularly well.

2.A.6.4 Second Content Review Prioritization Dataset.

In order to validate our observations of Section 4.1.5 on the content review proritization dataset, we consider a second small dataset ($D2$) of content similar to the one used in Section 4.1.5. Table 2 sums up the cumulative reward of each algorithms at $T = 2000$ steps selecting $K = 10$ out of $K_t = 200$ arms. Similarly to Table 1 linear algorithms performs best compared to the logistic algorithms with the same three algorithms getting the best performance.

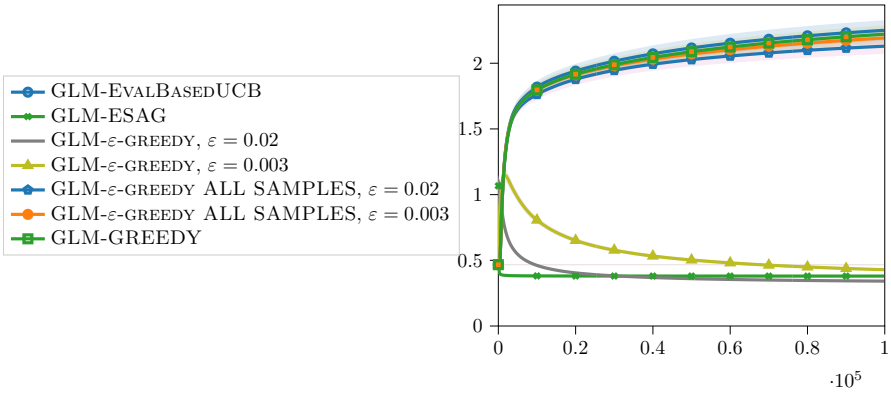


Figure 2.A.5: $\frac{\alpha_0}{\sigma_0} = 0.1$

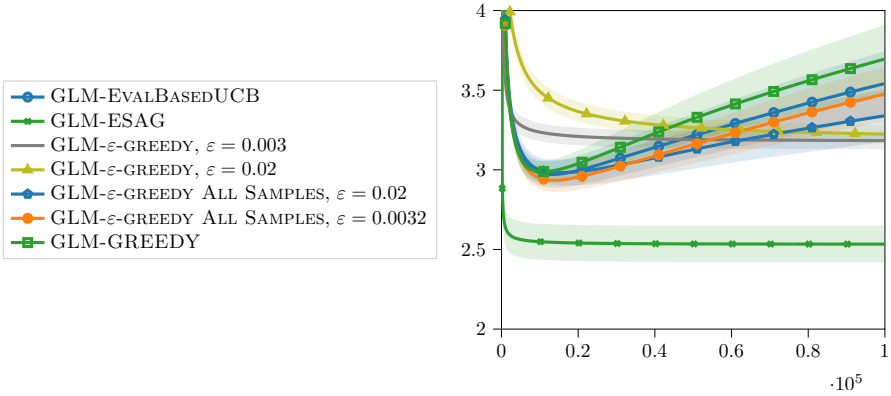


Figure 2.A.6: $\frac{\alpha_0}{\sigma_0} = 1$

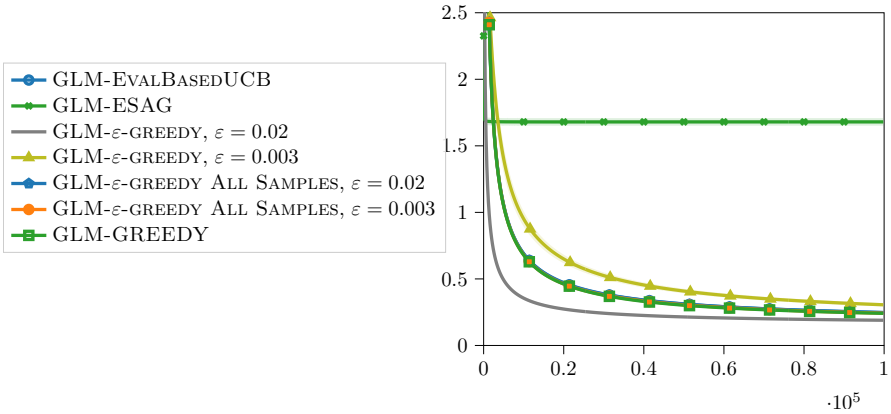
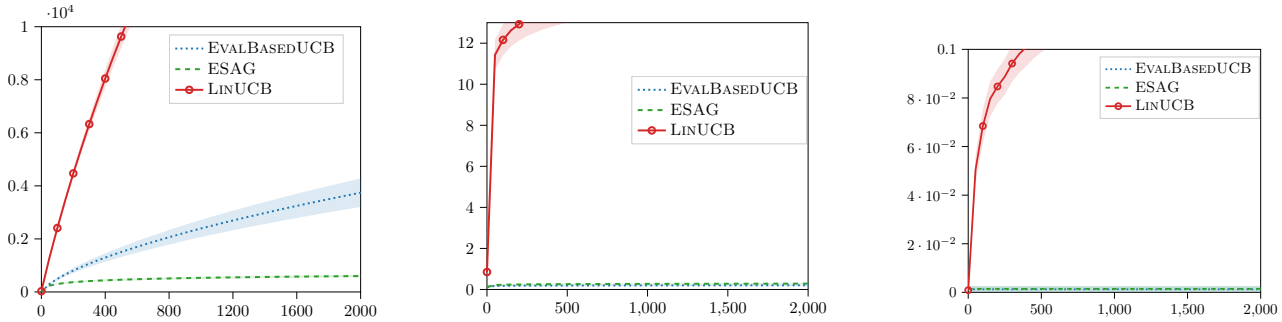


Figure 2.A.7: $\frac{\alpha_0}{\sigma_0} = 10$

Figure 2.A.8: Estimation error for the α parameters for different values of $\frac{\alpha_0}{\sigma_0}$



(a) $\frac{\alpha_0}{\sigma_0} = 0.1$ (b) $\frac{\alpha_0}{\sigma_0} = 1$ (c) $\frac{\alpha_0}{\sigma_0} = 10$

Figure 2.A.9: Regret w.r.t. to the oracle $\mathfrak{O}$ in Section 2.1.4 for different values of $\frac{\alpha_0}{\sigma_0}$

Table 2.A.1: Cumulative badness on $D2$ for $T = 2000$ steps

Alg.	Badness
Rand	60025.8
GLM-ε-greedy	105965.8
GLM-ε-greedy-all	105499.1
GLM-EvalBasedUCB	105545.4
GLM-LinUCB	60347.4
GLM-ESAG	106237.5
GLM-Greedy	105264.3
Exp4.P	88979.4
EvalBasedUCB	**138230.1**
LinUCB	**144009.4**
ESAG	**141172.1**
Greedy	106195.4

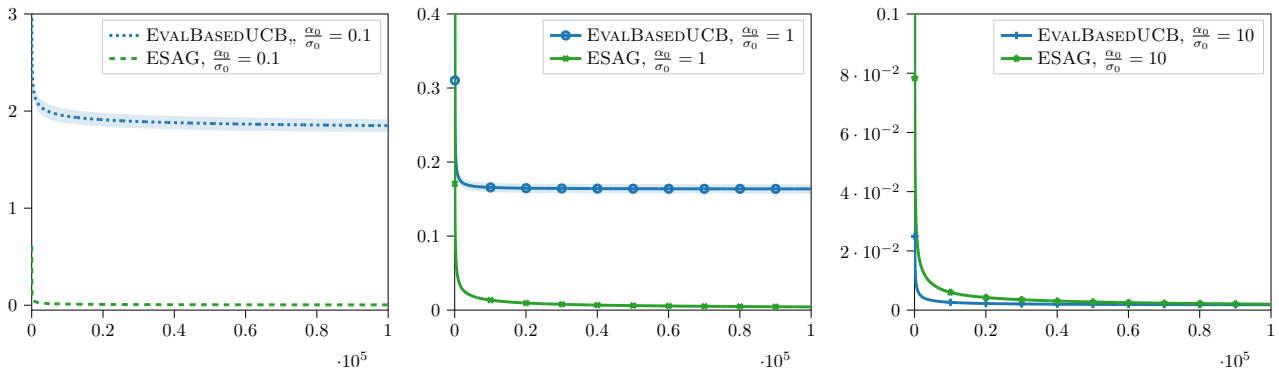


Figure 2.A.10: Estimation Bias in the linear case. For ESAG, we compute the error $\|\widetilde{\alpha}_t - \mathbb{E}_{r\sim\nu}(r)\alpha\|$

Table 2.A.2: Statistics about the evaluators. We report the R2 score of the evaluators w.r.t. the true ratings. The column $\widehat{\sigma}$ is the maximum estimate standard deviation.

algo	R2	$\widehat{\sigma}$
rf5-s18	0.864	0.115
dt-s50	0.448	0.174
linear	0.348	0.167
adaboost	0.440	0.048
mlp	0.917	0.110
rf	0.936	0.064
adaboost20-s60	0.309	0.086

Table 2.A.3: Average reward after $T = 6000$ steps in the Jester experiment averaged over 50 runs.

Algorithm	Average Reward $\frac{1}{T}\sum_{i=1}^{T} r_i$
EvalBasedUCB	0.87
ESAG	0.86
LinUCB	0.85
GLM-EvalBasedUCB	0.69
GLM-Greedy	0.69
GLM-ESAGesaglogistic	0.69
GLM-ε-greedy	0.69
RAND	0.57
GLM-LinUCB	0.51

2.A.6.5 Jester Experiments.

We consider the Jester dataset (Goldberg et al., 2001), which consists of joke ratings in a continuous range from -10 to 10 for a total of 100 jokes and 73421 users. We consider the same set of users and jokes as in (Riquelme et al., 2018). For a subset of 40 jokes and 19181 users rating all these 40 jokes, we build evaluators as follows. We fit 7 *contextual* models to predict the ratings from features –obtained as concatenation of user and joke features– extracted via a low-rank factorization of the full matrix (of dimension 36). Ratings are normalized and transformed through the logarithmic function. We trained the following models:[20]

- rf5-s18: a random forest with 5 trees trained over 5 randomly selected features;
- dt-s50: a decision tree with max depth 10 trained over 50 randomly selected features;
- linear: a linear model with intercept;
- adaboost: an implementation of AdaBoost.R2 with 20 trees;
- mlp: a neural network with two hidden layers ($[512, 128]$) and ReLu activation;
- adaboost20-s60: AdaBoost.R2 with 20 trees trained over 60 randomly selected features.

We use the predictions of each model as inputs for our algorithms. As rewards, we use the real ratings and we add to them zero-mean Gaussian noise (standard deviation is 0.5). The resulting problem is thus misspecified since the evaluators are not perfect, as shown in Tab. 2.A.2.

We estimate the parameters of the algorithms by computing statistics about the relationship between true ratings and predicted ones. For ε-greedy, we use the value $T^{1/3}$. From Tab. 2.A.3, we can see that linear algorithms outperform the logistic algorithms. EvalBasedUCB and ESAG behave similarly and perform better than LinUCB.

[20] We used the implementations provided by scikit-learn (Pedregosa et al., 2011).

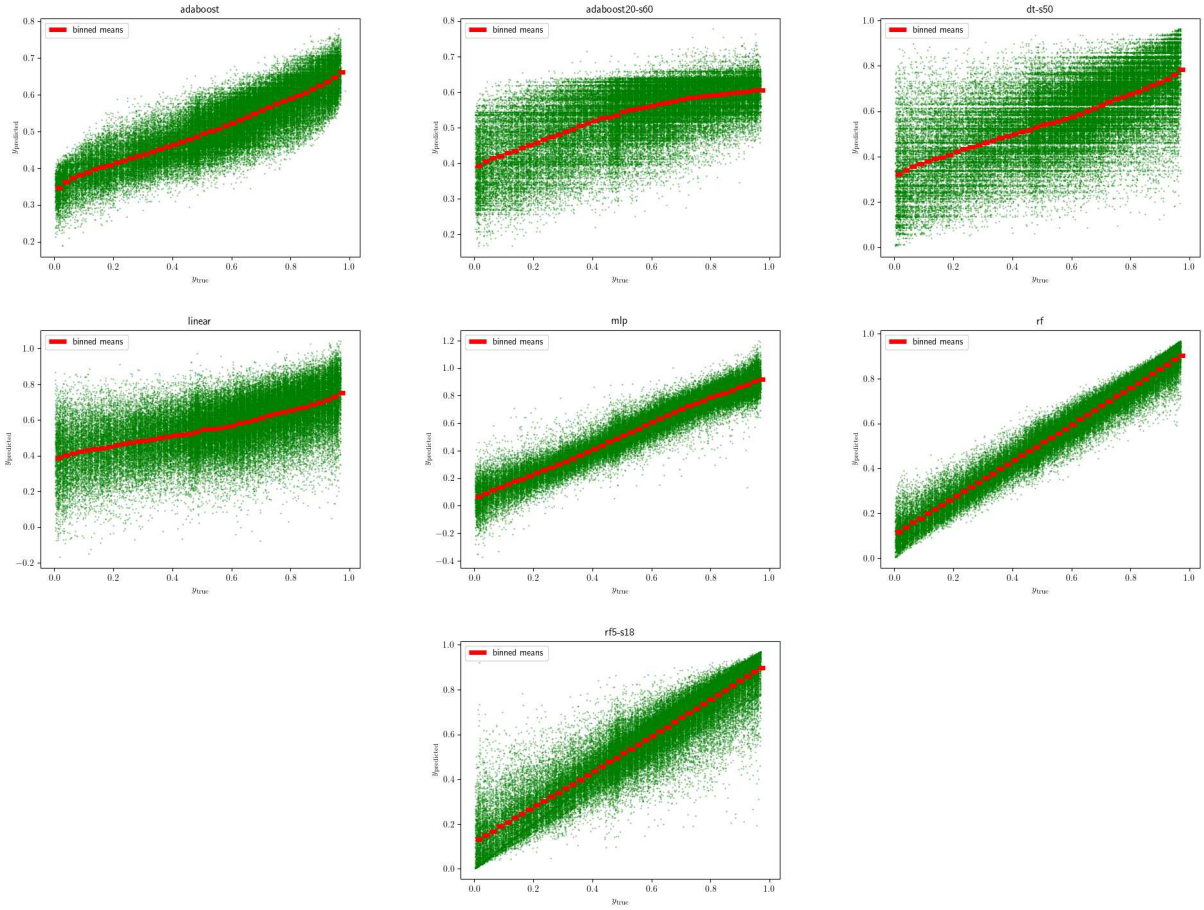


Figure 2.A.11: We report the predictions of the evaluators as a function of the true ratings (for clarity we reported only 10% of the samples). We also report the average value over a discretization of the ratings into 40 bins.

2.B Appendix for Improved Conservative Exloration for Linear Contextual Bandit

2.B.1 Proofs

2.B.1.1 Proof of Thm. 3

We define three sets of time steps: $S_t^\star$ is the steps before t (t included) where the optimistic arm is in the constraint set of (2.23) and it is thus selected, S_t^+ is the steps where the optimistic arm is not in the constraint set and the algorithm did not select the baseline b_t, finally let $S_t^b := [t] \setminus \left(S_t^\star \cup S_t^+\right)$ be the remaining time steps at which the baseline was played. We consider the high-probability event in which $\theta^\star \in \Theta_t$ (i.e., the true linear parameter belongs to the confidence ellipsoid). For any action a and any time t, we introduce the notation

$$\widehat{\mu}_a^t := \left\langle \hat{\theta}_t, x_{t,a} \right\rangle \tag{2.122}$$

$$\widetilde{\mu}_a^t := \left\langle \hat{\theta}_t, x_{t,a} \right\rangle + \beta_t \left|\left| x_{t,a_t} \right|\right|_{V_t^{-1}}. \tag{2.123}$$

The regret can be decomposed as

$$
\begin{aligned}
R_{\text{CLUCB2}}(n) &= \sum_{t\in S_n^\star}\left(\mu_{a_t^\star}^t - \mu_{a_t}^t\right) + \sum_{t\in S_n^+}\left(\mu_{a_t^\star}^t - \mu_{a_t}^t\right) + \sum_{t\in S_n^b}\left(\mu_{a_t^\star}^t - \mu_{b_t}^t\right) \\
&\overset{(a)}{\leq} \sum_{t\in S_n^\star} \mu_{a_t^\star}^t - \mu_{a_t}^t + \sum_{t\in S_n^+} \mu_{a_t^\star}^t - \mu_{a_t}^t + \Delta_h\left|S_n^b\right| \\
&\overset{(b)}{\leq} \sum_{t\in S_n^\star}\left(\widetilde{\mu}_{a_t}^t - \mu_{a_t}^t\right) + \sum_{t\in S_n^+}\left(\mu_{a_t^\star}^t - \mu_{b_t}^t\right) + \sum_{t\in S_n^+}\left(\mu_{b_t}^t - \mu_{a_t}^t\right) + \Delta_h\left|S_n^b\right| \\
&\overset{(c)}{\leq} \sum_{t\in S_n^\star} 2\beta_t\left|\left|x_{t,a_t}\right|\right|_{V_t^{-1}} + \sum_{t\in S_n^+}\left(\mu_{b_t}^t - \mu_{a_t}^t\right) + \Delta_h\left(\left|S_n^b\right| + \left|S_n^+\right|\right) \\
&\overset{(d)}{\leq} \sum_{t\in S_n^\star\cup S_n^+} 2\beta_t\left|\left|x_{t,a_t}\right|\right|_{V_t^{-1}} + \Delta_h\left(\left|S_n^b\right| + \left|S_n^+\right|\right), \qquad (2.124)
\end{aligned}
$$

where (a) is using the upper bound Δ_h to the per-step regret of the baseline, (b) follows from the high-probability upper-confidence bounds using in LinUCB steps, (c) is using the definition of the confidence ellipsoid and the bound on the baseline regret, and (d) follows by the definition of S_n^+. In fact, at any time $t \in S_n^+$, an arm different from the UCB arm and the baseline is selected following the action selection in (2.23). Since the baseline arm belongs to the constraint set by definition, then we have

$$
\mu_{b_t}^t \leq \left\langle \hat{\theta}_t, x_{t,a_t}\right\rangle + \beta_t\left|\left|x_{t,a_t}\right|\right|_{V_t^{-1}} \leq \mu_{a_t}^t + 2\beta_t\left|\left|x_{t,a_t}\right|\right|_{V_t^{-1}}. \qquad (2.125)
$$

Recalling the definition of β_t and using Theorem 2 in Abbasi-Yadkori et al. (2011), we obtain

$$
\sum_{t\in S_n^\star\cup S_n^+} 2\beta_t\left|\left|x_{t,a_t}\right|\right|_{V_t^{-1}} \leq 4\sqrt{nd\log\left(1+\frac{nD^2}{\lambda d}\right)}\left[\sqrt{\lambda}B + \sigma\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1+\frac{nD^2}{\lambda d}\right)}\right]
$$

Finally, we need to bound the number of times $\left|S_n^+\right| + \left|S_n^b\right|$ the optimistic arm was not selected (i.e., it was not in the constraint set).

Lemma 12. *For any $\delta > 0$, we have with probability $1-\delta$ that :*

$$
\left|S_n^b\right| + \left|S_n^+\right| \leq \frac{2\mu_h}{\alpha\mu_l} + \frac{4}{3\alpha\mu_l}L_\alpha + \frac{4}{\sqrt{3}(\alpha\mu_l)^{3/2}}\Big(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\Big)L_\alpha + \frac{2}{(\alpha\mu_l)^2}\Big(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\Big)^2 L_\alpha \qquad (2.126)
$$

where L_α is a logarithmic term,

$$
\begin{aligned}
L_\alpha =& \log\left[\frac{\sqrt{D_0}}{\sqrt{\alpha\mu_l\delta}} + \frac{12\sqrt{D_0}}{\sqrt{\delta}}\left(\frac{4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma}{\alpha\mu_l}\right)^2 \log\left(\frac{36\sqrt{D_0}}{\sqrt{\delta}\alpha\mu_l}\Big(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\Big)\right)^2\right]\times \\
&\times \log\left(\frac{36}{\alpha\mu_l}\sqrt{\frac{D_0}{\delta}}\Big(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\Big)\right)
\end{aligned}
$$

Including this result into the regret decomposition provides the final bound

$$
\begin{aligned}
R_{\text{CLUCB2}}(n) \leq& 4\sqrt{nd\log\left(1+\frac{nD^2}{\lambda d}\right)}\times\left[\sqrt{\lambda}B + \sigma\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1+\frac{nD^2}{\lambda d}\right)}\right] + \frac{2\mu_h}{\alpha\mu_l} + \frac{4}{3\alpha\mu_l}L_\alpha \\
&+ \frac{4}{\sqrt{3}(\alpha\mu_l)^{3/2}}\Big(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\Big)L_\alpha + \frac{2}{(\alpha\mu_l)^2}\Big(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\Big)^2 L_\alpha
\end{aligned}
$$

where $D_0 := \max\left\{2D^2/\lambda, 3\right\}$.

Proof of Lemma 12. Let $\tau := \max\left\{\ t \in [n]\ \ | \ t \in S_n^+ \cup S_n^b\right\}$, i.e., the last time the optimistic arm was not in the constraint set (and either the baseline or another arm was selected). Let $\widetilde{a}_\tau$ the optimistic arm at time τ, since it does not satisfy the constraint, we have

$$\sum_{t\in S_{\tau-1}} r_{a_t}^t - \psi_L(\tau) + \max\left\{\min_{\theta\in\Theta_\tau}\langle\theta, x_{\tau,\widetilde{a}_\tau}\rangle, 0\right\} + \sum_{t\in S_{\tau-1}^b}\mu_{b_t}^t \leq (1-\alpha)\sum_{t=1}^{\tau}\mu_{b_t}^t. \tag{2.127}$$

Reordering the baseline terms and recalling that $S_t = [t]\backslash S_t^b = S_t^\star \cup S_t^+$, we obtain

$$\alpha\sum_{t=1}^{\tau}\mu_{b_t}^t \leq \mu_{b_\tau}^\tau + \sum_{t\in S_{\tau-1}}\mu_{b_t}^t - \sum_{t\in S_{\tau-1}} r_{a_t}^t + \psi_L(\tau) - \max\left\{\min_{\theta\in\Theta_\tau}\langle\theta, x_{\tau,\widetilde{a}_\tau}\rangle, 0\right\}. \tag{2.128}$$

Using $\mu_{b_\tau}^\tau \leq \mu_h$ and since the last term is non-negative, we can further simplify the expression as

$$\alpha\sum_{t=1}^{\tau}\mu_{b_t}^t \leq \mu_h + \sum_{t\in S_{\tau-1}}\left(\mu_{b_t}^t - r_{a_t}^t\right) + \psi_L(\tau). \tag{2.129}$$

Using the same Friedman inequality as in the construction of the Martingale lower bound and the fact that whenever the algorithm does not select the baseline, the chosen arm is "optimistic" w.r.t. the baseline (see (2.125)) we have

$$\sum_{t\in S_{\tau-1}}\left(\mu_{b_t}^t - r_{a_i}^i\right) \leq \sum_{t\in S_{\tau-1}}\left(\mu_{b_t}^t - \mu_{a_t}^t\right) + \psi_L(\tau) \leq \sum_{t\in S_{\tau-1}} 2\beta_t\left|\left|x_{t,a_t}\right|\right|_{V_t^{-1}} + \psi_L(\tau) \tag{2.130}$$

$$\leq 4d\left(\sqrt{\lambda}B + \sigma\right)\sqrt{\left|S_{\tau-1}\right| + 1}\log\left(\frac{2D^2}{\lambda\delta}\left(\left|S_{\tau-1}\right| + 1\right)\right) + \psi_L(\tau), \tag{2.131}$$

where the last step follows from Lemma 4 in Kazerouni et al. (2017). As $\tau = 1 + |S_{\tau-1}^\star| + |S_{\tau-1}^+| + |S_{\tau-1}^b|$, we can lower-bound the LHS of (2.129) as

$$\alpha\sum_{t=1}^{\tau}\mu_{b_t}^t \geq \alpha\mu_l(1 + |S_{\tau-1}^\star| + |S_{\tau-1}^+| + |S_{\tau-1}^b|) \geq \frac{\alpha\mu_l}{2}(1 + |S_{\tau-1}^+| + |S_{\tau-1}^b|) + \frac{\alpha\mu_l}{2}(1 + |S_{\tau-1}^\star| + |S_{\tau-1}^+|). \tag{2.132}$$

Plugging these results back into (2.129) and using the definition of $\psi_L(\tau)$ we obtain

$$\frac{\alpha\mu_l}{2}\left(\left|S_{\tau-1}^b\right| + \left|S_{\tau-1}^+\right| + 1\right) \leq -\frac{\alpha\mu_l}{2}\left(\left|S_{\tau-1}^\star\right| + \left|S_{\tau-1}^+\right| + 1\right) + \mu_h + \frac{2}{3}\log\left(\left(1 + \left|S_{\tau-1}^\star\right| + \left|S_{\tau-1}^+\right|\right)\frac{D_0}{\delta}\right)$$
$$+ \left(4d\left(\sqrt{\lambda}B + \sigma\right) + \sqrt{2}\sigma\right)\sqrt{\left|S_{\tau-1}^\star\right| + \left|S_{\tau-1}^+\right| + 1}\log\left(\left(1 + \left|S_{\tau-1}^\star\right| + \left|S_{\tau-1}^+\right|\right)\frac{D_0}{\delta}\right)$$

where $D_0 := \max\left\{2D^2/\lambda, 3\right\}$. To finish, bounding the number of rounds where the algorithm played we use the following lemma (Lemma 13):

Lemma 13. *For any $x \geq 2$ and $a_1, a_2, a_3, a_4 > 0$ such that $a_2 \geq 2$, the function $f : \mathbb{R}^+ \to \mathbb{R}$ with $f(x) = a_1\sqrt{x}\log(a_2 x) + a_4\log(a_2 x) - a_3 x$, is bounded as follows:*

$$\max_{x\geq 2} f(x) \leq \left(a_1\sqrt{\frac{2a_4}{a_3}} + \frac{8a_1^2}{3a_3}\log\left(\frac{18a_1\sqrt{a_2}}{a_3}\right)\right)\log\left(\frac{\sqrt{a_2}}{e}\sqrt{\frac{2a_4}{a_3} + \frac{64}{9}\left(\frac{a_1}{a_3}\right)^2\log\left(\frac{18a_1\sqrt{a_2}}{a_3}\right)^2}\right)$$
$$+ a_4\log\left[a_2\left(\frac{2a_4}{a_3} + \frac{64}{9}\left(\frac{a_1}{a_3}\right)^2\log\left(\frac{18a_1\sqrt{a_2}}{a_3}\right)^2\right)\right]$$

Proof. f is a strictly concave function and goes to $-\infty$ as $x \to +\infty$ thus it admits a unique maximum, noted $x^\star$. Moreover, by putting the gradient of f to zero, we have that:

$$a_3 x^\star = a_1\sqrt{x^\star}\log\left(\sqrt{a_2 x^\star}e\right) + a_4 \tag{2.133}$$

Thus injecting equation 2.133 into the definition of f, we have that:

$$\max_{x\geq 2} f(x) = f(x^\star) \leq \frac{a_1}{2}\sqrt{x^\star}\log\left(a_2 x^\star\right) + a_4 \log\left(\frac{a_2 x^\star}{e}\right)$$

Finally, using eq. 2.133 and Lemma 8 of Kazerouni et al. (2017), we get:

$$x^\star \leq \frac{4a_1^2}{a_3^2}\log\left(\frac{4a_1\sqrt{a_2}e}{a_3}\right)^2 + \frac{2a_4}{a_3}$$

Hence, putting everything together we have that:

$$\max_{x\geq 2} f(x) \leq \left(a_1\sqrt{\frac{a_4}{2a_3}} + \frac{a_1^2}{a_3}\log\left(\frac{18a_1\sqrt{a_2}}{a_3}\right)\right)\log\left(\sqrt{a_2}\sqrt{\frac{2a_4}{a_3} + 4\left(\frac{a_1}{a_3}\right)^2\log\left(\frac{18a_1\sqrt{a_2}}{a_3}\right)^2}\right)$$
$$+a_4\log\left[a_2\left(\frac{2a_4}{a_3} + 4\left(\frac{a_1}{a_3}\right)^2\log\left(\frac{18a_1\sqrt{a_2}}{a_3}\right)^2\right)\right]$$

□

Using the previous lemma, we get

$$\frac{\alpha\mu_l}{2}\left(\left|S^b_{\tau-1}\right| + \left|S^+_{\tau-1}\right|\right) \leq \frac{2}{3}\log\left[\frac{8D_0}{3\delta\alpha\mu_l} + \frac{4D_0}{\delta(\alpha\mu_l)^2}\left(\left(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\right)\log\left[\frac{144d\sqrt{D_0}}{\sqrt{\delta}\alpha\mu_l}\left(\sqrt{\lambda}B+4\sigma\right)\right]\right)^2\right]$$
$$+\mu_h + \frac{2}{\sqrt{3\alpha\mu_l}}\left(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\right)\log\left[\frac{\sqrt{D_0}}{\sqrt{\delta}e}\left(\frac{2}{\sqrt{3\alpha\mu_l}} + 4\left(\frac{4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma}{\alpha\mu_l}\right)^2\times\right.\right.$$
$$\left.\left.\times\log\left(\frac{36\sqrt{D_0}}{\sqrt{\delta}\alpha\mu_l}\left(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\right)\right)^2\right)\right]$$
$$+\frac{1}{\alpha\mu_l}\left(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\right)^2\log\left(\frac{36}{\alpha\mu_l}\sqrt{\frac{D_0}{\delta}}\left(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\right)\right)\log\left[\frac{\sqrt{D_0}}{\sqrt{\alpha\mu_l\delta}}\right.$$
$$\left.+\frac{12\sqrt{D_0}}{\sqrt{\delta}}\left(\frac{4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma}{\alpha\mu_l}\right)^2\log\left(\frac{36\sqrt{D_0}}{\sqrt{\delta}\alpha\mu_l}\left(4d(\sqrt{\lambda}B+2\sigma)+\sqrt{2}\sigma\right)\right)^2\right]$$

Since neither baseline nor a non-UCB arm will be pulled anymore after τ, the final statement at n follows. □

2.B.1.2 Proof of Theorem 4

The proof follows the same regret decomposition as in Thm. 3

$$R(n) \leq \sum_{t\in S^\star_n \cup S^+_n} 2\beta_t\left|\left|x_{t,a_t}\right|\right|_{V_t^{-1}} + \Delta_h\left(\left|S^b_n\right| + \left|S^+_n\right|\right). \tag{2.134}$$

While the first term is exactly the regret of the LinUCB algorithm, we need to derive a bound on the number of times a non-UCB arm is selected similar to Lemma 12.

Lemma 14. *Let* $\tilde{T}^\alpha_\delta := \frac{\alpha\mu_l}{(1-\alpha)\mu_h+\alpha\mu_l}T$ *and :*

$$C_b(\alpha,\mu_l,\delta) := 28d^2\left(\frac{2/3+\sqrt{\lambda}B+2\sigma}{\alpha\mu_l}\right)^2\ln\left(\frac{696d^2D_0}{\delta\left(\alpha\mu_b\right)^2}\left(2/3+\sqrt{\lambda}B+2\sigma\right)^2\right)^2$$

where $D_0 := \max\{3, 2D^2/\lambda\}$. *Then the number of conservative plays for the algorithm 3 is such that :*

- *if* $T_\delta^\alpha \geq C_b(\alpha, \mu_l, \delta)$ *:*

$$\frac{\alpha\mu_l}{2}\left(|S_\tau^b| + |S_{\tau-1}^+|\right) \leq \max\left\{-\frac{\alpha\mu_l}{2}\left(T_\alpha^\delta + 1\right) + \mu_h + 4d\left(\sqrt{\lambda}B + 2\sigma + \frac{2}{3}\right)\sqrt{T_\alpha^\delta + 1} \times \log\left(\frac{D_0}{\delta}\left(T_\alpha^\delta + 1\right)\right), 0\right\}$$

- *else :*

$$\frac{\alpha\mu_l}{2}\left(|S_\tau^b| + |S_{\tau-1}^+|\right) \leq \frac{57d^2}{\alpha\mu_l}\left(2/3 + \sqrt{\lambda}B + 2\sigma\right)^2 \log\left(\frac{44d^2\sqrt{D_0}}{\sqrt{\delta}\alpha\mu_l}\left(2/3 + \sqrt{\lambda}B + 2\sigma\right)^2\right)^2$$

Proof. As previously, let's define τ as the last time the optimistic arm was not in the constraint set, and let k be such that, $\tau \in [\![kT + 1, (k+1)T]\!]$, i.e., the phase to which τ belongs and let $\tilde{a}_\tau$ be the optimistic arm at time τ. Because this arm does not satisfy the constraint we have :

$$\max\left\{\sum_{t\in S_{\tau-1}} r_{a_l}^l - \psi_L(\tau), 0\right\} + \sum_{l\in S_{\tau-1}^b} \mu_{b_l}^l + \max\left\{\min_{\theta\in\mathcal{C}_\tau}\langle\theta, x_{\tau,a_\tau}\rangle, 0\right\} + \alpha((k+1)T - \tau)\mu_l \leq (1-\alpha)\sum_{l=1}^{\tau}\mu_{b_l}^l.$$

where $S_{\tau-1} = S_{\tau-1}^\star \cup S_{\tau-1}^+$. This can be rewritten as :

$$\alpha\,\tau\,\mu_l \leq \sum_{l\in S_{\tau-1}} \mu_{b_l}^l - r_{a_l}^l + \psi_L(\tau) + \mu_h - \alpha\left((k+1)T - \tau\right)\mu_l - \max\left\{\min_{\theta\in\mathcal{C}_\tau}\langle\theta, x_{\tau,a_\tau}\rangle, 0\right\} \tag{2.135}$$

Now, $-\alpha\left((k+1)T - \tau\right)\Delta_l \leq 0$ and $-\max\left\{\min_{\theta\in\mathcal{C}_\tau}\langle\theta, x_{\tau,a_\tau}\rangle, 0\right\} \leq 0$. Thus using the same reasoning as in the regret analysis of CLUCB2, we have :

$$\begin{aligned}\frac{\alpha\mu_l}{2}\left(|S_\tau^b| + |S_{\tau-1}^+|\right) \leq &-\frac{\alpha\mu_l}{2}\left(|S_{\tau-1}^\star| + |S_{\tau-1}^+| + 1\right) + \mu_h \\ &+ 4d\left(\sqrt{\lambda}B + 2\sigma + \frac{2}{3}\right)\sqrt{|S_{\tau-1}^\star| + |S_{\tau-1}^+| + 1}\log\left(\left(1 + |S_{\tau-1}^\star| + |S_{\tau-1}^+|\right)\frac{D_0}{\delta}\right)\end{aligned} \tag{2.136}$$

where $D_0 = \max\{3, 2D^2/\lambda\}$. Let's define the function

$$f : x \mapsto -\frac{\alpha\mu_l}{2}x + \mu_h + 4d\left(\sqrt{\lambda}B + 2\sigma + \frac{2}{3}\right)\sqrt{x}\log\left(\frac{D_0 x}{\delta}\right)$$

Equation (2.136) can be rewritten as :

$$\frac{\alpha\mu_l}{2}\left(|S_{\tau-1}^b| + |S_{\tau-1}^+|\right) \leq f\left(|S_{\tau-1}^\star| + |S_{\tau-1}^+| + 1\right)$$

but function f has a maximum and computing it gives :

$$\frac{\alpha\mu_l}{2}\left(|S_\tau^b| + |S_{\tau-1}^+|\right) \leq \frac{57d^2}{\alpha\mu_l}\left(2/3 + \sqrt{\lambda}B + 2\sigma\right)^2 \log\left(\frac{44d^2\sqrt{D_0}}{\sqrt{\delta}\alpha\mu_l}\left(2/3 + \sqrt{\lambda}B + 2\sigma\right)^2\right)^2 \tag{2.137}$$

and it is attained at $x^\star$ such that :

$$\begin{aligned}x^\star &\leq 28d^2\left(\frac{2/3 + \sqrt{\lambda}B + 2\sigma}{\alpha\mu_l}\right)^2 \ln\left(\frac{696d^2 D_0}{\delta\left(\alpha\mu_b\right)^2}\left(2/3 + \sqrt{\lambda}B + 2\sigma\right)^2\right)^2 \\ &:= C_b(\alpha, \mu_l, \delta)\end{aligned}$$

Function f is increasing before $x^\star$ and decreasing afterwards. Now, equation (2.137) is the result obtained by Kazerouni et al. (2017). But, at the beginning of the first phase, i.e., when $k = 0$, we have that for $t \leq \frac{\alpha\mu_l}{(1-\alpha)\mu_h + \alpha\mu_l}T$ that :

$$\alpha\mu_l(T - t) \geq (1-\alpha)\mu_h \geq (1-\alpha)\sum_{l=1}^{t}\mu_{b_l}^l \tag{2.138}$$

Equation (2.138) implies that at the beginning of the algorithm the conservative condition is satisfied for every possible arm. Thus $\left|S^{\star}_{\tau-1}\right| + \left|S^{+}_{\tau-1}\right| \geq \frac{\alpha\mu_l}{(1-\alpha)\mu_h+\alpha\mu_l}T$. Therefore, if $\tilde{T}^{\alpha}_{\delta} := \frac{\alpha\mu_l}{(1-\alpha)\mu_h+\alpha\mu_l}T \geq C_b(\alpha,\mu_l,\delta)$, we can upper-bound $f\Big(\left|S^{\star}_{\tau-1}\right| + \left|S^{+}_{\tau-1}\right| + 1\Big)$ by the value of f evaluated at $\frac{\alpha\mu_l}{(1-\alpha)\mu_h+\alpha\mu_l}T$ that is to say we have:

$$\frac{\alpha}{2}\left(\left|S^{b}_{\tau}\right| + \left|S^{+}_{\tau-1}\right|\right)\mu_l \leq -\frac{\alpha\mu_l}{2}\left(T^{\delta}_{\alpha}+1\right) + \mu_h + 4d\left(\sqrt{\lambda}B + 2\sigma + \frac{2}{3}\right)\sqrt{T^{\delta}_{\alpha}+1}\log\left(\frac{D_0}{\delta}\left(T^{\delta}_{\alpha}+1\right)\right) \tag{2.139}$$

And, on the other hand $\alpha\left|S^{b}_{\tau}\right|\mu_l \geq 0$, we have :

$$\frac{\alpha}{2}\left(\left|S^{b}_{\tau}\right| + \left|S^{+}_{\tau-1}\right|\right)\mu_l \leq \max\left\{-\frac{\alpha\mu_l}{2}\left(T^{\delta}_{\alpha}+1\right) + \mu_h + 4d\left(\sqrt{\lambda}B + 2\sigma + \frac{2}{3}\right)\sqrt{T^{\delta}_{\alpha}+1}\times\log\left(\frac{D_0}{\delta}\left(T^{\delta}_{\alpha}+1\right)\right), 0\right\}$$

□

Combining the result of the lemma with LinUCB regret bound provides the final result

- If $\tilde{T}^{\alpha}_{\delta} \geq C_b(\alpha,\mu_l,\delta)$:

$$R(n) \leq \frac{\Delta_h}{\alpha\mu_l}\max\left\{-\frac{\alpha\mu_l}{2}\left(T^{\delta}_{\alpha}+1\right) + \mu_h + 4d\left(\sqrt{\lambda}B + 2\sigma + \frac{2}{3}\right)\sqrt{T^{\delta}_{\alpha}+1}\log\left(\frac{D_0}{\delta}\left(T^{\delta}_{\alpha}+1\right)\right), 0\right\}$$
$$+4\sqrt{nd\log\left(1+\frac{nD^2}{\lambda d}\right)}\times\left[\sqrt{\lambda}B + \sigma\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1+\frac{nD^2}{\lambda d}\right)}\right];$$

- else :

$$R(n) \leq 4\sqrt{nd\log\left(1+\frac{nD^2}{\lambda d}\right)}\left[\sqrt{\lambda}B + \sigma\sqrt{2\log\left(\frac{1}{\delta}\right) + d\log\left(1+\frac{nD^2}{\lambda d}\right)}\right]$$
$$+\frac{57\Delta_h d^2}{(\alpha\mu_l)^2}\left(2/3+\sqrt{\lambda}B+2\sigma\right)^2\times\log\left(\frac{44d^2\sqrt{D_0}}{\sqrt{\delta}\alpha\mu_l}\left(2/3+\sqrt{\lambda}B+2\sigma\right)^2\right)^2.$$

2.B.2 Experiments

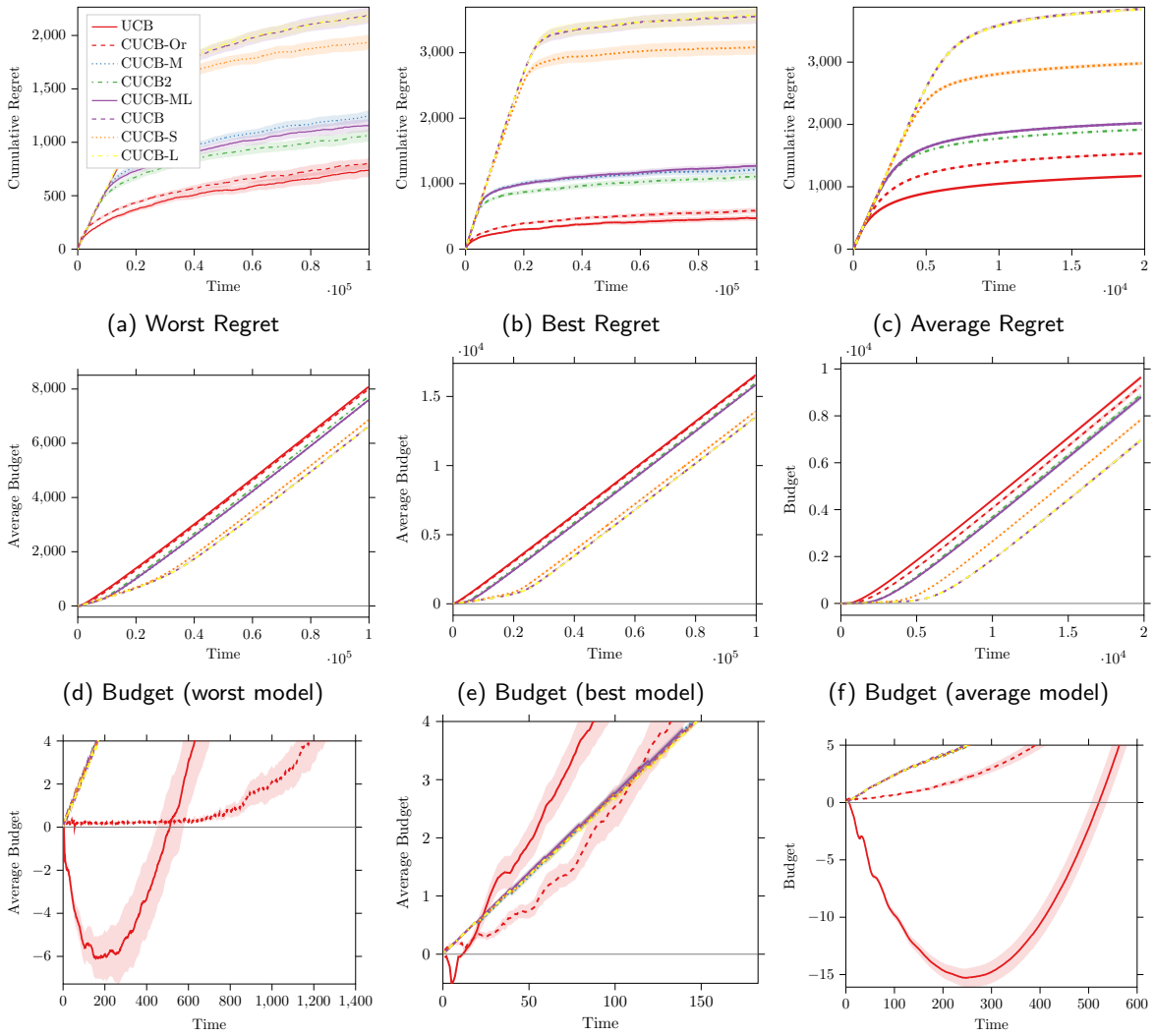
2.B.2.1 Worst Case Model for Synthetic Data

In this subsection, we present the protocol used to choose which model is used to present the results on synthetic data. To generate Figure 2.5, we have drawn n_m random bandit models on which each algorithm was ran n_s times.

In order to show the improvement of CUCB2 or CLUCB2 over their counterparts CUCB and CLUCB, we have selected the model in which the difference between the regret of CUCB2 and of CUCB at n is the smallest. More formally, if $R^{m}_{\mathrm{CUCB2}}(n)$ is the empirical regret of CUCB2 after n steps in the bandit model m averaged over n_s runs, $R^{m}_{\mathrm{CUCB}}(n)$ the same for CUCB, and $\mathcal{M}$ the set of models used for the experiments. The model, $m^{\star}$, selected for Figure 2.5 is such that

$$1 - \frac{R^{m^{\star}}_{\mathrm{CUCB2}}(n)}{R^{m^{\star}}_{\mathrm{CUCB}}(n)} = \min_{m\in\mathcal{M}} 1 - \frac{R^{m}_{\mathrm{CUCB2}}(n)}{R^{m}_{\mathrm{CUCB}}(n)} \tag{2.140}$$

As algorithm CUCB2 achieves consistently better regret than CUCB, the quantities involved in (2.140) are positive and thus selecting the minimum effectively gives the worst model in terms of improvement w.r.t. to CUCB. Empirically, the difference between the regrets was indeed positive for each model. We follow the same protocol for the linear case by comparing the performance of CLUCB2 and CLUCB.



(a) Worst Regret (b) Best Regret (c) Average Regret

(d) Budget (worst model) (e) Budget (best model) (f) Budget (average model)

(g) Budget (worst model) zoomed (h) Budget (best model) zoomed (i) Budget (average model) zoomed

Figure 2.B.1: We report the regret and the budget for the worst, best and average model in the Bernoulli experiment.

2.B.2.2 Multi-Armed Bandits

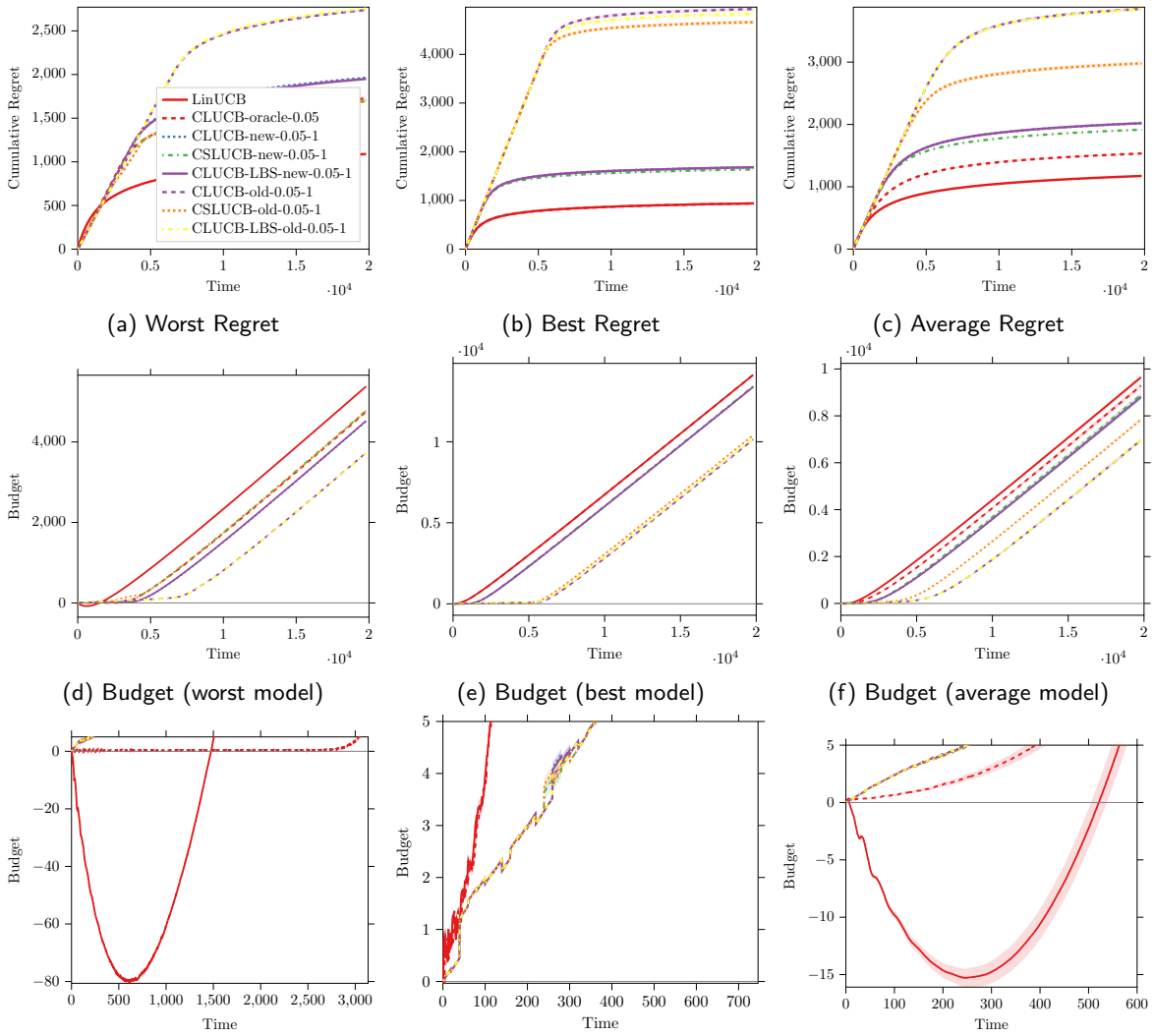
In order to give an idea about the difference of performance across different bandit problems, we report the regret for the worst model, best model and average of model (see Fig. 2.B.1). The best model is obtained by changing $\min$ to a $\max$ in Eq. 2.140. Notice that the best model and the average one are very similar, meaning that the distribution of the results is very concentrated close to the best one.

We also report the average violation of the conservative constraint. More precisely, for an algorithm, $\mathcal{A}$, which pulled arms $(a_1, \dots, a_t)$, the *exact* budget is defined as

$$B_{\mathcal{A}}(t) = \sum_{l=1}^{t} \left(\mu^l_{a_l} - (1-\alpha)\mu^l_b \right)$$

This quantity is what conservative algorithms like $\mathrm{CUCB2}$ is constrained to keep positive at every step, while $\mathrm{CUCB2T}$ is constrained to keep budget positive at certain predefined checkpoint. On the other hand UCB does not constraint the budget at all. We focus on the time steps where the budget is negative for UCB and for $\mathrm{CUCB2T}$ algorithms.[21] As it can be noticed the other conservative algorithms trade-off some level of performance (regret) in order to be safe w.r.t. the baseline.

[21] Note that the budget is negative since these algorithms are not constrained to be safe uniformly in time but only at the checkpoints.



(a) Worst Regret (b) Best Regret (c) Average Regret

(d) Budget (worst model) (e) Budget (best model) (f) Budget (average model)

(g) Budget (worst model) zoomed (h) Budget (best model) zoomed (i) Budget (average model) zoomed

Figure 2.B.2: We report the regret and the budget for the worst, best and average model in the linear bandit experiment.

Figures 2.B.1d-e-f show the budget of the worst, best and the average over models until $t = n$. Note that all the lines are parallel meaning that all the algorithms have reached nearly optimal policies. As the reader may notice, the better the algorithm the higher the budget. This is due to the fact that better algorithms tends to explore better (and/or faster) and quickly discard suboptimal arms. Figure 2.B.1 also reports the time steps where the budget is negative for UCB. In particular, as pointed out in the introduction, UCB explores heavily at the beginning which leads to potentially large violation of the constraint, see Fig. 2.B.1d. On the other hand, the conservative algorithms never violates the conservative constraint.

2.B.2.3 Synthetic Linear Data

We present the regret and budget for the best, average and worst model on different linear bandit problems. Figure 2.B.2 shows the regret and the budget for CLUCB, CLUCB2 and different ablated algorithms used in the experiments. Figure 2.B.2a shows that in the worst case, the algorithms using the action selection process introduced in this paper outperforms CLUCB-OR which can be surprising. However, the latter is an oracle with respect to algorithms using a two stage action selection process. Thus, Figure 2.B.2a shows that the introduction of a new action selection process can lead to major improvement in regards of the regret of conservative algorithms. The other possible comment is that the introduction of a martingale based concentration inequality does not lead to significant

improvement in that specific case. This indicates that in the linear setting, the impact of the choice of the lower bound is less flagrant than for multi-armed bandit because the lower bound used by CLUCB in in some way takes into account of the correlation between arms.

However, looking at Figure 2.B.2c, it is clear that in average changing the lower bound does impact positively the regret and that the worst case presented here is a corner case. Figures 2.B.2g-h-i shows the violation of the constraint by LinUCB and are similar to the multi-armed bandit case, in the sense that LinUCB explore heavily at the beginning of the problem which leads to large violation of the conservative constraint.

2.B.2.4 Jester Dataset

For the Jester Goldberg et al. (2001) experiment, we consider the standard linear setting. We performed a matrix factorization of the ratings (after filtering over users and jokes). This provides features for the arms and users, the reward (i.e., rating) is the dot product between the arm and user features (we make it stochastic by adding Gaussian noise). We consider a cold start problem where the user is randomly selected at the beginning of the repetition and the agent has to learn the best arm to recommend. When an arm is selected by the algorithm, its reward is computed as the dot product between the arm and user features.

We report the budget $B_{\mathcal{A}}(t)$ in the case of Jester dataset. We report the average over all the users and simulations. Fig. 2.B.3(*left*) shows that LinUCB and the checkpoint-based algorithms violates the *one-step* budget in the initial phase. CLUCB2T follows the exploratory behaviour of LinUCB until the conservative condition (2.24) forces them to revert to a conservative behavior by playing the baseline. If we observe the long-term behavior (Fig. 2.B.3(*right*)), all the lines are parallel, meaning that the algorithms have converged to nearly optimal policies. Second, LinUCB is the one building the higher budget since it is the first one to converge toward optimal arms. The other algorithms are ordered accordingly to their regret performance.

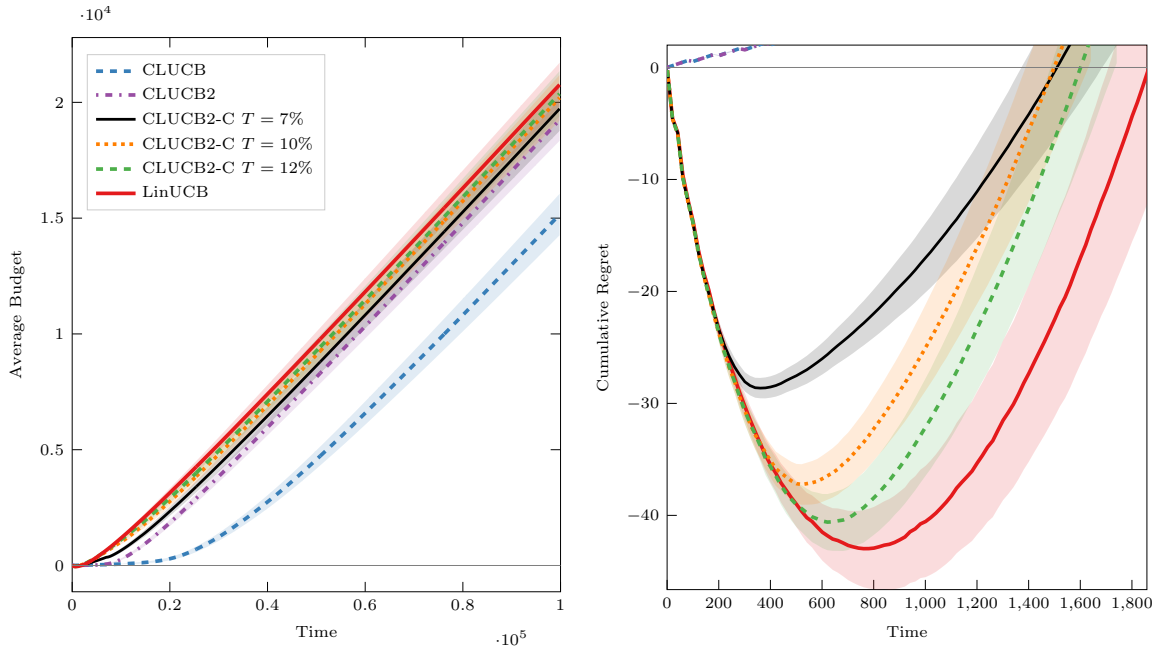


Figure 2.B.3: Budget as a function of time for the Jester dataset. The figure shows the average budget over users and repetitions.

2.C Appendix for Conservative Exploration for Reinforcement Learning

2.C.1 Policy Evaluation with Uncertainties

Consider a bounded parameter MDP $\mathcal{M}$ defined by a compact set $B_r(s,a) \subseteq [0, r_{\max}]$ and $B_p(s,a) \in \Delta_S$:

$$\mathcal{M} = \{M = (\mathcal{S}, \mathcal{A}, r, p), r(s,a) \in B_r(s,a), p(\cdot|s,a) \in B_p(s,a), \forall (s,a) \in \mathcal{S} \times \mathcal{A}\} \tag{2.141}$$

Algorithm 8: EVI.

Input: Operator $\mathcal{L} : \mathbb{R}^S \to \mathbb{R}^S$ and accuracy $\epsilon > 0$
Set $v_0 = 0$, $v_1 = \mathcal{L}v_0$, $n = 0$
while $sp(v_{n+1} - v_n) > \epsilon$ **do**
  $n = n + 1$
  $v_{n+1} = \mathcal{L}v_n$
Output: $g_n = \frac{1}{2}\big(\max_s\{v_{n+1}(s) - v_n(s)\} + \min_s\{v_{n+1}(s) - v_n(s)\}\big)$ and v_n

In this paper, we consider confidence sets B_r and B_p that are polytopes. We are interested in building a pessimistic (robust) estimate of the performance of a policy $\pi \in \Pi^{\mathsf{SD}}$ in $\mathcal{M}$. This robust optimization problem can be written as:

$$\underline{g}^{\pi} := \inf_{M \in \mathcal{M}} \{g^{\pi}(M)\} \tag{2.142}$$

where $g^{\pi}(M)$ is the gain of policy π in the MDP M. Lemma 15 shows that there exists a solution to this problem that can be computed using EVI when the set $\mathcal{M}$ contains an ergodic MDP.

We recall that any bounded parameter MDP admits an equivalent representation as an extended MDP (Jaksch et al., 2010a) with identical state space $\mathcal{S}$ but compact action space. For a deterministic policy $\pi \in \Pi^{\mathsf{SD}}$, the extended (pessimistic) Bellman operator $\mathcal{L}_\pi$ is defined as:

$$\forall v \in \mathbb{R}^S, \forall s \in \mathcal{S}, \quad \mathcal{L}_\pi v(s) := \min_{r \in B_r(s,\pi(s))} r + \min_{p \in B_p(s,\pi(s))} \{p^{\mathsf{T}} v\} \tag{2.143}$$

Lemma 15. *Let $\mathcal{M}$ be a bounded-parameter MDP defined as in Eq. 2.141 such that exists an* ergodic *MDP $M \in \mathcal{M}$ w.h.p. Consider a policy $\pi \in \Pi^{SD}$, then:*

1. *There exists a tuple $(\widetilde{g}, \widetilde{h}) \in \mathbb{R} \times \mathbb{R}^S$ such that:*

$$\forall s \in \mathcal{S}, \quad \widetilde{g} + \widetilde{h}(s) = \mathcal{L}_\pi \widetilde{h}(s)$$

   where $\mathcal{L}_\pi$ is the Bellman operator of the extended MDP $\mathcal{M}^+$ associated to $\mathcal{M}$ (see Eq. 2.143).

2. *In addition, we have the following inequalities on the pair $(\tilde{g}, \tilde{h})$:*

$$\tilde{g} \le g^{\pi}(M) \qquad \text{and} \qquad sp(\tilde{h}) \le \max_{\pi \in \Pi^{SD}(M)} \max_{s \neq s'} \mathbb{E}_M^{\pi}\left(\tau(s')|s\right) := \Upsilon < +\infty$$

   where $\mathbb{E}_M^{\pi}$ is the expectation of using policy π in the MDP M and $\tau(s')$ is the minimal number of steps to reach state s'.

Proof. **Point 1.** We show that this policy evaluation problem is equivalent to a planning problem in an extended MDP $\mathcal{M}^-$ with negative reward. Consider the extended MDP $\mathcal{M}^- = (\mathcal{S}, \mathcal{A}^-, p^-, r^-)$ such that $\mathcal{A}_s^- = \{\pi(s)\} \times B_r(s, \pi(s)) \times B_p(s, \pi(s))$. For any state $s \in \mathcal{S}$ and action $a^- = (\pi(s), r(s, \pi(s)), p(\cdot|s, \pi(s))) \in \mathcal{A}_s$,

$$r^-(s, a^-) = -r(s, \pi(s))$$
$$p^-(\cdot|s, a^-) = p(\cdot|s, \pi(s))$$

Denote by $\mathcal{L}^-$ the optimal Bellman operator of $\mathcal{M}^-$. Since $B_r(s, \pi(s))$ and $B_p(s, \pi(s))$ are polytopes, $\mathcal{L}^-$ can be interpreted as an optimal Bellman operator with finite number of actions. A sufficient condition for the existence of a solution of the optimality equations is that the MDP is weakly communicating (Puterman, 1994, Chap. 8-9). Note that $\mathcal{M}^-$ contains the model defined by P^{π}, i.e., the Markov chain induced by π in M.[22] Since P^{π} is ergodic, $\mathcal{M}^-$ is at least communicating and thus $\mathcal{L}^-$ converges to a solution of the optimality equations. Extended value iteration (Jaksch et al., 2010a) on $\mathcal{L}^-$ converges toward a gain and bias (g^-, h^-) such that:

$$\begin{aligned} g^- + h^-(s) = L^- h^-(s) &= \max_{a \in \mathcal{A}_s^-} \{r^-(s,a) + p^-(\cdot|s,a)^{\mathsf{T}} h^-\} \\ &= \max_{r \in B_r(s,\pi(s))} \{-r\} + \max_{p \in B_p(s,\pi(s))} p^{\mathsf{T}} h^- \\ &= -\min\{B_r(s, \pi(s))\} + \max_{p \in B_p(s,\pi(s))} p^{\mathsf{T}} h^- \end{aligned}$$

[22] We abuse of language since $\mathcal{M}^-$ is not formally a set. We should formally refer to the bounded parameter MDP associated to $\mathcal{M}^-$, i.e., built considering $B_p(s, \pi(s))$ and $B_r(s, \pi(s))$. Note that $p(\cdot|s, \pi(s)) \in B_p(s, \pi(s))$ w.h.p.

By rearranging, we have that:

$$-g^- + (-h^-)(s) = \min\{B_r(s,\pi(s))\} + \min_{p\in B_p(s,\pi(s))} p^\mathsf{T}(-h^-) = \mathcal{L}_\pi(-h^-)(s)$$

Thus follows that $\widetilde{g} = -g^-$ and $\widetilde{h} = -h^-$. This shows the relationship between maximizing over policies in the extended MDP $\mathcal{M}^-$ and minimizing over the set of models induced by π.

Point 2. Let's begin by bounding the span of the bias $\tilde{h}$. Thanks to Theorem 4 of Bartlett and Tewari (2009), we have that the span of $\tilde{h}$ is upper-bounded by the diameter of the extended MDP $\mathcal{M}^-$, i.e:

$$sp(\tilde{h}) \leq \max_{s\neq s'} \inf_{\pi^-\in\Pi^{SD}(\mathcal{M}^-)} \mathbb{E}_{\pi^-}(\tau(s')|s)$$

where $\mathbb{E}_{\pi^-}$ is the expectation of using policy π^- in the extended MDP $\mathcal{M}^-$ and $\tau(s')$ is the hitting time of state s'. But let's define the policy $\pi^\star$ in the extended MDP $\mathcal{M}^-$ such that for a state s, it chooses the action:

$$\pi^\star(s) = (\pi(s), r^\star(s,\pi(s)), p^\star(.|s,\pi(s)))$$

with $r^\star$ and $p^\star$ the true parameter of the MDP M, this is possible because w.h.p the MDP $M^\pi \in \mathcal{M}^-$ with M^π the Markov chain induced by using policy π in the MDP M. Thus for any pair of states (s,s'):

$$\mathbb{E}_{\pi^\star}(\tau(s')|s) = \mathbb{E}^\pi_M(\tau(s')|s)$$

with $\mathbb{E}^\pi_M$ the expectation of using policy π in the MDP M. Therefore:

$$\begin{aligned} sp(\tilde{h}) &\leq \max_{s\neq s'} \inf_{\pi^-\in\Pi^{SD}(M^-)} \mathbb{E}_{\pi^-}(\tau(s')|s) \\ &\leq \mathbb{E}^\pi_M(\tau(s')|s) \leq \Upsilon := \max_{\pi\in\Pi^{SD}(M)} \max_{s\neq s'} \mathbb{E}^\pi_M(\tau(s')|s) \end{aligned}$$

And $\Upsilon < +\infty$ because M is assumed to be ergodic.

Let's show that the gain $\tilde{g}$ is a lower bound on the gain of the policy π in the MDP M. Indeed, because the operator $\mathcal{L}^-$ converges toward solution of the optimality equations for negative rewards, we have that, see (Puterman, 1994, Th. 8.4.1):

$$g^- \geq -g^\pi(M)$$

because reversing the sign of the rewards in the MDP M changes the sign of the gain of a policy. Thus, $\tilde{g} \leq g^\pi(M)$. □

As a consequence, we can use EVI on $\mathcal{L}_\pi$ to compute a solution for problem 2.142. EVI generates a sequence of vectors (v_i) such that $v_{i+1} = \mathcal{L}_\pi v_i$ and $v_0 = 0$. If the algorithm is stopped when $sp(v_{n+1} - v_n) \leq \epsilon$ we have (Puterman, 1994, Sec. 8.3.1) that:

$$|g_n - \widetilde{g}| \leq \epsilon/2 \quad \text{and} \quad \|\mathcal{L}_\pi v_n - v_n - g_n e\|_\infty \leq \epsilon \tag{2.144}$$

where $e = (1,\ldots,1)$ and $g_n = \frac{1}{2}(\max_s\{v_{n+1}(s) - v_n(s)\} + \min_s\{v_{n+1}(s) - v_n(s)\})$. The following lemma shows how we can use the value produced by EVI to lower bound the expected sum of rewards under a policy π.

Lemma 16. *Let (g_n, v_n) the values computed by EVI using $\mathcal{L}_\pi$ and an accuracy ϵ. Then, the cumulative reward collected by policy π in M after t steps can be lower bounded by:*

$$\forall y \in \mathcal{S}, \quad \mathbb{E}_M\left[\sum_{i=1}^t r_i | s_1 = y, \pi\right] \geq t(g_n - \epsilon) - sp(v_n)$$

In addition,

$$sp(v_n) \leq \Upsilon$$

Proof. Using the inequalities in (2.144) we can write that:

$$\begin{aligned} v_n(s) + g_n \leq \mathcal{L}_\pi v_n(s) &= \min_{r \in B_r(s,\pi(s))} r + \min_{p \in B_p(s,\pi(s))} \{p^\intercal v\} + \epsilon \\ &\leq r(s,\pi(s)) + p(\cdot|s,\pi(s))^\intercal v_n + \epsilon \end{aligned}$$

since $r(s,\pi(s)) \in B_r(s,\pi(s))$ and $p(\cdot|s,\pi(s)) \in B_p(s,\pi(s))$ w.h.p. By iterating this inequality, we get that for all $t > 0$ and state s :

$$v_n(s) + t g_n \leq (t-1)\varepsilon + p^t(\cdot|s,\pi(s))^\intercal v_n + \mathbb{E}\left[\sum_{i=1}^{t} r_i\left(s_i, \pi(s_i)\right) | s_1 = s\right]$$

The statement follows by noticing that

$$sp(v_n) = \max_s v_n(s) - \min_s v_n(s) \geq \underbrace{p^t(\cdot|y,\pi(y))^\intercal v_n}_{\leq \max_s v_n(s)} - \underbrace{v_n(y)}_{\geq \min_s v_n(s)} \quad , \quad \forall y \in \mathcal{S}$$

The last statement is a direct consequence of the argument developed in subsection 4.3.1 of Jaksch et al. (2010a). This reasoning relies on the fact that the initial vector used in EVI is a zero span vector. □

2.C.2 Regret Bound for CUCRL

Lemma 17. *The regret of* CUCRL2 *can be upper-bounded for some $\beta > 0$, with probability at least $1 - \frac{2\delta}{5}$, by:*

$$R(\text{CUCRL2}, T) \leq \beta \cdot \left(R(\text{UCRL2}, T|\Lambda_T) + (g^\star - g^{\pi_b}) \sum_{k \in \Lambda_T^c} T_k + \max\{r_{\max}, sp\left(h^{\pi_b}\right)\} \sqrt{SAT \ln(T/\delta)} \right)$$

Proof. Recall that $k_t = \sup\{k > 0 : t > t_k\}$ is the episode at time t and that the regret is defined as $R(\text{CUCRL2}, T) = \sum_{t=1}^{T} \left(g^\star - r_t(s_t, a_t)\right)$.

Since the baseline policy π_b may be stochastic, as a first step we replace the observed reward by its expectation. As done in (Fruit et al., 2018b) we use Azuma's inequality that gives, with probability at least $1 - \frac{\delta}{5}$:

$$\forall T \geq 1, \quad -\sum_{t=1}^{T} r_t \leq -\sum_{t=1}^{T} \sum_{a \in \mathcal{A}} \pi_{k_t}(s_t, a) r(s_t, a) + 2 r_{\max} \sqrt{T \ln\left(\frac{5T}{\delta}\right)} \tag{2.145}$$

We denote by $\Lambda_T = \Lambda_{k_T} \cup \{k_T\} \cdot \mathbb{1}_{(Eq.\ 2.39)}$ the set of episodes where the algorithm played an UCRL policy. Note that we cannot directly consider Λ_{k_T} since the set is updated at the end of the episode and the last episode may not have ended at T. Similarly we denote by $\Lambda_T^c = \Lambda_{k_T}^c \cup \{k_T\} \cdot \mathbb{1}_{(\neg Eq.\ 2.39)}$. Then, the regret of CUCRL2 can be decomposed as follow:

$$\begin{aligned} R(\text{CUCRL2}, T) &= \sum_{t=1}^{T} \left(g^\star - \sum_{a \in \mathcal{A}} \pi_{k_t}(s_t, a) r(s_t, a) \right) + 2 r_{\max} \sqrt{T \ln\left(\frac{5T}{\delta}\right)} \\ &= 2 r_{\max} \sqrt{T \ln\left(\frac{5T}{\delta}\right)} + \underbrace{\sum_{k=1}^{k_T} \mathbb{1}_{(k \in \Lambda_T)} \sum_{t=t_k}^{t_{k+1}-1} (g^\star - r(s_t, a_t))}_{:= R(\text{UCRL2}, T|\Lambda_T)} \\ &\quad + \sum_{k=1}^{k_T} \mathbb{1}_{(k \in \Lambda_T^c)} \left((g^\star - g^{\pi_b})(t_{k+1} - t_k) + \underbrace{\sum_{t=t_k}^{t_{k+1}-1} \left(g^{\pi_b} - \sum_{a \in \mathcal{A}} \pi_b(s_t, a) r(s_t, a) \right)}_{:= \Delta_k^c} \right) \end{aligned} \tag{2.146}$$

Moreover, note that the UCRL2 policy is deterministic so we hate that $\sum_{a \in \mathcal{A}} \pi_{k_t}(s_t, a) r(s_t, a) = r(s_t, a_t)$ when $k_t \in \Lambda_T$. The second term, denoted $R(\text{UCRL2}, T|\Lambda_{k_T})$, is the regret suffered by UCRL2 over $\sum_{k \in \Lambda_{k_T}} T_k$ steps.

The only difference with the orginal analysis (Jaksch et al., 2010a) is that the confidence intervals used by UCRL are updated when using the baseline policy, however it does not affect the regret of UCRL because it only means the confidence intervals used shrinks faster for some state-action pairs. We will analyze this term in Lem. 19. To decompose Δ_k^c we can use the Bellman equations ($g^{\pi_b} e = L^{\pi_b} h^{\pi_b} - h^{\pi_b}$):

$$\begin{aligned}\sum_{k\in\Lambda_T^c} \Delta_k^c &= \sum_{k\in\Lambda_T^c} \sum_{t=t_k}^{t_{k+1}-1} \sum_a \pi_b(s_t,a) p(\cdot|s_t,a)^\intercal h^{\pi_b} - h_{\pi^b}(s_t) \\ &= \sum_{k\in\Lambda_T^c} \sum_{t=t_k}^{t_{k+1}-1} \underbrace{\sum_{a\in\mathcal{A}} \pi_b(s,a)\Big(p(\cdot|s_t,a)^\intercal h^{\pi_b}\Big) - h^{\pi_b}(s_{t+1})}_{:=\Delta_{k,t}^{c,p}} + \sum_{k\in\Lambda_T^c} \underbrace{\sum_{t=t_k}^{t_{k+1}-1} (h^{\pi_b}(s_{t+1}) - h^{\pi_b}(s_t))}_{:=\Delta_k^{c,2}}\end{aligned}$$

But, $\Delta_k^{c,2}$ can be bounded using a telescopic sum argument and the number of episodes:

$$\sum_{k\in\Lambda_T^c} \Delta_k^{c,2} = \sum_{k\in\Lambda_T^c} h^{\pi_b}(s_{t_{k+1}}) - h^{\pi_b}(s_{t_k}) \leq |\Lambda_T^c| sp\left(h^{\pi_b}\right)$$

Then it is easy to see that $(\Delta_{k,t}^{c,p})_{k,t}$ is a Martingale Difference Sequence with respect to the filtration $(\mathcal{F}_t)_{t\in\mathbb{N}}$ which is generated by all the randomness in the environment and in the algorithm up until time t: $|\Delta_{k,t}^{c,p}| \leq 2\|h^{\pi_b}\|_\infty \leq 2sp(h^{\pi_b})$ and $\mathbb{E}[\Delta_{k,t}^{c,p}|\mathcal{F}_t] = 0$. Thus with probability $1-\frac{\delta}{5}$:

$$\sum_{k\in\Lambda_{k_T}^c} \sum_{t=t_k}^{t_{k+1}-1} \Delta_{k,t}^{c,p} \leq 4sp\left(h^{\pi_b}\right)\sqrt{T\ln\left(\frac{5T}{\delta}\right)}$$

Therefore putting all the above together, we have that with probability at least $1-\frac{2\delta}{5}$:

$$\begin{aligned}R(\text{CUCRL2},T) \leq 2r_{\max}\sqrt{T\ln\left(\frac{5T}{\delta}\right)} + R(\text{UCRL2},T|\Lambda_T) + (g^\star - g^{\pi_b})\sum_{k=1}^{k_T} \mathbb{1}_{(k\in\Lambda_T^c)}(t_{k+1}-t_k) \\ + sp\left(h^{\pi_b}\right)\left(|\Lambda_T^c| + 4\sqrt{T\ln\left(\frac{5T}{\delta}\right)}\right)\end{aligned}$$

As shown in (Ouyang et al., 2017, Lem. 1), $k_T \leq \sqrt{2SAT\ln(T)}$ thus we can simply write that $|\Lambda_T^c| \leq \sqrt{2SAT\ln(T)}$. □

In the next lemma, we bound the total number of steps where CUCRL2 used the baseline policy.

Lemma 18. *For any, $\delta > 0$, the total length of episodes where the baseline policy is played by CUCRL after T steps is upper-bounded with probability $1-2\delta/5$ by:*

$$\sum_{l\in\Lambda_{k_T}^c} T_l \leq 2\sqrt{SAT\ln(T)} + \frac{16\sqrt{TL_T^\delta}}{\alpha g^{\pi_b}}\Big[(D+\Upsilon)\sqrt{SA} + r_{\max} + \sqrt{SA}sp(h^{\pi_b})\Big] + \frac{112SAL_T^\delta}{(\alpha g^{\pi_b})^2}(1+S(D+\Upsilon)^2)$$

with $L_T^\delta := \ln\left(\frac{5SAT}{\delta}\right)$ a logarithmic term in T.

Proof. Let τ be the last episode played conservatively: $\tau = \sup\{k>0 : k\in\Lambda_k^c\}$. At the beginning of episode τ the conservative condition is not verified that is to say:

$$\begin{aligned}\underbrace{\sum_{l\in\Lambda_{\tau-1}} T_l\left(g^{\pi_b} - g_l^- + \varepsilon_l\right)}_{:=\Delta_\tau^1} + sp(h^{\pi_b})\left(|\Lambda_{\tau-1}^c| + (1-\alpha)\right) + \sum_{l\in\Lambda_{\tau-1}\cup\{\tau\}} sp\left(h_l^-\right) + \\ + (T_{\tau-1}+1)\left((1-\alpha)g^{\pi_b} - g_\tau^- + \epsilon_\tau\right)\mathbb{1}_{\{(1-\alpha)g^{\pi_b} \geq g_\tau^- + \epsilon_\tau\}} \geq \alpha\sum_{l=1}^{\tau-1} T_l g^{\pi_b} \qquad (2.147)\end{aligned}$$

Let's proceeding by analysing each term on the RHS of Eq. 2.147. First, we have that $|\Lambda_{\tau-1}^c| \leq k_T \leq \sqrt{2SAT\ln(T)}$, thus:

$$sp(h^{\pi_b})\left(|\Lambda_{\tau-1}^c| + (1-\alpha)\right) \leq \left(\sqrt{2SAT\ln(T)} + 1\right) sp(h^{\pi_b}) \tag{2.148}$$

On the other hand, thanks to Lem. 16, we have:

$$\sum_{l \in \Lambda_{\tau-1} \cup \{\tau\}} sp\left(h_l^-\right) \leq (|\Lambda_{\tau-1}| + 1)\Upsilon \leq 2\sqrt{2SAT\ln(T)}\Upsilon \tag{2.149}$$

Before analysing Δ_τ^1, let's bound the contribution of episode τ:

$$(T_{\tau-1}+1)\left((1-\alpha)g^{\pi_b} - g_\tau^- - \epsilon_\tau\right)\mathbb{1}_{\{(1-\alpha)g^{\pi_b} \geq g_\tau^- + \epsilon_\tau\}} \leq (1-\alpha)g^{\pi_b}k_T \leq (1-\alpha)r_{\max}\sqrt{2SAT\ln(T)} \tag{2.150}$$

where we used the fact that for all episode k, we have $T_k \leq k$. Indeed the dynamic episode condition is such that for an episode k, $T_k \leq T_{k-1} + 1$ thus by iterating this inequality, $T_k \leq T_0 + k = k$. At this point using equations 2.147, 2.149 and 2.150 we have:

$$\Delta_\tau^1 + \left(\sqrt{2SAT\ln(T)} + 1\right) sp(h^{\pi_b}) + 2\sqrt{2SAT\ln(T)}\Upsilon + (1-\alpha)r_{\max}\sqrt{2SAT\ln(T)} \geq \alpha \sum_{l=1}^{\tau-1} T_l g^{\pi_b}$$

Let's finish by analysing Δ_τ^1. Let's define the event, $\Gamma = \left\{\exists T > 0, \exists k \geq 1, \text{ s.t } \; M \notin \mathcal{M}_k\right\}$, by definition of B_r^k and B_p^k, $\mathbb{P}(\Gamma) \leq \delta/5$, see (Lazaric et al., 2019, App. B.2) for a complete proof. We have that on the event Γ^c, for any $l \in \Lambda_{\tau-1}$, $(g_l^-, h_l^-) = \text{EVI}(\mathcal{L}_l^{\pi_l}, \varepsilon_l)$ is such that $|\underline{g}^{\pi_l} - g_l^-| \leq \epsilon_l$ (see App. 2.C.1) where $\underline{g}^{\pi_l}$ is the true gain: $\underline{g}^{\pi_l} + \underline{h}^{\pi_l} = \mathcal{L}_l^{\pi_l}\underline{h}^{\pi_l}$. Thus, since $\varepsilon_l \leq r_{\max}/\sqrt{t_l}$:

$$\begin{aligned}
\Delta_\tau^1 = \sum_{l \in \Lambda_{\tau-1}} T_l\left(g^{\pi_b} - g_l^- + \epsilon_l\right) &\leq 2\sum_{l \in \Lambda_{\tau-1}} T_l\varepsilon_l + \sum_{l \in \Lambda_{\tau-1}} T_l(g^{\pi_b} - \underline{g}^{\pi_l}) \\
&\leq 4r_{\max}\sqrt{T} + \sum_{l \in \Lambda_{\tau-1}} T_l(\widetilde{g}_l - \underline{g}^{\pi_l})
\end{aligned}$$

where $\widetilde{g}_l$ is the optimistic gain at episode l (see Lazaric et al. (2019)) thus the last inequality comes from $g^{\pi_b} \leq g^\star \leq \widetilde{g}_l$ for every episode l. We can also define the optimistic bias at episode l, $\widetilde{h}_l$, the pair $(\widetilde{g}_l, \widetilde{h}_l)$ is such that:

$$\forall s \in \mathcal{S}, \qquad \widetilde{g}_l + \widetilde{h}_l(s) := \mathcal{L}_l^+\widetilde{h}_l := \max_a \left\{\max_{r \in B_r^l(s,a)} r + \max_{p \in B_p^l(s,a))} p^\intercal \widetilde{h}_l\right\}$$

Recall that $\widetilde{\pi}_l \in \Pi^{\text{SD}}$ is the optimistic policy at episode l and when $l \in \Lambda_{\tau-1}$, $\pi_l = \widetilde{\pi}_l$. Then, by using Bellman equations:

$$\begin{aligned}
\sum_{l \in \Lambda_{\tau-1}} T_l(\widetilde{g}_l - \underline{g}^{\pi_l}) &= \sum_{l \in \Lambda_{\tau-1}} \sum_{t=t_l}^{t_{l+1}-1} (\widetilde{g}_l - \underline{g}^{\pi_l}) = \sum_{l \in \Lambda_{\tau-1}} \sum_{t=t_l}^{t_{l+1}-1} (\underbrace{\mathcal{L}_l^+\widetilde{h}_l(s_t)}_{:=\mathcal{L}_l^{+,\pi_l}\widetilde{h}_l(s_t)}, -\widetilde{h}_l(s_t) - \mathcal{L}_l^{\pi_l}\underline{h}^{\pi_l}(s_t) + \underline{h}^{\pi_l}(s_t)) \\
&= \sum_{l \in \Lambda_{\tau-1}} \sum_{t=t_l}^{t_{l+1}-1} \max_{r \in B_r^l(s_t,a_t)} r - \min_{r \in B_r^l(s_t,a_t)} r + \max_{p \in B_p^l(s_t,a_t)} p^\intercal \widetilde{h}_l - \min_{p \in B_p^l(s_t,a_t)} p^\intercal \underline{h}^{\pi_l} - \widetilde{h}_l(s_t) + \underline{h}^{\pi_l}(s_t) \\
&\leq \sum_{l \in \Lambda_{\tau-1}} \sum_{t=t_l}^{t_{l+1}-1} 2\max_{r \in B_r^l(s_t,a_t)} r + \max_{q \in B_p^l(s_t,a_t)} (q-p^\star)^\intercal \widetilde{h}_l \\
&\quad - \min_{q \in B_p^l(s_t,a_t)} (q-p^\star)^\intercal \underline{h}^{\pi_l} + p^\star(\cdot|s_t,a_t)^\intercal\left(\widetilde{h}_l - \underline{h}^{\pi_l}\right) - (\widetilde{h}_l(s_{t+1}) - \underline{h}^{\pi_l}(s_{t+1})) \\
&\quad + (\widetilde{h}_l(s_{t+1}) - \widetilde{h}_l(s_t) + \underline{h}^{\pi_l}(s_{t+1}) - \underline{h}^{\pi_l}(s_t))
\end{aligned}$$

where $p^\star$ is the transition probability of the true MDP, $M^\star$. By a simple telescopic sum argument, we have:

$$\sum_{l\in\Lambda_{\tau-1}}\sum_{t=t_l}^{t_{l+1}-1}\widetilde{h}_l(s_{t+1})-\widetilde{h}_l(s_t)+\underline{h}^{\pi_l}(s_{t+1})-\underline{h}^{\pi_l}(s_t)=|\Lambda_{\tau-1}|\big(sp\big(\widetilde{h}_l\big)+sp\big(\underline{h}^{\pi_l}\big)\big)$$

At this point we need to explicitly define the concentration inequality used to construct the confidence sets B_r^l and B_p^l. For every $(s,a)\in\mathcal{S}\times\mathcal{A}$, we define $\beta_l^k(s,a)$ such that:

$$\forall l\geq 1,\qquad B_r^l(s,a)\subset[\widetilde{r}_l(s,a)-\beta_r^l(s,a),\widetilde{r}_l(s,a)+\beta_r^l(s,a)]$$

where $\widehat{r}_l(s,a)$ is the empirical average of the reward received when visiting the state-action pairs (s,a) at the beginning of episode l. For every $(s,a)\in\mathcal{S}\times\mathcal{A}$, we define $\beta_p^l(s,a)$ as:

$$B_p^l(s,a)=\big\{p\in\Delta_S:\|p(\cdot|s,a)-\widehat{p}_l(\cdot|s,a)\|_1\leq\beta_p^l(s,a)\big\}$$

with $\widehat{p}_l$ is the empirical average of the observed transitions. Choosing those β_r^l and β_p^l is done thanks to concentration inequalities such that event Γ^c holds with high enough probability. In the following, we use:

$$\forall s,a\qquad \beta_r^l(s,a)=\sqrt{\frac{7SAL_T^\delta}{2\max\{1,N_l(s,a)\}}}\text{ and }\beta_p^l(s,a)=S\sqrt{\frac{14AL_T^\delta}{\max\{1,N_l(s,a)\}}}$$

where $L_T^\delta=\ln\big(\frac{5SAT}{\delta}\big)$. For other choices of β_r^l and β_p^l refer to (Lazaric et al., 2019). Similarly to what done in (Jaksch et al., 2010a, Sec. 4.3.1 and 4.3.2), by using Holder's inequality and recentering the bias functions, we write:

$$\begin{aligned}\sum_{l\in\Lambda_{\tau-1}}T_l(\widetilde{g}_l-\underline{g}^{\pi_l})\leq|\Lambda_{\tau-1}|\big(sp\big(\widetilde{h}_l\big)+sp\big(\underline{h}^{\pi_l}\big)\big)+\underbrace{\sum_{l\in\Lambda_{\tau-1}}\sum_{t=t_l}^{t_{l+1}-1}2\beta_r^l(s_t,a_t)+\beta_p^l(s_t,a_t)\big(sp\big(\widetilde{h}_l\big)+sp\big(\underline{h}^{\pi_l}\big)\big)}_{:=(a)}\\+\underbrace{\sum_{l\in\Lambda_{\tau-1}}\sum_{t=t_l}^{t_{l+1}-1}p^\star(\cdot|s_t,a_t)^\mathsf{T}\left(\widetilde{h}_l+\underline{h}^{\pi_l}\right)-(\widetilde{h}_l(s_{t+1})-\underline{h}^{\pi_l}(s_{t+1}))}_{:=(b)}\end{aligned}$$

To finish, the proof of this lemma, we need to bound the term (a) and (b). In the following, we use the fact that $sp\big(\widetilde{h}_l\big)+sp\big(\underline{h}^{\pi_l}\big)\leq D+\Upsilon$ (see Lem. 16) and again that $|\Lambda_{\tau-1}|\leq k_T\leq\sqrt{2SAT\ln(T)}$. Let's begin with (a), by definition of the radius of the confidence sets, we have:

$$\sum_{l\in\Lambda_{\tau-1}}\sum_{t=t_l}^{t_{l+1}-1}\beta_r^l(s_t,a_t)=\sqrt{\frac{7SAL_T^\delta}{2}}\sum_{l\in\Lambda_{\tau-1}}\sum_{t=t_l}^{t_{l+1}-1}\sqrt{\frac{1}{\max\{1,N_l(s_t,a_t)\}}}\leq\sqrt{\frac{7SAL_T^\delta}{2}}\sqrt{\sum_{l=1}^{\tau-1}T_l}$$

and,

$$\sum_{l\in\Lambda_{\tau-1}}\sum_{t=t_l}^{t_{l+1}-1}\beta_p^l(s_t,a_t)\leq S\sqrt{14L_T^\delta A}\sqrt{\sum_{l=1}^{\tau-1}T_l}$$

The second term (b) is easy to bound because it is a Martingale Difference Sequence with respect to the filtration generated by all the randomness in the algorithm and the environment before the current step. For any time t, the σ-algebra generated by the history up to time t included is $\mathcal{F}_t=\sigma(s_1,a_1,r_1,\dots,s_t,a_t,r_t,s_{t+1})$. Define $X_t=\mathbb{1}_{(k_t\in\Lambda_T)}(p(\cdot|s_t,\pi_{k_t}(s_t))^\mathsf{T}u_{k_t}-u_{k_t}(s_{t+1}))$ with $u_{k_t}=\widetilde{h}_{k_t}-\underline{h}^{\pi_{k_t}}$. Since π_{k_t} is $\mathcal{F}_t$ measurable, $E[X_t|\mathcal{F}_{t-1}]=0$ and $|X_t|\leq 2(D+\Upsilon)$. Then $(X_t,\mathcal{F}_t)_t$ is an MDS and nothing change compared to the analysis of UCRL2. Therefore using Azuma-Hoeffding inequality, we have, with probability $1-\frac{\delta}{5}$ that:

$$(b)\leq 2(D+\Upsilon)\sqrt{2TL_T^\delta}$$

⚠ Algorithmically, it is possible to evaluate the gain of the policies played in the past episodes at the beginning of the current episode. While this will provide a better estimate for the conservative condition, it will break the MDS structure in *(b)* since $\underline{h}^{\pi_l}$ will be not measurable w.r.t. $\mathcal{F}_l$ since it is computed with samples collected after episode l. Thus putting the bound for (a) and (b) together, we have:

$$\sum_{l\in\Lambda_{\tau-1}} T_l(\widetilde{g}_l - \underline{g}^{\pi_l}) \leq (D+\Upsilon)\sqrt{2SAT\ln(T)} + \sqrt{14SAL_T^\delta}\sqrt{\sum_{l=1}^{\tau-1} T_l} + (D+\Upsilon)S\sqrt{14L_T^\delta A}\sqrt{\sum_{l=1}^{\tau-1} T_l} + 2(D+\Upsilon)\sqrt{2TL_T^\delta}$$

That is to say,

$$\boxed{\begin{aligned} &4r_{\max}\sqrt{T} + (D+\Upsilon)\sqrt{2SAT\ln(T)} + 2(D+\Upsilon)\sqrt{2TL_T^\delta} \\ &+ \left(\sqrt{2SAT\ln(T)}+1\right)sp(h^{\pi_b}) + \sqrt{2SAT\ln(T)}\Upsilon + (1-\alpha)r_{\max}\sqrt{2SAT\ln(T)} \end{aligned}} := b_T$$
$$+ \sqrt{14SAL_T^\delta}\sqrt{\sum_{l=1}^{\tau-1} T_l} + (D+\Upsilon)S\sqrt{14L_T^\delta A}\sqrt{\sum_{l=1}^{\tau-1} T_l} \geq \alpha\sum_{l=1}^{\tau-1} T_l g^{\pi_b}$$

Rearranging the terms and calling $X = \sum_{l=1}^{\tau-1} T_l$, we have:

$$\alpha g^{\pi_b} X \leq b_T + \left(\sqrt{14SAL_T^\delta} + (D+\Upsilon)S\sqrt{14L_T^\delta A}\right)\sqrt{X}$$

We have a quadratic equation and thus:

$$\sum_{l=1}^{\tau-1} T_l \leq \frac{2b_T}{\alpha g^{\pi_b}} + \frac{56SAL_T^\delta}{(\alpha g^{\pi_b})^2}(2 + 2S(D+\Upsilon)^2)$$

Therefore, as τ is the last episode where CUCRL2 played the policy π_b, we have $\sum_{l\in\Lambda_T^c} T_l = \sum_{l\in\Lambda_\tau^c} T_l$. Also, because of the condition on the length of an episode $T_k \leq k$ for every k, therefore:

$$\sum_{l\in\Lambda_T^c} T_l = \sum_{l\in\Lambda_\tau^c} T_l \leq \sum_{l=1}^{\tau-1} T_l + T_\tau \leq k_T + \frac{2b_T}{\alpha g^{\pi_b}} + \frac{56SAL_T^\delta}{(\alpha g^{\pi_b})^2}(2 + 2S(D+\Upsilon)^2)$$

□

The following lemma states the regret of the UCRL2 algorithm conditioned on running only the episodes in the set Λ_T.

