

Imperial College London
Department of Mathematics

**Delays, Cooperation and Pairwise Comparisons in
Sequential Decision-Making Processes**

by Benjamin Matthew Howson

Submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

April 2025

Statement of Originality

The work presented in this thesis is my own and all work that has not been conducted by myself has been appropriately referenced.

Benjamin Howson

Copyright

The copyright of this thesis rests with the author. Unless otherwise indicated, its contents are licensed under a Creative Commons Attribution-Non Commercial 4.0 International Licence (CC BY-NC).

Under this licence, you may copy and redistribute the material in any medium or format. You may also create and distribute modified versions of the work. This is on the condition that: you credit the author and do not use it, or any derivative works, for a commercial purpose.

When reusing or sharing this work, ensure you make the licence terms clear to others by naming the licence and linking to the licence text. Where a work has been adapted, you should indicate that the work has been changed and describe those changes.

Please seek permission from the copyright holder for uses of this work that are not included in this licence or permitted under UK Copyright Law.

Abstract

This thesis presents novel algorithms for sequential decision-making in uncertain environments, focussing on the challenges introduced by complex feedback structures. Specifically, we study scenarios that frequently arise in real-world applications, including delayed feedback, cooperation in multi-agent systems, and learning from pairwise comparisons.

First, we study delayed feedback. Here, the decision-maker sequentially selects actions, but the corresponding feedback arrives later. For generalised linear bandits, we propose an algorithm and prove an upper bound on its regret. This guarantee shows that the delays cause a small additive penalty on top of what is possible with immediate feedback, substantially improving upon existing results. Subsequently, we extend the study of delayed feedback to episodic reinforcement learning and propose two approaches to handle the delays. For each approach, we provide regret bounds for numerous immediate feedback algorithms and prove that the price for delays is an additive penalty.

Second, we study problem settings where numerous cooperative decision-makers interact with the environment. For these problems, we propose a black-box algorithm that takes any single-agent algorithm as input and turns it into a cooperative multi-agent algorithm. Additionally, we prove that the black-box algorithm transfers the theoretical guarantees of any single-agent algorithm to the multi-agent setting. These guarantees are comparable to, or better than, existing works.

Finally, we study learning from pairwise comparisons, where the decision-maker selects two actions and receives the preferred action as their feedback. For this setting, we propose an algorithm that collects batches of preferences and uses this information to learn a policy that generates preferable actions. We prove that the cumulative sub-optimality of our algorithm grows sub-linearly without requiring strong assumptions on the preference structure.

Overall, this thesis provides practical algorithms with theoretical guarantees. These results make a step towards bridging the gap between idealised models for sequential decision-making and real-world challenges.

Acknowledgements

First, I would like to thank my wonderful supervisors, Ciara Pike-Burke and Sarah Filippi. The past four years have been thoroughly enjoyable. This is due to your guidance and the freedom you gave me to pursue my interests. Our weekly meetings, often running generously over time, were a consistent source of learning and support, and have had an immeasurable impact on my development as a researcher. I could not have asked for better supervisors. Additionally, I would like to thank Claire Vernade for making me feel so welcome in Tübingen. This short academic visit was an amazing experience that I will never forget.

Joseph, Pavrithra, Michael, James, Guiomar and Emmeran. I have enjoyed the time we have spent together at Imperial and the many conversations we have had over the years. Your camaraderie and support have made this journey all the more rewarding.

Matt, Scott and Rohan. Thank you for all the fun. From the holidays in Cornwall to the numerous nights out. Matt, thank you for always checking in on me and for letting me crash on your sofa.

Mum, Dad, Laura, Grandma, and Grandad. I am lucky to have such a loving family. Special thanks must go to my parents. Dad, you are the reason I found my interest in science. I would not be where I am today without your support and encouragement to pursue this interest further. Mum, you have always had more belief in me than I have had in myself, and you have never failed to put me first. Your selflessness has never gone unnoticed.

Finally, Chloe. Your unwavering support and belief over the past four years means more to me than I can ever fully express. Your patience, kindness, and quiet sacrifices are what have made all of this possible. For this, and so much more, I am eternally grateful.

Contents

Abstract	2
Acknowledgements	3
1 Introduction	12
1.1 Contributions	14
1.2 Thesis Organisation	15
I Background Material	17
2 Multi-Armed Bandits	18
2.1 Problem Setting	19
2.1.1 Measuring Performance	20
2.1.2 Standard Assumptions	22
2.1.3 Fundamental Limits	24
2.2 Algorithmic Approaches	25
2.2.1 The Optimistic Principle	27
2.2.2 Thompson Sampling	31
2.2.3 Other Principles	34
2.3 Linear Bandits	36
2.3.1 Fundamental Limits	38
2.3.2 The Optimistic Principle	39
2.3.3 Thompson Sampling	42
2.3.4 Beyond Linearity	43

3	Reinforcement Learning	44
3.1	Problem Setting	45
3.1.1	The Environment	45
3.1.2	The Value Functions	46
3.1.3	Dynamic Programming	49
3.1.4	Reinforcement Learning	50
3.2	Algorithmic Principles	52
3.2.1	Model-Based Optimism	53
3.2.2	Unifying Model-Based Optimism	55
3.2.3	Other Principles	56
II	Delayed Feedback	58
4	Delayed Feedback in Generalised Linear Bandits	59
4.1	Motivation	59
4.2	Problem Setting	60
4.2.1	Additional Assumptions	62
4.2.2	Useful Notation	64
4.3	Related Work	65
4.4	Delayed OFU-GLM	69
4.4.1	Input Parameters	70
4.4.2	Estimating the Unknown Parameters	70
4.5	The Confidence Sets	72
4.5.1	Preliminary Results for Theorem 4.1	72
4.5.2	Proof of Theorem 4.1	78
4.6	The Delayed Elliptical Potential	79
4.6.1	Preliminary Results for the Proof of Lemma 4.6	83
4.6.2	Proof of Lemma 4.6	87
4.7	The Pseudo-Regret of Delayed OFU-GLM	88
4.7.1	Discussion of the Result	92
4.8	Experimental Evaluation	94
4.9	Conclusion	97