Lemma 19. *For any $\delta > 0$, we have that after T, the regret of* UCRL2 *is upper bounded with probability at least $1 - \delta/5$ by:*

$$R(\mathrm{UCRL2}, T|\Lambda_T) \leq \beta DS\sqrt{AT\ln\left(\frac{5T}{\delta}\right)} + \beta DS^2 A\ln\left(\frac{5T}{\delta}\right)$$

with β a numerical constant.

Proof. The same type of bound has been shown in numerous work before Jaksch et al. (2010a); Lazaric et al. (2019), however the proof presented in those works can not be readily applied to our setting. Indeed, when the algorithm chooses to play the baseline policy for an episode, then the confidence sets used in CUCRL2 are updated for the state-action pairs encountered during this episode. However, in the classic proof for the UCRL2 algorithm the confidence sets are the same between the end of one episode and the beginning of the next one are the same. This may not be the case for CUCRL2.

Fortunately, when using the baseline policy during an episode, the confidence sets for every state-action pairs are either the same as the previous episode or are becoming tighter around the true parameters of the MDP $M^\star$. Thus,

proving Lemma 19 is similar to the proof presented in Lazaric et al. (2019), the only difference resides in bounding the sum, $\sum_{k\in\Lambda_{k_T}}\sum_{t=t_k}^{t_{k+1}-1} 1/\sqrt{N_k^+(s_t,a_t)}$, which is bounded by the square root of the total number of samples in the proof of Lazaric et al. (2019) whereas in the case $\mathrm{CUCRL2}$ it is bounded by the square root of the total number of samples gathered while exploring the set of policies plus the number of samples collected while playing the baseline policies. Therefore, at the end of the day both quantities are bounded by a constant times the square root of T.

A doubt someone could have is on controlling the term

$$\begin{aligned}
&\sum_{k=1}^{k_T}\mathbb{1}_{(k\in\Lambda_T)}\sum_{t=t_k}^{t_{k+1}-1}\left(p(\cdot|s_t,\pi_k(s_t))^\mathsf{T}u_k-u_k(s_t)\right)\\
&=\sum_{k=1}^{k_T}\mathbb{1}_{(k\in\Lambda_T)}\sum_{t=t_k}^{t_{k+1}-1}\underbrace{\left(p(\cdot|s_t,\pi_k(s_t))^\mathsf{T}u_k-u_k(s_{t+1})\right)}_{\Delta_k^p}+\sum_{k=1}^{k_T}\mathbb{1}_{(k\in\Lambda_T)}\sum_{t=t_k}^{t_{k+1}-1}u_k(s_{t+1})-u_k(s_t)\\
&=\sum_{k=1}^{k_T}\mathbb{1}_{(k\in\Lambda_T)}\Delta_k^p+\underbrace{\left(u_k(s_{t_{k+1}})-u_k(s_{t_k})\right)}_{\leq sp(w_k)\leq D}
\end{aligned}$$

For any time t, the σ-algebra generated by the history up to time t included is $\mathcal{F}_t=\sigma(s_1,a_1,r_1,\dots,s_t,a_t,r_t,s_{t+1})$. Define $X_t=\mathbb{1}_{(k_t\in\Lambda_T)}(p(\cdot|s_t,\pi_{k_t}(s_t))^\mathsf{T}u_k-u_k(s_{t+1}))$. Since π_{k_t} is $\mathcal{F}_t$ measurable, $E[X_t|\mathcal{F}_{t-1}]=0$ and $|X_t|\leq 2D$. Then $(X_t,\mathcal{F}_t)_t$ is an MDS and nothing change compared to the analysis of $\mathrm{UCRL2}$.

□

Finally, plugging Lemmas 18 and 19 into Lem. 17, we have that there exists a numerical constant C_1 such that with probability $1-\delta$:

$$\begin{aligned}
R(\mathrm{CUCRL2},T)\leq C_1\Bigg(DS\sqrt{ATL_T^\delta}+(g^\star-g^{\pi_b})\Bigg(\sqrt{SAT\ln(T)}+\frac{\sqrt{TSAL_T^\delta}}{\alpha g^{\pi_b}}\max\{sp(h^{\pi_b}),D+\Upsilon\}\\
+\frac{S^2AL_T^\delta}{(\alpha g^{\pi_b})^2}(D+\Upsilon)^2\Bigg)+\max\{r_{\max},sp\left(h^{\pi_b}\right)\}\sqrt{SAT\ln(T/\delta)}\Bigg)
\end{aligned}$$

2.C.3 Conservative Exploration in Finite Horizon Markov Decision Processes

In this subsection, we show how the conservative setting can be applied to finite horizon MDPs. Let's consider a finite-horizon MDP (Puterman, 1994, Chp. 4) $M=(\mathcal{S},\mathcal{A},p,r,H)$ with state space $\mathcal{S}$ and action space $\mathcal{A}$ as described in Section 1.2.3.1. Every state-action pair is characterized by a reward distribution with mean $r(s,a)$ and support in $[0,1]$ and a transition distribution $p(\cdot|s,a)$ over next state. We denote by $S=|\mathcal{S}|$ and $A=|\mathcal{A}|$ the number of states and actions, and by H the horizon of an episode.

In the following we assume that the learning agent known $\mathcal{S}$, $\mathcal{A}$ and r_{max}, while the reward and dynamics are *unknown* and need to be estimated online. Given a finite number of episode K, we evaluate the performance of a learning algorithm $\mathfrak{A}$ by its cumulative regret

$$R(\mathfrak{A},K)=\sum_{k=1}^{K}V_1^\star(s_{1,,k})-V_1^{\pi_k}(s_{1,,k})$$

where π_k is the policy executed by the algorithm at episode k.

Conservative Condition Designing a conservative condition, in this setting is much easier than in the average reward case as evaluating a policy can be done through the value function which gives an estimation of the expected reward over an episode. Thus, we can use this evaluation of a policy to use in place of rewards in the bandits condition. Formally, denote by $\pi_b\in\Pi^{\mathrm{MR}}$ the baseline policy and assume that $V_t^{\pi_b}$ is known. In general, this assumption is not restrictive since the baseline performance can be estimated from historical data. Given a conservative level $\alpha\in(0,1)$, we define the conservative condition as:

$$\forall k\in[K],\quad\sum_{l=1}^{k}V_1^{\pi_l}(s_{l,1})\geq(1-\alpha)\sum_{l=1}^{k}V_1^{\pi_b}(s_{l,1})\qquad\text{w.h.p}\tag{2.151}$$

where π_l is the policy executed by the algorithm at episode l and $s_{l,1}$ is the starting state of episode l before policy π_l is chosen. The initial state can be chosen arbitrarily but should be revealed at the beginning of each episode. Note that this condition is random due the choice of the policies $(\pi_l)_l$ and also because of the starting states thus the condition is required to hold with high probability.

Note that Eq. 2.151 requires to evaluated the performance of policy π_l on the true (unknown) MDP. In order derive a practical condition, we need to construct an estimate of $V_1^{\pi_l}$. In order to be conservative, we are interesting in deriving a lower bound on the value function of a generic policy π which can be used in Eq. 2.151.

Pessimistic value function estimate. We recall that OFU algorithms (e.g., UCB-VI and EULER) builds uncertainties around the rewards and dynamics that are used to perform an optimistic planning. Formally, denote by $\widehat{p}_k(\cdot|s,a)$ and $\widehat{r}_k(s,a)$ the empirical transitions and rewards at episode k. Then, with high probability

$$|(p(\cdot|s,a) - \widehat{p}_k(\cdot|s,a))^\mathsf{T} v| \leq \beta_k^p(s,a) \quad \text{and} \quad |r(s,a) - \widehat{r}_k(s,a)| \leq \beta_k^r(s,a)$$

for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ and $v \in [0,H]^S$. This uncertainties are used to compute an exploration bonus $b_k(s,a) = \beta_k^v(s,a) + \beta_k^r(s,a)$ that can be used to compute an optimistic estimate of the optimal value function. Formally, at episode k, optimistic backward induction (e.g., Azar et al., 2017a, Alg. 2) computes an estimate value function $\bar{v}_{k,h}$ such that $\bar{v}_{k,h} \geq V_t^\star$ for any state s. The same approach can be used to compute a pessimistic estimate of the optimal value function by subtracting the exploration bonus to the reward (e.g., Zanette and Brunskill, 2019).

The only difference in the conservative setting is that we are interesting to compute a pessimistic estimate for a policy different from the optimal one. We thus define the *pessimistic evaluation equations* for any episode k, step h, state s and policy $\pi \in MR$ as:

$$\underline{v}_{k,h}^\pi(s) := \underline{L}_{k,h}^\pi \underline{v}_{k,h+1}^\pi = \sum_a \pi_{k,h}(s,a) \left(\widehat{r}_k(s,a) - b_k(s,a) + \widehat{p}_k(\cdot|s,a)^\mathsf{T} \underline{v}_{k,h+1}^\pi\right) \tag{2.152}$$

with $\underline{v}_{k,H+1}^\pi(s) = 0$ for all states $s \in \mathcal{S}$. This value function is pessimistic (see Lem. 20) and can be computed using backward induction with $\underline{L}_k^\pi$.

Lemma 20. *Let $\pi = (d_1, \dots, d_H) \in MR$ and $(\underline{v}_{k,h}^\pi)_{h\in[H]}$ be the value function given by backward induction using Eq. 2.152 then with high probability:*

$$\forall (h,s) \in [H] \times \mathcal{S}, \qquad V_h^\pi(s) \geq \underline{v}_{k,h}^\pi(s)$$

Proof. On the event that the concentration inequalities holds, let $\widehat{r}_k(s,a)$ be the empirical reward at episode k and $\widehat{p}_k(.|s,a)$ the empirical distribution over the next state from (s,a) at episode k. We proceed with a backward induction. At time H the statement is true. For $h < H$:

$$\begin{aligned}
\underline{v}_{k,h}^\pi(s) - V_h^\pi(s) &= \sum_a d_h(s,a) \left(\widehat{r}_k(s,a) - b_k(s,a) + \widehat{p}_k(\cdot|s,a)^\mathsf{T} \underline{v}_{k,h+1}^\pi\right) - L_h^\pi V_{h+1}^\pi(s) \\
&= \sum_a d_h(s,a) \left(\underbrace{\widehat{r}_k(s,a) - r(s,a) - \beta_k^r(s,a)}_{\leq 0} \right) \\
&\quad + \sum_a d_h(s,a) \left(\widehat{p}_k(\cdot|s,a)^\mathsf{T} \underline{v}_{k,h+1}^\pi - p(\cdot|s,a)^\mathsf{T} V_{h+1}^\pi - \beta_k^p(s,a)\right) \\
&\leq \sum_a d_h(s,a) \left(\widehat{p}_k(\cdot|s,a)^\mathsf{T} \underline{v}_{k,h+1}^\pi - p(\cdot|s,a)^\mathsf{T} V_{h+1}^\pi - \beta_k^p(s,a)\right) \\
&\leq \sum_a d_h(s,a) \left((\widehat{p}_k(\cdot|s,a) - p(\cdot|s,a))^\mathsf{T} V_{h+1}^\pi - \beta_k^p(s,a)\right) \leq 0
\end{aligned}$$

where the first inequality is true because of the confidence intervals on the reward function and the penultimate inequality is true because of the backward induction hypothesis. □

Thanks to this result, we can formulate a condition that the algorithm can check, at the beginning of episode k to decide if a policy is safe to play or not :

$$\sum_{l \in \mathcal{S}_{k-1} \cup \{k\}} \underline{v}_{l,1}^{\pi_l}(s_{l,1}) + \sum_{l \in \mathcal{S}_{k-1}^c} V_{l,1}^{\pi_b}(s_{l,1}) \geq (1-\alpha) \sum_{l=1}^k V_{l,1}^{\pi_b}(s_{l,1}) \tag{2.153}$$

Algorithm 9: CUCB-VI algorithm.

Input: Policy π_b, $\delta \in (0,1)$, $r_{\max}$, $\mathcal{S}$, $\mathcal{A}$, $\alpha' \in (0,1)$, H
Set $\mathcal{H} = \emptyset$, $\mathcal{S}_0 = \emptyset$ and $\mathcal{S}_0^c = \emptyset$
for *episodes* $k = 1, 2, \ldots$ **do**
  Compute optimistic policy π_k using any OFU algorithm on history $\mathcal{H}$.
  Compute pessimistic estimate $\underline{v}_k^{\pi_k}$ as in Eq. 2.152.
  if *Equation (2.153) not verified:* **then**
    $\pi_k = \pi_b$, $\mathcal{S}_{k+1}^c = \mathcal{S}_k^c \cup \{k\}$ and $\mathcal{S}_{k+1} = \mathcal{S}_k$
  else
    $\mathcal{S}_{k+1} = \mathcal{S}_k \cup \{k\}$ and $\mathcal{S}_{k+1}^c = \mathcal{S}_k^c$
for $h = 1, \ldots, H$ **do**
  Execute $a_{k,h} = \pi_k(s_{k,h})$, obtain reward $r_{k,h}$, and observe $s_{k,h}$.
  if $\pi_k \neq \pi_b$: **then**
    Add $(s_{k,h}, a_{k,h}, r_{k,h}, s_{k,h+1})$ to $\mathcal{H}$

where $\mathcal{S}_{k-1}$ is the set of episodes where the algorithm previously played non-conservatively, $\mathcal{S}_{k-1}^c = [k-1] \setminus \mathcal{S}_{k-1}$ is the set of episodes played conservatively and $(\pi_l)_l$ is the policies that the OFU algorithm (e.g., UCB-VI) would execute without the conservative constraint.

Alg. 9 shows the generic structure of any conservative exploration algorithm for MDPs. First, it computes an optimistic policy by leveraging on an OFU algorithm and the collected history. Then it checks the conservative condition. When Eq. 2.153 is verified it plays the optimistic policy otherwise it plays conservatively by executing policy π_b. This allows to build some budget for playing exploratory actions in the future.

Regret Guarantees We analyse Alg. 9 with UCB-VI. Before to introduce the upper-bound to the regret of CUCB-VI we introduce the following assumption on the baseline policy.

Assumption 10. *The baseline policy* $\pi_b \in \Pi^{MR}$ *is such that* $r_b := \min_s\{V_1^{\pi_b}(s)\} > 0$.

We can now state the main results:

Proposition 5. *For* $\delta > 0$, *the regret of conservative* UCB-VI *(*CUCB-VI*) is upper-bounded with probability at least* $1-\delta$ *by:*

$$R(\textit{CUCB-VI}, K) \leq R(\textit{UCB-VI}, K) + \frac{1}{4\alpha r_b(\underline{\Delta}_b + \alpha r_b)}\left(16H^3 L_K + \left(200H^5S^2A + 128H^5SA\right)L_K^2\right) \tag{2.154}$$

where $L_K = \max\{\ln(3KHSA/\delta), 1\}$ *and* $\underline{\Delta}_b = \min_{s\in\mathcal{S}}\{V_1^\star(s) - V_1^{\pi_b}(s)\}$.

Proof. Let's define the high probability event, $\mathcal{E}$, that is such that in this event, all the concentration inequalities holds and the Martingale Difference Sequence concentration inequalities also holds :

$$\mathcal{E}_{1,\delta} := \bigcap_{(s,a)\in\mathcal{S}\times\mathcal{A}} \bigcap_{k\in[K]} \left\{ ||p(.|s,a) - \hat{p}_k(.|s,a)||_1 \leq \sqrt{\frac{2S\ln(3KSA/\delta)}{\max\{1, N_k(s,a)\}}} \right\}$$
$$\bigcap \left\{ |\hat{r}_k(s,a) - r(s,a)| \leq 2r_{\max}\sqrt{\frac{\ln(3KSA/\delta)}{\max\{1, N_k(s,a)\}}} \right\}$$

$$\mathcal{E}_{2,\delta} := \bigcap_{k\in[K]} \left\{ \sum_{l\in S_k}\sum_{h=1}^{H} \varepsilon_{k,h} \leq H^{3/2}\sqrt{2\#S_k \ln(3KH/\delta)} \right\}$$

and finally, $\mathcal{E} := \mathcal{E}_{1,\delta} \cap \mathcal{E}_{2,\delta}$, then $\mathcal{E}$ holds with probability at least $1-\delta$. Indeed,

$$\mathbb{P}(\mathcal{E}^c) \leq \sum_{t=1}^{HK} \frac{\delta}{3HK} + \sum_{s,a}\sum_{k} \frac{2\delta}{3KSA} \leq \delta$$

Under this event, we have that for all episode $k \in S_K$:

$$\overline{v}_{k,1}(s_{1,,k}) - \underline{v}^{\pi_k}_{k,1}(s_{1,,k}) \leq \sum_{h=1}^{H} \varepsilon_{k,h} + 5\beta^p_k(s_{k,h}, d^k_h(s_{k,h})) + 2\beta^r_k(s_{k,h}, d^k_h(s_{k,h})),$$

where $(\varepsilon_{k,h})_{k\in\mathcal{S}_K, h\in[H]}$ is a martingale difference sequence with respect to the filtration $(\mathcal{F}_{k,h})_{k\in\mathcal{S}_K,h\in[H]}$ that is generated by all the randomness before step h of episode k. Indeed, for an episode k, let $\pi_k = (d^k_1, \dots, d^k_H)$, decomposing π_k into successive decision rules.

$$\overline{v}_{k,1}(s_{1,,k}) - \underline{v}^{\pi_k}_{k,2} \leq 2\beta^r_k(s_{1,,k}, d^k_1(s_{1,,k})) + \hat{p}_k(. \mid s_{1,,k}, d^k_1(s_{1,,k}))^\intercal(\overline{v}_{k,2} - \underline{v}^{\pi_k}_{k,2}) + 2\beta^p_k(s_{1,,k}, d^k_1(s_{1,,k}))$$

Thus by defining, $B_{k,h} := 3\beta^p_k(s_{k,h}, d^k_h(s_{k,h})) + 2\beta^r_k(s_{k,h}, d^k_h(s_{k,h}))$, we have :

$$\begin{aligned}
\overline{v}_{k,1}(s_{1,,k}) - \underline{v}^{\pi_k}_{k,1}(s_{1,,k}) \leq & B_{k,1} + (\hat{p}_k(. \mid s_{1,,k}, d^k_1(s_{1,,k}) - p(.|s_{1,,k}, d^k_1(s_{1,,k}))^\intercal(\overline{v}_{k,2} - \underline{v}^{\pi_k}_{k,2}) + (\overline{v}_{k,2}(s_{k,2}) - \underline{v}^{\pi_k}_{k,2}(s_{k,2})) \\
& - (\overline{v}_{k,2}(s_{k,2}) - \underline{v}^{\pi_k}_{k,2}(s_{k,2})) + p(.|s_{1,,k}, d^k_1(s_{1,,k})^\intercal(\overline{v}_{k,2} - \underline{v}^{\pi_k}_{k,2}) \\
& \leq p(.|s_{1,,k}, d^k_1(s_{1,,k})^\intercal(\overline{v}_{k,2} - \underline{v}^{\pi_k}_{k,h}) - (\overline{v}_{k,2}(s_{k,2}) - \underline{v}^{\pi_k}_{k,2}(s_{k,2})) + 2\beta^p_k(s_{1,,k}, d^k_1(s_{1,,k})) + B_{k,1} \\
& + (\overline{v}_{k,2}(s_{k,2}) - \underline{v}^{\pi_k}_{k,2}(s_{k,2}))
\end{aligned}$$

But let's define $\varepsilon_{k,h} := p(.|s_{k,h}, d^k_h(s_{k,h}))^\intercal(\overline{v}_{k,h} - \underline{v}^{\pi_k}_{k,h}) - (\overline{v}_{k,h}(s_{k,h+1}) - \underline{v}^{\pi_k}_{k,h}(s_{k,h+1}))$ then $(\varepsilon_{k,h})_{k\in[K],h\in[H]}$ is a Martingale Difference Sequence with respect to the filtration $\mathcal{F}_{k,h}$ which is generated by all the randomness in the environment and the algorithm before step h of episode k . Then, by recursion, we have :

$$\overline{v}_{k,1}(s_{1,,k}) - \underline{v}^{\pi_k}_{k,1}(s_{1,,k}) \leq \sum_{h=1}^{H} B_{k,h} + \varepsilon_{k,h} + 2\beta^p_k(s_{k,h}, d^k_h(s_{k,h}))$$

The regret of algorithm CUCB-VI can be decomposed as :

$$\begin{aligned}
R(\text{CUCB-VI}, K) &= \sum_{k\in\mathcal{S}^c_K} V^\star_1(s_{1,,k}) - V^{\pi_b}_1(s_{1,,k}) + \sum_{k\in\mathcal{S}_K} V^\star_1(s_{1,,k}) - V^{\pi_k}_1(s_{1,,k}) \\
&\leq |\mathcal{S}^c_K|\overline{\Delta}_b + R(\text{UCB-VI}, |\mathcal{S}_K|)
\end{aligned}$$

where $\overline{\Delta}_b = \max_{s\in\mathcal{S}} V^\star(s) - V^{\pi_b}(s)$. Therefore bounding the regret amounts to bound the number of episode played conservatively. To do so, let's consider, τ the last episode played conservatively, then before the beginning of episode τ, the condition 2.153 is not verified and thus :

$$\alpha \sum_{k=1}^{\tau} V^{\pi_b}_1(s_{1,,k}) \leq \sum_{k\in\mathcal{S}_{\tau-1}\cup\{\tau\}} \underbrace{V^{\pi_b}_1(s_{1,,k}) - \underline{v}^{\pi_k}_{k,1}(s_{1,,k})}_{=\Delta_{k,1}}$$

Thus, let's finish this analysis by bounding $\Delta_{k,1} = V^{\pi_b}_1(s_{1,,k}) - \underline{v}^{\pi_k}_{k,1}(s_{1,,k})$ for all $k \in \mathcal{S}_K$. But:

$$\Delta_{k,1} = V^{\pi_b}_1(s_{1,,k}) - V^\star_1(s_{1,,k}) + V^\star_1(s_{1,,k}) - \underline{v}^{\pi_k}_{k,1}(s_{1,,k}) \leq -\underline{\Delta}_b + \overline{v}_{k,1}(s_{1,,k}) - \underline{v}^{\pi_k}_{k,1}(s_{1,,k}),$$

where $\underline{\Delta}_b := \min_s V^\star_1(s) - V^{\pi_b}_1(s)$. Now, we need to bound the sum over all the non-conservative episodes of the difference between the optimistic and pessimistic value function. That is to say :

$$\begin{aligned}
\sum_{l\in S_{\tau-1}} \sum_{h=1}^{H} \beta^r_k(s_{k,h}, d^k_h(s_{k,h})) &= \sum_{l\in\mathcal{S}_{\tau-1}} \sum_{h=1}^{H} 2Hr_{\max}\sqrt{\frac{2\ln(3KSA/\delta)}{\max\{1, N_k(s_{k,h}, d^k_h(s_{k,h}))}} \\
&\leq 2r_{\max}H^2\sqrt{2SAH|\mathcal{S}_{\tau-1}|(1+\ln(|\mathcal{S}_{\tau-1}|H))\ln(3KSA/\delta)}
\end{aligned}$$

Also :

$$\begin{aligned}
\sum_{l\in S_{\tau-1}} \sum_{h=1}^{H} \beta^p_k(s_{k,h}, d^k_h(s_{k,h})) &= \sum_{l\in S_{\tau-1}} \sum_{h=1}^{H} H\sqrt{\frac{2S\ln(3KSA/\delta)}{\max\{1, N_k(s_{k,h}, d^k_h(s_{k,h}))}} \\
&\leq H^2S\sqrt{2AH\#S_{\tau-1}(1+\ln(\#S_{\tau-1}H))\ln(3KSA/\delta)}
\end{aligned}$$

and, under the event $\mathcal{E}$, $\sum_{l\in\mathcal{S}_{\tau-1}}\sum_{h=1}^{H}\varepsilon_{k,h}\leq 2H^{3/2}\sqrt{2|\mathcal{S}_{\tau-1}|\ln(3KH/\delta)}$. On the other hand, for the episode τ, we can only bound the difference in value function by H. Finally, we have that $\tau=1+|\mathcal{S}^c_{\tau-1}|+|\mathcal{S}_{\tau-1}|$ and thus if we assume that $r_b:=\min_s V^{\pi_b}(s)>0$:

$$\begin{aligned}\alpha r_b(|\mathcal{S}^c_{\tau-1}|+1)\leq\alpha\sum_{k=1}^{\tau}V_1^{\pi_b}(s_{1,,k})\leq-(\underline{\Delta}_b+\alpha r_b)|\mathcal{S}_{\tau-1}|+2H^{3/2}\sqrt{2|\mathcal{S}_{\tau-1}|\ln(3KH/\delta)}\\ +5H^2S\sqrt{2AH|\mathcal{S}_{\tau-1}|(1+\mathcal{S}_{\tau-1}|H))\ln(3KSA/\delta)}\\ +4r_{\max}H^2\sqrt{2SAH|\mathcal{S}_{\tau-1}|(1+\ln(|\mathcal{S}_{\tau-1}|H))\ln(3KSA/\delta)}\end{aligned}$$

Thus, the function on the RHS in bounded and using lemma 8 of Kazerouni et al. (2017), we have :

$$\begin{aligned}\alpha r_b(|\mathcal{S}^c_{\tau-1}|+1)\leq\frac{1}{4(\underline{\Delta}_b+\alpha r_b)}\Bigg(16H^3\ln\left(\frac{3KH}{\delta}\right)+\left(200H^5S^2A+128r_{\max}^2H^5SA\right)\times\\ \times(1+\ln(HK))\ln\left(\frac{3KSA}{\delta}\right)\Bigg)\end{aligned}$$

But by definition, $|\mathcal{S}^c_{\tau-1}|+1=|\mathcal{S}^c_K|$. Hence the result. □

Experiments Finally, we end this presentation of conservativeness in finite horizon MDPs with some experiments. We consider a classic 3×4 gridworld problem with one goal state, a starting state and one trap state, we set $H=10$, and the reward of any action in all the state to -2, the reward in the goal state to 10 and the reward of falling in the trapping state to -20. We normalize the rewards to be in $[0,1]$. The baseline policy is describing a path around the pit, see Fig 2.C.1. On the two position adjacent to the goal the baseline policy is stochastic with a probability of

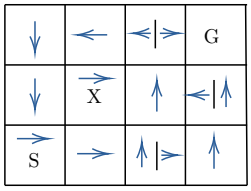


Figure 2.C.1: Illustration of the baseline policy. S is the starting state, X is the pit a,d G is the goal state.

reaching the goal of $1/2$ for the position on the right of the goal and below the goal, respectively. On the last line the probability of going up or right is also uniform. Figure 2.C.2 shows the impact of the conservative constraint on the regret of UCB-VI for a conservative coefficient $\alpha=0.05$. Fig 2.C.2 also shows the constraint as a function of the time for UCB-VI and CUCB-VI that is to say: $\sum_{l=1}^{t}V^{\pi_l}(s_0)-(1-\alpha)V^{\pi_b}(s_0)$ as a function of episode t with s_0 the starting state of the gridworld. In the first 10% episodes (i.e until episode 300) the condition was violated by UCB-VI 83% of the time.

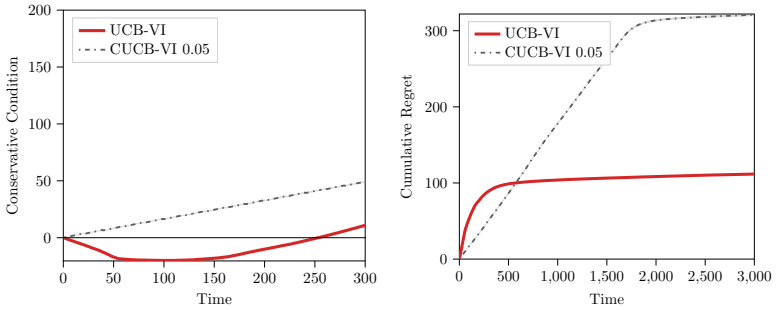


Figure 2.C.2: Regret and Conservative Condition for the gridworld problem

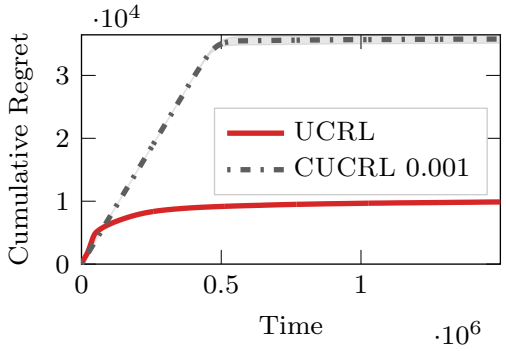


Figure 2.C.3: Regret of UCRL2 and CUCRL2 on the Cost-Based Maintenance problem described in 2.A.6

2.C.4 Experiments

For average reward problems we consider "simplified" Bernstein confidence intervals given by:

$$\beta_r^k(s,a) = \sigma_r(s,a)\sqrt{\frac{\ln(SA/\delta)}{N_k^+(s,a)}} + r_{\max}\frac{\ln(SA/\delta)}{N_k^+(s,a)} \quad \text{and} \quad \beta_p^k(s,a,s') = \sigma_p(s,a,s')\sqrt{\frac{\ln(SA/\delta)}{N_k^+(s,a)}} + \frac{\ln(SA/\delta)}{N_k^+(s,a)}$$

where $N_k^+(s,a) = \max\{1, N_k(s,a)\}$, $\sigma_r(s,a)$ is the empirical standard deviation and $\sigma_p(s,a,s') = \sqrt{\widehat{p}(s'|s,a)(1-\widehat{p}(s'|s,a))}$.

2.C.4.1 Single-Product Stochastic Inventory Control

Maintaining inventories is necessary for any company dealing with physical products. We consider the case of single product without backlogging. The state space is the amount of products in the inventory, $\mathcal{S} = \{0, \dots, M\}$ where M is the maximum capacity. Given the state s_t at the beginning of the month, the manager (agent) has to decide the amount of units a_t to order. We define D_t to be the random demand of month t and we assume a time-homogeneous probability distribution for the demand. The inventory at time $t+1$ is given by

$$s_{t+1} = \max\{0, s_t + a_t - D_t\}$$

The action space is $\mathcal{A}_s = \{0, \dots, M - s\}$. As in (Puterman, 1994), we assume a fixed cost $K > 0$ for placing orders and a varible cost $c(a)$ that increases with the quantity ordered: $O(a) = \begin{cases} K + c(a) & a > 0 \\ 0 & \text{otherwise} \end{cases}$. The cost of maintaining an inventory of s items is defined by the nondecreasing function $h(s)$. If the inventory is available to meet a demand j, the agent receives a revenue of $f(j)$. The reward is thus defined as $r(s_t, a_t, s_{t+1}) = -O(a_t) - h(s_t + a_t) + f(s_t + a_t - s_{t+1})$. In the experiments, we use $K = 4$, $c(x) = 2x$, $h(x) = x$ and $f(x) = 8x$.

In all the experiments, we normalize rewards such that the support is in $[0, 1]$ and we use noise proportional to the reward mean: $r_t(s,a) = (1 + c\eta_t)r(s,a)$ where $\eta_t \sim \mathcal{N}(0,1)$ (we set $c = 0.1$).

2.C.4.2 Cost-Based Maintenance

The system is composed by N components in an active redundant, parallel setting, which are subject to economic and stochastic dependence through load sharing. Each component $j \in [N]$ is described by its operational level $x_j = \{0, \dots, L\}$. The level L denotes that the component has failed. The deterioration process is modelled using a Poisson process. If all components have failed, the system is shut down and a penalty cost p is paid. The replacement of a failed component cost c_c, while the same operation on an active component cost c_p (usually $c_c \geq c_p$). There is also a fixed cost for maintenance c_s. At each time step, it is possible to replace simultaneously multiple components. Refer to (Olde Keizer, 2016) for a complete description of dynamics and rewards.

We terminate the analysis of CUCRL2 with a more challenging test. We consider the condition-based maintenance problem (CBM, Olde Keizer, 2016) a multi-component system subject to structural, economic and stochastic dependences. We report a complete description of the problem in App. 2.A.6. The resulting MDP has $S = 121$ states and $A = 4$ actions. The maintenance policy is often implemented as a threshold policy based on the deterioration level. Such a threshold policy is not necessarily optimal for a system with economic dependence and redundancy. We

simulate this scenario by considering a strong (almost optimal) threshold policy for CBM without economic dependence as baseline. We make it stochastic by selecting with probability 0.3 a random action. As a result we have that the optimal gain $g^\star = 0.89$ while the baseline gain is $g^{\pi_b} = 0.82$. Fig. 2.C.3 shows the cumulative regret for $\mathrm{UCRL2}$ and $\mathrm{CUCRL2}$ with $\alpha = 0.001$. $\mathrm{UCRL2}$ explores faster than $\mathrm{CUCRL2}$ but violates the conservative condition 53% of times in the initial phase (up to $t = 140000$), incurring in multiple complete system failures. On the other hand, $\mathrm{CUCRL2}$ never violates the conservative condition.

Chapter 3

Private Reinforcement Learning

In the last chapter, we studied two different performance constraints that one may encountered when deploying a Bandit or Reinforcement Learning system for a specific application. In this chapter, we study the impact of privacy constraints on the learning process in Reinforcement Learning. Privacy in Machine Learning has been studied extensively in the last two decades (Dwork et al., 2006, 2010b; Duchi et al., 2013). In this chapter, we focus on notion of differential privacy. From a high level point of view, differential privacy states that if a machine learning system is trained on two datasets that differs by at most a small number of different entries the outputs of those two systems are not statistically too different[1]. Although this notion of privacy is well defined in the standard supervised learning setting it is not the case for online learning.

This has led to a significant amount of work to define differential privacy in the MAB and linear contextual bandit (Shariff and Sheffet, 2018; Gajane et al., 2016). However, the definition of Differential Privacy for bandit does not take into account the state space in Reinforcement Learning. In this thesis, we studied how to define differential privacy in a tabular Reinforcement Learning setting. Defining this notion in RL requires to put a strong constraint on the exploration process as each action taken by the algorithm can leak information about the current state or previous visited states.

In this chapter, we first present how to build a local differential privacy algorithm for tabular Reinforcement Learning with regret guarantees. We then show how to apply a recent advances in the Differential Privacy literature to merge the two different notions of privacy –local and central differential privacy– in linear contextual bandit.

This chapter is based on the two following articles:

- **Evrard Garcelon**, Vianney Perchet, Ciara Pike-Burke, and Matteo Pirotta. Local differential privacy for regret minimization in reinforcement learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 10561–10573. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/
- **Evrard Garcelon**, Kamalika Chaudhuri, Vianney Perchet, and Matteo Pirotta. Privacy amplification via shuffling for linear contextual bandits. In Sanjoy Dasgupta and Nika Haghtalab, editors, *International Conference on Algorithmic Learning Theory, 29-1 April 2022, Paris, France*, volume 167 of *Proceedings of Machine Learning Research*, pages 381–407. PMLR, 2022b. URL https://proceedings.mlr.press/v167/garcelon22a.html

[1] The notion of "too different" is controlled quantitatively in this privacy setting.

Contents

3.1 (Local) Differential Privacy in Reinforcement Learning

The practical successes of Reinforcement Learning (RL) algorithms have led to them becoming ubiquitous in many settings such as digital marketing, healthcare and finance, where it is desirable to provide a personalized service (e.g., Mao et al., 2020; Wang and Yu, 2021). However, users are becoming increasingly wary of the amount of personal information that these services require. This is particularly pertinent in many of the aforementioned domains where the data obtained by the RL algorithm are highly sensitive. For example, in healthcare, the state encodes personal information such as gender, age, vital signs, etc. In advertising, it is normal for states to include browser history, geolocalized information, etc. Unfortunately, (Pan et al., 2019) has shown that, unless sufficient precautions are taken, the RL agent leaks information about the environment (i.e., states containing sensitive information). That is to say, observing the policy computed by the RL algorithm is sufficient to infer information about the data (e.g., states and rewards) used to compute the policy (scenario ①). This puts users' privacy at jeopardy. Users therefore want to keep their sensitive information private, not only to an observer but also to the service provider itself (i.e., the RL agent). In response, many services are adapting to provide stronger protection of user privacy and personal data, for example by guaranteeing privacy directly on the user side (scenario ②). This often means that user data (i.e., trajectories of states, actions, rewards) are privatized before being observed by the RL agent. In this paper, we study the effect that this has on the learning problem in RL.

Differential privacy (DP) (Dwork et al., 2006) is a standard mechanism for preserving data privacy, both on the algorithm and the user side. The (ε, δ)-DP definition guarantees that it is statistically hard to infer information about the data used to train a model by observing its predictions, thus addressing scenario ①. In online learning, (ε, δ)-DP has been studied in the multi-armed bandit framework (e.g., Mishra and Thakurta, 2015; Tossou and Dimitrakakis, 2016). However, (Shariff and Sheffet, 2018) showed that DP is incompatible with regret minimization in the contextual bandit problems. This led to considering weaker or different notions of privacy (e.g., Shariff and Sheffet, 2018; Boursier and Perchet, 2020). Recently, (Vietri et al., 2020) transferred some of these techniques to RL, presenting the first private algorithm for regret minimization in finite-horizon problems. In (Vietri et al., 2020), they considered a relaxed definition of DP called *joint differential privacy* (JDP) and showed that, under JDP constraints, the regret only increases by an additive term which is logarithmic in the number of episodes. Similarly to DP, in the JDP setting the privacy burden lies with the learning algorithm which directly observes user states and trajectories containing sensitive data. In particular, this means that the data itself is not private and could potentially be used –for example by the owner of the application– to train other algorithms with no privacy guarantees. An alternative and stronger definition of privacy is *Local Differential Privacy* (LDP) (Duchi et al., 2013). This requires that the user's data is protected at collection time before the learning agent has access to it. This covers scenario ② and implies that the learner is DP. Intuitively, in RL, LDP ensures that each sample (states and rewards associated to an user) is already private when observed by the learning agent, while JDP requires computation on the entire set of samples to be DP. Recently, (Zheng et al., 2020) showed that, in contrast to DP, LDP is compatible with regret minimization in contextual bandits.[2] LDP is thus a stronger definition of privacy, simpler to understand and more user friendly. These characteristics make LDP more suited for real-world applications. However, as we show in this paper, guaranteeing LDP in RL makes the learning problem more challenging.

In this chapter, we study LDP for regret minimization in finite horizon reinforcement learning problems with S states, A actions, a horizon of H and a number of episodes K.[3] Our contributions are as follows. **1)** We provide a regret lower bound for (ε, δ)-LDP of $\Omega\big(H\sqrt{SAK}/\min\{e^{\varepsilon}-1, 1\}\big)$, showing LDP is inherently harder than JDP, where the lower-bound is only $\Omega\big(H\sqrt{SAK}+SAH\log(KH)/\varepsilon\big)$ (Vietri et al., 2020). **2)** We propose the first LDP algorithm for regret minimization in RL. We use a general privacy-preserving mechanism to perturb information associated to each trajectory and derive $\mathrm{LDP\text{-}OBI}$, an optimistic model-based (ε, δ)-LDP algorithm with regret guarantees. **3)** We present multiple privacy-preserving mechanisms that are compatible with $\mathrm{LDP\text{-}OBI}$ and show that their regret is $\widetilde{O}(\sqrt{K}/\varepsilon)$ up to some mechanism dependent terms depending on S, A, H. **4)** We perform numerical simulations to evaluate the impact of LDP on the learning process. For comparison, we build a Thompson sampling algorithm (e.g., Osband et al., 2013) for which we provide LDP guarantees but no regret bound.

The notion of differential privacy was introduced in (Dwork et al., 2006) and is now a standard in machine

[2] This shows that there are peculiarities in the DP definitions that are unique to sequential decision-making problems such as RL. The discrepancy between DP and LDP in RL is due to the fact that, when guaranteeing DP, actions taken by the learner cannot depend on the current state (this would break the privacy guarantee). On the other hand, in the LDP setting, the user executes a policy prescribed by the learner on its end (i.e., directly on non-private states) and send a privatized result (sequence of states and rewards observed by executing the policy) to the learner. Hence the user can execute actions based on its current state leading to a sublinear regret.

[3] We do not explicitly focus on preventing malicious attacks or securing the communication between the RL algorithm and the users. This is outside the scope of the paper.

learning (e.g., Erlingsson et al., 2014; Dwork and Roth, 2014; Abowd, 2018). Several notions of DP have been studied in the literature, including the standard DP and LDP notions. While LDP is a stronger definition of privacy compared to DP, recent works have highlighted that it possible to achieve a trade-off between the two settings in terms of privacy and utility. The shuffling model of privacy (Cheu et al., 2019; Feldman et al., 2020; Chen et al., 2021; Balle et al., 2019b; Erlingsson et al., 2020) allows to build (ε, δ)-DP algorithm with an additional (ε', δ')-LDP guarantee (for $\varepsilon' \approx \varepsilon + \ln(n)$, any $\delta' > 0$ where n is the number of samples), hence it is possible to trade-off between DP, LDP, and utility in this setting. However, the scope of this paper is ensuring (ε, δ)-LDP guarantees for a fixed ε. In this case, shuffling will not provide an improvement in utility (error) (see Thm 5.2 in Sec. 5.1 of (Feldman et al., 2020) and App. 3.A.10).

The bandit literature has investigated different privacy notions, including DP, JDP and LDP (Mishra and Thakurta, 2015; Tossou and Dimitrakakis, 2016; Gajane et al., 2018; Shariff and Sheffet, 2018; Sajed and Sheffet, 2019; Chen et al., 2020; Zheng et al., 2020; Ren et al., 2020). In contextual bandits, (Shariff and Sheffet, 2018) derived an impossibility result for learning under DP by showing a regret lower-bound $\Omega(T)$ for any (ϵ, δ)-DP algorithm. Since the contextual bandit problem is a finite-horizon RL problem with horizon $H = 1$, this implies that DP is incompatible with regret minimization in RL as well. Regret minimization in RL with privacy guarantees has only been considered in (Vietri et al., 2020), where the authors extended the JDP approach from bandit to finite-horizon RL problems. They proposed a variation of UBEV (Dann et al., 2017) using a randomized response mechanism to guarantee ε-JDP with an additive cost to the regret bound. While *local differential privacy* (Duchi et al., 2013) has attracted increasing interest in the bandit literature (e.g., Gajane et al., 2018; Chen et al., 2020; Zheng et al., 2020; Ren et al., 2020), it remains unexplored in the RL literature, and we provide the first contribution in that direction. Finally, outside regret minimization, DP has been studied in off-policy evaluation (Balle et al., 2016), in control with DP guarantees on only the reward function (Wang and Hegde, 2019), and in distributional RL (Ono and Takahashi, 2020).

3.1.1 Basics of Differential Privacy in RL

We consider the finite-horizon time-homogeneous Markov Decision Process (MDP) (Puterman, 1994, Chp. 4) $M = (\mathcal{S}, \mathcal{A}, p, r, H)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, and horizon $H \in \mathbb{N}^+$ described in Section 1.2.3.1. Every state-action pair is characterized by a reward distribution with mean $r(s,a)$ supported in $[0,1]$ and a transition distribution $p(\cdot|s,a)$ over next state.[4] We denote by $S = |\mathcal{S}|$ and $A = |\mathcal{A}|$ the number of states and actions. A non-stationary Markovian deterministic (MD) policy is defined as a collection $\pi = (\pi_1, \dots, \pi_H)$ of MD policies $\pi_h : \mathcal{S} \to \mathcal{A}$. For any $h \in [H] := \{1, \dots, H\}$ and state $s \in \mathcal{S}$, the value functions of a policy π are defined as $Q_h^\pi(s,a) = r(s,a) + \mathbb{E}_\pi \left[\sum_{i=h+1}^H r(s_i, a_i)\right]$ and $V_h^\pi(s) = Q^\pi(s, \pi_h(s))$. There exists an optimal Markovian and deterministic policy $\pi^\star$ (Puterman, 1994, Sec. 4.4) such that $V_h^\star(s) = V_h^{\pi^\star}(s) = \max_\pi V_h^\pi(s)$. The Bellman equations at stage $h \in [H]$ are defined as $Q_h^\star(s,a) = r_h(s,a) + \max_{a'} \mathbb{E}_{s' \sim p_h(s,a')} \left[V_{h+1}^\star(s')\right]$. The value iteration algorithm (a.k.a. backward induction) computes $Q^\star$ by applying the Bellman equations starting from stage H down to 1, with $V_{H+1}^\star(s) = 0$ for any s. The optimal policy is simply the greedy policy: $\pi_h^\star(s) = \arg\max_a Q_h^\star(s,a)$. By boundness of the reward, all value functions $V_h^\pi(s)$ are bounded in $[0, H-h+1]$ for any h and s.

The general interaction protocol. The learning agent (e.g., a personalization service) interacts with an unknown MDP with multiple users in a sequence of episodes $k \in [K]$ of fixed length H. At each episode k, an user u_k arrives and their personal information (e.g., location, gender, health status, etc.) is encoded by the state $s_{1,k}$. The learner selects a policy π_k that is sent to the user u_k for local execution on "clear" states. The outcome of the execution, i.e., a trajectory, $X_k = (s_{kh}, a_{kh}, r_{kh}, s_{k,h+1})_{h \in [H]}$ is sent to the learner to update the policy. Note that we have not yet explicitly taken into consideration privacy in here. We evaluate the performance of a learning algorithm $\mathfrak{A}$ which plays policies $\pi_1, \dots, \pi_K$ by its cumulative regret after K episodes

$$\Delta(K) = \sum_{k=1}^K (V_1^\star(s_{1,k}) - V_1^{\pi_k}(s_{1,k})). \tag{3.1}$$

3.1.1.1 Local Differential Privacy in RL

In many application settings, when modelling a decision problem as a finite horizon MDP, it is natural to view each episode $k \in [K]$ as a trajectory associated to a specific user. In this paper, we assume that the sensitive information

[4] We can simply modify the algorithm to handle step dependent transitions and rewards. The regret is then multiplied by a factor $H\sqrt{H}$.

is contained in the states and rewards of the trajectory. Those quantities need to be kept private. This is reasonable in many settings such as healthcare, advertising, and finance, where states encode personal information, such as location, health, income etc. For example, an investment service may aim to provide each user with investment suggestions tailored to their income, deposit amount, age, risk level, properties owned, etc. This information is encoded in the user state and evolves over time as a consequence of investment decisions. The service provides guidances in the form of a policy (e.g., where, when and how much to invest) and the user follows the strategy for a certain amount of time. After that and based on the newly acquired information the provider may decide to change the policy. However, the user may want to keep their personal and sensitive information private to the company, while still receiving a personalised and meaningful service. This poses a fundamental challenge since in many cases, this information about actions taken in each state is essential for learning and creating a personalized experience for the user. The goal of a private RL algorithm is thus to ensure that the sensitive information remains private, while preserving the learnability of the problem.

Privacy in RL has been tackled in (Vietri et al., 2020) through the lens of *joint differential privacy* (JDP). Intuitively, JDP requires that when a user changes, the actions observed by the other $K-1$ users will not change much (Vietri et al., 2020). The privacy burden thus lies with the RL algorithm. The algorithm has access to all the information about the users (i.e., trajectories) containing sensitive data. It then has to provide guarantees about the privacy of the data and carefully select the policies to execute in order to guarantee JDP. This approach to privacy requires the user to trust the RL algorithm to privately handle the data and not to expose or share sensitive information, and does not cover the examples mentioned above.

In contrast to prior work, in this paper, we consider *local differential privacy* (LDP) in RL. This removes the requirement that the RL algorithm observes the true sensitive data, achieving stronger privacy guarantees. LDP requires that an algorithm has access to user information (trajectories in RL) only through samples that have been privatized before being passed to the learning agent. This is different to JDP or DP where the trajectories are directly fed to the RL agent. In LDP, information is secured locally by the user using a private randomizer $\mathcal{M}$, before being sent to the RL agent. The appeal of this local model is that *privatization can be done locally on the user-side*. Since nobody other than the user has ever access to any piece of non private data, this local setting is far more private. There are several variations of LDP available in the literature. In this paper, we focus on the non-interactive setting. We argue that this is more appropriate for RL. Indeed, due to the RL interaction framework, the data generated by user k is a function of the data of all users $l < k$, therefore the data are not i.i.d. and the standard definition of sequential interactivity for LDP (Eq. 1 in (Duchi et al., 2013)) is not applicable. It is therefore more natural to study the non-interactive setting (Eq. 2 in (Duchi et al., 2013)) in RL. We formally define this below.

Following the definition in (Vietri et al., 2020), a user u is characterized by a starting state distribution $\rho_{0,u}$ (i.e., for user u, $s_1 \sim \rho_{0,u}$) and a tree of depth H, describing all the possible sequence of states and rewards corresponding to all possible sequences of actions. Alg. 10 describes the LDP private interaction protocol between K unique users $\{u_1, \dots, u_K\} \subset \mathcal{U}^K$, with $\mathcal{U}$ the set of all users, and an RL algorithm $\mathfrak{A}$. For any $k \in [K]$, let $s_{1,k} \sim \rho_{0,u_k}$ be the initial state for user u_k and denote by $X_{u_k} = \{(s_{k,h}, a_{k,h}, r_{k,h}) \mid h \in [H]\} \in \mathcal{X}_{u_k}$ the trajectory corresponding to user u_k executing a policy π_k. We write $\mathcal{M}(X_{u_k})$ to denote the privatized data generated by the randomizer for user u_k. The goal of mechanism $\mathcal{M}$ is to privatize sensitive informations while encoding sufficient information for learning. With these notions in mind, LDP in RL can be defined as follows:

Definition 1. *For any $\varepsilon \geq 0$ and $\delta \geq 0$, a privacy preserving mechanism $\mathcal{M}$ is said to be (ε, δ)-Locally Differential Private (LDP) if and only if for all users $u, u' \in \mathcal{U}$, trajectories $(X_u, X_{u'}) \in \mathcal{X}_u \times \mathcal{X}_{u'}$ and all $O \subset \{\mathcal{M}(\mathcal{X}_u) \mid u \in \mathcal{U}\}$:*

$$\mathbb{P}\left(\mathcal{M}(X_u) \in O\right) \leq e^{\varepsilon}\, \mathbb{P}\left(\mathcal{M}(X_{u'}) \in O\right) + \delta \tag{3.2}$$

where $\mathcal{X}_u$ is the space of trajectories associated to user u.

Def. 3 ensures that if the RL algorithm observes the output of the privacy mechanism $\mathcal{M}$ for two different input trajectories, then it is statistically difficult to guess which output is from which input trajectory. As a consequence, the users' privacy is preserved.

3.1.2 Regret Lower Bound Under LDP Constraint in RL

We provide a lower bound on the regret that any LDP RL algorithm must incur. For this, as is standard when proving lower bounds on the regret in RL (e.g., Auer et al., 2002b; Lattimore and Szepesvári, 2020), we construct a hard instance of the problem. The proof (see App. 3.A.2) relies on the fact that LDP acts as Lipschitz function, with respect to the KL-divergence, in the space of probability distribution.

Algorithm 10: Locally Private Episodic RL

Input: Agent: $\mathfrak{A}$, Local Randomizer: $\mathcal{M}$, Users: $u_1, \dots, u_K$

for $k = 1$ ***to*** K **do**

- Agent $\mathfrak{A}$ computes π_k using $\{\mathcal{M}(X_{u_l})\}_{l\in[K-1]}$
- User u_k receives π_k from agent $\mathfrak{A}$ and observes $s_{1,k} \sim \rho_{0,u_k}$
- User u_k executes policy π_k on "non-private" states and observes a trajectory $X_{u_k} = \{(s_{h,k}, a_{h,k}, r_{h,k})\}_{h\in[H]}$
- User u_k sends back private data $\mathcal{M}(X_{u_k})$ to $\mathfrak{A}$

Algorithm 11: LDP-OBI ($\mathcal{M}$)

Input: $\delta \in (0,1)$, $\alpha > 1$, randomizer $\mathcal{M}$ with parameters (ϵ_0, δ_0)

for $k = 1$ ***to*** K **do**

- Compute $\widetilde{p}_k$ and $\widetilde{r}_k$ as in Eq. (3.4) using $\{\mathcal{M}(X_{u_l})\}_{l\in[K-1]}$, β_k^r and β_k^p as in Prop. 6 using $\{c_{k,i}(\varepsilon_0, \delta_0, \frac{3\delta}{2k^2\pi^2})\}_i$, and $b_{h,k}$
- Compute π_k as in Eq. (3.5) and send it to user u_k
- User u_k executes policy π_k, collects trajectory X_k and sends back privatized value $\mathcal{M}(X_k)$

Theorem 6 (Lower-Bound). *For any algorithm $\mathfrak{A}$ associated to a ε-LDP mechanism, any number of states $S \geq 3$, actions $A \geq 2$ and $H \geq 2\log_A(S-2)+2$, there exists an MDP M with S states and A actions such that:* $\mathbb{E}_M(\Delta(K)) \geq \Omega\left(\frac{H\sqrt{SAK}}{\min\{\exp(\varepsilon)-1,1\}}\right)$.

The lower bound of Thm. 6 shows that the price to pay for LDP in the RL setting is a factor $1/(\exp(\varepsilon)-1)$ compared to the non-private lower bound of $H\sqrt{SAK}$. The regret lower bound scales multiplicatively with the privacy parameter ε. The recent work of (Vietri et al., 2020) shows that for JDP, the regret in finite-horizon MDPs is lower-bounded by $\Omega\left(H\sqrt{SAK} + \frac{1}{\varepsilon}\right)$. Thm. 6 shows that the local differential privacy setting is inherently harder than the joint differential privacy one for small ϵ, as our lower-bound scales with $\sqrt{K}/\varepsilon$ when $\varepsilon \approxeq 0$. Both bounds scale with $\sqrt{K}$ when $\varepsilon \to +\infty$.

3.1.3 Exploration Under Local Differential Privacy

A standard approach to the design of the private randomizer $\mathcal{M}$ is to inject noise into the data to be preserved (Dwork and Roth, 2014). A key challenge in RL is that we cannot simply inject noise to each component of the trajectory since this will break the *temporal consistency* of the trajectory and possibly prevent learning. In fact, a trajectory is not an arbitrary sequence of states, actions, and rewards but obeys the Markov reward process induced by a policy. Fortunately, Def. 3 shows that the output of the randomizer need not necessarily be a trajectory but could be any private information built from it. In the next subsection, we show how to leverage this key feature to output succinct information that preserves the information encoded in a trajectory while satisfying the privacy constraints. We show that the output of such a randomizer can be used by an RL algorithm to build estimates of the unknown rewards and transitions. While these estimates are biased, we show that they carry enough information to derive optimistic policies for exploration. We leverage these tools to design LDP-OBI, an optimistic model-based algorithm for exploration with LDP guarantees.

3.1.3.1 Privacy-Preserving Mechanism

Consider the locally-private episodic RL protocol described in Alg. 10. At the end of each episode $k \in [K]$, user u_k uses a private randomizer $\mathcal{M}$ to generate a private statistic $\mathcal{M}(X_{u_k})$ to pass to the RL algorithm $\mathfrak{A}$. This statistic should encode sufficient information for the RL algorithm to improve the policy while maintaining the user's privacy. In *model-based* settings, a sufficient statistic is a local estimate of the rewards and transitions. Since this cannot be reliably obtained from a single trajectory, we resort to counters of visits and rewards that can be aggregated by the RL algorithm.

For a given trajectory $X = \{(s_h, a_h, r_h)\}_{h\in[H]}$, let $R_X(s,a) = \sum_{h=1}^H r_h \mathbb{1}_{\{s_h=s, a_h=a\}}$, $N_X^r(s,a) = \sum_{h=1}^H \mathbb{1}_{\{s_h=s,a_h=a\}}$ and $N_X^p(s,a,s') = \sum_{h=1}^{H-1} \mathbb{1}_{\{s_h=s,a_h=a,s_{h+1}=s'\}}$ be the true non-private statistics, which the agent will never observe. We design the mechanism $\mathcal{M}$ so that for a given trajectory X, $\mathcal{M}$ returns private versions $\mathcal{M}(X) = (\widetilde{R}_X, \widetilde{N}_X^r, \widetilde{N}_X^p)$ of these statistics. Here, $\widetilde{R}_X(s,a)$ is a noisy version of the cumulative reward $R_X(s,a)$, and $\widetilde{N}_X^r$ and $\widetilde{N}_X^p$ are perturbed counters of visits to state-action and state-action-next state tuples, respectively. At the beginning of episode k, the algorithm has access to the aggregated private statistics:

$$\widetilde{R}_k(s,a) = \sum_{l<k} \widetilde{R}_{X_{u_l}}(s,a),\ \ \widetilde{N}_k^r(s,a) = \sum_{l<k} \widetilde{N}_{X_{u_l}}^r(s,a),\ \ \widetilde{N}_k^p(s,a,s') = \sum_{l<k} \widetilde{N}_{X_{u_l}}^p(s,a,s') \tag{3.3}$$

We denote the non-private counterparts of these aggregated statistics as $R_k(s,a) = \sum_{l<k} R_{X_{u_l}}(s,a)$, $N_k^r(s,a) = \sum_{l<k} N_{X_{u_l}}^r(s,a)$ and $N_k^p(s,a,s') = \sum_{l<k} N_{X_{u_l}}^p(s,a,s')$, these are also *unknown* to the RL agent. Using these private statistics, we can define conditions that a private randomizer must satisfy in order for our RL agent, LDP-OBI, to be able to learn the reward and dynamics of the MDP.

Assumption 11. *The private randomizer $\mathcal{M}$ satisfies $(\varepsilon_0, \delta_0)$-LDP, Def. 3, with $\varepsilon_0, \delta_0 \geq 0$. Moreover, for any $\delta > 0$ and $k \geq 0$, there exist four finite strictly positive function, $c_{k,1}(\varepsilon_0,\delta_0,\delta), c_{k,2}(\varepsilon_0,\delta_0,\delta), c_{k,3}(\varepsilon_0,\delta_0,\delta), c_{k,4}(\varepsilon_0,\delta_0,\delta) \in \mathbb{R}_+^\star$ such that with probabilty at least $1-\delta$ for all $(s,a,s') \in \mathcal{S}\times\mathcal{A}\times\mathcal{S}$:*

$$\left|\widetilde{R}_k(s,a) - R_k(s,a)\right| \leq c_{k,1}(\varepsilon_0,\delta_0,\delta), \qquad \left|\widetilde{N}_k^r(s,a) - N_k^r(s,a)\right| \leq c_{k,2}(\varepsilon_0,\delta_0,\delta)$$

$$\left|\sum_{s'} N_k^p(s,a,s') - \widetilde{N}_k^p(s,a,s')\right| \leq c_{k,3}(\varepsilon_0,\delta_0,\delta), \qquad \left|N_k^p(s,a,s') - \widetilde{N}_k^p(s,a,s')\right| \leq c_{k,4}(\varepsilon_0,\delta_0,\delta)$$

The functions $c_{k,1}(\varepsilon_0,\delta_0,\delta)$, $c_{k,2}(\varepsilon_0,\delta_0,\delta)$, $c_{k,3}(\varepsilon_0,\delta_0,\delta)$ and $c_{k,4}(\varepsilon_0,\delta_0,\delta)$ must be increasing functions of k and decreasing functions of δ. We also write $c_{k,1}(\varepsilon_0,\delta)$, $c_{k,2}(\varepsilon_0,\delta)$, $c_{k,3}(\varepsilon_0,\delta)$ and $c_{k,4}(\varepsilon_0,\delta)$ when $\delta_0 = 0$.

In Sec. 3.1.4, we will present schemas satisfying Asm. 11 and discuss their impacts on privacy and regret.

3.1.3.2 Our LDP Algorithm For Exploration

In this subsection, we introduce LDP-OBI (*Local Differentially Private Optimistic Backward Induction*), a flexible optimistic model-based algorithm for exploration that can be paired with any privacy mechanism satisfying Asm. 11. When developing optimistic algorithms it is necessary to define confidence intervals using an estimated model that are broad enough to capture the true model with high probability, but narrow enough to ensure low regret. This is made more complicated in the LDP setting, since the estimated model is defined using randomized counters. In particular, this means we cannot use standard concentration inequalities such as those used in (Azar et al., 2017a; Zanette and Brunskill, 2019). Moreover, when working with randomized counters, classical estimators like the empirical mean can even be ill-defined as the number of visits to a state-action pair, for example, can be negative.

Nevertheless, we show that by exploiting the properties of the mechanism $\mathcal{M}$ in Asm. 11, it is still possible to define an empirical model which can be shown to be close to the true model with high probability. To construct this empirical estimator, we rely on the fact that for each state-action pair (s,a), $\widetilde{N}_k^r(s,a) + c_{k,2}(\varepsilon_0,\delta_0,\delta) \geq N_k^r(s,a) \geq 0$ with high probability where the precision $c_{k,2}(\varepsilon_0,\delta_0,\delta)$ ensures the positivity of the noisy number of visits to a state action-pair. A similar argument holds for the transitions. Formally, the estimated *private* rewards and transitions before episode k are defined as follows:

$$\widetilde{r}_k(s,a) = \frac{\widetilde{R}_k(s,a)}{\widetilde{N}_k^r(s,a) + \alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}, \qquad \widetilde{p}_k(s' \mid s,a) = \frac{\widetilde{N}_k^p(s,a,s')}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} \tag{3.4}$$

Note that unlike in classic optimistic algorithms, $\widetilde{p}_k$ is not a probability measure but a signed sub-probability measure. However, this does not preclude good performance. By leveraging properties of Asm. 11 we are able to build confidence intervals using these private quantities (see App. 3.A.5). Using standard concentration inequalities on each component would lead to wider confidence intervals.

Proposition 6. *For any $\varepsilon_0 > 0$, $\delta_0 \geq 0$, $\delta > 0$, $\alpha > 1$ and episode k, using mechanism $\mathcal{M}$ satisfying Asm. 11, then with probability at least $1-2\delta$, for any $(s,a) \in \mathcal{S}\times\mathcal{A}$*

$$|r(s,a) - \widetilde{r}_k(s,a)| \leq \beta_k^r(s,a) = \sqrt{\frac{2\ln\left(\frac{4\pi^2 SAHk^3}{3\delta}\right)}{\widetilde{N}_k^r(s,a) + \alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}} + \frac{(\alpha+1)c_{k,2}(\varepsilon_0,\delta_0,\delta) + c_{k,1}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^r(s,a) + \alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}$$

$$\|p(\cdot|s,a) - \widetilde{p}_k(\cdot|s,a)\|_1 \leq \beta_k^p(s,a) = \sqrt{\frac{14S\ln\left(\frac{4\pi^2 SAHk^3}{3\delta}\right)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}} + \frac{Sc_{k,4}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} + \frac{(\alpha+1)c_{k,3}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}$$

The shape of the bonuses in Prop. 6 highlights two terms. The first term is reminiscent of Hoeffding bonuses as it scales with $\mathcal{O}\left(1/\sqrt{\widetilde{N}_k^p}\right)$. The other term is of order $\mathcal{O}\left(1/\widetilde{N}_k^p\right)$ and accounts for the variance (and potentially bias) of the noise added by the privacy-preserving mechanism.

$\mathcal{M}$	Noise	(ϵ, δ)-LDP level	Regret $\Delta(T)$
Laplace	$\text{Lap}(6H/\varepsilon)$	$(\varepsilon, 0)$	$\widetilde{O}(H^3S^2A\sqrt{K}/\varepsilon)$
Gaussian	$\mathcal{N}(0, (H/\varepsilon)^2)$	(ε, δ_0)	$\widetilde{O}(H^3S^2A\sqrt{K\ln(1/\delta_0)}/\varepsilon)$
Randomized Response	$\text{Ber}((e^{\varepsilon/H}-1)^{-1})$	$(\varepsilon, 0)$	$\widetilde{O}(H^{7/2}S^2A\sqrt{K}/\varepsilon)$
Bounded Noise	See (Dagan and Kur, 2020) and App. 3.A.6.3	(ε, δ_0)	$\widetilde{O}(H^2S^3A^{3/2}\sqrt{K\ln(1/\delta_0)}/\varepsilon)$

Table 3.1.1: Summary of the guarantees of LDP-OBI with different randomizers for $\varepsilon > 0$ and $\delta_0 > 0$. For the mechanism in this table, we have approximately $c_{k,i} = \widetilde{\mathcal{O}}(\sqrt{kH}/\varepsilon)$ for $i \in \{1, 2, 4\}$ (ignoring $\log$ terms) and $c_{k,3} = \widetilde{\mathcal{O}}(\sqrt{SkH}/\varepsilon)$

As commonly done in the literature (e.g., Azar et al., 2017a; Qian et al., 2019; Neu and Pike-Burke, 2020), we use these concentration results to define a bonus function $b_{h,k}(s,a) := (H-h+1)\cdot\beta_k^p(s,a) + \beta_k^r(s,a)$ which is used to define an optimistic value function and policy by running the following backward induction procedure:

$$Q_{h,k}(s,a)s = \widetilde{r}_k(s,a) + b_{h,k}(s,a) + \widetilde{p}_k(\cdot|s,a)^\mathsf{T} V_{h+1,k}, \quad \pi_{h,k}(s) = \arg\max_a Q_{h,k}(s,a) \tag{3.5}$$

where $V_{h,k}(s) = \min\{H-h+1, \max_a Q_{h,k}(s,a)\}$ and $V_{H+1,k}(s) = 0$.

3.1.3.3 Regret Guarantees

We get the following general guarantees for any LDP mechanism satisfying Asm. 11 in LDP-OBI.

Theorem 7. *For any privacy mechanism $\mathcal{M}$ satisfying Asm. 11 with $\varepsilon > 0$, $\delta_0 \geq 0$, and for any $\delta > 0$ the regret of* LDP-OBI *is bounded with probability at least $1-\delta$ by:*

$$\begin{aligned}\Delta(K) \leq \tilde{\mathcal{O}}\Bigg(\underbrace{HS\sqrt{AT}}_{\text{❶}} &+ SAH^2 c_{K,3}\left(\varepsilon, \delta_0, \frac{3\delta}{2\pi^2K^2}\right) + H^2S^2Ac_{K,4}\left(\varepsilon, \delta_0, \frac{3\delta}{2\pi^2K^2}\right) \\ &+ SAHc_{K,2}\left(\varepsilon, \delta_0, \frac{3\delta}{2\pi^2K^2}\right) + SAHc_{K,1}\left(\varepsilon, \delta_0, \frac{3\delta}{2\pi^2K^2}\right)\Bigg)\end{aligned} \tag{3.6}$$

The combination of $\mathcal{M}$ and LDP-OBI *is also (ε, δ_0)-LDP.*

Thm. 7 shows that the regret of LDP-OBI *1)* is lower bounded by the regret in non-private settings; and *2)* depends directly on the precision of the privacy mechanism used though $c_{K,1}, \dots, c_{K,4}$. Thus improving the precision, that is to say reducing the amount of noise that needs to be added to the data to guarantee LDP of the privacy mechanism, directly improves the regret bounds of LDP-OBI. The first term in the regret bound (❶) is of the order expected in the non-private setting (see e.g., Jaksch et al. (2010a)). Classical results in DP suggest that the $\{c_{K,i}\}_{i\leq 4}$ terms should be *approximately* of order $\sqrt{K}/\varepsilon$ (this is indeed the case for many natural choices of randomizer). In such a case, the dominant term in (3.6), is no longer ❶ but rather a term of order $H^2S^2A\sqrt{K}/\varepsilon$ (from e.g. $c_{K,4}$). The dependency on S, A, H is larger than in the non-private setting. This is because the cost of LDP is multiplicative, so it also impacts the lower order terms in the concentration results (see e.g. the second term in 6), which are typically ignored in the non-private setting. In addition, this implies that variance reduction techniques for RL (e.g., based on Bernstein) classically used to decrease the dependence on S, H will not lead to any improvement here. This is to be contrasted with the JDP setting where Vietri et al. (2020) shows that the cost of privacy is additive so using variance reduction techniques can reduce the dependency of the regret on S, A, H.

3.1.4 Choice of Randomizer

There are several randomizers that satisfy Asm. 11, for example Laplace (Dwork and Roth, 2014), randomized response (Erlingsson et al., 2014; Kairouz et al., 2016), Gaussian (Wang et al., 2019b) and bounded noise (Dagan and Kur, 2020) mechanisms. Since one method can be preferred to another depending on the application, we believe it is important to understand the regret and privacy guarantees achieved by LDP-OBI with these randomizers.

Tab. 3.1.1 provides a global overview of the properties of LDP-OBI with different randomized mechanism. The detailed derivations are deferred to App. 3.A.6.

Privacy. All the mechanisms satisfy Asm. 11 but only the Laplace and Randomized Response mechanisms guarantees $(\varepsilon, 0)$-LDP. Note that in all cases, in order to guarantee a ε level of privacy (or (ε, δ) for the Gaussian and bounded noise mechanisms), it is necessary to scale the parameter ε proportional to $1/H$. This is because the statistics computed by the privacy-preserving mechanism are the sum of H observations which are bounded in $[0, 1]$, the sensitivity[5] of those statistics is bounded by H. Directly applying the composition theorem for DP (Dwork and Roth, 2014, Thm 3.14) over the different counters, would lead to an upper-bound on the privacy of the mechanism of $S^2AH\varepsilon$ and corresponding regret bound of $\widetilde{O}\left((H^4S^4A^2\sqrt{K})/\varepsilon\right)$. For the randomizers that we use, the impact on ε is lower thanks to fact that they are designed to exploit the structure of the input data (a trajectory).

Regret Bound. From looking at Table 3.1.1, we see that while all the mechanisms achieve a regret bound of order $\widetilde{O}(\sqrt{K})$ the dependence on the privacy level ε varies as well as the privacy guarantees. The regret of Laplace, Gaussian and bounded noise mechanisms scale with ε^{-1}, whereas the randomized response has an exponential dependence in ε similar to the lower bound. However, this improvement comes at the price of worse dependency in H when ε is small, and a worse multiplicative constant in the regret. This is due to the randomized response mechanism perturbing the counters for each stage $h \in [H]$, leading to up to HS^2A obfuscated elements. This worse dependence is also observed in our numerical simulations.

For many of the randomizers, our regret bounds scale as $\widetilde{O}(H^3S^2A\sqrt{K}/\varepsilon)$. Aside from the $\sqrt{K}/\epsilon$ rate which is expected, our bounds exhibit worse dependence on the MDP characteristics when compared to the non-private setting. We believe that this is unavoidable due to the fact that we have to make S^2A terms private, while the extra dependence on H comes from dividing ε by H to ensure privacy over the whole trajectory. Moreover, the DP literature (e.g., Duchi et al., 2016; Duchi and Rogers, 2019; Ye and Barg, 2018) suggests that the extra dependency on S, A, H may be inherent to model-based algorithms due to the explicit estimation of private rewards and transitions. Indeed, (Ye and Barg, 2018) shows that the minimax error rate in ℓ_1 norm for estimating a distribution over S states is $\Omega\left(\frac{S}{\sqrt{n}(\exp(\varepsilon)-1)}\right)$ with n samples in the high privacy regime ($\varepsilon < 1$), while there is no change in the low privacy regime. This means that in the high privacy regime the concentration scales with a multiplicative $\sqrt{S}$ term which would translate directly into the regret bound. Furthermore, this results assumes that the number n of samples is known to the learner. In our setting, n maps to $N_k(s, a)$ which is unknown to the algorithm. Since we only observe a perturbed estimate of n, estimating $p(\cdot|s, a)$ here is strictly harder than the aforementioned setting.[6] This suggests that it is impossible for any model-based algorithm which directly estimates the transition probabilities to match the lower bound. However, this does not rule out the possibility of a model-free algorithm being able to match the lower bound. Designing such a model-free algorithm which is able to work with LDP trajectories is non-trivial and we leave it to future work.

Another direction for future work is to investigate whether the recently developed shuffling model (Erlingsson et al., 2019) may be used to improve our regret bounds in the LDP setting. Preliminary investigations of the shuffling model (see App. 3.A.10) show that it is not possible while preserving a fixed ε-LDP constraint, which is the focus of this paper. Nonetheless, if we were to relax the privacy constraint to only guarantee ε-JDP then the shuffling model could be used to retrieve the regret bound in (Vietri et al., 2020) while guaranteeing some level of local differential privacy, although the level of LDP would be much weaker than the one considered in this paper. We believe the study of this model sitting in-between the joint and local DP settings for RL is a promising direction for future work and that the tools developed in this paper will be helpful for tackling this problem.

3.1.5 Numerical Evaluation

In this subsection, we evaluate the empirical performance of LDP-OBI on a toy MDP. We compare LDP-OBI with the *non-private* algorithm UCB-VI (Azar et al., 2017a). To the best of our knowledge there is no other LDP algorithm for regret minimization in MDPs in the literature. To increase the comparators, we introduce a novel LDP

[5] For a function $f : \mathcal{X} \to \mathbb{R}$ the sensitivity is defined as $S(f) = \max_{x,y\in\mathcal{X}} |f(x) - f(y)|$

[6] We are not aware of any lower-bound in the literature that applies to this setting but we believe that the $S^2A\sqrt{KH}/\varepsilon$ dependence may be unavoidable for model-based algorithms. This is because $N_k(s, a)$ and $\widetilde{N}_k(s, a)$ differ by at most $\sqrt{kH\log(SA)}$ (which is a well-known lower bound for the counting elements problem see (Bassily and Smith, 2015)). Intuitively this difference creates a bias when estimating each component $p(\cdot|s, a)$, a bias that would scale with the size of the support $p(\cdot|s, a)$ and the relative difference between $N_k(s, a)$ and $\widetilde{N}_k(s, a)$. Hence, the bias would scale with $S\sqrt{kH}/N_k(s, a)$. Summing over all episodes and SA counters gives the conjectured result.

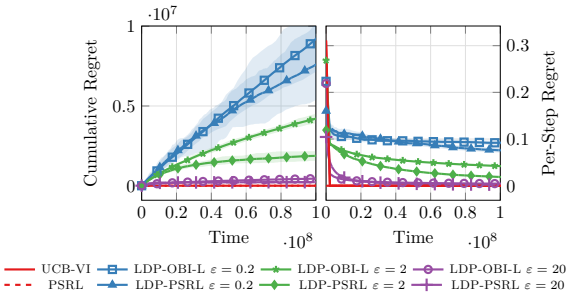


Figure 3.1.1: Evaluation of LDP-OBI with the Laplace mechanism and LDP-PSRL. *Left)* Cumulative regret. *Right)* per-step regret ($k \mapsto R_k/k$).

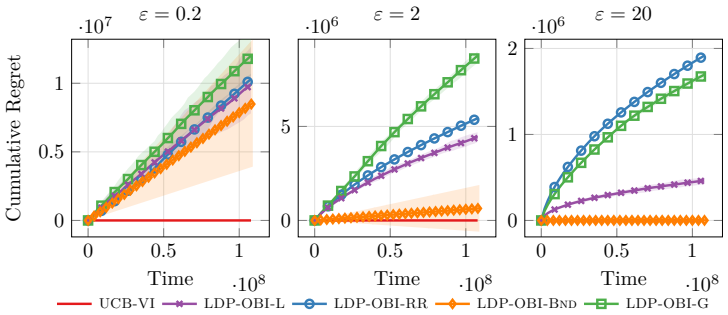


Figure 3.1.2: Regret for LDP-OBI coupled with different mechanisms. For all ε, $\delta = 0.1$ for the Gaussian and Bounded Noise mechanism.

algorithm based on Thompson sampling (e.g., Osband et al., 2013).

LDP-PSRL. Thompson sampling algorithms (e.g., PSRL, Osband et al., 2013) have proved to be effective in several applications (Russo et al., 2018). Due to their inherent randomization, one may imagine that they are also well suited to LDP regret minimization. Here, we introduce and evaluate LDP-PSRL, an LDP variant of PSRL and provide a first empirical evaluation. Informally, by defining by $\mathcal{W}_k = \{(S, A, p, r, H) : \|p - \widetilde{p}\|_1 \le \beta_k^p, |r - \widetilde{r}| \le \beta_k^r\}$ the *private* set of plausible MDPs constructed using the definition in Prop. 6, we can see posterior sampling as drawing an MDP from this set at each episode k and running the associated optimal policy:

$$i)\ M_k \sim \mathbb{P}(\mathcal{W}_k), \qquad ii)\ \pi_k = \max_{\pi} \{V_1^{\pi}(M_k)\}.$$

More formally, we consider Gaussian and Dirichlet prior for rewards and transition which lead to Normal-Gamma and Dirichlet distributions as posteriors. We use the private counters defined in Asm. 11 to update the parameters of the posterior distribution and thus the distribution over plausible models. We provide full details of this schema in App. 3.A.8 and show that it is LDP. However, we were not able to provide a regret bound for this algorithm.

Simulations. We consider the RandomMDP environment described in (Dann et al., 2017) where for each state-action pair transition probabilities are sampled from a Dirichlet(α) distribution (with $\alpha_{s,a,s'} = 0.1$ for all (s, a, s')) and rewards are deterministic in $\{0, 1\}$ with $r(s, a) = \mathbb{1}_{\{U_{s,a} \le 0.5\}}$ for $(U_{s,a})_{(s,a)\in\mathcal{S}\times\mathcal{A}} \sim \mathcal{U}([0, 1])$ sampled once when generating the MDP. We set the number of states $S = 2$, number of actions $A = 2$ and horizon $H = 2$. We evaluate the regret of our algorithm for $\varepsilon \in \{0.2, 2, 20\}$ and $K = 1 \times 10^8$ episodes. For each ε, we run 20 simulations. Confidence intervals are the minimum and maximum runs. Fig. 3.1.1 shows that the learning speed of the optimistic algorithm LDP-OBI is severely impacted by the LDP constraint. This is consistent with our theoretical results. The reason for this is the very large confidence intervals that are needed to deal with the noise from the privacy preserving mechanism that is necessary to guarantee privacy. While the regret looks almost linear for $\varepsilon = 0.2$, the decreasing trend of the per-step regret shows that LDP-OBI-L is learning. Although these experimental results only consider a small MDP, we expect that many of the observations will carry across to larger, more practical settings. However, further experiments are needed to conclusively assess the impact of LDP in large MDPs. Fig. 3.1.1 also shows that LDP-PSRL performs slightly better than LDP-OBI. This is to be expected, since even in the non-private case PSRL usually outperforms optimistic algorithm empirically. Finally, Fig. 3.1.2 compares the mechanisms with different privacy levels and illustrates the empirical impact of the privacy-preserving mechanism on the performance of LDP-OBI. We observe empirically that the bounded noise mechanism is the most effective approach, followed by the Laplace mechanism. However, the former suffers from a higher variance in its performance.

3.1.6 Concluding Remarks and Potential Extensions

We have introduced the definition of local differential privacy in RL and designed the first LDP algorithm, LDP-OBI, for regret minimization in finite-horizon MDPs. We provided an intuition why model-based approaches may suffer a higher dependence in the MDP characteristics. Designing a model-free algorithm able to reduce or close the gap with the lower-bound is an interesting technical question for future works. As mentioned in this subsection, the shuffling

privacy model does not provide any privacy/regret improvement in the strong LDP setting. An interesting direction is to investigate the trade-off between JDP and LDP that can be obtained in RL using shuffling. In particular, we believe that, sacrificing LDP guarantees, it is possible to achieve better regret leveraging variance reduction techniques (that are not helpful in strong LDP settings). Finally, there are other privacy definition that can be interesting for RL. For example, profile-based privacy (Geumlek and Chaudhuri, 2019; Acharya et al., 2020) allows to privatize only specific information or geo-privacy (Andrés et al., 2013) focuses on privacy between elements that are "similar".

In the next subsection, we further develop the notion of shuffling differential privacy, albeit in the linear contextual bandit setting, to show how to bridge the gap between the strong privacy guarantees of LDP (but at the cost of high regret) and the weaker but with less impact notion of privacy JDP. Using, this third definition of privacy, shuffling differential privacy it is possible to interpolate between the two setting and almost recover the privacy properties of LDP and the regret guarantees of JDP.

3.2 Improving Privacy by Shuffling

In a *contextual bandit algorithm*, at each time $t \in [T] := \{1, \dots, T\}$, a learner first observes a set of features $(x_{t,a})_{a\in[K]} \subset \mathbb{R}^d$, selects an action $a_t \in [K]$ out of a set of K actions, and observes a reward $r_t = r(x_{t,a_t}) + \eta_t$ where η_t is a conditionally independent zero-mean noise (r is not known beforehand), as described in Section 1.2.2. Consequently, the learning algorithm has to balance exploration of the environment with exploitation of the current knowledge to maximize the cumulative reward. The performance of the the learner is measured by the cumulative regret, which is the difference between its own cumulative reward, and the cumulative reward it would have received had it always played the best action. Contextual bandit algorithms have achieved great practical success, and have been used for many sensitive applications such as personalization, digital marketing, healthcare and finance (e.g., Mao et al., 2020; Wang and Yu, 2021). With these applications in mind, the literature has started investigated privacy guarantees both in bandits (e.g., Shariff and Sheffet, 2018; Zheng et al., 2020) and in RL (e.g., Vietri et al., 2020; **Garcelon** et al., 2021). In this paper, we focus on privacy-preserving contextual bandits.

For a contextual bandit problem on sensitive data, we assume that a single user enters the system at time t, and hence the context at time t is their private information. To measure privacy, we use differential privacy (Dwork et al., 2006) – a privacy definition introduced by cryptographers that has emerged as the gold standard for privacy-preserving data analysis (e.g., Erlingsson et al., 2014; Dwork and Roth, 2014; Abowd, 2018; Chaudhuri et al., 2011; Abadi et al., 2016; Boursier and Perchet, 2020). The standard differential privacy framework applies to static data in a batch setting, but two extensions have been proposed to address online problems. The first is Joint Differential Privacy (JDP) (e.g., Shariff and Sheffet, 2018), an analogue of central differential privacy, where the users trust the bandit algorithm. JDP ensures that changing a single user's private information in the data does not change the probability of any future outcome (namely, actions taken and rewards received by any other user) by much.

Definition 2 (Joint DP). *For $\varepsilon > 0$ and $\delta_0 > 0$, a randomized bandit agent $\mathfrak{A}$ is (ε, δ_0)-*joint differentially *private if for every $t \in [T]$, two sequences of users, $U = \{u_1, \dots, u_T\}$ and $U' = \{u'_1, \dots, u'_T\}$, that differs only for the t-th user and for all events $E \subset \mathcal{A}^{[T-1]}$ then:*

$$\mathbb{P}(\mathfrak{A}_{-t}(U) \in E) \leq e^{\varepsilon}\mathbb{P}(\mathfrak{A}_{-t}(U') \in E) + \delta_0 \tag{3.7}$$

where $\mathfrak{A}_{-t}(U)$ denotes all the outputs of algorithm $\mathfrak{A}$, i.e., all actions $(a_i)_{i\neq t}$ excluding the output of time t for the sequence of users U.

A second, stronger concept is Local Differential Privacy (LDP) (e.g., Zheng et al., 2020), where the users do not trust the bandit algorithm, and transmit only sanitized versions (using a private randomizer $\mathcal{M}$) of their contexts and rewards to the algorithm. Here, LDP ensures that user information is sanitized in such a manner that changing a single user's private value does not alter the distribution of the sanitized value by much.

Definition 3 (Local DP). *For any $\varepsilon \geq 0$ and $\delta \geq 0$, a privacy preserving mechanism $\mathcal{M}$ is said to be (ε, δ)-*locally differential *private if and only if for all users $u, u' \in \mathcal{U}$, contexts/rewards $((x_u, r_u), (x_{u'}, r_{u'})) \in (\mathbb{R}^d \times \mathbb{R})^2$ and all $O \subset \{\mathcal{M}(\mathcal{B}(0, L) \times [0, 1]) \mid u \in \mathcal{U}\}$:*

$$\mathbb{P}\left(\mathcal{M}((x_u, r_u)) \in O\right) \leq e^{\varepsilon}\,\mathbb{P}\left(\mathcal{M}((x_{u'}, r_{u'})) \in O\right) + \delta \tag{3.8}$$

where $\mathcal{B}(0, L) \times [0, 1]$ is the space of context/reward associated to user u.

Just like the standard batch setting, while LDP offers a strong notion of privacy, its utility is often much lower. Specifically, for contextual linear bandit algorithms, while ε-JDP guarantees can be obtained by paying a multiplicative factor in the regret, LDP comes with a much higher impact on the regret. In fact, Zheng et al. (2020) have shown that ε-LDP regret scales with $\widetilde{\mathcal{O}}(T^{3/4}/\sqrt{\varepsilon})$ instead of $\widetilde{\mathcal{O}}(T^{1/2}/\sqrt{\varepsilon})$ for a ε-JDP algorithm (see Tab. 3.2.1 for more details.)

Real applications are gradually moving away from the *centralized* model of privacy, favoring the simpler and stronger notion of local privacy. This change is illustrated by the rise of on-device computation for mobile application (e.g., Apple). The natural question we address in this paper is:

Is it possible to design a bandit algorithm with guarantees akin to local privacy but better utility?

To address this question, we consider the shuffle model of privacy (e.g., Cheu et al., 2019; Feldman et al., 2020; Chen et al., 2021; Balle et al., 2019b; Erlingsson et al., 2020) that, in supervised learning settings, allow to achieve a trade-off between central and local DP through a shuffler. The shuffler receives users' reports and permutes them before sending them to the server. This setting was first introduced in Bittau et al. (2017), named the *ESA* model (Encode-Shuffle-Analyze) and motivated by the need for anonymous data collection. Erlingsson et al. (2019) later provided an analysis of the amplification of privacy thanks to the combined use of shuffling and local differential privacy showing that the shuffling model of privacy is able to strike a middle ground between the totally decentralized but somewhat sample inefficient *local* model and the centralized but more sample efficient central model of privacy. It is currently unclear whether it is possible to achieve some form of privacy/utility trade-off between these two models in the contextual bandit setting.

In this paper, we investigate the linear contextual bandit problem under the shuffle model of privacy, for the first time considering this privacy model in contextual bandit. Compared to the standard shuffle model (e.g., in supervised learning), there are several challenges introduced by the sequential nature of the problem. First, the shuffler is executed continuously and not only once as normally considered in supervised learning. Second, the number of samples available grows with time and depends on the decisions of the learning agent. This makes the design of the algorithm non-trivial, in particular for efficiently trading-off privacy amplification and regret.

We address these challenges in two ways. First, we carefully design separate asynchronous batch schedules for the shuffler and the bandit algorithm (i.e., LinUCB); here, batching at the shuffler is used to ensure privacy, and not just improved regret. Second, we leverage the martingale structure of the problem to analyze these batching schedules and provide privacy guarantees on the entire sequence of outputs generated by the shuffler and bandit algorithm. We summarize our main contributions as follows (see also Tab. 3.2.1):

- If there is no adversary in between the shuffler and the algorithm (i.e., the communication channel is secure), we show that it is possible to achieve a regret bound of $\widetilde{\mathcal{O}}\left(dT^{2/3}/\varepsilon^{1/3}\right)$ with a fixed batch size for the shuffler and dynamic batch for the bandit algorithm.
- In the case of adversary in between the shuffler and the users, our algorithm achieves a regret bound of $\widetilde{\mathcal{O}}\left(T^{3/4}/\sqrt{\varepsilon}\right)$ with a fixed batch size for the shuffler and dynamic batch for the bandit algorithm.

Algorithm	Regret Bound	Privacy Model	
		Joint DP	Local DP
Shariff and Sheffet (2018)	$\widetilde{\mathcal{O}}\left(T^{1/2}/\varepsilon^{1/2}\right)$	(ε, δ)	N/A
Zheng et al. (2020)	$\widetilde{\mathcal{O}}\left(T^{3/4}/\varepsilon^{1/2}\right)$	(ε, δ)	(ε, δ)
Our Cor. 1 (*LDP optimization*)	$\widetilde{\mathcal{O}}\left(T^{3/4}/\varepsilon^{1/2}\right)$	$(\frac{\varepsilon^{3/2}}{T^{1/4}}, \delta)$	$(\varepsilon, 0)$
Our Cor. 2 (*regret optimization*)	$\widetilde{\mathcal{O}}\left(T^{2/3}/\varepsilon^{1/3}\right)$	(ε, δ)	$(\varepsilon^{2/3}T^{1/6}, 0)$

Table 3.2.1: Regret and privacy for algorithms in *linear contextual bandits* for $T \geq 1/(27\varepsilon)^4$.

3.2.1 The Shuffle Model in Linear Contextual Bandits

We consider linear contextual bandit problems, where rewards are linearly representable in the features, i.e., for any feature vector $x_{t,a}$, it writes as $r(x_{t,a}) = \langle x_{t,a}, \theta^\star \rangle$, where $\theta^\star \in \mathbb{R}^d$ is unknown. We do not pose any assumption on the context generating process but we rely on the following standard assumptions.

Assumption 12. *There exist $S > 0$ and $L > 0$ such that $\|\theta^\star\|_2 \leq S$ and, for all time $t \in [T]$, arm $a \in [K]$, $\|x_{t,a}\|_2 \leq L$. Furthermore, the noisy reward is $r_t = \langle x_{t,a}, \theta^\star \rangle + \eta_t \in [0,1]$ with η_t being σ-subGaussian for some $\sigma > 0$. These parameters, L, S and σ, are known.*

The performance of the learner $\mathfrak{A}$ over T steps is measured by the regret $R_T = \sum_{t=1}^{T} r(x_{t,a_t^\star}) - r(s_{t,a_t})$, which represents the cumulative difference between playing the optimal action $a_t^\star = \arg\max_{a\in[K]} r(x_{t,a})$ and a_t the action selected by the algorithm.

3.2.1.1 Shuffle-model in Contextual Bandits

In this subsection, we introduce the generic shuffle-model for contextual bandit, inspired by the ESA model. In Sec. 3.2.2, we will provide the details for instantiating it in linear contextual bandits. In the standard shuffle model, a shuffler is introduced in between the data and the algorithm. The shuffler enables privacy amplification by permuting information of l users. The larger the batch, the higher the privacy amplification but also the degradation of the utility (see e.g., Cheu et al., 2019), leading to some fundamental trade-off between privacy amplification and utility loss. In online learning, we observe users sequentially and it is natural to assume that, in order to achieve privacy amplification, the shuffler builds a batch of consecutive users before communicating with the bandit algorithm. The bandit algorithm can then behave synchronously or asynchronously w.r.t. the shuffler. In other words, it can update its internal statistics with the same frequency of the shuffler or use an independent batch schedule.

More formally, the shuffle-model for contextual bandit is described by the following interaction protocol (see also Fig. 3.2.1). At each time $t \in [T]$,

❶ A new user x_t receives model information from the bandit algorithm (e.g., estimated rewards and confidence intervals) that are used to *locally* compute the action to play. Then, the user plays the prescribed action a_t which generates the associated reward r_t.

❷ The user sends its own privatized version of the data $\mathcal{M}_{\mathrm{LDP}}(x_{t,a_t}, r_t)$ to the shuffler. This new data is added to the shuffler batch $B^S_{k_t} := \bigcup_{i=t_{k_t^S}}^{t} \left\{\mathcal{M}_{\mathrm{LDP}}(x_{i,a_i}, r_i)\right\}$, where k_t^S denotes the shuffler batch at time t and t_k is the starting time of batch k.

❸ The bandit algorithm queries statistics from the shuffler. If the shuffler is ready to send data (e.g., enough samples has been collected for privacy amplification), it computes a statistic u on a permutation of the data (i.e., $u(\sigma(B^S_{k_t}))$) and sends it to the bandit algorithm. Otherwise no information is provided. The bandit algorithm adds the new statistic to its batch (i.e., $B^A_{k_t^A} := \bigcup_{i=t_{k_t^A}}^{t} \left\{u(\sigma(B^S_{k_i^S}))\right\}$) and may then decide to update the model as soon as data is received (i.e., synchronously) or use an independent batch schedule (i.e., asynchronous).

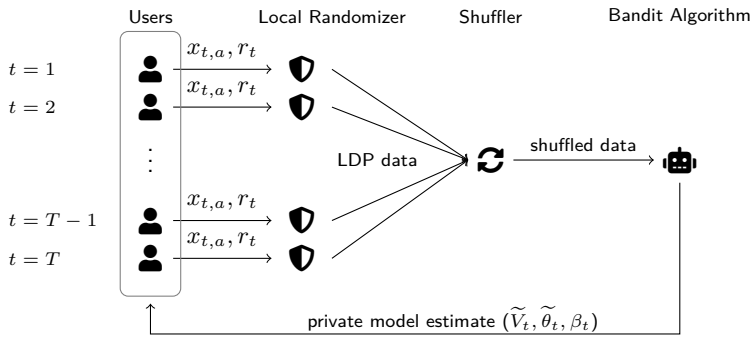


Figure 3.2.1: Illustration of the shuffle model for linear contextual bandits.

The objective is to minimize the (pseudo) regret and simultaneously guarantee privacy of the data and of the statistics. To this extent, we assume all users (including the shuffler and the bandit algorithms) behaves in an *honest but curious* manner (Oded, 2009), i.e., the users and the algorithm behaves as prescribed by the protocol. We consider different threat models for privacy, including an adversary in between a the user and the shuffler, b the shuffler and the bandit algorithm, and c the bandit algorithm and the user. We will show that different privacy/regret guarantees can be achieved in the different settings.

3.2.2 Shuffle Model with Fixed-Batch Shuffler

In this subsection, we provide an instantiation of the shuffle model for linear contextual bandit. We base our algorithm on the non-private low-switching LinUCB (Abbasi-Yadkori et al., 2011), that incrementally builds an estimate $\widehat{\theta}_j$ of

Algorithm 12: SBLB

Input: LDP parameter: ε_0, privacy parameters: ε, δ_0, regularizer: λ, context bound: L, failure probability: δ, low switching parameter: η, encoding parameter: m, dimension: d, fix batch size: ℓ

Initialize $j^S = j^A = 0$, $\widetilde{\theta}_0 = 0$, $\widetilde{V}_0 = \lambda I_d$ and $p = 2\big(\exp\left(\frac{2\varepsilon_0}{md(d+3)}\right) + 1\big)^{-1}$

for $t = 0, 1, \dots$ **do**

c Communication with the user

User receives $\widetilde{\theta}_{j^A}$, $\widetilde{V}_{j^A}$ and β_{j^A} and selects $a_t \in \arg\max_{a\in[K]} \langle x_{t,a}, \widetilde{\theta}_{j^A}\rangle + \beta_{j^A} \| x_{t,a}\|_{\widetilde{V}_{j^A}^{-1}}$

Observe reward r_t and compute private statistics $(\widetilde{b}_t, \widetilde{w}_t) = \mathcal{M}_{\mathrm{LDP}}(x_{t,a_t}, r_t, L, \varepsilon_0, m)$ (Alg. 21)

a Communication with the shuffler

$B^S_{j^S} = B^S_{j^S} \cup (\widetilde{b}_t, \widetilde{w}_t)$

if $|B^S_{j^S}| = l$ **then**

Set $t_{j^S+1} = t$, compute a permutation σ of $[\![t_{j^S}+1, t_{j^S+1}]\!]$ and compute aggregate statistics

$$\forall i \le d, k \le i, \qquad Z_{j^S,i} = \sum_{n=1}^{l}\sum_{q=1}^{m} \tilde{b}_{\sigma(n),i,q} \quad \text{and} \quad U_{j^S,i,k} = \sum_{n=1}^{l}\sum_{q=1}^{m} \widetilde{w}_{\sigma(n),i,k,q}$$

Set $U_{j^S,i,k} = U_{j^S,k,i}$, $B_{j^S+1} = \emptyset$ and $j^S = j^S + 1$

b Communication with the bandit algorithm

Receives (Z_{j^S-1}, U_{j^S-1}) and compute candidate statistics

$$\widetilde{B}_{j^A+1} = \widetilde{B}_{j^A+1} + \frac{Z_{j^S-1}}{m(1-p)} - \frac{l^S}{2(1-p)}$$
$$\widetilde{V}_{j^A+1} = \widetilde{V}_{j^A+1} + \frac{U_{j^S-1}}{m(1-p)} - \frac{l^S}{2(1-p)} + 2(\lambda_{j^A+1} - \lambda_{j^A})I_d$$

if $det(\widetilde{V}_{j^A+1}) \ge (1+\eta) det(\widetilde{V}_{j^A})$ **then**

Compute $\tilde{\theta}_{j^A+1} = \frac{1}{L}\widetilde{V}^{-1}_{j^A+1}\widetilde{B}_{j^A+1}$

Set $t_{j^A+1} = t$, β_{j^A+1} and λ_{j^A+1} as in Eq. (3.13) and Eq. (3.14)

Set $j^A = j^A + 1$, $\widetilde{B}_{j^A+1} = \widetilde{B}_{j^A}$ and $\widetilde{V}_{j^A+1} = \widetilde{V}_{j^A}$

the unknown parameter $\theta^\star$. Since the algorithm leverages sum of statistics received from the users, we consider the binary sum mechanism inspired by (Cheu et al., 2019) as building block for achieving privacy in the shuffle model. While this scheme allows us to obtain standard LDP guarantees on users information, the shuffler is responsible to provide privacy amplification via batching and shuffling. The main challenge is to combine these elements with the low-switching scheme of LinUCB. As we will explain later, adaptive batching at the level of LinUCB is not for computational efficiency but it is rather fundamental for obtaining a good privacy/regret trade-off.

3.2.2.1 Algorithmic Design

In this subsection, we provide a full description of the Shuffle-Batched Linear Bandit (SBLB) algorithm. Intuitively, the algorithm relies on a shuffler with fixed batch size to achieve privacy amplification from LDP data, and a variation of LinUCB with dynamic batch schedule based on the determinant condition. The pseudo-code is reported in Alg. 12.

❶ Action Selection. At each time t, the user x_t receives, from the bandit algorithm, an estimate of the model composed by a parameter $\widetilde{\theta}_{k_t^A} \in \mathbb{R}^d$, a design matrix $\widetilde{V}_{k_t^A} \in \mathbb{R}^{d\times d}$ and confidence width $\beta_{k_t^A}$. Notice that these are parameters computed at the beginning of the batch k_t^A of the bandit algorithm. Then, the action is selected by maximizing the following standard optimistic problem:

$$a_t \in \arg\max_{a\in[K]} \left\{ \langle x_{t,a}, \widetilde{\theta}_{k_t^A}\rangle + \beta_{k_t^A} \| x_{t,a}\|_{\widetilde{V}^{-1}_{k_t^A}} \right\}$$

where β_t is the size of the confidence ellipsoid, defined in Lem. 21, which roughly scales as $\widetilde{O}\left(t_{k_t^A}^{1/4}\right)$. Note that it is possible to directly access the features $x_{t,a}$ of the user since this computation happens locally. The action is played and a reward r_t is observed.

❷ Local Privacy and Shuffler. Users' information is then protected through a local private mechanism $\mathcal{M}_{\text{LDP}}$. As noticed in (Shariff and Sheffet, 2018), only the information required by the algorithm, to compute $\widetilde{\theta}$ through ridge regression and the associated confidence interval, must be privatized. We are thus interested in privatizing the quantities $x_{t,a_t} r_t$ and $x_{t,a_t} x_{t,a_t}^{\mathsf{T}}$. To obtain LDP quantities, we leverage a variation of the private mechanism introduce by Cheu et al. (2019). We independently privatize each component of the vector $x_{t,a_t} r_t$ and of the upper triangular part of the matrix $x_{t,a_t} x_{t,a_t}^{\mathsf{T}}$, the rest follows from the symmetric structure. Each entry is normalized to $[0,1]$ and approximated by a truncated 0/1-bit representation, which length is controlled by the parameter $m \in \mathbb{N}^*$. The full procedure is reported in Alg. 21.

The shuffler receives the privatize data $\mathcal{M}_{\text{LDP}}(x_{t,a_t}, r_t)$ and adds it to the current batch. The role of the shuffler is to provide additional privacy by sending data in a random order compared what it has received. At a high-level this provide an additional privacy guarantee because it breaks the link between a given user and its data. Indeed for an algorithm receiving data from the shuffler, the t-th row of data has little chance to come from user t. If the shuffler has access to a batch of size l, it can provide a privacy amplification of level $l^{-1/2}$ (see e.g., Cheu et al., 2019, Thm. 5.4). Ideally, we would like to shuffle all the data at each time t, achieving a privacy amplification of $t^{-1/2}$. However, this approach would not provide enough privacy due to the fact an adversary would have multiple observations of the same data, thus greatly decreasing the advantage of using the shuffling mechanism. To avoid this issue, we need to force the shuffler to use batches and discard samples after each batch. Let's denote by l^S the fix batch size of the shuffler. At time t, if the batch $B_{k_t}^S$ is of size l^S, the shuffler permutes the data and compute the statistics required by the bandit algorithm. To compute those statistics, the shuffler uses a secure and trusted third-party different that the shuffler. This third-party is assumed to be secure with for example the use of encrypted communication between the shuffler and it, like in (Cheu et al., 2019). When $|B_{k_t}^S| < l^S$, the shuffler do not provide any information to the bandit algorithm. The shuffling setting is not fundamentally different than the LDP one, but it allows to achieve a large gain in privacy in the high data regime from multiple users. Shuffling allows to achieve better privacy guarantees and, overall, it improves the standard LDP protocol with virtually no cost.

❸ Model Estimation (the bandit algorithm). As last step, the bandit algorithm queries new data to the shuffler which replies only if the batch is full. If no data is received, the bandit algorithm does nothing. Otherwise, the bandit algorithm receives summary statistics $Z_{k_t^S}$ and $U_{k_t^S}$ corresponding to the sum over the shuffled batch $B_{k_t^S}^S$ of the LDP data associated to xr and xx^{T}. The algorithm could behave synchronously with the batch schedule of the shuffler and update the model by updating the design matrix $\widetilde{V}_{k_t^S+1}$ and parameter $\widetilde{\theta}_{k_t^S+1}$. However, this behavior would lead to a worse privacy/regret trade-off than an asynchronous data-adaptive schedule. Although it is possible to achieve the same regret bound in non-private settings with static and dynamic batch schedules, in the private case it is no more the case because of required inflation of the confidence intervals by a factor $t^{1/4}$ to deal with concentrations of private statistics. In App. 3.B.3, we provide a more formal support to this claim.

As a consequence, we shall leverage the determinant-based condition introduced by Abbasi-Yadkori et al. (2011). Upon receiving the data at time t, the bandit algorithm has access to the following set of private statistics $\left\{(Z_i, U_i), i \in [k_t^S]\right\}$, which is further divided into batches of various lengths. Denote by $j = k_t^A$ the bandit batch at time t with associated parameters $\widetilde{V}_j$, $\widetilde{B}_j$ and $\widetilde{\theta}_j$ computed at the beginning of the batch. Then, we denote by $\widetilde{V}_t$ the new design matrix obtained by updating the matrix $\widetilde{V}_j$ with all the statistics received from the shuffler after t_j. If $\det(\widetilde{V}_t) \geq (1+\eta)\widetilde{V}_j$, then a new batch is started and the model is updated, i.e., $\widetilde{\theta}_{j+1} = \frac{1}{L}\widetilde{V}_{j+1}^{-1}\widetilde{B}_{j+1}$ is computed through ridge-regression. In a LinUCB fashion, the last step for the algorithm is to compute the size of a confidence intervals around $\widetilde{\theta}_{j+1}$ containing the true parameter $\theta^\star$. Contrary to the non-private setting (Abbasi-Yadkori et al., 2011), the algorithm uses wider confidence intervals to account for the noise added to ensure privacy. This increase is quite significant as the confidence intervals grow at a $t^{1/4}$ rate compared to $\log(t)$ in the non private setting. Refer to Lem. 21 for the explicit definition.

3.2.3 Analysis of The Shuffle Model with Fixed-Batch Shuffler

In this subsection, we provide the privacy and regret guarantees of SBLB. We first begin to describe which privacy guarantees are attainable in the different attack scenarios outlined in the introduction. Then we show how the regret of SBLB is impacted by the these attack models.

For sake of clarity, we recall the parameters that regulates the privacy/regret analysis of our algorithm. The first parameter ε_0 regulates the level of local differential privacy introduced by the local randomizer $\mathcal{M}_{\text{LDP}}$. However, to simplify the analysis, we often use the alternative parameter $p := 2\left(\exp\left(\frac{2\varepsilon_0}{md(d+3)}\right)+1\right)^{-1}$ derived from ϵ_0 (see Alg. 21). The other two parameters (ε, δ_0) controls the level of joint differential privacy that SBLB should attain.

3.2.3.1 Privacy Analysis of SBLB

As discussed in Sec. 3.2.1, the shuffling model encompasses all the multiple scenarios in which the privacy of users can be threatened.

a Compromised communication between the user and the shuffler. In the first and most harmful scenario, the communication between the users and the shuffler is not secured and the data can be observed by an adversary. This is the standard LDP setting in linear contextual bandit. In this case, the use of the local randomizer $\mathcal{M}_{\text{LDP}}$ guarantees that the data sent by the user to the shuffler are ε_0-LDP. That is to say the most stringent privacy guarantees in the differential privacy model.

Proposition 7 (LDP guarantee). *For any $\varepsilon_0 > 0$ and $m \in \mathbb{N}^\star$, $\mathcal{M}_{LDP}(.,.,\varepsilon_0, L, m)$ is ε_0-LDP.*

This particular scenario corresponds to a decentralized setting where the users do not trust the algorithm or the communication channel between them to be secure and they have to protect the privacy of their data at a individual level, that is to say to guarantee that the data sent could have been sent by anyone else. This setting (i.e., the "pure" LDP scenario) is also the one studied in (Zheng et al., 2020). We will show that we can recover their result when we want to guarantee the highest level of LDP privacy. However, at the cost of sacrificing a portion of LDP level, we can obtain a better regret bound, closing the gap with the less stringent JDP setting.

b Compromised communication between the shuffler and the bandit algorithm. In another privacy loss scenario, an adversary can observe the same data as the bandit algorithm. Stated otherwise, the adversary has access to the output of the shuffler. In that case, SBLB is still ε_0-LDP but stronger differential privacy guarantees can be achieved thanks to privacy amplification. In this scenario, the adversary observes the different outputs of the shuffler, that are statistics computed on a number of different users. The question, in the differential privacy setting, is whether it is possible to know that one particular user (i.e., user's data) was involved in the computation of those statistics.

Tenenbaum et al. (2021) studies a weaker version of this question in the multi-armed bandit setting where an adversary *only observes the output of the shuffler for one time step*, while we focus on the more challenging case where the adversary observes all the history. Technically, this is the same difference as ensuring event-level privacy in the continual observation model compared to a differential privacy on a single query. Note that it would be possible to obtain a better regret bound if we consider the adversary model in (Tenenbaum et al., 2021) since a smaller level of privacy is required (see Remark 2).

The complicated aspect is to guarantee that the whole sequence of M_S vectors and matrices $(Z_{j^S}, U_{j^S})_{j^S=1}^{M_S}$ is private, and not a single output at a given time. This issue is solved by leveraging batching. Formally, we can show in this scenario that the sequence $(Z_{j^S}, U_{j^S})_{j^S}$ is $(\varepsilon, \delta_0 + \delta)$-DP for any $\delta_0, \delta \in (0,1)$ and $\varepsilon \in (0,1)$.

Theorem 8. *For any $\varepsilon \in (0,1)$, $\delta_0, \delta \in (0,1)$, encoding parameter m and LDP parameter $\varepsilon_0 > 0$, let $p = 2(e^{2\varepsilon_0/md(d+3)}+1)^{-1}$. Then if $l^\star$, the length of a shuffler batch, satisfies $l^\star p \geq 14\log(8mT/\delta_0)$ and:*

$$\sqrt{\left(2 + \left(\frac{\varepsilon l^\star}{32d(d+3)\log(8mT/\delta_0)\sqrt{2T\ln(2T/\delta_0)}}\right)^2\right)^2 - 4} \geq 1 - 2p + 2\sqrt{\frac{2\log(2mT/\delta_0)}{l}} + \left(\frac{\varepsilon l^\star}{32d(d+3)\log(8mT/\delta_0)\sqrt{2T\ln(2T/\delta_0)}}\right)^2, \tag{3.9}$$

the sequence $(Z_{j^S}, U_{j^S})_{j^S}$ is central $(\varepsilon, \delta_0 + \delta)$-DP.[7]

The result of Thm. 8 is a consequence of the advanced composition theorem (Dwork et al., 2010a). Indeed, thanks to shuffling, for any batch j^S, the statistics (Z_{j^S}, U_{j^S}) are $\left(\frac{\sqrt{\varepsilon(1-p)}}{T^{1/4}}, \frac{\delta_0\varepsilon}{\sqrt{T}(1-p)}\right)$-DP, since the batch length l is approximately $\frac{\sqrt{T}(1-p)}{\varepsilon}$. As a consequence, when composing them together we get that the central DP level of each batch is $\widetilde{\mathcal{O}}\left(\varepsilon\sqrt{\frac{l^\star}{T}}\right)$. Therefore by advanced composition, since we have a total number of batches $M_S \approx \sqrt{T}$, the total privacy over the sequence of $(Z_{j^S}, U_{j^S})_{j^S}$ is of order $\widetilde{\mathcal{O}}\left(\varepsilon\sqrt{\frac{l^\star}{T}} \times \sqrt{\frac{T}{l^\star}}\right)$ that is to say of order $\widetilde{\mathcal{O}}(\varepsilon)$.

c Compromised Communication between the bandit algorithm and the users. Similarly to Shariff and Sheffet (2018), in the final scenario we consider, an adversary can observe the same data coming from SBLB as the users, i.e., the stream of estimates $(\widetilde{\theta}_{k_t^A}, \widetilde{V}_{k_t^A}, \beta_{k_t^A})_{t\in[T]}$. Recall that the bandit algorithm uses a dynamic batch schedule based on the determinant

[7] We provide the definition of central DP in Def. 4 in App. 3.B.2. Note that the concept of central DP is at the core for proving JDP results, in fact thanks to Claim 7 in (Shariff and Sheffet, 2018) having a sequence $(\widetilde{V}_t, B_t)_t$ is (ε, δ)-DP implies that a bandit algorithm based on this sequence is (ε, δ)-DP.

technique and it is asynchronous w.r.t. the shuffler. This leads to a number of bandit batches roughly of order $\log(T)$. While we have to guarantee privacy on a smaller number of element ($\log(T)$ compared to $\sqrt{T}$ in the shuffler), we are technically limited by the former scenario b. As shown in Prop. 8, SBLB is $(\varepsilon, \delta_0 + \delta)$-JDP w.r.t. the sequence $(\widetilde{\theta}_{j^A}, \widetilde{V}_{j^A}, \beta_{j^A})_{j^A}$ since $(Z_{j^S}, U_{j^S})_{j^S}$ is $(\varepsilon, \delta_0 + \delta)$-DP.

Proposition 8 (JDP guarantee). *For any $\varepsilon \in (0,1)$, $\varepsilon_0 > 0$, $\delta, \delta_0 \in (0,1)$, $m \in \mathbb{N}^\star$, selecting the length of a shuffler like in Thm. 8 ensures that the sequence of $(\widetilde{\theta}_{j^A}, \widetilde{V}_{j^A}, \beta_{j^A})_{j^A}$ is $(\varepsilon, \delta + \delta_0)$-DP. In other words* SBLB *is $(\varepsilon, \delta + \delta_0)$-JDP.*

Since we are directly leveraging advance composition, we cannot get any privacy amplification when we consider b and c together. Scenario c is indeed the most stringent adversary model in the shuffle-model, limiting the gain in the privacy/regret we can obtain compared to the pure LDP setting. It is however possible to achieve a better privacy/utility trade-off when considering only scenario c (and not b), but we believe it is a much weaker attack scenario. In both scenarios, b and c, the objective is to ensure Joint Differential Privacy. Model c deals with the issue when attackers can submit potentially false contexts to the bandit algorithm and observes the action recommended with the objective to learn the context/reward of a target user. Guaranteeing that this task is difficult is the objective of Joint Differential Privacy. In this paper, we use a deterministic bandit algorithm therefore in terms of privacy scenarios b and c are the same (thanks to the post-processing lemma). However, one could think of using a randomized algorithm and therefore improve the privacy of the whole scheme.

Remark 1. *In online learning, JDP and central-DP are not equivalent definitions. A DP constraint on the actions selected implies that the probability of selecting any action is strictly positive thus hindering the algorithm to select the optimal action. Indeed, as noted in (Shariff and Sheffet, 2018) (see Claim 13) any central-DP linear contextual bandit algorithm must incur linear regret, whereas in the weaker definition of JDP it is possible to attain a sublinear regret. The fact that the computation of the action is local is necessary to achieve a sublinear regret.*

3.2.3.2 Regret Analysis of SBLB

In the previous subsection, we stated several privacy guarantees of SBLB with different attack models. We shall now show the impact of those privacy guarantees on the regret. As mentioned, shuffling allows to regulate the level and type of privacy desired by trading-off the regret guarantee. In SBLB, this trade-off is regulated by the parameter ε_0 which has impact on all the main elements in the privacy and regret analysis (e.g., batch size, privacy p, etc.).

The first result we provide is a validation of our algorithm. The following proposition shows that SBLB recovers the results in (Zheng et al., 2020), providing the highest possible *local* DP level at the expense of the regret bound.

Corollary 1. *For any $\varepsilon_0 > 0$ and $\delta \in (0,1)$ then choosing $\varepsilon = \sqrt{\exp(\varepsilon_0) - 1}$ and $\delta_0 = \delta$ we have that* SBLB *is ε_0-LDP and with probability at least $1 - \delta$ is bounded by:*

$$R_T \leq \widetilde{\mathcal{O}}\left(\frac{T^{3/4}\sqrt{e^{\varepsilon_0}+1}}{\sqrt{e^{\varepsilon_0}-1}} + \frac{\log(T)\,(e^{\varepsilon_0}+1)^2}{4} + \frac{\sqrt{T}}{\sqrt{e^{\varepsilon_0}-1}}\right) \tag{3.10}$$

On the other hand, Cor. 2 shows that SBLB interpolates between the regret of (Zheng et al., 2020) (LDP setting studied under scenario a) and (Shariff and Sheffet, 2018) (JDP setting studied under scenario c). The structure of the shuffle-model requires to also consider scenario b that, as mentioned before, poses the highest restriction on the regret bound we can achieve.

Corollary 2. *For any $\varepsilon \leq \frac{1}{27T^{1/4}}$ and $\delta, \delta_0 \in (0,1)$, the choices of $\eta = 0.5$, $\lambda = \sqrt{T}$, $m = 1$ and $\varepsilon_0 = \frac{d(d+3)}{2}\ln\left(\frac{2}{1-\varepsilon^{2/3}T^{1/6}} - 1\right)$ ensures that with probability at least $1 - \delta$ the regret of* SBLB *is bounded by:*

$$R_T \leq \frac{4T^{2/3}}{\varepsilon^{1/3}}\left(S + d + \frac{1}{T^{1/4}}\widetilde{\mathcal{O}}(1)\right), \tag{3.11}$$

where $\widetilde{\mathcal{O}}(\cdot)$ hides poly-log factor (in T, δ, δ_0) and polynomial factors (in d, L). In addition SBLB *is $(\varepsilon, \delta_0 + \delta)$-JDP and $6d^2\varepsilon^{2/3}T^{1/6}$-LDP.*

One may be confused as to the utility of this result as it shows a worse regret bound compared to the one of Theorem 7. The improvement comes from the fact that the best known regret for LDP contextual linear bandit scales as $\mathcal{O}\left(T^{3/4}/\sqrt{\varepsilon}\right)$ (Han et al., 2021) (without any further assumption) whereas for LDP RL, we showed that the regret scales as $\mathcal{O}\left(T^{1/2}/\varepsilon\right)$. It is suspected that the difference between the two results comes from the finiteness of the state space in RL but not for the contexte space in linear contextual bandits.

For the complete regret bound refer to the end of App. 3.B.2. This shows that the regret bound of SBLB is of order $\mathcal{O}\left(dT^{2/3}/\varepsilon^{1/3}\right)$, while being (ε, δ)-JDP and approximately $(2\varepsilon^{2/3}T^{1/6}, 0)$-LDP. As expected, this indicates the regret bound can be improved by sacrificing some level of LDP. However, the $\sqrt{T}$ regret bound of (Shariff and Sheffet, 2018) cannot be

recovered directly. While the search for a better upper-bound or a lower-bound is an interesting future direction, we think it would be hard to match such JDP minimax result. Indeed, shuffling allows to interpolate between JDP (where the best minimax bound is $\sqrt{T}$) and LDP (where the best known upper bound is $T^{3/4}$). Since we will always have a non-zero LDP level of privacy in the considered ESA shuffle model, we believe it is almost impossible to achieve $\sqrt{T}$ regret in particular.[8]

Proof Sketch The proof of this theorem is presented in details in App. 3.B.2. To understand this result however we present how we build the confidence intervals around the parameter $\theta^\star$. As noticed in (Shariff and Sheffet, 2018), the estimator $\widetilde{\theta}_j$ is the result of a ridge regression computed by a design matrix regularized by a regularizer which is a function of the time. Therefore in order to apply Prop. 4 in (Shariff and Sheffet, 2018) we need to ensure that our estimator $\widetilde{V}_j$ of the design matrix, $\sum_t x_{t,a_t} x_{t,a_t}^\intercal$, is unbiased and to bound with high probability the deviation with respect to the design matrix. We also need the same type of guarantees with respect to the vector $\widetilde{B}_j$ and $\sum_t r_t x_{t,a_t}$.

Computation of our Estimators. The bandit algorithm receives the estimate (Z_{j^S}, U_{j^S}) from the shuffler but given the data those estimates are biased. For a couple of vector and reward, x and r, let us note $\mathcal{M}_{\text{LDP}}(x,r) = (b,w)$, so that

$$\mathbb{E}\left(b_{k,q} \mid x, r\right) = \frac{p}{2} + (1-p)\left[\mathbb{1}_{\{q<\lceil rx_k m\rceil\}} + \mathbb{1}_{\{q=\lceil rx_k m\rceil\}}(mrx_k - \lceil rx_k m\rceil + 1)\right]$$
$$\mathbb{E}\left(w_{k,l} \mid x, r\right) = \frac{p}{2} + (1-p)\left[\mathbb{1}_{\{q<\lceil x_l x_k m\rceil\}} + \mathbb{1}_{\{q=\lceil x_l x_k m\rceil\}}(mx_l x_k - \lceil x_l x_k m\rceil + 1)\right]$$

for all $k,l \le d$ and $q \le m$. Therefore, we introduce a debiased estimator for computing the estimators of SBLB, written as follows:[9]

$$\widetilde{V}_{j^A} = \sum_{t=1}^{t_{j^A}} \frac{x_{t,a_t} x_{t,a_t}^\top}{2L^2} + H_{j^A} + \lambda_{j^A} I_d \quad \text{and} \quad \widetilde{B}_{j^A} = \sum_{l=1}^{t_{j^A}} \frac{r_l x_{l,a_l}}{2L} + h_{j^A}, \tag{3.12}$$

where, for all batches, $H_{j^A} + \lambda^{j^A} I_d$ is with high probability a symmetric positive definite matrix decomposed as the sum of zero mean noise and a regularization λ_{j^A}, and h_{j^A} is a vector of zero mean noise. Both noises are due to the noise introduced in by the local randomizer $\mathcal{M}_{\text{LDP}}$. In addition, as we show in App. 3.B.2 controlling the eigenvalues of the regularizer $H_{j^A} + \lambda^{j^A} I_d$ and the noise h_{j^A} is bounded roughly by $\sqrt{t_{j^A}}$. Therefore thanks to Prop. 4 in (Shariff and Sheffet, 2018), the following proposition holds.

Lemma 21 (Confidence Ellipsoid). *For any $\delta \in (0,1)$, $\varepsilon_0 > 0$, $p = \frac{2}{e^{2\varepsilon_0/(md(d+3))}+1}$ and $\lambda > 0$, we have with probability at least $1-\delta$ that:*

$$\begin{aligned}\forall j^A \le M_S, \qquad \|\theta^\star - \widetilde{\theta}_{j^A}\|_{\widetilde{V}_{j^A}^{-1}} \le \beta_{j^A} := \sigma\sqrt{8\log\left(\frac{2t_{j^A}}{\delta}\right) + d\log\left(3 + \frac{t_{j^A}L^2}{\lambda_{j^A}}\right)} + S\sqrt{3\lambda_{j^A}} \\ + \frac{d}{\sqrt{\lambda_{j^A}}}\left(2\sqrt{p\left(1-\frac{p}{2}\right)t_{j^A} m \log\left(\frac{2t_{j^A}}{\delta}\right)} + \frac{8\log(2t_{j^A}/\delta)}{3} + \frac{\sqrt{8}}{m}\sqrt{t_{j^A}\log\left(\frac{2t_{j^A}}{\delta}\right)}\right)\end{aligned} \tag{3.13}$$

where $M_S = {}^T/_{l^\star}$ is the number of shuffler batch and for all $j^A \le M_S$,

$$\lambda_{j^A} = \frac{\sqrt{8t_{j^A}\ln(2t_{j^A}/\delta)}}{m} + \frac{2\sqrt{8t_{j^A}\ln(2t_{j^A}/\delta)}}{(1-p)\sqrt{m}} + \lambda \tag{3.14}$$

Given the definition of the confidence ellipsoid above, we can analyze the regret using a standard regret analysis for algorithms using the optimism-in-the-face-of-uncertainty principle. For a generic set of privacy parameters ε_0, ε and δ_0, the regret bound of SBLB is given in the following theorem.

Theorem 9. *For any $\delta, \delta_0 \in (0,1)$, $\varepsilon, \varepsilon_0 \in (0,1)$ and $T \ge 1$, let $p = 2(e^{2\varepsilon_0/md(d+3)}+1)^{-1}$ then with probability at least $1-\delta$, the regret of Alg. 21 is bounded by:*

[8] Note that in multi-armed bandit (MAB), it is possible to achieve a minimax regret bound of order $\sqrt{T}$ both in central DP and LDP (Ren et al., 2020; Basu et al., 2019). We think this is an important aspect leveraged by Tenenbaum et al. (2021) for shuffling in MAB. In addition, as already mentioned, they considered a weaker attack model.

[9] Note that this is an alternative but equivalent form to the one used in Alg. 12.

- *If* $p^2(1-p) \leq \frac{7T^{-1/2}\varepsilon}{64\sqrt{2\ln(2T/\delta_0)}d(d+1)}$:

$$R_T \leq \frac{2\sqrt{3}(S+md)T^{3/4}}{\sqrt{1-p}}\sqrt{(1+\eta)\log\left(1+\frac{T}{d\lambda}\right)} + \frac{dLm}{\sqrt{\lambda}}\left(1+\frac{d^{3/2}\log\left(\frac{L^2T}{d}+\frac{16\sqrt{T}\log(2T/\delta)}{(1-p)}\right)^{3/2}}{\log(1+\eta)}\right)\frac{14\log(8mT/\delta_0)}{p^2} \tag{3.15}$$

- *If* $p^2(1-p) \geq \frac{7T^{-1/2}\varepsilon}{64\sqrt{2\ln(2T/\delta_0)}d(d+1)}$:

$$R_T \leq \frac{2\sqrt{3}(S+md)T^{3/4}}{\sqrt{1-p}}\sqrt{(1+\eta)\log\left(1+\frac{T}{d\lambda}\right)} + \frac{264}{\sqrt{\lambda}}\sqrt{2}d^3\log\left(\frac{8mT}{\delta_0}\right)^{3/2}Lm\left(1+\frac{d^{3/2}\log\left(\frac{L^2T}{d}+\frac{16\sqrt{T}\log(2T/\delta)}{(1-p)}\right)^{3/2}}{\log(1+\eta)}\right)\frac{\sqrt{T}(1-p)}{\varepsilon} \tag{3.16}$$

The first term of the regret in Thm. 9 highlights the regret coming from the local privacy guarantees whereas the second term is coming from the mismatch between the batch of the shuffler and the batch of the bandit algorithm. Indeed, when the bandit algorithm updates its batch it means that during the last shuffler batch the determinant condition was satisfied at some point during the shuffler batch. However, the impact on the regret during this shuffler batch can only be bounded by the length of a shuffler batch times the maximum reward possible. But given Thm. 8 the length of a shuffler batch scales with $\widetilde{\mathcal{O}}\left(\sqrt{T}/\varepsilon\right)$. Hence the final regret scales with $\widetilde{\mathcal{O}}\left(T^{3/4}/\sqrt{1-p}+\sqrt{T}/\varepsilon\right)$. As a consequence, Cor. 1 and Cor. 2 are obtained by optimizing for the highest privacy level and smaller regret bound, respectively.

Remark 2. *A better regret bound can be obtained in the setting of (Tenenbaum et al., 2021), where the adversary only observes the output of the shuffler for one time step. In particular, this allows to improve the privacy analysis and obtain a generic regret bound of order* $\mathcal{O}\left(T^{3/4}/\sqrt{1-p}+\log(T)/\varepsilon^2\right)$ *that once optimized leads to a regret bound of* $T^{3/5}/\varepsilon^{2/5}$ *which is much closer to the best JDP regret bound. However, we think this setting is less practical than the one considered in this paper.*

3.2.4 Potential Extensions

We introduced SBLB, an algorithm for linear contextual bandits that achieves a trade-off between joint and local differential privacy. Our algorithm is a variant of batched LinUCB with dynamic schedule using a variant of the binary sum method to achieve privacy. Thanks to an asynchronous batch schedule between shuffler and bandit algorithm, it is able to take advantage of the privacy amplification through shuffling to reduce the gap between JDP and LDP regret bound.

An interesting question raised by our paper is whether it is possible to use a synchronous schedule between the shuffler and the bandit algorithm, e.g., by making the shuffler batch data dependent. We believe this would require to use some private technique (e.g., sparse vector technique by Dwork et al., 2009) to guarantee privacy at the output of the shuffler. Another direction inspired by our paper is to gain a better understanding about the intrinsic limitations of differential privacy in linear contextual bandits by studying lower-bounds for these settings.

3.3 Conclusion

In this chapter, we studied the problem of privacy in RL and Bandits under different aspects and showed how this constraint impact the exploration process. Indeed, asking to an RL algorithm to satisfy Local Differential Privacy (LDP) implies that the regret can not scale better than $\mathcal{O}\left(\frac{\sqrt{T}}{\varepsilon}\right)$, which can be prohibitively high for some applications. Experimentally we observe that the impact of LDP on the regret is significant. We also showed how to bridge the gap between the two main notion of privacy, a central and decentralized one, in linear contextual bandits. There are still many questions left unanswered by this work. For instance, in this chapter we assumed users are not recurring, an assumption that is too restrictive in real-world systems. This raises the question on the impact on the level of privacy of allowing recurring users. With the current tools, this level of privacy is expected to degrade with the square root of the number of visits from a user which is not desirable for regular users.

Local Differential Privacy requires the user to privatize their information before sending them to the bandit/RL algorithm. Therefore, in a sense it is similar to encrypting –by injecting noise in the data sent to the algorithm– the interaction between the users and the bandit/RL algorithm. In the next chapter, we aim to understand the potential effect of adversarial attacks

changing the feedback from users to the bandit/RL algorithm and present an encryption-based algorithm for linear contextual bandit that prevents the algorithm or any outside attackers to access the data from other users all while the algorithm is still able to minimize regret (at the cost of some additional computational complexity). We also investigate the change in the regret that adversarial attacks on contexts and rewards can have for well-known linear contextual bandit algorithms.

Appendix

3.A Appendix for (Local) Differential Privacy in Reinforcement LearningL

3.A.1 Extended Related Work

The notion of differential privacy was introduced in (Dwork et al., 2006) and is now a standard in machine learning (e.g., Erlingsson et al., 2014; Dwork and Roth, 2014; Abowd, 2018). In stochastic multi-armed bandits, ϵ-DP algorithms have been extensively studied (see e.g., Mishra and Thakurta, 2015; Tossou and Dimitrakakis, 2016). Recently, (Sajed and Sheffet, 2019) proposed an ϵ-DP algorithm for stochastic multi-armed bandits that achieves the private lower-bound presented in (Shariff and Sheffet, 2018). In contextual bandits, (Shariff and Sheffet, 2018) derived an impossibility result for learning under DP by showing a regret lower-bound $\Omega(T)$ for any (ϵ, δ)-DP algorithm. Instead, they considered the relaxed JDP setting and proposed an optimistic algorithm with sublinear regret and ϵ-JDP guarantees. Since the contextual bandit problem is an episodic RL problem with horizon $H = 1$, this suggests that DP is incompatible with regret minimization in RL as well.

Recently, *local differential privacy* (Duchi et al., 2013) has attracted increasing interest in the bandit literature. (Gajane et al., 2018) were the first to study LDP in stochastic MABs. (Chen et al., 2020) extended LDP to combinatorial bandits, and (Zheng et al., 2020; Ren et al., 2020) focused on LDP for MAB and contextual bandit. Private algorithms for regret minimization have also been investigated in multi-agent bandits (a.k.a. federated learning) in centralized and decentralized settings (e.g., Tossou and Dimitrakakis, 2015; Dubey and Pentland, 2020a,b), and empirical approaches have been considered in (Hannun et al., 2019; Malekzadeh et al., 2020).

In RL, (Balle et al., 2016) proposed the first private algorithm for policy evaluation with linear function approximation that ensures privacy with respect to the change of trajectories collected off-policy. (Wang and Hegde, 2019) considered the RL problem in continuous space, where reward information is protected. They designed a private version of Q-learning with function approximation where privacy with respect to different reward functions is achieved by injecting noise in the value function. (Ono and Takahashi, 2020) recently studied LDP for actor-critic methods in the context of distributed RL. None of these works considered regret minimization under privacy constraints. Regret minimization with privacy guarantees has only been considered in RL recently. (Vietri et al., 2020) designed a private optimistic algorithm for regret minimization with JDP. They proposed a variation of UBEV (Dann et al., 2017) using a randomized response mechanism with parameter ϵ/H to guarantee privacy. Their algorithm PUCB achieves a regret bound $\widetilde{O}(\sqrt{H^4SAK} + SAH^3(S+H)/\varepsilon)$ while enjoying ε-JDP. Compared to the worst case regret of UBEV, the penalty for JDP privacy is only additive, as shown by their lower-bound of $\widetilde{\Omega}\big(H\sqrt{SAK} + SAH/\varepsilon\big)$.

3.A.2 Regret Lower Bound (Proof of Thm. 6)

Let's consider the following MDP for a given number of states S and actions A. The initial state 0 has A actions which deterministically lead the next state. The MDP is a tree with A children for each node and exactly $S-2$ states.

We denote by $x_1, \dots, x_L$ the leaves of this tree. Each leaf can transition to one of the two terminal states denoted by $+$ and $-$, where the agent will receive reward of 1 or 0 respectively, and the agent will stay there until the end of the episode. There exists a unique action $a^\star$ and leaf $x_{i^\star}$ such that: $\mathbb{P}(+ \mid x_{i^\star}, a^\star) = 1/2 + \Delta$ for a chosen Δ. Each other leaf transitions with equal probability to two states $+$ and $-$ where each has a reward of 1 and 0. All other states have a reward of 0 and every other transition is deterministic.

Once the agent arrives at $+$ or $-$, it stay there until the end of the episode. In addition, we assume that $H \geq 2\ln(S-2)/\ln(A) + 2$. Let $d > 0$ be the depth of the tree, i.e., the depth of the tree with $S-2$ nodes is $d-1$ and nodes $+$,$-$ are at depth d. Then leaves $x_1, \dots, x_L$ are at depth either $d-1$ or $d-2$. Without loss of generality we assume that all $x_1, \dots, x_L$ are at depth $d-1$, i.e., the number of leaves is $L = A^{d-1} \geq (S-2)/2$, stated otherwise, the tree without the nodes $+$ and $-$ is a perfect A-ary tree. In the general case we have that $L \geq (S-2)/2$.

For a policy π, the value function can be written:

$$V^\pi(0) = (H-d)\mathbb{P}(s_d = +) = (H-d)(1/2 + \Delta\mathbb{P}\left(s_{d-1} = x_{i^\star}, a_{d-1} = a^\star\right)) \tag{3.17}$$

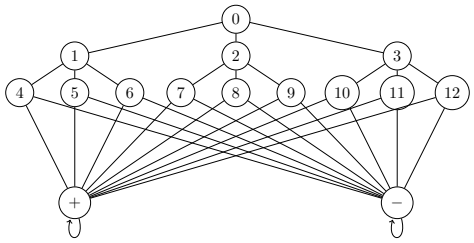


Figure 3.A.1: Example of an MDP described in this subsection with $S = 15$ and $A = 3$

Thus the regret can be written as:

$$R(K, I) = (H - d)\Delta\Big(K - \underbrace{\sum_{k=1}^{K} \mathbb{P}\left(s_{k,d-1} = x_{i^\star}, a_{k,d-1} = a^\star\right)}_{:=\mathbb{E}(T(K,I))}\Big) \tag{3.18}$$

where $I = (x_{i^\star}, a^\star)$ is the optimal state action pair and we define $T(K, I)$ as:

$$T(K, I) = \sum_{k=1}^{K} \mathbb{1}_{\{s_{k,d-1} = x_{i^\star}, a_{k,d-1} = a^\star\}}. \tag{3.19}$$

$T(K, I)$ is a function of the history observed by the algorithm. Since we consider the LDP setting, this history can be written as:

$$\mathcal{M}(\mathcal{H}_K) = \{\mathcal{M}(X_l) \mid l \leq K\} \tag{3.20}$$

where $X_l = \{(s_{l,h}, a_{l,h}, r_{l,h}) \mid h \leq H\}$ is the trajectory observed by the user for episode l and $\mathcal{M}$ is a privacy mechanism which maintains ε-LDP. Thus $T(K, I)$ is a function of $\mathcal{M}(\mathcal{H}_K)$. By Lem. A.1 in (Auer et al., 2002b):

$$\mathbb{E}(T(K, I)) \leq \mathbb{E}_0(T(K, I)) + K\sqrt{\mathsf{KL}\Big(\mathbb{P}_0(\mathcal{M}(\mathcal{H}_K)) \,||\, \mathbb{P}(\mathcal{M}(\mathcal{H}_K))\Big)} \tag{3.21}$$

where $\mathbb{E}_0$ is the expectation when $\Delta = 0$. However, because $T(K, I)$ can be seen as a function on the history only, we can use Exercise 14.4 in (Lattimore and Szepesvári, 2020) which states that for any random variable $Y : \Omega \to [a, b]$ with $(\Omega, \mathcal{F})$ a measurable space, $a < b$ and two distributions P and Q on $\mathcal{F}$, then:

$$\left|\int_{w\in\Omega} Y(w)dP(w) - \int_{w\in\Omega} Y(w)dQ(w)\right| \leq (b - a)\sqrt{\frac{\mathsf{KL}(P||Q)}{2}} \tag{3.22}$$

In our case the random variable Y is the combination of $T(K, I)$ and the privacy mechanism $\mathcal{M}$ so we have:

$$\mathbb{E}(T(K, I)) \leq \mathbb{E}_0(T(K, I)) + K\sqrt{\mathsf{KL}\Big(\mathbb{P}_0(\mathcal{H}_K) \,||\, \mathbb{P}(\mathcal{H}_K)\Big)} \tag{3.23}$$

Putting together Eq. (3.21) and (3.23), we get:

$$\mathbb{E}(T(K, I)) \leq \mathbb{E}_0(T(K, I)) + K \min\Bigg\{\underbrace{\sqrt{\mathsf{KL}\Big(\mathbb{P}_0(\mathcal{M}(\mathcal{H}_K)) \,||\, \mathbb{P}(\mathcal{M}(\mathcal{H}_K))\Big)}}_{①}, \underbrace{\sqrt{\mathsf{KL}\Big(\mathbb{P}_0(\mathcal{H}_K) \,||\, \mathbb{P}(\mathcal{H}_K)\Big)}}_{②}\Bigg\} \tag{3.24}$$

Bounding ①. Now we bound the KL-divergence between the two measures for the history. Using the chain rule we have:

$$\mathsf{KL}\left(\mathbb{P}_0(\mathcal{M}(\mathcal{H}_K)) \,||\, \mathbb{P}(\mathcal{M}(\mathcal{H}_K))\right) = \sum_{k=1}^{K} \mathbb{E}_{\mathcal{H}_{k-1}\sim\mathbb{P}_0}\left(\mathsf{KL}\left(\mathbb{P}_0(\cdot|\mathcal{M}(\mathcal{H}_{k-1})) \,||\, \mathbb{P}(\cdot|\mathcal{M}(\mathcal{H}_{k-1}))\right)\right) \tag{3.25}$$

But because $\mathcal{M}$ is an ε-LDP mechanism, Thm. 1 in (Duchi et al., 2013) ensures that:

$$\mathrm{KL}\left(\mathbb{P}_0(\cdot|\mathcal{M}(\mathcal{H}_{k-1}))\,||\,\mathbb{P}(\cdot|\mathcal{M}(\mathcal{H}_{k-1}))\right) \leq 4(\exp(\varepsilon)-1)^2\mathrm{KL}\left(\mathbb{P}_0(\cdot|\mathcal{H}_{k-1})\,||\,\mathbb{P}(\cdot|\mathcal{H}_{k-1})\right) \tag{3.26}$$

Additionally, the KL-divergence can be written as:

$$\mathrm{KL}\left(\mathbb{P}_0(\cdot|\mathcal{H}_{k-1})\,||\,\mathbb{P}(\cdot|\mathcal{H}_{k-1})\right) = \sum_{h=1}^{H}\mathbb{E}_{X_k\sim\mathbb{P}_0}\left(\ln\left(\frac{\mathbb{P}_0(s_{k,h},a_{k,h},r_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1})}{\mathbb{P}(s_{k,h},a_{k,h},r_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1})}\right)\right) \tag{3.27}$$

where $X_k = \{(s_{k,h},a_{k,h},r_{k,h})\mid h\leq H\}$ is a trajectory sampled from the MDP with the transitions distributed according to $\mathbb{P}_0$ and for each step h, $s_{k,h}$ is a state, $a_{k,h}$ an action and $r_{k,h}$ the reward associated with $(s_{k,h},a_{k,h})$.

Therefore for a step $h\geq 1$,

$$\begin{aligned}\ln\left(\mathbb{P}_0(s_{k,h},a_{k,h},r_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1})\right) &= \ln\left(\mathbb{P}_0(s_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1})\right)\\ &+\ln\left(\mathbb{P}_0(a_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1},s_{k,h})\right)\\ &+\ln\left(\mathbb{P}_0(r_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1},s_{k,h},a_{k,h})\right)\end{aligned}$$

By the Markov property of the environment:

$$\ln\left(\mathbb{P}_0(s_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1})\right) = \ln\left(\mathbb{P}_0(s_{k,h}\mid s_{k,h-1},a_{k,h-1})\right) \tag{3.28}$$

Also, since the reward only depends on the current state-action pair:

$$\ln\left(\mathbb{P}_0(r_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1},s_{k,h},a_{k,h})\right) = \ln\left(\mathbb{P}_0(r_{k,h}\mid s_{k,h},a_{k,h})\right). \tag{3.29}$$

The same results holds for $\mathbb{P}$, thus:

$$\begin{aligned}\mathrm{KL}\left(\mathbb{P}_0(\cdot|\mathcal{H}_{k-1})\,||\,\mathbb{P}(\cdot|\mathcal{H}_{k-1})\right) = \sum_{h=1}^{H}\mathbb{E}_{X_k\sim\mathbb{P}_0}\Bigg(&\ln\left(\frac{\mathbb{P}_0(s_{k,h}\mid s_{k,h-1},a_{k,h-1})}{\mathbb{P}(s_{k,h}\mid s_{k,h-1},a_{k,h-1})}\right)\\ &+\ln\left(\frac{\mathbb{P}_0(a_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1},s_{k,h})}{\mathbb{P}(a_{k,h}\mid\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1},s_{k,h})}\right)+\ln\left(\frac{\mathbb{P}_0(r_{k,h}\mid s_{k,h},a_{k,h})}{\mathbb{P}(r_{k,h}\mid s_{k,h},a_{k,h})}\right)\Bigg)\end{aligned} \tag{3.30}$$

But for $\mathbb{P}$ and $\mathbb{P}_0$ the rewards are distributed accordingly to the same distribution hence $\ln\left(\frac{\mathbb{P}_0(r_{k,h}|s_{k,h},a_{k,h})}{\mathbb{P}(r_{k,h}|s_{k,h},a_{k,h})}\right)=0$ for each $h\leq H$. The action taken at each step depends only the history of data and the current state, thus $\ln\left(\frac{\mathbb{P}_0(a_{k,h}|\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1})}{\mathbb{P}(a_{k,h}|\mathcal{H}_{k-1},(s_{k,j},a_{k,j},r_{k,j})_{j\leq h-1})}\right)=0$. Lastly, transition dynamics between $\mathbb{P}$ and $\mathbb{P}_0$ only differ when at step $d-1$ thus for all $h\neq d-1$, $\ln\left(\frac{\mathbb{P}_0(s_{k,h}|s_{k,h-1},a_{k,h-1})}{\mathbb{P}_0(s_{k,h}|s_{k,h-1},a_{k,h-1})}\right)=0$. Overall, we get:

$$\mathrm{KL}\left(\mathbb{P}_0(\cdot|\mathcal{H}_{k-1})\,||\,\mathbb{P}(\cdot|\mathcal{H}_{k-1})\right) = \sum_{l=1}^{L}\sum_{a=1}^{A}\sum_{j\in\{-,+\}}\mathbb{E}_{X_k\sim\mathbb{P}_0}\left(\ln\left(\frac{\mathbb{P}_0(j\mid x_l,a)}{\mathbb{P}(j\mid x_l,a)}\right)\mathbb{1}_{\left\{\substack{s_{k,d-1}=x_l,\\ a_{k,d-1}=a,\\ s_{k,d}=j}\right\}}\right)$$

Finally, for $j\in\{-,+\}$, $x_l\neq x_{i^\star}$ and $a\neq a^\star$, $\mathbb{P}(j\mid x_l,a)=\mathbb{P}_0(j\mid x_l,a)$. Hence,

$$\mathrm{KL}\left(\mathbb{P}_0(\cdot|\mathcal{H}_{k-1})\,||\,\mathbb{P}(\cdot|\mathcal{H}_{k-1})\right) = \frac{1}{2}\ln\left(\frac{1}{1-4\Delta^2}\right)\mathbb{E}_{X_k\sim\mathbb{P}_0}\left(\mathbb{1}_{\{s_{k,d-1}=x_{i^\star},a_{k,d-1}=a^\star\}}\right) \tag{3.31}$$

where we have used $\mathbb{P}(+\mid x_{i^\star},a^\star)=\frac{1}{2}+\Delta$, $\mathbb{P}_0(+\mid x_{i^\star},a^\star)=\frac{1}{2}$, $\mathbb{P}(-\mid x_{i^\star},a^\star)=\frac{1}{2}-\Delta$ and $\mathbb{P}_0(-\mid x_{i^\star},a^\star)=\frac{1}{2}$.

Therefore combining (3.26) and (3.31) and summing over the episodes, we get:

$$\begin{aligned}\mathrm{KL}\Big(\mathbb{P}_0(\mathcal{M}(\mathcal{H}_K))\,||\,\mathbb{P}(\mathcal{M}(\mathcal{H}_K))\Big) &\leq 2(e^\varepsilon-1)^2\ln\left(\frac{1}{1-4\Delta^2}\right)\sum_{k=1}^{K}\mathbb{P}_0\left(s_{k,d-1}=x_{i^\star},a_{k,d-1}=a^\star\right)\\ &= 2(e^\varepsilon-1)^2\ln\left(\frac{1}{1-4\Delta^2}\right)\mathbb{E}_0(T(K,I))\end{aligned} \tag{3.32}$$

Bounding ②. Using again the chain rule of the KL-divergence, we have that:

$$\mathrm{KL}\left(\mathbb{P}_0(\mathcal{H}_K) \,||\, \mathbb{P}(\mathcal{H}_K)\right) = \sum_{k=1}^{K} \mathbb{E}_{\mathcal{H}_{k-1}\sim\mathbb{P}_0}\left(\mathrm{KL}\left(\mathbb{P}_0(\cdot|\mathcal{H}_{k-1}) \,||\, \mathbb{P}(\cdot|\mathcal{H}_{k-1})\right)\right) \tag{3.33}$$

Therefore, using Eq. (3.31), we have:

$$\begin{aligned}\mathrm{KL}\left(\mathbb{P}_0(\mathcal{H}_K) \,||\, \mathbb{P}(\mathcal{H}_K)\right) &= \sum_{k=1}^{K} \mathbb{E}_{\mathcal{H}_{k-1}\sim\mathbb{P}_0}\left(\frac{1}{2}\ln\left(\frac{1}{1-4\Delta^2}\right)\mathbb{E}_{X_k\sim\mathbb{P}_0}\left(\mathbb{1}_{\left\{\substack{s_{k,d-1}=x_{i^\star},\\ a_{k,d-1}=a^\star}\right\}}\right)\right) \\ &= \frac{1}{2}\ln\left(\frac{1}{1-4\Delta^2}\right)\mathbb{E}_0(T(K,I))\end{aligned} \tag{3.34}$$

Finishing the proof. Hence using Eq. (3.32) and Eq. (3.34) in Eq. (3.24):

$$\mathbb{E}(T(K,I)) \leq \mathbb{E}_0(T(K,I)) + K\min\left\{\sqrt{2}(e^\varepsilon-1), \frac{1}{\sqrt{2}}\right\}\sqrt{\mathbb{E}_0(T(K,I))\ln\left(\frac{1}{1-4\Delta^2}\right)} \tag{3.35}$$

Now, let's assume that $I = (x_{i^\star}, a^\star)$ is distributed uniformly over $\{x_1,\dots,x_L\}\times[A]$. That is to say, that the leaf $i^\star \sim \mathcal{U}([L])$ and given the realization of $i^\star$, $a^\star$ is drawn uniformly in the action set of node $x_{i^\star}$ i.e., $a^\star \sim \mathcal{U}([A])$. We denote the expectation over the random variable $(x_{i^\star}, a^\star)$ by $\mathbb{E}_I$. It then holds that:

$$\mathbb{E}_I\mathbb{E}_0(T(K,I)) = \mathbb{E}_0\sum_{k=1}^{K}\sum_{l=1}^{L}\sum_{a=1}^{A}\frac{1}{LA}\mathbb{1}_{\{s_{k,d-1}=s, a_{k,d-1}=a\}} = \frac{K}{LA} \tag{3.36}$$

Therefore thanks to Jensen's inequality the regret is lower-bounded by:

$$\mathbb{E}_I R(K,I) \geq (H-d)\Delta K\left(1-\frac{1}{LA}-\min\left\{\sqrt{2}(e^\varepsilon-1), \frac{1}{\sqrt{2}}\right\}\sqrt{\frac{K}{LA}\ln\left(1+\frac{4\Delta^2}{1-4\Delta^2}\right)}\right) \tag{3.37}$$

Therefore for $LA \geq 2$, $K \geq \frac{LA}{\min\{8(e^\varepsilon-1),4\}^2}$ and choosing $\Delta = \sqrt{\frac{LA}{K}} \times \frac{1}{16\sqrt{2}\min\left\{(e^\varepsilon-1),\frac{1}{2}\right\}}$ we get that:

$$\min\left\{\sqrt{2}(\exp(\varepsilon)-1), \frac{1}{\sqrt{2}}\right\}\sqrt{\frac{K}{LA}\ln\left(1+\frac{4\Delta^2}{1-4\Delta^2}\right)} \leq \frac{1}{4}$$

Hence:

$$\max_{I\in\{x_1,\dots,x_L\}\times[A]} R(K,I) \geq \mathbb{E}_I R(K,I) \geq \frac{(H-d)\sqrt{KLA}}{64\min\left\{(\exp(\varepsilon)-1), \frac{1}{2}\right\}} \tag{3.38}$$

And because I is a finite random variable there exist $I^\star$ such that $\max_{I\in\{x_1,\dots,x_L\}\times[A]} R(K,I) = R(K,I^\star)$.

$$R(K,I^\star) \geq \frac{(H-d)\sqrt{KLA}}{64\min\left\{(\exp(\varepsilon)-1), \frac{1}{2}\right\}} \tag{3.39}$$

Thus we have that there exists an MDP such that its frequentist regret is $\Omega\left(\frac{H\sqrt{SAK}}{\min\{1,\exp(\varepsilon)-1\}}\right)$.

3.A.3 Concentration under Local Differential Privacy (Proof of Prop. 6):

In this subsection, we proceed with the proof of Prop. 6 (recalled below).

Proposition. *For any $\varepsilon_0 > 0$, $\delta_0 \geq 0$, $\delta > 0$, $\alpha > 1$ and episode k, using mechanism $\mathcal{M}$ satisfying Asm. 11, then with probability at least $1-2\delta$, for any $(s,a) \in \mathcal{S}\times\mathcal{A}$*

$$|r(s,a) - \widetilde{r}_k(s,a)| \leq \beta_k^r(s,a) = \sqrt{\frac{2\ln\left(\frac{4\pi^2 SAHk^3}{3\delta}\right)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}} + \frac{(\alpha+1)c_{k,2}(\varepsilon_0,\delta_0,\delta)+c_{k,1}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}$$

$$\|p(\cdot|s,a) - \widetilde{p}_k(\cdot|s,a)\|_1 \leq \beta_k^p(s,a) = \sqrt{\frac{14S\ln\left(\frac{4\pi^2 SAHk^3}{3\delta}\right)}{\widetilde{N}_k^p(s,a)+\alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}} + \frac{Sc_{k,4}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^p(s,a)+\alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} + \frac{(\alpha+1)c_{k,3}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^p(s,a)+\alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}$$

Proof. On the event that all inequalities of Def. 11 holds, we have:

$$\left|\frac{\widetilde{R}_k(s,a)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}-\frac{R_k(s,a)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}\right|\leq\frac{c_{k,1}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)} \tag{3.40}$$

since $\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)>N_k^k(s,a)\geq 0$ with $\alpha>1$. But, we also have that with probability $1-\delta$:

$$\begin{aligned}\left|\frac{R_k(s,a)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}-r(s,a)\right|&\leq\left|r(s,a)\left(\frac{N_k^r(s,a)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}-1\right)\right| \\ &\quad+\left|\frac{N_k^r(s,a)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}\times\underbrace{\left(\frac{R_k(s,a)}{N_k^r(s,a)}-r(s,a)\right)}_{:=\overline{r}_k(s,a)-r(s,a)}\right| \qquad (3.41)\\ &\leq\frac{N_k^r(s,a)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}\frac{L(\delta)}{\sqrt{N_k^r(s,a)}}+r(s,a)\left|1-\frac{N_k^r(s,a)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}\right| \qquad (3.42)\\ &\leq\frac{L(\delta)\sqrt{N_k^r(s,a)}}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}+\frac{(\alpha+1)c_{k,2}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)} \qquad (3.43)\end{aligned}$$

where the second inequality follows from Chernoff-Hoeffding bound on the empirical non-private rewards with $L(\delta)=\sqrt{2\ln(4\pi^2SAHk^3/3\delta)}$, and we use Def. 11 for the last. Furthermore:

$$\frac{L(\delta)\sqrt{N_k^r(s,a)}}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}\leq\frac{L(\delta)\sqrt{\widetilde{N}_k^r(s,a)+c_{k,2}(\varepsilon_0,\delta_0,\delta)}}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}\leq\frac{L(\delta)}{\sqrt{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}} \tag{3.44}$$

Therefore combining Eq. (3.40), (3.43) and (3.44), we have:

$$\begin{aligned}\left|\frac{\widetilde{R}_k(s,a)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}-r(s,a)\right|&\leq\frac{c_{k,1}(\varepsilon_0,\delta_0,\delta)+(\alpha+1)c_{k,2}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}\\ &\quad+\frac{L(\delta)}{\sqrt{\widetilde{N}_k^r(s,a)+\alpha c_{k,2}(\varepsilon_0,\delta_0,\delta)}}\end{aligned}$$

thus proving the first statement of the proposition. Now, we bound the deviation between the private estimate $\widetilde{p}_k$ and the true transition dynamics p. First, because $\alpha>1$, we have that $\sum_{s'}\widetilde{N}_k^p(s,a,s')+\alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)\geq\sum_{s'}N_k^p(s,a,s')+(\alpha-1)c_{k,3}(\varepsilon_0,\delta_0,\delta)>0$. We start by decomposing the error as

$$\begin{aligned}\sum_{s'\in\mathcal{S}}\left|\widetilde{p}(s'|s,a)-p(s'|s,a)\right|&=\sum_{s'\in\mathcal{S}}\left|\frac{\widetilde{N}_k^p(s,a,s')}{\sum_{s'}\widetilde{N}_k^p(s,a,s')+\alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}-p(s'|s,a)\right|\\ &\leq\underbrace{\sum_{s'\in\mathcal{S}}\left|\frac{N_k^p(s,a,s')}{\sum_{s'}\widetilde{N}_k^p(s,a,s')+\alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}-p(s'\mid s,a)\right|}_{①}+\underbrace{\sum_{s'\in\mathcal{S}}\left|\frac{\widetilde{N}_k^p(s,a,s')-N_k^p(s,a,s')}{\sum_{s'}\widetilde{N}_k^p(s,a,s')+\alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}\right|}_{②} \qquad (3.45)\end{aligned}$$

Recall that $\sum_{s'} \widetilde{N}_k^p(s,a,s') = \widetilde{N}_k^p(s,a)$ and $\sum_{s'} N_k^p(s,a,s') = N_k^p(s,a)$ and define $\overline{p}_k(\cdot|s,a) = \frac{N_k^p(s,a,\cdot)}{N_k^p(s,a)}$. Therefore:

$$
\begin{aligned}
① &= \sum_{s' \in \mathcal{S}} \left| \frac{N_k^p(s,a,s')}{N_k^p(s,a)} \frac{N_k^p(s,a)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} - p(s' \mid s,a) \right| \\
&= \sum_{s'} \left| \underbrace{\frac{\left(\frac{N_k^p(s,a,s')}{N_k^p(s,a)} - p(s'|s,a)\right) N_k^p(s,a)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}}_{>0} + p(s'|s,a) \left(\frac{N_k^p(s,a)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} - 1 \right) \right| \\
&\leq \sum_{s'} \left(p(s'|s,a) \frac{(\alpha+1) c_{k,3}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} \right) + \frac{N_k^p(s,a) \|\overline{p}_k(\cdot|s,a) - p(\cdot|s,a)\|_1}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} \\
&\overset{(a)}{\leq} \frac{(\alpha+1) c_{k,3}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} + \frac{N_k^p(s,a)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} \frac{L(\delta)}{\sqrt{N_k^p(s,a)}} \\
&\leq \frac{(\alpha+1) c_{k,3}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} + \frac{L(\delta)}{\sqrt{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}}
\end{aligned}
$$

where $L(\delta) = \sqrt{14S \ln(4\pi^2 SAHk^3/3\delta)}$ and inequality (a) follows from the Weissman inequality (Weissman et al., 2003), and we have again used the fact that the inequalities in Def. 11 hold.

In addition,we have:

$$
② \leq \sum_{s' \in \mathcal{S}} \frac{|c_{k,4}(\varepsilon_0,\delta_0,\delta)|}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} = \frac{S c_{k,4}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} \tag{3.46}
$$

Hence putting together Eq. (3.46) and Eq. (3.46), we have:

$$
\begin{aligned}
\sum_{s' \in \mathcal{S}} \left| \frac{\widetilde{N}_k^p(s,a,s')}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} - p(s' \mid s,a) \right| \leq & \frac{S c_{k,4}(\varepsilon_0,\delta_0,\delta) + (\alpha+1) c_{k,3}(\varepsilon_0,\delta_0,\delta)}{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)} \\
& + \frac{L(\delta)}{\sqrt{\widetilde{N}_k^p(s,a) + \alpha c_{k,3}(\varepsilon_0,\delta_0,\delta)}}
\end{aligned} \tag{3.47}
$$

□

3.A.4 Regret Upper Bound (Proof of Thm. 7)

In this subsection, we prove Thm 7, which we recall below.

Theorem. *For any privacy mechanism $\mathcal{M}$ satisfying Asm. 11 with $\varepsilon > 0$, $\delta_0 \geq 0$, and for any $\delta > 0$ the regret of* LDP-OBI *is bounded with probability at least $1 - \delta$ by:*

$$
\begin{aligned}
\Delta(K) \leq \tilde{\mathcal{O}} \Bigg(& \underbrace{HS\sqrt{AT}}_{①} + SAH^2 c_{K,3}\left(\varepsilon, \delta_0, \frac{3\delta}{2\pi^2 K^2}\right) + H^2 S^2 A c_{K,4}\left(\varepsilon, \delta_0, \frac{3\delta}{2\pi^2 K^2}\right) \\
& + SAH c_{K,2}\left(\varepsilon, \delta_0, \frac{3\delta}{2\pi^2 K^2}\right) + SAH c_{K,1}\left(\varepsilon, \delta_0, \frac{3\delta}{2\pi^2 K^2}\right) \Bigg)
\end{aligned} \tag{3.48}
$$

The combination of $\mathcal{M}$ and LDP-OBI *is also (ε, δ_0)-LDP.*

Good Event: Before proceeding the proof of the regret we define a good event under which all concentration inequalities holds with probability at least $1-\delta$. First, we define the event that all inequalities from Def. 11 holds. Let:

$$
\begin{aligned}
L_{1,k} &= \bigcap_{s,a}\left\{\left|\widetilde{R}_k(s,a) - R_k(s,a)\right| \leq c_{k,1}(\varepsilon_0,\delta_0,3\delta/2k^2\pi^2)\right\}\\
L_{2,k} &= \bigcap_{s,a}\left\{\left|\widetilde{N}^r_k(s,a) - N^r_k(s,a)\right| \leq c_{k,2}(\varepsilon_0,\delta_0,3\delta/2k^2\pi^2)\right\}\\
L_{3,k} &= \bigcap_{s,a}\left\{\left|\sum_{s'} N^p_k(s,a,s') - \sum_{s`} \widetilde{N}^p_k(s,a,s')\right| \leq c_{k,3}(\varepsilon_0,\delta_0,3\delta/2k^2\pi^2)\right\}\\
L_{4,k} &= \bigcap_{s,a,s'}\left\{\left|N^p_k(s,a,s') - \widetilde{N}^p_k(s,a,s')\right| \leq c_{k,4}(\varepsilon_0,\delta_0,3\delta/2k^2\pi^2)\right\}
\end{aligned}
$$

then thanks to Def. 11 we have :

$$
\mathbb{P}\left(\bigcup_{k=1}^{+\infty} L^c_{1,k}\cup L^c_{2,k}\cup L^c_{3,k}\cup L^c_{4,k}\right) \leq \sum_{k=1}^{+\infty}\frac{3\delta}{\pi^2k^2} = \frac{\delta}{4} \tag{3.49}
$$

In addition, for all $k\in\mathbb{N}^\star$, we can define $\overline{r}_k(s,a) = R_k(s,a)/N^r_k(s,a)$ and $\overline{p}_k = N^p_k(s,a,s')/\sum_{s'}N^p_k(s,a,s')$ as the empirical reward and transition probability computed with the non-private counters. Note that in this case $N_k(s,a) := N^r_k(s,a) = \sum_{s'}N^p_k(s,a,s')$. We also define $\overline{\beta}^r_k(\delta,s,a) = \sqrt{\frac{2\ln(1/\delta)}{N_k(s,a)}}$ and $\overline{\beta}^p_k(\delta,s,a) = \sqrt{\frac{14S\log(1/\delta)}{N_k(s,a)}}$. as the size of the confidence intervals using Hoeffding and Weissman inequalities. Thus, we get:

$$
\begin{aligned}
&\mathbb{P}\left(\bigcup_{k=1}^{+\infty}\bigcup_{s,a}|\overline{r}_k(s,a)-r(s,a)| \geq \overline{\beta}^r_k(3\delta/4\pi^2SAHk^3,s,a)\right)\\
&\leq \sum_{k=1}^{+\infty}\sum_{s,a}\mathbb{P}\left(|\overline{r}_k(s,a)-r(s,a)| \geq \sqrt{\frac{2\ln(4\pi^3SAHk^3/3\delta)}{N_k(s,a)}}\right)\\
&\leq \sum_{k=1}^{+\infty}\sum_{s,a}\sum_{n=0}^{kH}\mathbb{P}\left(|\overline{r}_k(s,a)-r(s,a)| \geq \sqrt{\frac{2\ln(4\pi^2SAHk^3/3\delta)}{n}}\right) \leq \sum_{k=1}^{+\infty}\sum_{s,a}\sum_{n=0}^{kH}\frac{3\delta}{4\pi^2SHAk^3} \leq \frac{\delta}{8}
\end{aligned}
$$

A similar result holds for the transition dynamics, i.e.,:

$$
\mathbb{P}\left(\bigcup_{k=1}^{+\infty}\bigcup_{s,a}||\overline{p}_k(\cdot|s,a)-p(\cdot|s,a)||_1 \geq \overline{\beta}^p_k(3\delta/4\pi^2SAHk^3,s,a)\right) \leq \frac{\delta}{8} \tag{3.50}
$$

Thus we can define the good event $\mathcal{G}_k$ by:

$$
\begin{aligned}
\mathcal{G}_k = \bigcap_{l=1}^{k-1}\bigcap_{i=1}^{4} L_{i,l} \cap \bigcap_{s,a}&\left\{|\overline{r}_l(s,a)-r(s,a)| \leq \overline{\beta}^r_l(3\delta/(4\pi^2SAHl^3),s,a)\right\}\\
&\cap\left\{||\overline{p}_k(\cdot|s,a)-p(\cdot|s,a)||_1 \leq \overline{\beta}^p_k(3\delta/(4\pi^2SAHl^3),s,a)\right\}
\end{aligned}
$$

Then $\mathbb{P}\left(\bigcap_{k=1}^{+\infty}\mathcal{G}_k\right)\geq 1-\delta/2$ and $\mathcal{G}_k\subset\sigma(\mathcal{H}_k)$ (i.e., the history before episode k).

Optimism: For each episode k, the value function $V_{k,1}$ computed by LDP-OBI is optimistic, that is to say: $V_{k,h}(s)\geq V^\star_h(s)$ for any h and state s. We sum up this with the following lemma:

Lemma 22. *For any episode $k\in[k]$, the value function $V_{k,1}$ computed by running Alg. 11 is such that with probability $1-\delta$:*

$$
\forall s\in\mathcal{S}, h\in[1,H] \qquad V_{k,h}(s)\geq V^\star_h(s) \tag{3.51}
$$

Proof. Fix an episode k then we proceed by backward induction conditioned on the event $\mathcal{G}_k$:

- For $h=H$, we have for any state s and action a:

$$
V_{k,H}(s)\geq Q_{k,H}(s,a)\geq \widetilde{r}_k(s,a)+\beta^r_k(s,a)\geq r(s,a) \text{ thanks to Prop. 6} \tag{3.52}
$$

- For $h < H$ when the property is true for $h+1$, we get for any state-action (s,a):

$$V_{k,h}(s) \geq Q_{k,h}(s,a) = \widetilde{r}_k(s,a) + \beta_k^r(s,a) + \widetilde{p}_k(\cdot|s,a)^\intercal V_{k,h+1} + H\beta_k^p(s,a) \tag{3.53}$$
$$\geq r(s,a) + p(\cdot|s,a)^\intercal V_{k,h+1} \geq Q_h^\star(s,a) \tag{3.54}$$

where we used the fact that $\|(\widetilde{p}_k(\cdot|s,a) - p(\cdot|s,a))^\intercal V_{k,h+1}\| \leq \|\widehat{p}_k(\cdot|s,a) - p(\cdot|s,a)\|_1 \|V_{k,h+1}\|_\infty \leq H\beta_k^p(s,a)$ and the inductive hypothesis.

□

Regret Decomposition: We are now ready to analyze the regret of LDP-OBI. Consider an episode k, then, conditioned on $\mathcal{G}_k$:

$$V_1^\star(s_{k,1}) - V_1^{\pi_k}(s_{k,1}) \leq V_{k,1}(s_{k,1}) - V_1^{\pi_k}(s_{k,1}) \leq \widetilde{r}_k(s_{k,1},a_{k,1}) + \beta_k^r(s_{k,1},a_{k,1}) - r(s_{k,1},a_{k,1})$$
$$+\widetilde{p}_k(\cdot|s,a)^\intercal V_{k,2} - p(\cdot|s,a)^\intercal V_2^{\pi_k} + H\beta_k^p(s_{k,1},a_{k,1})$$

where the last inequality follows from recursively applying the same technique. Then, observe that $(\eta_{k,h})_{k,h}$ is a Martingale Difference Sequence with respect to the history before episode k and thanks to Azuma-Hoeffding inequality we have that with probability at least $1-\delta/2$, $\sum_{k=1}^K \sum_{h=1}^{H-1} \eta_{k,h} \leq 2H\sqrt{KH\ln(2/\delta)}$. Therefore, we have with probability at least $1-\delta$:

$$R(\text{LDP-OBI}, K) \leq 2\sum_{k=1}^K \sum_{h=1}^H \beta_k^r(s_{k,h},a_{k,h}) + H\beta_k^p(s_{k,h},a_{k,h}) + \underbrace{2H\sqrt{T\ln(2/\delta)}}_{\text{MDS error term}} \tag{3.55}$$

Let $\nu_k(s,a) = \sum_{h=1}^H \mathbb{1}_{\{s_{k,h}=s, a_{k,h}=a\}}$. Then summing over the reward bonus and using the fact that $\alpha > 1$, we get:

$$\begin{aligned}\sum_{k=1}^K \sum_{h=1}^H \beta_k^r(s_{k,h},a_{k,h}) &= \sum_{s,a,k} \frac{\nu_k(s,a)L_{k,r}}{\sqrt{\widetilde{N}_k^r(s,a) + \alpha c_{k,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}} \\ &+ \sum_{s,a,k} \frac{\nu_k(s,a)(\alpha+1)c_{k,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}{\alpha c_{k,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right) + \widetilde{N}_k^r(s,a)} \\ &+ \sum_{s,a,k} \frac{\nu_k(s,a)c_{k,1}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}{\alpha c_{k,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right) + \widetilde{N}_k^r(s,a)}\end{aligned} \tag{3.56}$$

where $L_{k,r} = \sqrt{2\ln\left(\frac{4\pi^2SAHk^3}{3\delta}\right)}$. Then, using that $\widetilde{N}_k^r(s,a) + c_{k,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right) \geq N_k(s,a)$ on the good event from $\mathcal{G}_k$:

$$\begin{aligned}(3.56) \leq \sum_{s,a,k} \frac{\nu_k(s,a)L_{k,r}}{\sqrt{N_k(s,a) + (\alpha-1)c_{k,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}} &+ \frac{\nu_k(s,a)(\alpha+1)c_{k,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}{(\alpha-1)c_{k,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right) + N_k(s,a)} \\ &+ \sum_{s,a,k} \frac{\nu_k(s,a)c_{k,1}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}{(\alpha-1)c_{k,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right) + N_k(s,a)}\end{aligned} \tag{3.57}$$

But because $c_{k,2}$ is non-decreasing in k, we have that,

$$\begin{aligned}(3.57) \leq \left((\alpha+1)c_{K,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right) + c_{K,1}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)\right) &\sum_{k,s,a} \frac{\nu_k(s,a)}{N_k(s,a)} \\ &+ \sum_{s,a,k} \frac{\nu_k(s,a)L_{K,r}}{\sqrt{N_k(s,a)}}\end{aligned} \tag{3.58}$$

Which can be rewritten as:

$$\begin{aligned}(3.58) \leq 2\left((\alpha+1)c_{K,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right) + c_{K,1}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)\right) &SA(\ln(2TSA) + H) \\ &+\sqrt{6\ln\left(14SAT/\delta\right)}\left(\sqrt{2SAT} + HSA\right)\end{aligned} \tag{3.59}$$

where the last inequality comes from Lem. 19 in (Jaksch et al., 2010a). For the sum of the bonus on the transition dynamics we have that:

$$\begin{aligned}\sum_{k=1}^{K}\sum_{h=1}^{H} H\beta_k^p(s_{k,h},a_{k,h}) &= \sum_{s,a,k}\frac{H\nu_k(s,a)L_{k,p}}{\sqrt{\widetilde{N}_k^p(s,a)+\alpha c_{k,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}}\\ &+\sum_{s,a,k}\frac{HS\nu_k(s,a)c_{k,4}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}{\alpha c_{k,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)+\widetilde{N}_k^p(s,a)}\\ &+\sum_{s,a,k}\frac{H\nu_k(s,a)(\alpha+1)c_{k,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}{\alpha c_{k,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)+\widetilde{N}_k^p(s,a)}\end{aligned} \tag{3.60}$$

where $L_{k,p}=\sqrt{14S\ln\left(\frac{4\pi^2SAHk^3}{3\delta}\right)}$. Then similarly to the reasonning used to bound Eq. (3.56), we have:

$$\begin{aligned}(3.60) &\leq \sum_{s,a,k}\frac{H\nu_k(s,a)L_{k,p}}{\sqrt{N_k(s,a)+(\alpha-1)c_{k,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}}+\sum_{s,a,k}\frac{H\nu_k(s,a)(\alpha+1)c_{k,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}{(\alpha-1)c_{k,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)+N_k(s,a)}\\ &+\sum_{k,s,a}\frac{HSc_{k,4}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)}{(\alpha-1)c_{k,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2k^2}\right)+N_k(s,a)}\\ &\leq +\left((\alpha+1)c_{K,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)+Sc_{K,4}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)\right)\sum_{k,s,a}\frac{H\nu_k(s,a)}{N_k(s,a)}\\ &\sum_{s,a,k}\frac{H\nu_k(s,a)L_{K,p}}{\sqrt{N_k(s,a)}}\\ &\leq 2SAH\left((\alpha+1)c_{K,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)+Sc_{K,4}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)\right)(\ln(2TSA)+H)\\ &+H\sqrt{46S\ln\left(14SAT/\delta\right)}\left(\sqrt{2SAT}+HSA\right)\end{aligned}$$

where the last inequality comes from (Jaksch et al., 2010a, Lem. 19) and (Fruit et al., 2020, Lem. 8). Hence putting everything together, we get that with probability $1-\delta$:

$$\begin{aligned}R(\text{LDP-OBI},K) &\leq H\sqrt{46S\ln(14SAT/\delta)}(\sqrt{2SAT}+HSA)+\sqrt{6\ln(14SAT/\delta)}(\sqrt{2SAT}+HSA)\\ &+2SAH\left((\alpha+1)c_{K,3}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)+Sc_{K,4}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)\right)(\ln(2TSA)+H)\\ &+2\left((\alpha+1)c_{K,2}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)+c_{K,1}\left(\varepsilon_0,\delta_0,\frac{3\delta}{2\pi^2K^2}\right)\right)SA(\ln(2TSA)+H)+2H\sqrt{T\ln(2/\delta)}\end{aligned}$$

In addition, because LDP-OBI has only access to the privatized data, that is to say it only uses the output of $\mathcal{M}(\{(s_{k,h},a_{k,h},r_{k,h})_{h\leq H}\})$ for each episode k, the LDP constraint is satsified as long as the privacy mechanism $\mathcal{M}$ satisfies Def. 3.

Note: the proof of this regret upper-bound relies on concentration inequalities more generally used in the average reward regret minimization setting. Stated otherwise, we directly study the error between the estimated model and the true model, i.e., $|\widetilde{r}_k-r|$ and $||\widetilde{p}_k(.\mid s,a)-p(.\mid s,a)||_1$ for each s,a. In the non-private setting, it is possible to get a more refined regret using more precise concentration inequalities, mainly Bernstein inequality and other tools introduced in (Azar et al., 2017a). However, in the private setting, using such results only leads to a gain in lower order terms and terms independent of ε while the technical derivations are much more intricate.

3.A.5 The Laplace Mechanism for Local Differential Privacy

In this appendix, we show how the well-known Laplace mechanism (Dwork et al., 2006) can be used with LDP-OBI to ensure LDP and a sublinear regret.

Algorithm 13: Laplace mechanism for LDP

Input: Trajectory: $X = \{(s_h, a_h, r_h) \mid h \leq H\}$, Privacy Parameter: ε_0

Draw $(Y_{i,X}(s,a))_{(s,a)\in\mathcal{S}\times\mathcal{A}, i\leq 2}$ i.i.d Lap$(1/\varepsilon_0)$ and $(Z_X(s,a,s'))_{(s,a,s')\in\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$ i.i.d Lap$(1/\varepsilon_0)$ and independent from $Y_{i,X}$ for $i \in \{1,2\}$

for $(s,a) \in \mathcal{S}\times\mathcal{A}$ **do**

$\widetilde{R}_X(s,a) = \sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h,a_h=s,a\}} + Y_{1,X}(s,a)$

$\widetilde{N}^r_X(s,a) = \sum_{h=1}^{H} \mathbb{1}_{\{s_h,a_h=s,a\}} + Y_{2,X}(s,a)$

for $s' \in \mathcal{S}$ **do**

$\widetilde{N}^p_X(s,a,s') = \sum_{h=1}^{H-1} \mathbb{1}_{\{s_h,a_h,s_{h+1}=s,a,s'\}} + Z_X(s,a,s')$

Output: $(\widetilde{R}_X, \widetilde{N}^r_X, \widetilde{N}^p_X) \in \mathbb{R}^{S\times A} \times \mathbb{R}^{S\times A} \times \mathbb{R}^{S\times A\times S}$

3.A.5.1 The Laplace mechanism (Alg. 13) satisfies local differential privacy (Asm. 11)

We first prove Thm. 10 which states that using Alg. 13 with parameter $\varepsilon_0 = \varepsilon/6H$ guarantees (ε,δ)-LDP.

Theorem 10. *For any $\varepsilon > 0$, the Laplace mechanism described by Alg. 13 with parameter $\varepsilon_0 = \varepsilon/6H$ is $(\varepsilon, 0)$-LDP (and thus (ε,δ_0)-LDP for every $\delta_0 \geq 0$).*

Formally, we need to show that, for any two trajectories X and X' and tuple (r,n,n'), the following inequality holds

$$\mathbb{P}\Big(\mathcal{M}(X) = (r,n,n')\Big) \leq e^{\varepsilon}\mathbb{P}\Big(\mathcal{M}(X') = (r,n,n')\Big) + \delta \tag{3.61}$$

where r, n, n' are vectors of dimension SA, SA and S^2A, respectively. See the LDP definition in Def. 3.

Proof of Thm. 10. Let's consider two trajectories $X = \{(s_h,a_h,r_h) \mid h \leq H\}$ and $X' = \{(s'_h,a'_h,r'_h) \mid h \leq H\}$. We denote the output of the private randomizer $\mathcal{M}$ by $\mathcal{M}(X) = (\widetilde{R}_X, \widetilde{N}^r_X, \widetilde{N}^p_X)$ and $\mathcal{M}(X') = (\widetilde{R}_{X'}, \widetilde{N}^r_{X'}, \widetilde{N}^p_{X'})$. Recall that $\widetilde{R}_X(s,a) := \sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} + Y_{1,X}(s,a)$ where $(Y_{1,X}(s,a))_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ are independent Laplace variables with parameter $\varepsilon/(6H)$. Consider a vector $r \in \mathbb{R}^{S\times A}$, then:

$$\frac{\mathbb{P}\left(\forall(s,a), \widetilde{R}_X(s,a) = r_{s,a} \mid X\right)}{\mathbb{P}\left(\forall(s,a), \widetilde{R}_{X'}(s,a) = r_{s,a} \mid X'\right)} = \prod_{s,a} \frac{\mathbb{P}\left(Y_{1,X}(s,a) = \sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} - r_{s,a} \mid X\right)}{\mathbb{P}\left(Y_{1,X'}(s,a) = \sum_{h=1}^{H} r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}} - r_{s,a} \mid X'\right)} \tag{3.62}$$

since the Laplace distribution is symmetric. But $Y_{1,X}(s,a)$ and $Y_{1,X'}(s,a)$ are independent random variables for any state-action pair. Thus:

$$\begin{aligned}
\prod_{s,a} \frac{\mathbb{P}\left(Y_{1,X}(s,a) = \sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}} - r_{s,a} \mid X\right)}{\mathbb{P}\left(Y_{1,X'}(s,a) = \sum_{h=1}^{H} r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}} - r_{s,a} \mid X'\right)} &= \prod_{s,a} \frac{e^{\left(\varepsilon_0 \left|\sum_{h=1}^{H} (r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}} - r_{s,a}\right|\right)}}{e^{\left(\varepsilon_0 \left|\sum_{h=1}^{H} (r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}} - r_{s,a}\right|\right)}} \\
&\leq \exp\left(\varepsilon_0 \sum_{s,a} \left|\sum_{h=1}^{H} (r_h \mathbb{1}_{\{s_h=s,a_h=a\}} - r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}})\right|\right) \\
&\leq \exp\left(\varepsilon_0 \sum_{s,a,h} (|r_h| \mathbb{1}_{\{s_h=s,a_h=a\}} + |r'_h| \mathbb{1}_{\{s'_h=s,a'_h=a\}})\right) \\
&= \exp\left(\varepsilon_0 \sum_{h} (|r_h| + |r'_h|)\right) \leq \exp(2H\varepsilon_0) = \exp\left(\frac{\varepsilon}{3}\right)
\end{aligned} \tag{3.63}$$

where we used the definition of the Laplace distribution, $x \mapsto \frac{1}{2b}\exp(|x|/b)$. Let $n \in \mathbb{R}^{S\times A}$ and $n' \in \mathbb{R}^{S\times A\times S}$. Similarly, since $\widetilde{N}^r_X(s,a) = \sum_{h=1}^{H} \mathbb{1}_{\{s_h=s,a_h=a\}} + Y_{2,X}(s,a)$ and $\widetilde{N}^p_X(s,a,s') = \sum_{h=1}^{H-1} \mathbb{1}_{\{s_h=s,a_h=a,s_{h+1}=s'\}} + Z_X(s,a,s')$, we have:

$$\frac{\mathbb{P}\left(\forall(s,a), \widetilde{N}^r_X(s,a) = n_{s,a} \mid X\right)}{\mathbb{P}\left(\forall(s,a), \widetilde{N}^r_{X'}(s,a) = n_{s,a} \mid X'\right)} \leq \exp\left(\frac{\varepsilon}{3}\right) \tag{3.64}$$

and:

$$\frac{\mathbb{P}\left(\forall(s,a,s'), \widetilde{N}^p_X(s,a,s') = n'_{s,a,s'} \mid X\right)}{\mathbb{P}\left(\forall(s,a,s'), \widetilde{N}^p_{X'}(s,a,s') = n'_{s,a,s'} \mid X'\right)} \leq \exp\left(\frac{\varepsilon}{3}\right) \tag{3.65}$$

Then because $(Y_{i,X}(s,a))_{i\leq 2,(s,a)\in\mathcal{S}\times\mathcal{A}}$, $(Z_X(s,a,s'))_{(s,a,s')\in\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$ are independent it holds that:

$$\mathbb{P}\left(\widetilde{R}_X = r, \widetilde{N}^r_X = n, \widetilde{N}^p_X = n' \mid X\right) = \mathbb{P}\left(\widetilde{R}_X = r \mid X\right)\mathbb{P}\left(\widetilde{N}^r_X = n \mid X\right)\mathbb{P}\left(\widetilde{N}^p_X = n' \mid X\right)$$

Thus for any $(r,n,n') \in \mathbb{R}^{S\times A} \times \mathbb{R}^{S\times A} \times \mathbb{R}^{S\times A\times S}$ and any two trajectories X and X':

$$\mathbb{P}\Big(\mathcal{M}(X) = (r,n,n') \mid X\Big) = \mathbb{P}\left(\widetilde{R}_X = r, \widetilde{N}^r_X = n, \widetilde{N}^p_X = n' \mid X\right) \tag{3.66}$$

$$= \mathbb{P}\left(\widetilde{R}_X = r \mid X\right)\mathbb{P}\left(\widetilde{N}^r_X = n \mid X\right)\mathbb{P}\left(\widetilde{N}^p_X = n' \mid X\right) \tag{3.67}$$

where we use the convention that $\widetilde{R}_X = r$ implies that $\widetilde{R}_X(s,a) = r_{x,a}$, and similarly for $\widetilde{N}^r_X = n, \widetilde{N}^p_X = n'$. Therefore using inequalities (3.63), (3.64) and (3.65) in (3.66), we have:

$$\begin{aligned}
\mathbb{P}\Big(\mathcal{M}(X) = (r,n,n') \mid X\Big) &= \mathbb{P}\left(\widetilde{R}_X = r \mid X\right)\mathbb{P}\left(\widetilde{N}^r_X = n \mid X\right)\mathbb{P}\left(\widetilde{N}^p_X = n' \mid X\right) \\
&\leq \exp(\varepsilon)\mathbb{P}\left(\widetilde{R}_{X'} = r \mid X'\right)\mathbb{P}\left(\widetilde{N}^r_{X'} = n \mid X'\right)\mathbb{P}\left(\widetilde{N}^p_{X'} = n' \mid X'\right) \\
&= \exp(\varepsilon)\mathbb{P}\left(\widetilde{R}_{X'} = r, \widetilde{N}^r_{X'} = n, \widetilde{N}^p_{X'} = n' \mid X'\right) \\
&= \exp(\varepsilon)\mathbb{P}\left(\mathcal{M}(X') = (r,n,n') \mid X'\right)
\end{aligned}$$

This concludes the proof. □

Now that we shown the Laplace mechanism ensures LDP with the reight parameter, let's show that the latter satisfies Asm. 11 by showing the following proposition:

Proposition 9. *For any $\varepsilon > 0$, the Laplace mechnism, Alg. 13, with parameter $\varepsilon_0 = \varepsilon/(6H)$ satisfies Def. 11 for any $\delta > 0$ and $k \in \mathbb{N}$ with $c_{k,1}(\varepsilon,\delta) = c_{k,2}(\varepsilon,\delta)$, $c_{k,3}(\varepsilon,\delta) = \sqrt{S}c_{k,4}(\varepsilon,\delta)$ and:*

$$c_{k,1}(\varepsilon,\delta) = \max\left\{\sqrt{k}, \ln\left(\frac{6SA}{\delta}\right)\right\}\frac{\sqrt{8\ln\left(\frac{6SA}{\delta}\right)}}{\varepsilon/6H},$$

$$c_{k,3}(\varepsilon,\delta) = \max\left\{\sqrt{kS}, \ln\left(\frac{6S^2A}{\delta}\right)\right\}\frac{\sqrt{8\ln\left(\frac{6S^2A}{\delta}\right)}}{\varepsilon/6H}$$

Before proving Prop. 9 we state the following concentration inequality for the sum of Laplace variables.