5	Delayed Feedback in Episodic Reinforcement Learning	99
5.1	Introduction	99
5.1.1	Existing & Related Work	100
5.2	Problem Setting	101
5.2.1	Delayed Feedback	102
5.3	Decomposing the Pseudo-Regret	105
5.3.1	Preliminary Results for Lemma 5.1	106
5.3.2	Proof of Lemma 5.1	110
5.4	Active Updating	111
5.4.1	Preliminary Results for Theorem 5.1	112
5.4.2	Proof of Theorem 5.1	115
5.5	Lazy Updating	117
5.5.1	Preliminary Results for Theorem 5.2	119
5.5.2	Proof of Theorem 5.2	124
5.6	Discussion of Active & Lazy Updating	127
5.7	Experimental Results	129
5.8	Conclusion	131
III	Cooperation & Pairwise Comparisons	133
6	Cooperative Multi-Armed Bandits	134
6.1	Motivation	134
6.2	Problem Setting	135
6.2.1	Measuring Performance	137
6.2.2	Lower Bounds for Edge Cases	137
6.3	Literature Review	138
6.3.1	Gossip-Based Solutions	138
6.3.2	Leader-Follower Solutions	142
6.3.3	Related Problems	145
6.4	Black-Box Reduction	147
6.4.1	Message-Passing	148
6.4.2	Equivalence Between the Queues and the Environment	150
6.4.3	Guarantees on the Group Regret	152

6.5	Instances of QuACK	157
6.5.1	Standard Bandits	157
6.5.2	Heavy-Tailed Bandits	161
6.5.3	Duelling Bandits	163
6.5.4	Local Differential Privacy	164
6.6	Implementation Details	165
6.6.1	Shortest Path Tree	165
6.6.2	Leader Election	166
6.6.3	Optimising the Leader	167
6.7	Experimental Results	167
6.7.1	Standard Bandits	167
6.7.2	Heavy-Tailed Bandits	169
6.8	Conclusion	171
7	Pairwise Comparisons	172
7.1	Introduction	172
7.2	Problem Setting	173
7.2.1	Measuring Performance	176
7.3	Related Work	177
7.3.1	Duelling Bandits	178
7.3.2	Reward-Based Preferences	178
7.3.3	Reward-Free Preferences	181
7.4	An Algorithm for Identity Preference Optimisation	182
7.4.1	Preliminary Results for Theorem 7.1	188
7.4.2	Proof of Theorem 7.1	194
7.5	Conclusion	196
8	Conclusion	199
8.1	Summary of Contributions	199
8.2	Future Directions	201
8.2.1	Delayed Feedback	201
8.2.2	Multi-Agent Bandits	203
8.2.3	Pairwise Comparisons	204
8.3	Closing Remarks	205

A Useful Results	227
A.1 Concentration of Measure	227
B Additional Experimental Results	228
B.1 Chapter 5	228
B.1.1 Chain Environment with $H = S = 5$	228
B.1.2 Chain Environment with $H = S = 10$	229
B.1.3 Chain Environment with $H = S = 20$	230
B.1.4 Chain Environment with $H = S = 30$	231
B.2 Chapter 6	232
B.2.1 Cycle Graph	232
B.2.2 Grid Graph.	232
B.2.3 Star Graph.	233

List of Figures

2.1	Visualisation of the optimistic principle. The symbols $\star$ and $\bullet$ represent the true expected reward and an empirical estimate, respectively. The shaded region represents the set of statistically plausible values for the unknown expected reward of each action. Here, $A_t = 2$ because it is the action with the highest upper confidence bound, as depicted by the shaded regions.	27
4.1	Experimental results for $d = 5$ and $\mathbb{E}[\tau] = 100$. The boundary of the shaded regions represent the minimum and maximum pseudo-regret across the 30 independent runs.	95
4.2	Experimental results for $d = 10$ and $\mathbb{E}[\tau] = 100$. The boundary of the shaded regions represent the minimum and maximum pseudo-regret across the 30 independent runs.	95
4.3	Experimental results for $d = 20$ and $\mathbb{E}[\tau] = 100$. The boundary of the shaded regions represent the minimum and maximum pseudo-regret across the 30 independent runs.	95
4.4	Experimental results visualising the impact of the expected value and distribution of the delays on the final-round pseudo-regret of Algorithm 3. The boundary of the shaded regions represent the minimum and maximum pseudo-regret across the 30 independent runs.	96
5.1	Experimental results visualising the growth of the pseudo-regret as a function of the episode. These results correspond to the chain environment with $H = S = 30$.	130
5.2	Experimental results depicting the growth of the pseudo-regret at the end of the final episode as a function of the expected delay. These results correspond to the chain environment with $H = S = 30$	131