Proposition 10. *(Dwork and Roth, 2014, Cor. 12.3) Let $Y_1, \dots, Y_k$ be independent Lap(b) random variables with $b > 0$ and $\delta \in (0,1)$ then for any $\nu > b\max\left\{\sqrt{k}, \sqrt{\ln(2/\delta)}\right\}$,*

$$\mathbb{P}\left(\left|\sum_{l=1}^k Y_l\right| > \nu\sqrt{8\ln(2/\delta)}\right) \leq \delta$$

We can now prove Prop. 9 that shows that Alg. 13 satisfies Def. 11.

Proof of Prop. 9. Let $X_1, \dots, X_{k-1}$ be the $k-1$ trajectories generated before episode $k \geq 1$. Consider the private statistic $\widetilde{R}_k(s,a)$ generated by the private randomizer before episode k. Then for any state-action pair $(s,a) \in \mathcal{S}\times\mathcal{A}$:

$$\begin{aligned}
\left|\widetilde{R}_k(s,a) - R_k(s,a)\right| &= \left|\sum_{l<k}(\widetilde{R}_{X_l}(s,a) - R_{X_l}(s,a))\right| \\
&= \left|\sum_{l<k}\left(Y_{1,X_l}(s,a) + \sum_{h=1}^H r_h \mathbb{1}_{\left\{\substack{s_{l,h}=s,\\ a_{l,h}=a}\right\}}\right) - \sum_{l<k}\sum_{h=1}^H r_h \mathbb{1}_{\left\{\substack{s_{l,h}=s,\\ a_{l,h}=a}\right\}}\right| \\
&= \left|\sum_{l=1}^{k-1} Y_{1,X_l}(s,a)\right|
\end{aligned}$$

Algorithm 14: Gaussian mechanism for LDP

Input: Trajectory: $X = \{(s_h, a_h, r_h) \mid h \leq H\}$, Privacy Parameter: ε_0, c

Draw $(Y_{i,X}(s,a))_{(s,a)\in\mathcal{S}\times\mathcal{A}, i\leq 2}$ i.i.d $\mathcal{N}\left(0,\sigma^2\right)$ and $(Z_X(s,a,s'))_{(s,a,s')\in\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$ i.i.d $\mathcal{N}\left(0,\sigma^2\right)$ and independent from $Y_{i,X}$ for $i \in \{1,2\}$ with $\sigma = cH/\varepsilon_0$

for $(s,a) \in \mathcal{S}\times\mathcal{A}$ **do**

$\widetilde{R}_X(s,a) = \sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s, a_h=a\}} + Y_{1,X}(s,a)$

$\widetilde{N}^r_X(s,a) = \sum_{h=1}^{H} \mathbb{1}_{\{s_h=s, a_h=a\}} + Y_{2,X}(s,a)$

for $s' \in \mathcal{S}$ **do**

$\widetilde{N}^p_X(s,a,s') = \sum_{h=1}^{H-1} \mathbb{1}_{\{s_h=s, a_h=a, s_{h+1}=s'\}} + Z_X(s,a,s')$

Output: $(\widetilde{R}_X, \widetilde{N}^r_X, \widetilde{N}^p_X) \in \mathbb{R}^{S\times A} \times \mathbb{R}^{S\times A} \times \mathbb{R}^{S\times A\times S}$

which is the sum of independent Laplace variables. Let $\delta > 0$. By Prop. 10 we have that with probability at least $1 - \delta/(3SA)$

$$\left|\sum_{l=1}^{k-1} Y_{1,X_l}(s,a)\right| \leq \frac{1}{\varepsilon_0} \max\left\{\sqrt{k-1}, \ln\left(\frac{6SA}{\delta}\right)\right\} \sqrt{8\ln\left(\frac{6SA}{\delta}\right)} \tag{3.68}$$

The same property holds for $\widetilde{N}^r_k$ and $\widetilde{N}^p_k$ and we again apply Prop. 10. Properties in Def. 11 follow from union bounds. □

3.A.6 Other Privacy Preserving Mechanisms

We have shown in App. 3.A.5.1 that the Laplace mechanism, Alg. 13, satisfies Def. 11. However it is not the only mechanism to do so. In this appendix we present the Gaussian, Randomized Response and bounded noise mechanisms and show that these also satisfy Def. 11.

3.A.6.1 Gaussian Mechanism:

The Gaussian mechanism is a fundamental mechanism in the differential privacy literature (see e.g., Dwork and Roth, 2014). However, contrary to the Laplace mechanism the Gaussian mechanism can only guarantees (ε,δ)-LDP for $\delta > 0$. The mechanism is based on the same idea as the Laplace mechanism, that is to say it adds Gaussian noise to the result of a given computation on the input data. This noise is centered and the standard deviation $\sigma(\varepsilon,\delta)$ is $\frac{cH}{\epsilon_0}$.

In the following, we show that the Gaussian mechanism almost satisfies Def. 11. The Gaussian mechanism can not guarantee $(\varepsilon_0, 0)$-LDP for any $\varepsilon_0 > 0$, however we show that it satisfies the other necessary conditions, including (ε_0,δ)-LDP for any $\delta > 0$. First, we show that the mechanism guarantees Local Differential Privacy for high enough noise.

Proposition 11. *For any $1 \geq \varepsilon_0 > 0$ and $\delta_0 > 0$ and parameter $c > 4\ln\left(\frac{24}{\delta_0}\right)$, the Gaussian mechanism, Alg. 14, is (ε_0,δ_0)-LDP.*

Proof of Prop. 11: The proof is based on the proof presented in (Dwork and Roth, 2014). Similarly to the proof of Prop. 9 let's consider two trajectories $X = \{(s_h, a_h, r_h) \mid h \leq H\}$ and $X' = \{(s'_h, a'_h, r'_h) \mid h \leq H\}$ and also denote the output of the private randomizer $\mathcal{M}$ by $\mathcal{M}(X) = (\widetilde{R}_X, \widetilde{N}^r_X, \widetilde{N}^p_X)$ and $\mathcal{M}(X') = (\widetilde{R}_{X'}, \widetilde{N}^r_{X'}, \widetilde{N}^p_{X'})$.

For a given vector $r \in \mathbb{R}^{S\times A}$,

$$\frac{\mathbb{P}\left(\forall(s,a), \widetilde{R}_X(s,a) = r_{s,a} \mid X\right)}{\mathbb{P}\left(\forall(s,a), \widetilde{R}_{X'}(s,a) = r_{s,a} \mid X'\right)} = \prod_{s,a} \frac{\mathbb{P}\left(Y_{1,X}(s,a) = \sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} - r_{s,a} \mid X\right)}{\mathbb{P}\left(Y_{1,X'}(s,a) = \sum_{h=1}^{H} r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}} - r_{s,a} \mid X'\right)} \tag{3.69}$$

since the Gaussian distribution is symmetric. Then,

$$\begin{aligned} &\prod_{s,a} \frac{\mathbb{P}\left(Y_{1,X}(s,a) = \sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} - r_{s,a} \mid X\right)}{\mathbb{P}\left(Y_{1,X'}(s,a) = \sum_{h=1}^{H} r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}} - r_{s,a} \mid X'\right)} \\ &= \prod_{s,a} \exp\left(\frac{\left(\sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} - r_{s,a}\right)^2 - \left(\sum_{h=1}^{H} r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}} - r_{s,a}\right)^2}{2\sigma^2}\right) \end{aligned} \tag{3.70}$$

But, considering the squared term, we get

$$\left(\sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}} - r_{s,a}\right)^2 = \left(\sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}} - \sum_{h=1}^{H} r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}} + \sum_{h=1}^{H} r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}} - r_{s,a}\right)^2$$
$$= \left(\sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}} - \sum_{h=1}^{H} r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}}\right)^2 + \left(\sum_{h=1}^{H} r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}} - r_{s,a}\right)^2$$
$$+ 2\left(\sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}} - \sum_{h=1}^{H} r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}}\right)\left(\sum_{h=1}^{H} r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}} - r_{s,a}\right)$$

Hence we get that

$$\text{(3.70)} = \prod_{s,a} \exp\left(\frac{1}{2\sigma^2}\left(\left(\sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}} - \sum_{h=1}^{H} r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}}\right)^2 \right.\right.$$
$$\left.\left. - 2\left(\sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}} - r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}}\right)\left(\sum_{h=1}^{H} r'_h \mathbb{1}_{\left\{\substack{s'_h=s,\\ a'_h=a}\right\}} - r_{s,a}\right)\right)\right). \tag{3.71}$$

But, $\sum_{s,a}\left(\sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} - \sum_{h=1}^{H} r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}}\right)^2 \leq 2H^2$ because for each step h, $r_h \in [0,1]$. By the same reasonning, we have $\sum_{s,a}\left|\left(\sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} - r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}}\right)\sum_{h=1}^{H} r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}}\right| \leq H^2$. Therefore, we have:

$$\text{(3.70)} \leq \exp\left(\frac{1}{2\sigma^2}\left(2\sum_{s,a}\left(\sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} - r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}}\right) r_{s,a} + 3H^2\right)\right)$$
$$\leq \exp\left(\frac{1}{2\sigma^2}\left(2\sqrt{2}H\sqrt{\sum_{s,a} r_{s,a}^2} + 3H^2\right)\right) \tag{3.72}$$

where the last inequality follows from Cauchy-Schwartz. Note that if $||r||_2 \leq \frac{\sigma^2\varepsilon_0}{3\sqrt{2}H} - \frac{3H}{2\sqrt{2}}$, Eq. (3.72) is bounded by $\exp(\varepsilon_0/3)$. Therefore, to finish, we partition $\mathbb{R}^{S\times A}$ in two subspaces $R_1 = \left\{x \in \mathbb{R}^{S\times A} \mid ||x||_2 \leq \frac{c^2 H}{3\sqrt{2}\varepsilon_0} - \frac{3H}{2\sqrt{2}}\right\}$ and $R_2 = \left\{x \in \mathbb{R}^{S\times A} \mid ||x||_2 > \frac{c^2 H}{3\sqrt{2}\varepsilon_0} - \frac{3H}{2\sqrt{2}}\right\}$ where we used the fact that $\sigma = cH/\varepsilon_0$ with c a constant to be chosen later. Then for $c^2 \geq 4\ln\left(\frac{3}{\delta_1}\right)$, for δ_1 to be chosen later, $\mathbb{P}\left(Y_{1,X} \in R_2\right) \leq \delta_1$ and $\mathbb{P}\left(Y_{1,X'} \in R_2\right) \leq \delta_1$. Thus for Eq. (3.69):

$$\mathbb{P}\left(\forall(s,a), \widetilde{R}_X(s,a) = r_{s,a} \mid X\right) = \mathbb{P}\left(\forall(s,a), \widetilde{R}_X(s,a) = r_{s,a} \mid X\right) \mathbb{1}_{\{r-(\sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}})_{s,a} \in R_1\}} \tag{3.73}$$
$$+ \mathbb{P}\left(\forall(s,a), \widetilde{R}_X(s,a) = r_{s,a} \mid X\right) \mathbb{1}_{\{r-(\sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_h=s,\\ a_h=a}\right\}})_{s,a} \in R_2\}}$$
$$\leq e^{\frac{\varepsilon_0}{3}} \mathbb{P}\left(\forall(s,a), \widetilde{R}_{X'}(s,a) = r_{s,a} \mid X'\right) \mathbb{1}_{\{r-(\sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s,\\ a_h=a\}})_{s,a} \in R_1\}} \tag{3.74}$$
$$+ \mathbb{P}\left(Y_{1,X} \in R_2\right)$$
$$\leq \exp(\varepsilon_0/3)\mathbb{P}\left(\forall(s,a), \widetilde{R}_{X'}(s,a) = r_{s,a} \mid X'\right) + \delta_1 \tag{3.75}$$

We get the same results for $\widetilde{N}^r$ and $\widetilde{N}^p$. Then, because $(Y_{i,X}(s,a))_{i\leq 2,(s,a)\in\mathcal{S}\times\mathcal{A}}$, $(Z_X(s,a,s'))_{(s,a,s')\in\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$ are independent, see Alg. 14 it holds that:

$$\mathbb{P}\left(\widetilde{R}_X = r, \widetilde{N}^r_X = n, \widetilde{N}^p_X = n' \mid X\right) = \mathbb{P}\left(\widetilde{R}_X = r \mid X\right)\mathbb{P}\left(\widetilde{N}^r_X = n \mid X\right)\mathbb{P}\left(\widetilde{N}^p_X = n' \mid X\right)$$

and so,

$$\mathbb{P}\Big(\mathcal{M}(X) = (r,n,n') \mid X\Big) = \mathbb{P}\left(\widetilde{R}_X = r, \widetilde{N}^r_X = n, \widetilde{N}^p_X = n' \mid X\right)$$
$$= \mathbb{P}\left(\widetilde{R}_X = r \mid X\right)\mathbb{P}\left(\widetilde{N}^r_X = n \mid X\right)\mathbb{P}\left(\widetilde{N}^p_X = n' \mid X\right)$$

Then for any two trajectories X and X', we have:

$$\begin{aligned}\mathbb{P}\left(\widetilde{R}_X = r \mid X\right)\mathbb{P}\left(\widetilde{N}^r_X = n \mid X\right)\mathbb{P}\left(\widetilde{N}^p_X = n' \mid X\right) &\leq \left(e^{\frac{\varepsilon_0}{3}}\mathbb{P}\left(\widetilde{R}_{X'} = r \mid X'\right) + \delta_1\right)\\ &\times \left(e^{\frac{\varepsilon_0}{3}}\mathbb{P}\left(\widetilde{N}^r_{X'} = n \mid X'\right) + \delta_1\right)\\ &\times \left(e^{\frac{\varepsilon_0}{3}}\mathbb{P}\left(\widetilde{N}^p_{X'} = n' \mid X'\right) + \delta_1\right)\\ \leq e^{\varepsilon_0}\mathbb{P}\left(\widetilde{R}_{X'} = r \mid X'\right)\mathbb{P}\left(\widetilde{N}^r_{X'} = n \mid X'\right)\mathbb{P}\left(\widetilde{N}^p_{X'} = n' \mid X'\right) &+ 2\delta_1\exp\left(2\varepsilon_0/3\right)\\ &+2\delta_1^2\exp\left(\varepsilon_0/3\right) + \delta_1^3\end{aligned}$$

Thus by choosing $\delta_1 = \delta_0/8$, it holds that $2\delta_1\exp\left(2\varepsilon_0/3\right) + 2\delta_1^2\exp\left(\varepsilon_0/3\right) + \delta_1^3 \leq \delta_0$ for $\varepsilon_0 \leq 1$, and so we can conclude that the Gaussian mechanism is $(\varepsilon_0, \delta_0)$-LDP. □

In addition, the precision of the Gaussian mechanism is of the same order as the Laplace mechanism, that is to say:

Proposition 12. *The Gaussian mechanism, Alg. 14, with parameter $\varepsilon_0 > 0$ and $c^2 \geq 4\ln\left(\frac{24}{\delta_0}\right)$ for any $\delta_0 > 0$ satisfies Def. 11 for any $\delta > 0$ and $k \in \mathbb{N}^\star$ with:*

$$c_{k,1}(\varepsilon_0, \delta_0, \delta) = c_{k,2}(\varepsilon_0, \delta_0, \delta) = c_{k,4}(\varepsilon_0, \delta_0, \delta) = \max\left\{\frac{cH}{\varepsilon_0}\sqrt{(k-1)\ln\left(\frac{6SA}{\delta}\right)}, 1\right\}$$

$$c_{k,3}(\varepsilon_0, \delta_0, \delta) = \max\left\{\frac{cH}{\varepsilon_0}\sqrt{(k-1)S\ln\left(\frac{6SA}{\delta}\right)}, 1\right\}$$

This result shows that using the Gaussian mechanism rather than the Laplace mechanism would not lead to improved regret rate as the utilities $c_{k,1}, c_{k,2}, c_{k,3}, c_{k,4}$ have the same depency of S, A, H, ε_0 and k . Moreover, the Gaussian mechanism only guarantees LDP for $\delta > 0$ whereas using the Laplace mechanism ensures that we can guarantee LDP for $\delta = 0$ as well.

Proof of Prop. 12: Following the same steps as in the proof of Prop 9, we have that at the beginning of episode k with probability at least $1 - \frac{\delta}{3SA}$:

$$\left|\widetilde{R}_k(s,a) - R_k(s,a)\right| = \left|\sum_{l<k}(\widetilde{R}_{X_l}(s,a) - R_{X_l}(s,a))\right| \tag{3.76}$$

$$= \left|\sum_{l<k}\left(Y_{1,X_l}(s,a) + \sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_{l,h}=s,\\ a_{l,h}=a}\right\}}\right) - \sum_{l<k}\sum_{h=1}^{H} r_h \mathbb{1}_{\left\{\substack{s_{l,h}=s,\\ a_{l,h}=a}\right\}}\right| \tag{3.77}$$

$$= \left|\sum_{l=1}^{k-1} Y_{1,X_l}(s,a)\right| \leq \sigma\sqrt{2(k-1)\ln\left(\frac{6SA}{\delta}\right)} \tag{3.78}$$

for $\sigma = cH/\varepsilon_0$ thanks to Chernoff bounds. The same result follows for $\widetilde{N}^r$ and $\widetilde{N}^p$. Therefore, the Gaussian mechanism satisfies Def. 11 with $c_{k,1}(\varepsilon_0, \delta_0, \delta) = c_{k,2}(\varepsilon_0, \delta_0, \delta) = c_{k,4}(\varepsilon_0, \delta_0, \delta)$ with:

$$c_{k,1}(\varepsilon_0, \delta_0, \delta) = \max\left\{\frac{cH}{\varepsilon_0}\sqrt{(k-1)\ln\left(\frac{6SA}{\delta}\right)}, 1\right\} \tag{3.79}$$

with $c > 0$ and:

$$c_{k,3}(\varepsilon_0, \delta_0, \delta) = \max\left\{\frac{cH}{\varepsilon_0}\sqrt{(k-1)S\ln\left(\frac{6SA}{\delta}\right)}, 1\right\} \tag{3.80}$$

where $c_{k,3}(\varepsilon_0, \delta_0, \delta)$ is defined such that $\left|\sum_{s'} N^p_k(s,a,s') - \sum_{s'} \widetilde{N}^p_k(s,a,s')\right| \leq c_{k,3}(\varepsilon_0, \delta_0, \delta)$. □

Algorithm 15: Randomized Response mechanism for LDP

Input: Trajectory: $X = \{(s_h, a_h, r_h) \mid h \leq H\}$, Privacy Parameter: ε_0

Draw $(Y_{i,X}(s,a))_{(s,a)\in\mathcal{S}\times\mathcal{A}, i\leq 2}$ i.i.d $\mathcal{N}\left(0,\sigma^2\right)$ and $(Z_X(s,a,s'))_{(s,a,s')\in\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$ i.i.d $\mathcal{N}\left(0,\sigma^2\right)$ and independent from $Y_{i,X}$ for $i \in \{1,2\}$ with $\sigma = cH/\varepsilon_0$

for $(s,a) \in \mathcal{S}\times\mathcal{A}$ **do**

  for $h = 1, \ldots, H$ **do**

    Sample $Y_{1,X}(h,s,a) \sim \text{Ber}\left(\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1} r_h \mathbb{1}_{\{s_h=s, a_h=a\}} + \frac{1}{e^{\varepsilon_0}+1}\right)$

    $\widetilde{R}_X(h,s,a) = \frac{e^{\varepsilon_0}+1}{e^{\varepsilon_0}-1}\left(Y_{1,X}(h,s,a) - \frac{1}{e^{\varepsilon_0}+1}\right)$

    Sample $\widetilde{n}^r_X(h,s,a) \sim \text{Ber}\left(\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1}\mathbb{1}_{\{s_h=s,a_h=a\}} + \frac{1}{e^{\varepsilon_0}+1}\right)$

    if $h < H$ **then**

      for $s' \in \mathcal{S}$ **do**

        Sample $\widetilde{n}^p_X(h,s,a,s') \sim \text{Ber}\left(\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1}\mathbb{1}_{\{s_h=s,a_h=a,s_{h+1}=s'\}} + \frac{1}{e^{\varepsilon_0}+1}\right)$

        $\widetilde{N}^p_X(h,s,a,s') = \frac{e^{\varepsilon_0}+1}{e^{\varepsilon_0}-1}\left(\widetilde{n}^p_X(h,s,a,s') - \frac{1}{e^{\varepsilon_0}+1}\right)$

Output: $(\widetilde{R}_X, \widetilde{N}^r_X, \widetilde{N}^p_X) \in \left\{\frac{-1}{e^{\varepsilon_0}-1}, \frac{e^{\varepsilon_0}}{e^{\varepsilon_0}-1}\right\}^{HSA} \times \left\{\frac{-1}{e^{\varepsilon_0}-1}, \frac{e^{\varepsilon_0}}{e^{\varepsilon_0}-1}\right\}^{HSA} \times \left\{\frac{-1}{e^{\varepsilon_0}-1}, \frac{e^{\varepsilon_0}}{e^{\varepsilon_0}-1}\right\}^{(H-1)SAS}$

3.A.6.2 Randomized Response Mechanism:

The second alternative mechanism we consider is the Randomized Response mechanism. In general, it is used for discrete data like indicator functions $(\mathbb{1}_{\{s_h=s,a_h=a\}})_{h,s,a}$. We therefore use it to privatize the number of visits of a state-action pair and state-action-next-state tuple for each trajectory. With the assumption that reward are supported in $[0,1]$, we can also use this mechanism for privatizing the cumulative reward of a given trajectory. Contrary to previous ones, the output of the Randomized Response mechanism is three vectors, two of size $H \times S \times A$, and the last one of size $(H-1)\times S \times A \times S$. We slightly modify the requirements of Def. 11 by changing the size of the output of the privacy preserving mechanism. We summarize the mechanism in Alg. 15.

Just as for the Gaussian mechanism, we show that Alg. 15 satisfies Def. 11. We begin by showing that this mechanism satisfies $(\varepsilon_0, 0)$-LDP for any $\varepsilon_0 > 0$.

Proposition 13. *For any $\varepsilon > 0$, the Randomized Response mechanism, Alg. 15, with parameter $\varepsilon_0 = \varepsilon/6H$ is $(\varepsilon, 0)$-LDP.*

Proof of Prop. 13: Just as in the proof of Prop. 11 and Prop. 9, let's consider two trajectories $X = \{(s_h, a_h, r_h) \mid h \leq H\}$ and $X' = \{(s'_h, a'_h, r'_h) \mid h \leq H\}$ and also denote the output of the private randomizer $\mathcal{M}$ by $\mathcal{M}(X) = (\widetilde{R}_X, \widetilde{N}^r_X, \widetilde{N}^p_X)$ and $\mathcal{M}(X') = (\widetilde{R}_{X'}, \widetilde{N}^r_{X'}, \widetilde{N}^p_{X'})$.

For a given $r \in \left\{\frac{-1}{e^{\varepsilon_0}-1}, \frac{e^{\varepsilon_0}}{e^{\varepsilon_0}-1}\right\}^{HSA}$ (note that by definition of r in Alg. 15, these are the only values it can take), we have that:

$$\frac{\mathbb{P}\left(\forall(h,s,a), \widetilde{R}_X(h,s,a) = r_{h,s,a} \mid X\right)}{\mathbb{P}\left(\forall(h,s,a), \widetilde{R}_{X'}(h,s,a) = r_{h,s,a} \mid X'\right)} = \prod_{h,s,a}\left(\frac{\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} + \frac{1}{e^{\varepsilon_0}+1}}{\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1} r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}} + \frac{1}{e^{\varepsilon_0}+1}}\right)^{y^r_{h,s,a}} \times \times \left(\frac{1 - \left(\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1} r_h \mathbb{1}_{\{s_h=s,a_h=a\}} + \frac{1}{e^{\varepsilon_0}+1}\right)}{1 - \left(\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1} r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}} + \frac{1}{e^{\varepsilon_0}+1}\right)}\right)^{1-y^r_{h,s,a}} \tag{3.81}$$

where for every $(h,s,a) \in H\times\mathcal{S}\times\mathcal{A}$, we define $y^r_{h,s,a} = \frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1} r + \frac{1}{e^{\varepsilon_0}+1}$ belongs to $\{0,1\}$ because $r \in \left\{\frac{-1}{e^{\varepsilon_0}-1}, \frac{e^{\varepsilon_0}}{e^{\varepsilon_0}-1}\right\}^{HSA}$. Eq. (3.81) can be rewritten as:

$$(3.81) = \prod_{h,s,a}\left(\frac{(e^{\varepsilon_0}-1) r_h \mathbb{1}_{\{s_h=s,a_h=a\}} + 1}{(e^{\varepsilon_0}-1) r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}} + 1}\right)^{y^r_{h,s,a}} \left(\frac{e^{\varepsilon_0} - (e^{\varepsilon_0}-1) r_h \mathbb{1}_{\{s_h=s,a_h=a\}}}{e^{\varepsilon_0} - (e^{\varepsilon_0}-1) r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}}}\right)^{1-y^r_{h,s,a}} \tag{3.82}$$

Then for a given (h, s, a), because $r_h \in [0, 1]$ we have:

$$\frac{(e^{\varepsilon_0}-1)r_h \mathbb{1}_{\{s_h=s,a_h=a\}}+1}{(e^{\varepsilon_0}-1)r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}}+1} \leq \begin{cases} e^{\varepsilon_0} & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 1 \\ 1 & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 0 \\ e^{\varepsilon_0} & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = 1 \text{ and } \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 0 \\ 1 & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = 0 \text{ and } \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 1 \end{cases} \tag{3.83}$$

$$\frac{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)r_h \mathbb{1}_{\{s_h=s,a_h=a\}}}{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}}} \leq \begin{cases} e^{\varepsilon_0} & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 1 \\ 1 & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 0 \\ 1 & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = 1 \text{ and } \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 0 \\ e^{\varepsilon_0} & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = 0 \text{ and } \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 1 \end{cases} \tag{3.84}$$

Therefore, we can simplify each term in (3.82) by:

$$\frac{(e^{\varepsilon_0}-1)r_h \mathbb{1}_{\{s_h=s,a_h=a\}}+1}{(e^{\varepsilon_0}-1)r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}}+1} \leq \exp\left(\varepsilon_0\left(\mathbb{1}_{\{s_h=s,a_h=a\}}+\mathbb{1}_{\{s'_h=s,a'_h=a\}}\right)\right)$$
$$\frac{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)r_h \mathbb{1}_{\{s_h=s,a_h=a\}}}{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)r'_h \mathbb{1}_{\{s'_h=s,a'_h=a\}}} \leq \exp\left(\varepsilon_0\left(\mathbb{1}_{\{s_h=s,a_h=a\}}+\mathbb{1}_{\{s'_h=s,a'_h=a\}}\right)\right)$$

Hence, using the two inequalities above:

$$\begin{aligned}(3.82) &\leq \prod_{h,s,a} \exp\left(y^r_{h,s,a}\varepsilon_0\left(\mathbb{1}_{\left\{\substack{s_h=s,\\a_h=a}\right\}}+\mathbb{1}_{\left\{\substack{s'_h=s,\\a'_h=a}\right\}}\right)+(1-y^r_{h,s,a})\varepsilon_0\left(\mathbb{1}_{\left\{\substack{s'_h=s,\\a'_h=a}\right\}}+\mathbb{1}_{\left\{\substack{s_h=s,\\a_h=a}\right\}}\right)\right) \\ &= \prod_{h,s,a} \exp\left(\varepsilon_0\left(\mathbb{1}_{\left\{\substack{s_h=s,\\a_h=a}\right\}}+\mathbb{1}_{\left\{\substack{s'_h=s,\\a'_h=a}\right\}}\right)\right) \\ &= \exp\left(2\varepsilon_0 H\right)\end{aligned}$$

In addition, let's consider $m \in \left\{\frac{-1}{e^{\varepsilon_0}-1}, \frac{e^{\varepsilon_0}}{e^{\varepsilon_0}-1}\right\}^{H\times S\times A}$ and $y = \frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1}m + \frac{1}{e^{\varepsilon_0}+1} \in \{0, 1\}$, we then have that:

$$\begin{aligned}\frac{\mathbb{P}\left(\forall(h,s,a), \widetilde{N}^r_X(h,s,a) = m_{h,s,a} \mid X\right)}{\mathbb{P}\left(\forall(h,s,a), \widetilde{N}^r_{X'}(h,s,a) = m_{h,s,a} \mid X'\right)} &= \prod_{h,s,a}\left(\frac{\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1}\mathbb{1}_{\{s_h=s,a_h=a\}}+\frac{1}{e^{\varepsilon_0}+1}}{\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1}\mathbb{1}_{\{s'_h=s,a'_h=a\}}+\frac{1}{e^{\varepsilon_0}+1}}\right)^{y_{h,s,a}} \times \\ &\times\left(\frac{1-\left(\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1}\mathbb{1}_{\{s_h=s,a_h=a\}}+\frac{1}{e^{\varepsilon_0}+1}\right)}{1-\left(\frac{e^{\varepsilon_0}-1}{e^{\varepsilon_0}+1}\mathbb{1}_{\{s'_h=s,a'_h=a\}}+\frac{1}{e^{\varepsilon_0}+1}\right)}\right)^{1-y_{h,s,a}}\end{aligned} \tag{3.85}$$

Which can be rewritten as:

$$\begin{aligned}\frac{\mathbb{P}\left(\forall(h,s,a), \widetilde{N}^r_X(h,s,a) = m_{h,s,a} \mid X\right)}{\mathbb{P}\left(\forall(h,s,a), \widetilde{N}^r_{X'}(h,s,a) = m_{h,s,a} \mid X'\right)} &= \prod_{h,s,a}\left(\frac{(e^{\varepsilon_0}-1)\mathbb{1}_{\{s_h=s,a_h=a\}}+1}{(e^{\varepsilon_0}-1)\mathbb{1}_{\{s'_h=s,a'_h=a\}}+1}\right)^{y_{h,s,a}} \times \\ &\times\left(\frac{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)\mathbb{1}_{\{s_h=s,a_h=a\}}}{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)\mathbb{1}_{\{s'_h=s,a'_h=a\}}}\right)^{1-y_{h,s,a}}\end{aligned} \tag{3.86}$$

Thus for a given (h, s, a):

$$\frac{(e^{\varepsilon_0}-1)\mathbb{1}_{\{s_h=s,a_h=a\}}+1}{(e^{\varepsilon_0}-1)\mathbb{1}_{\{s'_h=s,a'_h=a\}}+1} = \begin{cases} 1 & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = \mathbb{1}_{\{s'_h=s,a'_h=a\}} \\ e^{\varepsilon_0} & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = 1 \text{ and } \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 0 \\ e^{-\varepsilon_0} & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = 0 \text{ and } \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 1 \end{cases} \tag{3.87}$$

$$\frac{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)\mathbb{1}_{\{s_h=s,a_h=a\}}}{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)\mathbb{1}_{\{s'_h=s,a'_h=a\}}} = \begin{cases} 1 & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = \mathbb{1}_{\{s'_h=s,a'_h=a\}} \\ e^{-\varepsilon_0} & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = 1 \text{ and } \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 0 \\ e^{\varepsilon_0} & \text{if } \mathbb{1}_{\{s_h=s,a_h=a\}} = 0 \text{ and } \mathbb{1}_{\{s'_h=s,a'_h=a\}} = 1 \end{cases} \tag{3.88}$$

Therefore, here again we can simplify each term in (3.86) by:

$$\frac{(e^{\varepsilon_0}-1)\mathbb{1}_{\{s_h=s,a_h=a\}}+1}{(e^{\varepsilon_0}-1)\mathbb{1}_{\{s'_h=s,a'_h=a\}}+1} \leq \exp\left(\varepsilon_0\left(\mathbb{1}_{\{s_h=s,a_h=a\}}-\mathbb{1}_{\{s'_h=s,a'_h=a\}}\right)\right)$$
$$\frac{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)\mathbb{1}_{\{s_h=s,a_h=a\}}}{e^{\varepsilon_0}-(e^{\varepsilon_0}-1)\mathbb{1}_{\{s'_h=s,a'_h=a\}}} \leq \exp\left(\varepsilon_0\left(\mathbb{1}_{\{s_h=s,a_h=a\}}-\mathbb{1}_{\{s'_h=s,a'_h=a\}}\right)\right)$$

Therefore:

$$\begin{aligned}(3.86) &= \prod_{h,s,a}\exp\left(y_{h,s,a}\varepsilon_0\left(\mathbb{1}_{\left\{\substack{s_h=s,\\a_h=a}\right\}}-\mathbb{1}_{\left\{\substack{s'_h=s,\\a'_h=a}\right\}}\right)+(1-y_{h,s,a})\varepsilon_0\left(\mathbb{1}_{\left\{\substack{s'_h=s,\\a'_h=a}\right\}}-\mathbb{1}_{\left\{\substack{s_h=s,\\a_h=a}\right\}}\right)\right)\\ &= \prod_{h,s,a}\exp\left((2y_{h,s,a}-1)\varepsilon_0\left(\mathbb{1}_{\{s_h=s,a_h=a\}}-\mathbb{1}_{\{s'_h=s,a'_h=a\}}\right)\right)\\ &\leq \exp\left(2\varepsilon_0 H\right)\end{aligned}$$

Using the same reasonning we have that for any $m' \in \left\{-\frac{1}{e^{\varepsilon_0}-1}, \frac{e^{\varepsilon_0}}{e^{\varepsilon_0}-1}\right\}^{(H-1)\times S\times A\times S}$:

$$\frac{\mathbb{P}\left(\forall(h,s,a,s'), \widetilde{N}^p_X(h,s,a,s') = m'_{h,s,a,s'} \mid X\right)}{\mathbb{P}\left(\forall(h,s,a,s'), \widetilde{N}^p_{X'}(h,s,a,s') = m'_{h,s,a,s'} \mid X'\right)} \leq \exp(2\varepsilon_0 H) \tag{3.89}$$

We conclude the proof the same way as the proof of Prop. 10. □

In addition, the precision $c_{k,1}$, $c_{k,2}$, $c_{k,3}$ and $c_{k,4}$ of the Randomized Response mechanism are still of order $\sqrt{k}$ just as the Gaussian and Laplace mechanisms. Contrary to any of those two, the dependence is exponential on ε_0 which is closer to the lower bound of Sec. 3.1.2. Indeed, we have an additional factor S for $c_{k,3}$ compared to the other mechanisms but those terms scale with $1/(e^{\varepsilon_0}-1)$ instead of the worse dependency $1/\varepsilon$.

Proposition 14. *The Randomized Response mechanism, Alg. 15, with parameter $\varepsilon_0 > 0$ satisfies Def. 11 for any $\delta > 0$ and $k \in \mathbb{N}^\star$ with:*

$$c_{k,1}(\varepsilon_0,\delta) = c_{k,2}(\varepsilon_0,\delta) = \max\left\{1, \frac{2e^{\varepsilon_0}-1}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4SA}{\delta}\right)}\right\}$$
$$c_{k,3}(\varepsilon_0,\delta) = \max\left\{1, \frac{S(2e^{\varepsilon_0}-1)}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4SA}{\delta}\right)}\right\}$$
$$c_{k,4}(\varepsilon_0,\delta) = \max\left\{1, \frac{2e^{\varepsilon_0}-1}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4S^2A}{\delta}\right)}\right\}$$

Proof of Prop. 14: Let's consider a given state-action-next state tuple, (s,a,s'), then when summing over h:

$$\left|\sum_{h=1}^H \widetilde{N}^r_k(h,s,a) - \sum_{l<k}\sum_{h=1}^H \mathbb{1}_{\{s_{l,h}=s,a_{l,h}=a\}}\right| = \left|\sum_{h=1}^H\sum_{l<k}\widetilde{N}^r_{X_l}(h,s,a) - \mathbb{1}_{\{s_{l,h}=s,a_{l,h}=a\}}\right| \tag{3.90}$$

We now construct a filtration $(\mathcal{F}_{k,h})_{k,h}$ such that $(\widetilde{N}^r_{X_l}(h,s,a) - \mathbb{1}_{\{s_{l,h}=s,a_{l,h}=a\}})_{l,h}$ is a Martingale Difference Sequence. For an episode k and step h, define $\mathcal{F}_{k,h} = \sigma(\{(s_{l,j},a_{l,j},r_{l,j})_{j\leq H}, \mathcal{M}((s_{l,j},a_{l,j},r_{l,j})_{j\leq H})\} \mid l<k\} \cup \{(s_{k,j},a_{k,j},r_{k,j})_{j\leq h}\})$ to be the filtration that contains the history before episode k. Then $\mathbb{1}_{\{s_{k,h}=s,a_{k,h}=a\}}$ is $\mathcal{F}_{k,h}$-measurable and thus we have:

$$\begin{aligned}\mathbb{E}\left(\widetilde{N}^r_{X_k}(h,s,a) - \mathbb{1}_{\{s_{k,h}=s,a_{k,h}=a\}} \mid \mathcal{F}_{k,h}\right) = \frac{e^{\varepsilon_0}+1}{e^{\varepsilon_0}-1}\left(\mathbb{E}\left(\widetilde{n}_{X_k}(h,s,a) \mid \mathcal{F}_{k,h}\right) - \frac{1}{e^{\varepsilon_0}+1}\right)\\ -\mathbb{1}_{\{s_{k,h}=s,a_{k,h}=a\}} = 0\end{aligned}$$

where $\tilde{n}_{X_k}(h,s,a)$ is a Randomized Response random variable generated by Alg. 15 for each step h, state s, action a and trajectory X_k. And $\left|\widetilde{N}^r_{X_k}(h,s,a) - \mathbb{1}_{\{s_{k,h}=s,a_{k,h}=a\}}\right| \leq \frac{2e^{\varepsilon_0}-1}{e^{\varepsilon_0}-1}$. Then thanks to Azuma-Hoeffding inequality we have that with probability at least $1-\delta/(4SA)$:

$$\left|\sum_{h=1}^H \widetilde{N}^r_k(h,s,a) - \sum_{l<k}\sum_{h=1}^H \mathbb{1}_{\{s_{l,h}=s,a_{l,h}=a\}}\right| \leq \frac{2e^{\varepsilon_0}-1}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4SA}{\delta}\right)} \tag{3.91}$$

With the same reasonning, we have with probability at least $1-\delta/4S^2A$:

$$\left|\sum_{h=1}^{H}\widetilde{N}_k^p(h,s,a,s')-\sum_{l<k}\sum_{h=1}^{H-1}\mathbb{1}_{\{s_{l,h}=s,a_{l,h}=a,s_{l,h+1}=s'\}}\right|\leq\frac{2e^{\varepsilon_0}-1}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4S^2A}{\delta}\right)} \tag{3.92}$$

Also, we have:

$$\left|\sum_{h=1}^{H}\widetilde{R}_k^r(h,s,a)-\sum_{l<k}\sum_{h=1}^{H}r_h\mathbb{1}_{\{s_{l,h}=s,a_{l,h}=a\}}\right|\leq\frac{2e^{\varepsilon_0}-1}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4SA}{\delta}\right)} \tag{3.93}$$

with $\widetilde{R}_k^r(h,s,a)=\sum_{l<k}\widetilde{R}_{X_l}$. Finally, with probability at least $1-\delta/4SA$:

$$\left|\sum_{h=1}^{H}\sum_{s'}\widetilde{N}_k^p(h,s,a,s')-\sum_{s'}\sum_{l<k}\sum_{h=1}^{H-1}\mathbb{1}_{\left\{\substack{s_{l,h}=s,\\ a_{l,h}=a,\\ s_{l,h+1}=s'}\right\}}\right|\leq\frac{S(2e^{\varepsilon_0}-1)}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4SA}{\delta}\right)} \tag{3.94}$$

Compared to the bounds we derived for previous mechanisms there is an additional factor $\sqrt{S}$. This comes from using a triangular inequality instead of using concentration inequalities like in previous mechanisms. Then thanks to a union bound over the state-action pair and the state-action-next state tuple we have that the Randomized Response mechanism satisfies Def. 11 with:

$$c_{k,1}(\varepsilon_0,\delta)=c_{k,2}(\varepsilon_0,\delta)=\max\left\{1,\frac{2e^{\varepsilon_0}-1}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4SA}{\delta}\right)}\right\} \tag{3.95}$$

$$c_{k,3}(\varepsilon_0,\delta)=\max\left\{1,\frac{S(2e^{\varepsilon_0}-1)}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4SA}{\delta}\right)}\right\}, \tag{3.96}$$

$$c_{k,4}(\varepsilon_0,\delta)=\max\left\{1,\frac{2e^{\varepsilon_0}-1}{e^{\varepsilon_0}-1}\sqrt{\frac{(k-1)H}{2}\ln\left(\frac{4S^2A}{\delta}\right)}\right\} \tag{3.97}$$

□

3.A.6.3 Bounded Noise Mechanism for DP:

Recently, Dagan and Kur (2020) showed how to construct a differential privacy with an almost surely bounded noise mechanism. This mechanism, $\mathcal{M}$, computes an (ε,δ)-DP approximation of the average of a dataset $\mathcal{D}=\{x_1,\dots,x_n\}\subset\mathbb{R}^{n\times k}$, for any $\varepsilon>0$ and $\delta\in[\exp(-k/\log(k)^8),1/2]$ (see Theorem 1.1 in (Dagan and Kur, 2020)). In the local differentially private setting in RL, we apply this bounded noise mechanism to each user k in order to compute the cumulative reward for each state-action (s,a), the number of visits to (s,a) and the number of visits to state-action-next state tuple (s,a,s').

This noise mechanism is similar to the Laplace or Gaussian mechanism and add a noise drawn from a well-chosen distribution, $\mu_{\mathsf{DE},R}$ supported on $(-R,R)$ for any R, whose density at $\eta\in(-R,R)$ is:

$$\frac{\exp(-f_{\mathsf{DE},\mathsf{R}}(\eta))}{Z_{\mathsf{DE},\mathsf{R}}}\text{ with }f_{\mathsf{DE},\mathsf{R}}(\eta)=\exp\left(\frac{R^2}{R^2-\eta^2}\right)\text{ and }Z_{\mathsf{DE},\mathsf{R}}=\int_{-R}^{R}e^{-f_{\mathsf{DE},\mathsf{R}}(\eta)}d\eta \tag{3.98}$$

Dagan and Kur (2020) shows that when taking $\delta\geq\exp(-k/\log(k)^8)$ and $\varepsilon\in(0,1)$ there exists a universal constant $C>0$ such that when taking $R=\frac{C}{\varepsilon n}\sqrt{k\log\left(\frac{1}{\delta}\right)}$ adding noise from $\mu_{\mathsf{DE},\mathsf{R}}$ ensures (ε,δ)-DP to the average of n data of dimension k.

Similarly to the previous mechanisms we studied we can show the following proposition, which states the parameter we need to use to ensure (ε,δ)-DP.

Proposition 15. *For any $\varepsilon\in(0,1)$, $\delta_0\geq\exp(-SA/\log(SA)^8)$ and $\delta_1\geq\exp(-S^2A/\log(S^2A)^8)$ then the bounded noise mechanism, Alg. 16, is $(3H\varepsilon,\delta')$-LDP with $\delta_0'=\delta_0\frac{e^{H\varepsilon}-1}{e^{\varepsilon}-1}$, $\delta_1'=\delta_1\frac{e^{H\varepsilon}-1}{e^{\varepsilon}-1}$ and $\delta'=\delta_1'e^{2H\varepsilon}+2\delta_0'e^{2H\varepsilon}+2\delta_0'\delta_1'e^{H\varepsilon}+(\delta_0')^2e^{H\varepsilon}+(\delta_0')^2\delta_1'$.*

Proof. of Prop. 15

For any $\varepsilon\in(0,1)$ and $\delta_0\geq\exp(-SA/\log(SA)^8)$, for any $r\in\mathbb{R}^{S\times A}$ and two trajectories $X=\{(s_h,a_h,r_h)_{h\leq H}\}$ and $X'=\{(s_h',a_h',r_h')_{h\leq H}\}$ let's define $R_X(s,a)=\sum_{h=1}^{H}r_h\mathbb{1}_{\{s_h=s,a_h=a\}}$ the cumulative reward in state-action (s,a) associated to trajectory X. Finally, let's define for a set of indexes $I\subset[H]]$ the new trajectory X_I where for $h\in I$, $(X_I)_h=(s_h,a_h,r_h)$

Algorithm 16: Bounded Noise Mechanism for LDP

Input: Trajectory: $X = \{(s_h, a_h, r_h) \mid h \leq H\}$, Privacy Parameter: ε, δ, Constant: C

Set $R_1 = \frac{C}{\varepsilon}\sqrt{SA\ln(1/\delta)}$ and $R_2 = \frac{CS}{\varepsilon}\sqrt{A\ln(1/\delta)}$

for $(s,a) \in \mathcal{S} \times \mathcal{A}$ **do**

Sample $Y_{1,X}(s,a) \sim \mu_{\mathsf{DE},\mathsf{R}_1}$

$\widetilde{R}_X(s,a) = Y_{1,X}(s,a) + \sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s, a_h=a\}}$

Sample $\widetilde{n}^r_X(s,a) \sim \mu_{\mathsf{DE},\mathsf{R}_1}$

$\widetilde{N}^r_X(s,a) = \widetilde{n}^r_X(s,a) + \sum_{h=1}^{H} \mathbb{1}_{\{s_h=s, a_h=a\}}$

for $s' \in \mathcal{S}$ **do**

Sample $\widetilde{n}^p_X(s,a,s') \sim \mu_{\mathsf{DE},\mathsf{R}_2}$

$\widetilde{N}^p_X(s,a,s') = \widetilde{n}^r_X(s,a,s') + \sum_{h=1}^{H-1} \mathbb{1}_{\{s_h=s, a_h=a, s_{h+1}=s'\}}$

$\widetilde{N}^p_X(s,a,s') = \widetilde{n}^r_X(s,a,s') + \sum_{h=1}^{H-1} \mathbb{1}_{\{s_h=s, a_h=a, s_{h+1}=s'\}}$

Output: $(\widetilde{R}_X, \widetilde{N}^r_X, \widetilde{N}^p_X) \in \mathbb{R}^{S\times A} \times \mathbb{R}^{S\times A} \times \mathbb{R}^{S\times A\times S}$

and for $h \notin I$, $(X_I)_h = (s'_h, a'_h, r'_h)$. Therefore, using Theorem 3.2 from Dagan and Kur (2020), we have that for $I = [H-1]\!]$ and $\widetilde{R}_X$ defined as in Alg. 16,

$$\mathbb{P}\left(\widetilde{R}_X = r\right) \leq \exp(\varepsilon)\mathbb{P}\left(\widetilde{R}_{X_I} = r\right) + \delta_0 \tag{3.99}$$

$$\leq \exp(\varepsilon)\left(\exp(\varepsilon)\mathbb{P}\left(\widetilde{R}_{X_{[H-2]}} = r\right) + \delta_0\right) + \delta_0 \tag{3.100}$$

Therefore repeating the same argument H times, we have that:

$$\mathbb{P}\left(\widetilde{R}_X = r\right) \leq \exp(H\varepsilon)\mathbb{P}\left(\widetilde{R}_{X'} = r\right) + \delta_0 \sum_{h=0}^{H-1} \exp(h\varepsilon) \tag{3.101}$$

$$= \exp(H\varepsilon)\mathbb{P}\left(\widetilde{R}_{X'} = r\right) + \delta_0 \frac{\exp(H\varepsilon) - 1}{\exp(\varepsilon) - 1} \tag{3.102}$$

In addition, we have with the same reasoning that for any $n \in \mathbb{R}^{S\times A}$ and $n^p \in \mathbb{R}^{S\times A\times S}$ that:

$$\mathbb{P}\left(\widetilde{N}^r_X = n\right) \leq \exp(H\varepsilon)\mathbb{P}\left(\widetilde{N}_{X'} = n\right) + \delta_0 \frac{\exp(H\varepsilon) - 1}{\exp(\varepsilon) - 1} \tag{3.103}$$

and for any $\delta_1 \geq \exp(-S^2A/\log(S^2A)^8)$:

$$\mathbb{P}\left(\widetilde{N}^p_X = n^p\right) \leq \exp(H\varepsilon)\mathbb{P}\left(\widetilde{N}^p_{X'} = n^p\right) + \delta_1 \frac{\exp(H\varepsilon) - 1}{\exp(\varepsilon) - 1} \tag{3.104}$$

Therefore we have that:

$$\begin{aligned}
&\mathbb{P}\left(\widetilde{R}_X = r, \widetilde{N}^r_X = n, \widetilde{N}^p_X = n^p \mid X\right) = \mathbb{P}\left(\widetilde{R}_X = r \mid X\right)\mathbb{P}\left(\widetilde{N}^r_X = n \mid X\right)\mathbb{P}\left(\widetilde{N}^p_X = n^p \mid X\right) \\
&\leq \left(e^{H\varepsilon}\mathbb{P}\left(\widetilde{R}_{X'} = r\right) + \delta_0\frac{e^{H\varepsilon}-1}{e^{\varepsilon}-1}\right)\left(e^{H\varepsilon}\mathbb{P}\left(\widetilde{N}_{X'} = n\right) + \delta_0\frac{e^{H\varepsilon}-1}{e^{\varepsilon}-1}\right) \times \\
&\times \left(e^{H\varepsilon}\mathbb{P}\left(\widetilde{N}^p_{X'} = n^p\right) + \delta_1\frac{e^{H\varepsilon}-1}{e^{\varepsilon}-1}\right) \\
&\leq e^{3H\varepsilon}\mathbb{P}\left(\widetilde{R}_{X'} = r, \widetilde{N}^r_{X'} = n, \widetilde{N}^p_{X'} = n^p\right) + \delta_1' e^{2H\varepsilon}\mathbb{P}\left(\widetilde{R}_{X'} = r\right)\mathbb{P}\left(\widetilde{N}^r_{X'} = n\right) \\
&+ \delta_0' e^{2H\varepsilon}\mathbb{P}\left(\widetilde{N}^p_{X'} = n^p\right)\left(\mathbb{P}\left(\widetilde{N}^r_{X'} = n\right) + \mathbb{P}\left(\widetilde{R}_{X'} = r\right)\right) \\
&+ \delta_0'\delta_1' e^{H\varepsilon}\left(\mathbb{P}\left(\widetilde{N}^r_{X'} = n\right) + \mathbb{P}\left(\widetilde{R}_{X'} = r\right)\right) + (\delta_0')^2 e^{H\varepsilon}\mathbb{P}\left(\widetilde{N}^p_{X'} = n^p\right) + (\delta_0')^2\delta_1'
\end{aligned}$$

with $\delta_0' = \delta_0\frac{e^{H\varepsilon}-1}{e^{\varepsilon}-1}$ and $\delta_1' = \delta_1\frac{e^{H\varepsilon}-1}{e^{\varepsilon}-1}$. Therefore, we have that the mechanism is $(3H\varepsilon, \delta')$-LDP that is to say:

$$\begin{aligned}
\mathbb{P}\left(\widetilde{R}_X = r, \widetilde{N}^r_X = n, \widetilde{N}^p_X = n^p \mid X\right) \leq e^{3H\varepsilon}\mathbb{P}\left(\widetilde{R}_{X'} = r, \widetilde{N}^r_{X'} = n, \widetilde{N}^p_{X'} = n^p\right) + \delta_1' e^{2H\varepsilon} \\
+ 2\delta_0' e^{2H\varepsilon} + 2\delta_0'\delta_1' e^{H\varepsilon} + (\delta_0')^2 e^{H\varepsilon} + (\delta_0')^2\delta_1'
\end{aligned}$$

with $\delta_0' = \delta_0\frac{e^{H\varepsilon}-1}{e^{\varepsilon}-1}$, $\delta_1' = \delta_1\frac{e^{H\varepsilon}-1}{e^{\varepsilon}-1}$ and $\delta' = \delta_1' e^{2H\varepsilon} + 2\delta_0' e^{2H\varepsilon} + 2\delta_0'\delta_1' e^{H\varepsilon} + (\delta_0')^2 e^{H\varepsilon} + (\delta_0')^2\delta_1'$. □

In addition, because the noise is bounded we can apply standard sub-gaussian concentration inequalities to show that Alg. 16 satisfies Def. 3.

Proposition 16. *The bounded noise mechanism, Alg. 16, with parameter $\varepsilon_0 > 0$ satisfies Def. 11 for any $\delta > 0$ and $k \in \mathbb{N}^\star$ with:*

$$c_{k,1}(\varepsilon_0, \delta) = c_{k,2}(\varepsilon_0, \delta) = R\sqrt{2(k-1)\ln\left(\frac{6SA}{\delta}\right)}$$
$$c_{k,3}(\varepsilon_0, \delta) = R_2\sqrt{2S(k-1)\ln\left(\frac{6S^2A}{\delta}\right)}$$
$$c_{k,4}(\varepsilon_0, \delta) = R_2\sqrt{2(k-1)\ln\left(\frac{6S^2A}{\delta}\right)}$$

with $R = \frac{1}{\varepsilon}\sqrt{SA\ln(1/\delta_0)}$ and $R_2 = \frac{S}{\varepsilon}\sqrt{A\ln(1/\delta_0)}$

Proof. of Prop. 16 For any $\delta > 0$ and at the beginning of episode k, we have thanks to Hoeffding inequality that with probability at least $1 - \frac{\delta}{3SA}$ for any state-action $(s,a) \in \mathcal{S} \times \mathcal{A}$:

$$\left|\widetilde{R}_k(s,a) - R_k(s,a)\right| = \left|\sum_{l=1}^{k-1} Y_{1,X_l}(s,a)\right| \leq R\sqrt{2(k-1)\ln\left(\frac{6SA}{\delta}\right)} \tag{3.105}$$

with $(Y_{1,X_l}(s,a))_{l\leq k-1}$ are i.i.d distributed according to $\mu_{\mathsf{DE},\mathsf{R}_1}$. With the same reasonning, we have that with probability at least $1 - \frac{\delta}{3SA}$:

$$\left|\widetilde{N}_k^r(s,a) - N_k^r(s,a)\right| = \left|\sum_{l=1}^{k-1} \widetilde{n}_{X_l}^r(s,a)\right| \leq R\sqrt{2(k-1)\ln\left(\frac{6SA}{\delta}\right)} \tag{3.106}$$

Finally, still using Hoeffding inequality, and definning $R_2 = \frac{CS}{\varepsilon}\sqrt{A\ln(1/\delta)}$, we have that with probability at least $1 - \frac{\delta}{3S^2A}$:

$$\left|\widetilde{N}_k^p(s,a,s') - \sum_{l<k}\sum_{h=1}^{H-1} \mathbb{1}_{\{s_{l,h}=s, a_{l,h}=a, s_{l,h+1}=s'\}}\right| \leq R_2\sqrt{2(k-1)\ln\left(\frac{6S^2A}{\delta}\right)} \tag{3.107}$$

And finally with probability at least $1 - \frac{\delta}{3SA}$:

$$\left|\sum_{s'\in\mathcal{S}} \widetilde{N}_k^p(s,a,s') - \sum_{s'\in\mathcal{S}}\sum_{l<k}\sum_{h=1}^{H-1} \mathbb{1}_{\{s_{l,h}=s, a_{l,h}=a, s_{l,h+1}=s'\}}\right| \leq R_2\sqrt{2S(k-1)\ln\left(\frac{6S^2A}{\delta}\right)} \tag{3.108}$$

□

3.A.7 Experimental Results:

We show empirical results for three mechanisms discussed in the RandomMDP environment in Figures 3.A.2, 3.A.3 and 3.A.4.

As we have seen in Fig. 3.1.1, the LDP constraint has a significant impact on the regret especially as ε decreases. In particular for $\varepsilon = 0.2$, LDP-OBI-L, LDP-OBI-G, LDP-OBI-B, LDP-OBI-BND have not reached the usual square root growth phase of the regret usually seen in UCB-VI or other regret minimizing algorithm.

From figures 3.A.2, 3.A.3 and 3.A.4, we can observe that the bounded noise mechanism has a lower impact on the regret compared to the Laplace, Gaussian and Randomized Response mechanisms. However, this benefit does not appear in the regret bound of Table 3.1.1. This suggests that the regret analysis of Sec. 4.2.4 may be improved to show this empirically observed advantage.

3.A.8 Posterior Sampling for Local Differential Privacy

The Posterior Sampling for Reinforcement Learning algorithm (PSRL, Osband et al., 2013) is a Thompson Sampling based algorithm for Reinforcement Learning. It works by maintaining a Bayesian posterior distribution over MDPs. We focus on a particular instantiation of PSRL where for each state-action pair (s,a) we have an independent Gaussian prior for the reward distribution and a Dirichlet prior for the transition dynamics. With those priors, the posterior distributions are Normal-Gamma and Dirichlet distributions.

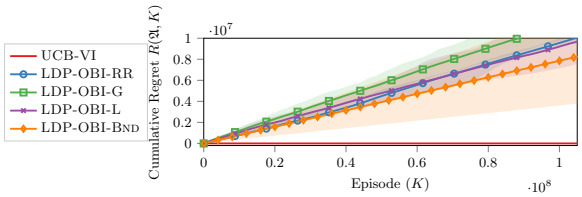


Figure 3.A.2: $\varepsilon = 0.2$ and $\delta = 0.1$ (*only* for the Gaussian and bounded noise mechanism)

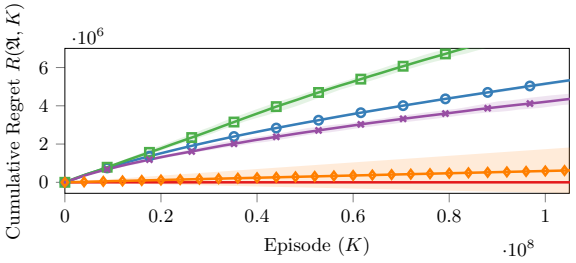


Figure 3.A.3: $\varepsilon = 2$ and $\delta = 0.1$ (*only* for the Gaussian and bounded noise mechanism)

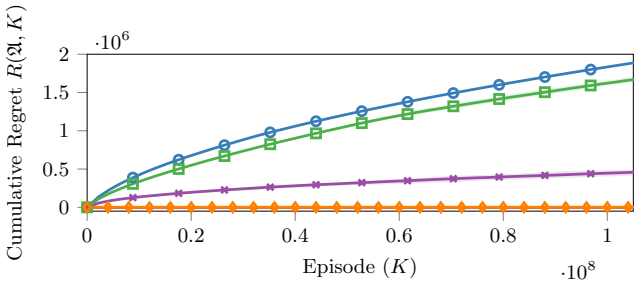


Figure 3.A.4: $\varepsilon = 20$ and $\delta = 0.1$ (*only* for the Gaussian and bounded noise mechanism)

Let $\alpha_0(s,a)$ denote the parameters of the prior distribution over the transition dynamics, so the prior is given by $\text{Dir}(\alpha_0(s,a))$. In addition, let $\mu_0(s,a) \in \mathbb{R}$, $\lambda_0(s,a) \in \mathbb{R}_+^\star$, $\nu_0(s,a) \in \mathbb{R}_+^\star$ and $\beta_0(s,a) \in \mathbb{R}_+^\star$ be the parameters of the Normal-Gamma prior distribution that we place on the rewards. Then, at the beginning of episode k and for a given pair $(s,a) \in \mathcal{S} \times \mathcal{A}$, let $\alpha_k(s,a) \in (\mathbb{R}_+^\star)^S$ be such that the posterior distribution over the transition dynamics is $\text{Dir}(\alpha_k(s,a))$. We then define $\mu_k(s,a) \in \mathbb{R}$, $\lambda_k(s,a) \in \mathbb{R}_+^\star$, $\nu_k(s,a) \in \mathbb{R}_+^\star$ and $\beta_k(s,a) \in \mathbb{R}_+^\star$ to the parameters of the Normal-Gamma posterior distributions. Using standard results from Bayesian Learning we have that, for all state $s' \in \mathcal{S}$:

$$\alpha_k(s,a) = \alpha_0(s,a) + N_k(s,a,s') \tag{3.109}$$

$$\lambda_k(s,a) = \lambda_0(s,a) + N_k(s,a) \tag{3.110}$$

$$\nu_k(s,a) = \nu_0(s,a) + \frac{N_k(s,a)}{2} \tag{3.111}$$

$$\mu_k(s,a) = \frac{\lambda_0(s,a)\mu_0(s,a) + N_k(s,a)\hat{R}_k(s,a)}{\lambda_0(s,a) + N_k(s,a)} \tag{3.112}$$

$$\beta_k(s,a) = \beta_0(s,a) + \frac{1}{2}\widehat{\text{Var}}(R(s,a)) + \frac{N_k(s,a)\lambda_0(s,a)}{2(\lambda_0(s,a) + N_k(s,a))}\left(\hat{R}_k(s,a) - \mu_0(s,a)\right)^2 \tag{3.113}$$

where $\alpha_0, \mu_0, \lambda_0, \nu_0, \beta_0$ are prior parameters provided at the beginning of the algorithm. We denote by $N_k(s,a)$, the number of visits to the state-action pair (s,a), $N_k(s,a,s')$ the number visits to (s,a,s'), $\hat{R}_k(s,a)$ the average reward observed for (s,a) and $\widehat{\text{Var}}(R(s,a))$ the empirical variance for (s,a).

At each episode k, PSRL samples an MDP from the posterior distributions, then computes the optimal policy and executes it in the true MDP. (Osband et al., 2013) showed that the *Bayesian* regret of this algorithm is bounded by $\tilde{O}\left(HS\sqrt{AT}\right)$.

Locally Differentially Private Posterior Sampling for Reinforcement Learning: We now discuss how to adapt PSRL to ensure it is locally differentially private. Def. 3 states that LDP is ensured at the collection time of trajectories therefore it is enough for us to design a LDP posterior sampling algorithm which takes as input the trajectories outputted by a mechanism similar to Alg. 13. Here, we use the LDP mechanism to pertub the statistics used to define the parameters of the posterior distribution in PSRL. More precisely, we replace the aggregate counts in Eqs. 3.109-3.113 by noisy counts provided by an LDP mechanism. In order to do this, we need to modify the initial values of those parameters to guarantee they are non-negative.

In this appendix, we assume that the privacy-preserving mechanism $\mathcal{M}$ is such that for a given trajectory X, $\mathcal{M}(X) =$

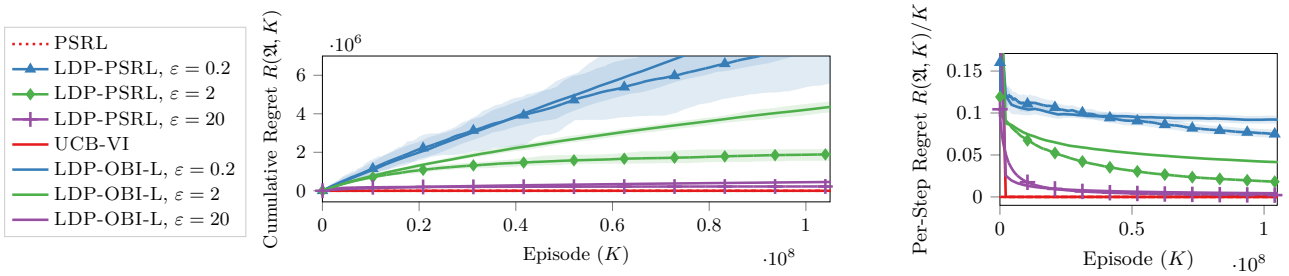


Figure 3.A.5: Evaluation of LDP-PSRL in the RandomMDP environment. *Left)* Cumulative regret. *Right)* per-step regret ($k \mapsto R_k/k$). Results are averaged over 20 runs and the the confidence intervals are the minimum and maximum runs. While the regret looks almost linear for $\varepsilon = 0.2$, the decreasing trend of the per-step regret shows that the algorithms are learning.

$(\widetilde{R}_X, \widetilde{R}_{2,X}, \widetilde{N}^r_X, \widetilde{N}^p_X)$ where $\widetilde{R}_X, \widetilde{R}_{2,X}, \widetilde{N}^r_X$ and $\widetilde{N}^p_X$ are noisy version of the following aggregate statistics:

$$R_X(s,a) = \sum_{h=1}^{H} r_h \mathbb{1}_{\{s_h=s, a_h=a\}}, \qquad R_{2,X}(s,a) = \sum_{h=1}^{H} r_h^2 \mathbb{1}_{\{s_h=s, a_h=a\}}$$

$$N^r_X(s,a) = \sum_{h=1}^{H} \mathbb{1}_{\{s_h=s, a_h=a\}}, \qquad N^p_X(s,a,s') = \sum_{h=1}^{H-1} \mathbb{1}_{\{s_h=s, a_h=a, s_{h+1}=s'\}}$$

In particular, $\widetilde{R}_X, \widetilde{N}^r_X$ and $\widetilde{N}^p_X$ are defined as for the optimistic algorithm in subsection 3.1.3.1 and $\widetilde{R}_{2,X}$ is a privatized version of $R_{2,X}(s,a) = \sum_{h=1}^{H} r_h^2 \mathbb{1}_{\{s_h=s, a_h=a\}}$ for a trajectory X.s

The posterior updates we use in LDP-PSRL are then for all $s' \in \mathcal{S}$:

$$\begin{aligned}
\widetilde{\alpha}_k(s,a) &= \alpha_0(s,a) + \widetilde{N}^p_k(s,a,s') \\
\widetilde{\mu}_k(s,a) &= \frac{\lambda_0(s,a)\mu_0(s,a) + \widetilde{R}_k(s,a)}{\lambda_0(s,a) + \widetilde{N}^r_k(s,a)} \\
\widetilde{\lambda}_k(s,a) &= \lambda_0(s,a) + \widetilde{N}^r_k(s,a) \\
\widetilde{\nu}_k(s,a) &= \tilde{\nu}_0(s,a) + \frac{\widetilde{N}^r_k(s,a)}{2} \\
\widetilde{\beta}_k(s,a) &= \beta_0(s,a) + \frac{\lambda_0(s,a)\widetilde{N}^r_k(s,a)\mu_0^2(s,a) - \widetilde{R}^2_k(s,a)}{2(\lambda_0(s,a) + \widetilde{N}^r_k(s,a))} \\
&\qquad + \frac{1}{2}\sum_{l \le k-1} \widetilde{R}_{2,l} - \frac{\mu_0(s,a)\widetilde{R}_k(s,a)}{\lambda_0(s,a) + \widetilde{N}^r_k(s,a)}
\end{aligned} \tag{3.114}$$

In the following, we choose the Laplace mechanism as our privacy-preserving mechanism for LDP-PSRL, although we believe that it should be possible to use one of the other mechanisms we discussed. For each trajectory X, we add independent Laplace variables to $(R_X(s,a), R_{X,2}(s,a), N^r_X(s,a), N^p_X(s,a))$ with parameter $8H/\varepsilon$. Following the same argument outlined in the proof of Thm. 10, we can show that this privacy-preserving mechanism is $(\varepsilon, 0)$-LDP.

To ensure positivity, by concentration of Laplace variables we set the initial values of the parameters of the posterior distributions to:

$$\alpha_0(s,a,s') = \max\{\sqrt{KS}, \ln(6S^2A/\delta)\}\frac{\sqrt{8\ln(6S^2A/\delta)}}{\varepsilon_0} \tag{3.115}$$

$$\mu_0(s,a) = 0 \tag{3.116}$$

$$\lambda_0(s,a) = \max\{\sqrt{K}, \ln(6SA/\delta)\}\frac{\sqrt{8\ln(6SA/\delta)}}{\varepsilon_0} \tag{3.117}$$

$$\nu_0(s,a) = \max\{\sqrt{K}, \ln(6SA/\delta)\}\frac{\sqrt{8\ln(6SA/\delta)}}{\varepsilon_0} \tag{3.118}$$

$$\beta_0(s,a) = 5\max\{\sqrt{K}, \ln(6SA/\delta)\}\frac{\sqrt{8\ln(6SA/\delta)}}{\varepsilon_0} \tag{3.119}$$

where K is the total number of episodes. The pseudocode of LDP-PSRL is reported in Alg. 17.

Algorithm 17: LDP-PSRL

Input: Initial values: $\alpha_0, \mu_0, \lambda_0, \nu_0$ and β_0

for *episodes* $k = 1, \dots, K$ **do**

Draw empirical MDP, θ_k from the posterior and compute π_k as the optimal policy for MDP θ_k

User u_k executes policy π_k, collect trajectory $X_k = \{(s_{k,h}, a_{k,h}, r_{k,h}) \mid h \leq H\}$

Update noisy counts with $(\widetilde{R}_{X_k}(s,a), \widetilde{R}_{X_k,2}(s,a), \widetilde{N}^r_{X_k}(s,a), \widetilde{N}^p_{X_k}(s,a))$ and posterior distribution

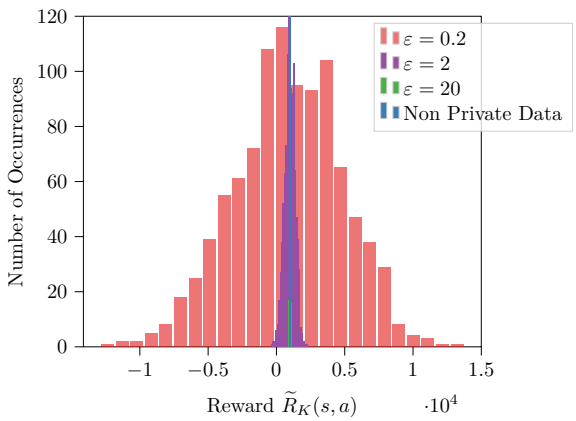


Figure 3.A.6: Aggregate reward for privatized data with $\varepsilon \in \{0.2, 2, 20\}$ and non-privatized data for state 0 and action 1

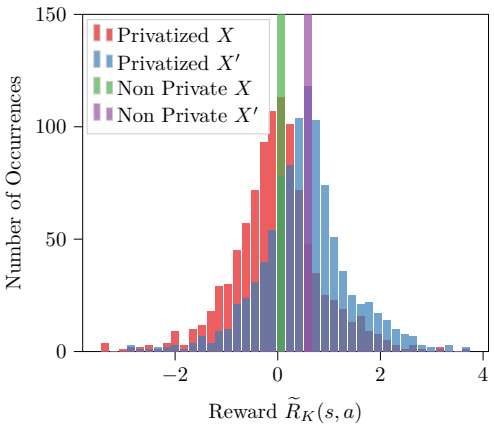


Figure 3.A.7: Privatized cumulative reward over an episode for a given state-action pair and two different trajectories X and X' with $\varepsilon = 20$ for state 0 and action 1

Empirical results We show empirical results for the LDP-PSRL algorithm in the RandomMDP environment in Figure 3.A.5. While we have shown that this algorithm is ε-LDP and empirically outperforms optimistic approaches, we leave the regret analysis to future work.

3.A.9 Additional Experiment

In this subsection, we explore a second experiment, in which we use the same the RandomMDP environment with the same parameters as in Sec. 4.1.5 in order to investigate the effect of differential privacy on the learning process. For this, we run the UCB-VI algorithm for $K = 10^3$ episodes and collect the aggregate noisy statistics, $(\widetilde{R}_K(s,a))_{(s,a)\in\mathcal{S}\times\mathcal{A}}$, $(\widetilde{N}^r_K(s,a))_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ and $(\widetilde{N}^p_K(s,a,s'))_{(s,a,s')\in\mathcal{S}\times\mathcal{A}\times\mathcal{S}}$ that have been generated by using the Laplace mechanism for each episode. We collect those statistics, 10^3 times. We compare the histogram of those noisy statistics to that of the noiseless statistics used by UCB-VI in Fig. 3.A.6. This demonstrates that, as expected, there is much more variation in the statistics provided by the private mechanism. In Fig. 3.A.7, we applied the Laplace mechanism to two different random trajectories, X and X'. We can see that, after applying the Laplace mechanism, the two distinct trajectories become almost indistinguishable. These two figures combined demonstrate the difficulty of learning from locally differentially private data.

3.A.10 Privacy Amplification by Shuffling in RL

In recent years, the shuffle model for privacy (Cheu et al., 2019; Feldman et al., 2020; Chen et al., 2021; Balle et al., 2019b; Erlingsson et al., 2020, 2019) has attracted a lot of attention thanks its amplification property tof the differential privacy guarantees of locally differential data.

In this model of privacy, we consider n users equipped with a local differential privacy mechanism, each user submits a locally private report to a random shuffler which computes a random permutation of the users' reports. Those randomly shuffled reports are then sent to a analyzer which computes functions of interests based on them. This setting was first introduced in Bittau et al. (2017) and was named the *ESA* model (Encode-Shuffle-Analyze) and motivated by need for anonymous data collection. (Erlingsson et al., 2019) later provided an analysis of the amplification of privacy thanks to the combined use of shuffling and local differential privacy showing that the shuffling model of privacy is able to strike a middle ground between the totally decentralized but somewhat sample inefficient *local* model and the centralized but more sample efficient central model of privacy.

Algorithm 18: Shuffling Protocol

Input: number of episodes K, horizon H, failure probability $\delta \in (0,1)$, bias $\alpha > 1$, private randomizer $\mathcal{M}_{\text{sh}}$ with LDP parameters (ϵ_0, δ_0)

for $k = 1$ ***to*** K **do**

Shuffler $\mathcal{R}$ sends $(\mathcal{M}_{\text{sh}}(X_{u_{\sigma_k(l)}}))_{l \leq k-1}$ with σ_k a random permutatioon at each episode

LDP-OBI computes policy π_k based on $(\mathcal{M}_{\text{sh}}(X_{u_{\sigma_k(l)}}))_{l \leq k-1}$

User u_k executes policy π_k in the environment, collects trajectory $X_k = \{(s_{k,h}, a_{k,h}, r_{k,h})_{h \leq H}\}$ and sends the privatized trajectory $\mathcal{M}_{\text{sh}}(X_k)$ to $\mathcal{R}$

Algorithm 19: Local randomizer $R_p^{0/1}$

Input: Randomization probability: $p \in [0,1]$, $x \in \{0,1\}$

Let $b \sim \text{Ber}(p)$

if $b = 0$ **then**

Return x

else

Return $\text{Ber}(1/2)$

The shuffling model has then been refined to study the impact on the size of the reports sent by users, i.e., how the accuracy of a shuffling protocol can be improved when user are allowed to have higher communication threshold (Cheu et al., 2019; Balle et al., 2019a). It has also been studied for different analyzer function, for instance histograms (Balcer and Cheu, 2020) or summation (Cheu et al., 2019; Balle et al., 2019b), obtaining optimal protocol with better accuracy and lesser communication costs (i.e., the number of messages or the size of those messages sent by a user). Finally, the shuffle model has inspired a privacy amplification algorithm for learning in distributed setting without server-initiated communication (Balle et al., 2019b).

Overall, the most attractive feature of this privacy model is that it offers a smooth transition in terms of privacy/utility tradeoff between stringent LDP requirements and differential privacy requirements (see (Feldman et al., 2020) for an example of this transition in the problem of estimating a distribution).

Formally, in our RL setting each episode k represents a user u_k which completes a trajectory X_{u_k} in the MDP. The user computes a locally private version of its trajectory thanks to a privacy-preserving mechanism $\mathcal{M}$. The result $\mathcal{M}(X_{u_k})$ is passed to a shuffler $\mathcal{R}$. This shuffler stores all the previous privatized trajectories before the current episode k, $(\mathcal{M}(X_{u_l}))_{l<k}$, computes a random permutation $\sigma : [k-1] \to [k-1]$ and sends the permuted set of privatized trajectories, $(\mathcal{M}(X_{u_{\sigma(l)}}))_{l \leq k-1}$ to an RL algorithm like LDP-OBI. This interaction protocol is detailed in Alg. 18.

In the specific case of RL, thanks to (Vietri et al., 2020) we know that any regret minimizing algorithm using (ε, δ)-DP counters, like $(N_k^p)_{k \leq K}$ is (ε, δ)-joint differentially private.

3.A.10.1 Privacy-preserving mechanism $\mathcal{M}_{\text{sh}}$

A trajectory $X_u := \{(s_h, a_h, r_h) \mid h \leq H\}$ is a sequence of H states, actions and rewards. In order to build a model of the MDP, LDP-OBI uses counters of the numbers of occurrences of each tuple of state-action (s,a) and state, actions and next-state (s,a,s'). We adapt to the RL setting, the algorithm for bit-sum protocol presented in (Cheu et al., 2019). The first step of the process $\mathcal{M}_{\text{sh}}$ is to apply a one-hot encoding the trajectory for each state-action. Let $x \in \{0,1\}^{H \times S \times A}$ and $y \in \{0,1\}^{(H-1) \times S \times A \times S}$ such that for each $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$

$$\forall h \in [\![1,H]\!], \qquad x_{h,s,a} = \mathbb{1}_{\{s_h = s, a_h = a\}}, \text{ and } y_{h,s,a,s'} = \mathbb{1}_{\{s_h = s, a_h = a, s_{h+1} = s'\}} \tag{3.120}$$

To encode the reward, we first compute the reward for each state-action pair, $\left(r_h \mathbb{1}_{\{s_h = s, a_h = a\}}\right)_{(h,s,a) \in [\![1,H]\!] \times \mathcal{S} \times \mathcal{A}}$ then given a parameter $m \in \mathbb{N}^\star$ for each state-action pair (s,a), we compute $b_{h,s,a} \in \{0,1\}^m$ such that for $j \in [\![1,m]\!]$:

$$(b_{h,s,a})_j = \begin{cases} 1 & \text{if } j < \mu_{h,s,a} \\ \text{Ber}\left(p_{h,s,a}\right) & \text{if } j = \mu_{h,s,a} \\ 0 & \text{if } j > \mu_{h,s,a} \end{cases} \tag{3.121}$$

with $\mu_{h,s,a} = \left\lceil m r_h \mathbb{1}_{\{s_h = s, a_h = a\}} \right\rceil$ and $p_{h,s,a} = m r_h \mathbb{1}_{\{s_h = s, a_h = a\}} - \mu_{h,s,a} + 1$.

It is a well known result, (Cheu et al., 2019) that Alg. 22 with parameter p guarantees $\ln(2/p - 1)$ differential privacy. Finally, the privacy-preserving mechanism $\mathcal{M}_{\text{sh}}$ is described by Alg. 20.

Using standard analysis, we can show that this local mechanism $R_p^{0/1}$ is roughly $H\varepsilon$-LDP for any $\varepsilon > 0$.Upon receiving the shuffled privatized, the algorithm LDP-OBI computes the different counts $(\tilde{N}_k^p(s,a,s'))_{(s,a,s')}$, $(\tilde{N}_k^r(s,a))_{(s,a)}$ and

Algorithm 20: Privacy-preserving mechanism $\mathcal{M}_{\text{sh}}$

Input: trajectory $\tau = \{(s_h, a_h, r_h)_{h\leq H}\}$, privacy parameter $\varepsilon > 0$, parameter $m \in \mathbb{N}^\star$
Compute x and y as in Eq. (3.120) and $(b_{h,s,a})_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ as in Eq. (3.121)
Set $p = \frac{2}{\exp(\varepsilon)+1}$
Output: $(R_p^{0/1}(x_{h,s,a}))_{(h,s,a)}$, $(R_p^{0/1}(y_{h,s,a,s'}))_{(h,s,a,s')}$ and $((R_p^{0/1}((b_{h,s,a})_j)_{j\leq m})_{(h,s,a)}$

$(\tilde{R}_k(s,a))_{(s,a)}$. For any $(s,a,s') \in \mathcal{S}\times\mathcal{A}\times\mathcal{S}$, we define the counters as:

$$\tilde{N}_k^r(s,a) = \frac{1}{1-p}\left(\sum_{l=1}^{k-1}\sum_{h=1}^{H}\left[R_p^{0/1}(x_{h,s,a}) - \frac{p}{2}\right]\right) \tag{3.122}$$

$$\tilde{N}_k^p(s,a,s') = \frac{1}{1-p}\left(\sum_{l=1}^{k-1}\sum_{h=1}^{H}\left[R_p^{0/1}(y_{h,s,a,s'}) - \frac{p}{2}\right]\right) \tag{3.123}$$

$$\tilde{R}_k^r(s,a) = \frac{1}{m(1-p)}\left(\sum_{j=1}^{m}\sum_{l=1}^{k-1}\sum_{h=1}^{H}\left[R_p^{0/1}((b_{h,s,a})_j) - \frac{p}{2}\right]\right) \tag{3.124}$$

Therefore, thanks to Claim 4.6 of (Cheu et al., 2019), we have at the beginning of episode k,$(\tilde{N}_k^r(s,a))_{(s,a)}$ and $(\tilde{N}_k^p(s,a,s'))_{(s,a,s')}$ are $(\varepsilon_{k,c},\delta_0)$-DP with any $\delta_0 > 0$ and:

$$\varepsilon_{k,c} = \frac{32\log(4/\delta_0)/\sqrt{(k-1)H}}{\sqrt{p - \sqrt{\frac{2p\log(2/\delta_0)}{(k-1)H}}}}\left(1 - \left(p - \sqrt{\frac{2p\log(2/\delta_0)}{(k-1)H}}\right)\right) \tag{3.125}$$

with $p \in \left[\frac{14}{(k-1)H}\log(4/\delta_0), 1\right]$. But we have that with probability at least $1-\delta$, for any $\delta > 0$, that:

$$\left|\sum_{l=1}^{k-1}\sum_{h=1}^{H}\mathbb{1}_{\{s_{l,h}=s,a_{l,h}=a\}} - \tilde{N}_k^r(s,a)\right| \leq \frac{1}{1-p}\left(\sqrt{(k-1)Hp(1-p/2)\ln(1/\delta)} + \frac{2\ln(1/\delta)}{3}\right)$$

$$\left|\sum_{l=1}^{k-1}\sum_{h=1}^{H-1}\mathbb{1}_{\left\{\substack{s_{k,h}=s,\\ a_{k,h}=a\\ ,s_{k,h+1}=s'}\right\}} - \tilde{N}_k^p(s,a,s')\right| \leq \frac{1}{1-p}\left(\sqrt{(k-1)Hp(1-p/2)\ln(1/\delta)} + \frac{2\ln(1/\delta)}{3}\right)$$

The same type of result of result holds for the cumulative reward in each state-action pair (s,a), albeit some small technical difficulties due the estimated sum being in $\mathbb{R}$ and not an integer contrary to the counters for the number of visits.

3.A.10.2 Impact on the Regret

We have mentioned that thanks to the shuffling mechanism the counters $(\tilde{R}_k(s,a))_{(s,a)}$, $(\tilde{N}_k^r(s,a))_{(s,a)}$, $(\tilde{N}_k^p(s,a,s'))_{(s,a,s')}$ enjoy a (ε_c,δ)-DP guarantee, in addition to the ϵ_0-LDP guarantee. But the utility bound in the last subsubsection highlights that for a strict constraint on the level of local differential privacy the utility of each counters is of order $\frac{\sqrt{kH}}{\exp(\varepsilon_0)-1}$ therefore using Thm. 7, the regret of LDP-OBI coupled with $\mathcal{M}_{\text{sh}}$ is bounded with high probability by $\frac{H^2S^2A\sqrt{KH}}{\exp(\varepsilon_0/H)-1}$. This result is similar to the result of (Feldman et al., 2020) of Sec. 5.1 about density estimation where the shuffle model recovers the known rate of convergence of $\mathcal{O}(1/\varepsilon\sqrt{n})$ under an ε-LDP constraint with n samples.

However, in the reinforcement learning setting the shuffle model might allow to interpolate between LDP setting presented in this paper and the joint differential privacy setting of Shariff and Sheffet (2018); Vietri et al. (2020). One difficulty here being that because each user interacts only once with the RL algorithm the probability used by the local randomizer $R_p^{0/1}$ ha to be dependent on the number of previous episode to ensure a good (ε,δ)-JDP guarantee. In other words, for the very first episodes the privacy amplification of the shuffle model is negligible therefore the privacy parameter for those early users has to be stronger than for the latter ones which are somewhat hidden by the crowd. Albeit this minor issue, a good choice of the probabilities $(p_i)_{k\leq K}$ may be able to guarantee (ε,δ)-JDP (for any $\varepsilon > 0$ and $\delta > 0$) and a regret of order $\mathcal{O}(\sqrt{K} + \frac{\log(K)}{\varepsilon})$.

3.B Appendix for Improving Privacy by Shuffling

3.B.1 Local Privatizer $\mathcal{M}_{\text{LDP}}$

In this appendix, we present the privacy-preserving mechanism $\mathcal{M}_{\text{LDP}}$ used in this paper.

Algorithm 21: Local Privatizer $\mathcal{M}_{\mathsf{LDP}}$

Input: context: $x \in \mathbb{R}^d$, reward: $r \in [0,1]$, context bound: L, privacy parameter: ε_0, encoding parameter: m

//*Encoder*//

Set $\tilde{y} = \frac{rx}{2L} + \frac{1}{2}$ and $\tilde{z} = \frac{xx^\intercal}{2L^2} + \frac{\mathbb{1}\mathbb{1}^\intercal}{2}$

for $j = 1, \ldots, d$ **do**

  Compute $\mu_j = \lceil \tilde{y}_j \cdot m \rceil$ and $p_j = m \cdot \tilde{y}_j - \mu_j + 1$

  for $k = 1, \ldots, m$ **do**

    Let $b_{j,k} = \begin{cases} 1 & \text{if } k < \mu_j \\ \mathsf{Ber}(p_j) & \text{if } k = \mu_j \\ 0 & \text{if } k > \mu_j \end{cases}$

for $i = 1, \ldots, d$ **do**

  for $j = 1, \ldots, i$ **do**

    Compute $\kappa_{i,j} = \lceil \tilde{z}_{i,j} \cdot m \rceil$ and $q_{i,j} = m \cdot \tilde{z}_{i,j} - \kappa_{i,j} + 1$

    for $k = 1, \ldots, m$ **do**

      Let $w_{i,j,k} = \begin{cases} 1 & \text{if } k < \kappa_{i,j} \\ \mathsf{Ber}(q_{i,j}) & \text{if } k = \kappa_{i,j} \\ 0 & \text{if } k > \kappa_{i,j} \end{cases}$

      Let $w_{j,i,k} = w_{i,j,k}$

//*Local Randomizer*//

Set probabilities $p = \frac{2}{\exp(2\varepsilon_0/md(d+3))+1}$ and compute private values $\tilde{b}_j = \left(R_p^{0/1}(b_{j,1}), \ldots, R_p^{0/1}(b_{j,m})\right)$ for all $j \in [\![1,d]\!]$, $\tilde{w}_{i,j} = \left(R_p^{0/1}(w_{i,j,1}), \ldots, R_p^{0/1}(w_{i,j,m})\right)$ for all $i \in [\![1,d]\!]$, $j \leq i$

Algorithm 22: Local Randomizer $R_p^{0/1}$

Input: probability: p, $x \in \{0,1\}$

Let $\mathbf{b} \sim \mathsf{Ber}(p)$;

if $\mathbf{b} = 0$ **then**

  Return x

else

  Return $\mathsf{Ber}(1/2)$

3.B.2 Proofs

In this appendix, we provide the full derivation of the results stated in the main text. We start introducing the notion of central (ε, δ)-DP that is widely used in the proofs.

Definition 4. *A randomized mechanism, $\mathcal{M} : \mathbb{R}^d \to \mathcal{Z}$, is said to be central (ε, δ) differential private (DP) if for all sequence of values $z \in \mathcal{R}^d$ and z' such that there exists a unique $i \leq t$ for which $z_i \neq z'_i$ and for all $j \neq i$, $z_j = z'_j$ then*

$$\mathbb{P}\left(\mathcal{M}(z) \in A \mid z\right) \leq e^{\varepsilon}\mathbb{P}\left(\mathcal{M}(z) \in A \mid z'\right) + \delta$$

for any $A \subset$ Range$(\mathcal{M})$.

Note that the concept of central DP is at the core for proving JDP results, in fact thanks to Claim 7 in (Shariff and Sheffet, 2018) having a sequence $(\widetilde{V}_t, B_t)_t$ is (ε, δ)-DP implies that a bandit algorithm based on this sequence is (ε, δ)-DP.

3.B.2.1 Proof of Lem. 21

Here, we detail how to obtain the confidence intervals around $\theta^\star$ using the privatized estimator $\widetilde{\theta}_j$ for any batch $j^S \leq M_S$ (with $M_S = Tl^\star$ the total number of batches from the shuffler side). First, let's define the sequence of random variables $(Y_{t,k,l,q})_{t\leq T,k,l\leq d,q\leq m}$, $(Z_{t,k,l,q})_{t\leq T,k,l\leq d,q\leq m}$ two independent sequences of i.i.d. Bernoulli distributed random variable with parameters $p = 2/(\exp(2\varepsilon_0/md(d+3)) + 1)$ and $1/2$ and such that for all $k, l \leq d$, $Y_{t,k,l,q} = Y_{t,l,k,q}$ and $Z_{t,k,l,q} = Z_{t,l,k,q}$. For every $(t, k, l, q) \in [T] \times [d] \times [d] \times [m]$, $Y_{t,k,l,q}$ is sampled by Alg. 22 if $Y_{t,k,l,q} = 1$ then it return the random variable $Z_{t,k,l,q}$ otherwise it returns the true data.

In addition, let's define $(A_{t,k,l} = w_{t,k,l,\kappa_{t,k,l}})_{t\leq T,k,l\leq d}$ a sequence of Bernoulli random variable with parameter $(q_{t,k,l})_{t\leq T,k,l\leq d}$ defined by the two sequences $(\widetilde{z}_{t,k,l})_{t\leq T,k,l\leq d}$ and $(\kappa_{t,k,l})_{t\leq T,k,l\leq d}$ in the mechanism $\mathcal{M}_{\text{LDP}}$, Alg. 21. Finally, let's note the sequence of data computing by the encoding part of Alg. 21.

For any batch $j^A \leq M_S$, we can write the approximate design matrix and vector B_j as follows for every coordinate $k, l \leq d$:

$$\begin{aligned}\widetilde{V}_{j,k,l} =& \frac{1}{m(1-p)}\sum_{t=1}^{t_j}\sum_{q=1}^{m} Y_{t,k,l,q}Z_{t,k,l,q} - \frac{p}{2} + \sum_{t=1}^{t_j}\frac{x_{t,a_t}x_{t,a_t}^\intercal}{2L^2} + +2\lambda_j \mathbb{1}_{\{k=l\}} \\ &+ \frac{1}{m}\sum_{t=1}^{t_j} A_{t,k,l} - (m\widetilde{z}_{t,k,l} - \kappa_{t,k,l} + 1) + \frac{1}{m(1-p)}\sum_{t=1}^{t_j}\sum_{q=1}^{m}(p - Y_{t,k,l,q})w_{t,k,l,q}\end{aligned} \tag{3.126}$$

where λ_j is defined in Eq. (3.14).

$$\widetilde{B}_{j,k} = \frac{1}{m(1-p)}\sum_{l=1}^{t_j}\sum_{q=1}^{m}\left(\tilde{b}_{l,i,q} - \frac{p}{2}\right) - \frac{t_j}{2} \tag{3.127}$$

Now, given an well-chosen regularization λ_j the approximate design matrix $\widetilde{V}_j$ can be written as the sum of the true design matrix $\sum_t x_{t,a_t}x_{t,a_t}^\intercal$ and a time-varying regularizer similar to (Shariff and Sheffet, 2018). We just need to bound with high probability the deviation of the eigenvalues of $\widetilde{V}_j - \sum_{t=1}^{t_j}\frac{x_{t,a_t}x_{t,a_t}^\intercal}{2L^2} - \lambda_j I_d$.

Let's consider a vector $v \in \mathbb{R}^d$ such that $\|v\|_2 = 1$ then for any time $t_j \leq T$ and $\delta \in (0,1)$ we have with probability at least $1-\delta$:

$$\left|\left\langle v, \left(\sum_{t=1}^{t_j}\sum_{q=1}^{m} Y_{t,.,.,q}Z_{t,.,.,q} - \frac{p\mathbb{1}\mathbb{1}^\intercal}{2}\right)v\right\rangle\right| \leq 2\sqrt{2t_j m\ln(2/\delta)} \tag{3.128}$$

Therefore because the matrix $\left(\sum_{t=1}^{t_j}\sum_{q=1}^{m} Y_{t,.,.,q}Z_{t,.,.,q} - \frac{p\mathbb{1}\mathbb{1}^\intercal}{2}\right)$ is symmetric we have that with high probability:

$$\max\left\{\left|\lambda_{\min}\left(\sum_{t,q} Y_{t,.,.,q}Z_{t,.,.,q} - \frac{p\mathbb{1}\mathbb{1}^\intercal}{2}\right)\right|, \lambda_{\max}\left(\sum_{t,q} Y_{t,.,.,q}Z_{t,.,.,q} - \frac{p\mathbb{1}\mathbb{1}^\intercal}{2}\right)\right\} \leq 2\sqrt{2t_j m\ln(2/\delta)}$$

where $\lambda_{\min}$ and $\lambda_{\max}$ are the minimum and maximum eigenvalues. Similarly, using the martingale difference structure,we have that for any $v \in \mathbb{R}^d$, $\|v\|_2 \leq 1$ and $\delta \in (0,1)$, we have with probability at least $1-\delta$:

$$\left|\left\langle v, \left(\sum_{t=1}^{t_{j+1}}\sum_{q=1}^{m}(p\mathbb{1}\mathbb{1}^\intercal - Y_{t,.,.,q})w_{t,.,.,q}\right)v\right\rangle\right| \leq 2\sqrt{2t_{j+1}m\ln(2/\delta)} \tag{3.129}$$

and

$$\left| \left\langle v, \left(\sum_{t=1}^{t_{j+1}} A_t - (m\widetilde{z}_t - \kappa_t + 1) \right) v \right\rangle \right| \leq 2\sqrt{2t_{j+1}\ln(2/\delta)} \tag{3.130}$$

Indeed, for every $t \leq T$, let's define the filtration $\mathcal{F}_t$ which is the filtration generated by all the history up to time t included except for the noise added by the mechanism $\mathcal{M}_{\text{LDP}}$ that is to say $\mathcal{F}_t = \sigma((x_{l,a_l}, r_l)_{l\leq t}, (Y_{l,i,j,q})_{l<t-1,i,j\leq d,q\leq m}, (Z_{l,i,j,q})_{l<t-1,i,j\leq d,q\leq m}, (w_{t,k},$ Therefore, we have that:

$$\begin{aligned} &\mathbb{E}\left((p - Y_{t,k,l,q})w_{t,k,l,q} \mid \mathcal{F}_t\right) = \mathbb{E}(p - Y_{t,k,l,q})\mathbb{E}\left(w_{t,k,l,q} \mid \mathcal{F}_t\right) = 0 \\ &\mathbb{E}\left(A_{t,k,l} - (m\widetilde{z}_{t,k,l} - \kappa_{t,k,l} + 1) \mid \mathcal{F}_t\right) = \mathbb{E}(A_{t,k,l}\mathcal{F}_t) - (m\widetilde{z}_{t,k,l} - \kappa_{t,k,l} + 1) = 0 \end{aligned} \tag{3.131}$$

because Y_t is independent of $\mathcal{F}_t$ and w_t. The second equality comes from the fact that given $\mathcal{F}_t$, $A_{t,k,l}$ is a Bernoulli random variable with parameter $m\widetilde{z}_{t,k,l} - \kappa_{t,k,l} + 1$.

Hence, when choosing $\lambda_j = \frac{\sqrt{8t_j \ln(2t_j/\delta)}}{m} + \frac{2\sqrt{8t_j \ln(2t_j/\delta)}}{(1-p)\sqrt{m}}$, we have that with probability at least $1-\delta$:

$$\forall j \leq M_S, \qquad \lambda_{\min}\left(\widetilde{V}_j - \sum_{t=1}^{t_j} \frac{x_{t,a_t}x_{t,a_t}^\intercal}{2L^2}\right) \geq \frac{\sqrt{8t_j \ln\left(\frac{2t_j}{\delta}\right)}}{m} + \frac{2\sqrt{8t_j \ln\left(\frac{2t_j}{\delta}\right)}}{(1-p)\sqrt{m}} + \tag{3.132}$$

$$\lambda_{\max}\left(\widetilde{V}_j - \sum_{t=1}^{t_j} \frac{x_{t,a_t}x_{t,a_t}^\intercal}{2L^2}\right) \leq \frac{2\sqrt{8t_j \ln\left(\frac{2t_j}{\delta}\right)}}{m} + \frac{4\sqrt{8t_j \ln\left(\frac{2t_j}{\delta}\right)}}{(1-p)\sqrt{m}} \tag{3.133}$$

In addition, with the same reasoning, we have with probability at least $1-\delta$:

$$\left\| \sum_{l=1}^{t_{j+1}} \frac{r_l x_{l,a_l}}{2L} - B_j \right\| \leq 2\sqrt{dp\left(1-\frac{p}{2}\right)t_j m \log\left(\frac{2t_j}{\delta}\right)} + \frac{4}{3}\sqrt{d}\log\left(\frac{2t_j}{\delta}\right) + \frac{2}{m}\sqrt{dt_j \log\left(\frac{2t_j}{\delta}\right)} \tag{3.134}$$

Therefore, using Prop. 5 in (Shariff and Sheffet, 2018), we have that the result.

3.B.2.2 Proof of Prop. 7

We now move to prove the following proposition which implies Prop. 7;

Proposition 17. *For any encoding parameter $m \in \mathbb{N}^\star$ and LDP parameter $\varepsilon_0 > 0$, $\mathcal{M}_{LDP}(x, r)$ is ε_0-LDP for any $\|x\| \leq L$ and $r \in [0, 1]$.*

Proof. For any $x, x' \in \mathbb{R}^d$ and $r, r' \in [0, 1]$ such that $\|x\| \leq L$ and $\|x'\| \leq L$ let's note $\mathcal{M}_{\text{LDP}}(x, r) = \left((\tilde{w}_{i,j})_{i,j\leq d}, (\tilde{b}_j)_{j\leq d}\right) \in \{0,1\}^{d^2m\times dm}$ and $\mathcal{M}_{\text{LDP}}(x', r') = \left((\tilde{w}'_{i,j})_{i,j\leq d}, (\tilde{b}'_j)_{j\leq d}\right) \in \{0,1\}^{d^2m\times dm}$. Therefore, let's consider a tuple $(W_0, B_0) \in \{0,1\}^{d^2m\times dm}$ then we want to show that:

$$\mathbb{P}\left(\mathcal{M}_{\text{LDP}}(x, r) = (W_0, B_0)\right) \leq e^{\varepsilon_0}\mathbb{P}\left(\mathcal{M}_{\text{LDP}}(x', r') = (W_0, B_0)\right) \tag{3.135}$$

But we have:

$$\mathbb{P}\left(\forall i, j \leq d, \tilde{w}_{i,j} = W_{0,i,j}, \tilde{b}_j = B_{0,j}\right) = \mathbb{P}\left(\forall i, j \leq d, \tilde{w}_{i,j} = W_{0,i,j}\right)\mathbb{P}\left(\forall j \leq d, \tilde{b}_j = B_{0,j}\right) \tag{3.136}$$

In addition, because the mechanism $R_p^{0/1}$ is an example of a randomized response mechanism (Dwork et al., 2010a), we have that for all $j \leq d, q \leq m$, $\mathbb{P}(R_p^{0/1}(b_{j,m}) \mid b_{j,m}) \leq (2/p-1)\mathbb{P}(R_p^{0/1}(b'_{j,m}) \mid b'_{j,m})$. Therefore, because of the independence of the sequence $(\tilde{b}_j)_j$:

$$\begin{aligned} \mathbb{P}\left(\forall j \leq d, \tilde{b}_j = B_{0,j}\right) &= \prod_{j,q}\mathbb{P}\left(\tilde{b}_{j,q} = B_{0,j,q}\right) \\ &\leq \prod_{j,q}\mathbb{P}\left(\tilde{b}'_{j,q} = B_{0,j,q}\right)\left(\frac{2}{p} - 1\right) = \left(\frac{2}{p} - 1\right)^{dm}\mathbb{P}\left(\forall j, \tilde{b}'_j = B_{0,j}\right) \end{aligned} \tag{3.137}$$

For all $i,j \leq d$, we have that $\tilde{w}_{i,j} = \tilde{w}_{j,i}$ therefore:

$$\begin{aligned}\mathbb{P}\left(\forall i,j \leq d, \tilde{w}_{i,j} = W_{0,i,j}\right) &= \prod_{i,j\leq i,q} \mathbb{P}\left(\tilde{w}_{i,j,q} = W_{0,i,j,q}\right) \\ &\leq \prod_{i,j\leq i,q} \left(\frac{2}{p} - 1\right) \mathbb{P}\left(\tilde{w}'_{i,j,q} = W_{0,i,j,q}\right) \\ &= \left(\frac{2}{p} - 1\right)^{md(d+1)/2} \mathbb{P}\left(\forall i,j \leq d, \tilde{w}'_{i,j} = W_{0,i,j}\right)\end{aligned} \tag{3.138}$$

Hence the resulting when setting $p = \frac{2}{\exp\left(\frac{\varepsilon_0}{md(d+3)/2}\right)+1}$. □

3.B.2.3 Proof of Thm. 8

Before proving the JDP guarantees of our algorithm, that is to say Thm. 8. We first prove the following proposition that is a consequence of Thm. 5.4 in (Cheu et al., 2019).

Proposition 18. *For any $\delta_0, \delta \in (0,1)$, number of batch M_S and length l, encoding parameter m, LDP parameter $0 < \varepsilon_0 \leq \ln\left(\frac{l}{(7\ln(8m/\delta_0))} - 1\right)$ and for all batch $j \leq M_S$ of length l, the statistics (Z_j, U_j) computed by the shuffler (with $p = 2/(e^{2\varepsilon_0/md(d+3)}+1)$)) are $(\varepsilon_{j,c}, \delta+\delta_0)$-DP with*

$$\frac{\varepsilon_{j,c}}{2d(d+3)\sqrt{8m\log(8m/\delta_0)}} = \left(1 - \left(p - \sqrt{\frac{2p\log\left(\frac{2m}{\delta_0}\right)}{l}}\right)\right) \sqrt{\frac{32\log(8m/\delta_0)}{l\left(p - \sqrt{\frac{2p\log(8\delta_0/m)}{l}}\right)}} \tag{3.139}$$

of Prop. 18. Let's consider $\delta \in (0,1)$ and define

$$\begin{aligned}E_\delta = \bigcap_{T=1}^{+\infty} \Bigg\{ &\left\| \frac{1}{m(1-p)} \sum_{t=1}^{T}\sum_{q=1}^{m} Y_{t,\ldots,q} Z_{t,\ldots,q} - \frac{p}{2}\mathbb{1}\mathbb{1}^\intercal \right\| \\ &+ \left\| \frac{1}{m}\sum_{t=1}^{T} A_t - (m\widetilde{z}_t - \widetilde{\theta}_t + 1) \right\| \\ &+ \left\| \frac{1}{m(1-p)} \sum_{t=1}^{T}\sum_{q=1}^{m} (p - Y_{t,\ldots,q}) w_{t,\ldots,q} \right\| \leq \frac{\sqrt{8T\ln(2T/\delta)}}{m} + \frac{2\sqrt{8T\ln(2T/\delta)}}{(1-p)\sqrt{m}} \Bigg\}\end{aligned}$$

This event is such that $\mathbb{P}(E_\delta) \geq 1-\delta$. Therefore for a batch j and any event A, we have that:

$$\begin{aligned}&\mathbb{P}\left(\left(\mathcal{M}_{LDP}(x_{\sigma_j(t)}x_{\sigma_j(t)}^\intercal, r_{\sigma_j(t)}x_{\sigma_j(t)})\right)_{t\in[\![t_j+1,t_{j+1}]\!]} \in A\right) = \\ &\mathbb{P}\left(\left(\mathcal{M}_{LDP}(x_{\sigma_j(t)}x_{\sigma_j(t)}^\intercal, r_{\sigma_j(t)}x_{\sigma_j(t)})\right)_{t\in[\![t_j+1,t_{j+1}]\!]} \in A, \mathcal{E}_\delta\right) \\ &+ \mathbb{P}\left(\left(\mathcal{M}_{LDP}(x_{\sigma_j(t)}x_{\sigma_j(t)}^\intercal, r_{\sigma_j(t)}x_{\sigma_j(t)})\right)_{t\in[\![t_j+1,t_{j+1}]\!]} \in A, \mathcal{E}_\delta^c\right)\end{aligned}$$

Therefore, we have that:

$$\begin{aligned}&\mathbb{P}\left(\left(\mathcal{M}_{LDP}(x_{\sigma_j(t)}x_{\sigma_j(t)}^\intercal, r_{\sigma_j(t)}x_{\sigma_j(t)})\right)_{t\in[\![t_j+1,t_{j+1}]\!]} \in A\right) \leq \\ &\mathbb{P}\left(\left(\mathcal{M}_{LDP}(x_{\sigma_j(t)}x_{\sigma_j(t)}^\intercal, r_{\sigma_j(t)}x_{\sigma_j(t)})\right)_{t\in[\![t_j+1,t_{j+1}]\!]} \in A, \mathcal{E}_\delta\right) + \delta\end{aligned}$$

And thanks to the definition of privacy with shuffling we have that:

$$\begin{aligned}&\mathbb{P}\left(\left(\mathcal{M}_{LDP}(x_{\sigma_j(t)}x_{\sigma_j(t)}^\intercal, r_{\sigma_j(t)}x_{\sigma_j(t)})\right)_{t\in[\![t_j+1,t_{j+1}]\!]} \in A\right) \leq \\ &\mathbb{P}\left(\left(\mathcal{M}_{LDP}(x'_{\sigma_j(t)}(x'_{\sigma_j(t)})^\intercal, r_{\sigma_j(t)}x_{\sigma_j(t)})\right)_{t\in[\![t_j+1,t_{j+1}]\!]} \in A\right)\exp(\varepsilon_{j,c}) + \delta_0 + \delta\end{aligned}$$

where $\varepsilon_{j,c}$ is such that:

$$\frac{\varepsilon_{j,c}}{2d(d+3)\sqrt{8m\log\left(\frac{8m}{\delta_0}\right)}} = \left(1-\left(p-\sqrt{\frac{2p\log\left(\frac{2m}{\delta_0}\right)}{l}}\right)\right)\sqrt{\frac{32\log(8m/\delta_0)}{l\left(p-\sqrt{\frac{2p\log(8\delta_0/m)}{l}}\right)}} \tag{3.140}$$

according to Thm. 5.4 in (Cheu et al., 2019). □

Now let's consider a set of parameters $\delta_0, \delta \in (0,1)$ and $\varepsilon, \varepsilon_0 \in (0,1)$ and a length l that satisfies Eq. (3.9). Such length l exists for any $p \in [0,1]$ as

$$\begin{aligned}\lim_{l\to+\infty} 2\sqrt{\frac{2\log(2m/\delta_0)}{l}} + \left(\frac{l\varepsilon}{2^5 d(d+3)\log(8m/\delta_0)\sqrt{2T\ln(1/\delta_0)}}\right)^2 \\ - \sqrt{\left(2+\left(\frac{l\varepsilon}{2^5 d(d+3)\log(8m/\delta_0)\sqrt{2T\ln(1/\delta_0)}}\right)^2\right)^2 - 4} = -2\end{aligned} \tag{3.141}$$

Therefore, thanks to Prop. 18, we have that each update to the design matrix is $\left(\frac{\varepsilon\sqrt{l}}{\sqrt{T}}, \delta_0+\delta\right)$-DP. Therefore, using advanced composition yields the result.

3.B.2.4 Proof of Thm. 9

Let's now move on to the proof of the main theorem, Thm. 9. Let's note $l^\star = T/M_S$ where M_S is the number of batch from the shuffler point of view, this parameter is given to the shuffler. Now let's consider a shuffler batch $j \leq M_S$, sent to the bandit algorithm, let's note then $q_j < j$ the last shuffler batch where Alg. 12 has updated the estimate $\widetilde{\theta}$. Therefore, if Alg. 12 decides to update the parameter $\widetilde{\theta}$ after receiving the data from the shuffler batch j, we have that:

$$\det(\tilde{V}_j) \geq (1+\eta)\det(\tilde{V}_{q_j}) \tag{3.142}$$

Let's consider any bandit batch r, between time t_r+1 and t_{r+1} we can then decompose the interval $\{t_r+1,\dots,t_{r+1}\}$ into successive shuffler batches and we note the last of them j_r. That is to say, upon receiving the shuffler batch j_r and t_{j_r} the time step at which this batch begins, Alg. 12 updates the parameter $\widetilde{\theta}$, so increasing the bandit batch from r to $r+1$. Therefore, for all shuffler batch $j \leq j_r - 1$, we have that $\det(\widetilde{V}_j) \leq (1+\eta)\det(\widetilde{V}_r)$ therefore for any vector $x \in \mathbb{R}^d$, $\left|\langle \theta^\star - \widetilde{\theta}_r, x\rangle\right| \leq \sqrt{1+\eta}\beta_r \|x\|_{\widetilde{V}_r^{-1}}$ (see App. D in (Abbasi-Yadkori et al., 2011)). In addition, for any time step t during a batch j, $\|x\|_{\widetilde{V}_j^{-1}} \leq \|x\|_{\widetilde{V}_t^{-1}}$ where $\widetilde{V}_t$ is the design matrix computed with only data from the first t time steps. In addition for $t \in \{t_{j_r},\dots,t_{r+1}\}$, we have that the norm $\|x\|_{\widetilde{V}_r^{-1}}$ can not be related to the norm of $\|x\|_{\widetilde{V}_t^{-1}}$ but we have that:

$$\left|\langle \theta^\star - \widetilde{\theta}_r, x\rangle\right| \leq \beta_r \|x\|_{\widetilde{V}_r^{-1}} \leq \frac{\beta_r \|x\|_2}{\sqrt{\lambda_{\min}(V_r)}} \tag{3.143}$$

Therefore, we can write the regret as:

$$R_T = \sum_{t=1}^{T} \langle \theta^\star, x_{t,a_t^\star} - x_{t,a_t}\rangle = \sum_{p=0}^{M_R}\sum_{t=t_p+1}^{t_{p+1}} \langle \theta^\star, x_{t,a_t^\star} - x_{t,a_t}\rangle \tag{3.144}$$

where M_R is the number of batch of Alg. 12. Using the reasoning above, we have:

$$R_T \leq \sum_{p=0}^{M_R-1}\sum_{t=t_p+1}^{t_{j_p}} 2\beta_p\sqrt{1+\eta}\|x_{t,a_t}\|_{\widetilde{V}_t^{-1}} + \sum_{t=t_{j_p}+1}^{t_{p+1}} 2\beta_p \|x_{t,a_t}\|_{\widetilde{V}_{t_p}^{-1}} \tag{3.145}$$

$$\leq 2\beta_T \sum_{t=1}^{T}\sqrt{1+\eta}\|x_{t,a_t}\|_{\widetilde{V}_t^{-1}} + \sum_{p=0}^{M_R-1}\frac{2\beta_p L l^\star}{\sqrt{\lambda_{\min}(\widetilde{V}_p)}} \tag{3.146}$$

where $l^\star$ is the length of a shuffler batch. In addition, the design matrix $\widetilde{V}_p$ is regularized to ensure that its minimum eigenvalues grows at a rate of $\sqrt{t_p}$. Therefore we have that for any bandit algorithm batch r:

$$\frac{2\beta_r L l^\star}{\sqrt{\lambda_{\min}(\widetilde{V}_r)}} \leq 2Ll^\star \left(\frac{\sigma\sqrt{2\log\left(\frac{2T}{\delta}\right) + d\log\left(3+\frac{TL^2}{\lambda}\right)}}{\sqrt{\lambda_r}} + S\sqrt{3} + \frac{d\left(\sqrt{t_r m \log\left(\frac{2}{\delta}\right)} + \frac{2\log(2/\delta)}{3} + \frac{\sqrt{2}}{m}\sqrt{t_r \log\left(\frac{2}{\delta}\right)}\right)}{\lambda_r} \right)$$

Therefore using (Carpentier et al., 2020), the regret can be bounded by:

$$R_T \leq 2\beta_T\sqrt{(1+\eta)\,T\log\left(1+\frac{T}{d\lambda}\right)} + \sum_{r=0}^{M_R-1} 2Ll^\star \left(\frac{\sigma\sqrt{2\log\left(\frac{2T}{\delta}\right) + d\log\left(3+\frac{TL^2}{\lambda}\right)}}{\sqrt{\lambda_r}} + S\sqrt{3} + \frac{d\left(\sqrt{t_r m \log\left(\frac{2}{\delta}\right)} + \frac{2\log(2/\delta)}{3} + \frac{\sqrt{2}}{m}\sqrt{t_r \log\left(\frac{2}{\delta}\right)}\right)}{\lambda_r} \right) \tag{3.147}$$

We now proceed to bound each term individually. First, we have:

$$\sum_{r=0}^{M_R-1} 2\sqrt{3}Ll^\star S \leq 2\sqrt{3}LSl^\star M_R \tag{3.148}$$

This is because the shuffler sends data on a fix length schedule. Also, we have:

$$\sum_{r=0}^{M_R-1} 2Ll^\star \frac{\sigma\sqrt{2\log\left(\frac{2T}{\delta}\right) + d\log\left(3+\frac{TL^2}{\lambda}\right)}}{\sqrt{\lambda_r}} \leq \frac{2M_R Ll^\star\sigma}{\sqrt{\lambda}}\sqrt{2\log\left(\frac{2}{\delta}\right) + d\log\left(3+\frac{TL^2}{\lambda}\right)} \tag{3.149}$$

Finally,

$$\sum_{r=0}^{M_R-1} \frac{2Ll^\star d}{\lambda_r}\left(\sqrt{t_r m\log\left(\frac{2}{\delta}\right)} + \frac{2\log(2T/\delta)}{3} + \frac{\sqrt{2}}{m}\sqrt{t_r\log\left(\frac{2}{\delta}\right)}\right) \leq \frac{4Ll^\star d M_R}{3}\log\left(\frac{2T}{\delta}\right) + \frac{\sqrt{2}Ll^\star d M_R m}{4} \tag{3.150}$$

Therefore with probability at least $1-\delta$ the regret is bounded by:

$$R_T \leq \underbrace{2\beta_T\sqrt{(1+\eta)\,T\log\left(1+\frac{T}{d\lambda}\right)}}_{:=ⓐ} + \frac{2M_R Ll^\star\sigma}{\sqrt{\lambda}}\sqrt{2\log\left(\frac{2}{\delta}\right) + d\log\left(3+\frac{TL^2}{\lambda}\right)} + \frac{4Ll^\star d M_R}{3}\log\left(\frac{2T}{\delta}\right) + \frac{\sqrt{2}Ll^\star d M_R m}{4} + 2\sqrt{3}LSl^\star M_R \tag{3.151}$$

Bounding ⓐ. Given the expression of β_T, we have that:

$$ⓐ \leq \sigma\sqrt{\left(8\log\left(\frac{2T}{\delta}\right) + d\log\left(3+\frac{TL^2}{\lambda}\right)\right)(1+\eta)\,T\log\left(1+\frac{T}{d\lambda}\right)} + \frac{2\sqrt{3}ST^{1/4}}{\sqrt{1-p}}\sqrt{(1+\eta)\,T\log\left(1+\frac{T}{d\lambda}\right)} + \left(4dmT^{1/4} + \frac{8\log(2T/\delta)\sqrt{m}}{3}\right)\sqrt{(1+\eta)\,T\log\left(1+\frac{T}{d\lambda}\right)} \tag{3.152}$$

Now, we are left with bounding the remaining of the right hand part of Eq. (3.151). The first step to do so is to notice that the number of bandit algorithm batch is bounded by roughly $\mathcal{O}\left(\log(T)\right)$, more precisely:

$$M_R \leq 1 + \frac{d\log\left(\frac{L^2T}{d} + \frac{16\sqrt{T}\log(2T/\delta)}{(1-p)}\right)}{\log(1+\eta)} \tag{3.153}$$

In addition, if $l^\star$ satisfies Eq. (3.9) then we have that:

$$l^\star \leq \max\left\{\frac{8\log(2m/\delta_0)}{p^2}, \frac{128\sqrt{2T\ln(2/\delta_0)}d(d+1)\log(8m/\delta_0)(1-p)}{\varepsilon}, \frac{14\log(2m/\delta_0)}{p}\right\} \tag{3.154}$$

In Eq. (3.154) we have that the regret is bounded with probability at least $1-\delta$:

$$\begin{aligned} R_T \leq & \frac{2\sqrt{3}(S+md)T^{3/4}}{\sqrt{1-p}}\sqrt{(1+\eta)\log\left(1+\frac{T}{d\lambda}\right)} \\ & + \frac{dLm}{\sqrt{\lambda}}\left(1+\frac{d^{3/2}\log\left(\frac{L^2T}{d}+\frac{16\sqrt{T}\log(2T/\delta)}{(1-p)}\right)^{3/2}}{\log(1+\eta)}\right)\times \\ & \times\max\left\{\frac{14\log(8m/\delta_0)}{p^2}, \frac{128\sqrt{2T\ln\left(\frac{2}{\delta_0}\right)}d(d+1)\log\left(\frac{8m}{\delta_0}\right)(1-p)}{\varepsilon}\right\} \end{aligned} \tag{3.155}$$

Therefore, we can differentiate two different scenarios:

- If $p^2(1-p) \leq \frac{7T^{-1/2}\varepsilon}{64\sqrt{2\ln(2/\delta_0)}d(d+1)}$:

$$R_T \leq \frac{2\sqrt{3}(S+md)T^{3/4}}{\sqrt{1-p}}\sqrt{(1+\eta)\log\left(1+\frac{T}{d\lambda}\right)} \tag{3.156}$$

$$+ \frac{dLm}{\sqrt{\lambda}}\left(1+\frac{d^{3/2}\log\left(\frac{L^2T}{d}+\frac{16\sqrt{T}\log(2T/\delta)}{(1-p)}\right)^{3/2}}{\log(1+\eta)}\right)\frac{14\log(8m/\delta_0)}{p^2} \tag{3.157}$$

- If $p^2(1-p) \geq \frac{7T^{-1/2}\varepsilon}{64\sqrt{2\ln(2/\delta_0)}d(d+1)}$:

$$\begin{aligned} R_T \leq & \frac{2\sqrt{3}(S+md)T^{3/4}}{\sqrt{1-p}}\sqrt{(1+\eta)\log\left(1+\frac{T}{d\lambda}\right)} \\ & + \frac{264}{\sqrt{\lambda}}\sqrt{2}d^3\log\left(\frac{8m}{\delta_0}\right)^{3/2}Lm\left(1+\frac{d^{3/2}\log\left(\frac{L^2T}{d}+\frac{16\sqrt{T}\log(2T/\delta)}{(1-p)}\right)^{3/2}}{\log(1+\eta)}\right)\frac{\sqrt{T}(1-p)}{\varepsilon} \end{aligned} \tag{3.158}$$

The last step now is to choose the parameter ε_0 to optimize the regret. Therefore, if $\varepsilon \leq \frac{1}{27T^{1/4}}$, so in a high privacy regime, when choosing $p = 1-\varepsilon^{2/3}T^{1/6}$ we are in the second scenario above and:

$$\begin{aligned} R_T \leq \frac{T^{2/3}}{\varepsilon^{1/3}}\Bigg[& \frac{264}{\sqrt{\lambda}}\sqrt{2}d^3\log\left(\frac{8m}{\delta_0}\right)^{3/2}Lm\left(1+\frac{d^{3/2}\log\left(\frac{L^2T}{d}+\frac{16\sqrt{T}\log(2T/\delta)}{(1-p)}\right)^{3/2}}{\log(1+\eta)}\right) \\ & + 2\sqrt{3}(S+md)\sqrt{(1+\eta)\log\left(1+\frac{T}{d\lambda}\right)}\Bigg] \end{aligned}$$

3.B.3 Regret with Scheduled Update Algorithm

In this appendix, we present a bandit algorithm using a fixed schedule update instead of the determinant based condition used in Alg. 12. The main consequence of using a fixed batch bandit algorithm is a worse regret compared to Alg. 12. That is a consequence of the inflated bonus needed by the use of the local randomizer algorithm $\mathcal{M}_{\text{LDP}}$. Let's consider the batched algorithm described in Alg. 23.

In terms of privacy Alg. 23 enjoys the same guarantees as Alg. 12. For any $\delta \in (0,1)$, we have that with probability at least $1-\delta$:

$$R_T = \sum_{t=1}^{T}\langle\theta^\star, x_{t,a_t^\star} - x_{t,a_t}\rangle = \sum_{j=1}^{M}\sum_{t=t_j+1}^{t_{j+1}}\langle\theta^\star, x_{t,a_t^\star} - x_{t,a_t}\rangle \tag{3.159}$$

Algorithm 23: FixedBatchedShuffling-LinUCB

Input: LDP parameter: ε_0, privacy parameter: ε, δ', regularization parameter:λ, context bound:L, failure probability:δ, low switching parameter: η, encoding parameter:m, dimension: d

Initialize $j^S = j^A = 0$, $\widetilde{\theta}_0 = 0$, $\widetilde{V}_0 = \lambda I_d$, $p = \frac{2}{\exp(2\varepsilon_0/(md(d+3)))+1}$

for $t = 0, 1, \ldots$ **do**

User receives $\widetilde{\theta}_{j^A}$, $\widetilde{V}_{j^A}$ and β_{j^A} and selects $a_t \in \arg\max_{a\in[K]} \langle x_{t,a}, \widetilde{\theta}_{j^A}\rangle + \beta_{j^A} \| x_{t,a}\|_{\widetilde{V}_{j^A}^{-1}}$

Observe reward r_t and compute private statistics $(\widetilde{b}_t, \widetilde{w}_t) = \mathcal{M}_{\mathrm{LDP}}((x_{t,a_t}, r_t), p, m, L)$

Communication with the shuffler

$B^S_{j^S} = B^S_{j^S} \cup (\widetilde{b}_t, \widetilde{w}_t)$

if $|B^S_{j^S}| = l$ **then**

Set $t_{j^S+1} = t$, compute a permutation σ of $[\![t_{j^S}+1, t_{j^S+1}]\!]$ and compute aggregate statistics

$$\forall i \le d, k \le i, \qquad Z_{j^S,i} = \sum_{n=1}^{l}\sum_{q=1}^{m} \tilde{b}_{\sigma(n),i,q} \quad \text{and} \quad U_{j^S,i,k} = \sum_{n=1}^{l}\sum_{q=1}^{m} \widetilde{w}_{\sigma(n),i,k,q}$$

Set $U_{j^S,i,k} = U_{j^S,k,i}$, $B_{j^S+1} = \emptyset$ and $j^S = j^S + 1$

Communication with the bandit algorithm

Receives (Z_{j^S-1}, U_{j^S-1}) and compute candidate statistics

$$\widetilde{B}_{j^A+1} = \widetilde{B}_{j^A+1} + \frac{Z_{j^S-1}}{m(1-p)} - \frac{l^S}{2(1-p)}$$
$$\widetilde{V}_{j^A+1} = \widetilde{V}_{j^A+1} + \frac{U_{j^S-1}}{m(1-p)} - \frac{l^S}{2(1-p)} + 2(\lambda_{j^A+1} - \lambda_{j^A})I_d$$

Compute $\theta_{j^A+1} = \frac{1}{L}\widetilde{V}^{-1}_{j^A+1}\widetilde{B}_{j^A+1}$

Set $t_{j^A+1} = t$, β_{j^A+1} and λ_{j^A+1} as in Eq. (3.13) and Eq. (3.14)

Set $j^A = j^A + 1$, $\widetilde{B}_{j^A+1} = \widetilde{B}_{j^A}$ and $\widetilde{V}_{j^A+1} = \widetilde{V}_{j^A}$

But using Lem.3 in (Han et al., 2020), we have that for any batch j:

$$\begin{aligned}\sum_{j=1}^{M}\sum_{t=t_j+1}^{t_{j+1}}\|x_{t,a_t}\|_{\widetilde{V}_j^{-1}} &\leq \sqrt{\frac{T}{M}}\sum_{j=1}^{M}\sqrt{\mathsf{Tr}\left(\widetilde{V}_j^{-1}\sum_{t=t_j+1}^{t_{j+1}}x_{t,a_t}x_{t,a_t}^{\intercal}\right)} \\ &\leq \sqrt{\frac{10T}{M}}\log(T+1)\left(\sqrt{Md}+d\sqrt{\frac{T}{M}}\right)\end{aligned} \tag{3.160}$$

where M is the total number of batch. Therefore, the regret is bounded with high probability by:

$$R_T \leq 2\beta_M\sqrt{\frac{10T}{M}}\log(T+1)\left(\sqrt{Md}+d\sqrt{\frac{T}{M}}\right) = \mathcal{O}\left(\frac{T^{3/4}}{\sqrt{1-p}}+\frac{T^{3/4}\sqrt{1-p}}{\varepsilon}\right) \tag{3.161}$$

Where we used the fact that M is defined in Eq. (3.154). Therefore, using a fixed schedule algorithm the trade-off highlighted in Thm. 9 does not appear.

Chapter 4

Secure Reinforcement Learning

The last chapter of this thesis deals with the problem of security in Reinforcement Learning and the associated constraints. We consider two different aspects of this problem. The first one is adversarial attacks on linear contextual bandit. The objective is to understand how sensitive to perturbation in the observed rewards and contexts standard algorithms like Thompson Sampling or UCB are. That is to say, we aim to answer to the different questions of what happens when an attacker can control the rewards observed by the bandit algorithm (while minimizing the total change between the true rewards and the ones observed by the bandit algorithm) or when the attacker can modify the contexts but not the reward. The second problem (and potential solution to the problem mentioned above) is about encryption in linear contextual bandit. Indeed, most software uses end-to-end encryption to secure the communication between the users and the servers. Some application go even further by providing a service to the users without observing the true data but only an encrypted version of it. Combined with the previous result about attackers able to perturb the algorithm feedback, we aim to design a secure bandit algorithm leveraging homomorphic encryption to prevent any outside attackers to modify the reward and context sent to the algorithm. But also, to prevent the bandit algorithms to observe true contexts and rewards, solely encrypted version of it. The resulting algorithm designed around homomorphic encryption for linear contextual bandit has to trade-off constraints like being able to perform a finite number of multiplication before making the data indecipherable and minimizing regret. To do so, we leverage techniques from the slow-switching in bandit literature and present an algorithm with a regret similar to the non-encrypted case.

This chapter is based on the two following articles:

- **Evrard Garcelon**, Baptiste Roziere, Laurent Meunier, Jean Tarbouriech, Olivier Teytaud, Alessandro Lazaric, and Matteo Pirotta. Adversarial attacks on linear contextual bandits. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14362–14373. Curran Associates, Inc., 2020c. URL https://papers.nips.cc/paper/2020
- **Evrard Garcelon**, Matteo Pirotta, and Vianney Perchet. Encrypted linear contextual bandit. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 2519–2551. PMLR, 28–30 Mar 2022c. URL https://proceedings.mlr.press/v151/garcelon22a.html

Contents

4.1 Attacks on Linear Contextual Bandit

Recommender systems are at the heart of the business model of many industries like e-commerce or video streaming (Davidson et al., 2010; Gomez-Uribe and Hunt, 2015). The two most common approaches for this task are based either on matrix factorization (Park et al., 2017) or bandit algorithms (Li et al., 2010), which both rely on a unaltered feedback loop between the recommender system and the user. In recent years, a fair amount of work has been dedicated to understanding how targeted perturbations in the feedback loop can fool a recommender system into recommending low quality items.

Following the line of research on adversarial attacks in supervised learning (Biggio et al., 2012; Goodfellow et al., 2014; Jagielski et al., 2018; Li et al., 2016; Liu et al., 2017), attacks on recommender systems have been focused on filtering-based algorithms (Christakopoulou and Banerjee, 2019; Mehta and Nejdl, 2008) and offline contextual bandits (Ma et al., 2018). The question of adversarial attacks for online bandit algorithms has only been studied quite recently (Jun et al., 2018; Liu and Shroff, 2019; Immorlica et al., 2018; Guan et al., 2020), and solely in the multi-armed stochastic setting. Although the idea of online adversarial bandit algorithms is not new (see EXP3 algorithm in Auer et al. (2002b)), the focus is different from what we are considering in this article. Indeed, algorithms like EXP3 or EXP4 (Lattimore and Szepesvári, 2018) are designed to find optimal actions in hindsight in order to adapt to any rewards stream.

The opposition between adversarial and stochastic bandit settings has sparked interests in studying a middle ground. In Bubeck and Slivkins (2012), the learning algorithm has no knowledge of the type of feedback it receives (either stochastic or adversarial). In Lykouris et al. (2018); Li et al. (2019); Gupta et al. (2019); Lykouris et al. (2019); Kapoor et al. (2019), the rewards are assumed to be corrupted by adversarial rewards. The authors focus on building algorithms able to find the optimal actions even in the presence of some non-random perturbations. This setting is different from what is studied in this article because those perturbations are bounded and agnostic to arms pulled by the learning algorithm, i.e., the adversary corrupt the rewards before the algorithm chooses an arm.

In the broader Deep Reinforcement Learning (DRL) literature, the focus is placed on modifying the observations of different states to fool a DRL system at inference time (Hussenot et al., 2019; Sun et al., 2020) or the rewards (Ma et al., 2019). In this chapter, we first follow the research direction opened by Jun et al. (2018) where the attacker has the objective of fooling a learning algorithm into taking a specific action as much as possible. For example in a news recommendation problem, as described in Li et al. (2010), a bandit algorithm chooses between K articles to recommend to a user, based on some information about them, called context. We assume that an attacker sits between the user and the website, they can choose the reward (i.e., click or not) for the recommended article observed by the recommending algorithm. Their goal is to fool the bandit algorithm into recommending some articles to most users. The contributions of our work can be summarized as follows:

- We extend the work of Jun et al. (2018); Liu and Shroff (2019) to the contextual linear bandit setting showing how to perturb rewards for both stochastic and adversarial algorithms, forcing **any** bandit algorithms to pull a specific set of arms, $o(T)$ times for logarithmic cost for the attacker.
- We analyze, for the first time, the setting in which the attacker can only modify the context x associated with the current user (the reward is not altered). The goal of the attacker is to fool the bandit algorithm into pulling arms of a target set for most users (i.e., contexts) while minimizing the total norm of their attacks. We show that the widely known LINUCB algorithm (Abbasi-Yadkori et al., 2011; Lattimore and Szepesvári, 2018) is vulnerable to this new type of attack.
- We present a harder setting for the attacker, where the latter can only modify the context associated to a specific user. This situation may occur when a malicious agent has infected some computers with a Remote Access Trojan (RAT). The attacker can then modify the history of navigation of a specific user and, as a consequence, the information seen by the online recommender system.We show how the attacker can attack the two very common bandit algorithms LINUCB and Linear Thompson Sampling (LINTS) Agrawal and Goyal (2012); Abeille et al. (2017) and, in certain cases, force them to pull a set of arms most of the time when a specific context (i.e., user) is presented to the algorithm (i.e., visits a website).

4.1.1 Linear Contextual Bandit

We consider the standard contextual linear bandit setting with $K \in \mathbb{N}$ arms and feature independent arms described in Section 1.2.2. At each time t, the agent observes a context $x_t \in \mathbb{R}^d$, selects an action $a_t \in [\![1, K]\!]$ and observes a reward: $r_{t,a_t} = \langle \theta_{a_t}, x_t \rangle + \eta^t_{a_t}$ where for each arm a, $\theta_a \in \mathbb{R}^d$ is a feature vector and $\eta^t_{a_t}$ is a conditionally independent zero-mean, σ^2-subgaussian noise. The contexts are assumed to be sampled *stochastically* except in App. 4.A.4.

Assumption 13. *There exist $L > 0$ and $\mathcal{D} \subset \mathbb{R}^d$, such that for all t, $x_t \in \mathcal{D}$ and, $\forall x \in \mathcal{D}, \forall a \in [\![1, K]\!]$, $\|x\|_2 \leq L$ and $\langle \theta_a, x \rangle \in (0, 1]$. In addition, we assume that there exists $S > 0$ such that $\|\theta_a\|_2 \leq S$ for all arms a.*

The agent minimizes the cumulative regret after T steps $R_T = \sum_{t=1}^{T} \langle \theta_{a^\star_t}, x_t \rangle - \langle \theta_{a_t}, x_t \rangle$, where $a^\star_t := \arg\max_a \langle \theta_a, x_t \rangle$. A bandit learning algorithm $\mathfrak{A}$ is said to be *no-regret* when it satisfies $R_T = o(T)$, i.e., the average expected reward received by $\mathfrak{A}$ converges to the optimal one. Classical bandit algorithms (e.g.,LINUCB and LINTS) compute an estimate of the unknown parameters θ_a using past observations. Formally, for each arm $a \in [K]$ we define S^t_a as the set of times up to $t-1$ (included)

where the agent played arm a. Then, the estimated parameters are obtained through regularized least-squares regression as $\widehat{\theta}_a^t = (X_{t,a}X_{t,a}^\top + \lambda I)^{-1}X_{t,a}Y_{t,a}$, where $\lambda > 0$, $X_{t,a} = (x_i)_{i \in S_a^t} \in \mathbb{R}^{d \times |S_a^t|}$ and $Y_{t,a} = (r_{i,a_i})_{i \in S_a^t} \in \mathbb{R}^{|S_a^t|}$. Denote by $V_{t,a} = \lambda I + X_{t,a}X_{t,a}^\top$ the design matrix of the regularized least-square problem and by $\|x\|_V = \sqrt{x^\top V x}$ the weighted norm w.r.t. any positive matrix $V \in \mathbb{R}^{d \times d}$. We define the confidence set:

$$\mathcal{C}_{t,a} = \left\{ \theta \in \mathbb{R}^d \, : \, \left\| \theta - \widehat{\theta}_{t,a} \right\|_{V_{t,a}} \leq \beta_{t,a} \right\} \tag{4.1}$$

where $\beta_{t,a} = \sigma \sqrt{d \log\left((1 + L^2(1 + |S_a^t|)/\lambda)/\delta \right)} + S\sqrt{\lambda}$,

which guarantees that $\theta_a \in \mathcal{C}_{t,a}$, for all $t > 0$, w.p. $1 - \delta$. This uncertainty is used to balance the exploration-exploitation trade-off either through optimism (e.g.,LinUCB) or through randomization (e.g., LinTS).

4.1.2 Online Adversarial Attacks on Rewards

The ultimate goal of a malicious agent is to force a bandit algorithm to perform a desired behavior. An attacker may simply want to induce the bandit algorithm to perform poorly—ruining the users' experience—or to force the algorithm to suggest a specific arm. The latter case is particularly interesting in advertising where a seller may want to increase the exposure of its product at the expense of the competitors. Note that the users' experience is also compromised by the latter attack since the suggestions they will receive will not be tailored to their needs. Similarly to Liu and Shroff (2019); Jun et al. (2018), we focus on the latter objective, i.e., to fool the bandit algorithm into pulling arms in $A^\dagger$, a set of target arms, for $T - o(T)$ time steps (*independently of the user*).

A way to obtain this behavior is to dynamically modify the reward in order to make the bandit algorithm believe that $a^\dagger$ is optimal, for some $a^\dagger \in A^\dagger$. Clearly, the attacker has to pay a price in order to modify the perceived bandit problem and fool the algorithm. If there is no restriction on when and how the attacker can alter the reward, the attacker can easily fool the algorithm. However, this setting is not interesting since the attacker may pay a cost higher than the loss suffered by the attacked algorithm. An attack strategy is considered successful when the total cost of the attack is sublinear in T.

In this subsection , we show that under Assumption 13, there exists an attack algorithm that is successful against any bandit algorithm, stochastic or adverserial.

Setting. We assume that the attacker has the same knowledge as the bandit algorithm $\mathfrak{A}$ about the problem (i.e., knows σ and L). The attacker is assumed to be able to observe the context x_t, the arm a_t pulled by $\mathfrak{A}$, and can modify the reward received by $\mathfrak{A}$. When the attacker modifies the reward r_{t,a_t} into $\widetilde{r}_{t,a_t}$ the *instantaneous cost* of the attack is defined as $c_t := \left| r_{t,a_t} - \widetilde{r}_{t,a_t} \right|$. The goal of the attacker is to fool algorithm $\mathfrak{A}$ such that the arms in $A^\dagger$ are pulled $T - o(T)$ times and $\sum_{t=1}^T c_t = o(T)$. We also assume that the action for the arms in the target set is strictly positive for every context $x \in \mathcal{D}$. That is to say that $\Delta := \min_{x \in \mathcal{D}} \left\{ \langle x, \theta_{a_\star^\dagger(x)} \rangle - \max_{a \in A^\dagger, a \neq a_\star^\dagger(x)} \langle x, \theta_a \rangle \right\} > 0$ where $a_\star^\dagger(x) = \arg\max_{a \in A^\dagger} \langle x, \theta_a \rangle$ for every $x \in \mathcal{D}$.

Attack idea. We leverage the idea presented in Liu and Shroff (2019) and Jun et al. (2018) where the attacker lowers the reward of arms $a \notin A^\dagger$ so that algorithm $\mathfrak{A}$ learns that an arm of the target set is optimal for every context. Since $\mathfrak{A}$ is assumed to be no-regret, the attacker only needs to modify the rewards $o(T)$ times to achieve this goal. Lowering the rewards has the effect of shifting the vectors $(\theta_a)_{a \notin A^\dagger}$ to new vectors $(\theta'_a)_{a \notin A^\dagger}$ such that for all arms $a \notin A^\dagger$ and all contexts $x \in \mathcal{D}$, there exists an arm $a^\dagger \in A^\dagger$ such that $\langle \theta'_a, x \rangle \leq \langle \theta_{a^\dagger}, x \rangle$. Since rewards are assumed to be bounded (see Asm. 13), this objective can be achieved by simply forcing the reward of non-target arms $a \notin A^\dagger$ to the minimum value. Contextual ACE (see Fig. 25) implements a soft version of this idea by leveraging the knowledge of the reward distribution. At each round t, Contextual ACE modifies the reward perceived by $\mathfrak{A}$ as follows:

$$\widetilde{r}^1_{t,a_t} = \eta'_t \mathbb{1}_{\{a_t \notin A^\dagger\}} + r_{t,a_t} \mathbb{1}_{\{a_t \in A^\dagger\}} \tag{4.2}$$

where η'_t is a σ-subgaussian random variable generated by the attacker independently of all other random variables. Contextual ACE transforms the original problem into a *stationary* bandit problem in which there is a targeted arm that is optimal for all contexts and all non targeted arms have expected reward of 0. The following propostion shows that the cumulative cost of the attack is sublinear.

Proposition 1. *For any $\delta \in (0, 1/K]$, when using Contextual ACE algorithm (Fig. 24) with perturbed rewards $\widetilde{r}^1$, with probability at least $1 - K\delta$, algorithm $\mathfrak{A}$ pulls an arm in $A^\dagger$ for $T - o(T)$ time steps and the total cost of attacks is $o(T)$.*

The proof of this proposition is provided in App. 4.A.1.1. While Prop. 1 holds for any no-regret algorithm $\mathfrak{A}$, we can provide a more precise bound on the total cost by inspecting the algorithm. For example, we can show (see App. 4.A.5), that, with probability at least $1 - K\delta$, the number of times LinUCB Abbasi-Yadkori et al. (2011) pulls arms not in $A^\dagger$ is at most $\sum_{j \notin A^\dagger} N_j(T) \leq \frac{64K\sigma^2\lambda S^2}{\Delta^2} \left(d \log\left(\frac{\lambda + \frac{TL^2}{d}}{\delta^2} \right) \right)^2$. This directly translates into a bound on the total cost.

Comparison with ACE Liu and Shroff (2019). In the stochastic setting, the ACE algorithm Liu and Shroff (2019) leverages a bound on the expected reward of each arm in order to modify the reward. However, the perturbed reward process seen by algorithm $\mathfrak{A}$ is non-stationary and in general there is no guarantee that an algorithm minimizing the regret in a stationary bandit problem keeps the same performance when the bandit problem is not stationary anymore. Nonetheless, transposing the idea of the ACE algorithm to our setting would give an attack of the following form, where at time t, Alg. $\mathfrak{A}$ pulls arm a_t and receives rewards $\widetilde{r}^2_{t,a_t}$:

$$\widetilde{r}^2_{t,a_t} = (r_{t,a_t} + \max(-1, \min(0, C_{t,a_t})))\mathbb{1}_{\{a_t \notin A^\dagger\}} + r_{t,a_t}\mathbb{1}_{\{a_t \in A^\dagger\}}$$

with $C_{t,a_t} = (1-\gamma)\min_{a^\dagger \in A^\dagger} \min_{\theta \in \mathcal{C}_{t,a^\dagger}} \langle \theta, x_t \rangle - \max_{\theta \in \mathcal{C}_{t,a_t}} \langle \theta, x_t \rangle$. Note that $\mathcal{C}_{t,a}$ is defined as in Eq. 4.1 using the *non-perturbed* rewards, i.e., $Y_{t,a} = (r_{i,a_i})_{i \in S^t_a}$.

Bounded Rewards. The bounded reward assumption is necessary in our analysis to prove a formal bound on the total cost of the attacks for *any* no-regret bandit algorithm, otherwise we need more information about the attacked algorithm. In practice, the second attack on the rewards, $\widetilde{r}^2$, can be used in the case of unbounded rewards for any algorithms. The difficulty for unbounded reward is that the attacker has to adapt to the environment reward but in order to do so the reward process observed by the bandit algorithm becomes non-stationary under the attack. Thus, there is no guarantee that an algorithm like LinUCB will pull a target arm as the proof relies on the environment observed by the bandit algorithm being stationary. We observe empirically that the total cost of attack is sublinear when using $\widetilde{r}^2$.

Jun et al. (2018) does not assume that rewards are bounded but focus on attacking algorithms in the stochastic multi-armed setting. That is to say they study attacks only designed for ε-greedy and UCB while we provide an efficient attack for any algorithms in the linear contextual case. We can extend their work, and thus remove the bounded reward assumption, in the linear contextual case by using the following attack, designed only for LinUCB:

$$\widetilde{r}^3_{t,a_t} = \left(r_{t,a_t} + \min_{a^\dagger \in A^\dagger} \min_{\theta \in \mathcal{C}_{t,a^\dagger}} \langle \theta, x_t \rangle - \max_{\theta \in \mathcal{C}_{t,a_t}} \langle \theta, x_t \rangle \right) \mathbb{1}_{\{a_t \notin A^\dagger\}} + r_{t,a_t}\mathbb{1}_{\{a_t \in A^\dagger\}} \tag{4.3}$$

with $C_{t,a}$ defined as in Eq. (4.1). Although, the attack $\widetilde{r}^3$ is not stationary, it is possible to prove that the total cost of attack is $\mathcal{O}(\log(T))$ because we know that the attacked bandit algorithm is LinUCB.

Constrained Attack. When the attacker has a constraint on the instantaneous cost of the attack, using the perturbed reward $\widetilde{r}^1$ may not be possible as the cost of the attack at time t is not decreasing over time. Using the perturbed reward $\widetilde{r}^2$ offers a more flexible type of attack with more control on the instantaneous cost thanks to the parameter γ. But it still suffers from a minimal cost of attack from lowering rewards of arms not in $A^\dagger$.

Defense mechanism. The attack based on reward $\widetilde{r}_1$ is hardly detectable without prior knownledge about the problem. In fact, the reward process associated to $\widetilde{r}_1$ is stationary and compatible with the assumption about the true reward (e.g., subgaussian). While having very low rewards is reasonable in advertising, it can make the attack easily detectable in some other problems. On the other hand, the fact that $\widetilde{r}_2$ is a non-stationary process makes this attack easier to detect. When some data are already available on each arm, the learner can monitor the difference between the average rewards per action computed on new and old data.

4.1.3 Online Adversarial Attacks on Contexts

In this subsection , we consider the attacker to be able to alter the context x_t perceived by the algorithm rather than the reward. The attacker is now restricted to change the type of users presented to the learning algorithm $\mathfrak{A}$, hence changing its perception of the environment. We show that under the assumption that the attacker knows a lower-bound to the reward of the target set, it is possible to fool LinUCB.

Setting. As in Sec. 4.1.2, we consider the attacker to have the same knowledge about the problem as $\mathfrak{A}$. The main difference with the previous setting is that the attacker attacks before the algorithm. We adopt a *white-box* Goodfellow et al. (2014) setting attacking LinUCB. The goal of the attacker is unchanged: they aim at forcing the algorithm to pull arms in $A^\dagger$ for $T - o(T)$ time steps while paying a sublinear total cost. We denote by $\widetilde{x}_t$ the context after the attack and by $c_t = \|x_t - \widetilde{x}_t\|_2$ the instantaneous cost.

Difference between attacks on contexts and rewards. Perturbing contexts is fundamentally different from perturbing the rewards. The attacker only modifies the context that is shown to the bandit algorithm. The true context, which is used to compute the reward, remains unchanged. In other words, the attacker cannot modify the reward observed by the bandit algorithm. Instead, the attack algorithm described in this subsection fools the bandit algorithm by making the rewards appear small relative to the contexts and requires more assumptions on the bandit algorithm than in Sec. 4.1.2.

Attack Idea. The idea of the attack in this setting is similar to the attack of Sec. 4.1.2. The attacker builds a bandit problem where arm an $a^\dagger \in A^\dagger$ is optimal for all contexts by lowering the perceived value of all other arms not in $A^\dagger$. The attacker cannot modify the reward but, thanks to the linear reward assumption, they can scale the contexts to decrease the predicted rewards in the original context.

At time t, the attacker receives the context x_t and computes the attack. Thanks to the white-box setting, it computes the arm a_t that algorithm $\mathfrak{A}$ would pull if presented with context x_t. If $a_t \notin A^\dagger$ then the attacker changes the context to $\widetilde{x}_t = \alpha_{a_t} x_t$ with $\alpha_{a_t} > \max_{x \in \mathcal{D}} \min_{a^\dagger \in A^\dagger} \langle \theta_{a_t}, x \rangle / \langle \theta_{a^\dagger}, x \rangle$.This factor is chosen such that for a ridge regression computed on the dataset $(\alpha x_i, \langle \theta, x_i \rangle)_i$ outputs a parameter close to θ/α therefore the attacker needs to choose α such that for every context $x \in \mathcal{D}$, $\langle x, \theta/\alpha \rangle \leq \max_{a^\dagger \in A^\dagger} \langle x, \theta_{a^\dagger}, x \rangle$. In other words, the attacker performs a dilation of the incoming context every time algorithm $\mathfrak{A}$ does not pull an arm in $A^\dagger$. The fact that the decision rule used by LinUCB is invariant by dilation guarantees that the attacker will not inadvertently lower the perceived rewards for arms in $A^\dagger$. Because the rewards are assumed to be linear, presenting a large context αx and receiving the reward associated with the normal context x will skew the estimated rewards of LinUCB. The attack protocol is summarized in Fig. 25.

In order to compute the parameter α used in the attack, we make the following assumption concerning the performance of the arms in the target set:

Assumption 14. *For all $x \in \mathcal{D}$, there exists $a^\dagger \in A^\dagger$, such that $0 < \nu \leq \langle x, \theta_{a^\dagger} \rangle$ and ν is known to the attacker.*

Knowing ν. For advertising and recommendation systems, knowing ν is not problematic. Indeed in those cases, the reward is the probability of impression of the ad ($r \in [0, 1]$). The attacker has the freedom to choose one of multiple target arms with strictly positive click probability in every context. This freedom is an important aspect for the attacker since it allows the attacker to cherry pick the target ad(s). In particular, the attacker can estimate ν based on data from previous campaigns (only for the target ad). For instance, a company could have run many ad campaigns for one of their products and try to get the defender's system to advertise it.

An issue is that the norm of the attacked context can be greater that the upper bound L of Assumption 13. To prevent this issue, we choose a context-dependent multiplicative constant $\alpha(x) = \min\{2/\nu, L/\|x\|_2\}$ which amounts to clip the norm of the attacked context to L. In Sec. 4.1.5, we show that this attack is effective for different size of target arms sets. We also show that in the case of contexts such that $\|x\|_2 \leq \nu L/2$ that the cost of attacks is logarithmic in the horizon T.

Algorithm 24: Contextual ACE algorithm

for *time* $t = 1, 2, ..., T$ **do**
- Alg. $\mathfrak{A}$ chooses arm a_t based on context x_t
- Environment generates reward: $r_{t,a_t} = \langle \theta_{a_t}, x_t \rangle + \eta_t$ with $\eta^t_{a_t}$ conditionally σ^2-subgaussian
- Attacker observes reward r_{t,a_t} and feeds the perturbed reward $\widetilde{r}^1_{t,a_t}$ (or $\widetilde{r}^2_{t,a_t}$) to $\mathfrak{A}$

Algorithm 25: ConicAttack algorithm.

Input: attack parameter: α

for *time* $t = 1, 2, ..., T$ **do**
- Attacker observes the context x_t, computes potential arm a'_t and sets $\widetilde{x}_t = x_t + (\alpha(x_t) - 1) x_t \mathbb{1}_{\{a'_t \notin A^\dagger\}}$
- Alg. $\mathfrak{A}$ chooses arm a_t based on context $\widetilde{x}_t$
- Environment generates reward: $r_{t,a_t} = \langle \theta_{a_t}, x_t \rangle + \eta_t$ with η_t conditionally σ^2-subgaussian
- Alg. $\mathfrak{A}$ observes reward r_{t,a_t}

Proposition 2. *Using the attack described in Alg. 25 and assuming that $\|x\|_2 \leq \nu L/2$ for all contexts $x \in \mathcal{D}$, for any $\delta \in (0, 1/K]$, with probability at least $1 - K\delta$, the number of times LinUCB does not pull an arm in $A^\dagger$ before time T is at most $\sum_{j \notin A^\dagger} N_j(T) \leq 32K^2 \left(\frac{\lambda}{\alpha^2} + \sigma^2 d \log\left(\frac{\lambda d + TL^2\alpha^2}{d\lambda\delta} \right) \right)^3$ with $N_j(T)$ the number of times arm j has been pulled during the first T steps, The total cost for the attacker is bounded by: $\sum_{t=1}^T c_t \leq \frac{64K^2}{\nu} \left(\frac{\lambda}{\alpha^2} + \sigma^2 d \log\left(\frac{\lambda d + TL^2\alpha^2}{d\lambda\delta} \right) \right)^3$ with $\alpha = 2/\nu$.*

The proof of Proposition 2 (see App. 4.A.1.2) assumes that the attacker can attack at any time step, and that they can know in advance which arm will be pulled by Alg. $\mathfrak{A}$ in a given context. Thus it is not applicable to random exploration algorithms like LinTS Agrawal and Goyal (2012) and ε-greedy. We also observed empirically that thowe two randomized algorithms are more robust to attacks (see Sec. 4.1.5) than LinUCB.

Norm Clipping. Clipping the norm of the attacked contexts is not beneficial for the attacker. Indeed, this means that an attacked context was violating the assumption (used by the bandit algorithm) that contexts are bounded by L. The attack could then be easily detectable and may succeed only because it is breaking an underlying assumption used by the bandit algorithm. Prop. 2 provides a theoretical grounding for the proposed attack when contexts are bounded by $\nu L/2$ and not only L. Although, we can not prove a bound on the cumulative cost of attacks in general, we show in Sec. 4.1.5 that attacks are still successful for multiple datasets where contexts are not bounded by $\nu L/2$.

4.1.4 Offline attacks on a Single Context

Previous subsection s focused on the man-in-the-middle (MITM) attack either on reward or context. The MITM attack allows the attacker to arbitrarily change the information observed by the recommender system at each round. This attack may be hardly feasible in practice, since the exchange channels are generally protected by authentication and cryptographic systems. In this subsection , we consider the scenario where the attacker has control over a single user u. As an example, consider the case where the device of the user is infected by a malware (e.g., Trojan horse), giving full control of the system to the malicious agent. The attacker can thus modify the context of the specific user (e.g., by altering the cookies) that is perceived by the recommender system. We believe that changes to the context (e.g., cookies) are more subtle and less easily detectable than changes to the reward (e.g., click). Moreover, if the reward is a purchase, it cannot be altered easily by taking control of the user's device. Clearly, the impact of the attacker on the overall performance of the recommender system depends on the frequency of the specific user, that is out of the attacker's control. It may be thus difficult to obtain guarantees on the cumulative regret of algorithm $\mathfrak{A}$. For this reason, we mainly focus on the study of the feasibility of the attack.

The attacker targets a specific user (i.e., the infected user) associated to a context $x^\dagger$. Similarly to Sec. 4.1.3, the objective of the attacker is to find the minimal change to the context presented to the recommender system $\mathfrak{A}$ such that $\mathfrak{A}$ selects an arm in $A^\dagger$. $\mathfrak{A}$ observes a modified context $\widetilde{x}$ instead of $x^\dagger$. After selecting an arm a_t, $\mathfrak{A}$ observes the true noisy reward $r_{t,a_t} = \langle \theta_{a_t}, x^\dagger \rangle + \eta^t_{a_t}$. We still study a white-box setting: the attacker can access all the parameters of $\mathfrak{A}$.

In this subsection , we show under which condition it is possible for an attacker to fool both an optimistic and posterior sampling algorithm.

4.1.4.1 Optimistic Algorithm: LinUCB

We consider the LinUCB algorithm which chooses the arm to pull by maximizing an upper-confidence bound on the expected reward. For each arm a and context x, the UCB value is given by $\max_{\theta \in \mathcal{C}_{t,a}} \langle x, \theta \rangle = \langle x, \hat{\theta}^t_a \rangle + \beta_{t,a} \|x\|_{\widetilde{V}^{-1}_{t,a}}$ (see Sec. 4.1.1).

The objective of the attacker is to force LinUCB to pull an arm in $A^\dagger$ once presented with context $x^\dagger$. This means to find a perturbation of context $x^\dagger$ that makes any arm in $A^\dagger$ the most optimistic arm. Clearly, we would like to keep the perturbation as small as possible to reduce the cost for the attacker and the probability of being detected. Formally, the attacker needs to solve the following *non-convex* optimization problem:

$$\min_{y \in \mathbb{R}^d} \|y\|_2 \qquad \text{s.t} \qquad \max_{a \notin A^\dagger} \max_{\theta \in \widetilde{\mathcal{C}}_{t,a}} \langle x^\dagger + y, \theta \rangle + \xi \leq \max_{a^\dagger \in A^\dagger} \max_{\theta \in \widetilde{\mathcal{C}}_{t,a^\dagger}} \langle x^\dagger + y, \theta \rangle \tag{4.4}$$

where $\xi > 0$ is a parameter of the attacker and $\widetilde{\mathcal{C}}_{t,a} := \left\{ \theta \mid \|\theta - \hat{\theta}^t_a\|_{\widetilde{V}_{t,a}} \leq \beta_{t,a} \right\}$ is the confidence set constructed by LinUCB. We use the notation $\widetilde{\mathcal{C}}, \widetilde{V}$ to stress the fact that LinUCB observes only the modified context. In contrast to Sec. 4.1.2 and 4.1.3, the attacker may not be able to force the algorithm to pull any of the target arms in $A^\dagger$. In other words, Problem 4.4 may not be feasible. However, we are able to characterize the feasibility of (4.4).

Theorem 11. *Problem* (4.4) *is feasible at time* t iff.

$$\exists \theta \in \cup_{a^\dagger \in A^\dagger} \widetilde{\mathcal{C}}_{t,a^\dagger}, \ \theta \notin \mathit{Conv}\Big(\cup_{a \notin A^\dagger} \widetilde{\mathcal{C}}_{t,a} \Big) \tag{4.5}$$

The condition given by Theorem 11 says that this attack can be done when there exists a vector x for which an arm in $A^\dagger$ is assumed to be optimal according to LinUCB. The condition mainly stems from the fact that optimizing a linear product on a convex compact set will reach its maximum on the edge of this set. In our case this set is the convex hull of the confidence ellipsoids of LinUCB. Although it is possible to use an optimization algorithm for this class of non-convex problems—e.g., DC programming Tuy (1995)—they are still slow compared to convex algorithms. Therefore, we present a simple convex relaxation of the previous problem for a single target arm $a^\dagger \in A^\dagger$ that still enjoys some empirical performance compared to Problem (4.4). The final attack can then be computed as the minimum of the attacks obtained for each $a^\dagger \in A^\dagger$. The relaxed problem is the following for each $a^\dagger \in A^\dagger$:

$$\min_{y \in \mathbb{R}^d} \|y\|_2 \qquad \text{s.t} \qquad \max_{a \neq a^\dagger, a \notin A^\dagger} \max_{\theta \in \mathcal{C}_{t,a}} \langle x^\dagger + y, \theta - \hat{\theta}^t_{a^\dagger} \rangle \leq -\xi \tag{4.6}$$

Since the RHS of the constraint in Problem (4.4) can be written as $\max_{\theta \in \mathcal{C}_{t,a^\dagger}} \langle \theta, x^\dagger + y \rangle$ for any y, the relaxation here consists in using $\langle \theta, x^\dagger + y \rangle$ as a lower-bound to this maximum for any $\theta \in \mathcal{C}_{t,a^\dagger}$.

For the relaxed Problem (4.6), the same type of reasoning as for Problem (4.4) gives that Problem (4.6) is feasible if and only if $\hat{\theta}_{a^\dagger}(t) \notin \mathrm{Conv}\left(\bigcup_{a \neq a^\dagger, a \notin A^\dagger} \mathcal{C}_{t,a} \right)$.

If Condition (4.5) is not met, no arm $a^\dagger \in A^\dagger$ can be pulled by LinUCB. Indeed, the proof of Theorem 11 shows that the upper-confidence of every arm in $A^\dagger$ is always dominated by another arm for any context. In other words, if any arm in $A^\dagger$ is optimal for some contexts then the condition is satisfied a linear number of times for LinUCB (for formal proof of this fact see App. 4.A.1.4).

4.1.4.2 Random Exploration Algorithm: LinTS

The previous subsubsection focused on LinUCB, however we can obtain similar guarantees for algorithms with random exploration such as LinTS. In this case, it is not possible to guarantee that a specific arm will be pulled for a given context because of the randomness in the arm selection process. The objective is to guarantee that an arm from $A^\dagger$ is pulled with probability at least $1-\delta$. Similarly to the previous subsubsection , the problem of the attacker can be written as:

$$\min_{y\in\mathbb{R}^d} \|y\| \qquad \text{s.t} \qquad \mathbb{P}\left(\exists a^\dagger \in A^\dagger,\ \forall a \notin A^\dagger,\ \langle x^\dagger + y, \widetilde{\theta}_a - \widetilde{\theta}_{a^\dagger}\rangle \leq -\xi\right) \geq 1-\delta \tag{4.7}$$

where the $\widetilde{\theta}_a$ for different arms a are independently drawn from a normal distribution with mean $\hat{\theta}_a(t)$ and covariance matrix $\upsilon^2 \bar{V}_a^{-1}(t)$ with $\upsilon = \sigma\sqrt{9d\ln(T/\delta)}$. Solving this problem is not easy and in general not possible, even for a single arm. For a given x and arm a, the random variable $\langle x, \widetilde{\theta}_a\rangle$ is normally distributed with mean $\mu_a(x) := \langle \hat{\theta}_a(t), x\rangle$ and variance $\sigma_a^2(x) := \nu^2 ||x||^2_{\bar{V}_a^{-1}(t)}$. We can then write $\langle x, \widetilde{\theta}_a\rangle = \mu_a(x) + \sigma_a(x) Z_a$ with $(Z_a)_a \sim \mathcal{N}(0, I_K)$. For the sake of clarity, we drop the variable x when writing $\mu_a(x)$ and $\sigma_a(x)$.

Let's imagine (just for this paragraph) that $A^\dagger = \{a^\dagger\}$, then the constraint in Problem (4.7) becomes:

$$\left[1 - \mathbb{E}_{Z_{a^\dagger}}\left(\Pi_{a\notin A^\dagger}\Phi\left(\frac{\sigma_{a^\dagger} Z_{a^\dagger} + \mu_{a^\dagger} - \mu_a}{\sigma_a}\right)\right)\right] \leq \delta$$

$$\mathbb{E}_{Z_{a^\dagger}}\left(\Pi_{a\neq a^\dagger}\Phi\left(\frac{\sigma_{a^\dagger} Z_{a^\dagger} + \mu_{a^\dagger} - \mu_a}{\sigma_a}\right)\right) \geq 1-\delta$$

where Φ is the cumulative distribution function of a normally distributed Gaussian random variable. Unfortunately, computing exactly this expectation is an open problem.

In the more general case where $|A^\dagger| \geq 1$, rewriting the constraints of Problem (4.7) is not possible. Following the idea of Liu and Shroff (2019), for every single target arm $a^\dagger \in A^\dagger$, a possible relaxation of the constraint in Problem (4.7) is, to ensure that there exists an arm $a^\dagger \in A^\dagger$ such that for every arm $a \notin A^\dagger$, $1 - \Phi\left((\mu_{a^\dagger} - \mu_a - \xi)/(\sqrt{\sigma_a^2 + \sigma_{a^\dagger}^2})\right) \leq \frac{\delta}{K - |A^\dagger|}$, where $|A^\dagger|$ is the cardinal of $A^\dagger$. Thus the relaxed version of the attack on LinTS for a single arm $a^\dagger$ is:

$$\min_{y\in\mathbb{R}^d} \|y\| \quad \text{s.t.} \quad \forall a \notin A^\dagger, \langle x^\dagger + y, \hat{\theta}_{a^\dagger} - \hat{\theta}_a\rangle - \xi \geq \nu\Phi^{-1}\left(1 - \tfrac{\delta}{K-|A^\dagger|}\right)\left\|x^\dagger + y\right\|_{\bar{V}_a^{-1} + \bar{V}_{a^\dagger}^{-1}} \tag{4.8}$$

Problem (4.8) is similar to Problem (4.6) as the constraint is also a Second Order Cone Program but with different parameters (see App. 4.A.3). As in subsection 4.1.4.1, we compute the final attack as the minimum of the attacks computed for each arm in $A^\dagger$.

4.1.5 Experiments

In this subsection , we conduct experiments on the attacks on contextual bandit problems with simulated data and two real-word datasets: MovieLens25M Harper and Konstan (2015) and Jester Goldberg et al. (2001). The synthetic dataset and the data preprocessing step are presented in App. 4.A.2.1.

4.1.5.1 Attacks on Rewards

We study the impact of the reward attack for 4 contextual algorithms: LinUCB, LinTS, ε-greedy and Exp4. As parameters, we use $L = 1$ for the maximal norm of the contexts, $\delta = 0.01$, $\upsilon = \sigma\sqrt{d\ln(t/\delta))/2}$, $\varepsilon_t = 1/\sqrt{t}$ at each time step t and $\lambda = 0.1$. We choose only a *unique target arm* $a^\dagger$. For Exp4, we use $N = 10$ experts with $N-2$ experts returning a random arm at each time, one expert choosing arm $a^\dagger$ every time and one expert returning the optimal arm for every context. With this set of experts the regret of bandits with expert advice is the same as in the contextual case. To test the performance of each algorithm, we generate 40 random contextual bandit problems and run each algorithm for $T = 10^6$ steps on each. We report the average cost and regret for each of the 40 problems. Figure 4.1.1 (Top) shows the attacked algorithms using the attacked reward $\widetilde{r}^1$ (reported as "stationary CACE") and the rewards $\widetilde{r}^2$ (reported as CACE).

These experiments show that, even though the reward process is non-stationary, usual stochastic algorithms like LinUCB can still adapt to it and pull the optimal arm for this reward process (which is arm $a^\dagger$). The true regret of the attacked algorithms is linear as $a^\dagger$ is not optimal for all contexts. In the synthetic case, for the algorithms attacked with the rewards $\widetilde{r}^2$, over 1M iterations and $\gamma = 0.22$, the target arm is drawn more than 99.4% of the time on average for every algorithm and more than 97.8% of the time for the stationary attack $\widetilde{r}^1$ (see Table 4.A.1 in App. 4.A.2.2). The dataset-based environments (see Figure 4.1.1 (Left)) exhibit the same behavior: the target arm is pulled more than 94.0% of the time on average for all our attacks on Jester and MovieLens and more than 77.0% of the time in the worst case (for LinTS attacked with the stationary rewards) (see Table 4.A.1).

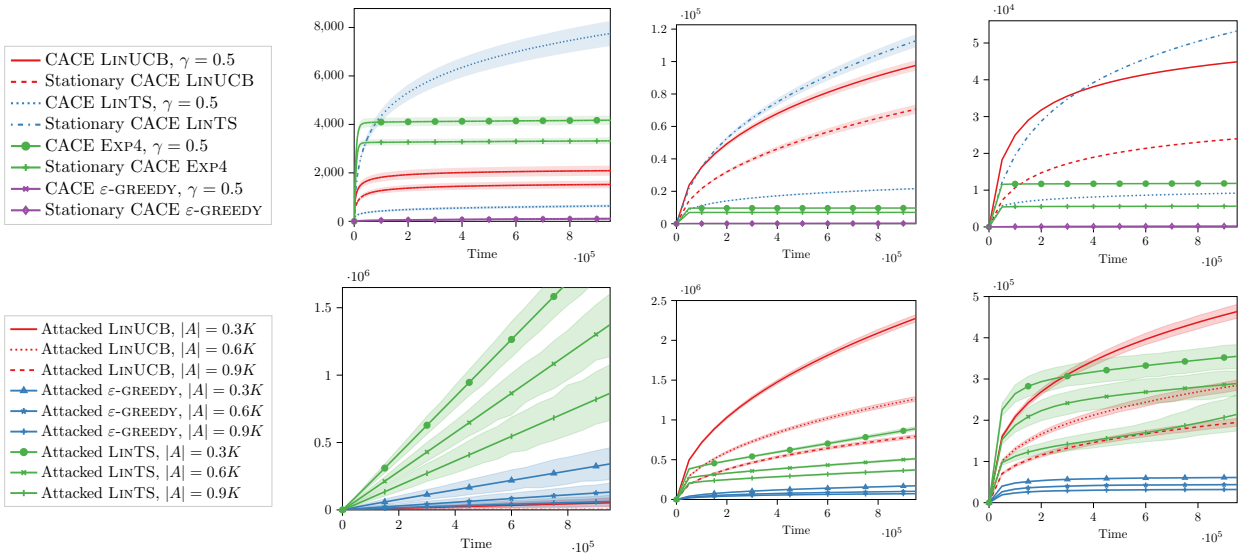


Figure 4.1.1: Total cost of attacks on rewards for the synthetic (Left, $\gamma = 0.22$), Jester (Center, $\gamma = 0.5$) and MovieLens (Right, $\gamma = 0.5$) environments. Bottom, total cost of ContextualConic attacks on the synthetic (Left), Jester (Center) and MovieLens (Right) environments.

4.1.5.2 Attacks on Contexts

We now illustrate the effectiveness of the attack in Alg. 25. We study the behavior of attacked LinUCB, LinTS, ε-greedy with different size of target arms set ($|A^\dagger|/K \in \{0.3, 0.6, 0.9\}$ with K the total number of arms). We test the performance of LinUCB with the same parameters as in the previous experiments. Yet since the variance is much smaller in this case, we generate a random problem and run 20 simulations for each algorithm. The target arms are chosen randomly and we use the exact lower-bound on the reward of those arms to compute ν.

Table 4.1.1: Percentage of iterations for which the algorithm pulled an arm in the target set $A^\dagger$ (with a target set size of $0.3K$ arms) (**Left**) Online attacks using ContextualConic (CC) algorithm. Percentages are averaged over 20 runs of 1M iterations. (**Right**) Offline attacks with exact (Full) and Relaxed optimization problem. Percentages are averaged over 40 runs of 1M iterations.

	Synthetic	Jester	Movilens
LinUCB	28.91%	26.59%	31.13%
CC LinUCB	98.55%	98.36%	99.61%
ε-greedy	25.7%	25.85%	31.78%
CC ε-greedy	89.71%	99.85%	99.92%
LinTS	27.2%	26.10%	33.24%
CC LinTS	30.93%	97.26%	98.82%

	Synthetic	Jester	MovieLens
LinUCB	0.07%	0.01%	0.39%
LinUCB Relaxed	13.76%	97.81%	4.09%
LinUCB Full	88.30%	99.98%	99.99%
ε-greedy	0.01%	0.00%	0.03%
ε-greedy Full	99.98%	99.95%	99.97%
LinTS	0.02%	0.01%	0.05%
LinTS Relaxed	18.21%	80.48%	5.56%

Table 4.1.1 (Left) shows the percentage of times an arm in $A^\dagger$, for $|A^\dagger| = 0.3K$, has been selected by the attacked algorithm. We see that, as expected, CC LinUCB reaches a ratio of almost 1, meaning the target arms are indeed pulled a linear number of times. A more surprising result (at least not covered by the theory) is that ε-greedy exhibits the same behavior. Similarly to LinTS, ε-greedy exhibits some randomness in the action selection process. It can cause an arm $a^\dagger \in A^\dagger$ to be chosen when the context is attacked and interfere with the principle of the attack. We suspect that is what happens for LinTS. Fig. 4.1.1 (Bottom) shows the total cost of the attacks for the attacked algorithms . Despite the fact that the estimate of $\theta_{a^\dagger}$ can be polluted by attacked samples, it seems that LinTS can still pick up $a^\dagger$ as being optimal for some dataset like MovieLens and Jester but not on the simulated dataset.

4.1.5.3 Offline attacks on a Single Context

We now move to the setting described in Sec. 4.1.4 and test the same algorithms as in Sec. 4.1.5.2. We run 40 simulations for each algorithm and each attack type. The target context $x^\dagger$ is chosen randomly and the target arm as the arm minimizing the expected reward for $x^\dagger$. The attacker is only able to modify the incoming context for the target context (which corresponds to

the context of one user) and the incoming contexts are sampled uniformly from the set of all possible contexts (of size 100). Table 4.1.1 (Right) shows the percentage of success for each attack. We observe that the non-relaxed attacks on ε-GREEDY and LINUCB work well across all datasets. However, the relaxed attack for LINUCB and LINTS are not as successful, on the synthetic dataset and MovieLens25M. The Jester dataset seems to be particularly suited to this type of attacks because the true feature vectors are well separated from the convex hull formed by the feature vectors of the other arms: only 5% of Jester's feature vectors are within the convex hull of the others versus 8% for MovieLens and 20% for the synthetic dataset. As expected, the cost of the attacks is linear on all the datasets (see Figure 4.A.4 in App. 4.A.2.4). The cost is also lower for the non-relaxed than for the relaxed version of the attack on LINUCB. Unsurprisingly, the cost of the attacks on LINTS is the highest due to the need to guarantee that $a^\dagger$ will be chosen with high probability (95% in our experiments).

4.1.6 Concluding Remarks and Extensions

We presented several settings for online attacks on contextual bandits. We showed that an attacker can force any contextual bandit algorithm to almost always pull an arbitrary target arm $a^\dagger$ with only sublinear modifications of the rewards. When the attacker can only modify the contexts, we prove that LINUCB can still be attacked and made to almost always pull an arm in $A^\dagger$ by adding sublinear perturbations to the contexts. When the attacker can only attack a single context, we derive a feasibility condition for the attacks and we introduce a method to compute some attacks of small instantaneous cost for LINUCB, ε-GREEDY and LINTS. To the best of our knowledge, this paper was the first to describe effective attacks on the contexts of contextual bandit algorithms. Our numerical experiments, conducted on both synthetic and real-world data, validate our results and show that the attacks on all contexts are actually effective on several algorithms and with more permissible settings.

In the next section, we focus on building an end-to-end encrypted linear contextual algorithm. That is to say all the contexts and rewards are encrypted using homomorphic encryption. Using this encryption method we design a bandit algorithm that does not need to observe the true contexts and rewards of the users but only encrypted version of it. Offering some protection against adversarial attacks especially on rewards but also and maybe more importantly prevent any privacy issue when the users do not trust the bandit algorithm.

4.2 Encryption in Linear Contextual Bandit

Contextual bandits have become a key part of several applications such as marketing, healthcare and finance; as they can be used to provide personalized e.g., adaptive service (Bastani and Bayati, 2020; Sawant et al., 2018). In such application, algorithms receives as input users' features, i.e. the "contexts", to tailor their recommendations. Those features may disclose sensitive information, as personal (e.g., age, gender, etc.) or geo-localized features are commonly used in recommendation systems. Privacy awareness has increased over years and users are less willing to disclose information and are more and more concerned about how their personal data is used (Das et al., 2021). For example, a user may be willing to receive financial investment suggestion but not to share information related to income, deposits, properties owned and other assets. However, without observing this important information about a user, a service provider may not be able to provide meaningful investment guidance to the user. This example extends to many other applications. For instance, suppose an user is looking for a restaurant nearby, if the provider has no access to even a coarse geo-location, it would not be able to provide meaningful suggestions to the user. An effective approach to address these concerns is to resort to end-to-end encryption to guarantee that data is readable only by the users (Kattadige et al., 2021). In this scenario, the investment company or the service provider observes only an encrypted version of user's information and have no ability to decrypt it. While this guarantee high level of privacy, it is unclear whether the problem remains learnable and how to design effective online learning algorithms in this secure scenarios.

In this paper, we introduce - and analyze - the setting of encrypted contextual bandit to model the mentioned scenarios. At each round, a bandit algorithm observes encrypted features (including e.g., geo-location, food preferences, visited restaurants), chooses an action (e.g., a restaurant) and observes an encrypted reward (e.g., user's click), that is used to improve the quality of recommendations. While it is possible to obtain end-to-end encryption –i.e., the bandit algorithm only observes encrypted information that is not able to decrypt– using standard encryption methods (e.g., AES, RSA, TripleDES), the provider may no longer be able to provide a meaningful service since may not be able to extract meaningful information from encrypted features. We thus address the following question:

Is it possible to learn with encrypted contexts and rewards? And what is the associated computational and learning cost?

Homomorphic Encryption (Halevi, 2017, HE) is a powerful encryption method that allows to carry out computation of encrypted numbers. While this is a very powerful idea, only a limited number of operations can be performed, notably only addition and/or multiplication. While HE has been largely investigated in supervised learning (Badawi et al., 2020; Graham, 2015), little is known about online learning. In this paper we aim to look into this direction. We approach the aforementioned question via HE and from a theoretical point-of-view. We consider the case of linear rewards and investigate the design of a "secure" algorithm able to achieve sub-linear regret in this setting. There are several challenges in the design of bandit algorithms that

makes the application of HE techniques not easy. First, it is not obvious that all the operations required by a bandit algorithm (notably optimism) can be carried out only through additions and multiplications. Second, errors or approximations introduced by the HE framework to handle encrypted data may compound and prevent to achieve provably good performance. Finally, a careful algorithmic design is necessary to limit the total number of HE operations, which are computationally demanding.

Contributions. Our main contributions can be summarized as follows: **1)** We introduce and formalize the problem of secure contextual bandit with homomorphic encryption. **2)** We provide the first bandit algorithm able to learn over encrypted data in contextual linear bandits, a standard framework that allows us to describe and address all the challenges in leveraging HE in online learning. Leveraging optimism (e.g., Abbasi-Yadkori et al., 2011) and HE, we introduce HELBA which balances security, approximation error due to HE and computational cost to achieve a $\widetilde{O}(\sqrt{T})$ regret bound. This shows that i) it is possible to learn *online* with encrypted information; ii) preserving users' data security has negligible impact on the learning process. This is a large improvement w.r.t. ε-LDP which has milder security guarantees and where the best known bound is $\widetilde{O}(T^{3/4}/\varepsilon)$. **3)** We discuss practical limitations of HE and ways of improving the efficiency of the proposed algorithm, mainly how the implementation of some procedures can speed up computations and allow to scale dimension of contexts. We report preliminary numerical simulations confirming the theoretical results.

Related work. To prevent information leakage, the bandit literature has mainly focused on Differential Privacy (DP) (e.g., Shariff and Sheffet, 2018; Tossou and Dimitrakakis, 2016). While standard (ϵ, δ)-DP enforces statistical diversity of the output of an algorithm, it does not provide guarantees on the security of user data that can be accessed directly by the algorithm. A stronger privacy notion, called local DP, requires data being privatized before being accessed by the algorithm. While it may be conceptually similar to encryption, i) it does not provide the same security guarantee as encryption (having access to a large set of samples may allow some partial denoising Cheu et al. (2021)); and ii) it has a large impact on the regret of the algorithm. For example, Zheng et al. (2020) recently analyzed ε-LDP in contextual linear bandit and derived an algorithm with $\widetilde{O}(T^{3/4}/\varepsilon)$ regret bound to be compared with a $\widetilde{O}(\sqrt{T})$ regret of non-private algorithms. Homomorphic Encryption (e.g. Halevi, 2017) has only been merely used to encrypt rewards in bandit problems (Ciucanu et al., 2019), but in some inherently simpler setting than the setting considered here (see App. 4.B.2).

4.2.1 Homomorphic Encryption

Homomorphic Encryption (Halevi, 2017) is a *probabilistic encryption method* that enables an untrusted party to perform some computations (addition and/or multiplication) on encrypted data. Formally, given two original messages m_1 and $m_2 \in \mathbb{R}$, the addition (resp. multiplication) of their encrypted versions (called ciphertexts) is equal to the encryption of their sum $m_1 + m_2$ (resp. $m_1 \times m_2$), hence the name "homomorphic".[1] We consider a generic homomorphic schemes that generate a public key **pk** (distributed widely and used to encrypt messages), and private keys **sk** (used for decryption of encrypted messages). This private key is, contrary to the public key, obviously assumed to be kept private.

More precisely, we shall consider *Leveled Fully Homomorphic* encryption (LFHE) schemes for real numbers. This type of schemes supports both additions and multiplications but only for a fixed and finite number of operations, referred to as the *depth*. This limitation is a consequence of HE's probabilistic approach. Although noisy encryption allows to achieve high security, after a certain number of operations the data is drown in the noise (e.g., Albrecht et al., 2015), resulting in an indecipherable ciphertext (the encrypted message). In most LFHE schemes, the depth is the maximum number of operations possible before losing the ability to decrypt the message. Often multiplications have a significantly higher noise growth than addition and the depth refers to the maximum number of multiplication between ciphertexts possible. The security of a LFHE schemes is defined by $\kappa \in \mathbb{N}$, usually $\kappa \in \{128, 192, 256\}$. A κ-bit level of security means that an attacker has to perform roughly 2^κ operations to break the encryption scheme, i.e., to decrypt a ciphertext without the secret key.

Formally, an LFHE scheme is defined by:

- *A key generator function KeyGen*(N, D, κ): takes as input the maximum depth D (e.g., max. number of multiplications), a security parameter κ and the degree N of polynomials used as ciphertexts (App. 4.B.3.1). It outputs a secret key **sk** and a public key **pk** .
- *An encoding function* $\mathit{Enc}_{\mathbf{pk}}(m)$: encrypts the message $m \in \mathbb{R}^d$ with the public key **pk**. The output is a ciphertext **ct**, a representation of m in the space of complex polynomials of degree N.
- *A decoding function* $\mathit{Dec}_{\mathbf{sk}}(\mathbf{ct})$: decrypts the ciphertext **ct** of $m \in \mathbb{R}^d$ using the secret key **sk** and outputs message m.
- *An additive operator* $\mathit{Add}(\mathbf{ct}_1, \mathbf{ct}_2)$: for ciphertexts $\mathbf{ct}_1$ and $\mathbf{ct}_2$ of messages m_1 and m_2, it outputs ciphertext $\mathbf{ct}_{add}$ of $m_1 + m_2$: $\mathit{Dec}_{\mathbf{sk}}\Big(\mathit{Add}\big(\mathit{Enc}_{\mathbf{pk}}(m_1), \mathit{Enc}_{\mathbf{pk}}(m_2)\big)\Big) = m_1 + m_2$.
- *A multiplicative operator* $\mathit{Mult}(\mathbf{ct}_1, \mathbf{ct}_2)$: similar to *Add* but for ciphertexts $\mathbf{ct}_1$ and $\mathbf{ct}_2$ of messages m_1 and m_2 and output ciphertext $\mathbf{ct}_{mult}$ of $m_1 \cdot m_2$.