6.1	The Grid Graph and its Shortest Path Tree when the leader is the central vertex.	149
6.2	The group regret of various cooperative multi-armed bandit algorithms on the cycle, grid and star graphs with $m = 196$ agents.	169
6.3	The final round group regret as a function of the network size in the cycle, grid and star graphs.	169
6.4	The final round group regret as a function of the network size on cycle graphs of varying sizes.	170
B.1	Cumulative Regret ($S = 5, \mathbb{E}[\tau] = 100$).	228
B.2	Cumulative Regret ($S = 5, \mathbb{E}[\tau] = 300$).	229
B.3	Cumulative Regret ($S = 5, \mathbb{E}[\tau] = 500$).	229
B.4	Delay Dependence ($S = 5$).	229
B.5	Cumulative Regret ($S = 10, \mathbb{E}[\tau] = 100$).	229
B.6	Cumulative Regret ($S = 10, \mathbb{E}[\tau] = 300$).	230
B.7	Cumulative Regret ($S = 10, \mathbb{E}[\tau] = 500$).	230
B.8	Delay Dependence ($S = 10$).	230
B.9	Cumulative Regret ($S = 20, \mathbb{E}[\tau] = 100$).	230
B.10	Cumulative Regret ($S = 20, \mathbb{E}[\tau] = 300$).	231
B.11	Cumulative Regret ($S = 20, \mathbb{E}[\tau] = 500$).	231
B.12	Delay Dependence ($S = 20$).	231
B.13	Cumulative Regret ($S = 30, \mathbb{E}[\tau] = 300$).	231
B.14	Cumulative Regret ($S = 30, \mathbb{E}[\tau] = 500$).	232
B.15	The group regret of various cooperative multi-armed bandit algorithms on cycle graphs of varying sizes.	232
B.16	The group regret of various cooperative multi-armed bandit algorithms on grid graphs of varying sizes.	233
B.17	The group regret of various cooperative multi-armed bandit algorithms on star graphs of varying sizes.	233

List of Tables

- 3.1 Theoretical guarantees for model-optimistic episodic reinforcement learning algorithms. This table has been extracted from the results presented by [Neu and Pike-Burke \(2020\)](#). Here, $\Gamma = \max_{s,a,h} \sum_{s'} 1\{P_h(s' | s, a) > 0\}$ is an upper bound on the number of states that can be reached from any state-action-stage triple. . 54
- 5.1 A selection of upper bounds on the pseudo-regret of various immediate feedback algorithms when paired with Algorithms 4 and 5 in the delayed feedback setting. 128

Chapter 1

Introduction

The problem of decision-making with uncertain information is an important challenge in the field of machine learning. The multi-armed bandit and reinforcement learning problems have emerged as the most popular mathematical models that capture many of the salient features involved in sequential decision-making processes (Lattimore and Szepesvári, 2020; Sutton and Barto, 2020). Of these two problems, the multi-armed bandit is the simplest, and it gets its name from an analogy with a problem faced by a gambler who is standing in a casino filled with slot machines. The gambler is sensible and has a limited budget that they are willing to spend. Each machine, when chosen, consumes a portion of the gambler's limited budget and reveals a random payoff. Naturally, the gambler wants to maximise their total payoff. However, they do not know the location of the slot machine with the largest payoff, and the gambler can only estimate the payoffs from the random values generated by each slot machine. How many times and in what order should the gambler play each machine to maximise their total reward? Before spending each unit of their budget, the gambler is presented with the following dilemma: should they explore lesser-known machines in the hopes of discovering hidden potential, or exploit the ones that have given the highest payoffs in the past? This is the *exploration-exploitation dilemma* and it is a feature of sequential decision-making processes where the outcomes related to each decision are uncertain. The dilemma arises because the gambler who exclusively explores will never capitalise on the best machine, while the gambler who only exploits their current information risks missing out on discovering slot machines with higher payoffs.

This casino analogy serves as a foundation for the stochastic multi-armed bandit problem, which is a mathematical framework that isolates the challenge of balancing exploration and exploitation.

Consequently, the multi-armed bandit problem has become the primary framework for studying how to address this challenge effectively. Notably, many of the insights from the stochastic multi-armed bandit carry over into more complex problems. For example, when the slot machines generate correlated payoffs, we have a structured bandit problem. Alternatively, if playing a slot machine changes the state of the casino, in a certain technical sense, we have the reinforcement learning problem. There have been several books and articles on the topic of bandits and reinforcement learning that are highly recommended ([Lattimore and Szepesvári, 2020](#); [Slivkins, 2019](#); [Bubeck and Cesa-Bianchi, 2012](#); [Sutton and Barto, 2020](#)).

One of the central theoretical challenges in sequential decision-making is designing algorithms and quantifying how well they perform. Typically, the performance is measured by the regret. Informally, this sums the difference between the highest possible payoff the agent could have received and the payoff they actually received across the rounds. The objective is to design algorithms that minimise the regret and this requires algorithms that quickly learn an optimal or near-optimal strategy. Having algorithms with low regret is particularly important in high-stakes applications. Consequently, there has been an extensive research effort to design algorithms that provably minimise the regret in both the multi-armed bandit and reinforcement learning models for sequential decision-making. Some of these algorithms guarantee that the regret is as small as is information-theoretically possible ([Audibert and Bubeck, 2009](#); [Azar et al., 2017](#)).

Despite these advances, standard guarantees on the regret often rely on idealised assumptions that do not hold in real-world applications. For example, classical models assume that feedback is immediate. Yet, in settings like online advertising ([Chapelle and Li, 2011](#)), clinical trials ([Simon, 1977](#); [Villar et al., 2015](#)), and e-commerce ([Zhang et al., 2024](#)), there is often a delay between playing an action and receiving the corresponding feedback. Similarly, traditional models consider a single agent interacting with the environment, while many real-world scenarios involve multiple agents cooperating or competing ([Tran-Thanh et al., 2011](#); [Tekin et al., 2014](#)). Additionally, direct feedback on decision quality is not always available or practical. Instead, learning from pairwise comparisons, where feedback is given in terms of relative preferences rather than absolute rewards, offers a more scalable alternative ([Christiano et al., 2017](#)).

1.1 Contributions

This thesis explores extensions of the multi-armed bandit and reinforcement learning problems, addressing challenges that arise in real-world learning systems. Specifically, it tackles three fundamental issues: learning under delayed feedback, cooperative decision-making in multi-agent settings, and learning from pairwise comparisons instead of direct rewards.

The central theme of Chapters 4 and 5 is delayed feedback, which is a major obstacle in sequential decision-making that arises in many real-world applications. Here, an agent does not receive immediate feedback about the quality of its decisions. Instead, this information is revealed later, forcing the agent to continue making choices without knowing how well past actions performed. These two chapters investigate the impact of such delays in different learning settings and are each based on published work ([Howson et al., 2023a,b](#)).

Chapter 4 begins the study of stochastically delayed feedback in generalised linear bandits, which is a class of structured bandit problems where rewards depend on actions through a generalised linear model. The chapter introduces a novel algorithm that can efficiently handle delayed feedback without requiring prior knowledge of the delay distribution. Unlike previous approaches, which show that delays cause a multiplicative increase in regret, our algorithm ensures that delays introduce only a small additive penalty. This theoretical advancement is complemented by empirical results demonstrating substantial improvements over existing methods in simulated environments.

Subsequently, Chapter 5 seeks to extend the results that we obtained for delayed feedback in the bandit setting to the episodic reinforcement learning problem. This chapter proposes two approaches for handling delayed feedback in episodic reinforcement learning. The first involves incorporating new information as soon as it becomes available, whereas the second waits before using newly observed information. These approaches are generic and can be used in conjunction with any immediate feedback algorithm. Theoretically, for both approaches, we prove that the delays cause only a small additive increase in the regret for many existing immediate feedback algorithms. Empirically, we compare these two algorithms and show that waiting before updating does not significantly degrade the performance in simulated experiments.

Moving beyond delayed feedback, Chapter 6 explores cooperative learning in multi-agent bandits, and is based on published work ([Howson et al., 2025](#)). Here, there is a network of decision-makers who are working together to maximise their collective performance. For this setting, we propose a general-purpose black-box algorithm that transforms any single-agent bandit algorithm into a cooperative multi-agent bandit algorithm. The proposed algorithm ensures that the theoretical guarantees of the single-agent algorithm transfer directly to the multi-agent setting. Furthermore, this guarantee holds under mild assumptions, which means that the black-box algorithm can be used to simplify the design of cooperative algorithms for multi-agent extensions of various bandit problems. Additionally, we show that our black-box approach is competitive with or superior to existing multi-agent approaches, both theoretically and empirically.

Finally, Chapter 7 investigates learning from pairwise comparisons rather than direct rewards, a setting that has become particularly relevant due to its application in fine-tuning large language models. In this setting, the agent uses a policy to select a pair of actions, and the feedback it receives indicates the preferred action. Unlike previous works, which focus exclusively on learning from a fixed dataset or updating immediately after receiving each feedback, we focus on the intermediate and practical setting of updating after selecting a batch of action pairs. For this setting, we propose an algorithm and prove theoretical guarantees that hold for any batching scheme, without imposing strong assumptions on the preference-generating process.

1.2 Thesis Organisation

This thesis begins by introducing the relevant background material. Chapters 2 and 3 provide an introduction to the multi-armed bandit and reinforcement learning problems, respectively. Here, the reader can find an overview of the problem settings and a description of popular principles for designing theoretically justified algorithms. Subsequently, Chapters 4-7 contain the main contributions of this thesis.

- Chapter 4 studies the generalised linear bandit problem with delayed feedback ([Howson et al., 2023b](#)).
- Chapter 5 continues to investigate the impact of delayed feedback, but in the setting of episodic reinforcement learning ([Howson et al., 2023a](#)).
- Chapter 6 revisits the multi-armed bandit problem but investigates the setting where there

are multiple agents interacting with the environment ([Howson et al., 2025](#)).

- Chapter 7 investigates the problem of learning from pairwise comparisons in the bandit setting.

Finally, Chapter 8 summarises the research conducted within the thesis and presents several avenues for future research.

Part I

Background Material

Chapter 2

Multi-Armed Bandits

[Thompson \(1933\)](#) introduced the stochastic multi-armed bandit problem that is associated with the casino analogy from the introduction. However, his motivation was not maximising his fortune in the casino. Instead, Thompson was interested in clinical trials with two treatments. This fictitious trial consisted of patients who arrived one by one and a clinician who prescribed each patient one of the two treatments. These treatments either did or did not cure the disease of interest. Thompson modelled the outcome associated with prescribing each treatment as a random variable drawn from an unknown Bernoulli distribution. Here, each treatment was associated with an unknown probability of curing the patient. His work focused on a setting where the clinician could adjust the probability of assigning each treatment based on the outcomes of previous patients. Primarily, Thompson wanted to know how the clinician should use the historical information to adjust the assignment probabilities in a manner that maximises the chances of each patient receiving the most effective treatment. The difficulty faced by the clinician comes from the need to balance exploration and exploitation. On the one hand, they should explore each treatment sufficiently to learn how effective it is at curing the disease. On the other hand, they should exploit the treatment that appears to be the most effective. Later, [Robbins \(1952\)](#) and [Lai and Robbins \(1985\)](#) revisited this model for sequential decision-making and are credited with popularising what is now known as the stochastic multi-armed bandit problem.

2.1 Problem Setting

Formally, the stochastic multi-armed bandit problem consists of a sequential interaction between an *agent* and an *environment*. The environment consists of a finite set of decision variables, which are called *actions* or *arms*:

$$\mathcal{A} = \{1, 2, \dots, k\}.$$

Each $a \in \mathcal{A}$ in the environment is associated with an underlying probability distribution over the *rewards*, which are all assumed to remain fixed throughout time:

$$\nu = \{P_1, P_2, \dots, P_k\}$$

where P_a denotes the univariate cumulative distribution function for the corresponding action. Typically, the minimal assumption of a finite expected value is imposed on these distributions:

$$\mu_a(\nu) = \int_{-\infty}^{\infty} x dP_a(x) < \infty$$

for all $a \in \mathcal{A}$. [Thompson \(1933\)](#), for instance, considered the setting where the set of actions are the available treatments and the reward distribution for each action is the Bernoulli distribution.