To avoid to complicate the notation we will use classical symbols to denote addition and multiplication between ciphertexts. Choosing D as small as possible is essential, as it is the major bottleneck for performance, in particular at the keys generation step. This cost comes from the fact that the dimension of a ciphertext N needs to grow with D for a given security level κ:

[1] Most schemes also support Single Instruction Multiple Data (SIMD), i.e., the same operation on multiple data points in parallel.

Algorithm 26: Encrypted Contextual Bandit (Server-Side)

Input: Agent: $\mathfrak{A}$, public key: **pk**, horizon: T
for $t = 1, \dots, T$ **do**
  Agent $\mathfrak{A}$ observes encrypted context $(x_{t,a})_{a\in[K]} = (\mathit{Enc}_{\mathbf{pk}}(s_{t,a}))_{a\in[K]}$
  Agent $\mathfrak{A}$ computes the next action as a function of the encrypted history and $(x_{t,a})_{a\in[K]}$ and outputs an encrypted action $u_t = \mathit{Enc}_{\mathbf{pk}}(a_t)$
  Agent $\mathfrak{A}$ observes encrypted reward $y_t = \mathit{Enc}_{\mathbf{pk}}(r_t)$

Algorithm 27: Encrypted Contextual Bandit (User-Side)

Input: Public key: **pk**, Secret key: **sk**
for $t = 1, \dots, T$ **do**
  User t observes features $(s_{t,a})_{a\leq K}$ and sends $(x_{t,a})_{a\in[K]} = (\mathit{Enc}_{\mathbf{pk}}(s_{t,a}))_{a\in[K]}$ to the server
  User t receives encrypted action u_t
  User t decrypts action $a_t = \mathit{Dec}_{\mathbf{sk}}(u_t)$
  User t observes reward $r_t = r(s_{t,a_t}) + \eta_t$ and sends $\mathit{Enc}_{\mathbf{pk}}(r_t)$ to the server

namely $N \geq \Omega(\kappa D)$ (refer to App. 4.B.3.1 for more details). In this paper, we choose to use the CKKS scheme (Cheon et al., 2017) because it supports operations on real numbers.

Other HE schemes. Most HE schemes (ElGamal, 1985; Paillier, 1999; Rivest et al., 1978) are *Partially Homomorphic* and only support either additions or multiplications, but not both. Other schemes that support any number of operations are called *Fully Homomorphic* encryption (FHE) schemes. Most LFHE schemes can be turned into FHE schemes thanks to the bootstrapping technique introduced by Gentry and Boneh (2009). However, the computational cost is extremely high. It is thus important to optimize the design of the algorithm to minimize its multiplicative depth and (possibly) avoid bootstrapping (Acar et al., 2018; Ducas and Micciancio, 2015; Zhao and Wang, 2018).

4.2.2 Contextual Bandit And Encryption

Let's consider the contextual bandit problem with arm dependent features described in Section 1.2.2 with $K \in \mathbb{N}_+$ arms and horizon $T \in \mathbb{N}_+$ (e.g., Lattimore and Szepesvári, 2020). At each time $t \in [T] := \{1, \dots, T\}$, a learner first observes a set of features $(s_{t,a})_{a\in[K]} \subset \mathbb{R}^{d}$[2], selects an action $a_t \in [K]$ and finally observes a reward $r_t = r(s_{t,a_t}) + \eta_t$ where η_t is a conditionally independent zero-mean noise. We do not assume anything on the distribution of the features $(s_{t,a})_a$. The performance of the learner $\mathfrak{A}$ over T steps is measured by the regret, that measures the cumulative difference between playing the optimal action and the action selected by the algorithm. Formally, let $a_t^\star = \arg\max_{a\in[K]} r(s_{t,a})$ be the optimal action at step t, then the pseudo-regret is defined as:

$$R_T = \sum_{t=1}^{T} r(s_{t,a_t^\star}) - r(s_{t,a_t}). \tag{4.9}$$

To protect privacy and avoid data tempering, we introduce end-to-end encryption to this protocol. Contexts and rewards are encrypted before being observed by the learner; we call this setting encrypted contextual bandit (Alg. 26). Formally, at time $t \in [T]$, the learner $\mathfrak{A}$ observes encrypted features $x_{t,a} = \mathit{Enc}_{\mathbf{pk}}(s_{t,a})$ for all actions $a \in \mathcal{A}$, and the encrypted reward $y_t = \mathit{Enc}_{\mathbf{pk}}(r_t)$ associated to the selected action a_t. The learner may know the public key **pk** but not the secure key **sk**. The learner is thus not able to decrypt messages and it *never* observes the true contexts and rewards. We further assume that both the agent $\mathfrak{A}$ and the users follow the honest-but-curious model, that is to say each parties follow their protocol honestly but try to learn as much as possible about the other parties private data. [3] As a consequence, the learner can only do computation on the encrypted information. As a result, all the internal statistics used by the bandit algorithm are now encrypted. On user's side (see Alg. 27), upon receiving an encrypted action $u_t = Enc_{\mathbf{pk}}(a_t)$ and decrypting it $a_t = Dec_{\mathbf{sk}}(u_t)$ using the secure key **sk**, the user generates a reward $r_t = r(s_{t,a_t}) + \eta_t$ and sends to the learner the associated ciphertext y_t. The learning algorithm is able to encrypt the action since the public key is publicly available. See App. 4.B.3 for additional details.

We focus on the well-known linear setting where rewards are linearly representable in the features. Formally, for any feature vector $s_{t,a}$, the reward is $r(s_{t,a}) = \langle s_{t,a}, \theta^\star \rangle$, where $\theta^\star \in \mathbb{R}^d$ is unknown. For the analysis, we rely on the following standard assumption:

Assumption 15. *There exists $S > 0$ such that $\|\theta^\star\|_2 \leq S$ and there exists $L \geq 1$ such that, for all time $t \in [T]$ and arm $a \in [K]$, $\|s_{t,a}\|_2 \leq L$ and $r_t = \langle s_{t,a}, \theta^\star \rangle + \eta_t \in [-1, 1]$ with η_t being σ-subGaussian for some $\sigma > 0$.*

[2] Contrary to the notation in Section 1.2.2, the unencrypted features are denoted by s_t as they are not observed by the agent. The encrypted features observed by the agent are still denoted by x_t

[3] A trusted third party can be used to generate a public and secret keys. Those keys are then sent to the users but **not** to the agent $\mathfrak{A}$ (see Sec. 4.2.5).

4.2.3 An Algortihm For Encrypted Linear Contextual Bandits

In the previous subsection , we have introduced a generic framework for contextual bandit with encrypted information. Here, we provide the first algorithm able to learn with encrypted observations.

Algorithm 28: Simplified HELBA

for $t = 1, \dots, T$ **do**
  if *Update (Step ❹)* **then**
    Step ❶: Estimate encrypted parameter using $\{x_{l,a_l}, y_l\}_{l\in[t-1]}$
  Observe encrypted contexts $(x_{t,a})_{a\in[K]} = (\mathit{Enc}_{\mathsf{pk}}(s_{t,a}))_{a\in[K]}$
  Step ❷: Compute encrypted indexes $(\rho_a(t))_{a\in[K]}$
  Step ❸: Compute $\arg\max_a\{\rho_a(t)\}$

In the non-secure protocol, algorithms based on the optimism-in-the-face-of-uncertainty (OFU) principle such as LinUCB (Chu et al., 2011) and OFUL (Abbasi-Yadkori et al., 2011) have been proved to achieve the regret bound $O\big(Sd\sqrt{T}\ln(TL)\big)$. Clearly, they will fail to be used as is in the secure protocol and need to be rethinked around the limitations of HE (mainly approximations in most operations). As mentioned in the introduction, there are many, both theoretical and practical, challenges to leverage HE in this setting. Indeed, **1)** computing an estimate of the parameter $\theta^\star$ from ridge regression is extremely difficult with HE as finding the inverse of a matrix is not directly feasible for a leveled scheme (Esperança et al., 2017). **2)** Similarly, computing the bonus for the optimistic action selection requires invoking operations that are not naturally available in HE hence incurring a large computational cost. Finally, **3)** computing the maximum element (or maximum index) of a list of encrypted values is non-trivial for the algorithm alone, as it cannot observe the values to compare. In this subsection , we will provide HE compatible operations addressing these three issues. Each step is highly non-trivial and correctly combining them is even more challenging due to error compounding. We believe the solution we provide for each individual step may be of independent interest.

Alg. 28 report a simplified version of our HE bandit algorithm. Informally, at each round t, our algorithm HELBA (*Homomorphically Encrypted Linear Bandits*) builds an HE estimate ω_t of the unknown $\theta^\star$ ($\omega^\star = \mathit{Enc}_{\mathsf{pk}}(\theta^\star)$) using the observed encrypted samples, compute HE optimistic indexes $(\rho_a(t))_a$ for each action and select the action maximizing the index. We stress that all the mentioned statistics (ω_t and $\rho_a(t)$) are encrypted values. Indeed, HELBA operates directly in the encrypted space, i.e., the space of *complex polynomials* of degree N. Let's analyze those three steps.

Step ❶: HE Friendly Ridge Regression

The first step is to build an estimate of the parameter $\theta^\star$. In the non-encrypted case, we can simply use $\theta_t = V_t^{-1}\sum_{l=1}^{t-1} s_{l,a_l} r_l$, where $V_t = \sum_{l=1}^{t-1} s_{l,a_l} s_{l,a_l}^T + \lambda I$. With encrypted values $(x_{l,a_l}, y_l)_{l\in[t-1]}$, it is possible to compute an encrypted matrix $\Lambda_t = \sum_{l=1}^{t-1} x_{l,a_l} x_{l,a_l}^T + \lambda \mathit{Enc}_{\mathrm{pk}}(I) = \mathit{Enc}_{\mathrm{pk}}(V_t)$ and vector $\sum_{l=1}^{t-1} x_{l,a_l} y_l$ as these operations (summing and multiplying) are HE compatible. The issue resides in the computation of Λ_t^{-1}. An approximate inversion scheme can be leveraged though.

Given a matrix $V \in \mathbb{R}^d$ with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_d > 0$ and $c \in \mathbb{R}$ such that for all $i \in [d]$, $\lambda_i \in \mathrm{Conv}\left(\{z \in \mathbb{R} \mid |z - c| \leq c\}, 2c\right)\backslash \{0, 2c\}$[4], we define the following sequence of matrices (Guo and Higham, 2006)

$$X_{k+1} = X_k(2I_d - M_k), \quad M_{k+1} = (2I_d - M_k)M_k, \tag{4.10}$$

initialized at $X_0 = \frac{1}{c} I_d$ and $M_0 = \frac{1}{c} V$. We can show that this sequence converges to V^{-1}.

Proposition 19. *If $V \in \mathbb{R}^{d\times d}$ is a symmetric positive definite matrix, $c \geq \mathit{Tr}(V)$ and for some precision level $\varepsilon > 0$, the iterate in (4.10) satisfies $\|X_k - V^{-1}\| \leq \varepsilon$ for any $k \geq k_1(\varepsilon)$ with $k_1(\varepsilon) = \frac{1}{\ln(2)} \ln\left(\frac{\ln(\lambda)+\ln(\varepsilon)}{\ln\left(1-\frac{\lambda}{c}\right)}\right)$, where $\lambda \leq \lambda_d$ is a lower bound to the minimal eigenvalue of V and $\|\cdot\|$ is the matrix spectral-norm.*

Since V_t is a regularized matrix, it holds that $\lambda_d \geq \lambda > 0$ and by setting $c = \lambda d + L^2 t$ we get that $c \geq \mathrm{Tr}(V_t) \geq \max_i\{\lambda_i\}$, for any step $t \in [T]$. Therefore, we can apply iterations (4.10) to $\Lambda_t = \mathit{Enc}_{\mathrm{pk}}(V_t)$ since are all HE compatible operations (additions and matrix multiplications). For $\varepsilon_t > 0$, iterations (4.10) gives a ε_t-approximation $A_t := X_{k_1(\varepsilon_t)}$ of V_t^{-1}, i.e., $\|\mathit{Dec}_{\mathrm{sk}}(A_t) - V_t^{-1}\| \leq \varepsilon_t$. As a consequence, an encrypted estimate of the unknown parameter $\theta^\star$ can be computed by mere simple matrix multiplications $\omega_t = A_t \sum_{l=1}^{t-1} x_{l,a_l} y_l$. Leveraging the concentration of the inverse matrix, the following error bound for the estimated parameter holds.

Corollary 3. *Setting $\varepsilon_t = \left(Lt^{3/2}\sqrt{L^2 t + \lambda}\right)^{-1}$ in Prop. 19, then $\|\mathit{Dec}_{\mathsf{sk}}(\omega_t) - \theta_t\|_{V_t} \leq t^{-1/2}$, $\forall t$.*

This result, along with the standard concentration for linear bandit (Abbasi-Yadkori et al., 2011, Thm. 2), implies that, at all time steps t, with probability at least $1 - \delta$:

$$\theta^\star \in \widetilde{\mathcal{C}}_t := \{\, \theta \in \mathbb{R}^d \mid \|\mathit{Dec}_{\mathsf{sk}}(\omega_t) - \theta\|_{V_t} \leq \widetilde{\beta}_t\}, \tag{4.11}$$

[4] $\mathrm{Conv}(E)$ is the convex hull of set E.

where $\|a\|_B = \sqrt{a^\top B a}$ and $\widetilde{\beta}_t = t^{-1/2} + S\sqrt{\lambda} + \sigma\sqrt{d\left(\ln\left(1 + L^2 t/\lambda\right) + \ln(\pi^2 t^2/(6\delta))\right)}$ is the inflated confidence interval due to the approximate inverse (see Prop. 23 in App. 4.B.4.4). Note that $\widetilde{\beta}_t$ is a plain scalar, not an encrypted value.

Step ❷: Computing The Optimistic Index

Once solved the encrypted ridge regression, the next step for HELBA is to compute an optimistic index $\rho_a(t)$ such that $r(s_{t,a}) \lessapprox \mathit{Dec}_{\mathsf{sk}}(\rho_a(t))$. For any feature vector $s_{t,a}$, by leveraging the confidence interval in (4.11), the optimistic (unencrypted) index is given by $\max_{\theta \in \widetilde{\mathcal{C}}_t} \langle \theta, s_{t,a}\rangle = \langle \mathit{Dec}_{\mathsf{sk}}(\omega_t), s_{t,a}\rangle + \widetilde{\beta}_t \|s_{t,a}\|_{V_t^{-1}}$. Leveraging Prop. 19, the definition of ε_t in Cor. 3 and $\|s_{t,a}\|_2 \leq L$, it holds that:

$$\begin{aligned}\forall s_{t,a},\ \|s_{t,a}\|^2_{V_t^{-1}} - \|s_{t,a}\|^2_{\mathit{Dec}_{\mathsf{sk}}(A_t)} &\leq L^2 \|V_t^{-1} - \mathit{Dec}_{\mathsf{sk}}(A_t)\| \\ &\leq L t^{-\frac{3}{2}} (\lambda + L^2 t)^{-1/2}\end{aligned}$$

which leads to $\max_{\theta \in \widetilde{\mathcal{C}}_t} \langle \theta, s_{t,a}\rangle \leq \langle \mathit{Dec}_{\mathsf{sk}}(\omega_t), s_{t,a}\rangle + \sqrt{\|s_{t,a}\|^2_{\mathit{Dec}_{\mathsf{sk}}(A_t)} + L\left(t^{3/2}\sqrt{\lambda + L^2 t}\right)^{-1}}$. As a consequence, we can write that the encrypted optimistic index is given by:

$$\rho_a(t) \approx \langle \omega_t, x_{t,a}\rangle + \widetilde{\beta}_t \mathrm{sqrt}_{\mathrm{HE}}\Big(\underbrace{x_{t,a}^\top A_t x_{t,a} + L\left(t^{3/2}\sqrt{\lambda + L^2 t}\right)^{-1}}_{\text{✣}}\Big) \tag{4.12}$$

where $\mathrm{sqrt}_{\mathrm{HE}}$ is an approximate root operator in the encryption space. Unfortunately, computing the root is a non-native operation in HE and we need to build an approximation of it.

For a real value $z \in [0,1]$, we define the following sequences (Cheon et al., 2020)

$$q_{k+1} = q_k\left(1 - \frac{v_k}{2}\right), \quad v_{k+1} = v_k^2\left(\frac{v_k - 3}{4}\right) \tag{4.13}$$

where $q_0 = z$ and $v_0 = z - 1$. It is possible to show that this sequence converges to $\sqrt{z}$.

Proposition 20. *For any $z \in \mathbb{R}_+$, $c_1, c_2 > 0$ with $c_2 \geq z \geq c_1$ and a precision $\varepsilon > 0$, let q_k be the result of k iterations of Eq. (4.13), with $q_0 = \frac{z}{c_2}$ and $v_0 = \frac{z}{c_2} - 1$. Then, $|q_k\sqrt{c_2} - \sqrt{z}| \leq \varepsilon$ for any $k \geq k_0(\varepsilon) := \frac{1}{\ln(2)}\left(\ln\left(\ln\left(\varepsilon\right) - \ln\left(\sqrt{c_2}\right)\right) - \ln\left(4\ln\left(1 - \frac{c_1}{4c_2}\right)\right)\right)$.*

Therefore, by setting $z = \|x_{t,a}\|^2_{A_t} + c_1$ (i.e., as ✣ in Eq. (4.12)), $c_1 = L(t^{3/2}\sqrt{\lambda + L^2 t})^{-1}$, $c_2 = c_1 + L^2\lambda^{-1/2}\left(1 + \lambda^{-1/2}\right)$ and $\varepsilon = t^{-1}$, we set

$$\rho_a(t) = \langle \omega_t, x_{t,a}\rangle + \widetilde{\beta}_t\left(\sqrt{c_2} q_{k_0(1/t)} + \frac{1}{t}\right), \tag{4.14}$$

which implies that $r(s_{t,a}) \lessapprox \max_{\theta \in \widetilde{\mathcal{C}}_t} \langle \theta, s_{t,a}\rangle \leq \mathit{Dec}_{\mathsf{sk}}(\rho_a(t))$. Note that while ω_t, $x_{t,a}$ and q_i are encrypted values, $\widetilde{\beta}_t$, c_1, c_2 and t are plain scalars.

Step ❸: HE Approximate Argmax

The last challenge faced by the learning algorithm is to compute $\arg\max_{a \in [K]}\{\rho_a(t)\}$. Although, it is theoretically possible to compute an argmax procedure operating on encrypted numbers (Gentry and Boneh, 2009), it is highly non practical because it relies on bootstrapping. Recently, Cheon et al. (2020) introduced an homomorphic compatible algorithm (i.e., approximate), called **NewComp**, that builds a polynomial approximation of $\mathrm{Comp}(a,b) = \mathbb{1}_{\{a>b\}}$ for any $a, b \in [0,1]$. This algorithm allows to compute an HE friendly approximation of $\max\{a,b\}$ for any $a, b \in [0,1]$. We leverage this idea to derive **acomp**, a homomorphic compatible algorithm to compute an approximation of the maximum index (see Alg. 39 in App. 4.B.4.5). Precisely, **acomp** does not compute $\arg\max_{a \in [K]}\{\rho_a(t)\}$ but an approximate vector $b_t \approxeq (\mathbb{1}_{\{a = \arg\max_i \rho_i(t)\}})_{a \in [K]}$. The maximum index is the value a such that $(b_t)_a$ is greater than a threshold accounting for the approximation error.

The **acomp** algorithm works in two phases. First, **acomp** computes an approximation M of $\max_{i \in [K]}\{\rho_i(t)\}$ by comparing each pair $(\rho_i(t), \rho_j(t))$ with $i < j \leq K$. Second, each value $\rho_a(t)$ is compared to this approximated maximum value M to obtain $(b_t)_a$, an approximate computation of $\mathbb{1}_{\{\rho_a(t) > M\}}$. Cor. 4 shows that if a component of b_t is big enough, the difference between $\max_a\ \rho_a(t)$ and any arm with $4(b_t)_a \geq t^{-1}$ is bounded by $\widetilde{O}(1/t)$ (proof in App. 4.B.4.5).

Corollary 4. *At any time $t \in [T]$, any arm $a \in [K]$ satisfying $(b_t)_a \geq \frac{1}{4t}$ is such that:*

$$\rho_a(t) \geq \max_{a' \in [K]}\{\rho_{a'}(t)\} - \frac{1}{t} - \frac{\widetilde{\beta}_t}{t}\left[\frac{2}{t} + \sqrt{\frac{L}{t^{3/2}\sqrt{\lambda + L^2 t}}} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}}\right] \tag{4.15}$$

Cor. 4 shows that while an action a such that $4t(b_t)_a \geq 1$ may not belong to $\arg\max_{a \in [K]}\{\rho_t(a)\}$, it can be arbitrarily close, hence limiting the impact on the regret. As shown later, this has little impact on the final regret of the algorithm as the approximation error decreases fast enough. Since b_t is encrypted, the algorithm does not know the action to play. b_t is sent to the user who decrypts it and selects the action to play (the user is the only one having access to **sk**). $b_t \approx (\mathbb{1}_{\{a = \max_{i \in [K]} \rho_i(t)\}}$ indicates to the user which action to take which is necessary by design of the bandit problem. However, if the user is able to

invert the polynomial functions used to compute b_t thanks to the rescaling of the estimates $(\rho_a(t))_a$ the latter can only learn a relative ranking for this particular user and not the actual estimates.

Step ❹: Update Schedule

Thanks to these steps, we can prove (see App. 4.B.5) a $\sqrt{T}$ regret bound for HELBA when ω_t is recomputed at each step t. However, this approach would be impractical due to the extremely high number of multiplications performed. In fact, inverting the design matrix at each step incurs a large multiplicative depth and computational cost. The most natural way of reducing this cost is to reduce the number of times the ridge regression is solved. The arm selection policy will not be updated at each time step but rather only when necessary. Reducing the number of policy changes is exactly the aim of low switching algorithms (see e.g., Abbasi-Yadkori et al., 2011; Perchet et al., 2016; Bai et al., 2019; Calandriello et al., 2020; Dong et al., 2020). We focus on a dynamic, data-dependent batching since $\sqrt{T}$ regret is not attainable using a fixed known-ahead-of-time schedule (Han et al., 2020).

Abbasi-Yadkori et al. (2011) introduced a low switching variant of OFUL (RSOFUL) that recomputes the ridge regression only when the following condition: $\det(V_{t+1}) \geq (1+C)\det(V)$ is met, with V the design matrix after the last update. The regret of RSOFUL scales as $\widetilde{\mathcal{O}}(d\sqrt{(1+C)T})$. In the secure setting, computing the determinant of an encrypted matrix is costly (see e.g. Kaltofen and Villard, 2005) and requires multiple matrix multiplications. The complexity of checking the above condition with HE outweights the benefits introduced by the low switching regime, rendering this technique non practical. Instead of a determinant-based condition, we consider a trace-based condition, inspired by the update rule for GP-BUCB (Desautels et al., 2014; Calandriello et al., 2020).

The "batch j" is defined as the set of time steps between j-th and $(j+1)$-th updates of ω, and we denote by t_j the first time step of this batch. The design matrix is now denoted by $\overline{\Lambda}_j = \lambda \mathit{Enc}_{\mathsf{pk}}(I) + \sum_{l=1}^{t_j-1} x_{l,a_l} x_{l,a_l}^\intercal$, and more importantly is only updated at the beginning of each batch j (and similarly for the inverse $\overline{A}_j$ and vector ω_j). The current batch j is ended if and only if the following *trace-based condition* is met at some time t:

$$C \leq \mathsf{Tr}\left(\sum_{l=t_j+1}^{t-1} \overline{A}_j x_{l,a_l} x_{l,a_l}^\intercal\right) = \sum_{l=t_j+1}^{t-1} \|x_{l,a_l}\|^2_{\overline{A}_j} \tag{4.16}$$

The intuition behind this condition is that the trace of $\overline{V}_j = \lambda I + \sum_{l=1}^{t_j-1} s_{l,a_l} s_{l,a_l}^\intercal$ is enough to directly control the regret. The following proposition shows that the error due to the computation in the encrypted space remains small.

Proposition 21. *Let* $\varepsilon_j = \left(L t_j^{3/2}\sqrt{\lambda + L^2 t_j}\right)^{-1}$ *and* $\overline{A}_j = X_{k_1(\varepsilon_j)}$ *as in Eq.* (4.10) *starting from* $M_0 = \overline{\Lambda}_j/c$ *with* $c \geq \lambda + t_j L^2$. *Then, for any* $j > 0$: $\left|\mathit{Tr}\left(\sum_{l=t_j+1}^{t-1}\left(\mathit{Dec}_{\mathit{sk}}(A_j) - \overline{V}_j^{-1}\right) s_{l,a_l} s_{l,a_l}^\intercal\right)\right| \leq L^2 \varepsilon_j (t-1-t_j)$.

Since the switching condition involves data-dependent encrypted quantities, we leverage a similar procedure as to compare indexes. We compute an (*encrypted*) homomorphic approximation of the sign function thanks to the **acomp** algorithm. The result is an encryption of the approximation of $\mathbb{1}_{\{\}}$. Similarly to computing the argmax of $(\rho_a(t))_a$, the algorithm cannot access the result, thus it relies on the user to decrypt and send the result of the comparison to decide whenever the algorithm needs to update the approximate inverse $\bar{A}_j$, . However, to prevent any information leakage, that is to say the algorithm or the user learning about the features of other users, we use a masking procedure which obsfucates the result of the decryption to the user (detailed in App. 4.B.5.1 and App. 4.B.5.1).

In non-encrypted setting, Cond. 4.16 can be used to dynamically control the growth of the regret, that is bounded by $\mathcal{O}\left(\sum_{j=0}^{M_T}\sum_{t=t_j+1}^{t_{j+1}} \|\overline{V}_j^{-1/2} s_{t,a_t}\|_2\right)$. But in the secure setting, the regret can not be solely bounded as before. The condition for updating the batch has to take into account the approximation error introduced by all the approximate operations. Let M_T be the total number of batches, then the contribution of the approximations to the regret scales as $\sum_{j=0}^{M_T-1} \widetilde{\mathcal{O}}((t_{j+1}-t_j)^2\varepsilon_j)$. We thus introduce an additional condition aiming at explicitly controlling the length of each batch. Let $\eta > 0$, then a new batch is started if Cond. (4.16) is met or if: $t \geq (1+\eta)t_j$. This ensures that the additional regret term grows proportionally to the total number of batches M_T. Note that t_j and t are not encrypted values and the comparison is "simple". The full algorithm is reported in App. 4.B.1.

4.2.4 Theoretical Guarantees

The regret analysis of HELBA is decomposed in two parts. First, we show that, the number of batches is logarithmic in T. Then, we bound the error of approximations per batch.

Proposition 22. *For any* $T > 1$, *if* $C - \frac{L\eta}{\sqrt{\lambda+L^2}} > \frac{1}{4}$, *the number of episodes* M_T *of* HELBA *(see Alg. 28) is bounded by:*

$$M_T \leq 1 + \frac{d\ln\left(1+\frac{L^2 T}{\lambda d}\right)}{2\ln\left(\frac{3}{4} + C - \frac{L\eta}{\sqrt{\lambda+L^2}}\right)} + \frac{\ln(T)}{\ln(1+\eta)} \tag{4.17}$$

The total number of multiplications to compute ω_j is T/M_T-times smaller thanks to the low-switching condition. This leads to a vast improvement in computational complexity. Note that at each round t, HELBA still computes the upper-confidence bound on the reward and the maximum action. Leveraging this result, when any of the batch conditions is satisfied, the regret can be controlled in the same way as the non-batched case, up to a multiplicative constant.

Theorem 12. *Under Asm. 15, for any $\delta > 0$ and $T \geq d$, there exists constants $C_1, C_2 > 0$ such that the regret of* HELBA *(Alg. 28) is bounded with probability $1-\delta$ by:*

$$R_T \leq C_1\beta^\star\left(\sqrt{(1.25+C)\,dT\ln\left(\frac{TL}{\lambda d}\right)} + \frac{L^{3/2}}{\sqrt{\lambda}}\ln(T)\right) + C_2\beta^\star M_T \max\left\{\sqrt{L} + \frac{\eta}{\sqrt{L}}, \eta^2 + \frac{L}{\sqrt{\lambda + L^{2^3}}}\right\}$$

with $\beta^\star = 1 + \sqrt{\lambda}S + \sigma\sqrt{d\left(\ln\left(1+\frac{L^2T}{\lambda d}\right) + \ln\left(\frac{\pi^2T^2}{6\delta}\right)\right)}$ and M_T as in Prop. 22.

The first term of the regret highlights the impact of the approximation of the square root and maximum that are computed at each round. The second term shows the impact of the approximation of the inverse. It depends on the number of batches since the inverse is updated only once per batch. By Prop. 22, we notice that this term has a logarithmic impact on the regret. Finally, the last term is the regret incurred due to low-switch of the optimistic algorithm. We can notice that the parameter C regulates a trade-off between regret and computational complexity. This term is also the regret incurred by running OFUL with trace condition instead of the determinant-based condition. This further stress that the cost of encryption on the regret is only logarithmic, leading to a regret bound of the same order of the non-secure algorithms. But the computationnal complexity of HELBA is multiple orders higher than any non-encrypted bandit algorithm. For example the complexity of computing a scalar product with HE now scales with the ring dimension N and not the dimension of the contexts anymore $d \ll N$.

4.2.5 Discussion And Extensions

In this subsection , a numerical validation of the proposed algorithm in a secure linear bandit problem is privded as well as a discussion about the limitations of the current setting and possible extensions.

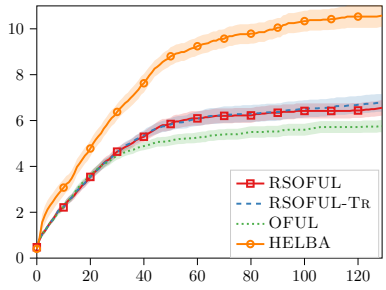


Figure 4.2.1: Regret on a toy problem with 4 random uniform contexts.

Numerical simulation. Despite the mainly theoretical focus of the paper, we illustrate the performance of the proposed algorithm on a toy example, where we aim at empirically validating the theoretical findings. We consider a linear contextual bandit problem with 4 contexts in dimension 2 and 2 arms. As baselines, we consider OFUL, RSOFUL and RSOFUL-Tr (a version of RSOFUL where the determinant-based condition is replaced by the trace-condition in (4.16)). We run these baselines on non-encrypted data and compare the performance with HELBA working with encrypted data. In the latter case, at each step, contexts and rewards are encrypted using the CKKS (Cheon et al., 2017) scheme with parameter $\kappa = 128$, $D = 100$ and $N = 2^{16}$, a modulus $\log(q_0) = 4982$ and a cyclotomic degree of $M = 131072$ chosen automatically by the PALISADE library (PAL, 2020) used for the implementation. The size of the ciphertext is not allowed to grow and a relinearization is performed after every operation. The variance of the noise in the reward is $\sigma = 0.5$. Finally, we use $C = 1$ and $\eta = 0.1$ in HELBA. The regularization parameter is set to 1 and $L = 5.5$. Fig. 4.2.1 shows the regret of the algorithms averaged over 25 repetitions. We notice that while the non-encrypted low-switching algorithms (i.e., RSOFUL and RSOFUL-Tr) recompute the ridge regression only 11 times on average, their performance is only slightly affected by this and it is comparable to the one of OFUL. The reduced number of updates is a significant improvement in light of the current limitation in the multiplicative depth of homomorphic schemes. This was the enabling factor to implement HELBA. Note that the update condition in HELBA increases the number of updates to about 20 on average. As expected, the successive approximations and low-switching combined worsen the regret of HELBA. However, this small loss in performance comes with a provable guarantee on the security of users' data.

Computational Complexity. Even though we reduced the number of multiplications and additions, the total runtime of HELBA is still significant, several orders of magnitude higher compared to the unencrypted setting, the total time for $T = 130$ steps and $\kappa = 128$ bits was 20 hours and 39 minutes. We believe that a speed up can be obtained by optimizing how matrix multiplication is handled. For example, implementation optimization can increase the speed of computation of logistic regression (Blatt et al., 2020). However, we stress that HELBA is almost (up to the masking procedure) agnostic to the homomorphic scheme used, hence any improvement in the HE literature can be leveraged by our algorithm. Bootstrapping procedures (Gentry and Boneh, 2009) can be used for converting a leveled schema into a *Fully HE* scheme. This mechanism, together with the low-switching nature of our algorithm, can be the enabling tool for scaling this approach to large problems as the multiplicative depth scales linearly with the dimension.

Discussion. Many other approaches are possible to increase the computational efficiency, for example using a trusted execution environment (Sabt et al., 2015) or leveraging user-side computational capacities. We decided to design an algorithm where the major computation (except for comparisons) are done server-side, having in mind cloud-computing or recommendations running on mobile phone. The objective was to make as secure as possible this protocol so that the server can leverage the information coming from all users. However, if we assume that users have greater computation capabilities, the algorithm can delegate some computations (see e.g., Blatt et al., 2020). For example, for the inverse, the algorithm can generate a random (invertible) matrix N_t, homomorphically compute V_tN_t and sends the masked matrix, V_tN_t to the user. The latter decrypts, inverts, re-encrypts the inverse and sends it to the algorithm (see (Bost et al., 2015, Sec. 8) for more details). A similar scenario, can be imagined for computing a square root or a matrix multiplication. This protocol requires users to perform computationally heavy operations (inverting a matrix) locally. To ensure security with this delegation, a verification step is needed (see e.g., Bost et al., 2015) further increasing communications between the user and the bandit algorithm. We believe that an interesting direction for future work is to integrate this protocol in a distributed setting (i.e., federated learning). Using a server-side trusted execution environment can speed up computations as operations are executed in the clear in private regions of the memory.

Multi-users Setting. Usually contexts represent different users, described by their features s_t and some users may want to use their own public key $\mathbf{pk}_t$ (and secret key $\mathbf{sk}_t$) to encrypt those features. In that case, HELBA can be used with a KeySwitching Fan and Vercauteren; Brakerski (2012); Brakerski et al. (2014) component. This operation takes a ciphertext c_1 decipherable by a secret key $\mathbf{sk}_1$ and output a ciphertext c_2 decipherable by a secret key $\mathbf{sk}_2$. A user send the encrypted context/reward to the bandit algorithm which perform a key switching (see App. 4.B.3.2) with the help of trusted third party who generate the set of keys used by the learning algorithm such that all ciphertexts received are decipherable by the same key and compatible for homomorphic operations. KeySwitching can be performed without accessing the data and with some (or all) users using their own set of private/public keys for encryption/decryption.

4.3 Conclusion

In the last chapter of this thesis, we explored the security aspect of linear contextual bandit that is to say, if and how bandit algorithms are sensitive to adversarial attacks but also how to leverage advance in the E2E encryption technology to design a bandit algorithm with an almost optimal regret only using encrypted data. As discussed in this chapter, there are still many questions left to explore around those topics. For example, in the case of adversarial attacks on contexts a question left unanswered is to get a lower bound on the percentage of contexts needed to force a linear regret, similar to results in the literature of asynchronous deterministic consensus systems (Fischer et al., 1985). An other interesting question in encrypted linear contextual bandits is if it is possible to derive a regret bound that explicitly trade-off the computational capacity and the regret bound. That is to say, if the algorithm can perform M operations per iteration in average how does the regret changes. This question would also be of interest for a general bandit problem especially when trying to use those in real systems.

Appendix

4.A Appendix for Attacks on Linear Contextual Bandit

4.A.1 Proofs

In this appendix, we present the proofs of different theoretical results presented in the paper.

4.A.1.1 Proof of Proposition 1

Proposition 1. *For any $\delta \in (0, 1/K]$, when using Contextual ACE algorithm (Alg. 24) with perturbed rewards $\tilde{r}^1$, with probability at least $1 - K\delta$, algorithm $\mathfrak{A}$ pulls an arm in $A^\dagger$ for $T - o(T)$ time steps and the total cost of attacks is $o(T)$.*

Proof. Let us consider the contextual bandit problem $\mathcal{A}_1$, with K arms with contexts $x \in \mathcal{D}$ such that every arm in $a^\dagger \in A^\dagger$ has mean reward $\langle \theta_{a^\dagger}, x\rangle$ and all other arms has mean 0. Then the regret of algorithm $\mathfrak{A}$ for this bandit problem is upper-bounded with probability at least $1 - \delta$ by a function $f_{\mathfrak{A}}(T)$ such that $f_{\mathfrak{A}}(T) = o(T)$. In addition, the reward process fed to Alg. $\mathfrak{A}$ by the attacker is a stationary reward process with σ^2-subgaussian noise. Therefore, the number of times algorithm $\mathfrak{A}$ pulls an arm not in $A^\dagger$ is upper-bounded by $f_{\mathfrak{A}}(T)/\min_{x\in\mathcal{D}} \Delta(x)$ where for every context $x \in \mathcal{D}$, let $a^\dagger_\star(x) := \arg\max_{a\in A^\dagger} \langle x, \theta_a\rangle$ and $\Delta(x) = \langle x, \theta_{a^\dagger_\star(x)}\rangle - \max_{a\in A^\dagger, a\neq a^\dagger_\star(x)} \langle x, \theta_a\rangle$.

In addition, the total cost of the attack is upper-bounded by $\max_{a\in[\![1,K]\!]} \max_{x\in\mathcal{D}} |\langle x, \theta_a\rangle|(T - N_{A^\dagger}(T))$ where $N_{A^\dagger}(T)$ is the number of times an arm in $A^\dagger$ has been pulled up to time T. Thanks to the previous argument, $T - N_{A^\dagger}(T) \leq f_{\mathfrak{A}}(T)/\min_{x\in\mathcal{D}} \Delta(x)$. □

4.A.1.2 Proof of Proposition 2

Proposition 2. *Using the attack described in Alg. 25, for any $\delta \in (0, 1/K]$, with probability at least $1 - K\delta$, the number of times* LinUCB *does not pull an arm in $A^\dagger$ is at most:*

$$\sum_{j\notin A^\dagger} N_j(T) \leq 32K^2 \left(\frac{\lambda}{\alpha^2} + \sigma^2 d \log\left(\frac{\lambda d + TL^2\alpha^2}{d\lambda\delta}\right)\right)^3$$

with $N_j(T)$ the number of times arm j has been pulled after T steps, $||\theta_a|| \leq S$ for all arms a, λ the regularization parameter of LinUCB *and for all $x \in \mathcal{D}$, $||x||_2 \leq L$. The total cost for the attacker is bounded by:*

$$\sum_{t=1}^{T} c_t \leq \frac{64K^2}{\nu}\left(\frac{\lambda}{\alpha^2} + \sigma^2 d \log\left(\frac{\lambda d + TL^2\alpha^2}{d\lambda\delta}\right)\right)^3$$

Proof. Let a_t be the arm pulled by LinUCB at time t. For each arms a, let $\tilde{\theta}_a(t)$ be the result of the linear regression with the attacked context and $\hat{\theta}_a(t, \lambda/\alpha^2)$ the one with the unattacked context and a regularization of $\frac{\lambda}{\alpha^2}$. At any time step t, we can write, for all $a \notin A^\dagger$:

$$\tilde{\theta}_a(t) = \left(\lambda I_d + \sum_{l=0, a_l=a}^{t} \alpha^2 x_l x_l^\intercal\right)^{-1} \sum_{k=0,a_k=a}^{t} r_k \alpha x_k = \frac{1}{\alpha}\left(\frac{\lambda}{\alpha^2} I_d + \sum_{k=0,a_k=a}^{t} x_k x_k^\intercal\right)^{-1} \sum_{k=0,a_k=a}^{t} r_k x_k = \frac{\hat{\theta}_a(t, \lambda/\alpha^2)}{\alpha}$$

We also note that, since the contexts are not modified for arms in $a^\dagger \in A^\dagger$: $\tilde{\theta}_{a^\dagger}(t) = \hat{\theta}_{a^\dagger}(t, \lambda)$. In addition, for any context x and arm $a \notin A^\dagger$, the exploration term used by LinUCB becomes:

$$||x||_{\tilde{V}^{-1}_{a,t}} = \frac{1}{\alpha}||x||_{\hat{V}^{-1}_{a,t}} \tag{4.18}$$

where $\tilde{V}_{a,t} = \lambda I_d + \sum_{l=0,a_l=a}^{t} \alpha^2 x_l x_l^\intercal$ and $\hat{V}_{a,t}^{-1} = \lambda/\alpha^2 I_d + \sum_{k=0,a_k=a}^{t} x_k x_k^\intercal$. For a time t, if presented with context x_t LinUCB pulls arm $a_t \notin A^\dagger$, we have:

$$\alpha\left(\left\langle \hat{\theta}_{a^\dagger}(t), x_t\right\rangle + \beta_{a^\dagger}(t)||x_t||_{V_{a^\dagger,t}^{-1}}\right) \leq \left\langle \hat{\theta}_{a_t}(t, \lambda/\alpha^2), x_t\right\rangle + \beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}}$$

As $\alpha = \frac{2}{\nu} \geq \min_{a^\dagger \in A^\dagger} \frac{2}{\langle \theta_{a^\dagger}, x_t\rangle}$, we deduce that on the event that the confidence sets (Theorem 2 in Abbasi-Yadkori et al. (2011)) hold for arm $a^\star$:

$$2 \leq \left\langle \hat{\theta}_{a_t}(t, \lambda/\alpha^2), x_t\right\rangle + \beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}} \leq \langle \theta_{a_t}, x_t\rangle + 2\beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}}$$

Thus, $1 \leq 2 - \langle \theta_{a_t}, x_t\rangle \leq 2\beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}}$. Therefore,

$$\sum_{t=1}^{T} \mathbb{1}_{\{a_t \notin A^\dagger\}} \leq \sum_{t=1}^{T} \min(2\beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}}, 1)\mathbb{1}_{\{a_t \notin A^\dagger\}} \leq \sum_{j \notin A^\dagger} 2\beta_j(T)\sqrt{\sum_{t=1}^{T} \mathbb{1}_{\{a_t=j\}} \sum_{t=1,a_t=j}^{T} \min(1, ||x_t||^2_{\hat{V}_{j,t}^{-1}})}$$

But using Lemma 11 from Abbasi-Yadkori et al. (2011) and the bound on the $\beta_j(T)$ for all arms j, we have with Jensen inequality:

$$\sum_{t=1}^{T} \mathbb{1}_{\{a_t \notin A^\dagger\}} \leq 4\sqrt{K \sum_{t=1}^{T} \mathbb{1}_{\{a_t \notin A^\dagger\}} d \log\left(1 + \frac{\alpha^2 T L^2}{\lambda d}\right)}\left(\sqrt{\lambda/\alpha^2}S + \sigma\sqrt{2\log(1/\delta) + d\log(1 + \frac{\alpha^2 T L^2}{\lambda d})}\right)$$

□

4.A.1.3 Proof of Theorem 11

Theorem. *For any $\xi > 0$, Problem (4.4) is feasible if and only if:*

$$\exists \theta \in \bigcup_{a^\dagger \in A^\dagger} \mathcal{C}_{t,a^\dagger}, \qquad \theta \notin \mathit{Conv}\left(\bigcup_{a \notin A^\dagger} \mathcal{C}_{t,a}\right) \tag{4.19}$$

where for every arm a, $\mathcal{C}_{t,a} := \left\{\theta \mid ||\theta - \hat{\theta}_a(t)||_{\tilde{V}_{a,t}} \leq \beta_a(t)\right\}$ with $\hat{\theta}_a(t)$ the least squares estimate for arm a built by LinUCB *and*

$$\tilde{V}_{a,t} = \lambda I_d + \sum_{l=1,x_l \neq x^\dagger}^{t} \mathbb{1}_{\{a_l=a\}} x_l x_l^\intercal + \sum_{l=1,x_l=x^\dagger}^{t} \mathbb{1}_{\{a_l=a\}} \tilde{x}_l \tilde{x}_l^\intercal$$

the design matrix of LinUCB *at time t for all arms a (where $\tilde{x}_l$ is the modified context)*

Proof. The proof of Theorem 11 is decomposed in two parts.

First, let us assume that Equation (4.19) is satisfied. Then, let us define $a^\dagger \in A^\dagger$ such that $\theta \in \mathcal{C}_{t,a^\dagger} \setminus \mathrm{Conv}\left(\bigcup_{a \notin A^\dagger} \mathcal{C}_{t,a}\right)$, then by the theorem of separation of convex sets applied to $\mathcal{C}_{t,a^\dagger}$ and $\{\theta\}$. There exists a vector v and $c_1 < c_2$ such that for all $y \in \mathrm{Conv}\left(\bigcup_{a \neq a^\dagger} \mathcal{C}_{t,a}\right)$:

$$\langle y, v\rangle \leq c_1 < c_2 \leq \langle \theta, v\rangle .$$

Hence, for $\xi > 0$ we have that for $\tilde{v} = \frac{\xi}{c_2 - c_1} v$ that:

$$\langle y, \tilde{v}\rangle + \xi \leq \langle \theta, \tilde{v}\rangle$$

So the problem is feasible.

Secondly, let us assume that an attack is feasible. Then there exists a vector y such that:

$$\max_{a^\dagger \in A^\dagger} \max_{\theta \in \mathcal{C}_{t,a^\dagger}} \langle y, \theta\rangle > c_1 := \max_{a \notin A^\dagger} \max_{\theta \in \mathcal{C}_{t,a}} \langle y, \theta\rangle \tag{4.20}$$

Let us reason by contradiction. We assume that $\bigcup_{a \in A^\dagger} \mathcal{C}_{t,a^\dagger} \subset \mathrm{Conv}\left(\bigcup_{a \notin A^\dagger} \mathcal{C}_{t,a}\right)$ and consider

$$\theta^* \in \bigcup_{a \in A^\dagger} \mathcal{C}_{t,a^\dagger} \text{ such that } \langle y, \theta^*\rangle = \max_{a^\dagger \in A^\dagger} \max_{\theta \in \mathcal{C}_{t,a^\dagger}} \langle y, \theta\rangle$$

As we assumed $\bigcup_{a\in A^\dagger} \mathcal{C}_{t,a^\dagger} \subset \text{Conv}\left(\bigcup_{a\notin A^\dagger} \mathcal{C}_{t,a}\right)$, there exists $n \in \mathbb{N}^\star$, $\lambda_1, \cdots, \lambda_n \geq 0$ and $\theta_1, \cdots, \theta_n \in \bigcup_{a\notin A^\dagger} \mathcal{C}_{t,a}$ such that

$$\theta^* = \sum_{i=1}^{n} \lambda_i \theta_i \text{ and } \sum_{i=1}^{n} \lambda_i = 1$$

Thus

$$\langle y, \theta^* \rangle = \sum_i \lambda_i \langle y, \theta_i \rangle \leq c_1 \sum_{i=1}^{n} \lambda_i = c_1 \tag{4.21}$$

We assumed that the problem is feasible, so $c_1 < \langle y, \theta^* \rangle$ according to Eq. 4.20. It contradicts Eq. 4.21. □

4.A.1.4 Condition of Sec. 4.1.4

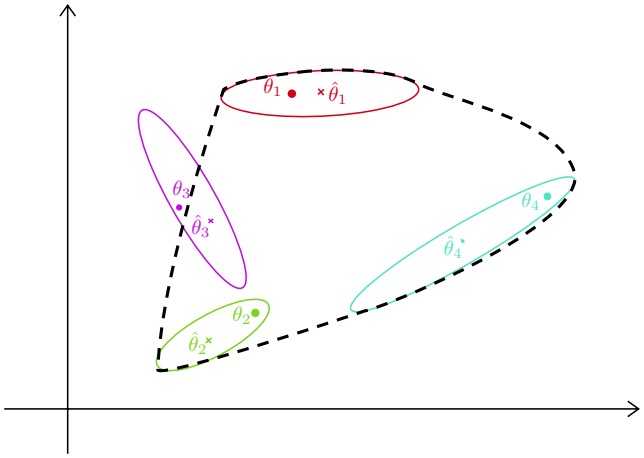


Figure 4.A.1: Illustrative example of condition (4.5). The target arm is arm 3 or 5 and the dashed black line is the convex hull of the other confidence sets. The ellipsoids are the confidence sets $\mathcal{C}_{t,a}$ for each arm a. If we consider only arms $\{1, 2, 4, 5\}$, and we use 5 as the target arm, the condition (4.5) is satisfied as there is a θ outside the convex hull of the other confidence sets. On the other hand, if we consider arms $\{1, 2, 3, 4\}$ and we use 3 as the target arm, the condition is not satisfied anymore.

Let us assume that there is an arm in $a^\dagger \in A^\dagger$ which is optimal for some contexts. More formally, there exists a subspace $V \subset \mathcal{D}$ such that:

$$\forall x \in V, \exists a^\dagger_\star(x) \in A^\dagger, \forall a \in [\![1, K]\!] \setminus \{a^\dagger_\star(x)\} \qquad \langle x, \theta_{a^\dagger_\star(x)} \rangle > \langle x, \theta_a \rangle .$$

We also assume that the distribution of the contexts is such that, for all t, $\mu := \mathbb{P}(x_t \in V) > 0$. Then, the regret is lower-bounded in expectation by:

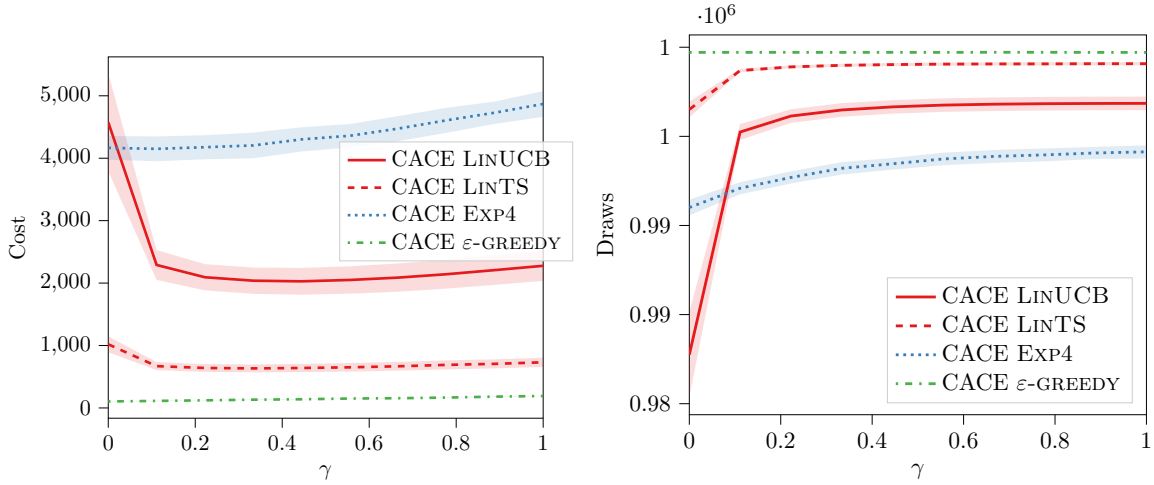
$$\mathbb{E}(R_T) = \mathbb{E}\left(\sum_{t=1}^{T} \mathbb{1}_{\{x_t \in V\}} \left(\left\langle x_t, \theta_{a^\dagger_\star(x_t)} - \theta_{a_t} \right\rangle \right)\right) \geq \mu m(T) \min_{x\in V} \max_{a\neq a^\dagger_\star(x)} \langle \theta_{a^\dagger_\star(x)} - \theta_a, x \rangle$$

where $m(T)$ is the expected number of times $t \leq T$ such that condition (4.5) is not met. LinUCB guarantees that $\mathbb{E}(R_T) \leq \mathcal{O}(\sqrt{T})$ for every T. Hence, $m(T) \leq \mathcal{O}\left(\frac{\sqrt{T}}{\mu \min_{x\in V} \max_{a\neq a^\dagger_\star(x)} \langle \theta_{a^\dagger_\star(x)} - \theta_a, x \rangle}\right)$. This means that, in an unattacked problem, condition (4.5) is met $T - \mathcal{O}(\sqrt{T})$ times. On the other hand, when the algorithm is attacked the regret of LinUCB is not sub-linear as the confidence bound for the target arm is not valid anymore. Hence we cannot provide the same type of guarantees for the attacked problem.

4.A.2 Experiments

4.A.2.1 Datasets and preprocessing

We present here the datasets used in the article and how we preprocess them for numerical experiments conducted in subsection 4.1.5.



(a) Total cost

(b) Number of draws

Figure 4.A.2: Total cost of attacks and number of draws of the target arm at $T = 10^6$ as a function of γ on synthetic data

We consider two types of experiments, one on synthetic data with a contextual MAB problems with $K = 10$ arms such that for every arm a, θ_a is drawn from a folded normal distribution in dimension $d = 30$. We also use a finite number of contexts (10), each of them is drawn from a folded normal distribution projected on the unit circle multiplied by a uniform radius variable (i.i.d. across all contexts). Finally, we scale the expected rewards in $(0, 1]$ and the noise is drawn from a centered Gaussian distribution $\mathcal{N}(0, 0.01)$.

The second type of experiments is conducted in the real-world datasets Jester Goldberg et al. (2001) and MovieLens25M Harper and Konstan (2015). Jester consists of joke ratings on a continuous scale from -10 to 10 for 100 jokes from a total of 73421 users. We use the features extracted via a low-rank matrix factorization ($d = 35$) to represent the actions (i.e., the jokes). We consider a complete subset of 40 jokes and 19181 users . Each user rates all the 40 jokes. At each time, a user is randomly selected from the 19181 users and mean rewards are normalized in $[0, 1]$. The reward noise is $\mathcal{N}(0, 0.01)$. The second dataset we use is MovieLens25M. It contains 25000095 ratings created by 162541 users on 62423 movies. We perform a low-rank matrix factorization to compute users features and movies features. We keep only movies with at least 1000 ratings, which leave us with 162539 users and 3794 movies. At each time step, we present a random user, and the reward is the scalar product between the user feature and the recommend movie feature. All rewards are scaled to lie in $[0, 1]$ and a Gaussian noise $\mathcal{N}(0, 0.01)$ is added to the rewards.

4.A.2.2 Attacks on Rewards

In this appendix, we present empirical evolution of the total cost and the number of draws for a unique target arm as a function of the attack parameter γ for the Contextual ACE attack with perturbed rewards $\tilde{r}^2$ on generated data.

Fig. 4.A.2 (left) shows that the total cost of attacks seems to be quite invariant w.r.t. γ except when $\gamma \to 0$ because the difference between the target arm and the other becomes negligible. This is also depicted by the total number of draws (Fig. 4.A.2, Right) as the number of draws plummets when $\gamma \to 0$.

4.A.2.3 Attacks on all Contexts

Fig. 4.A.3 shows the regret for all the attacks. This figure shows that even though the total cost of attacks is linear for algorithms like LinTS in the synthetic dataset, the regret is linear. More generally, we observe that the regret is linear for all attacked algorithms on all datasets.

4.A.2.4 Attack on a single context

The attacks are computed by solving the optimization problems 4.4 and 4.6 (Sec. 4.1.4). We choose the libraries according to their efficiency for each problem we need to solve. For Problem (4.6) and Problem (4.8) we use CVXPY Agrawal et al. (2018) and the ECOS solver. For Problem (4.4) we use the SLSQP method from the Scipy optimize library Virtanen et al. (2019) to solve the full LinUCB problem (Equation 4.4) and QUADPROG to solve the quadratic problem to attack ε-greedy.

Table 4.A.1: Number of draws of the target arm $a^\dagger$ at $T = 10^6$, for the synthetic data, $\gamma = 0.22$ for the Contextual ACE algorithm and for the Jester and MovieLens datasets $\gamma = 0.5$.

	Synthetic	Jester	Movilens
LinUCB	$86,731.6$	$23,548.16$	$25,017.31$
CACE LinUCB	$996,238.6$	$921,083.69$	$944,721.28$
Stationary CACE LinUCB	$995,578.88$	$862,095.67$	$931,531.6$
ε-greedy	$111,380.44$	$21,911.54$	$3,165.81$
CACE ε-greedy	$999,812.92$	$999,755.72$	$999,776.82$
Stationary CACE ε-greedy	$999,806.32$	$999,615.98$	$999,316.76$
LinTS	$91,664.8$	$23,398.3$	$30,189.84$
CACE LinTS	$998,997.04$	$976,708.9$	$990,250.67$
Stationary CACE LinTS	$977,850.96$	$784,715.62$	$845,512.98$
Exp4	$93,860.4$	$29,147.01$	$17,985.78$
CACE Exp4	$992,793.36$	$989,214.36$	$936,230.4$
Stationary CACE Exp4	$993,673.24$	$988,463.56$	$934,304.23$

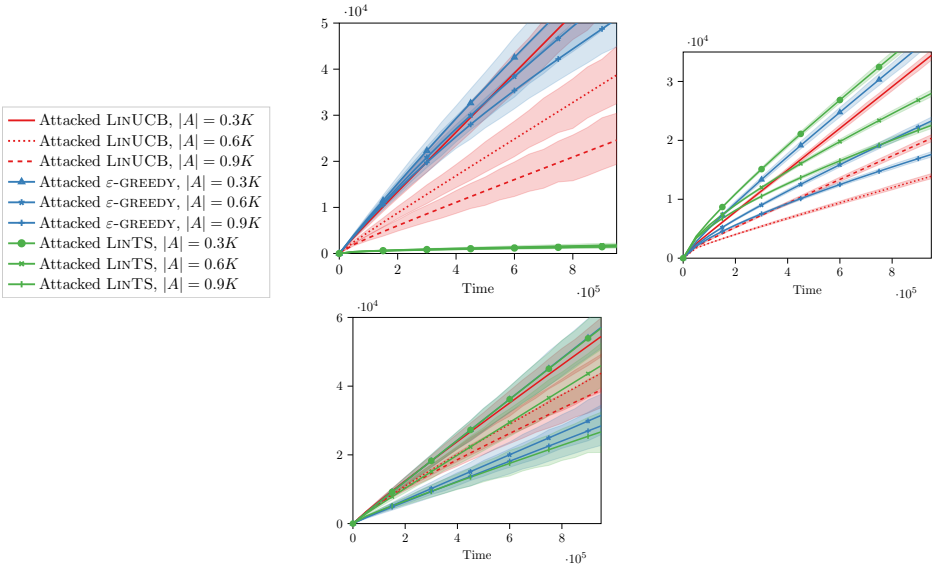


Figure 4.A.3: Total cost of attacks and number of draws of the target arm at $T = 10^6$ as a function of γ on synthetic data

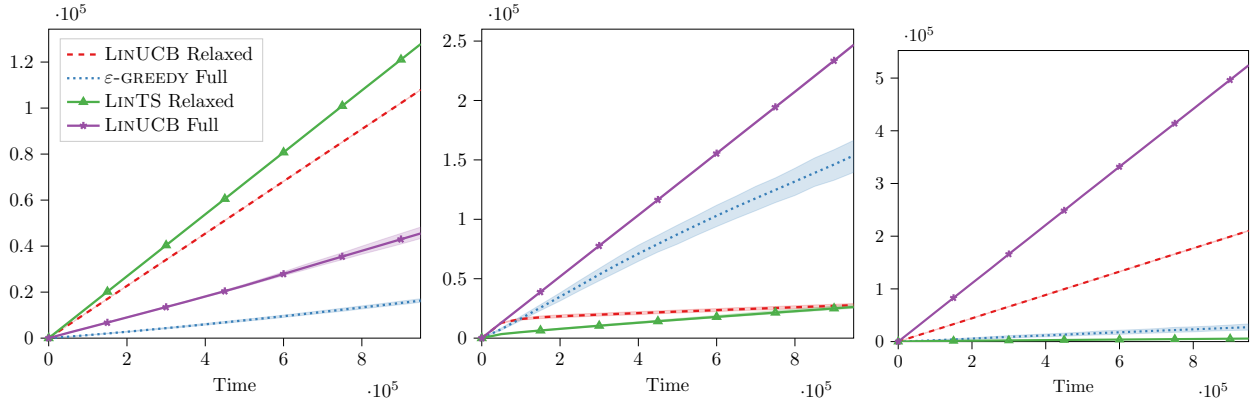
4.A.3 Problem (4.8) as a Second Order Cone (SOC) Program

Problem (4.6) and Problem (4.8) are both SOC programs. We can see the similarities between both problems as follows. Let us define for every arm $a \notin A^\dagger$, the ellipsoid:

$$\mathcal{C}'_{t,a} := \left\{ y \in \mathbb{R}^d \mid ||y - \hat{\theta}_a(t)||_{A_a^{-1}(t)} \leq \upsilon \Phi^{-1}\left(1 - \frac{\delta}{K - |A^\dagger|}\right) \right\}$$

with $A_a(t) = \tilde{V}_a^{-1}(t) + \tilde{V}_{a^\dagger}^{-1}(t)$ with $\tilde{V}_a(t)$ and $\tilde{V}_{a^\dagger}(t)$ the design matrix built by LinTS and $\hat{\theta}_a(t)$ the least squares estimate of θ_a at time t. Therefore for an arm a, the constraint in Problem (4.8) can be written for any $y \in \mathbb{R}^d$ and some arm $a^\dagger \in A^\dagger$ as:

$$\left\langle x^\star + y, \hat{\theta}_{a^\dagger}(t) \right\rangle - \xi \geq \max_{z \in \mathcal{C}'_{t,a}} \langle z, x^\star + y \rangle$$



(a) Synthetic data (b) Jester Dataset (c) MovieLens Dataset

Figure 4.A.4: Total cost of the attacks for the attacks one one context on our synthetic dataset, Jester and MovieLens. As expected, the total cost is linear.

Indeed for any $x \in \mathbb{R}^d$,

$$\begin{aligned}\max_{y \in \mathcal{C}'_{t,a}} \langle y, x\rangle &= \left\langle x, \hat{\theta}_a(t)\right\rangle + \upsilon\Phi^{-1}\left(1 - \frac{\delta}{K - |A^\dagger|}\right) \times \max_{||A_a^{-1/2}(t)u||_2 \leq 1} \langle u, x\rangle \\ &= \left\langle x, \hat{\theta}_a(t)\right\rangle + \upsilon\Phi^{-1}\left(1 - \frac{\delta}{K - |A^\dagger|}\right) \max_{||z||_2 \leq 1} \left\langle z, A_a^{1/2}(t)x\right\rangle \\ &= \left\langle x, \hat{\theta}_a(t)\right\rangle + \upsilon\Phi^{-1}\left(1 - \frac{\delta}{K - |A^\dagger|}\right) \|A_a^{1/2}(t)x\|_2\end{aligned}$$

Thus, the constraint is feasible if and only if:

$$\hat{\theta}_{a^\dagger}(t) \notin \text{Conv}\left(\bigcup_{a \notin A^\dagger} \mathcal{C}'_{t,a}\right)$$

4.A.4 Attacks on Adversarial Bandits

In the previous subsection s, we studied algorithms with sublinear regret R_T, i.e., mainly bandit algorithms designed for stochastic stationary environments. Adversarial algorithms like Exp4 do not provably enjoy a sublinear **stochastic** regret R_T (as defined in the introduction) [5]. In addition, because this type of algorithms are, by design, robust to non-stationary environments, one could expect them to induce a linear cost on the attacker. In this subsection , we show that this is not the case for most contextual adversarial algorithms. Contextual adversarial algorithms are studied through the reduction to the bandit with expert advice problem. This is a bandit problem with K arms where at every step, N experts suggest a probability distribution over the arms. The goal of the algorithm is to learn which expert gets the best expected reward in hindsight after T steps. The regret in this type of problem is defined as $R_T^{\text{exp}} = \mathbb{E}\left(\max_{m \in [\![1,N]\!]} \sum_{t=1}^T \sum_{j=1}^K E_{m,j}^{(t)} r_{t,j} - r_{t,a_t}\right)$ where $E_{m,j}^{(t)}$ is the probability of selecting arm j for expert m. In the case of contextual adversarial bandits, the experts first observe the context x_t before recommending an expert m. Assuming the current setting with linear rewards, we can show that if an algorithm $\mathfrak{A}$, like Exp4, enjoys a sublinear regret R_T^{exp}, then, using the Contextual ACE attack with either $\tilde{r}^1$ or $\tilde{r}^2$, the attacker can fool the algorithm into pulling arm $a^\dagger$ a linear number of times under some mild assumptions. However, attacking contexts for this type of algorithm is difficult because, even though the rewards are linear, the experts are not assumed to use a specific model for selecting an action.

Proposition 3. *Suppose an adversarial algorithm $\mathfrak{A}$ satisfies a regret R_T^{exp} of order $o(T)$ for any bandit problem and that there exists an expert $m^\star$ such that $T - \sum_{t=1}^T \mathbb{E}\left(E_{m^\star, a_{t,\star}^\dagger}^{(t)}\right) = o(T)$ with $a_{t,\star}^\dagger$ the optimal arim in $A^\dagger$ at time t. Then attacking alg. $\mathfrak{A}$ with Contextual ACE leads to pulling arm $a^\dagger$, $T - o(T)$ of times in expectation with a total cost of $o(T)$ for the attacker.*

[5] Exp4 enjoys a sublinear hindsight regret though. Showing a sublinear upper-bound for the stochastic regret of Exp4 is still an open problem (see subsection 29.1 in Lattimore and Szepesvári (2018))

Proof. Similarly to the proof of Proposition 1, let's define the bandit with expert advice problem, $\mathcal{A}_i$, such that at each time t the reward vector is $(\tilde{r}^i_{t,a})_a$ (with $i \in \{1,2\}$). The regret of this algorithm is: $\tilde{R}^{i,\exp}_T = \mathbb{E}\left(\max_{m\in[\![1,N]\!]} \sum_{t=1}^T E^{(t)}_m \tilde{r}^i_t - \tilde{r}^i_{t,a_t}\right) \in o(T)$. The regret of the learner is: $\mathbb{E}\left(\max_{m\in[\![1,N]\!]} \sum_{t=1}^T E^{(t)}_m r_t - r_{t,a_t}\right)$ where a_t are the actions taken by algorithm $\mathcal{A}_i$ to minimize $\tilde{R}^{i,\exp}_T$. Then we have:

$$\tilde{R}^{i,\exp}_T \geq \mathbb{E}\left(\sum_{t=1}^T \sum_{j=1}^K (E^{(t)}_{m^\star,j} - \mathbb{1}_{\{j=a^\dagger_{t,\star}\}})\tilde{r}^i_{t,j} + \sum_{t=1}^T \tilde{r}^i_{t,a^\dagger_{t,\star}} - \tilde{r}^i_{t,a_t}\right)$$

Therefore,

$$\begin{aligned}
\mathbb{E}\left(\sum_{t=1}^T \tilde{r}^i_{t,a^\dagger_{t,\star}} - \tilde{r}^i_{t,a_t}\right) &\leq \tilde{R}^{i,\exp}_T + \mathbb{E}\left(\sum_{t=1}^T \sum_{j=1}^K (\mathbb{1}_{\{j=a^\dagger_{t,\star}\}} - E^{(t)}_{m^\star,j})\tilde{r}^i_{t,j}\right) \\
&\leq \tilde{R}^{i,\exp}_T + \mathbb{E}\left(\sum_{t=1}^T (1 - E^{(t)}_{m^\star,a^\dagger_{t,\star}})\tilde{r}^i_{t,j}\right) \\
&\leq \tilde{R}^{i,\exp}_T + \mathbb{E}\left(\sum_{t=1}^T (1 - E^{(t)}_{m^\star,a^\dagger_{t,\star}})\right)
\end{aligned}$$

For strategy $i = 1$, we have:

$$\mathbb{E}\left(\sum_{t=1}^T \tilde{r}^1_{t,a^\dagger_{t,\star}} - \tilde{r}^1_{t,a_t}\right) = \sum_{t=1}^T \mathbb{E}\left(r_{t,a^\dagger_{t,\star}} - \mathbb{1}_{\{a_t \in A^\dagger\}}\right) \geq \left(T - \mathbb{E}\left(\sum_{t=1}^T \mathbb{1}_{\{a_t = a^\dagger_{t,\star}\}}\right)\right)\Delta$$

where $\Delta := \min_{x\in\mathcal{D}}\left\{\langle\theta_{a^\dagger(x)}, x\rangle - \max_{a\in A^\dagger, a\neq a^\dagger(x)}\langle\theta_{a'}, x\rangle\right\}$ with $a^\dagger(x) := \arg\max_{a\in A^\dagger}\langle\theta_a, x\rangle$. Then, as $\tilde{R}^{1,\exp}_T \in o(T)$ and $\mathbb{E}\left(\sum_{t=1}^T (1 - E^{(t)}_{m^\star,a^\dagger_{t,\star}})\right) \in o(T)$, we deduce that $\mathbb{E}(\sum_t \mathbb{1}_{\{a_t = a^\dagger_{t,\star}\}}) = T - o(T)$.

For strategy $i = 2$, and $\delta > 0$, let us denote by E_δ the event that all confidence intervals hold with probability $1-\delta$. But on the event E_δ, for a time t where $a_t \neq a^\dagger_{t,\star}$ and such that $-1 \leq C_{t,a_t} \leq 0$:

$$\begin{aligned}
\tilde{r}^2_{t,a_t} = r_{t,a_t} + C_{t,a_t} &= (1-\gamma)\min_{a^\dagger\in A^\dagger}\min_{\theta\in\mathcal{C}_{t,a^\dagger}}\langle\theta, x_t\rangle + \eta_{a_t,t} + \langle\theta_a, x_t\rangle - \max_{\theta\in\mathcal{C}_{t,a_t}}\langle\theta, x_t\rangle \\
&\leq (1-\gamma)\langle\theta_{a^\dagger_{t,\star}}, x_t\rangle + \eta_{a_t,t}
\end{aligned}$$

when $C_{t,a_t} > 0$ (still on the event E_δ):

$$\tilde{r}^2_{t,a_t} = r_{t,a_t} \leq (1-\gamma)\langle\theta_{a^\dagger_{t,\star}}, x_t\rangle + \eta_{a_t,t}$$

because $C_{t,a_t} > 0$ means that $(1-\gamma)\langle\theta_{a^\dagger_{t,\star}}, x_t\rangle \geq (1-\gamma)\min_{a^\dagger\in A^\dagger}\min_{\theta\in\mathcal{C}_{t,a^\dagger}}\langle\theta, x_t\rangle \geq \max_{\theta\in\mathcal{C}_{t,a_t}}\langle\theta, x_t\rangle \geq \langle\theta_a, x_t\rangle$. But finally, when $C_{t,a_t} \leq -1$, $\tilde{r}^2_{t,a_t} = r_{t,a_t} - 1 \leq \eta_{a_t,t} \leq (1-\gamma)\langle\theta_{a^\dagger_{t,\star}}, x_t\rangle + \eta_{a_t,t}$. But on the complementary event E^c_δ, $\tilde{r}^2_{t,a_t} \leq r_{t,a_t}$. Thus, given that the expected reward is assumed to be bounded in $(0,1]$ (Assumption 13):

$$\mathbb{E}\left(\sum_{t=1}^T \tilde{r}^2_{t,a^\dagger_{t,\star}} - \tilde{r}^2_{t,a_t}\right) = \mathbb{E}\left(\sum_{t=1}^T (r_{t,a^\dagger} - \tilde{r}^2_{t,a_t})\mathbb{1}_{\{a_t \neq a^\dagger_{t,\star}\}}\right) \geq \mathbb{E}\left(\sum_{t=1}^T \min\{\gamma \min_{x\in\mathcal{D}}\langle x, \theta_{a^\dagger_{t,\star}}\rangle, \Delta\}\mathbb{1}_{\{a_t \neq a^\dagger_{t,\star}\}}\mathbb{1}_{\{E_\delta\}}\right) - T\delta$$

Finally, putting everything together we have:

$$\mathbb{E}\left(\sum_{t=1}^T \gamma \min_{x\in\mathcal{D}}\langle x, \theta_{a^\dagger_{t,\star}}\rangle\mathbb{1}_{\{a_t \neq a^\dagger_{t,\star}\}}\right) \leq \tilde{R}^{2,\exp}_T + \mathbb{E}\left(\sum_{t=1}^T (1 - E^{(t)}_{m^\star,a^\dagger_{t,\star}})\right) + \delta T\left(\min\{\gamma \min_{a^\dagger\in A^\dagger}\min_{x\in\mathcal{D}}\langle x, \theta_{a^\dagger}\rangle, \Delta\} + 1\right)$$

Hence, because $\tilde{R}^{1,\exp}_T = o(T)$ and $\mathbb{E}\left(\sum_{t=1}^T (1 - E^{(t)}_{m^\star,a^\dagger})\right) = o(T)$ we have that for $\delta \leq 1/T$, the expected number of pulls of the optimal arm in $A^\dagger$ is of order $o(T)$. In addition, the cost for the attacker is bounded by:

$$\mathbb{E}\left(\sum_{t=1}^T c_t\right) = \mathbb{E}\left(\sum_{t=1}^T \mathbb{1}_{\{a_t \neq a^\dagger_{t,\star}\}}\big|\max(-1, \min(C_{t,a_t}, 0))\big|\right) \leq \mathbb{E}\left(\sum_{t=1}^T \mathbb{1}_{\{a_t \neq a^\dagger_{t,\star}\}}\right)$$

□

The proof is similar to the one of Prop. 1. The condition on the expert in Prop. 3 means that there exists an expert which believes an arm $a^\dagger \in A^\dagger$ is optimal most of the time. The adversarial algorithm will then learn that this expert is optimal. Algorithm EXP4 has a regret R_T^{exp} bounded by $\sqrt{2TK\log(N)}$, thus the total number of pulls of arms not in $A^\dagger$ is bounded by $\sqrt{2TK\log(M)}/\gamma$. This result also implies that for adversarial algorithms like EXP3 Auer et al. (2002b), the same type of attacks could be used to fool $\mathfrak{A}$ into pulling arms in $A^\dagger$ because the MAB problem can be seen as a reduction of the contextual bandit problem with a unique context and one expert for each arm.

4.A.5 Contextual Bandit Algorithms

In this appendix, we present the different bandit algorithms studied in this paper. All algorithms we consider except EXP4 uses disjoint models for building estimate of the arm feature vectors $(\theta_a)_{a\in[\![1,K]\!]}$. Each algorithm (except EXP4) builds least squares estimates of the arm features.

Algorithm 29: Contextual LINUCB

Input: regularization λ, number of arms K, number of rounds T, bound on context norms: L, bound on norms θ_a: D
Initialize for every arm a, $\bar{V}_a^{-1}(t) = \frac{1}{\lambda} I_d$, $\hat{\theta}_a(t) = 0$ and $b_a(t) = 0$
for $t = 1, \dots, T$ **do**
Observe context x_t
Compute $\beta_a(t) = \sigma\sqrt{d\log\left(\frac{1+N_a(t)L^2/\lambda}{\delta}\right)}$ with $N_a(t)$ the number of pulls of arm a
Pull arm $a_t = \arg\max_a \langle \hat{\theta}_a(t), x_t\rangle + \beta_a(t)||x_t||_{\bar{V}_a^{-1}(t)}$
Observe reward r_t and update parameters $\hat{\theta}_a(t)$ and $\bar{V}_a^{-1}(t)$ such that:

$$\bar{V}_{a_t}(t+1) = \bar{V}_{a_t}(t) + x_t x_t^\intercal, \quad b_{a_t}(t+1) = b_{a_t}(t) + r_t x_t, \quad \theta_{a_t}(t+1) = \bar{V}_{a_t}^{-1}(t+1) b_{a_t}(t+1)$$

Algorithm 30: Linear Thompson Sampling with Gaussian prior

Input: regularization λ, number of arms K, number of rounds T, variance υ
Initialize for every arm a, $\bar{V}_a^{-1}(t) = \lambda I_d$ and $\hat{\theta}_a(t) = 0$, $b_a(t) = 0$
for $t = 1, \dots, T$ **do**
Observe context x_t
Draw $\tilde{\theta}_a \sim \mathcal{N}(\hat{\theta}_a(t), \upsilon^2 \bar{V}_a^{-1}(t))$
Pull arm $a_t = \arg\max_{a\in[\![1,K]\!]} \left\langle \tilde{\theta}_a, x_t \right\rangle$
Observe reward r_t and update parameters $\hat{\theta}_a(t)$ and $\bar{V}_a^{-1}(t)$

$$\bar{V}_{a_t}(t+1) = \bar{V}_{a_t}(t) + x_t x_t^\intercal, \quad b_{a_t}(t+1) = b_{a_t}(t) + r_t x_t, \quad \theta_{a_t}(t+1) = \bar{V}_{a_t}^{-1}(t+1) b_{a_t}(t+1)$$

Algorithm 31: ε-GREEDY

Input: regularization λ, number of arms K, number of rounds T, exploration parameter $(\varepsilon)_t$
Initialize, for all arms a, $\bar{V}_a^{-1}(t) = \lambda I_d$ and $\hat{\theta}_a(t) = 0$, $\varepsilon_t = 1$, $b_a(t) = 0$ **for** $t = 1, \dots, T$ **do**
Observe context x_t
With probability ε_t, pull $a_t \sim \mathcal{U}([\![1, K]\!])$, or pull $a_t = \arg\max\langle \theta_a, x_t\rangle$
Observe reward r_t and update parameters $\hat{\theta}_a(t)$ and $\bar{V}_a^{-1}(t)$

$$\bar{V}_{a_t}(t+1) = \bar{V}_{a_t}(t) + x_t x_t^\intercal, \quad b_{a_t}(t+1) = b_{a_t}(t) + r_t x_t,$$
$$\theta_{a_t}(t+1) = \bar{V}_{a_t}^{-1}(t+1) b_{a_t}(t+1)$$

Algorithm 32: EXP4

Input: number of arms K, experts: $(E_m)_{m\in[\![1,N]\!]}$, parameter η
Set $Q_1 = (1/N)_{j\in[\![1,N]\!]}$
for $t = 1, \dots, T$ **do**
  Observe context x_t and probability recommendation $(E_m^{(t)})_{m\in[\![1,N]\!]}$
  Pull arm $a_t \sim P_t$ where $P_{t,j} = \sum_{k=1}^N Q_{t,k} E_{j,k}^{(t)}$
  Observe reward r_t and define for all arms i $\hat{r}_{t,i} = 1 - \mathbb{1}_{\{a_t=i\}}(1-r_t)/P_{t,i}$
  Define $\tilde{X}_{t,k} = \sum_a E_{k,a}^{(t)} \hat{r}_{t,a}$
  Update $Q_{t+1,j} = \exp(\eta Q_{t,i}) / \sum_{j=1}^N \exp(\eta Q_{t,j})$ for all experts i

4.A.6 Semi-Online Attacks

Liu and Shroff (2019) studies what they call the offline setting for adversarial attacks on stochastic bandits. They consider a setting where a bandit algorithm is successively updated with mini-batches of fixed size B. The attacker can tamper with some of the incoming mini-batches. More precisely, they can modify the context, the reward and even the arm that was pulled for any entry of the attacked mini-batches. The main difference between this type of attacks and the online attacks we considered in the main paper is that we do not assume that we can attack from the start of the learning process: the bandit algorithm may have already converged by the time we attack.

We can still study the cumulative cost for the attacker to change the mini-batch in order to fool a bandit algorithm to pull a target arm $a^\dagger$ (here we take $A^\dagger = \{a^\dagger\}$). Contrarily to Liu and Shroff (2019), we call this setting semi-online. We first study the impact of an attacker on LINUCB where we show that, by modifying only $(K-1)d$ entries from the batch $\mathcal{B}$, the attacker can force LINUCB to pull arm $a^\dagger$, $M'B - o(M'B)$ times with M' the number of remaining batches updates. The cost of our attack is $\sqrt{MB}$ with M the total number of batches.

Cost of an attack: If presented with a mini-batch $\mathcal{B}$, with elements (x_t, a_t, r_t) composed of the context x_t presented at time t, the action taken a_t and the reward received r_t, the attacker modifies element i, namely (x_t^i, a_t^i, r_t^i) into $(\tilde{x}_t^i, \tilde{a}_t^i, \tilde{r}_t^i)$. The cost of doing so is $c_t^i = ||x_t^i - \tilde{x}_t^i||_2 + \left|\tilde{r}_t^i - r_t^i\right| + \mathbb{1}_{\{a_t^i \neq \tilde{a}_t^i\}}$ and the total cost for mini-batch $\mathcal{B}$ is defined as $c_{\mathcal{B}} = \sum_{i\in\mathcal{B}} c_t^i$. Finally, we consider the cumulative cost of the attack over M different mini-batches $\mathcal{B}_1, \dots, \mathcal{B}_M$, $\sum_{l=1}^M c_{\mathcal{B}_l}$. The interaction between the environment, the attacker and the learning algorithm is summarized in Alg. 33.

Algorithm 33: Semi-Online Attack Setting.

Input: Bandit alg. $\mathfrak{A}$, size of a mini-batch: B
Set $t = 0$
while *True* **do**
  $\mathfrak{A}$ observe context x_t
  $\mathfrak{A}$ pulls arm a_t and observes reward r_t
  Interaction (x_t, a_t, r_t) is saved in mini-batch $\mathcal{B}$
  if $\left|\mathcal{B}\right| = B$ **then**
    Attacker modifies mini-batch $\mathcal{B}$ into $\tilde{\mathcal{B}}$
    Update alg. $\mathfrak{A}$ with poisoned mini-batch $\tilde{\mathcal{B}}$

The attack presented here is based on the Ahlberg–Nilson–Varah bound Varah (1975), which gives a control on the sup norm of a matrix with dominant diagonal elements. More precisely, when presented with a mini-batch $\mathcal{B}$, the attacker needs to modify the contexts and the rewards. We assume that the attacker knows the number of mini-batch updates M and has access to a lower-bound on the reward of the target arm, ν as in Assumption 14.

The attacker changes $(K-1)\times d$ rows of the first mini-batch to rewards of 0 with a context $\delta_a e_i$ for each arm $a \neq a^\dagger$ with (e_i) the canonical basis of $\mathbb{R}^d$. Moreover, δ_a is chosen such that:

$$\delta_a > \max\left(\sqrt{\frac{2MBL^2d}{\nu} + dMB}, \sqrt{\frac{4\beta_{max}^2 L^2 d}{\nu^2} + dMB}\right) \tag{4.22}$$

with $\beta_{max} = \max_{t=0}^{MB} \beta_a(t)$ and M the number of mini-batch updates.

Proposition 4. *After the first attack, with probability* $1-\delta$, LINUCB *always pulls arm* $a^\dagger$,

Proof. After having poisoned the first mini-batch $\mathcal{B}$, the latter can be partitioned into two subsets, $\mathcal{B}_c$ (with non-perturbed rows) and $\mathcal{B}_{nc}$ (with the poisoned rows). The design matrix of arm $a \neq a^\dagger$ for every time t after the poisoning is:

$$V_{t,a} = \lambda I_d + \sum_{l=1, a_l=a}^{t} x_l x_l^\intercal + \delta_a^2 \sum_{i=1}^{d} e_i e_i^\intercal \tag{4.23}$$

For every time t, non diagonal elements of $V_{t,a} = (v_{i,j})_{i,j}$ are bounded by:

$$\forall i, r_i := \sum_{j \neq i} v_{i,j} \leq \sum_{j \neq i} \sum_{l=1, a_l=a}^{t} \|x_l x_l^\intercal\|_\infty \leq dN_a(kB) \tag{4.24}$$

Whereas for all diagonal elements, $v_{i,i} \geq \delta_a^2 > r_i$. Thus $V_{t,a}$ is strictly diagonal dominant and by the Ahlberg–Nilson–Varah bound Varah (1975):

$$\|V_{t,a}^{-1}\|_\infty \leq \frac{1}{\min_i \left(\|v_{i,i}\| - r_i\right)} \leq \frac{1}{\delta_a^2 - dMB} \tag{4.25}$$

Then, for every arm $a \neq a^\dagger$ and any context $x \in \mathcal{D}$ and any time t after the attack:

$$\begin{aligned}
\langle \hat{\theta}_{a,t}, x \rangle + \beta_a(t) ||x||_{V_{t,a}^{-1}} &\leq \sum_{l=1, a_l=a}^{t} r_t (V_{t,a}^{-1} x_t)^\intercal x + \beta_a(t) ||x||_1 \sqrt{\|V_{t,a}^{-1}\|_\infty} \\
&\leq \|V_{t,a}^{-1}\|_\infty dN_t(a) \sup_{y \in \mathcal{D}} ||y||_2^2 + \beta_{\max} \sqrt{d} \sup_{y \in \mathcal{D}} ||y||_2 \sqrt{\|V_{t,a}^{-1}\|_\infty} < \nu
\end{aligned}$$

We have shown that for any arm $a \neq a^\dagger$ and any time step t after the attack, the upper confidence bound computed by LinUCB is upper-bounded bu ν the arm $a^\dagger$. Then, with probability $1 - \delta$, the confidence set for arm $a^\dagger$ holds and, for all $x \in \mathcal{D}$, arm $a^\dagger$ is chosen by LinUCB. The total cost of this attack is $d \sum_{a \neq a^\dagger} \delta_a L = O(\sqrt{MB})$ □

4.B Appendix for Encrypted Linear Contextual Bandits

4.B.1 Slow-Switching Algorithm

In this subsection , we present the detailed algorithm of Sec. 4.2.3.

Algorithm 34: Low-Switching HELBA (Server-Side)

Input: horizon: T, regularization factor: λ, failure probability: δ, feature bound: L, $\theta^\star$ norm bound: S, dimension: d, batch growth: η, trace condition: C

Set $w_1 = Enc_{\mathsf{pk}}(0)$, $\Lambda_1 = Enc_{\mathsf{pk}}(\lambda I)$, $\bar{A}_1 = Enc_{\mathsf{pk}}(\lambda^{-1} I)$, $\check{V}_1 = Enc_{\mathsf{pk}}(\lambda I)$, $\check{y}_0 = 0$, $j = 0$ and $t_0 = 1$

for $t = 1, \ldots, T$ **do**

  Set $\widetilde{\beta}(t) = \sigma \sqrt{d \ln\left(\left(1 + \frac{L^2 t_j}{\lambda}\right)\left(\frac{\pi^2 t^2}{6\delta}\right)\right)} + t_j^{-1/2} + S\sqrt{\lambda}$ and $\epsilon_j = L(t_j^{3/2}\sqrt{\lambda + L^2 t_j})^{-1}$

  Observe encrypted contexts $(x_{t,a})_{a \in [K]} = (\mathit{Enc}_{\mathsf{pk}}(s_{t,a}))_{a \in [K]}$

  for $a = 1, \ldots, K$ **do**

    Compute approximate square root $\mathrm{sqrt}_{\mathrm{HE}}\left(x_{t,a}^\top \bar{A}_j x_{t,a} + \varepsilon_j\right)$

    Compute encrypted indexes $\rho_a(t) = \langle x_{t,a}, w_j \rangle + \widetilde{\beta}(t)\left(\mathrm{sqrt}_{\mathrm{HE}}\left(x_{t,a}^\top \bar{A}_j x_{t,a} + \varepsilon_j\right) + t^{-1}\right)$ (Step ❷)

    Rescale encrypted indexes $\widehat{\rho}_a(t) = \frac{\rho_a(t) - r_{\min}}{\rho_{\max} - r_{\min}}$ with $\rho_{\max} = r_{\max} + 2\widetilde{\beta}(t)\left[\frac{2}{t} + \frac{L}{t^{3/2}\sqrt{\lambda + L^2 t}} + L^2\left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}\right)\right]$

  Compute comparison vector $b_t \in \mathbb{R}^K$ using **acomp** (see Alg. 39 in App. 4.B.4.5) with precision $\varepsilon_t' = (4.1t)^{-1}$ (Step ❸)

  Observe encrypted reward y_t and encrypted context x_{t,a_t}

  Update $\check{V}_{t+1} = \check{V}_t + x_{t,a_t} x_{t,a_t}^\intercal$ and $\check{y}_{t+1} = \check{y}_t + y_t x_{t,a_t}$

  Compute Cond. (4.16) by computing δ_t with $\varepsilon = 0.45$ and $\varepsilon_t' = L^2(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}})(t - 1 - t_j))$ (see App. 4.B.5.1).

  Use masking procedure on δ_t (Alg. 40) and sends the masked ciphertext to the user

  if $\delta_t \geq 0.45$ ***or*** $t \geq (1 + \eta) t_j$ **then**

    Set $t_{j+1} = t$, $j = j + 1$ and $\Lambda_{j+1} = \check{V}_t$

    Compute $\bar{A}_{j+1} = X_{k_1(\varepsilon_{j+1}/L^2)}$ as in Prop. 19 ($V = \Lambda_{j+1}$, $c = \lambda d + L^2 t_{j+1}$) and $w_{j+1} = \bar{A}_{j+1} \check{y}_{t_{j+1}}$

4.B.2 Additional Related Work

In Federated Learning (a.k.a., collaborative multi-agent), DP and LDP guarantees can provide a higher level of privacy at a small regret cost, leveraging collaboration between users Wang et al. (2020); Zhu et al. (2020). Another collaborative approach to privacy-preserving machine learning, called Secure Multi-Party Computation (MPC) (e.g. Damgård et al., 2012), divides computations between parties, while guarantying that it is not possible for any of them to learn anything about the others. This has been recently empirically investigated in the bandit framework Hannun et al. (2019). However, there is an additional strong assumption, that each party provides a subset of the features observed at each round.

Finally, Homomorphic Encryption (HE) (e.g. Halevi, 2017) aims at providing a set of tools to perform computation on encrypted data, outsourcing computations to potentially untrusted third parties (in our setting the bandit algorithm) since data cannot be decrypted. HE has only been merely used to encrypt rewards in bandit problems Ciucanu et al. (2019), but in some inherently simpler setting: i) contexts are not considered and arms' features are not encrypted; ii) a trusted party decrypts data. In particular, the second point makes algorithm design much easier but requires users to trust the third party which, in turn, can lead again to privacy/security concerns. In the supervised learning literature, HE has been used to train neural networks (Badawi et al., 2020) achieving 77.55% classification accuracy on CIFAR-10 (compared to a state-of-the-art accuracy of 96.53% (Graham, 2015)) highlighting the potentially high impact of the approximation error due to HE.

4.B.3 Protocol Details

The learning algorithm may try to break encryption by inferring a mapping between ciphertexts and values or by storing all data. HE relies on the hardness of the *Learning With Error* problem (Albrecht et al., 2015) to guarantee security. To break an HE scheme, an attacker has to perform at least 2^κ operations to be able to differentiate noise from messages in a given ciphertext. We refer to (Albrecht et al., 2018) for a survey on the actual number of operations needed to break HE schemes with most of the known attacks. Although collecting multiple ciphertexts may speedup some attacks, the security of any HE scheme is still guaranteed as long as long the number of ciphertexts observed by an attacker is polynomial in N (Regev, 2009).