Regardless of the specific details of the environment, the interaction between the agent and the environment occurs at discrete points in time known as *rounds*. Each round involves the agent selecting an action and the environment revealing a reward that is randomly drawn from the distribution corresponding to the chosen action. Suppose $A_t \in \mathcal{A}$ is chosen in the t -th round by the agent. Then, the environment will randomly generate a reward from the corresponding probability distribution:

$$X_t = X_t(A_t) \stackrel{\text{i.i.d.}}{\sim} P_{A_t}$$

Choosing $A_t \in \mathcal{A}$ is done using a *policy*, π , which is just a sequence of mappings from the observed history to a distribution over the set of actions:

$$\pi = \{\pi_1, \pi_2, \dots\} \text{ where } \pi_t(a) = \mathbb{P}(A_t = a \mid A_1, X_1, \dots, A_{t-1}, X_{t-1}) \quad (2.1)$$

which captures the intuitive fact that the agent cannot rely on future information to make their decisions in the present. However, it does allow the agent to make their decisions based on all of

the historical information.

Procedure 2.1: Multi-Armed Bandits

For $t \in \{1, 2, \dots, n\}$:

1. **Agent.** Selects $A_t \sim \pi_t(\cdot)$
2. **Environment.** Reveals $X_t = X_t(A_t) \stackrel{\text{iid}}{\sim} P_{A_t}$

Procedure 2.1 brings all of the above information together and formalises the sequential interaction between the agent and the environment in the stochastic multi-armed bandit problem.

2.1.1 Measuring Performance

[Robbins \(1952\)](#), in his seminal paper, asked how actions should be chosen to maximise the expected value of the following summation:

$$S_n = \sum_{t=1}^n X_t$$

where $n \in \mathbb{N}$ is known as the *horizon* of the game. Generally, S_n is a random variable because the rewards themselves are random. However, an additional source of randomness comes from the policy, which is permitted to select actions based on the historical sequence of actions and rewards. Therefore, the selected action inherits some randomness. This is true even if the policy is deterministic given the history. Equation (2.2) defines the expected value of the summation in a manner that highlights the two sources of randomness:

$$S_n(\pi, \nu) = \sum_{t=1}^n \mathbb{E}[X_t] \tag{2.2}$$

where the expectation is with respect to the policy and the set of reward distributions that define the bandit environment. Equation (2.2) is maximised by the policy that always selects the action with the highest expected reward:

$$a_\star = \arg \max_{a \in \mathcal{A}} \mu_a(\nu) \quad \text{and} \quad \mu_\star(\nu) = \mu_{a_\star}(\nu) = \max_{a \in \mathcal{A}} \mu_a(\nu) \tag{2.3}$$

where $a_\star$ is called the *optimal action*. Therefore, a natural objective is to design policies that accumulate an expected sum of rewards that is similar to that of always selecting the optimal

action. This leads to the main measure of performance in the literature. The *regret* is defined as follows:

$$R_n(\pi, \nu) = n \cdot \mu_\star(\nu) - \sum_{t=1}^n \mathbb{E}[X_t] . \quad (2.4)$$

Equation (2.4) measures the cumulative expected loss in rewards due to not knowing the reward distributions. Here, the expectation includes the randomness from the rewards and from the policy. Generally, the expected loss in rewards for picking a sub-optimal action is referred to as the *sub-optimality gap* or the *gap* in the literature:

$$\Delta_a(\nu) = \mu_\star(\nu) - \mu_a(\nu) . \quad (2.5)$$

Often, Equation (2.4) is rewritten in terms of the sub-optimality gaps and the number of times the agent plays each action. The number of plays is an important quantity and has the following definition:

$$T_a(n) = \sum_{t=1}^n 1\{A_t = a\} \quad (2.6)$$

for all $n \in \mathbb{N}$. This quantity is a random variable because the agent can explicitly use randomisation to select their actions or choose their action using a deterministic function of the historical sequence of rewards, which are random samples from unknown distributions. Equations (2.5) and (2.6) can be used to write down an equivalent expression for the cumulative regret:

$$R_n(\pi, \nu) = \sum_{a \neq \star} \Delta_a(\nu) \cdot \mathbb{E}[T_a(n)] \quad (2.7)$$

which follows from a straightforward application of the tower rule.

The regret is defined as an expectation with respect to interaction between the policy and the environment. However, there is another measure of performance known as the *pseudo-regret*:

$$\hat{R}_n(\pi, \nu) = n \cdot \mu_\star(\nu) - \sum_{t=1}^n \mu_{A_t}(\nu) \quad (2.8)$$

which filters out some of the randomness in the rewards. The multi-armed bandit literature tends to focus on the regret given in Equation (2.4). However, Section 2.3 introduces the linear bandit problem, which tends to use the pseudo-regret as the measure of performance. Furthermore, Chapter 3 introduces episodic reinforcement learning, where it is common to focus on an appropriately modified version of the pseudo-regret.

2.1.2 Standard Assumptions

The regret explicitly depends on three quantities: the policy, the environment, and the number of rounds n involved in the summation. Often, n is referred to as the *horizon* of the sequential decision-making problem. Therefore, when designing policies, incorporating any available knowledge of these factors can enhance decision-making and improve the agent's ability to minimise their regret.

Horizon. Generally, assuming that the horizon is known is done for mathematical convenience. Converting policies that require knowledge of the horizon into policies that do not require knowledge of the horizon can be done through the doubling trick (Auer et al., 1995; Besson and Kaufmann, 2018). Essentially, the doubling trick takes in an initial guess of the horizon and instructs the agent to select their actions as if this were the actual horizon. Once the agent hits this horizon, the entire history is deleted and the policy is restarted with a new value for the horizon, which is typically double the previous guess. Alternatively, the theoretical analysis can be adjusted to avoid requiring knowledge of the horizon. However, this can become quite complex, especially when it is of interest to retain certain theoretical properties (Degenne and Perchet, 2016). Although requiring knowledge of the horizon might seem unreasonable, it does occur in practical applications where the experimental budget is limited, such as online experimentation and clinical trials. Exploiting knowledge of this quantity allows for the design of policies that favour exploitation towards the end of the interaction, where the benefits of exploration have diminishing returns.

Environment. Even if the horizon is known, the exact nature of the distributions that characterise the multi-armed bandit environment are unknown. Maximising Equation (2.2) for a particular environment is straightforward and is a matter of always selecting the action with the highest expected reward. However, such a policy will incur linear regret in all environments whose optimal action has a different index. Therefore, it is common to assume that the true bandit environment belongs to a restricted class of distributions, $\mathcal{E}$. Naturally, the expectation is that policies tailored towards a specific class of environments will suffer low regret against any environment that it faces within the class. Thompson (1933), for example, restricted himself to the class of environments where each distribution over the rewards is Bernoulli:

$$\mathcal{E}_B = \{(\text{Bernoulli}(\mu_a))_{a \in \mathcal{A}} : \mu \in [0, 1]^k\}$$

which is now known as the Bernoulli bandit. Environments that belong to this class have distributions whose variances decrease as their expected value approaches the boundary points. This prior knowledge could be incorporated when designing the policy. The Gaussian bandit is another class of environments where each action has a reward distribution with a specific parametric form (Lai and Robbins, 1985; Katehakis and Robbins, 1995):

$$\mathcal{E}_G = \left\{ (\text{Normal}(\mu_a, \sigma^2))_{a \in \mathcal{A}} : \mu \in [0, 1]^k \right\}$$

where σ^2 , or an upper bound thereof, is known. However, it might be unreasonable to make parametric assumptions about the class of distributions. Instead, it might be more appropriate to assume that the set of reward distributions generate bounded random variables. Auer et al. (2002a) take this non-parametric approach and focus on designing policies for environments with bounded support:

$$\mathcal{E}_{[0,1]} = \left\{ (P_a)_{a \in \mathcal{A}} : \text{Supp}(P_a) \subseteq [0, 1] \right\}$$

which contains the class of Bernoulli bandits and many other parametric environments as special cases. Even more classes of bandit environments exist. Bubeck and Cesa-Bianchi (2012) popularised the subgaussian bandit environment, which places conditions on the moment-generating function of the noise around the expected rewards of each action.

Definition 2.1. *A random variable X is σ -subgaussian if there exists a positive scalar $\sigma > 0$ such that, for all $\lambda \in \mathbb{R}$, the following inequality holds:*

$$\mathbb{E}[\exp(\lambda(X - \mathbb{E}[X]))] \leq \exp\left(\frac{\lambda^2 \sigma^2}{2}\right).$$

Definition 2.1 says that the distribution of the centred random variable has tails that decay at least as fast as those of the Gaussian distribution. Typically, the subgaussian parameter σ is assumed to be known and this leads to the following class of environments:

$$\mathcal{E}_{SG} = \left\{ (P_a)_{a \in \mathcal{A}} : P_a \text{ is } \sigma\text{-subgaussian for all } a \right\}.$$

This class of bandit environments contains all the previous examples as special cases. Nevertheless, there exist even larger classes of environments that subsume the σ -subgaussian setting, such as distributions with bounded kurtosis (Lattimore, 2017) and heavy tails (Bubeck et al., 2013a).

2.1.3 Fundamental Limits

Equation (2.7) makes it immediately clear that the regret of any policy is non-negative and is equal to zero when the agent selects the optimal action in every round. Furthermore, we can upper bound the regret of any policy by the strategy of always choosing the action with the lowest expected reward. These simple insights reveal that the regret can be bounded as follows:

$$0 \leq R_n(\pi, \nu) \leq n \cdot \max_{a \in \mathcal{A}} \Delta_a$$

where the upper bound has a linear dependence on the horizon. This is often referred to as *linear regret* and it corresponds to the worst possible policy for the environment. Therefore, the objective is to design policies for a class of environments whose regret is a *sub-linear* function of the horizon. However, this leads to the fundamental question: *what is the lowest cumulative regret achievable by any policy?*

There exist two types of answer to this question for the stochastic multi-armed bandit problem. These are the instance-dependent or problem-dependent lower bounds, and the minimax or problem-independent lower bounds.

Instance/Problem Dependent. Informally, instance-dependent, or problem-dependent, lower bounds can be viewed as a function that takes a specific multi-armed bandit environment as input and returns the smallest possible value for the regret that is attainable by some class of policies. [Lai and Robbins \(1985\)](#), in their seminal paper, prove an instance-dependent asymptotic lower bound on the cumulative regret for all *consistent* policies:

$$\lim_{n \rightarrow \infty} \frac{R_n(\pi, \nu)}{\ln(n)} \geq \sum_{a \neq \star} \frac{\Delta_a(\nu)}{D_{\text{KL}}(P_a \| P_\star)} \quad (2.9)$$

where $D_{\text{KL}}(P \| P')$ denotes the relative entropy between the distributions. Consistency is an asymptotic notation and contains the set of all policies that eventually find and exploit the action with the highest expected reward. Formally, π is consistent if:

$$\lim_{n \rightarrow \infty} \frac{R_n(\pi, \nu)}{n^\alpha} \rightarrow 0$$

for every $\alpha > 0$. This is extremely useful for characterising what is possible in terms of the regret in a particular bandit environment for many algorithms of interest. Furthermore, Equation (2.9)

suggests that the number of times an asymptotically optimal policy will play a sub-optimal action in n rounds is approximately:

$$\mathbb{E}[T_a(n)] \approx \frac{\Delta_a(\nu)}{D_{\text{KL}}(P_a \| P_\star)} \ln(n) .$$

[Lai and Robbins \(1985\)](#) used this insight to design policies for various environments, such as the Gaussian and Bernoulli bandits from Section 2.1.2.