4.B.3.1 CKKS Encryption Scheme

In this subsection , we introduce the CKKS scheme Cheon et al. (2017). This scheme is inspired by the BGV scheme Brakerski et al. (2014) but has been modified to handle the encryption of real numbers. The security of those schemes relies on the assumption of hardness of the Learning With Errors (LWE), ring-LWE (RLWE) Regev (2009). The scheme can be divided into 2 parts: encoding/decoding and encryption/decryption.

Encoding and Decoding of Messages. In CKKS, the space of message is defined as $\mathbb{C}^{N/2}$ for some big even integer $N \in \mathbb{N}$. This integer is a parameter of the scheme chosen when generating the private and secret keys. CKKS scheme does not work directly on the space $\mathbb{C}^{N/2}$ but rather on an integer polynomial ring $\mathcal{R} = \mathbb{Z}[X]/\left(X^N + 1\right)$ (the plaintext space) Seidenberg (1978). Encoding a message $m \in \mathbb{C}^{N/2}$ into the plaintext space $\mathcal{R}$ is not as straightforward as using a classical embedding of a vector into a polynomial because we need the coefficients of the resulting polynomial to be integers. To solve this issue the CKKS scheme use a more sophisticated construction that the canonical embedding, based on the subring $\mathbb{H} = \{z \in \mathbb{C}^N \mid z_j = \bar{z}_{N-j}, j \leq N/2\}$ which is isomorphic to $\mathbb{C}^{N/2}$. Finally, using a canonical embedding $\sigma : \mathcal{R} \to \sigma(\mathcal{R}) \subset \mathbb{H}$ and the *coordinate-wise random rounding* technique developed in Lyubashevsky et al. (2013b), the CKKS scheme is able to construct an isomorphism between $\mathbb{C}^{N/2}$ and $\mathcal{R}$.

Encryption and Decryption of Ciphertexts. Most public key scheme relies on the hardness of the *Learning with Error* (LWE) problem introduced in Regev (2009). The LWE problem consists in distinguishing between noisy pairs $(a_i, \langle a_i, s\rangle + e_i)_{i\leq n} \subset (\mathbb{Z}/q\mathbb{Z})^n \times \mathbb{Z}/q\mathbb{Z}$ and uniformly sampled pairs in $(\mathbb{Z}/q\mathbb{Z})^n \times \mathbb{Z}/q\mathbb{Z}$ where $(e_i)_{i\leq n}$ are random noises and $q \in \mathbb{N}$. However, building a cryptographic public key system based on LWE is computationally inefficient. That's why CKKS relies on the *Ring Learning with Error* (RLWE) introduced in Lyubashevsky et al. (2013a) which is based on the same idea as LWE but working with polynomials $\mathbb{Z}_q[X]/(X^N + 1)$ instead on integer in $\mathbb{Z}/q\mathbb{Z}$. RLWE (and LWE) problem are assumed to be difficult to solve and are thus used as bases for cryptographic system. The security of those problems can be evaluated thanks to Albrecht et al. (2015) which gives practical bounds on the number of operations needed for known attacks to solve the LWE (RLWE) problem.

The CKKS scheme samples a random s on $\mathcal{R}$ and defines the secret key as $\mathbf{sk} = (1, s)$. It then samples a vector a uniformly on $\mathcal{R}/q_L\mathcal{R}$ (with $q_L = 2^L q_0$ where L is the depth of the scheme and q_0 its modulus) and an error term e sampled on $\mathcal{R}$ (usually each coefficient is drawn from a discrete Gaussian distribution). The public key is then defined as $\mathbf{pk} = (a, -a.s + e)$. Finally, to encrypt a message $m \in \mathbb{C}^{N/2}$ identified by a plaintext $\mathfrak{m} \in \mathcal{R}$ the scheme samples an encrypting noise $\nu \sim \mathcal{ZO}(0.5)$[6]. The scheme then samples $e_0, e_1 \in \mathbb{Z}^N$ two independent random variable from any distribution on $\mathcal{R}$, usually a discrete Gaussian distribution. The ciphertext associated to the message m is then $[(\nu \cdot \mathbf{pk} + (\mathfrak{m} + e_0, e_1))]_{q_L}$ with $[.]_{q_L}$ the modulo operator

[6] A random variable $X \sim \mathcal{ZO}(0.5)$ such that $X \in \{0, 1, -1\}^N$, $(X_i)_{i\leq N}$ are i.i.d such that for all $i \leq N$ $\mathbb{P}(X_i = 0) = 1/2$,$\mathbb{P}(X_i = 1) = 1/4$ and $\mathbb{P}(X_i = -1) = 1/4$

and $q_L = 2^L$. Finally, to decrypt a ciphertext $c = (c_0, c_1) \in \mathcal{R}_{q_l}^2$ (with l the level of the ciphertext, that is to say the depth of the ciphertext), the scheme computes the plaintext $\mathfrak{m}' = [c_0 + c_1 s]_{q_l}$[7] and returns the message m' associated to the plaintext $\mathfrak{m}'$.

4.B.3.2 Key Switching

Homomorphic Encryption schemes needs all ciphertexts to be encrypted under the same public key in order to perform additions and multiplications. As we mentioned in Sec. 4.2.5 one way to circumvent this issue is to use a *KeySwitching* operation. The *KeySwitching* operation takes as input a cyphertext c_1 encrypted thanks to a public key pk_1 associated to a secret key sk_1 and transform it into a cyphertext encrypting the same message as c_1 but under a different secret key sk_2.

The exact *KeySwitching* procedure for each scheme is different. We will use the CKKS scheme, inspired by the BGV scheme Brakerski et al. (2014), where *KeySwitching* relies on two operations BitDecomp and PowerOf2, described below,

1. BitDecomp(c, q) takes as input a ciphertext $c \in \mathbb{R}^N$ with m the size of the ring dimension used in CKKS and an integer q. This algorithm decomposes c in its bit representation $(u_0, \dots, u_{\lceil \log_2(q) \rceil}) \in \mathbb{R}^{N \times \lceil \log_2(q) \rceil}$ such that $c = \sum_{j=0}^{\lfloor \log_2(q) \rfloor} 2^j u_j$
2. PowerOf2(c, q) takes as input a ciphertext $c \in \mathbb{R}^N$ and an integer q. This algorithm outputs $(c, 2c, \dots, 2^{\lfloor \log_2(q) \rfloor} c) \in \mathbb{R}^{m \times \lceil \log_2(q) \rceil}$

The *KeySwitching* operation can then be decomposed as:

- the first party responsible for sk_1 generates a new (bigger, in the sense that the parameter N is bigger than sk_1) public key $\tilde{\mathsf{pk}}_1$ still associated to sk_1
- the owner of secret key sk_2 computes PowerOf2(sk_2) and add it to $\tilde{\mathsf{pk}}_1$. This object is called the *KeySwitchingKey*.
- the new cyphertext is computed by mulitiplying BitDecomp(c_1) with the KeySwitchingKey. This gives a new cyphertext decryptable with the secret key sk_2 and encrypted under a new public key pk_2

Algorithm 35: KeySwitching Procedure

Input: Cyphertext: c, User: u, User public key/secret key: $\mathsf{pk}_u, \mathsf{sk}_u$, Bandit Algorithm: $\mathfrak{A}$, Trusted Third Party: $\mathfrak{B}$, integer q

Alg. $\mathfrak{A}$ receives cypthertext c encrpyted with key pk_u
$\mathfrak{B}$ sends public key **pk** to u
u computes $\mathsf{Enc}_{\mathbf{pk}_u}(\mathbf{ksk}_u) = \mathsf{Enc}_{\mathbf{pk}_u}(\text{PowerOf2}(\mathbf{sk}_u, q) + \mathbf{pk})$
u sends $\mathsf{Enc}_{\mathbf{pk}_u}(\mathbf{ksk}_u)$ to $\mathfrak{A}$
$\mathfrak{A}$ computes the new cyphertext $c' = \mathsf{Enc}_{\mathbf{pk}_u}(\text{BitDecomp}(c, q)^\intercal)\mathsf{Enc}_{\mathbf{pk}_u}(\mathbf{ksk}_u) = \mathsf{Enc}_{\mathbf{pk}_u}(\mathsf{Enc}_{\mathbf{pk}}(c))$
u decrypts c' and sends the result to $\mathfrak{A}$

Alg. 35 allows us to perform the *KeySwitching* in a private manner for the CKKS scheme. Indeed, the key switch operation requires to decompose a secret key thanks to the PowerOf2 procedure. If not done in a secure fashion this could lead to a leak of the frist private key. It is thus necessary to ensure that this key is not distributed in the clear. However, our private procedure requires communication between the bandit algorithm $\mathfrak{A}$ and the user u. In particular, the user still needs to receives the public key from the trusted third party. However, the user does not need to be known ahead of time as previously.

4.B.4 Toward An Encrypted OFUL

In this subsection , we provide the proof of the results of Step ❶, ❷ and ❸, i.e., the speed of convergence of iterating Eq. (4.10) or Eq. (4.13), how to build a confidence intervals around $\theta^\star$ and how the approximate argmax is computed in Alg. 28.

4.B.4.1 Computing an Approximate Inverse

First, we prove Prop. 19. The proof of convergence the Newton method for matrix inversion is rather standard but the proof of convergence for the stable method (Eq. (4.10)) is often not stated. We derive it here for completeness. First, we recall Prop. 19.

Proposition. *Given a symmetric positive definite matrix $V \in \mathbb{R}^{d \times d}$, $c \geq Tr(V)$ and a precision level $\varepsilon > 0$, the iterate in (4.10) satisfies*

$$\|X_k - V^{-1}\| \leq \varepsilon$$

for any $k \geq k_1(\varepsilon)$ with

$$k_1(\varepsilon) = \frac{1}{\ln(2)} \ln\left(\frac{\ln(\lambda) + \ln(\varepsilon)}{\ln\left(1 - \frac{\lambda}{c}\right)} \right)$$

, where $\lambda \leq \lambda_d$ is a lower bound to the minimal eigenvalue of V and $\|\cdot\|$ is the matrix spectral-norm.

[7] for any $n \in \mathbb{N}$, $[.]_n$ is the remainder of the division by n

Proof. of Prop. 19. After k iterations of Eq. (4.10), we have that $VX_k = M_k$. Indeed we proceed by induction:

- For $k=0$, $M_0 = \frac{1}{c}V = VX_0$
- For $k+1$ given the property at time k, $VX_{k+1} = VX_k(2I_d - M_k) = M_k(2I_d - M_k) = M_{k+1}$

Let's note $E_k = X_k - V^{-1}$ and $\tilde{E}_k = M_k - I_d$ then:

$$\begin{aligned} E_{k+1} = \left(X_{k+1}V - I_d\right)V^{-1} &= \left(M_{k+1} - I_d\right)V^{-1} \\ &= -\left(M_k^2 - 2M_k + I_d\right)V^{-1} \\ &= -\left(M_k - I_d\right)^2 V^{-1} = -\tilde{E}_k^2 V^{-1} \end{aligned}$$

where the second equality is possible because V and $(X_k)_{k\in\mathbb{N}}$ commute as for all $k \in \mathbb{N}$, X_k is a polynomial function of V.

Therefore, we have for any $k \in \mathbb{N}$:

$$\|E_{k+1}\| = \|\tilde{E}_k^2 V^{-1}\| \leq \|V^{-1}\| \times \|\tilde{E}_k\|^2 \tag{4.26}$$

But at the same time:

$$\|\tilde{E}_{k+1}\| = \|M_{k+1} - I_d\| = \|M_k(2I_d - M_k) - I_d\| = \|-\left(M_k - I_d\right)^2\| \leq \|\tilde{E}_k\|^2 \tag{4.27}$$

thus iterating Eq. (4.27), we have that for all $k \in \mathbb{N}$, $\|\tilde{E}_k\| \leq \|\tilde{E}_0\|^{2^k}$. And then $\|\tilde{E}_k\| \leq \|\tilde{E}_0\|^{2^k}\|V^{-1}\|$, therefore using that any V symmetric definite positive $\|V^{-1}\| = \|V\|^{-1}$ then for all $k \in \mathbb{N}$:

$$\|E_k\| \leq \left\|\frac{V}{c} - I_d\right\|^{2^k} \|V\|^{-1} \tag{4.28}$$

But $\|\tilde{E}_0\| = \left\|\frac{1}{c}V - I_d\right\| = \max_{i\in[d]}\left|\frac{\lambda_i}{c} - 1\right|$ where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d \geq 0$ are the (ordered) eigenvalues of V. However $c \geq \mathrm{Tr}(V)$ thus $0 \leq \lambda_i/c \leq 1$ for all $i \leq d$. Therefore $\|\tilde{E}_0\| \leq 1 - \frac{\lambda_d}{c}$. We also have that $\|V\| = \lambda_1$. Using Eq. (4.28), we have for all k:

$$\|E_k\| \leq \left(1 - \frac{\lambda_d}{c}\right)^{2^k} \lambda_1^{-1} \leq \left(1 - \frac{\lambda}{c}\right)^{2^k} \lambda^{-1} \tag{4.29}$$

for any $0 \geq \lambda \leq \lambda_d$. Finally, Eq. (4.29) implies that $\|E_k\| \leq \varepsilon$ as soon as:

$$k \geq \frac{1}{\ln(2)} \ln\left(\frac{\ln(\lambda) + \ln(\varepsilon)}{\ln\left(1 - \frac{\lambda}{c}\right)}\right) \tag{4.30}$$

for any $0 \geq \lambda \leq \lambda_d$ and $\lambda\varepsilon \leq 1$. □

4.B.4.2 Computing an Approximate Square Root

The proof of Prop. 20 is very similar to the proof of Prop. 19 thanks the analysis of the convergence speed in Cheon et al. (2019). First, let us recall Prop. 20.

Proposition. *For any $z \in \mathbb{R}_+$, $c_1, c_2 > 0$ with $c_2 \geq z \geq c_1$ and a precision $\varepsilon > 0$, let q_k be the result of k iterations of Eq. (4.13), with $q_0 = \frac{z}{c_2}$ and $v_0 = \frac{z}{c_2} - 1$. Then, $|q_k\sqrt{c_2} - \sqrt{z}| \leq \varepsilon$ for any $k \geq k_0(\varepsilon) := \frac{1}{\ln(2)} \ln\left(\frac{\ln(\varepsilon) - \ln\left(\sqrt{c_2}\right)}{4\ln\left(1 - \frac{c_1}{4c_2}\right)}\right)$.*

Proof. of Prop. 20. Because $0 \leq c_1 < x < c_2$, we have that $\frac{x}{c_2} \in (0,1)$, hence thanks to Lemma 2 of Cheon et al. (2019), we have that after k iterations:

$$\left|q_k - \sqrt{\frac{x}{c_2}}\right| \leq \left(1 - \frac{x}{4c_2}\right)^{2^{k+1}} \tag{4.31}$$

where q_k is the k-th iterate from iterating Eq. (4.13) with $q_0 = \frac{x}{c_2}$ and $v_0 = q_0 - 1$. Then because $x \geq c_1$, we have that $1 - \frac{x}{4c_2} \leq 1 - \frac{c_1}{4c_2}$. Stated otherwise,

$$\left|q_k - \sqrt{\frac{x}{c_2}}\right| \leq \left(1 - \frac{c_1}{4c_2}\right)^{2^{k+1}} \tag{4.32}$$

Therefore, for $k \geq \frac{1}{\ln(2)} \ln\left(\frac{\ln(\varepsilon) - \ln\left(\sqrt{c_2}\right)}{2\ln\left(1 - \frac{c_1}{4c_2}\right)}\right)$, the result follows since:

$$\sqrt{c_2}\left|q_k - \sqrt{\frac{x}{c_2}}\right| \leq \varepsilon \tag{4.33}$$

□

4.B.4.3 Computing an Optimistic Ellipsoid Width.

The next step to build an optimistic algorithm is to compute a confidence ellipsoid around the estimate $\widetilde{\theta}_t$ such that the true parameter $\theta^\star$ belongs to this confidence ellipsoid with high probability. First, we need an estimate of the distance between $\theta^\star$ and $\widetilde{\theta}_t$ that is the object of Cor. 3. The proof of Cor. 3, is based on the fact that the approximated inverse is closed enough to the true inverse. Let's recall Cor. 3 first.

Corollary. *Setting* $\varepsilon_t = \left(Lt^{3/2}\sqrt{L^2t+\lambda}\right)^{-1}$ *in Prop. 19, then* $\|\mathit{Dec}_{\mathit{sk}}(\omega_t) - \theta_t\|_{V_t} \leq t^{-1/2}$, $\forall t$.

Proof. of Cor. 3. Let's note $\bar{A}_t$, the result of iterating Eq. (4.10), $k_1(\varepsilon_t)$ times with $V = V_t$ and $c = \lambda d + L^2 t$. Thanks to the definition of $\mathsf{Dec}_{\mathsf{sk}}(w_t)$ and $\theta_t = V_t^{-1} b_t$, we have:

$$\|\mathsf{Dec}_{\mathsf{sk}}(w_t) - \theta_t\|_{V_t} = \left\| V_t^{1/2}\left(V_t^{-1} - \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_t)\right)\sum_{l=1}^{t-1} r_l s_{l,a_l}\right\|_2 \tag{4.34}$$

$$= \left\| \left(V_t^{-1} - \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_t)\right) V_t^{1/2}\sum_{l=1}^{t-1} r_l s_{l,a_l}\right\|_2 \tag{4.35}$$

$$\leq \|\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_t) - V_t^{-1}\| \left\| V_t^{1/2}\sum_{l=1}^{t-1} r_l s_{l,a_l}\right\|_2 \tag{4.36}$$

But $\mathrm{Tr}(V_t) \leq \lambda d + L^2 t$ and $\lambda_{\min}(V_t) \geq \lambda$. Therefore thanks to Prop. 19 $\bar{A}_t$ is such that:

$$\|\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_t) - V_t^{-1}\| \leq \varepsilon_t \tag{4.37}$$

We also have that:

$$\left\| V_t^{1/2}\sum_{l=1}^{t-1} r_l s_{l,a_l}\right\|_2 \leq \|\sqrt{V_t}\| \left\|\sum_{l=1}^{t-1} r_l s_{l,a_l}\right\|_2 \tag{4.38}$$

$$\leq Lt\sqrt{\|V_t\|} \leq Lt\sqrt{\lambda + L^2 t} \tag{4.39}$$

because $r_l \in [-1, 1]$ for all $l \leq t$ and $\lambda_{\max}(V_t) \leq \lambda + L^2 t$. Finally, we have that:

$$\|\theta_t - \tilde{\theta}_t\|_{V_t} \leq \varepsilon_t Lt\sqrt{\lambda + L^2 t} \leq t^{-1/2} \tag{4.40}$$

□

4.B.4.4 Approximate Confidence Ellipsoid

Finally thanks to Cor. 3, we can now prove that with high probability $\theta^\star$ belongs to the inflated confidence intervals $\tilde{\mathcal{C}}_t$ for all time t. That is the object of Prop. 23.

Proposition 23. *For any* $\delta > 0$, *we have that with probability at least* $1 - \delta$*:*

$$\theta^\star \in \bigcap_{t=1}^{+\infty} \mathcal{C}_t(\delta) := \left\{\theta \mid \|\theta - \mathit{Dec}_{\mathit{sk}}(w_t)\|_{V_t} \leq \widetilde{\beta}(t)\right\} \tag{4.41}$$

with $\widetilde{\beta}(t) = t^{-1/2} + \sqrt{\lambda}S + \sigma\sqrt{d(\ln(1 + L^2 t/(\lambda d)) + \ln(\pi^2 t^2/(6\delta))}$

Proof. of Prop. 23. Using Cor. 3 and Thm. 2 in Abbasi-Yadkori et al. (2011), we have that for any time t that with probability at least $1 - \delta$:

$$\|\theta^\star - \mathsf{Dec}_{\mathsf{sk}}(w_t)\|_{V_t} \leq \|\theta_t - \mathsf{Dec}_{\mathsf{sk}}(w_t)\|_{V_t} + \|\theta^\star - \theta_t\|_{V_t} \tag{4.42}$$

$$\leq t^{-1/2} + \sqrt{\lambda}S + \sigma\sqrt{d(\ln(1 + L^2 t/(\lambda d)) + \ln(1/\delta))} \tag{4.43}$$

where w_t computed as in Alg. 34 and θ_t is the ridge regression estimate computed at every time step in OFUL. Taking a union bound with high-probability event means that with probability at least $1 - \frac{6\delta}{\pi^2}$, we have:

$$\|\theta^\star - \theta_t\|_{V_t} \leq \|\theta_t - \mathsf{Dec}_{\mathsf{sk}}(w_t)\|_{V_t} + \|\theta^\star - \theta_t\|_{V_t} \tag{4.44}$$

$$\leq t^{-1/2} + \sqrt{\lambda}S + \sigma\sqrt{d(\ln(1 + L^2 t/(\lambda d)) + \ln(\pi^2 t^2/(6\delta)))} \tag{4.45}$$

□

4.B.4.5 Homomorphic Friendly Approximate Argmax

As mentioned in Sec. 4.2.3, an homomorphic algorithm can not directly compute the argmax of a given list of values. In this work, we introduce the algorithm Alg. 39 to compute the comparison vector $b_t \approxeq \left(\mathbb{1}_{\{a=\arg\max_{i\in[K]}\rho_i(t)\}}\right)$ with $(\rho_a(t))_{a\in[K]}$ the UCBs defined in Sec. 4.2.3. This algorithm is divided in two parts. First, it computes an approximate maximum, M of $(\rho_a(t))_{a\in[K]}$ thanks to Alg. 38 and then compares each values $(\rho_a(t))_{a\in[K]}$ to this approximate maximum M thanks to the algorithm **NewComp** of Cheon et al. (2020) (recalled as Alg. 36).

Algorithm 36: NewComp

Input: Entry numbers: $a, b \in [0,1]$, n and depth d
Set $x = a - b$
for $k = 1, \dots, d$ **do**
  Compute $x = f_n(x) = \sum_{i=0}^{n} \frac{1}{4^i}\binom{2i}{i} x(1-x^2)^i$
Output: $(x+1)/2$

Algorithm 37: NewMax

Input: Entry numbers: $a, b \in [0,1]$, n and depth d
Set $x = a - b$, $y = \frac{a+b}{2}$
for $k = 1, \dots, d$ **do**
  Compute $x = f_n(x) = \sum_{i=0}^{n} \frac{1}{4^i}\binom{2i}{i} x(1-x^2)^i$
Output: $y + \frac{a+b}{2} \cdot x$

Algorithm 38: amax

Input: Entry numbers: $(a_i)_{i\le K}$, n and depth d
Set $m = a_1$
for $i = 2, \dots, K$ **do**
  Compute $m = \max\{m, a_i\}$ thanks to **NewMax** in Cheon et al. (2020) with parameter $a = m$, $b = a_i$, n and d

Algorithm 39: acomp

Input: Entry numbers: $(a_i)_{i\le K}$, precision ε
Set depth $d = 1 + \left\lfloor 3.2 + \frac{\ln(1/\varepsilon)}{\ln(3/2)} + \frac{\ln\left(\frac{\ln(1/\varepsilon)}{\ln(2)} - 2\right)}{\ln(2)} \right\rfloor$ and depthmax $d' = \frac{1}{\ln(3/2)} \ln\left(\frac{\alpha \ln\left(\frac{1}{\varepsilon}\right)}{\ln(2)} - 2\right)$ with $\alpha = \frac{3}{2} + \frac{5.2\ln(3/2)}{\ln(4)} + \frac{\ln(3/2)}{2\ln(2)}$
Compute $M = \text{amax}((a_i)_{i\le K}, n, d)$
for $i = 2, \dots, K$ **do**
  Set $b_i = \textbf{NewComp}(a_i, M, n, d')$

Rescaling the UCB index: In order to use the HE-friendly algorithms of Cheon et al. (2020), we need to rescale the UCB-index to lies in $[0,1]$. Determining the range of those indexes is the purpose of the following proposition.

Proposition 24. *For every time $t \ge 1$, assuming $r_l \in [-1,1]$ for any $l \le t$ and $L \ge 1$ then for any $\delta > 0$ we have that with probability at least $1-\delta$:*

$$-1 \le \rho_a(t) \le 1 + 2\widetilde{\beta}(t)\left[2t^{-1} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}} + \sqrt{\frac{L}{t^{3/2}\sqrt{\lambda} + L^2 t}}\right] \tag{4.46}$$

where $\rho_a(t) = \langle \widetilde{\theta}_t, x_{t,a}\rangle + \widetilde{\beta}(t)\left[q_{k_0(t^{-1})} + t^{-1}\right]$ the UCB index of arm a at time t.

Proof. of Prop. 24. For $\delta > 0$, we denote $E = \bigcap_{l=1}^{+\infty}\left\{\theta^\star \in \tilde{\mathcal{C}}_l(\delta)\right\}$ so that, using Prop. 23, $\mathbb{P}(E) \ge 1-\delta$. Under the event E, we have for any arm a:

$$-1 \le \langle x_{t,a}, \theta^\star\rangle \le \rho_a(t) \le \langle x_{t,a}, \theta^\star\rangle + 2\widetilde{\beta}(t)\left[q_{k_0(1/t)} + t^{-1}\right] \tag{4.47}$$

On the other hand thanks to Prop. 20, we have that $q_{k_0(t^{-1})} \le \sqrt{\|x\|^2_{\mathsf{Dec_{sk}}(A_t)} + \frac{L}{t^{3/2}\sqrt{\lambda+L^2 t}}} + t^{-1}$. and also $\|x\|^2_{A_t} \le L^2\left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}\right)$.

Indeed because $\mathsf{Dec_{sk}}(A_t)$ is a polynomial function of V_t, we have that $\mathsf{Dec_{sk}}(A_t)$ is symmetric and $\mathsf{Dec_{sk}}(A_t)V_t = V_t\mathsf{Dec_{sk}}(A_t)$, hence $\mathsf{Dec_{sk}}(A_t)$ and V_t^{-1} are diagonalizable in the same basis therefore $\|\mathsf{Dec_{sk}}(A_t) - V_t^{-1}\| = \max_{i\le d}|\lambda_i(\mathsf{Dec_{sk}}(A_t)) -$

$\lambda_i(V_t^{-1})|$ with $\lambda_i(M)$ the i-th biggest eigenvalue of M. Hence:

$$\lambda_1(\mathsf{Dec}_{\mathsf{sk}}(A_t)) \leq \frac{1}{\lambda} + \frac{1}{Lt^{3/2}\sqrt{\lambda + L^2 t}} \tag{4.48}$$

and:

$$\lambda_d(\mathsf{Dec}_{\mathsf{sk}}(A_t)) \geq \frac{1}{\lambda + L^2 t} - \frac{1}{Lt^{3/2}\sqrt{\lambda + L^2 t}} > 0 \tag{4.49}$$

for $t \geq 2$. Therefore, we have that for any arm a:

$$\rho_a(t) \leq \langle \theta^\star, x_{t,a} \rangle + 2\widetilde{\beta}(t) \left[2t^{-1} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}} + \sqrt{\frac{L}{t^{3/2}\sqrt{\lambda + L^2 t}}} \right] \tag{4.50}$$

□

Computing the Comparaison Vector: The algorithm Alg. 39 operates on values in $[0,1]$ therefore using Prop. 24, we can compute rescaled UCB index, noted $\tilde{\rho}_a(t) \in [0,1]$. We are then almost ready to prove Cor. 4, we just need two lemmas which relates the precision of Alg. 39 and Alg. 38 to the precision of **NewComp** and **NewMax** of Cheon et al. (2020).

The first lemma (Lem. 23) gives a lower bound on the depth needed for Alg. 38 to achieve a given precision.

Lemma 23. *For any sequences $(a_i)_{i \leq K} \in [0,1]^K$, for any precision $0 < \varepsilon < K/4$, $n \in \mathbb{N}^\star$ and*

$$d(\varepsilon, n) \geq \frac{\ln\left(\frac{\ln\left(\frac{K}{\varepsilon}\right)}{\ln(2)} - 2\right)}{\ln(c_n)} \tag{4.51}$$

with $c_n = \frac{2n+1}{4^n}\binom{2n}{n}$. Noting M the result of Alg. 38 with parameter $(a_i)_i$, n and $d(\varepsilon, n)$, we have that:

$$\left| M - \max_i a_i \right| \leq \varepsilon \tag{4.52}$$

Proof. of Lemma 23. Thanks to Corollary 4 in Cheon et al. (2020), we have that for any n and depth $d \geq \frac{\ln\left(\frac{\ln(1/\varepsilon)}{\ln(2)} - 2\right)}{\ln(c_n)}$ (with $c_n = \frac{2n+1}{4^n}\binom{2n}{n}$) and number a, b:

$$|\mathbf{NewMax}(a, b, n, d) - \max\{a, b\}| \leq \varepsilon \tag{4.53}$$

Let's note m_k the iterate m of Alg. 38 at step $k \in [K]$ in the for loop. We show that by induction $\left| m_k - \max_{i \in [k]} a_i \right| \leq k\varepsilon$.

- By definition $m_1 = a_1$ and $|m_1 - \max_{i \leq 1} a_i| = 0$
- Using that $|\max\{a, c\} - \max\{b, c\}| \leq |a - b|$ for any $a, b, c \in \mathbb{R}$, we have:

$$\begin{aligned}
\left| m_{k+1} - \max_{i \leq k+1} a_i \right| &= \Big| \mathbf{NewMax}(m_k, a_{k+1}, n, d) - \max\{m_k, a_{k+1}\} \\
&\qquad + \max\{m_k, a_{k+1}\} - \max\{\max_{i \leq k} a_i, a_{k+1}\} \Big| \\
&\leq |\mathbf{NewMax}(m_k, a_{k+1}, n, d) - \max\{m_k, a_{k+1}\}| \\
&\qquad + |\max\{m_k, a_{k+1}\} - \max\{\max_{i \leq k} a_i, a_{k+1}\}| \\
&\leq \varepsilon + |m_k - \max_{i \leq k} a_i| \leq (k+1)\varepsilon
\end{aligned}$$

Finally, because $M = m_K$, we just need to choose $d \geq \frac{\ln\left(\frac{\ln(K/\varepsilon)}{\ln(2)} - 2\right)}{\ln(c_n)}$ to get the result. □

The next lemma (Lem. 24) has the same purpose of Lem. 23 but this time for Alg. 39. The proof is based on properties of the polynomial function used by the algorithm **NewComp** in order to predict the result of the comparison when the margin condition of **NewComp** (that is to say the result of the comparison of $a, b \in [0,1]$ is valid if and only if $|a - b| \geq \varepsilon$ for some $\varepsilon > 0$) is not satisfied.

Lemma 24. *For $\varepsilon \in (0, 1/4)$ and sequence $(a_i)_{i\leq K} \in [0,1]^K$, let's denote $(b_i)_{i\leq K}$ te result of Alg. 39 ruuned with parameter $(a_i)_{i\leq K}$, $n = 1$, $d' = d_2(\varepsilon)$ and $d = d_3(\varepsilon)$ with:*

$$d_2(\varepsilon) = \left\lfloor 3.2 + \frac{\ln(1/\varepsilon)}{\ln(c_n)} + \frac{\ln\left(\ln\left(\frac{1}{\varepsilon}\right)/\ln(2) - 2\right)}{\ln(n+1)} \right\rfloor + 1 \tag{4.54}$$

$$d_3(\varepsilon) \ \geq \frac{1}{\ln(c_n)} \ln\left(\frac{\alpha \ln\left(\frac{1}{\varepsilon}\right)}{\ln(2)} - 2\right) \tag{4.55}$$

where $\alpha = \frac{3}{2} + \frac{5.2\ln(c_n)}{\ln(4)} + \frac{\ln(c_n)}{2\ln(n+1)}$. Then selecting any $i \leq K$ such that $b_i \geq \varepsilon$ (and there is at least one such index i), we have that $a_i \geq \max_k a_k - 2\varepsilon$

Proof. of Lemma 24. Thanks to Corollary 1 in Cheon et al. (2019), we have that for each $i \leq K$, $|b_i - \mathsf{Comp}(a_i, M)| \leq \varepsilon$ as soon as $|a_i - M| > \varepsilon$ and $d' = \left\lfloor 3.2 + \frac{\ln(1/\varepsilon)}{\ln(c_n)} + \frac{\ln\left(\ln\left(\frac{1}{\varepsilon}\right)/\ln(2)-2\right)}{\ln(n+1)} \right\rfloor + 1$. For $i \in [K]$, we have that:

- If $\max_{k\leq K} a_k \geq a_i \geq M + \varepsilon$ then $\mathsf{Comp}(a_i, M) = 1$, $|b_i - 1| \leq \varepsilon$ and $a_i \geq \max_{k\leq K} a_k - |\max_{k\leq K} a_k - M| - \varepsilon$
- If $a_i \leq M - \varepsilon$ then $\mathsf{Comp}(a_i, M) = 0$, thus $|b_i| \leq \varepsilon$ and $a_i \leq \max_{k\leq K} a_k + |\max_{k\leq K} a_k - M| - \varepsilon$

Therefore for any a_i such that $|a_i - M| > \varepsilon$ then the resulting b_i is either bounded by $1 - \varepsilon$ or ε.

The second option is if $|a_i - M| \leq \varepsilon$ then the **NewComp** algorithm provides no guarantee to the result of the algorithm. However the algorithm applies a function f_n[8] multiple times to its input. For every $x \in [-1, 1]$:

$$|f_n(x)| \leq c_n|x| \text{ and } f_n([-1,1]) \subset [-1,1] \tag{4.56}$$

with $c_n = \frac{2n+1}{4^n}\binom{2n}{n}$. Hence:

$$\forall x \in [-1,1] \qquad |f_n^{(d')}(x)| \leq c_n|f_n^{(d'-1)}(x)| \leq c_n^{d"}|x| \tag{4.57}$$

But if $|a_i - M| \leq \varepsilon$, $f_n^{(d')}(a_i - M) \leq c_n^{d'}|a_i - M| \leq c_n^{d'}\varepsilon$ thus $\left|b_i - \frac{1}{2}\right| \leq \frac{c_n^{d'}\varepsilon}{2}$.

Finally for each i, we only three options for b_i:

- If $|a_i - M| \leq \varepsilon$ then $\left|b_i - \frac{1}{2}\right| \leq \frac{c_n^{d'}\varepsilon}{2}$ and $a_i \geq \max_k a_k \ - (\varepsilon + |\max_k a_k - M|) \geq \max_k a_k \ - 2\varepsilon$
- If $|a_i - M| \geq \varepsilon$ and $a_i \leq M - \varepsilon$ then $|b_i| \leq \varepsilon$ and $a_i \leq \max_k a_k + |M - \max_k a_k| - \varepsilon \leq \max_k a_k$
- If $|a_i - M| \geq \varepsilon$ and $a_i \geq M + \varepsilon$ then $|b_i - 1| \leq \varepsilon$ and $a_i \geq \max_k a_k - (|M - \max_k a_k| + \varepsilon) \geq \max_k a_k - 2\varepsilon$

To finish the proof, we just need to ensure that there exists at least one i such that $b_i \geq \varepsilon$. Noting $i^\star = \arg\max_k a_k$, if the amax algorithm is used with depth d such that:

$$d \geq \frac{1}{\ln(c_n)} \ln\left(\frac{\alpha \ln\left(\frac{1}{\varepsilon}\right)}{\ln(2)} - 2\right) \tag{4.58}$$

where $\alpha = \frac{3}{2} + \frac{5.2\ln(c_n)}{\ln(4)} + \frac{\ln(c_n)}{2\ln(n+1)}$, we have $|a_{i^\star} - M| \leq \varepsilon^\alpha \leq \varepsilon$ and $b_{i^\star} \geq \frac{1}{2} - \frac{c_n^{d'}\varepsilon^\alpha}{2} > \varepsilon$. Hence there always exists an index i such that $b_i \geq \varepsilon$. □

Finally, thanks to Lem. 24, we can finally prove Cor. 4. The proof of this corollary simply amounts to choose the right precision for **NewComp** algorithm at every step of Alg. 39. First let's recall Cor. 4.

Corollary. *For any time t, selecting any arm a such that $(b_t)_a \geq \frac{1}{4t}$ then:*

$$\rho_a(t) \geq \max_{k\leq K} \rho_k(t) - \frac{1}{t}\left(1 + \widetilde{\beta}^\star(t)\right) \tag{4.59}$$

where $\widetilde{\beta}^\star(t) = \widetilde{\beta}(t)\left[2t^{-1} + \sqrt{\frac{L}{t^{3/2}\sqrt{\lambda + L^2 t}}} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}}\right]$

Proof. of Cor. 4. Using Lem. 24 with $\varepsilon = \frac{1}{4t}$ yields the following result:

$$\tilde{\rho}_i(t) \geq \max_{k\leq K} \tilde{\rho}_k(t) - \frac{1}{2t} \tag{4.60}$$

But for any $i \leq K$, $\tilde{\rho}_i(t) = (\rho_i(t) + 1)/\left(2 + 2\widetilde{\beta}(t)\left[2t^{-1} + \sqrt{\frac{L}{t^{3/2}\sqrt{\lambda + L^2 t_j}}} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}}\right]\right)$. Hence the result. □

[8] For all $x \in [-1,1]$, $f_n(x) = \sum_{i=0}^n \frac{1}{4^i}\binom{2i}{i}x(1-x^2)^i$.

4.B.5 Slow Switching Condition and Regret of HELBA

In this appendix, we present the analysis of the regret of HELBA. The proof is decomposed in two steps. The first one is the analysis of the number of batches for any time T. That is the object of the Sec. 4.B.5.1. The second part of the proof amounts to bounding the regret as a function of the number of batches (Sec. 4.B.5.2).

4.B.5.1 Number of batches of HELBA (Proof of Prop. 22)

We first prove Prop. 22 which states that the total number of batches for HELBA is logarithmic in T contrary to HELBA where the parameter are updated a linear number of times. The proof of this proposition is itself divided in multiple steps. First, we show how using **NewComp** to compare the parameter C and $\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right)$ (for any batch j) relate to the comparison of C and $\mathsf{Tr}\left(\bar{V}_j^{-1}\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right)$. Then, we show how Condition 4.16 relates to the det-based condition used in RSOFUL which allows us to finish the proof of Prop. 22 following the same reasoning as in Abbasi-Yadkori et al. (2011).

Homomorphically Friendly Comparison for Condition 4.16 We first prove the following proposition, bounding the error made by our algorithm when using $\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right)$ instead of $\mathsf{Tr}\left(\bar{V}_j^{-1}\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right)$.

Proposition 25. *For an batch j, time $t \geq t_j+1$, $\varepsilon < 1/2$ and $\varepsilon' > 0$, let's note δ_t the result of **NewComp** applied with parameters $a = \frac{\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right)}{L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)}$, $b = \frac{C}{L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)}$* [9], *$n=1$ and $d_5(\varepsilon)$ such that:*

$$d_5(\varepsilon) \geq 3.2 + \frac{\ln(1/\varepsilon')}{\ln(c_n)} + \frac{\ln\left(\ln\left(\frac{1}{\varepsilon}\right)/\ln(2)-2\right)}{\ln(n+1)} \tag{4.61}$$

then:

- *if $\delta_t > \varepsilon$:*

$$C - \varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j) \leq \mathit{Tr}\left(\mathit{Dec}_{\mathit{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right) \tag{4.62}$$

- *else if $\delta_t \leq \varepsilon$:*

$$\mathit{Tr}\left(\mathit{Dec}_{\mathit{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right) \leq C + \varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j) \tag{4.63}$$

Proof. of Prop. 25. We consider the two cases, depending if δ_t is bigger than ε or not.

If $\delta_t > \varepsilon$: we proceed by separation of cases.

- If $\left|\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right) - C\right| > \varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)$:

$$\left|\delta_t - \mathsf{Comp}\left(\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right), C\right)\right| \leq \varepsilon$$

thanks to Cor. 1 in Cheon et al. (2020) for the precision of **NewComp**. We also used the fact that for any $x, y \in \mathbb{R}$ and $z \in \mathbb{R}_+^\star$, $\mathsf{Comp}(x/z, y/z) = \mathsf{Comp}(x, y)$. Using the equation above:

$$\mathsf{Comp}\left(\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right), C\right) \geq \delta_t - \varepsilon > 0$$

because we assumed here that $\delta_t > \varepsilon$. This readily implies that $\mathsf{Comp}\left(\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right), C\right) = 1$ because $\mathsf{Comp}(a,b) \in \{0,1\}$ for any $a, b \in [0,1]$. But, because we are in the case that:

$$\left|\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right) - C\right| > \varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)$$

[9] with the convention that $0/0 = 0$ and $C/0 = 1$

we have that either $\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) > C+\varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)$ or $\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) < C-\varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)$. Hence, because $\mathsf{Comp}\left(\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right), C\right) = 1$, we have that $\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) > C$ that is to say:

$$\begin{aligned}\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) &\geq C+\varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)\\ &\geq C-\varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)\end{aligned}$$

- If $\left|\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) - C\right| \leq \varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)$: We can not use Cor. 4 from Cheon et al. (2020). However, in this case we directly have by definition of the absolute value that:

$$-\varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j) \leq \mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) - C \tag{4.64}$$

If $\delta_t \leq \varepsilon$: Again, we distinguish the two different cases possible.

- If $\left|\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) - C\right| > \varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)$: Using Cor. 1 from Cheon et al. (2020), we have once again that:

$$\left|\delta_t - \mathsf{Comp}\left(\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right), C\right)\right| \leq \varepsilon$$

Therefore $\mathsf{Comp}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}, C\right) \leq \delta_t + \varepsilon \leq 2\varepsilon < 1$ (because $\varepsilon < 1/2$). But $\mathsf{Comp}(a,b) \in \{0,1\}$ for any $a, b \in [0,1]$ which means that $\mathsf{Comp}\left(\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right), C\right) = 0$. But we assumed that $\left|\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) - C\right| > \varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)$, in other words:

$$\begin{aligned}&\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) > C+\varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j) \geq C \text{ or}\\ &\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) < C-\varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)\end{aligned} \tag{4.65}$$

But $\mathsf{Comp}\left(\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right), C\right) = 0$, it is thus only possible that

$$\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) \leq C-\varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)$$

- If $\left|\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) - C\right| \leq \varepsilon'$:
In this case, by definition we have

$$\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^{\intercal}\right) \leq C+\varepsilon' L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)$$

□

The previous proposition ensures that when a batch is ended because $\delta_t > 0.45$ then we have, for a small enough ε', that, $\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} x_{l,a_l}x_{l,a_l}^{\intercal}\right) \geq C'$ for some constant C'. However, thanks to Prop. 19 we have that for any batch j,

that for all $l \in \{t_j + 1, \dots, t-1\}$:

$$\|x\|^2_{\bar{V}_j^{-1}} - \|x\|^2_{\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)} \leq L^2 \|\bar{V}_j^{-1} - \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\| \leq \frac{L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \tag{4.66}$$

Summing over all time steps $l \in [t_j + 1, t-1]$, we have that:

$$\left| \mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j) \sum_{l=t_j+1}^{t-1} s_{l,a_l} s_{l,a_l}^\intercal \right) - \mathsf{Tr}\left(\bar{V}_j^{-1} \sum_{l=t_j+1}^{t-1} s_{l,a_l} s_{l,a_l}^\intercal \right) \right| \leq \frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \tag{4.67}$$

Therefore when $\delta_t > 0.45$, we that:

$$\mathsf{Tr}\left(\bar{V}_j^{-1} \sum_{l=t_j+1}^{t-1} s_{l,a_l} s_{l,a_l}^\intercal \right) \geq C - \varepsilon_t' L^2 \left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}} \right) (t-1-t_j) - \frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \tag{4.68}$$

but $\varepsilon_t' = \frac{1}{4tL^2\left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}\right)(t-1-t_j)}$ so:

$$\mathsf{Tr}\left(\bar{V}_j^{-1} \sum_{l=t_j+1}^{t-1} s_{l,a_l} s_{l,a_l}^\intercal \right) \geq C - \frac{1}{4t} - \frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \tag{4.69}$$

But if $\delta_t \leq 0.45$:

$$\mathsf{Tr}\left(\bar{V}_j^{-1} \sum_{l=t_j+1}^{t-1} s_{l,a_l} s_{l,a_l}^\intercal \right) \leq C + \varepsilon_t' L^2 \left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}} \right) (t-1-t_j) + \frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \tag{4.70}$$

or using the definition of ε_t':

$$\mathsf{Tr}\left(\bar{V}_j^{-1} \sum_{l=t_j+1}^{t-1} s_{l,a_l} s_{l,a_l}^\intercal \right) \leq C + \frac{1}{4t} + \frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \tag{4.71}$$

Masking Procedure: In order to prevent any leakage of information when the user decrypts the result of this approximate comparison, we use a masking procedure where the algorithm adds a big noise to the bit encrypting the approximation of the comparison, somehow masking its value to the user. In order for this procedure to be secure, the algorithm needs to sample the noise from a distribution such the resulting distribution of the result observed by the user is independent of the value of δ_t (see Prop. 25). Formally, we add a noise $\xi \sim \Xi$ such that for any $x, x' \in [0, 1]$:

$$\mathbb{P}\left(Dec_{\mathsf{sk}}(\xi + x)\right) = \mathbb{P}\left(Dec_{\mathsf{pk}}(\xi + x')\right) \tag{4.72}$$

Finding such distribution is highly dependent on the encryption scheme used and its parameters. In our implementation, we used the CKKS scheme with depth $D = 100$, level of security $\kappa = 128$ and a log size of modulus $\log_2(q_0) = 4982$. Therefore, for a cyphertext ct at a given level l encrypting a number $x \in [0, 1]$ with a pair of public and secret key $(\mathbf{pk}, \mathbf{sk})$ we have that:

$$Dec_{\mathsf{sk}}(ct) = \langle ct, \mathbf{sk} \rangle (\mathrm{mod} q_l) \tag{4.73}$$

with $q_l = 2^l q_0$. When sampling an integer r uniformly in $\{0, \dots, q_l - 1\}$, we have that for any $k \in \{0, \dots, q_l - 1\}$ the distribution of $r + k (\mathrm{mod} q_l)$ is uniform over $\{0, \dots, q_l - 1\}$. We leverage this result to creates a masking procedure detailed in Alg. 40

Algorithm 40: Masking Procedure

Input: ciphertext: ct, modulus factor: q_l, cyclotomial polynomial degree: M
Sample uniformly r in $\{0, \dots, q_l - 1\}$
Compute the polynomial $\widetilde{r} \in \mathbb{Z}[X]/(X^M + 1)$ such that $\widetilde{r}(X) = r$
Output: $ct + \widetilde{r}$

Upon receiving the decryption of $ct + \widetilde{r}$, the unmasking procedure is consists in simply subtracting r.

Impact on the Growth of the determinant: In this subsection , we study the impact on the determinant of the design matrix when Condition 4.16 is satisfied for some constant C'. Our result is based on the classic following lemma.

Lemma 25. *For any positive definite symmetric matrix A, B and symmetric semi-positive definite matrix C such that $A = B + C$ we have that:*

$$\frac{det(A)}{det(B)} \geq 1 + Tr\left(B^{-1/2} C B^{-1/2}\right) \tag{4.74}$$

Proof. of Lemma 25. Using that $A = B + C$:

$$\frac{\det(A)}{\det(B)} = \det\left(I_d + B^{-1/2} C B^{-1/2}\right) \geq 1 + \mathrm{Tr}\left(B^{-1/2} C B^{-1/2}\right) = 1 + \mathrm{Tr}\left(B^{-1} C\right) \tag{4.75}$$

The last inequality is a consequence of the following inequality:

$$\forall n \in \mathbb{N}^\star, \forall a \in \mathbb{R}_+^n, \qquad 1 + \sum_{i=1}^{n} a_i \leq \prod_{i=1}^{n} (1 + a_i) \tag{4.76}$$

Indeed $I_d + B^{-1/2} C B^{-1/2}$ is symmetric definite positive hence its eignevalues are positive. □

Therefore, using the lemma above applied to the design matrix, $V_t = \bar{V}_j + \sum_{l=t_j+1}^{t-1} s_{l,a_l} s_{l,a_l}^\intercal$ for $t \geq t_j + 1$, we have that:

$$\det\left(V_t\right) \geq \left(1 + \mathrm{Tr}\left(\bar{V}_j^{-1} \sum_{l=t_j+1}^{t-1} s_{l,a_l} s_{l,a_l}^\intercal\right)\right) \det(\bar{V}_j) \tag{4.77}$$

Putting Everything Together: We are finally, ready to prove an upper-bound on the number of batches. First, let's recall Prop. 22.

Proposition. *If $C - \frac{L\eta}{\sqrt{\lambda + L^2}} > \frac{1}{4}$, the number of episodes in Alg. 34, M_T for T steps, is bounded by:*

$$M_T \leq 1 + \frac{d \ln\left(1 + \frac{L^2 T}{\lambda d}\right)}{2 \ln\left(\frac{3}{4} + C - \frac{L\eta}{\sqrt{\lambda + L^2}}\right)} + \frac{\ln(T)}{\ln(1+\eta)} \tag{4.78}$$

Proof. of Lem. 22. Let's define for $i \geq 1$, the macro-episode:

$$n_i = \min\left\{t > n_{i-1} \mid \delta_t > \varepsilon_t\right\} \tag{4.79}$$

with $n_0 = 0$. In other words, macro-episodes are episodes such that the norm of the context has grown too big. It means that for all episodes between two macro-episodes the batches are ended because the current batch is too long. Therefore for macro-episode i, thanks to Eq. (4.67) and Prop. 25:

$$C - \varepsilon'_{n_i} L^2 \left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}\right)(n_i - 1 - t_j) - \frac{L(n_i - 1 - t_j)}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \leq \mathrm{Tr}\left(\bar{V}_j^{-1} \sum_{l=t_j+1}^{n_i - 1} s_{l,a_l} s_{l,a_l}^\intercal\right) \tag{4.80}$$

where $\varepsilon'_{n_i} = \left(4 n_i L^2 \left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}\right)(n_i - 1 - t_j)\right)^{-1}$ as defined in Alg. 34. But the batch j for which $n_i = t_{j+1}$ is such that $t_{j+1} - t_j \leq \eta t_j + 1$ (thanks to the second if condition in Alg. 34). Hence:

$$\frac{L(n_i - 1 - t_j)}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \leq \frac{L\eta}{\sqrt{t_j(\lambda + L^2 t_j)}} \leq \frac{L\eta}{\sqrt{\lambda + L^2}} \tag{4.81}$$

Therefore by Lem. 25 we have that:

$$\det \bar{V}_{j+1} \geq \left(1 + C - \left(\frac{1}{4} + \frac{L\eta}{\sqrt{\lambda + L^2}}\right)\right) \det \bar{V}_j \tag{4.82}$$

because for all t, $\varepsilon'_t L^2 \left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}\right)(t - 1 - t_j) \leq \frac{1}{4}$ and because for any episode j between two macro-episodes i and $i+1$ the determinant of the design matrix is an increasing function of the episode (because for two matrices M, N symmetric semi-definite positive $\det(M + N) \geq \det(M)$).

Thus $\det \bar{V}_{j_i} \geq (\frac{3}{4} + C - \frac{L\eta}{\sqrt{\lambda+L^2}}) \det \bar{V}_{j_{i-1}}$ where j_i is the episode such that $n_i = t_{j_i+1}$. Therefore the number of macro-episodes M_1 is such that:

$$\left(\frac{3}{4} + C - \frac{L\eta}{\sqrt{\lambda + L^2}}\right)^{M_1 - 1} \leq \frac{\det(\bar{V}_{M_T})}{\det(\bar{V}_0)} \tag{4.83}$$

where $\bar{V}_{M_T}$ is the design matrix after T steps (or M_T batches) and $\bar{V}_0 = \lambda I_d$. This upper bound gives that:

$$M_1 \leq 1 + \frac{\ln\left(\frac{\det(\bar{V}_{M_T})}{\det(\bar{V}_0)}\right)}{\ln\left(\frac{3}{4} + C - \frac{L\eta}{\sqrt{\lambda+L^2}}\right)} \tag{4.84}$$

if $\frac{3}{4} + C - \frac{L\eta}{\sqrt{\lambda+L^2}} > 1$. Moreover, thanks to Lemma 10 in Abbasi-Yadkori et al. (2011), the log-determinant of the design matrix is bounded by: $\ln\left(\frac{\det(\bar{V}_{M_T})}{\det(\bar{V}_0)}\right) \leq d \ln\left(1 + \frac{TL^2}{\lambda d}\right)$. In addition, there is at most $1 + \frac{\ln(n_{i+1}/n_i)}{\ln(1+\eta)}$ batches between macro-episode i and $i+1$. Therefore:

$$\begin{aligned} M_T \leq \sum_{i=0}^{M_1 - 1} 1 + \frac{1}{\ln(1+\eta)} \ln(n_{i+1}/n_i) &= M_1 + \frac{\ln(T)}{\ln(1+\eta)} \\ &\leq 1 + \frac{d \ln\left(1 + \frac{L^2 T}{\lambda d}\right)}{2 \ln\left(\frac{3}{4} + C - \frac{L\eta}{\sqrt{\lambda+L^2}}\right)} + \frac{\ln(T)}{\ln(1+\eta)} \end{aligned}$$

□

4.B.5.2 Regret Upper Bound (Proof of Thm. 12)

Now that we have shown an upper-bound on the number of bathes for the HELBA algorithm, we are ready to prove the regret bound of Thm. 12. The proof of this theorem follows the same logic as the regret analysis of OFUL. That is to say, we first show a high-probability upper bound on the regret thanks to optimism and then proceed to bound each term of the bonus used in HELBA.

We first show the following lemma giving a first upper bound on the regret relating the error due to the approximation of the argmax and optimism.

Lemma 26. *For any $\delta > 0$, the regret of Alg. 34 is bounded with probability at least $1 - \delta$ by:*

$$\begin{aligned} R_T(\text{HELBA}) \leq & \underbrace{\sum_{j=0}^{M_T - 1} \sum_{t=t_j+1}^{t_{j+1}} \frac{4}{t} \left(1 + 2\widetilde{\beta}(j) \left[\frac{2}{t} + \sqrt{\frac{L}{t_j^{3/2}\sqrt{t_j L^2 + \lambda}}} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}}\right]\right)}_{:=①} \\ & + \underbrace{\sum_{j=0}^{M_T - 1} \sum_{t=t_j+1}^{t_{j+1}} 2\widetilde{\beta}(j) \left[\text{sqrt}_{\text{HE}}\left(s_{t,a}^\top \mathit{Dec}_{\mathit{sk}}(\bar{A}_j) s_{t,a} + \frac{L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}}\right) + \frac{1}{t}\right]}_{:=②} \end{aligned} \tag{4.85}$$

where for every time step t, $a_t^\star = \arg\max_{a \in [K]} \langle x_{t,a}, \theta^\star \rangle$, M_T is the number of batches and $R_T(\text{HELBA}) = \sum_{t=1}^T \langle \theta^\star, s_{t,a_t^\star} - s_{t,a_t} \rangle$.

Proof. of Lem. 26. First, let's define E the event that all confidence ellipsoids, $\tilde{\mathcal{C}}_j$, contain $\theta^\star$ with probability at least $1 - \delta$. That is to say $E = \left\{\theta^\star \in \bigcap_{j=1}^{+\infty} \tilde{\mathcal{C}}_j(\delta)\right\}$. Thanks to Prop. 23, $\mathbb{P}(E) \geq 1 - \delta$. Because E is included in the event described by Prop. 23.

Therefore conditioned on the event E, after T steps the regret can be decomposed as:

$$\begin{aligned} R_T(\text{HELBA}) = \sum_{t=1}^T & \langle \theta^\star, s_{t,a_t^\star} \rangle - \max_{a \leq K} \mathsf{Dec}_{\mathsf{sk}}(\rho_a(t)) + \max_{a \leq K} \mathsf{Dec}_{\mathsf{sk}}(\rho_a(t)) - \mathsf{Dec}_{\mathsf{sk}}(\rho_{a_t}(t)) \\ & + \mathsf{Dec}_{\mathsf{sk}}(\rho_{a_t}(t)) - \langle \theta^\star, s_{t,a_t} \rangle \end{aligned} \tag{4.86}$$

where $\rho_a(t)$ is the optimistic upper bound on the reward of arm a computed by Alg. 34 and $a_t^\star = \arg\max_{a\in[K]}\langle\theta^\star, s_{t,a_t^\star}\rangle$. Now for any $t \le T$, under the event E, $\langle\theta^\star, s_{t,a_t^\star}\rangle \le \max_{a\le K} \mathsf{Dec}_{\mathsf{sk}}(\rho_a(t))$. But thanks to Cor. 4, for any $t \ge 1$ inside batch j:

$$\max_a \mathsf{Dec}_{\mathsf{sk}}(\rho_a(t)) - \mathsf{Dec}_{\mathsf{sk}}(\rho_{a_t}(t)) \le \frac{1}{t}\left(1 + \widetilde{\beta}(j)\left[\frac{2}{t} + \sqrt{\frac{L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}}} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}}\right]\right) \tag{4.87}$$

In addition, we have that under event E:

$$\mathsf{Dec}_{\mathsf{sk}}(\rho_{a_t}(t)) - \langle\theta^\star, s_{t,a_t}\rangle = \langle\tilde{\theta}_j - \theta^\star, s_{t,a_t}\rangle + \widetilde{\beta}(j)\left[\mathrm{sqrt}_{\mathrm{HE}}\left(s_{t,a}^\top \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j) s_{t,a} + \frac{L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}}\right) + \frac{1}{t}\right]$$

But still conditioned on the event E,

$$\begin{aligned}\langle\tilde{\theta}_j - \theta^\star, s_{t,a_t}\rangle &\le \|\tilde{\theta}_j - \theta^\star\|_{\bar{V}_j}\|s_{t,a_t}\|_{\bar{V}_j^{-1}} \le \tilde{\beta}(j)\|s_{t,a_t}\|_{\bar{V}_j^{-1}} \\ &\le \widetilde{\beta}(j)\left[\mathrm{sqrt}_{\mathrm{HE}}\left(s_{t,a_t}^\top \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j) s_{t,a_t} + \frac{L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}}\right) + \frac{1}{t}\right].\end{aligned} \tag{4.88}$$

Putting the last two equations together, for every step $t \le T$:

$$\begin{aligned}\langle\theta^\star, s_{t,a_t^\star} - s_{t,a_t}\rangle \le& \frac{1}{t}\left(1 + \widetilde{\beta}(j)\left[\frac{2}{t} + \sqrt{\frac{L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}}} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}}\right]\right) \\ &+ 2\widetilde{\beta}(j)\left[\mathrm{sqrt}_{\mathrm{HE}}\left(s_{t,a_t}^\top \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j) s_{t,a_t} + \frac{L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}}\right) + \frac{1}{t}\right]\end{aligned}$$

□

Bounding ①. We now proceed to bound each term in Eq. (4.85). The following lemma is used to ①.

Lemma 27. *For all $t \ge 1$:*

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}} \frac{1}{t}\left(1 + \widetilde{\beta}(j)\left[\frac{2}{t} + \sqrt{\frac{L}{t_j^{3/2}\sqrt{t_j L^2 + \lambda}}} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}}\right]\right) \le \mathcal{O}(\ln(T)^{3/2}) \tag{4.89}$$

Proof. of Lem. 27. Because $t_j \ge 1$:

$$\frac{L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \le \frac{L}{\sqrt{\lambda}}, \quad \widetilde{\beta}(j) \le 1 + \sqrt{\lambda}S + \sigma\sqrt{d\left(\ln\left(1 + \frac{L^2 T}{\lambda d}\right) + \ln\left(\frac{\pi^2 T^2}{6\delta}\right)\right)} \tag{4.90}$$

Bounding each component of the sum of ① in Eq. (4.85) individually, we get:

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}} \frac{1}{t} \le (1 + \ln(T)) \tag{4.91}$$

Hence:

$$\begin{aligned}\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}} \frac{1}{t}\left(1 + \widetilde{\beta}(j)\left[\frac{2}{t} + \sqrt{\frac{L}{t_j^{3/2}\sqrt{t_j L^2 + \lambda}}} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}}\right]\right) \le (1 + \ln(T))\Bigg[1 + \\ \left(1 + \sqrt{\lambda}S + \sigma\sqrt{d\left(\ln\left(1 + \frac{L^2 T}{\lambda d}\right) + \ln\left(\frac{\pi^2 T^2}{6\delta}\right)\right)}\right)\left(2 + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}} + \sqrt{\frac{L}{\sqrt{\lambda}}}\right)\Bigg]\end{aligned}$$

□

Lem. 27 shows that the error from our procedure to select the argmax induces only an additional logarithmic cost in T compared with the regret of directly selecting the argmax of the UCBs $(\rho_a(t))_{a\le K}$.

Bounding ②. We are now left with bounding the second term in Eq. (4.85). This term is usually the one that appears in regret analysis for linear contextual bandits. First, ② can be further broke down thanks to the following lemma.

Lemma 28. *For all $t \geq 1$,*

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}} \widetilde{\beta}(j)\left[\mathrm{sqrt}_{\mathrm{HE}}\left(s_{t,a_t}^\top \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)s_{t,a_t} + \frac{L}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}\right) + \frac{1}{t}\right] \leq \underbrace{\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}} \frac{2\widetilde{\beta}(j)}{t}}_{:=ⓐ} + \underbrace{\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}} \widetilde{\beta}(j)\|s_{t,a_t}\|_{\bar{V}_j^{-1}}}_{:=ⓑ} + \underbrace{\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}} \widetilde{\beta}(j)\sqrt{\frac{2L}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}}}_{:=ⓒ} \tag{4.92}$$

Proof. of Lem. 28. For any time $t \geq 1$ thanks to Prop. 20, we have:

$$\begin{aligned}
\mathrm{sqrt}_{\mathrm{HE}}\left(s_{t,a_t}^\top \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)s_{t,a_t} + \frac{L}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}\right) &\leq \frac{1}{t} + \sqrt{\|s_{t,a_t}\|^2_{\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)} + \frac{L}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}} \\
&\leq \frac{1}{t} + \sqrt{\frac{L}{t_j^{3/2}\sqrt{\lambda+L^2t_j}} + \|s_{t,a_t}\|^2_{\bar{V}_j^{-1}} + \|s_{t,a_t}\|_2^2\|\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j) - \bar{V}_j^{-1}\|} \\
&\leq \frac{1}{t} + \sqrt{\frac{2L}{t_j^{3/2}\sqrt{\lambda+L^2t_j}} + \|s_{t,a_t}\|^2_{\bar{V}_j^{-1}}} \\
&\leq \frac{1}{t} + \sqrt{\frac{2L}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}} + \|s_{t,a_t}\|_{\bar{V}_j^{-1}}
\end{aligned}$$

□

We proceed to bound each term ⓐ, ⓑ, ⓒ. Bounding ⓑ is similar to the analysis of OFUL. On the other hand, bounding neatly ⓒ is the reason why we introduced the condition that a new episode is started is $t \geq (1+\eta)t_j$.

The following lemma bounds ⓐ which is simply a numerical error due to the approximation of the square root.

Lemma 29. *For any $T \geq 1$,*

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}} \frac{4\widetilde{\beta}(j)}{t} \leq 4\left(1 + \sqrt{\lambda}S + \sigma\sqrt{d\left(\ln\left(1+\frac{L^2T}{\lambda d}\right) + \ln\left(\frac{\pi^2T^2}{6\delta}\right)\right)}\right)(1+\ln(T)) \tag{4.93}$$

Proof. of Lem. 29. Using the upper bound on the $\widetilde{\beta}(j)$ shown in the proof of Lem. 27, we get the result. □

We are finally left with the two terms ⓑ and ⓒ. The first term, ⓑ, will be compared to the bonus used in OFUL so that we can use Lemma 11 in Abbasi-Yadkori et al. (2011) to bound it. But first, we need to show how the norm for two different matrices A and B relates to each other.

Lemma 30. *For any context $x \in \mathbb{R}^d$ and symmetric semi-definite matrix A,B and C such that $A = B + C$ then:*

$$\|x\|^2_{B^{-1}} \leq \lambda_{\max}\left(I_d + B^{-1/2}CB^{-1/2}\right)\|x\|^2_{A^{-1}} \leq \left(1 + \mathit{Tr}\left(B^{-1/2}CB^{-1/2}\right)\right)\|x\|^2_{A^{-1}} \tag{4.94}$$

where $\lambda_{\max}(.)$ returns the maximum eigenvalue of a matrix.

Proof. of Lemma 30. We have by definition of A and B:

$$\begin{aligned}
\langle x, A^{-1}x\rangle = \langle x, (B+C)^{-1}x\rangle &= \langle x, B^{-1/2}(I_d + B^{-1/2}CB^{-1/2})^{-1}B^{-1/2}x\rangle &(4.95)\\
&= \langle B^{-1/2}x, (I_d + B^{-1/2}CB^{-1/2})^{-1}(B^{-1/2}x)\rangle &(4.96)\\
&\geq \lambda_{\min}\left((I_d + B^{-1/2}CB^{-1/2})^{-1}\right)\|B^{-1/2}x\|^2 &(4.97)\\
&\geq \frac{1}{\lambda_{\max}(I_d + B^{-1/2}CB^{-1/2})}\|x\|^2_{B^{-1}} &(4.98)
\end{aligned}$$

Hence:

$$\|x\|_{B^{-1}}^2 \leq \lambda_{\max}\left(I_d + B^{-1/2}CB^{-1/2}\right)\|x\|_{A^{-1}}^2 \tag{4.99}$$

The result follows from Weyl's inequality Horn and Johnson (1991), that is to say for all symmetric matrix M, N $\lambda_{\max}(M+N) \leq \lambda_{\max}(M) + \lambda_{\max}(N)$. And the fact that all eigenvalues of $B^{-1/2}CB^{-1/2}$ are positive hence $\lambda_{\max}(B^{-1/2}CB^{-1/2}) \leq \mathsf{Tr}(B^{-1/2}CB^{-1/2}) = \mathsf{Tr}(CB^{-1})$. □

We are now able to bound ⓑ using Lemma 11 in Abbasi-Yadkori et al. (2011).

Lemma 31. *If $\lambda \geq L^2$ we have:*

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}} 2\beta(j)\|s_{t,a_t}\|_{\bar{V}_j^{-1}} \leq \beta^\star\sqrt{2d\ln\left(1+\frac{TL^2}{\lambda d}\right)}\left[\sqrt{T\left(1.25 + C + L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)\right)}\right.$$
$$\left.+\sqrt{M_T\left(\eta^2 + \frac{L}{(\lambda+L^2)^{3/2}}\right)}\right] \tag{4.100}$$

with $\beta^\star = 1 + \sqrt{\lambda}S + \sigma\sqrt{d\left(\ln\left(1+\frac{L^2T}{\lambda d}\right) + \ln\left(\frac{\pi^2T^2}{6\delta}\right)\right)}$

Proof. of Lem. 31. For any time t in batch j, we have thanks to Lem. 30 that:

$$\|s_{t,a_t}\|_{\bar{V}_j^{-1}} \leq \sqrt{1 + \mathsf{Tr}\left(\bar{V}_j^{-1}\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right)}\|s_{t,a_t}\|_{V_t^{-1}} \tag{4.101}$$

with $V_t = \lambda I_d + \sum_{l=1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal$. The rest of the proof relies on bounding $\mathsf{Tr}\left(\bar{V}_j^{-1}\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right)$. To do so, we will use the following inequality, see Eq. (4.67):

$$-\frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda+L^2t_j}} \leq \mathsf{Tr}\left(\left(\bar{V}_j^{-1} - \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\right)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right) \leq \frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}$$

Therefore$\delta_t \leq 0.45$ during batch j, because it is not over while no condition is satisfied. Thanks to Prop. 25 with $\varepsilon' = \frac{1}{4tL^2(t-1-t_j)}$, we get:

$$\forall t \in \{t_j+1, \dots t_{j+1}-1\}, \qquad \mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right) \leq C + \frac{t-1-t_j}{4t}$$

However, for $t = t_{j+1}$ we have either that $\delta_{t_{j+1}} > 0.45$ or $t_{j+1} \geq (1+\eta)t_j$:

- If $\delta_{t_{j+1}} \leq 0.45$, then $\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t_{j+1}-1} s_{l,a_l}s_{l,a_l}^\intercal\right) \leq C + \frac{t_{j+1}-1-t_j}{4t}$
- If $\delta_{t_{j+1}} > 0.45$, then $\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t_{j+1}-1} s_{l,a_l}s_{l,a_l}^\intercal\right) \geq C - \frac{t_{j+1}-1-t_j}{4t}$ but $\delta_{t_{j+1}-1} \leq 0.45$ thus $\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t_{j+1}-2} s_{l,a_l}s_{l,a_l}^\intercal\right) \leq C + \frac{t-1-t_j}{4t}$. Therefore, using that $\|s_{t,a_t}\|_{\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)}^2 \leq \lambda_{\max}(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j))\|s_{t,a_t}\|_2^2 \leq L^2\left(\frac{1}{\lambda} + \frac{1}{L\sqrt{\lambda+L^2}}\right)$. Hence, we have that:

$$\mathsf{Tr}\left(\mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j)\sum_{l=t_j+1}^{t_{j+1}-1} s_{l,a_l}s_{l,a_l}^\intercal\right) \leq C + \frac{t-1-t_j}{4t} + L^2\left(\frac{1}{\lambda} + \frac{1}{L\sqrt{\lambda+L^2}}\right) \tag{4.102}$$

To sum up, for all $t_j + 1 \leq t \leq t_{j+1}$:

$$\mathsf{Tr}\left(\bar{V}_j^{-1}\sum_{l=t_j+1}^{t-1} s_{l,a_l}s_{l,a_l}^\intercal\right) \leq C + \frac{t-1-t_j}{4t} + L^2\left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}\right) + \frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda+L^2t_j}} \tag{4.103}$$

Overall, we have that:

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}}\|s_{t,a_t}\|_{\bar{V}_j^{-1}} \leq \sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}}\sqrt{1+\mathsf{Tr}\left(\bar{V}_j^{-1}\sum_{l=t_j+1}^{t-1}s_{l,a_l}s_{l,a_l}^{\intercal}\right)}\|s_{t,a_t}\|_{V_t^{-1}} \tag{4.104}$$

$$\leq \sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}}\| s_{t,a_t}\|_{V_t^{-1}}\sqrt{1+C+\frac{t-1-t_j}{4t}+L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)+\frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}} \tag{4.105}$$

$$\leq \sqrt{\frac{5}{4}+C+L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)}\sqrt{T\sum_{t=1}^{T}\|s_{t,a_t}\|^2_{V_t^{-1}}+\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}}\| s_{t,a_t}\|_{V_t^{-1}}\sqrt{\frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}}} \tag{4.106}$$

where the last inequality is due to Cauchy-Schwarz inequality. The first term in inequality Eq. (4.106) is bounded by using Lemma 29 in Ruan et al. (2020),

$$\sum_{t=1}^{T}\|x_{t,a_t}\|^2_{V_t^{-1}} \leq 2\ln\left(\frac{\det(V_T)}{\det(V_0)}\right) \leq 2d\ln\left(1+\frac{TL^2}{\lambda d}\right) \tag{4.107}$$

In addition, the last term in Eq. (4.106) is bounded by:

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}}\frac{L(t-1-t_j)}{t_j^{3/2}\sqrt{\lambda+L^2t_j}} \leq \sum_{j=0}^{M_T-1}\frac{L(t_{j+1}-t_j)^2}{2t_j^{3/2}\sqrt{\lambda+L^2t_j}} \tag{4.108}$$

But a consequence of the second condition is that the length of batch j satisfies $t_{j+1}-t_j \leq \eta t_j+1$. Therefore:

$$\sum_{j=0}^{M_T-1}\frac{Lf(t_{j+1}-t_j)^2}{2t_j^{3/2}\sqrt{\lambda+L^2t_j}} \leq \sum_{j=0}^{M_T-1}\frac{L(\eta t_j+1)^2}{2t_j^{3/2}\sqrt{\lambda+L^2t_j}} \tag{4.109}$$

$$\leq \sum_{j=0}^{M_T-1}\frac{L(\eta^2t_j^2+1)}{t_j^{3/2}\sqrt{\lambda+L^2t_j}} \tag{4.110}$$

$$\leq \sum_{j=0}^{M_T-1}\eta^2+\frac{L}{(\lambda+L^2)^{3/2}} \leq \eta^2M_T+\frac{LM_T}{(\lambda+L^2)^{3/2}} \tag{4.111}$$

Putting everything together we get:

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}}\|s_{t,a_t}\|_{\bar{V}_j^{-1}} \leq \sqrt{\frac{5}{4}+C+L^2\left(\frac{1}{\lambda}+\frac{1}{\sqrt{\lambda}}\right)}\sqrt{2Td\ln\left(1+\frac{TL^2}{\lambda d}\right)}$$
$$+\sqrt{2d\ln\left(1+\frac{TL^2}{\lambda d}\right)\left(\eta^2M_T+\frac{LM_T}{(\lambda+L^2)^{3/2}}\right)}$$

Hence the result using the upper bound on $\widetilde{\beta}(j)$ proved in Lem. 27. □

Finally, the last term to bound is ©, that we do similarly to the end of the proof of Lem.31.

Lemma 32. *For all $T \geq 1$,*

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}}2\widetilde{\beta}(j)\sqrt{\frac{2L}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}} \leq 2\sqrt{2L}M_T\beta^\star\left[1+\frac{\eta}{\sqrt{L}}\right] \tag{4.112}$$

with $\beta^\star = 1+\sqrt{\lambda}S+\sigma\sqrt{d\left(\ln\left(1+\frac{L^2T}{\lambda d}\right)+\ln\left(\frac{\pi^2T^2}{6\delta}\right)\right)}$ and M_T the number of episodes.

Proof. of Lem. 32. We have:

$$\sum_{j=0}^{M_T-1}\sum_{t=t_j+1}^{t_{j+1}}2\widetilde{\beta}(j)\sqrt{\frac{2L}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}} \leq 2\sqrt{2L}\max_j\widetilde{\beta}(j)\sum_{j=0}^{M_T-1}\sqrt{\frac{1}{t_j^{3/2}\sqrt{\lambda+L^2t_j}}}(t_{j+1}-t_j) \tag{4.113}$$

But the condition on the length of the batch ensures that for any batch j, $t_{j+1} - t_j \leq \eta t_j + 1$, thus equation above can be bounded by:

$$\sum_{j=0}^{M_T-1} \sum_{t=t_j+1}^{t_{j+1}} 2\widetilde{\beta}(j) \sqrt{\frac{2L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}}} \leq 2\sqrt{2L} \max_j \widetilde{\beta}(j) \left[M_T + \sum_{j=0}^{M_T-1} \eta \sqrt{\frac{\sqrt{t_j}}{\sqrt{\lambda + L^2 t_j}}} \right] \tag{4.114}$$

$$\leq 2\sqrt{2L} \max_j \widetilde{\beta}(j) \left[M_T + \sum_{j=0}^{M_T-1} \frac{\eta}{\sqrt{L}} \right] \tag{4.115}$$

$$\leq 2\sqrt{2L} M_T \max_j \widetilde{\beta}(j) \left[1 + \frac{\eta}{\sqrt{L}} \right] \tag{4.116}$$

Hence the result. □

Finally, we can finish the proof of Thm. 12, but we first recall its statement.

Theorem. *Under Asm. 15, for any $\delta > 0$ and $T \geq d$, there exists universal constants $C_1, C_2 > 0$ such that the regret of* HELBA *(Alg. 34) is bounded with probability at least $1 - \delta$ by:*

$$R_T \leq C_1 \beta^\star \left(\sqrt{\left(\frac{5}{4} + C\right) dT \ln\left(\frac{TL}{\lambda d}\right)} + \frac{L^{3/2}}{\sqrt{\lambda}} \ln(T) \right) + C_2 \beta^\star M_T \max\left\{ \sqrt{L} + \eta, \eta^2 + \frac{L}{\sqrt{\lambda + L^2}^3} \right\}$$

Proof. of Thm. 12. For any $\delta > 0$, let's define the event E as in the proof of Lem. 26. Then conditioned on this event, we have using Lem. 26:

$$\begin{aligned} R_T(\text{HELBA}) \leq & \sum_{j=0}^{M_T-1} \sum_{t=t_j+1}^{t_{j+1}} \frac{4}{t} \left(1 + 2\widetilde{\beta}(j) \left[\frac{2}{t} + \sqrt{\frac{L}{t_j^{3/2}\sqrt{t_j L^2 + \lambda}}} + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}} \right] \right) \\ & + \sum_{j=0}^{M_T-1} \sum_{t=t_j+1}^{t_{j+1}} 2\widetilde{\beta}(j) \left[\text{sqrt}_{\text{HE}} \left(s_{t,a_t}^\top \mathsf{Dec}_{\mathsf{sk}}(\bar{A}_j) s_{t,a_t} + \frac{L}{t_j^{3/2}\sqrt{\lambda + L^2 t_j}} \right) + \frac{1}{t} \right] \end{aligned} \tag{4.117}$$

But using Lem. 27 to bound the first term of the RHS equation above but also Lem. 28, 29, 31 and 32 to the bound the second term, we get:

$$\begin{aligned} R_T\left(\text{HELBA}\right) \leq (1 + \ \ln(T)) & \left[1 + \beta^\star \left(6 + L\sqrt{\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}}} + \sqrt{\frac{L}{\sqrt{\lambda}}} \right) \right] \\ + \beta^\star \sqrt{2d \ln\left(1 + \frac{TL^2}{\lambda d}\right)} & \left[\sqrt{\frac{5}{4} + C + L^2 \left(\frac{1}{\lambda} + \frac{1}{\sqrt{\lambda}} \right)} \sqrt{T} + \left(\eta^2 + \frac{L}{(\lambda + L^2)^{3/2}} \right) M_T \right] \\ & + 2\sqrt{2L} M_T \beta^\star \left[1 + \frac{\eta}{\sqrt{L}} \right] \end{aligned} \tag{4.118}$$

with $\beta^\star = 1 + \sqrt{\lambda} S + \sigma \sqrt{d \left(\ln\left(1 + \frac{L^2 T}{\lambda d}\right) + \ln\left(\frac{\pi^2 T^2}{6\delta}\right) \right)}$ and $M_T = 1 + \frac{d \ln\left(1 + \frac{L^2 T}{\lambda d}\right)}{2 \ln\left(\frac{3}{4} + C - \frac{L\eta}{\sqrt{\lambda + L^2}}\right)} + \frac{\ln(T)}{\ln(1+\eta)}$ □

4.B.6 Implementation Details

In this subsection , we further detail how HELBA is implemented. In particular, we present the matrix multiplication and matrix-vector operations.

For the experiments, we used the PALISADE library (development version v1.10.4) PAL (2020). This library automatically chooses most of the parameters used for the CKKS scheme. In particular the ring dimension of the ciphertext space is chosen automatically. In the end, the user only need to choose four parameters: the maximum multiplicative depth (here chosen at 100), the number of bits used for the scaling factor (here 50), the batch size that is to say the number of plaintext slots used in the ciphertext (here 8) and the security level (here chosen at 128 bits for Fig. 4.2.1).

4.B.6.1 Matrix/Vector Encoding

Usually, when dealing with matrices and vectors in homomorphic encryption there are multiple ways to encrypt those. For example, with a vector $y \in \mathbb{R}^d$ one can create d ciphertexts encrypting each value y_i for all $i \leq d$. This approach is nonetheless expensive in terms of memory. An other approach is to encrypt directly the whole vector in a single ciphertext. A ciphertext is a polynomial $(X \mapsto \sum_{i=0}^{N} a_i X^i)$ where each coefficient is used to encrypt a value of y ($a_i = y_i$ for $i \leq d$). This second method is oftentimes preferred as it reduce memory usage.

It is possible to take advantage of this encoding method in order to facilitate computations, e.g., matrix multiplication, matrix-vector operation or scalar product. In this work, we need to compute the product of square matrices of size $d \times d$, thus we choose to encrypt each matrix/vector as a unique ciphertext (assuming $d \leq N$). We have two different encoding for matrices and vectors. For a matrix $A = (a_{i,j})_{i \in \{0,\dots,p-1\}, j \in \{0,\dots,q-1\}}$ with $p, q \in \mathbb{N}$, we first transform A into a vector of size pq, $\tilde{a} = (a_{0,0}, a_{0,1}, \dots, a_{0,q-1}, \dots, a_{1,0}, \dots, a_{1,q-1}, \dots, a_{p-1,q-1})$. This vector is then encrypted into a single ciphertext. But for a vector $y \in \mathbb{R}^q$, we create a bigger vector of dimension pq (here p is a parameter of the encoding method for vectors), $\tilde{y} = (y_j)_{i \in \{0,\dots,p-1\}, j \in \{0,\dots,q-1\}} = (y_0, \dots, y_{q-1}, y_0, \dots, y_{q-1}, \dots, y_{q-1})$. We choose those two encodings because the homomorphic multiplication operation of PALISADE only perform a coordinate-wise multiplication between two ciphertexts. Therefore, using this encoding, a matrix-vector product for a matrix $A \in \mathbb{R}^{p \times q}$, a vector $y \in \mathbb{R}^q$, a public key **pk** can be computed as:

$$c_A \times c_y = \mathsf{Enc}_{\mathbf{pk}}(\tilde{a} \cdot \tilde{y}) = \mathsf{Enc}_{\mathbf{pk}}\Big(\big(a_{0,0}y_0, a_{0,1}y_1, \dots, a_{0,q-1}y_{q-1}, a_{1,0}y_0, a_{1,1}y_1, \dots, a_{1,q-1}y_{q-1}, \dots, a_{p-1,q-1}y_{q-1}\big)\Big)$$

with $\tilde{a}$ the encoding of A, $c_A = \mathsf{Enc}_{\mathbf{pk}}(\tilde{a})$, $\tilde{y}$ the encoding of y of dimension pq and $c_y = \mathsf{Enc}_{\mathbf{pk}}(\tilde{y})$, $\times$ the homomorphic multiplication operation and $\tilde{a} \cdot \tilde{y}$ the element-wise product. Then using EvalSumCol (an implementation of the SumColVec method from Han et al. in the PALISADE library) to compute partial sums of the coefficients of $c_A \times c_y$, we get:

$$\mathsf{EvalSumCol}(c_A \times c_y, p, q) = \mathsf{Enc}_{\mathbf{pk}}\Big(\big(\sum_{j=0}^{q-1} a_{0,j}y_j, \dots, \sum_{j=0}^{q-1} a_{0,j}y_j, \sum_{j=0}^{q-1} a_{1,j}y_j, \dots, \sum_{j=0}^{q-1} a_{1,j}y_j, \dots, \sum_{j=0}^{q-1} a_{p-1,j}y_j\big)\Big)$$

Finally, the matrix-vector product Ay is computed by $\mathsf{EvalSumCol}(c_A \times c_y, p, q)$ taking the coefficient at $(j + j \cdot p)_{j \in [p]}$.

4.B.6.2 Matrix Multiplication

Using the encoding of App. 4.B.6.1 we have a way to compute a matrix-vector product therefore computing the product between two square matrices $M, N \in \mathbb{R}^{p \times p}$ can be done using a series of matrix-vector products. However, this approach requires p ciphertexts to represent a matrix. We then prefer to use the method introduced in Sec. 3 of Jiang et al. (2018). This method relies on the following identity for any matrices $M, N \in \mathbb{R}^{p \times p}$ and $i, j \in \{0, \dots, p-1\}$:

$$\begin{aligned}(MN)_{i,j} &= \sum_{k=0}^{p-1} M_{i,k}N_{k,j} = \sum_{k=0}^{p-1} M_{i,[i+k+j]_p}N_{[i+k+j]_p,j} \\ &= \sum_{k=0}^{p-1} \sigma(M)_{i,[j+k]_p}\tau(N)_{[i+k]_p,j} = \sum_{k=0}^{p-1} (\phi^k \circ \sigma(M))_{i,j}(\psi^k \circ \tau(N))_{i,j}\end{aligned} \tag{4.119}$$

where we define σ, τ, ψ and ϕ as: and $[.]_p$ is the modulo operator. Therefore, using Eq. (4.119) we have that computing the

- $\sigma(M)_{i,j} = M_{i,[i+j]_p}$
- $\tau(M)_{i,j} = M_{[i+j]_p,j}$
- $\psi(M)_{i,j} = M_{i,[i+1]_p}$
- $\phi(M)_{i,j} = M_{[i+1]_p,j}$

product between M and N can simply be done by computing a component-wise multiplication between $(\phi^k \circ \sigma(M))_{i,j}$ and $(\psi^k \circ \tau(N))_{i,j}$ for all $k \in \{0, \dots, p-1\}$. Those quantities can in turn be easily computed thanks to a multiplication between a plaintext and a ciphertext (this does not impact the depth of the ciphertext).

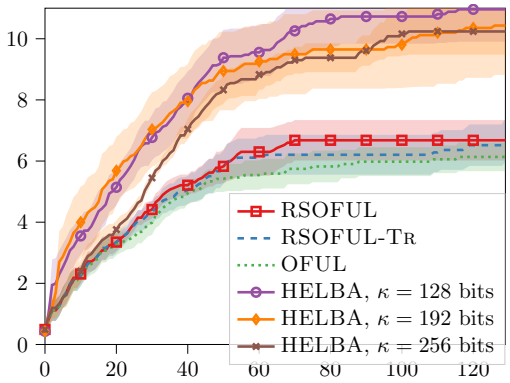


Figure 4.B.1: Regret of HELBA for $\kappa \in \{128, 192, 256\}$

Table 4.B.1: Ratio of running time for HELBA as a function of κ for the bandit problem of Sec. 4.2.5. We use the running time with $\kappa = 128$ bits and $T = 130$ steps as a reference to compute the ratio between this time and the total time for $\kappa \in \{192, 256\}$.

κ (Bits)	Ratio Execution Time
128	1
192	1.016
256	1.026

4.B.6.3 Influence of the Security Level

Finally, we investigate the influence of the security level κ on the running time and regret of HELBA. As mentioned in Sec. 4.2.5 the security parameter κ ensures that an attacker has to perform at least 2^κ operations in order to decrypt a ciphertext encrypted using an homomorphic encryption scheme. But, the security parameter also has an impact on the computational efficiency of our algorithm. Indeed the dimension N of the ciphertext space, i.e., the degree of the polynomials in $\mathbb{Z}[X]/(X^N + 1)$, increases with the multiplicative depth D and κ. However, this means that our algorithm has to compute operations with polynomials of higher dimensions hence more computationally demanding.

The library PALISADE allows us to choose $\kappa \in \{128, 192, 256\}$. We executed HELBA with the same parameter and the same environment of Sec. 4.2.5 except for the parameter κ which now varies in $\{128, 192, 256\}$. First, we investigate the regret of for each parameter κ, this parameter should have no impact on the regret HELBA, as showed in Fig. 4.B.1.

Second, we investigate the running time for each $\kappa \in \{128, 192, 256\}$. Table 4.B.1 shows the ratio between the total computation time of 130 steps using the environment described in Sec. 4.2.5 with HELBA for different security parameters and $\kappa = 128$ bits. In order to investigate only the effect of the parameter κ, the results in Table 4.B.1 are expressed as a ratio. For reference, the total time for $T = 130$ steps and $\kappa = 128$ bits was 20 hours and 39 minutes. As we observe in Table 4.B.1 the impact on the security parameter is around 1% and 2% of the total computation time for 128 bits. This increase in computation time represents between 20 and 40 minutes of computation which in some applications can be prohibitive.

Chapter 5

Final Conclusion and Perspectives

5.1 Conclusion and Contributions of this Thesis

The goal of this thesis was to study different critical constraints in deploying Reinforcement Learning algorithms in different application. While the three problems we focused on are different in nature, they all involve a tight control on the exploration process done by standard RL algorithms.

In the first chapter, we focused on controlling the performance of the bandit/RL algorithm. The first constraint we studied, see Section 2.1, is a bandit problem where the algorithm can not observe true features but only scores produced by different models. The goal was then to be able to get a better performance than simply taking the actions with the best scores (potentially biased) computed by other machine learning models. We showed in this section that under some assumptions on the reward it is possible to recover a $\mathcal{O}\left(\sqrt{T}\right)$ regret. In the next two sections, Section 2.2 and Section 2.3, we studied the problem of *conservative exploration* in bandits and Reinforcement Learning. In the former, we improved the empirical performance of existing algorithms (at the time) and improved the regret in terms of constant quantities. In Section 2.3, we extended this notion of conservative exploration to Reinforcement Learning both in the Average Reward setting and Finite-Horizon setting. The main lesson from this work is that conservative exploration is more difficult in the Average Reward setting than in the Finite-Horizon one, as the regret scales differently with the parameter α of conservativeness.

In the second chapter, we focused on the challenge of privacy on Reinforcement Learning algorithms. We defined the notion of Local Differential Privacy in the Finite-Horizon setting, showed a regret lower-bound for this constraint and proposed two different privacy-compliant algorithms, $\mathrm{LDP\text{-}OBI}$ and $\mathrm{LDP\text{-}PSRL}$. We finally showed a regret upper-bound for $\mathrm{LDP\text{-}OBI}$. This upper-bound is weaker than in the non-private case but matches the optimal dependence in the privacy parameter ε. In the Section 3.2, we used a recent advances in Differential Privacy to bridge the gap between two notions of Joint Differential Privacy and Local Differential Privacy in the linear contextual bandit case. We showed a regret that interpolates the two existing results in JDP and LDP.

In the final chapter of this thesis, we looked into the security aspect of Reinforcement Learning algorithms. In particular, we first focused, see Section 4.1, on adversarial attacks in linear contextual bandit algorithms. We showed that for a small price an adversary can break a typical bandit algorithm like $\textsc{LinUCB}$ or $\textsc{LinTS}$, forcing them to have a linear regret when controlling the reward or all the contexts presented to the algorithm. Finally, we focused on building an E2E encrypted bandit algorithm using homomorphic encryption, with the objective that the bandit algorithm can not access the data of the different users but still provide value to them in the recommended action. We showed that with a small increase in the regret and an increased computational complexity, it is possible to build an E2E encrypted linear contextual bandit algorithm.

In this thesis, we only focused on three different constraints but there is a myriad of other constraints when deploying a RL algorithm. For instance something we did not focused on in this thesis but is also a fundamental aspect to deploy RL algorithms is safety. The safe RL literature has been very active in the recent years and a lot of work trying to bring safety to real-world algorithms has been made.

Chapter 6

Résumé étendu

Contents

6.1 Aperçu

L'apprentissage supervisé a alimenté une quantité substantielle d'applications (Sharma et al., 2017; Wang et al., 2016; Ghazanfar and Prugel-Bennett, 2010) dont certaines où d'aucuns pourraient prétendre que les algorithmes dit en ligne pourraient être plus adaptés (Sikka et al., 2012; Netflix). Cependant, ces dernières années, de nombreux efforts ont été déployés pour adapter et utiliser des algorithmes en ligne pour des applications telles que les problèmes de recommandation (Spotify; le et al., 2019). À l'avant-garde de cette poussée, il y a l'apprentissage par renforcement (RL) et les algorithmes de bandits[1] (Shi et al., 2018; Zhao et al., 2019; Guo et al., 2020). En effet, ces paradigmes d'apprentissage permettent aux praticiens de prendre en compte la valeur à long terme d'un client (Wang et al., 2019a), ou d'autres objectifs qui sont fonction des interactions précédentes d'un utilisateur donné avec un produit. Néanmoins, ce changement n'est pas sans difficulté. Une des principales différences, sur laquelle nous nous concentrons dans cette thèse, entre les algorithmes basés sur l'apprentissage supervisé et ceux basés sur le RL est l'*exploration*.

Pour comprendre ce qu'est l'exploration dans les algorithmes de bandits et RL, considérons l'exemple classique, décrit dans (Lattimore and Szepesvári, 2020), où l'on fait face à deux machines à sous différentes avec des probabilités de gain différentes. Nous pouvons jouer à l'une des deux machines 10 fois dans le but de maximiser le nombre total de gains après 10 tirages. Imaginez qu'après 6 tirages aléatoires, vous observiez que la première machine a un taux de gain plus élevé que la seconde. Comment doit-on répartir les tirages restants? Doit-on s'engager à ne tirer que sur la première machine ou essayer la deuxième machine à nouveau? Le premier choix exploite les résultats passés et prend une décision gloutonne basée sur ceux-ci. Alors que la deuxième option implique que l'on continue d'explorer toutesles actions potentielles, peut-être en supposant que les résultats actuels sont dus au hasard. Le compromis entre l'optimisation du nombre de tirages gagnants et la collecte du plus de données possible sur le taux de gain de chaque machine est nommé, dans la littérature, *the explore-exploit tradeoff*. Lors du déploiement d'un algorithme RL, l'exploration peut être problématique pour différentes raisons. Par exemple, même si cela peut conduire à de meilleurs résultats à long terme, l'exploration peut conduire à de moins bons résultats à court terme. On peut penser à

[1] Dans cette thèse, nous pouvons parfois utiliser le terme d'algorithmes de RL pour désigner à la fois des algorithmes d'apprentissage par renforcement et des algorithmes de bandits

l'exemple où un algortihme recommande un film en allemand (sans sous-titres) à un spectateur parlant uniquement français. Dans le pire des cas, l'exploration peut même être nocive lorsqu'elle pousse un algorithme à recommander un film interdit au moins de 18 ans à un enfant de 13 ans. Ceci illustra la nécessité de limiter l'exploration lors du déploiement d'algorithmes RL.

Cette thèse est motivée par les questions autour du processus d'exploration pour l'Apprentissage par Renforcement et de bandits. Nous espérons qu'à travers les différentes contributions présentées ici, nous avons clarifié certaines questions pour le déploiement d'algorithmes de RL pour des applications réelles mais aussi présenté des directions de recherche prometteuses pour faciliter d'avantage l'utilisation du RL.

6.2 Bandits et Bandits contextuels

Avant toute choses, nous proposons une introduction à l'apprentissage par renforcement tabulaire et aux bandits contextuels qui sont les deux principaux scénario d'apprentissage étudiés dans cette thèse. Nous commençons par une brève introduction aux Multi-Armed Bandit qui est le cadre de base pour comprendre le scénario d'interaction entre l'algorithme d'apprentissage et le reste de son environnement.

6.2.1 Bandits à plusieurs bras

Un problème de bandits à plusieurs bras est un problème de prise de décision en ligne. Un algorithme (souvent appelé agent) doit choisir une action parmi un ensemble de $K \in \mathbb{N}^\star$ actions possibles pour chaque instant $t \geq 1$ Chaque action génère une récompense, $r_{t,a}$ (qui peut être stochastique ou fixé par un adversaire) l'objectif de l'apprenant est de maximiser les récompenses cumulées. Dans cette thèse, nous nous concentrons sur le bandit stochastique ou problème d'apprentissage par renforcement. C'est-à-dire que nous supposons que pour chaque action $a \leq K$ et étape t, la récompense $r_{t,a} \sim \nu_a$ où ν_a est une distribution *inconnue* (pour l'algorithme) sur $\mathbb{R}$. L'objectif de l'algorithme est alors de maximiser la récompense cumulée, $\sum_t \mathbb{E}_{r\sim\nu_{a_t}}(r)$. Ou de manière équivalente de minimiser le regret, $R(T) = \sum_{t=1}^T \mathbb{E}_{r\sim\nu_{a^\star}}(r) - \mathbb{E}_{r\sim\nu_{a_t}}(r)$ pour tout $T \geq 1$, où $a^\star := \arg\max_{a\in\{1,\dots,K\}} \mathbb{E}_{r\sim\nu_a}(r)$ est la meilleure action en ayant connaissance des distributions $(\nu_a)_a$.

Comme expliqué dans la section précédente, minimiser le regret nécessite que l'apprenant trouve un équilibre entre l'exploration des bras et l'exploitation des meilleurs bras jusqu'à présent. Il existe une grande variété d'hypothèses qui rendent le problème du bandit à plusieurs bras plus ou moins complexe. Par exemple, nous pouvons supposer que la distribution des récompenses est sous-gaussienne ou avec des queues de distributions lourdes (Bubeck and Cesa-Bianchi, 2012; Zhuang and Sui, 2021). De façon générale, nous faisons l'hypothèse que les distributions $(\nu_a)_{a\leq K}$ sont $\sigma > 0$ sous gaussiennes. C'est-à-dire que pour tout $\lambda \in \mathbb{R}$,

$$\forall a \leq K, \qquad \mathbb{E}_{X\sim\nu_a}\left(e^{\lambda(X-\mu_a)}\right) \leq e^{\frac{\lambda^2\sigma^2}{2}} \tag{6.1}$$

où $\mu_a = \mathbb{E}_{X\sim\nu_a}(X)$ for all $a \leq K$. Nous supposons de plus que l'algorithme connaît la constante σ.

Les deux algorithmes les plus connus pour résoudre ce problème sont l'algorithme Upper Confidence Bound (UCB), Alg. 41, et Thompson Sampling (TS), Alg. 42.

Upper Confidence Bound. L'algorithme UCB tiens son nom du fait que ce dernier construit des intervalles de confiance autour des moyennes des distributions de chaque bras et selectionne le bras qui maximise la partie supérieur de l'interval de confiance associé. Aprés t étapes, notons $N_a(t)$ le nombre de fois où l'algorithme a décidé de selectionner le bras a, il est alors possible de montrer (Lattimore and Szepesvári, 2018) aue pour tout $\delta \in (0,1)$ on a:

$$\mathbb{P}\left(\left|\frac{1}{N_a(t)}\sum_{l=1}^t \mathbb{1}_{\{a_l=a\}} r_{l,a} - \mathbb{E}_{X\sim\nu_a}(X)\right| \geq \sqrt{\frac{2\sigma^2\ln(2/\delta)}{N_a(t)}}\right) \leq \delta \tag{6.2}$$

Ainsi en se basant sur cette equation nous avons qu'avec probabilité au moins $1-\delta$:

$$\mathbb{E}_{X\sim\nu_a}(X) \in \left[\frac{1}{N_a(t)}\sum_{l=1}^t \mathbb{1}_{\{a_l=a\}} r_{l,a} - \sqrt{\frac{2\sigma^2\ln(2/\delta)}{N_a(t)}}, \frac{1}{N_a(t)}\sum_{l=1}^t \mathbb{1}_{\{a_l=a\}} r_{l,a} + \sqrt{\frac{2\sigma^2\ln(2/\delta)}{N_a(t)}}\right] \tag{6.3}$$

Ce type d'algorithme est dit optimistique car il construit une evaluation positivement biaisée de la vraie moyenne de la distribution des récompenses.

Thompson Sampling (TS). Le second type d'algortihmes souvent utilisés pour les problémes de bandits à plusieurs bras sont des algorithmes bayésien dont le plus connu est Thompson Sampling (Thompson, 1933). Étant donné un prior P sur les distributions $(\nu_a)_a$ puis génère un posterior Q_t grâce aux différentes données collectées . Pour selectionner un bras, Thompson Sampling genere un nouveau problème de bandit à partir du posterior Q_t puis selectionne le meilleur bras selon ce nouveau problème.

Algorithm 41: Algorithme Upper Confidence Bound (UCB)

Input: Horizon: T, paramètre de sous-gaussianité: $\sigma > 0$, probabilité: $\delta \in (0,1)$, nombre de bras: K
Pour chaque bras, $a \leq K$ definir $N_a = 0$ et $R_a(t) = 0$
for $a = 1, \dots, K$ **do**
  Tirer le bras a, recevoir la récompense $r_{a,a}$, incrémente $N_a = N_a + 1$ et $R_a = R_a + r_{a,a}$
for $t = K+1, \dots, T$ **do**
  Pour chaque bras a calculer l'index $I_a(t) = \frac{R_a}{N_a} + \sqrt{\frac{8\sigma^2 \ln(2t/\delta)}{N_a}}$
  Selectionner le bras $a_t = \arg\max_a I_a(t)$, observer récompense r_{t,a_t}, incrémenter $N_{a_t} = N_{a_t} + 1$ et $R_{a_t} = R_{a_t} + r_{t,a_t}$

Algorithm 42: L'algorithme Thompson Sampling (TS)

Input: Nombre de bras: K, Prior: P, Horizon: T
Initialiser le posterior $Q_1 = P$
for $t = 1, \dots, T$ **do**
  Genérer le problème de bandits $(\nu_{t,a})_{a \leq K} \sim Q_t(. \mid a_1, \dots, a_{t-1}, r_{1,a_1}, \dots r_{t-1,a_{t-1}})$
  Selectionner $a_t = \arg\max_a \mathbb{E}_{X \sim \nu_{t,a}}(X)$, observer la récompense r_{t,a_t}

Il est possible de prouver que Thompson Sampling tout comme UCB sont presque optimal. C'est-à-dire que le regret,

$$R_T = \sum_{t=1}^{T} \mathbb{E}_{X \sim \nu_{a^\star}}(X) - \mathbb{E}_{X \sim \nu_{a_t}}(X) \leq \mathcal{O}\left(\sqrt{T}\right)$$

, où $a^\star = \arg\max_a \mathbb{E}_{X \sim \nu_a}(X)$. Ce qui est la meilleur dépendence possible en T comme plusieurs résulats l'ont montrés (Abbasi-Yadkori et al., 2011; Abeille et al., 2017).

La preuve de ce résultat pour UCB est une bonne exercice de compréhension de l'interaction entre l'algorithme et l'environnement. En effet la première étape est de décomposer le regret comme:

$$R_T = \sum_{t=1}^{T} \mathbb{E}_{X \sim \nu_{a^\star}}(X) - \mathbb{E}_{X \sim \nu_{a_t}}(X) \tag{6.4}$$

$$= \sum_{a=1}^{K} \Delta_a \mathbb{E}(N_a(T)) \tag{6.5}$$

où $\Delta_a = \mathbb{E}_{X \sim \nu_{a^\star}}(X) - \mathbb{E}_{X \sim \nu_a}(X)$ est la différence d'espérance de gain entre le bras a et le bras optimal $a^\star$ (supposé unique dans la suite). Il reste maintenant à borner l'espérance du nombre de fois où le bras est selectionné. Or cette esperance peut se réecrire comme:

$$\mathbb{E}(N_a(T)) = \sum_{t=1}^{T} \mathbb{E}\left(\mathbb{1}_{\{a_t = a\}}\right) \tag{6.6}$$

Mais pour un instant t si l'algorithme sélectionne le bras a à la place du bras $a^\star$ cela siginifie que:

$$I_a(t) \geq I_{a^\star}(t) \Leftrightarrow$$
$$\frac{R_a}{N_a} + \sqrt{\frac{8\sigma^2 \ln(2t/\delta)}{N_a}} \geq \frac{R_{a^\star}}{N_{a^\star}} + \sqrt{\frac{8\sigma^2 \ln(2t/\delta)}{N_{a^\star}}} \Leftrightarrow$$

Mais dans l'évènement où tout les intervalles de confiances tiennent c'est-à-dire

$$W_t = \bigcap_{a=1}^{K} \left\{ \left| \frac{R_a(t)}{N_a(t)} - \mathbb{E}_{X \sim \nu_a}(X) \right| \leq \sqrt{\frac{8\sigma^2 \ln(2t/\delta)}{N_a(t)}} \right\} \tag{6.7}$$

avec c une constante numérique. Dans cet évènement, nous avons que

$$\frac{R_a}{N_a} + \sqrt{\frac{8\sigma^2 \ln(2t/\delta)}{N_a}} \geq \frac{R_{a^\star}}{N_{a^\star}} + \sqrt{\frac{8\sigma^2 \ln(2t/\delta)}{N_{a^\star}}} \Leftrightarrow \tag{6.8}$$

$$\frac{R_a}{N_a} - \mathbb{E}_{X\sim\nu_a}(X) + \mathbb{E}_{X\sim\nu_a}(X) + \sqrt{\frac{8\sigma^2 \ln(2t/\delta)}{N_a}} \geq \frac{R_{a^\star}}{N_{a^\star}} - \mathbb{E}_{X\sim\nu_{a^\star}}(X) + \mathbb{E}_{X\sim\nu_{a^\star}}(X) + \sqrt{\frac{8\sigma^2 \ln(2t/\delta)}{N_{a^\star}}} \Rightarrow \tag{6.9}$$

$$\mathbb{E}_{X\sim\nu_a}(X) + 2\sqrt{\frac{8\sigma^2 \ln(2t/\delta)}{N_a}} \geq \mathbb{E}_{X\sim\nu_{a^\star}}(X) \Rightarrow \tag{6.10}$$

$$N_a(t) \leq \frac{32\sigma^2 \ln\left(\frac{2t}{\delta}\right)}{\Delta_a^2} \tag{6.11}$$

Ainsi nous avons que:

$$N_a(T) = 1 + \sum_{t=K+1}^{T} \mathbb{1}_{\{\ a_t = a\}} \tag{6.12}$$

$$\leq 1 + \frac{32\sigma^2 \ln\left(\frac{2T}{\delta}\right)}{\Delta_a^2} + \sum_{t=K+1}^{T} \mathbb{1}_{\left\{\ a_t = a, N_a(t) \geq \frac{32\sigma^2 \ln\left(\frac{2t}{\delta}\right)}{\Delta_a^2}\right\}} \tag{6.13}$$

$$\tag{6.14}$$

Mais par le raisonnement précédent, nous avons que:

$$\left\{ a_t = a, N_a(t) \geq \frac{32\sigma^2 \ln\left(\frac{2t}{\delta}\right)}{\Delta_a^2} \right\} \subset W_t^c$$

Par conséquent,

$$\begin{aligned}
\mathbb{E}\left(N_a(T)\right) &\leq 1 + \frac{32\sigma^2 \ln\left(\frac{2T}{\delta}\right)}{\Delta_a^2} + \sum_{t=K+1}^{T} \mathbb{P}(W_t^c) \\
&\leq 1 + \frac{32\sigma^2 \ln\left(\frac{2T}{\delta}\right)}{\Delta_a^2} + \sum_{t=K+1}^{T} \sum_{a=1}^{K} \frac{\delta}{8t^4} \\
&\leq 1 + \frac{32\sigma^2 \ln\left(\frac{2T}{\delta}\right)}{\Delta_a^2} + cK\delta
\end{aligned}$$

avec c une constante numéirque. Ainsi le regret est obtenu en multipliant l'équation ci-dessus par Δ_a et en sommant sur les bras a.

6.2.2 Bandits contextuels linéaire

Le développement des bandits contextuel est un point cardinal de la théorie des bandits grâce à l'ajout d'informations secondaires appelées contextes qui modifient le processus de génération de récompense permettant par exemple la personnalisation dans certaines applications. En général dans les bandits contextuels, la récompense est fonction d'un contexte et une action donnés à l'algorithme sans autre hypothèse sur la relation entre la récompense et le contexte/action (Beygelzimer et al., 2010, 2011; Bietti et al., 2018). Dans ce travail, nous nous intéressons au cas où cette relation est linéaire. Le problème de bandits contextuel linéaire est l'une des versions les plus étudiées du problème de bandits contextuel. Il existe deux formulations légèrement différentes du problème qui sont équivalentes mais plus ou moins confortables selon les applications.

Contextes différents selon l'action Considérons le problem de bandits linéaire contextuel standard avec $K \in \mathbb{N}$ bras. A chaque instant t, l'agent observe un contexte $x_t \in \mathbb{R}^{d\times K}$, sélectionne une action $a_t \in [\![1, K]\!]$ et observe une récompense: $r_{t,a_t} = \langle \theta^\star, x_{t,a_t} \rangle + \eta_{a_t}^t$ où $\theta^\star \in \mathbb{R}^d$ est un vecteur inconnu et $\eta_{a_t}^t$ est une bruit σ^2-sous-gaussien de moyenne nulle conditionnellement indépendante par rapoort à l'historique. Aucune hypothèse n'est faite sur la façon dont les contextes sont présentés à l'agent. C'est-à-dire qu'ils pourraient être échantillonnés de manière stochastique ou selectionnés par un adversaire. Dans la plupart des cas, les contextes sont supposés avoir tous une norme majorée par une constante connue L et l'algorithme a aussi accès à une borne supérieure sur la norme du vecteur inconnu $\theta^\star$. Le but de l'agent est de minimiser le regret cumulé après les étapes T $R_T = \sum_{t=1}^{T} \langle \theta^\star, x_{t,a_t^\star} \rangle - \langle \theta^\star, x_{t,a_t} \rangle$, où $a_t^\star := \arg\max_a \langle \theta^\star, x_{t,a} \rangle$.

Contextes independant de l'action Dans cette formulation à chaque instant t, l'agent reçoit un contexte $x_t \in \mathbb{R}^d$. La principale différence avec le formalisme ci-dessus est que lorsque l'agent sélectionne une action $a_t \in [\![1, K]\!]$, il observe une récompense: $r_{t,a_t} = \langle \theta_{a_t}, x_t \rangle + \eta^t_{a_t}$ où pour chaque bras a, $\theta_a \in \mathbb{R}^d$ est un vecteur inconnu et $\eta^t_{a_t}$ est une bruit σ^2-sous-gaussien de moyenne nulle conditionnellement indépendante par rapoort à l'historique. Le regret peut maintenant être écrit comme, $R_T = \sum_{t=1}^T \langle \theta_{a^\star_t}, x_t \rangle - \langle \theta_{a_t}, x_t \rangle$, où $a^\star_t := \arg\max_a \langle \theta_a, x_t \rangle$.

Or il est possible de montrer (Lattimore and Szepesvári, 2018) que ces deux formulations sont équivalentes. En effet si l'on a des contextes indépendant de l'action il suffit de définir θ^{star} comme la concaténation des vecteurs $(\theta^\star_a)_{a \leq K}$ et les nouveaux contextes en plongeant le contexte original dans un espace de dimension plus grande en ajoutant des zéros. Si l'on est dans le scénario contraire, il suffit de transformer la matrice x_t en un vecteur et définir les vecteurs $\theta^\star_a$ en concaténant le vecteur $\theta^\star$ et des vecteurs nulles au bonnes coordonées. Dans cette thèse nous utilisons indifféremment l'une ou l'autre des formulations.

Comme pour les bandits à plusieurs les deux algorithmes les plus connnus pour ce type de problème sont basés sur l'optimisme et une formulation bayésienne. Il s'agit de LinUCB (voir Alg.43) et LinTS (voir Alg. 44). Tous les algorithmes que nous considérons utilisent des modèles disjoints pour construire l'estimation des vecteurs $(\theta^\star_a)_{a \in [\![1,K]\!]}$.

Algorithme LinUCB. L'algortihme LinUCB construit une ellipsoïde de confiance autour de chacun des vecteurs $\theta^\star_a$ en utilisant une régression Ridge. Cet algorithme se base sur un résultat démontrer dans Lattimore and Szepesvári (2018) qui montre qu'il est possible de construire une ellipsoïde de confiance pour tout niveau de confiance δ.

Algorithm 43: Algorithme LinUCB

Input: Régularisation: λ, Nombre de bras: K, Horizon: T, Norme contextes maximale: L, Norme maximale $(\theta^\star_a)_a$:D
Initialiser pour chaque bras a, $\bar{V}_a^{-1}(t) = \frac{1}{\lambda} I_d$, $\hat{\theta}_a(t) = 0$ et $b_a(t) = 0$
for $t = 1, \ldots, T$ **do**
  Observer le contexte x_t
  Calculer $\beta_a(t) = \sigma \sqrt{d \log\left(\frac{1 + N_a(t) L^2/\lambda}{\delta}\right)}$ avec $N_a(t)$ le nombre de tirages du bras a
  Tirer le bras $a_t = \arg\max_a \langle \hat{\theta}_a(t), x_t \rangle + \beta_a(t) ||x_t||_{\bar{V}_a^{-1}(t)}$
  Observez la récompense r_t et mettez à jour les paramètres $\hat{\theta}_a(t)$ et $\bar{V}_a^{-1}(t)$ tels que :

$$\bar{V}_{a_t}(t+1) = \bar{V}_{a_t}(t) + x_t x_t^\intercal, \quad b_{a_t}(t+1) = b_{a_t}(t) + r_t x_t, \quad \theta_{a_t}(t+1) = \bar{V}_{a_t}^{-1}(t+1) b_{a_t}(t+1)$$

Algorithme LinTS. L'algortihme LinTS que nous présentons ici se base lui aussi sur une régression lineaire afin d'estimer les paramètres du posterior car nous supposons un prior gaussien sur les vecteurs $(\theta^\star_a)_a$.

Algorithm 44: Linear Thompson Sampling avec un prior gaussien

Input: régularisation λ, nombre de bras K, horizon T, variance υ
Initialiser pour chaque bras a, $\bar{V}_a^{-1}(t) = \lambda I_d$ et $\hat{\theta}_a(t) = 0$, $b_a(t) = 0$
for $t = 1, \ldots, T$ **do**
  Observer le contexte x_t
  Dessinez $\tilde{\theta}_a \sim \mathcal{N}(\hat{\theta}_a(t), \upsilon^2 \bar{V}_a^{-1}(t))$
  Tirer le bras $a_t = \arg\max_{a \in [\![1,K]\!]} \left\langle \tilde{\theta}_a, x_t \right\rangle$
  Observez la récompense r_t et mettez à jour les paramètres $\hat{\theta}_a(t)$ et $\bar{V}_a^{-1}(t)$

$$\bar{V}_{a_t}(t+1) = \bar{V}_{a_t}(t) + x_t x_t^\intercal, \quad b_{a_t}(t+1) = b_{a_t}(t) + r_t x_t, \quad \theta_{a_t}(t+1) = \bar{V}_{a_t}^{-1}(t+1) b_{a_t}(t+1)$$

Il est possible de prouver que ces deux algorithmes, tout comme dans le cas des bandits à plusieurs bras, ont un regret sous-linéaire, c'est-à-dire

$$R_T = \sum_{t=1}^T \langle \theta_{a^\star_t}, x_t \rangle - \langle \theta_{a_t}, x_t \rangle \leq \mathcal{O}\left(\sqrt{T}\right)$$

Or il s'agit d'un résultat connu dans la littérature que la dépence en racine carré en T est optimale pour les problèmes de bandits contextuels lineaire.

6.3 Apprentissage par Renforcement

L'apprentissage par renforcement (RL) est une généralisation du cadre des bandits en incorporant une notion d'état, similaire aux contextes dans le cas du bandit contextuel. Au fil des ans, différentes formulations de RL ont été développées. La plupart des travaux sur l'apprentissage par renforcement profond, c'est-à-dire utilisant des réseaux de neurones, se déroulent dans un cadre appelé l'horizon infini pondéré (Mnih et al., 2013). Dans cette thèse, nous nous concentrons sur deux scénarios souvent plus utilisés dans la littérature théorique d'apprentissage par renforcement: l'horizon fini (dans la section 3.1 et section 2.C.3) et l'apprentissage par renforcement à récompense moyenne (dans la section 2.3).

6.3.1 Apprentissage par Renforcement à Horizon Fini

Considérons un processus décision Markovien à horizon fini (Puterman, 1994, Chp. 4) $M = (\mathcal{S}, \mathcal{A}, p, r, H)$ avec espace d'état $\mathcal{S}$ et espace d'action $\mathcal{A}$. Chaque paire état-action est caractérisée par une distribution de récompense d'espérance $r(s,a)$ à support dans $[0,1]$ ainsi qu'une distribution de transition des états $p(\cdot|s,a)$. L'interaction entre l'algorithme et l'environnement à lieu sous forme d'épisodes. Au début de chaque un état s_0 est générée selon une distribution μ puis l'algorithme recommande une action, a_0, pour l'état s_0. Celui-ci obtient une récompense $r_0(s_0,a_0)$. L'état evolue alors vers un état s_1 tel que $s_1 \sim p(.\mid s_0, a_0)$. Ce processus dure pendant H étapes avant la fin de l'épisode.

On note $S = |\mathcal{S}|$ et $A = |\mathcal{A}|$ le nombre d'états et d'actions, et H l'horizon d'un épisode. L'objectif d'un algorithme est alors de calculer une règle de décision d'action pour chaque état. Une règle de décision markovienne randomisée $d : \mathcal{S} \to P(\mathcal{A})$ fait correspondre les états aà une distribution sur l'espace d'actions. Une politique π est une séquence de règles de décision, i.e., $\pi = (d_1, d_2, \ldots, d_H)$. On note Π^{MR} (resp.Π^{MD}) l'ensemble des politiques markovienne randomisées (resp. déterministes). Afin d'ordonner les différentes politiques nous utilisons une fonction appelé fonction valeur, V_1^π, définie par:

$$\forall t \in [H], \forall s \in \mathcal{S} \qquad V_t^\pi(s) = \mathbb{E}^\pi \left[\sum_{l=t}^{H} r_l(s_l, a_l) \mid s_t = s \right]$$

où l'espérance est prise par rapporrt à l'aléatoire du modèle et la politique (c'est-à-dirw $a_l \sim d_l(s_l)$). Cette fonction donne la récompense totale attendue que l'on pourrait obtenir en suivant la politique π à partir de l'état s, au temps t. L'un des résultats fondamental dans ce scénario d'apprentissage est l'existence d'une politique deterministe optimale $\pi^\star \in \Pi^{\text{MD}}$ (Puterman, 1994, Sec. 4.4) pour lequel $V_t^\star = V_t^{\pi^\star}$ satisfait les *équations d'optimalité*:

$$\forall t \in [H], \forall s \in \mathcal{S}, \qquad V_t^\star(s) = \max_{a \in \mathcal{A}} \left\{ r_t(s,a) + p(\cdot|s,a)^\mathsf{T} V_{t+1}^\star \right\} := L_t^\star V_t^\star \tag{6.15}$$

où $V_{H+1}^\star(s) = 0$ pour tout état $s \in \mathcal{S}$. Lorsque la récompense et les transitions sont connues, la fonction valeur peut être calculée en utilisant un algorithme de programmation dynamique (e.g., Puterman, 1994; Bertsekas, 1995). Étant donné une politique $\pi \in \Pi^{\text{MD}}$, la fonction de valeur associée satisfait les *équations d'évaluation* $V_t^\pi(s) := L_t^\pi V_{t+1}^\pi(s) = r(s, d_t(s)) + p(\cdot|s, d_t(s))^\mathsf{T} V_{t+1}^\pi$. La politique optimale est ainsi définie comme $\pi^\star = \arg\max_{\pi \in \Pi^{\text{MD}}} \{L_t^\pi V_t^\star\}$, $\forall t \in [H]$.

Dans ce qui suit, nous supposons que l'agent d'apprentissage connaît $\mathcal{S}$, $\mathcal{A}$ et $r_{\max} = 1$, tandis que la récompense et la distribution du prochain état sont *inconnues* et doivent être estimées en ligne. Étant donné un nombre fini d'épisodes K, on évalue les performances d'un algorithme d'apprentissage $\mathfrak{A}$ par son regret cumulé

$$R(\mathfrak{A}, K) = \sum_{k=1}^{K} V_1^\star(s_{1,k}) - V_1^{\pi_k}(s_{1,k})$$

où π_k est la politique exécutée par l'algorithme à l'épisode k.

Un algorithme classique pour résoudre ce problème d'apprentissage par renforcement est l'algorithme UCB-VI (Azar et al., 2017b). Ce dernier se base sur le principe de l'optimisme tout comme UCB. Pour cela, l'algorithme construit deux intervalles de confiance. Un autour des récompenses de chaque pair état-action et de chaque distribtion de transition d'eétats. Pour tout $\delta \in (0,1)$,

$$\forall (s,a) \in \mathcal{S} \times \mathcal{A}\ \forall v \in \mathbb{R}^S \qquad |(p(\cdot|s,a) - \widehat{p}_k(\cdot|s,a))^\mathsf{T} v| \le \beta_p^k(s,a) \quad \text{et} \quad |r(s,a) - \widehat{r}_k(s,a)| \le \beta_r^k(s,a)$$

où $\beta_r^k(s,a)$ et $\beta_p^k(s,a)$ sont les tailles des intervalles de confiance. De plus, $\widehat{p}_k$ et $\widehat{r}_k$ sont les moyennes empiriques des probabilités de transition et des récompenses observés par l'algorithme au début du $k$ème épisode, c'est-à-dire après $k-1$ épisodes. En notant,

$$\mathcal{H}_k = \Big\{ s_{1,0}, a_{1,0}, r_{1,0}(s_{1,0}, a_{1,0}), \ldots, s_{1,H-1}, a_{1,H-1}, r_{1,H-1}(s_{1,H-1}, a_{1,H-1}),$$
$$\ldots, s_{k-1,0}, a_{k-1,0}, r_{k-1,0}(s_{k-1,0}, a_{k-1,0}), \ldots, s_{k-1,H-1}, a_{k-1,H-1}, r_{k-1,H-1}(s_{k-1,H-1}, a_{k-1,H-1}) \Big\}$$

l'ensemble des états, actions et récompenses observés par l'algorithme après $k-1$ épisodes, whp_k et $\widehat{r}_k$sont définis comme

$$\forall s' \in \mathcal{S}\ \forall (s,a) \in \mathcal{S}\times\mathcal{A} \qquad \widehat{p}_k(s' \mid s,a) = \frac{\sum_{l=1}^{k-1}\sum_{h=0}^{H-2} \mathbb{1}_{\{s_{l,h+1}=s', s_{l,h}=s, a_{l,h}=a\}}}{\sum_{l=1}^{k-1}\sum_{h=0}^{H-1} \mathbb{1}_{\{s_{l,h}=s, a_{l,h}=a\}}}$$

$$\forall (s,a) \in \mathcal{S}\times\mathcal{A} \qquad \widehat{r}_k(s,a) = \frac{1}{\sum_{l=1}^{k-1}\sum_{h=0}^{H-1} \mathbb{1}_{\{s_{l,h}=s, a_{l,h}=a\}}} \sum_{l=1}^{k-1}\sum_{h=0}^{H-1} r_{l,h}(s_{l,h},a_{l,h}) \mathbb{1}_{\{s_{l,h}=s, a_{l,h}=a\}}$$

À ce stade, nous devons définir explicitement l'inégalité de concentration utilisée pour construire les ensembles de confiance B_r^k et B_p^k. Pour chaque $(s,a) \in \mathcal{S}\times\mathcal{A}$, on définit $\beta_r^k(s,a)$ tel que:

$$\forall k \geq 1, \qquad B_r^k(s,a) \subset [\widehat{r}_k(s,a) - \beta_r^k(s,a), \widehat{r}_k(s,a) + \beta_r^k(s,a)] \tag{6.16}$$

où $\widehat{r}_l(s,a)$ est la moyenne empirique de la récompense reçue lors de la visite des paires état-action (s,a) au début de l'épisode l. Pour chaque $(s,a) \in \mathcal{S}\times\mathcal{A}$, nous définissons $\beta_p^l(s,a)$ comme:

$$B_p^k(s,a) = \left\{p \in \Delta_S : \|p(\cdot|s,a) - \widehat{p}_k(\cdot|s,a)\|_1 \leq \beta_p^k(s,a)\right\} \tag{6.17}$$

avec $\widehat{p}_l$ est la moyenne empirique des transitions observées. Le choix de ces β_r^l et β_p^l se fait grâce à des inégalités de concentration telles que l'événement Γ^c est vrai avec une probabilité suffisamment élevée. Dans ce qui suit, nous utilisons:

$$\forall s,a \qquad \beta_r^k(s,a) = \sqrt{\frac{21SA\ln\left(\frac{5SAkH}{\delta}\right)}{2\max\{1, N_k(s,a)\}}} \tag{6.18}$$

$$\beta_p^k(s,a) = S\sqrt{\frac{28A\ln\left(\frac{5SAkH}{\delta}\right)}{\max\{1, N_k(s,a)\}}} \tag{6.19}$$

où $N_k(s,a) = \sum_{l=1}^{k-1}\sum_{h=0}^{H-1} \mathbb{1}_{\{s_{l,h}=s, a_{l,h}=a\}}$. Pour d'autres choix de β_r^l et β_p^l, voir (Lazaric et al., 2019).

Algorithm 45: Algorithme UCB-VI

Input: $\delta \in (0,1)$, $\mathcal{S}$, $\mathcal{A}$, H
Définir $\mathcal{H} = \emptyset$, $\mathcal{S}_0 = \emptyset$ et $\mathcal{S}_0^c = \emptyset$
for *Épisodes* $k = 1, 2, \ldots$ **do**
  Définir $V_H(s) = 0$ pour tout les états $s \in \mathcal{S}$
  for $h = H-1, \ldots, 0$ **do**
    Calculer $V_h(s) = \max_a \widehat{r}_k(s,a) + \beta_r^k(s,a) + \widehat{p}(\cdot \mid s,a)^\mathsf{T} V_{h+1} + \beta_p^k(s,a)$
    Définir la règle de décision $\pi_{k,h}(s) = \arg\max_a \widehat{r}_k(s,a) + \beta_r^k(s,a) + \widehat{p}(\cdot \mid s,a)^\mathsf{T} V_{h+1} + \beta_p^k(s,a)$
for $h = 0, \ldots, H-1$ **do**
  Exécutez $a_{k,h} = \pi_{k,h}(s_{k,h})$, obtenez la récompense $r_{k,h}$ et observez $s_{k,h+1}$.

Le regret de cet algorithme tout comme UCB et LinUCB pour les bandits, est optimal dans sa dépendence vis-à-vis du nombre d'épisodes. Azar et al. (2017b) montre que le regret de UCB-VI est borné avec très grande probabilité par:

$$R_K = \sum_{k=1}^{K} V_0^\star(s_{k,0}) - V_0^{\pi_k}(s_{k,0}) \leq \mathcal{O}\left(\sqrt{K}\right) \tag{6.20}$$

L'apprentissage pqr renforcement à horizon fini procure un scénario d'apprentissage versatile qui correspond à plusieurs applications. Cependant comme expliqué dans Puterman (1994) certaines applications tel que le contrôle d'inventaire peuvent nécessiter un formalisme plus complexe appelé Apprentissage par Renforcement à Récompense Moyenne.

6.3.2 Apprentissage par Renforcement à Récompense Moyenne

Le scénaro d'apprentissage par renforcement à récompense moyenne est défini par un processus de décision de Markov (Puterman, 1994, Sec. 8.3) (MDP) $M = (\mathcal{S}, \mathcal{A}, p, r)$ avec espace d'état $\mathcal{S}$ et espace d'action $\mathcal{A}$. Chaque couple état-action (s,a) est caractérisé par une distribution de récompense de moyenne $r(s,a)$ et à support dans $[0, r_{\max}]$, et une distribution de transition $p(\cdot|s,a)$ sur les états suivants. On note $S = |\mathcal{S}|$ et $A = |\mathcal{A}|$ le nombre d'états et d'actions.

Une politique aléatoire de Markov stationnaire $\pi : \mathcal{S} \to P(\mathcal{A})$ associe les états aux distributions sur les actions. L'ensemble des politiques stationnaires randomisées (resp. déterministes) est noté Π^{SR} (resp. Π^{SD}).

La principale différence entre ce scénario et le précédent est l'absence de notion d'épisodes. C'est-à-dire que les états évoluent toujours seulement en fonction de l'état précédent et de l'action précédente. Il n'y a pas comme préceddment une distribution μ pour générer un état initiale à l'épisode. Ainsi, une mauvaise décision de l'algorithme peut le mner à rester coincé dans une région de l'espace d'état à faible récompense.

Toute politique $\pi \in \Pi^{\mathsf{SR}}$ a une *récompense moyenne à long terme* (ou gain) associée et une *fonction de biais* définie comme

$$g^\pi(s) := \lim_{T\to+\infty} \mathbb{E}_s^\pi\left[\frac{1}{T}\sum_{t=1}^{T} r(s_t,a_t)\right] \quad \text{et}$$

$$h^\pi(s) := \underset{T\to+\infty}{C\text{-}\lim}\, \mathbb{E}_s^\pi\left[\sum_{t=1}^{T}\Big(r(s_t,a_t) - g^\pi(s_t)\Big)\right],$$

où $\mathbb{E}_s^\pi$ désigne l'espérance sur les trajectoires générées à partir de $s_1 = s$ avec $a_t \sim \pi(s_t)$. Le biais $h^\pi(s)$ mesure la différence totale attendue entre la récompense et la récompense stationnaire par une limite de *Cesaro* (noté par $C\text{-}\lim$). On note $sp(h^\pi) := \max_s h^\pi(s) - \min_s h^\pi(s)$ le *span* (ou range) de la fonction de biais.

Dans cette thèse nous faisons ue hypothèse cruciale à propos du processus de décision markovien sous-jacent.

Assumption 16. *Le MDP M est* ergodique. *C'est-à-dire que pour toute politique de décision π il est possible de rejoindre deux états différents en suivant la politique π.*

Pour un MDP ergodique, toute politique $\pi \in \Pi^{\mathsf{SR}}$ a un gain *constant*, i.e., $g^\pi(s) = g^\pi$ pour tout $s \in \mathcal{S}$. Il existe une politique $\pi^\star \in \arg\max_\pi g^\pi$ pour laquelle $(g^\star, h^\star) = (g^{\pi^\star}, h^{\pi^\star})$ satisfont les *équations d'optimalité*,

$$h^\star(s) + g^\star = Lh^\star(s) := \max_{a\in\mathcal{A}}\{r(s,a) + p(\cdot|s,a)^\mathsf{T} h^\star\},$$

où L est l'opérateur de Bellman *optimal*. On utilise $D = \max_{s\neq s'} \min_{\pi\in\Pi^{\mathsf{SD}}} \mathbb{E}[\tau_\pi(s'|s)]$ pour dénoter le diamètre de M , où $\tau_\pi(s'|s)$ est le temps de frappe de s' à partir de s. Nous introduisons le diamètre du "pire cas"

$$\Upsilon = \max_{s\neq s'} \max_{\pi\in\Pi^{\mathsf{SD}}} \mathbb{E}\left[\tau_\pi(s'|s)\right], \tag{6.21}$$

qui définit le temps qu'il faut dans le pire des cas pour qu'une politique π passe de n'importe quel état s à s'. Asm. 16 garantit que $D \leq \Upsilon < \infty$.

Encore une fois, un algorithme très repandu pour ce type de scénario d'apprentissage repose sur l'optimisme. Cet algorithme porte le nom d'UCRL (Jaksch et al., 2010b) (Alg. 47). Tout comme UCB-VI précedemment il utilise les mêmes intervalles de confiance définis par les équations (6.16), (6.17) et (6.18).

Algorithm 46: Algorithme dit "Extended Value Iteration" (EVI)

Input: Opérateur à itérer $\mathcal{L} : \mathbb{R}^S \to \mathbb{R}^S$ et précision $\epsilon > 0$
Définir $v_0 = 0$, $v_1 = \mathcal{L}v_0$, $n = 0$
while $sp(v_{n+1} - v_n) > \epsilon$ **do**
  $n = n + 1$
  $v_{n+1} = \mathcal{L}v_n$
Output: $g_n = \frac{1}{2}\big(\max_s\{v_{n+1}(s) - v_n(s)\} + \min_s\{v_{n+1}(s) - v_n(s)\}\big)$ and v_n

UCRL bénificie lui aussi d'un regret quasi-optimal en fonction du nombre d'interactions T. C'est-à-dire,

$$R_T = Tg^\star - \sum_{t=1}^{T} r_t \leq \mathcal{O}(\sqrt{T}) \tag{6.25}$$

6.4 Contributions

Dans cette thèse, nous explorons les implications théoriques ou comment concilier le processus d'exploration dans les algorithmes d'apprentissage par renforcement et de bandits avec des contraintes pratiques qui, à première vue, nécessiteraient de limiter ou même de supprimer l'exploration. Dans chacun des chapitres, nous explorons une contrainte en montrant son effet sur le regret. Les résultats théoriques sont illustrés par des expériences sur des ensembles de données synthétiques et des ensembles de données du monde réel. Les preuves techniques se trouvent dans les annexes à la fin de chaque chapitre.

Algorithm 47: Algorithme UCRL

Input: $\delta \in (0,1)$, $r_{\max}$, $\mathcal{S}$, $\mathcal{A}$

Définir $t = 0$ et observer l'état initial s_1

for *Épisodes* $k = 1, 2, ...$ **do**

Définir l'étape de départ de l'épisode k, $t_k = t$

Pour tout $(s,a) \in \mathcal{S} \times \mathcal{A}$ initialiser les comptuers pour l'épisode k, $\nu_k(s,a) = 0$.

Définir le nombre de visites à (s,a) dans les épisodes précédents:

$$N_k(s,a) = \#\{\tau > t_k \mid s_\tau = s, a_\tau = a\} \tag{6.22}$$

Pour tout $(s,a,s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ initialiser les compteurs de récompenses accumulées en (s,a) et le nombre de transitions vers s' depuis (s,a)

$$R_k(s,a) = \sum_{\tau=1}^{t_k - 1} r_\tau \mathbb{1}_{\{ s_\tau = s, a_\tau = a\}} \tag{6.23}$$

$$P_k(s,a,s') = \#\{\{\tau > t_k \mid s_\tau = s, a_\tau = a\}, s_{\tau+1} = s'\} \tag{6.24}$$

Calculer les moyennes empiriques, $r_k(s,a) = \frac{R_k(s,a)}{\max\{1, N_k(s,a)\}}$ et $p_k(s' \mid s,a) = \frac{P_k(s,a,s')}{\max\{1, N_k(s,a)\}}$

Définir l'ensemble des MDPs $\mathcal{M}_k$ avec les mêmes états, $\mathcal{S}$, et les mêmes actions, $\mathcal{A}$, et dont les probabilités de transition $\tilde{p}(. \mid s,a)$ sont proches de $p_k(. \mid s,a)$ et les récompenses $\tilde{r}(s,a)$ proches de $r_k(s,a)$. C'est-à-dire:

$$|\tilde{r}(s,a) - r_k(s,a)| \leq \beta_r^k(s,a)$$
$$\|\tilde{p}(. \mid s,a) - p_k(. \mid s,a)\|_1 \leq \beta_p^k(s,a)$$

Utiliser l'algorithme EVI (Alg. 46) afin de calculer une politique de décision π_k et un MDP optimiste $\tilde{M}_k$ tel que:

$$g^\star - \frac{1}{\sqrt{t_k}} \leq g(\tilde{M}_k, \pi_k)$$

où $g(\tilde{M}_k, \pi_k)$ est le gain de la politique π_k dans le MDP $\tilde{M}_k$.

while $\nu_k(s_t, \pi_k(s_t)) \leq \max\{1, N_k(s_t, \pi_k(s_t))$ **do**

Selectionner l'action $a_t = \pi_k(s_t)$, obtenir la récompense r_t, et observer l'état s_{t+1}.

Mettre à jour $\nu_k(s_t, a_t) = \nu_k(s_t, a_t) + 1$.

Incrémenter $t = t + 1$.

6.4.1 Bandits avec évaluations bruitées et Exploration conservatrice

Le premier type de contrainte, qui fait l'objet de la Section 2.1 du Chapitre 2, peut être vu comme un type de contrainte de performance. Dans un implémentation ML de recommandation standard, il n'est pas rare que des objets soient notées par un ensemble de modèles spécialisés avant d'être envoyées à l'algorithme en charge de la recommandation finale. Les raisons de cette conception sont multiples. Dans certains cas, cela est simplement dû à des contraintes de puissance de calcul et de latence. Pour les problèmes avec des millions d'éléments à recommander, il est plus efficace d'obtenir un classement approximatif de tous les éléments, puis de calculer une estimation plus précise d'un nombre réduit d'éléments. Une autre raison peut être que la nature des objets à classer est hétérogène c'est-à-dire que l'algorithme peut avoir à classer des images et des données textuelles. Dans ce cas, il est souvent recommandé d'utiliser des algorithmes de scoring spécialisés. La seconde contrainte de performance, étudiée dans les Sections 2.2 et 2.3 du Chapitre 2 traite de la perte de récompense à court terme due à l'exploration dans les algorithmes RL et Bandits. Comme indiqué ci-dessus, en raison de l'exploration, les algorithmes RL prennent des actions qui ne sont pas optimales afin de les éliminer rapidement et de se concentrer sur les autres actions. Cependant, souvent lors du déploiement d'un algorithme RL, il existe un algorithme de base (qui repose peut-être sur des règles artisanales) résolvant le même problème que l'algorithme RL mais potentiellement sous-optimal. Si cet algorithme existant est effectivement sous-optimal, l'algorithme RL le surpassera à long terme. Dans le pire des cas, l'écart de performance peut être en grave défaveur de l'algorithme RL pendant une durée indeterminée. Cette situation est à éviter à tout prix et nécessite de

contrôler l'exploration de l'algorithme de manière à ce qu'il n'entreprenne de mauvaises actions que s'il a construit un "budget" pour cela. C'est-à-dire que nous voulons introduire un paramètre contrôlant la différence maximale entre les performances de l'algorithme existant et celui de RL alors que ce dernier est moins performant que le premier. Ce problème est appelé *exploration conservatrice*[2].

6.4.2 Confidentialité des données dans l'apprentissage par renforcement

Le deuxième type de contraintes auxquelles nous nous attaquons est motivé par la poussée récente pour plus de confidentialité dans les services en ligne. En effet, la question de la confidentialité dans les algorithmes d'apprentissage automatique est une question de longue date (Dwork et al., 2010a). Récemment, la notion de Confidentialité Différentielle a néanmoins été acceptée comme définition par défaut de la vie privée dans la littérature ML (Abadi et al., 2016). Dans la section 3.1 du chapitre 3, nous examinons comment cette définition a été adaptée au scénario RL tabulaire. La principale conséquence de la confidentialité dans le RL est une contrainte sur la séquence d'actions pouvant être selectioné par l'agent. Nous proposons le premier algorithme pour imposer une définition plus forte de la confidentialité, une définition décentralisée (Bebensee, 2019), et montrons comment cela affecte le regret en prouvant une borne inférieure et en présentant un algorithme faisant correspondre cette borne inférieure à des termes polynomiaux dans la taille de l'espace d'états et la taille de l'espace des actions. De plus, dans la Section 3.2 du Chapitre 3 nous montrons comment relier une avancée récente dans la théorie de la Confidentialité Différentielle (Feldman et al., 2020), qui permet de réaliser un équilibre entre une forte confidentialité et une dégradation des performances, à un problème de bandit contextuel linéaire.

6.4.3 Sécurité dans l'apprentissage par renforcement.

Le dernier type de contrainte que nous considérons dans cette thèse est une contrainte de sécurité. Dans le Chapitre 4, nous étudions deux aspects différents de la sécurité. Tout d'abord, dans la section 4.1, nous étudions la sensibilité des bandits contextuels linéaires aux attaques d'adversaires. Dans ces attaques, les attaquants sont autorisés à modifier les informations envoyés à l'algorithme. Leur objectif étant de minimiser le changement cumulatif dans la rétroaction tout en s'assurant que l'algorithme de bandit est incapable de trouver les meilleures actions. Dans la section 4.2, nous explorons un deuxième aspect de la sécurité selon lequel toutes les informations envoyées et reçues par l'algorithme sont chiffrées de bout en bout. L'algorithme ne peut pas décrypter les données mais peut effectuer des calculs sur les données cryptées. Ce cryptage s'accompagne de sérieux inconvénients et limitations de calcul. C'est-à-dire qu'il y a un nombre limité d'additions ou de multiplications possibles avant de rendre les données indéchiffrables. Malgré cette limitation, nous montrons comment construire un algorithme de bandit contextuel linéaire avec un regret comparable aux algorithmes de bandit contextuel linéaire standard.

6.4.4 Liste des publications dans les conférences internationales avec journal.

La liste ci-dessous contient les publications auxquelles j'ai participé durant cette thèse. Le chapitre 2 est basé sur (**Garcelon** et al., 2022c, 2020a,b). Le chapitre 3 est basé sur (**Garcelon** et al., 2021, 2022b) et le dernier chapitre, Chapter 4 est basé sur (**Garcelon** et al., 2020c, 2022c).

Publications presented in this thesis:

- **Evrard Garcelon**, Vashist Avadhanula, Alessandro Lazaric, and Matteo Pirotta. Top k ranking for multi-armed bandit with noisy evaluations. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 6242–6269. PMLR, 28–30 Mar 2022a. URL https://proceedings.mlr.press/v151/garcelon22b.html
- **Evrard Garcelon**, Mohammad Ghavamzadeh, Alessandro Lazaric, and Matteo Pirotta. Improved algorithms for conservative exploration in bandits. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3962–3969, Apr. 2020a. doi: 10.1609/aaai.v34i04.5812. URL https://ojs.aaai.org/index.php/AAAI/article/view/5812
- **Evrard Garcelon**, Mohammad Ghavamzadeh, Alessandro Lazaric, and Matteo Pirotta. Conservative exploration in reinforcement learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1431–1441. PMLR, 26–28 Aug 2020b. URL https://proceedings.mlr.press/v108/garcelon20a.html
- **Evrard Garcelon**, Vianney Perchet, Ciara Pike-Burke, and Matteo Pirotta. Local differential privacy for regret minimization in reinforcement learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 10561–10573. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/

[2]Pour chaque type de contraintes, une discussion plus approfondie sur les travaux connexes est fournie dans le chapitre correspondant.

- **Evrard Garcelon**, Kamalika Chaudhuri, Vianney Perchet, and Matteo Pirotta. Privacy amplification via shuffling for linear contextual bandits. In Sanjoy Dasgupta and Nika Haghtalab, editors, *International Conference on Algorithmic Learning Theory, 29-1 April 2022, Paris, France*, volume 167 of *Proceedings of Machine Learning Research*, pages 381–407. PMLR, 2022b. URL https://proceedings.mlr.press/v167/garcelon22a.html
- **Evrard Garcelon**, Baptiste Roziere, Laurent Meunier, Jean Tarbouriech, Olivier Teytaud, Alessandro Lazaric, and Matteo Pirotta. Adversarial attacks on linear contextual bandits. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14362–14373. Curran Associates, Inc., 2020c. URL https://papers.nips.cc/paper/2020
- **Evrard Garcelon**, Matteo Pirotta, and Vianney Perchet. Encrypted linear contextual bandit. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 2519–2551. PMLR, 28–30 Mar 2022c. URL https://proceedings.mlr.press/v151/garcelon22a.html

Other publications not presented in this thesis:

- Rémy Degenne, **Evrard Garcelon**, and Vianney Perchet. Bandits with side observations: Bounded vs. logarithmic regret. In Amir Globerson and Ricardo Silva, editors, *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, pages 467–476. AUAI Press, 2018. URL http://auai.org/uai2018/proceedings/supplements/Supplementary-Paper182.pdf
- Jean Tarbouriech, **Evrard Garcelon**, Michal Valko, Matteo Pirotta, and Alessandro Lazaric. No-regret exploration in goal-oriented reinforcement learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9428–9437. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/tarbouriech20a.html
- Yunchang Yang, Tianhao Wu, Han Zhong, **Evrard Garcelon**, Matteo Pirotta, Alessandro Lazaric, Liwei Wang, and Simon Shaolei Du. A reduction-based framework for conservative bandits and reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=AcrlgZ9BKed

References

PALISADE Lattice Cryptography Library (release 1.10.4). https://palisade-crypto.org/, September 2020.

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *NIPS*, pages 2312–2320, 2011.

Marc Abeille, Alessandro Lazaric, et al. Linear thompson sampling revisited. *Electronic Journal of Statistics*, 11(2):5165–5197, 2017.

John M Abowd. The us census bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2867–2867, 2018.

Abbas Acar, Hidayet Aksu, A. Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Comput. Surv.*, 51(4), July 2018. ISSN 0360-0300. doi: 10.1145/3214303. URL https://doi.org/10.1145/3214303.

Jayadev Acharya, Kallista Bonawitz, Peter Kairouz, Daniel Ramage, and Ziteng Sun. Context aware local differential privacy. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 52–62. PMLR, 2020.

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 22–31. PMLR, 2017.

Alekh Agarwal, Daniel J. Hsu, Satyen Kale, John Langford, Lihong Li, and Robert E. Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In *ICML*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1638–1646. JMLR.org, 2014.

Akshay Agrawal, Robin Verschueren, Steven Diamond, and Stephen Boyd. A rewriting system for convex optimization problems. *Journal of Control and Decision*, 5(1):42–60, 2018.

Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on learning theory*, pages 39–1, 2012.

Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *ICML (3)*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 127–135. JMLR.org, 2013.

Martin Albrecht, Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Shai Halevi, Jeffrey Hoffstein, Kim Laine, Kristin Lauter, Satya Lokam, Daniele Micciancio, Dustin Moody, Travis Morrison, Amit Sahai, and Vinod Vaikuntanathan. Homomorphic encryption security standard. Technical report, HomomorphicEncryption.org, Toronto, Canada, November 2018.

Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

Miguel E. Andrés, Nicolás Emilio Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: differential privacy for location-based systems. In *CCS*, pages 901–914. ACM, 2013.

Apple. Learning with privacy at scale. https://machinelearning.apple.com/research/learning-with-privacy-at-scale.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning Journal*, 47:235–256, 2002a.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002b.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, January 2003. ISSN 0097-5397. doi: 10.1137/S0097539701398375. URL https://doi.org/10.1137/S0097539701398375.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 263–272. PMLR, 2017a.

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 263–272. PMLR, 2017b.

Ahmad Al Badawi, Jin Chao, Jie Lin, Chan Fook Mun, Jun Jie Sim, Benjamin Hong Meng Tan, Xiao Nan, Khin Mi Mi Aung, and Vijay Ramaseshan Chandrasekhar. Towards the alexnet moment for homomorphic encryption: Hcnn, thefirst homomorphic cnn on encrypted data with gpus, 2020.

Yu Bai, Tengyang Xie, Nan Jiang, and Yu-Xiang Wang. Provably efficient q-learning with low switching cost. In *Advances in Neural Information Processing Systems*, pages 8004–8013, 2019.

Victor Balcer and Albert Cheu. Separating local & shuffled differential privacy via histograms. In *ITC*, volume 163 of *LIPIcs*, pages 1:1–1:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

Borja Balle, Maziar Gomrokchi, and Doina Precup. Differentially private policy evaluation. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2130–2138. JMLR.org, 2016.

Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. Differentially private summation with multi-message shuffling. *CoRR*, abs/1906.09116, 2019a.

Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In *CRYPTO (2)*, volume 11693 of *Lecture Notes in Computer Science*, pages 638–667. Springer, 2019b.

Peter L. Bartlett and Ambuj Tewari. REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs. In *UAI*, pages 35–42. AUAI Press, 2009.

Raef Bassily and Adam Smith. Local, private, efficient protocols for succinct histograms. *Proceedings of the forty-seventh annual ACM symposium on Theory of Computing*, Jun 2015. doi: 10.1145/2746539.2746632. URL http://dx.doi.org/10.1145/2746539.2746632.

Hamsa Bastani and Mohsen Bayati. Online decision making with high-dimensional covariates. *Operations Research*, 68(1): 276–294, 2020.

Debabrota Basu, Christos Dimitrakakis, and Aristide C. Y. Tossou. Differential privacy for multi-armed bandits: What is it and what is its cost? *CoRR*, abs/1905.12298, 2019.

Björn Bebensee. Local differential privacy: a tutorial, 2019. URL https://arxiv.org/abs/1907.11908.

Felix Berkenkamp, Matteo Turchetta, Angela P. Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *NIPS*, pages 908–918, 2017.

Dimitri P Bertsekas. *Dynamic programming and optimal control. Vol II.* Number 2. Athena scientific Belmont, MA, 1995.

Dimitri P Bertsekas. Approximate policy iteration: A survey and some new methods. *Journal of Control Theory and Applications*, 9(3):310–335, 2011.

Alina Beygelzimer, John Langford, Lihong Li, Lev Reyzin, and Robert E. Schapire. An optimal high probability algorithm for the contextual bandit problem. *CoRR*, abs/1002.4058, 2010. URL http://arxiv.org/abs/1002.4058.

Alina Beygelzimer, John Langford, Lihong Li, Lev Reyzin, and Robert E. Schapire. Contextual bandit algorithms with supervised learning guarantees. In *AISTATS*, volume 15 of *JMLR Proceedings*, pages 19–26. JMLR.org, 2011.

Alberto Bietti, Alekh Agarwal, and John Langford. A contextual bandit bake-off, 2018. URL https://arxiv.org/abs/1802.04064.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo. *Proceedings of the 26th Symposium on Operating Systems Principles*, Oct 2017. doi: 10.1145/3132747.3132769. URL http://dx.doi.org/10.1145/3132747.3132769.

Marcelo Blatt, Alexander Gusev, Yuriy Polyakov, and Shafi Goldwasser. Secure large-scale genome-wide association studies using homomorphic encryption. *Proceedings of the National Academy of Sciences*, 117(21):11608–11613, 2020.

Raphael Bost, Raluca Ada Popa, Stephen Tu, and Shafi Goldwasser. Machine learning classification over encrypted data. 2015.

L. Bottou, J. Peters, J. Quinonero-Candela, D. Charles, D. Chickering, E. Portugaly, D. Ray, P. Simard, and E. Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 14:3207–3260, 2013.

Djallel Bouneffouf. Corrupted contextual bandits: Online learning with corrupted context. In *ICASSP*, pages 3145–3149. IEEE, 2021.

Etienne Boursier and Vianney Perchet. Utility/privacy trade-off through the lens of optimal transport, 2020.

Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *Annual Cryptology Conference*, pages 868–886. Springer, 2012.

Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36, 2014.

Sébastien Bubeck and Aleksandrs Slivkins. The best of both worlds: Stochastic and adversarial bandits. In *Conference on Learning Theory*, pages 42–1, 2012.

Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012. ISSN 1935-8237. doi: 10.1561/2200000024.

Daniele Calandriello, Luigi Carratino, Alessandro Lazaric, Michal Valko, and Lorenzo Rosasco. Near-linear time gaussian process optimization with adaptive batching and resparsification. In *International Conference on Machine Learning*, pages 1295–1305. PMLR, 2020.

Alexandra Carpentier, Claire Vernade, and Yasin Abbasi-Yadkori. The elliptical potential lemma revisited, 2020.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

Lijie Chen, Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. On distributed differential privacy and counting distinct elements. In *ITCS*, volume 185 of *LIPIcs*, pages 56:1–56:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

Xiaoyu Chen, Kai Zheng, Zixin Zhou, Yunchang Yang, Wei Chen, and Liwei Wang. (locally) differentially private combinatorial semi-bandits. In *ICML*, 2020.

Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 409–437. Springer, 2017.

Jung Hee Cheon, Dongwoo Kim, Duhyeong Kim, Hun Hee Lee, and Keewoo Lee. Numerical method for comparison on homomorphically encrypted numbers. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 415–445. Springer, 2019.

Jung Hee Cheon, Dongwoo Kim, and Duhyeong Kim. Efficient homomorphic comparison methods with optimal complexity. In *ASIACRYPT (2)*, volume 12492 of *Lecture Notes in Computer Science*, pages 221–256. Springer, 2020.

Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. *Lecture Notes in Computer Science*, page 375–403, 2019. ISSN 1611-3349. doi: 10.1007/978-3-030-17653-2_13. URL http://dx.doi.org/10.1007/978-3-030-17653-2_13.

Albert Cheu, Adam D. Smith, and Jonathan R. Ullman. Manipulation attacks in local differential privacy. *J. Priv. Confidentiality*, 11(1), 2021.

Yinlam Chow, Ofir Nachum, Edgar A. Duéñez-Guzmán, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In *NeurIPS*, pages 8103–8112, 2018.

Konstantina Christakopoulou and Arindam Banerjee. Adversarial attacks on an oblivious recommender. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 322–330, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362436. doi: 10.1145/3298689.3347031. URL https://doi.org/10.1145/3298689.3347031.

Wei Chu, Lihong Li, Lev Reyzin, and Robert E. Schapire. Contextual bandits with linear payoff functions. In *AISTATS*, volume 15 of *JMLR Proceedings*, pages 208–214. JMLR.org, 2011.

Radu Ciucanu, Pascal Lafourcade, Marius Lombard-Platet, and Marta Soare. Secure best arm identification in multi-armed bandits. In *International Conference on Information Security Practice and Experience*, pages 152–171. Springer, 2019.

Yuval Dagan and Gil Kur. A bounded-noise mechanism for differential privacy, 2020.

Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 643–662. Springer, 2012.

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*, pages 5713–5723, 2017.

Sanchari Das, Robert S. Gutzwiller, Rod D. Roscoe, Prashanth Rajivan, Yang Wang, L. Jean Camp, and Roberto Hoyle. Panel: Humans and technology for inclusive privacy and security, 2021.

James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. The youtube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 293–296, 2010.

Rémy Degenne, **Evrard Garcelon**, and Vianney Perchet. Bandits with side observations: Bounded vs. logarithmic regret. In Amir Globerson and Ricardo Silva, editors, *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, pages 467–476. AUAI Press, 2018. URL http://auai.org/uai2018/proceedings/supplements/Supplementary-Paper182.pdf.

Julien Déry, Angel B. Ruiz, François Routhier, Marie-Pierre Gagnon, André Côté, Daoud Ait-kadi, Valérie Bélanger, Simon Deslauriers, and Marie-Ève Lamontagne. Patient prioritization tools and their effectiveness in non-emergency healthcare services: a systematic review protocol. *Systematic Reviews*, 8, 2019.

Thomas Desautels, Andreas Krause, and Joel W Burdick. Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. *Journal of Machine Learning Research*, 15:3873–3923, 2014.

Kefan Dong, Yingkai Li, Qin Zhang, and Yuan Zhou. Multinomial logit bandit with low switching cost. In *International Conference on Machine Learning*, pages 2607–2615. PMLR, 2020.

Abhimanyu Dubey and Alex Pentland. Differentially-private federated linear bandits. Technical report, Massachusetts Institute of Technology, 2020a.

Abhimanyu Dubey and Alex Pentland. Private and byzantine-proof cooperative decision-making. In *AAMAS*, pages 357–365. International Foundation for Autonomous Agents and Multiagent Systems, 2020b.

Léo Ducas and Daniele Micciancio. Fhew: bootstrapping homomorphic encryption in less than a second. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 617–640. Springer, 2015.

John Duchi and Ryan Rogers. Lower bounds for locally private estimation via communication complexity, 2019.

John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy, data processing inequalities, and statistical minimax rates. *arXiv preprint arXiv:1302.3203*, 2013.

John C. Duchi, Martin J. Wainwright, and Michael I. Jordan. Minimax optimal procedures for locally private estimation. *CoRR*, abs/1604.02390, 2016.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006.

Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil P. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *STOC*, pages 381–390. ACM, 2009.

Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 715–724, 2010a.

Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. Boosting and differential privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 51–60. IEEE, 2010b.

Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 1054–1067, 2014.

Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *SODA*, pages 2468–2479. SIAM, 2019.

Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Shuang Song, Kunal Talwar, and Abhradeep Thakurta. Encode, shuffle, analyze privacy revisited: Formalizations and empirical evaluation. *CoRR*, abs/2001.03618, 2020.

Pedro M Esperança, Louis JM Aslett, and Chris C Holmes. Encrypted accelerated least squares regression. *arXiv preprint arXiv:1703.00839*, 2017.

Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption.

Vitaly Feldman, Audra McMillan, and Kunal Talwar. Hiding among the clones: A simple and nearly optimal analysis of privacy amplification by shuffling, 2020.

Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. Parametric bandits: The generalized linear case. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper/2010/file/c2626d850c80ea07e7511bbae4c76f4b-Paper.pdf.

Michael J Fischer, Nancy A Lynch, and Michael S Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)*, 32(2):374–382, 1985.

David A Freedman. On tail probabilities for martingales. *the Annals of Probability*, 3(1):100–118, 1975.

Ronan Fruit, Matteo Pirotta, and Alessandro Lazaric. Near optimal exploration-exploitation in non-communicating markov decision processes. In *NIPS*, 2018a.

Ronan Fruit, Matteo Pirotta, Alessandro Lazaric, and Ronald Ortner. Efficient bias-span-constrained exploration-exploitation in reinforcement learning. In *ICML*, Proceedings of Machine Learning Research. PMLR, 2018b.

Ronan Fruit, Matteo Pirotta, and Alessandro Lazaric. Improved analysis of ucrl2 with empirical bernstein inequality. *arXiv preprint arXiv:2007.05456*, 2020.

Pratik Gajane, Tanguy Urvoy, and Emilie Kaufmann. Corrupt Bandits. In *EWRL 13*, 2016.

Pratik Gajane, Tanguy Urvoy, and Emilie Kaufmann. Corrupt bandits for preserving local privacy. In *ALT*, volume 83 of *Proceedings of Machine Learning Research*, pages 387–412. PMLR, 2018.

Craig Gentry and Dan Boneh. *A fully homomorphic encryption scheme*, volume 20. Stanford university Stanford, 2009.

Joseph Geumlek and Kamalika Chaudhuri. Profile-based privacy for locally private computations. In *ISIT*, pages 537–541. IEEE, 2019.

Mustansar Ghazanfar and Adam Prugel-Bennett. An improved switching hybrid recommender system using naive bayes classifier and collaborative filtering. 2010.

Kenneth Y. Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Inf. Retr.*, 4(2):133–151, 2001.

Gene H. Golub and Charles F. Van Loan. An analysis of the total least squares problem. *SIAM Journal on Numerical Analysis*, 17(6):883–893, 1980. ISSN 00361429. URL http://www.jstor.org/stable/2156807.

Carlos A Gomez-Uribe and Neil Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, 6(4):1–19, 2015.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Aditya Gopalan and Shie Mannor. Thompson sampling for learning parameterized markov decision processes. In *COLT*, volume 40 of *JMLR Workshop and Conference Proceedings*, pages 861–898. JMLR.org, 2015.

Benjamin Graham. Fractional max-pooling, 2015.

Ziwei Guan, Kaiyi Ji, Donald J Bucci Jr, Timothy Y Hu, Joseph Palombo, Michael Liston, and Yingbin Liang. Robust stochastic bandit algorithms under probabilistic unbounded adversarial attack. *arXiv preprint arXiv:2002.07214*, 2020.

Chun-Hua Guo and Nicholas J Higham. A schur–newton method for the matrix\boldmath p th root and its inverse. *SIAM Journal on Matrix Analysis and Applications*, 28(3):788–804, 2006.

Dalin Guo, Sofia Ira Ktena, Pranay Kumar Myana, Ferenc Huszar, Wenzhe Shi, Alykhan Tejani, Michael Kneier, and Sourav Das. Deep bayesian bandits: Exploring in online personalized recommendations. In *Fourteenth ACM Conference on Recommender Systems*, pages 456–461, 2020.

Anupam Gupta, Tomer Koren, and Kunal Talwar. Better algorithms for stochastic bandits with adversarial corruptions. *arXiv preprint arXiv:1902.08647*, 2019.

Shai Halevi. Homomorphic encryption. In *Tutorials on the Foundations of Cryptography*, pages 219–276. Springer, 2017.

Kyoohyung Han, Seungwan Hong, Jung Hee Cheon, and Daejun Park. Efficient logistic regression on large encrypted data.

Yanjun Han, Zhengqing Zhou, Zhengyuan Zhou, Jose Blanchet, Peter W. Glynn, and Yinyu Ye. Sequential batch learning in finite-action linear contextual bandits, 2020.

Yuxuan Han, Zhipeng Liang, Yang Wang, and Jiheng Zhang. Generalized linear bandits with local differential privacy. *Advances in Neural Information Processing Systems*, 34:26511–26522, 2021.

Awni Y. Hannun, Brian Knott, Shubho Sengupta, and Laurens van der Maaten. Privacy-preserving multi-party contextual bandits. *CoRR*, abs/1910.05299, 2019.

F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis) DOI=http://dx.doi.org/10.1145/2827872*, 5(4):1–19, 2015.

Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991. doi: 10.1017/CBO9780511840371.

Léonard Hussenot, Matthieu Geist, and Olivier Pietquin. Targeted attacks on deep reinforcement learning agents through adversarial observations. *arXiv preprint arXiv:1905.12282*, 2019.

Eugene Ie, Vihan Jain, Jing Wang, Sanmit Narvekar, Ritesh Agarwal, Rui Wu, Heng-Tze Cheng, Morgane Lustman, Vince Gatto, Paul Covington, Jim McFadden, Tushar Chandra, and Craig Boutilier. Reinforcement learning for slate-based recommender systems: A tractable decomposition and practical methodology, 2019. URL https://arxiv.org/abs/1905.12767.

Nicole Immorlica, Karthik Abinav Sankararaman, Robert E. Schapire, and Aleksandrs Slivkins. Adversarial bandits with knapsacks. *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 202–219, 2018.

Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35. IEEE, 2018.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11:1563–1600, 2010a.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010b.

Xiaoqian Jiang, Miran Kim, Kristin Lauter, and Yongsoo Song. Secure outsourced matrix computation and application to neural networks. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1209–1222, 2018.

Chi Jin, Zeyuan Allen-Zhu, Sébastien Bubeck, and Michael I. Jordan. Is q-learning provably efficient? *CoRR*, abs/1807.03765, 2018.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I. Jordan. Provably efficient reinforcement learning with linear function approximation. *CoRR*, abs/1907.05388, 2019.

Kwang-Sung Jun, Lihong Li, Yuzhe Ma, and Jerry Zhu. Adversarial attacks on stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 3640–3649, 2018.

Peter Kairouz, Keith Bonawitz, and Daniel Ramage. Discrete distribution estimation under local privacy. *arXiv preprint arXiv:1602.07387*, 2016.

Sham M. Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *ICML*, pages 267–274. Morgan Kaufmann, 2002.

Erich Kaltofen and Gilles Villard. On the complexity of computing determinants. *computational complexity*, 13(3-4):91–130, 2005.

Sayash Kapoor, Kumar Kshitij Patel, and Purushottam Kar. Corruption-tolerant bandit learning. *Machine Learning*, 108(4): 687–715, 2019.

Sumeet Katariya, Branislav Kveton, Zheng Wen, and Vamsi K. Potluru. Conservative exploration using interleaving. In *AISTATS*, volume 89 of *Proceedings of Machine Learning Research*, pages 954–963. PMLR, 2019.

Chamara Kattadige, Aravindh Raman, Kanchana Thilakarathna, Andra Lutu, and Diego Perino. 360norvic: 360-degree video classification from mobile encrypted video traffic. *arXiv preprint arXiv:2105.03611*, 2021.

Abbas Kazerouni, Mohammad Ghavamzadeh, Yasin Abbasi, and Benjamin Van Roy. Conservative contextual linear bandits. In *NIPS*, pages 3910–3919, 2017.

Romain Laroche, Paul Trichelair, and Remi Tachet des Combes. Safe policy improvement with baseline bootstrapping. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 3652–3661. PMLR, 2019.

Tor Lattimore and Csaba Szepesvári. Bandit algorithms. Pre-publication version, 2018. URL http://downloads.tor-lattimore.com/banditbook/book.pdf.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Alessandro Lazaric, Matteo Pirotta, and Ronan Fruit. Regret minimization in infinite-horizon finite markov decision processes. Tutorial at ALT'19, 2019. URL https://rlgammazero.github.io/.

Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In *Advances in neural information processing systems*, pages 1885–1893, 2016.

Changmao Li, Elaine Fisher, Rebecca Thomas, Steve Pittard, Vicki Hertzberg, and Jinho D. Choi. Competence-level prediction and resume & job description matching using context-aware transformer models. In *EMNLP (1)*, pages 8456–8466. Association for Computational Linguistics, 2020.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

Lihong Li, Yu Lu, and Dengyong Zhou. Provably optimal algorithms for generalized linear contextual bandits. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 2071–2080. PMLR, 2017.

Yingkai Li, Edmund Y Lou, and Liren Shan. Stochastic linear optimization with adversarial corruption. *arXiv preprint arXiv:1909.02109*, 2019.

Chang Liu, Bo Li, Yevgeniy Vorobeychik, and Alina Oprea. Robust linear regression against training data poisoning. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 91–102, 2017.

Fang Liu and Ness Shroff. Data poisoning attacks on stochastic bandits. *arXiv preprint arXiv:1905.06494*, 2019.

Thodoris Lykouris, Vahab Mirrokni, and Renato Paes Leme. Stochastic bandits robust to adversarial corruptions. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 114–122, 2018.

Thodoris Lykouris, Max Simchowitz, Aleksandrs Slivkins, and Weidong Sun. Corruption robust exploration in episodic reinforcement learning. *ArXiv*, abs/1911.08689, 2019.

Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM (JACM)*, 60(6):1–35, 2013a.

Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-lwe cryptography. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 35–54. Springer, 2013b.

Yuzhe Ma, Kwang-Sung Jun, Lihong Li, and Xiaojin Zhu. Data poisoning attacks in contextual bandits. In *International Conference on Decision and Game Theory for Security*, pages 186–204. Springer, 2018.

Yuzhe Ma, Xuezhou Zhang, Wen Sun, and Jerry Zhu. Policy poisoning in batch reinforcement learning and control. In *Advances in Neural Information Processing Systems*, pages 14543–14553, 2019.

Mohammad Malekzadeh, Dimitrios Athanasakis, Hamed Haddadi, and Benjamin Livshits. Privacy-preserving bandits. In *MLSys*. mlsys.org, 2020.

Yishay Mansour, Aleksandrs Slivkins, and Vasilis Syrgkanis. Bayesian incentive-compatible bandit exploration. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, pages 565–582. ACM, 2015.

Hongzi Mao, Shannon Chen, Drew Dimmery, Shaun Singh, Drew Blaisdell, Yuandong Tian, Mohammad Alizadeh, and Eytan Bakshy. Real-world video adaptation with reinforcement learning, 2020.

Bhaskar Mehta and Wolfgang Nejdl. Attack resistant collaborative filtering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 75–82, 2008.

Nikita Mishra and Abhradeep Thakurta. (nearly) optimal differentially private stochastic multi-arm bandits. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, UAI'15, page 592–601, Arlington, Virginia, USA, 2015. AUAI Press. ISBN 9780996643108.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602.

Netflix. Artwork personalization at netflix. https://netflixtechblog.com/artwork-personalization-c589f074ad76.

Gergely Neu and Ciara Pike-Burke. A unifying view of optimism in episodic reinforcement learning. *arXiv preprint arXiv:2007.01891*, 2020.

Goldreich Oded. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, USA, 1st edition, 2009. ISBN 052111991X.

Minou Catharina Anselma Olde Keizer. *Condition-based maintenance for complex systems: Coordinating maintenance and logistics planning for the process industries*. PhD thesis, University of Groningen, 2016.

Hajime Ono and Tsubasa Takahashi. Locally private distributed reinforcement learning. *arXiv preprint arXiv:2001.11718*, 2020.

Ian Osband and Benjamin Van Roy. Posterior sampling for reinforcement learning without episodes. *CoRR*, abs/1608.02731, 2016.

Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. In *NIPS*, pages 3003–3011, 2013.

Yi Ouyang, Mukul Gagrani, Ashutosh Nayyar, and Rahul Jain. Learning unknown markov decision processes: A thompson sampling approach. In *NIPS*, pages 1333–1342, 2017.

Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International conference on the theory and applications of cryptographic techniques*, pages 223–238. Springer, 1999.

Xinlei Pan, Weiyao Wang, Xiaoshuai Zhang, Bo Li, Jinfeng Yi, and Dawn Song. How you act tells a lot: Privacy-leaking attack on deep reinforcement learning. In *AAMAS*, pages 368–376. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

Haekyu Park, Jinhong Jung, and U Kang. A comparative study of matrix factorization and random walk with restart in recommender systems. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 756–765. IEEE, 2017.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Vianney Perchet, Philippe Rigollet, Sylvain Chassang, Erik Snowberg, et al. Batched bandit problems. *The Annals of Statistics*, 44(2):660–681, 2016.

M. Petrik, M. Ghavamzadeh, and Y. Chow. Safe policy improvement by minimizing robust baseline regret. In *Advances in Neural Information Processing Systems*, pages 2298–2306, 2016.

Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Adaptive step-size for policy gradient methods. In *NIPS*, pages 1394–1402, 2013a.

Matteo Pirotta, Marcello Restelli, Alessio Pecorino, and Daniele Calandriello. Safe policy iteration. In *ICML (3)*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 307–315. JMLR.org, 2013b.

A. R. Provenzano, D. Trifirò, A. Datteo, L. Giada, N. Jean, A. Riciputi, G. Le Pera, M. Spadaccino, L. Massaron, and C. Nordio. Machine learning approach for credit scoring, 2020.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1994. ISBN 0471619779.

Jian Qian, Ronan Fruit, Matteo Pirotta, and Alessandro Lazaric. Exploration bonus for regret minimization in discrete and continuous average reward mdps. In *NeurIPS*, pages 4891–4900, 2019.

Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6): 1–40, 2009.

Wenbo Ren, Xingyu Zhou, Jia Liu, and Ness B Shroff. Multi-armed bandits with local differential privacy. *arXiv preprint arXiv:2007.03121*, 2020.

Carlos Riquelme, George Tucker, and Jasper Snoek. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. In *ICLR (Poster)*. OpenReview.net, 2018.

Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

Yufei Ruan, Jiaqi Yang, and Yuan Zhou. Linear bandits with limited adaptivity and learning distributional optimal design. *arXiv preprint arXiv:2007.01980*, 2020.

Daniel Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. A tutorial on thompson sampling. *Found. Trends Mach. Learn.*, 11(1):1–96, 2018.

Mohamed Sabt, Mohammed Achemlal, and Abdelmadjid Bouabdallah. Trusted execution environment: what it is, and what it is not. In *2015 IEEE Trustcom/BigDataSE/ISPA*, volume 1, pages 57–64. IEEE, 2015.

Touqir Sajed and Or Sheffet. An optimal private stochastic-mab algorithm based on optimal private stopping rule. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 5579–5588. PMLR, 2019.

Neela Sawant, Chitti Babu Namballa, Narayanan Sadagopan, and Houssam Nassif. Contextual multi-armed bandits for causal marketing. *arXiv preprint arXiv:1810.01859*, 2018.

Abraham Seidenberg. Constructions in a polynomial ring over the ring of integers. *American Journal of Mathematics*, 100(4): 685–703, 1978.

Roshan Shariff and Or Sheffet. Differentially private contextual linear bandits. In *NeurIPS*, pages 4301–4311, 2018.

Ritu Sharma, Dinesh Gopalani, and Yogesh Meena. Collaborative filtering-based recommender system: Approaches and research challenges. In *2017 3rd international conference on computational intelligence & communication technology (CICT)*, pages 1–6. IEEE, 2017.

Jing-Cheng Shi, Yang Yu, Qing Da, Shi-Yong Chen, and An-Xiang Zeng. Virtual-taobao: Virtualizing real-world online retail environment for reinforcement learning, 2018. URL https://arxiv.org/abs/1805.10000.

Reema Sikka, Amita Dhankhar, and Chaavi Rana. A survey paper on e-learning recommender system. *International Journal of Computer Applications*, 47(9):27–30, 2012.

Thiago D. Simão and Matthijs T. J. Spaan. Safe policy improvement with baseline bootstrapping in factored environments. In *AAAI*, pages 4967–4974. AAAI Press, 2019.

Spotify. Shifting consumption towards diverse content via reinforcement learning. https://research.atspotify.com/shifting-consumption-towards-diverse-content-via-reinforcement-learning/.

Jianwen Sun, Tianwei Zhang, Xiaofei Xie, Lei Ma, Yan Zheng, Kangjie Chen, and Yang Liu. Stealthy and efficient adversarial attacks against deep reinforcement learning. *To appear in Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

A. Swaminathan and T. Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *Proceedings of The 32nd International Conference on Machine Learning*, 2015.

Jean Tarbouriech, **Evrard Garcelon**, Michal Valko, Matteo Pirotta, and Alessandro Lazaric. No-regret exploration in goal-oriented reinforcement learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9428–9437. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/tarbouriech20a.html.

Jay Tenenbaum, Haim Kaplan, Yishay Mansour, and Uri Stemmer. Differentially private multi-armed bandits in the shuffle model. *CoRR*, abs/2106.02900, 2021.

Evrard Garcelon, Mohammad Ghavamzadeh, Alessandro Lazaric, and Matteo Pirotta. Improved algorithms for conservative exploration in bandits. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3962–3969, Apr. 2020a. doi: 10.1609/aaai.v34i04.5812. URL https://ojs.aaai.org/index.php/AAAI/article/view/5812.

Evrard Garcelon, Mohammad Ghavamzadeh, Alessandro Lazaric, and Matteo Pirotta. Conservative exploration in reinforcement learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1431–1441. PMLR, 26–28 Aug 2020b. URL https://proceedings.mlr.press/v108/garcelon20a.html.

Evrard Garcelon, Baptiste Roziere, Laurent Meunier, Jean Tarbouriech, Olivier Teytaud, Alessandro Lazaric, and Matteo Pirotta. Adversarial attacks on linear contextual bandits. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14362–14373. Curran Associates, Inc., 2020c. URL https://papers.nips.cc/paper/2020.

Evrard Garcelon, Vianney Perchet, Ciara Pike-Burke, and Matteo Pirotta. Local differential privacy for regret minimization in reinforcement learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 10561–10573. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/.

Evrard Garcelon, Vashist Avadhanula, Alessandro Lazaric, and Matteo Pirotta. Top k ranking for multi-armed bandit with noisy evaluations. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 6242–6269. PMLR, 28–30 Mar 2022a. URL https://proceedings.mlr.press/v151/garcelon22b.html.

Evrard Garcelon, Kamalika Chaudhuri, Vianney Perchet, and Matteo Pirotta. Privacy amplification via shuffling for linear contextual bandits. In Sanjoy Dasgupta and Nika Haghtalab, editors, *International Conference on Algorithmic Learning Theory, 29-1 April 2022, Paris, France*, volume 167 of *Proceedings of Machine Learning Research*, pages 381–407. PMLR, 2022b. URL https://proceedings.mlr.press/v167/garcelon22a.html.

Evrard Garcelon, Matteo Pirotta, and Vianney Perchet. Encrypted linear contextual bandit. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 2519–2551. PMLR, 28–30 Mar 2022c. URL https://proceedings.mlr.press/v151/garcelon22a.html.

P. Thomas, G. Theocharous, and M. Ghavamzadeh. High confidence off-policy evaluation. In *Proceedings of the Twenty-Ninth Conference on Artificial Intelligence*, 2015a.

P. Thomas, G. Theocharous, and M. Ghavamzadeh. High confidence policy improvement. In *Proceedings of the Thirty-Second International Conference on Machine Learning*, pages 2380–2388, 2015b.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

Aristide Tossou and Christos Dimitrakakis. Differentially private multi-agent multi-armed bandits. In *European Workshop on Reinforcement Learning (EWRL-15)*, 2015.

Aristide C. Y. Tossou and Christos Dimitrakakis. Algorithms for differentially private multi-armed bandits. In *AAAI*, pages 2087–2093. AAAI Press, 2016.

Hoang Tuy. Dc optimization: theory, methods and algorithms. In *Handbook of global optimization*, pages 149–216. Springer, 1995.

J.M. Varah. A lower bound for the smallest singular value of a matrix. *Linear Algebra and its Applications*, 11(1):3 – 5, 1975. ISSN 0024-3795. doi: https://doi.org/10.1016/0024-3795(75)90112-3. URL http://www.sciencedirect.com/science/article/pii/0024379575901123.

Giuseppe Vietri, Borja de Balle Pigem, Akshay Krishnamurthy, and Steven Wu. Private reinforcement learning with pac and regret guarantees. In *ICML*, 2020.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0–Fundamental Algorithms for Scientific Computing in Python. *arXiv e-prints*, art. arXiv:1907.10121, Jul 2019.

Lionel A. Walford and William Edwards Deming. Statistical adjustment of data. *Copeia*, 1944:65, 1944.

Baoxiang Wang and Nidhi Hegde. Privacy-preserving q-learning with functional noise in continuous spaces. In *NeurIPS*, pages 11323–11333, 2019.

Fan Wang, Xiaomin Fang, Lihang Liu, Hao Tian, and Zhiming Peng. Mbcal: Sample efficient and variance reduced reinforcement learning for recommender systems. *arXiv preprint arXiv:1911.02248*, 2019a.

Haoran Wang and Shi Yu. Robo-advising: Enhancing investment with inverse optimization and deep reinforcement learning, 2021.

Huazheng Wang, Qian Zhao, Qingyun Wu, Shubham Chopra, Abhinav Khaitan, and Hongning Wang. Global and local differential privacy for collaborative bandits. In *Fourteenth ACM Conference on Recommender Systems*, RecSys '20, page 150–159. Association for Computing Machinery, 2020. ISBN 9781450375832.

Teng Wang, Jun Zhao, Xinyu Yang, and Xuebin Ren. Locally differentially private data collection and analysis. *arXiv preprint arXiv:1906.01777*, 2019b.

Yaozheng Wang, Dawei Feng, Dongsheng Li, Xinyuan Chen, Yunxiang Zhao, and Xin Niu. A mobile recommendation system based on logistic regression and gradient boosting decision trees. In *2016 international joint conference on neural networks (IJCNN)*, pages 1896–1902. IEEE, 2016.

Tsachy Weissman, Erik Ordentlich, Gadiel Seroussi, Sergio Verdu, and Marcelo J Weinberger. Inequalities for the l1 deviation of the empirical distribution. 2003.

Yifan Wu, Roshan Shariff, Tor Lattimore, and Csaba Szepesvári. Conservative bandits. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1254–1262. JMLR.org, 2016.

Lin F. Yang and Mengdi Wang. Reinforcement leaning in feature space: Matrix bandit, kernels, and regret bound. *CoRR*, abs/1905.10389, 2019.

Luting Yang, Jianyi Yang, and Shaolei Ren. Multi-feedback bandit learning with probabilistic contexts. In *IJCAI*, pages 3087–3093. ijcai.org, 2020.

Yunchang Yang, Tianhao Wu, Han Zhong, **Evrard Garcelon**, Matteo Pirotta, Alessandro Lazaric, Liwei Wang, and Simon Shaolei Du. A reduction-based framework for conservative bandits and reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=AcrlgZ9BKed.

Min Ye and Alexander Barg. Optimal schemes for discrete distribution estimation under locally differential privacy. *IEEE Trans. Inf. Theory*, 64(8):5662–5676, 2018.

Se-Young Yun, Jun Hyun Nam, Sangwoo Mo, and Jinwoo Shin. Contextual multi-armed bandits under feature uncertainty. *CoRR*, abs/1703.01347, 2017.

Andrea Zanette and Emma Brunskill. Tighter problem-dependent regret bounds in reinforcement learning without domain knowledge using value function bounds. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 7304–7312. PMLR, 2019.

X. Zhao and Ailan Wang. Generalized bootstrapping technique based on block equality test algorithm. *Secur. Commun. Networks*, 2018:9325082:1–9325082:8, 2018.

Xiangyu Zhao, Changsheng Gu, Haoshenglun Zhang, Xiwang Yang, Xiaobing Liu, Jiliang Tang, and Hui Liu. Dear: Deep reinforcement learning for online advertising impression in recommender systems, 2019. URL https://arxiv.org/abs/1909.03602.

Kai Zheng, Tianle Cai, Weiran Huang, Zhenguo Li, and Liwei Wang. Locally differentially private (contextual) bandits learning. *CoRR*, abs/2006.00701, 2020.

Zhaowei Zhu, Jingxuan Zhu, Ji Liu, and Yang Liu. Federated bandit: A gossiping approach. *CoRR*, abs/2010.12763, 2020.

Vincent Zhuang and Yanan Sui. No-regret reinforcement learning with heavy-tailed rewards, 2021. URL https://arxiv.org/abs/2102.12769.

ECOLE
DOCTORALE
DE MATHEMATIQUES
HADAMARD

Titre : Exploration sous Contrainte dans l'Apprentissage par Renforcement

Mots clés : Bandits, Exploration sous Contrainte, Apprentissage par Renforcement

Résumé : Une application majeure de l'apprentissage machine automatisée est la personnalisation de contenu recommandé à différents utilisateurs. Généralement, les algorithmes étant à la base de ces systèmes sont dit supervisé. C'est-à-dire que les données utilisées lors de la phase d'apprentissage proviennent de la même distribution. Cependant, les données sont générées par des interactions entre un utilisateur et ces mêmes algorithmes. Ainsi, les recommandations pour un utilisateur à un instant t peuvent modifier l'ensemble des recommandations pertinentes à un instant ultérieur. Il est alors nécessaire de prendre en compte ces interactions afin de produire un service de la meilleure qualité possible. Ce type d'interaction est réminiscent du problème d'apprentissage en ligne. Parmi les algorithmes dit en ligne, les algorithmes de bandits et d'apprentissage par renforcement (RL) semblent être les mieux positionnés afin de remplacer les méthodes d'apprentissage supervisé pour des applications nécessitant un certain degré de personnalisation. Le déploiement en production d'algorithmes d'apprentissage par renforcement présente un certain nombre de difficultés tel que garantir un certain niveau de performance lors des phases d'exploration ou encore comment garantir la confidentialité des données collectées. Dans cette thèse, nous considérons différentes contraintes freinant l'utilisation d'algorithmes d'apprentissage par renforcement, en fournissant des résultats à la fois empirique et théorique sur la vitesse d'apprentissage en présence de différentes contraintes.

Title : Constrained Exploration in Reinforcement Learning

Keywords : Bandits, Constrained Exploration, Reinforcement Learning

Abstract : A major application of machine learning is to provide personnalized content for different users. In general, the algorithms providing those recommandation are supervised learning algorithm. That is to say, the data used to train those algorithms are assumed to be sampled from the same distribution. However, those data are generated through interactions between the users and the recommendation algorithms. Thus, recommendations for a user at time t can have an impact on the set of pertinent recommandation at a later time. Therefore, it is necessary to take those interactions into account. This setting is reminiscent of the online learning setting. Among online learning algorithms, Reinforcement Learning algorithms (RL) are the most promising to replace supervised learning algorithms for applications requiring a certain degree of personnalization. The deployement in production of RL algorithms presents some challenges such as being able to guarantee a certain level of performance during exploration phases or how to guarantee privacy of the data collected by RL algorithms. In this thesis, we consider different constraints limiting the use of RL algorithms and provides both empirical and theoretical results on the impact of those constraints on the learning process.

Institut Polytechnique de Paris
91120 Palaiseau, France