Minimax/Problem-Independent. The minimax, or problem-independent, lower bound takes another perspective. This kind of guarantee takes in a class of environments as input and provides a lower bound on the regret of the best policy within the class. [Auer et al. \(2002b\)](#) present a worst-case lower bound for the class of bandit environments with bounded support:

$$\inf_{\pi \in \Pi} \sup_{\nu \in \mathcal{E}_{[0,1]}} R_n(\pi, \nu) \geq \min \left\{ \frac{\sqrt{nk}}{20}, n \right\} . \quad (2.10)$$

[Lattimore and Szepesvári \(2020\)](#) verify a similar lower bound for the class of Gaussian bandits using the techniques introduced by [Bubeck et al. \(2013b\)](#).

2.2 Algorithmic Approaches

Equations (2.9) and (2.10) establish fundamental lower bounds on what can be expected of algorithms that seek to minimise the regret in multi-armed bandit problems. The objective is to design policies whose regret is as close as possible to these limits. Equation (2.7) is immediately useful for verifying that a careful balance of exploration and exploitation is necessary for this objective. Relying exclusively on exploration will fail to achieve sub-linear regret. For example, selecting an action uniformly at random in each round will yield:

$$R_n(\text{Uniform}, \nu) = \sum_{a \neq \star} \Delta_a(\nu) \cdot \left(\sum_{t=1}^n \mathbb{P}(A_t = a) \right) = \frac{n}{k} \sum_{a \neq \star} \Delta_a$$

where Uniform denotes the policy that sets $\pi_t(a) = 1/k$ for all $(t, a) \in \mathbb{N} \times \mathcal{A}$. Relying exclusively on exploitation will also fail to achieve sub-linear regret. For example, selecting each action once and committing to the action with the highest reward will give:

$$R_n(\text{Greedy}, \nu) = \sum_{a \neq \star} \Delta_a \cdot \left(\sum_{t=1}^n \mathbb{P}(A_t = a) \right)$$

$$= \sum_{a \neq a_*} \Delta_a + (n - k) \sum_{a \neq \star} \Delta_a(\nu) \mathbb{P} \left(a = \arg \max_{t \leq k} \{X_t\} \right)$$

where Greedy has $\pi_t(a) = 1$ for $t \in \{1, 2, \dots, k\}$ and is otherwise the action that received the highest reward. Thus, the regret is linear in environments where the optimal action cannot be identified with one sample from each action. Essentially, this is the class of all non-trivial bandit problems. See Chapter 6 of [Lattimore and Szepesvári \(2020\)](#) for an upper bound on the probability in the bound of the greedy policy.

These examples highlight that balancing exploration and exploitation is essential for regret minimisation. Remarkably, simple and efficient algorithms exist that nearly match these lower bounds, and the mechanisms that balance exploration and exploitation have been studied for many years. [Thompson \(1933\)](#) suggested that selecting an action should be done in accordance with the probability that it is optimal in the view of the available information. However, this approach remained relatively obscure for quite some time. Meanwhile, other approaches were introduced that were quicker to gain popularity. [Robbins \(1952\)](#) proposed a relatively straightforward idea of exploring in a carefully chosen set of rounds. [Anscombe \(1963\)](#) investigates a similar principle that better aligns with clinical trials. This principle involves an initial exploration phase that is followed by an exploitation phase that always selects the action with the highest empirical mean, which is now known as explore-then-commit. Subsequently, [Lai and Robbins \(1985\)](#) proposed using upper confidence bounds on the expected rewards to balance exploration and exploitation. [Auer et al. \(2002a\)](#) popularised this technique, and it has since become one of the most widely used strategies for multi-armed bandits and reinforcement learning. This approach has been continuously refined and extended since its inception.

Primarily, research tends to focus on extending ideas of [Thompson \(1933\)](#) and [Auer et al. \(2002a\)](#) beyond the canonical multi-armed bandit problem. This thesis will frequently make use of, or compare to, algorithms that build upon the principles underlying these algorithmic ideas. Therefore, Sections 2.2.1 and 2.2.2 present a thorough explanation of these approaches. Section 2.2.3 highlights other algorithmic principles for solving the multi-armed bandit problem for completeness, and serves the purpose of providing the corresponding references for the interested reader.

2.2.1 The Optimistic Principle

Algorithmically, the idea of optimism can be traced back to the seminal work of [Lai and Robbins \(1985\)](#). However, research within the field of psychology has independently discovered that optimism is exhibited by humans and animals in their decision-making processes ([Sharot, 2011](#); [O’Sullivan, 2015](#)). Generally, the optimistic principle suggests that an agent should select actions as if they were operating in the most favourable version of the environment, as warranted by the current statistical evidence. Loosely, the idea is to use past data to construct the highest reasonable estimate for the expected reward of each action and select the next action greedily based on these optimistic estimates. This approach works because the estimates incorporate uncertainty, making actions desirable for two reasons: either their true reward is high, or the uncertainty itself makes them worth exploring. Figure 2.1 illustrates what an optimistic policy might see and do in an arbitrary round of their interaction.

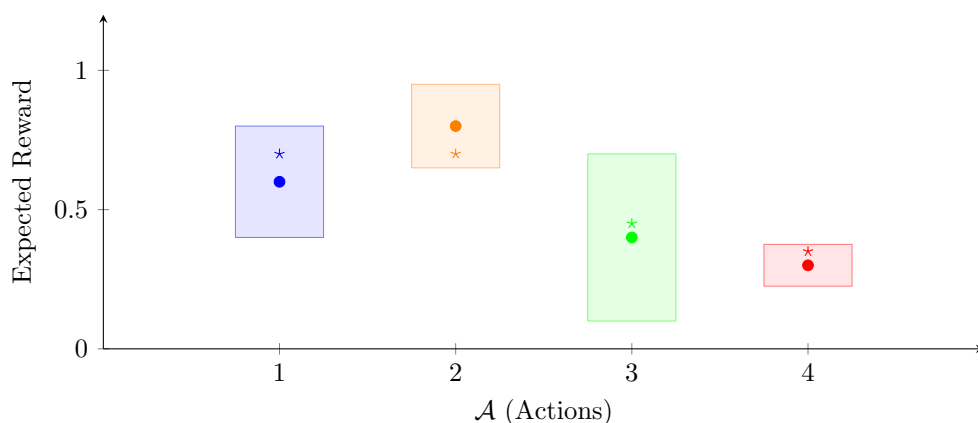


Figure 2.1: Visualisation of the optimistic principle. The symbols ★ and ● represent the true expected reward and an empirical estimate, respectively. The shaded region represents the set of statistically plausible values for the unknown expected reward of each action. Here, $A_t = 2$ because it is the action with the highest upper confidence bound, as depicted by the shaded regions.

Typically, implementing the optimistic principle is done through the use of confidence bounds to construct optimistic estimates, which was introduced by [Lai and Robbins \(1985\)](#). Subsequently, [Lai \(1987\)](#), [Katehakis and Robbins \(1995\)](#) and [Agrawal \(1995\)](#) built upon this idea and developed early versions of what is now known as the *upper confidence bound algorithm*. Notably, [Agrawal \(1995\)](#) observed all that is required is an appropriate sequence of upper bounds on the unknown expected reward for each action. These papers all studied the cumulative regret from an asymptotic point of view. [Auer et al. \(2002a\)](#) later leveraged the insight on the confidence

sequences and provided the first finite-time analysis of the cumulative regret for the upper confidence bound strategy in environments with bounded support.

The optimistic principle provides a widely applicable framework for sequential decision-making in uncertain environments. Algorithm 1 gives the generic pseudo-code of this principle, and serves as a useful reference for the remainder of this subsection.

Algorithm 1 Upper Confidence Bounds (UCB)

Initialise. Select $A_t = t$ for $t \in [1, k]$
for $t = k + 1, k + 2, \dots$ **do**
 Select $A_t = \arg \max_{a \in \mathcal{A}} \text{UCB}_{t-1}(a)$
end for

Auer et al. (2002a) popularised the upper confidence bound algorithm and focussed on bandit environments with support in $[0, 1]$. Constructing the upper confidence bound for each action starts with computing an empirical estimate of the expected reward:

$$\hat{\mu}_a(T_a(t)) = \frac{1}{T_a(t)} \cdot \sum_{s=1}^t X_s 1\{A_s = a\}$$

which is just the empirical mean of the rewards received by the end of the t -th round for action $a \in \mathcal{A}$. Hoeffding's inequality (Hoeffding, 1963) is then used to construct the upper confidence bounds for usage within Algorithm 1 as follows:

$$\text{UCB1}_t(a) = \hat{\mu}_a(T_a(t)) + \frac{\sqrt{2 \ln(t)}}{\sqrt{T_a(t)}}. \quad (2.11)$$

Notably, these optimistic estimates do not require knowledge of the horizon n because the upper confidence bounds depend only on the number of completed rounds, the empirical mean, and the number of plays for each action. Theorem 1 of Auer et al. (2002a) shows that the regret of the policy defined by the upper confidence bounds in Equation (2.11) satisfies:

$$R_n(\text{UCB1}, \nu) \leq \sum_{a \neq \star} \frac{8 \ln(n)}{\Delta_a(\nu)} + \left(1 + \frac{\pi^2}{3}\right) \cdot \sum_{a \neq \star} \Delta_a(\nu) \quad (2.12)$$

for all $\nu \in \mathcal{E}_{[0,1]}$. Remarkably, this result verifies that the upper confidence bounds in Equation (2.11) are enough to achieve the same logarithmic dependence on the horizon as the lower bound in Equation (2.9). Replacing Hoeffding's inequality with an appropriate tail bound for the random variables allows for the construction of optimistic estimates for broader classes of

environments, such as σ -subgaussian bandits, which can be used within Algorithm 1. Chapter 2 of [Bubeck and Cesa-Bianchi \(2012\)](#) make this generalisation and their analysis verifies that the instance-dependent upper bound matches the instance-dependent lower bound up to absolute constants for the class of Gaussian bandits.

Converting Equation (2.12) into an instance-independent guarantee is straightforward. This is achieved by partitioning the actions into sets where the sub-optimality gap is above or below some threshold. Choosing this threshold to make the regret as large as possible gives:

$$R_n(\text{UCB1}, \nu) \leq \sqrt{nk \ln(n)}$$

which matches the minimax lower bound up to the $\sqrt{\ln(n)}$ factor.

UCB1 of [Auer et al. \(2002a\)](#) is easy to implement and only requires storing two lists of k elements; one for tracking the empirical mean for each action, and another for counting the number of times each action has been selected. These lists are sufficient for computing the upper confidence bounds required to execute the policy in each round. Furthermore, computing the upper confidence bounds only involves elementary operations. This means the time complexity is of the order k in each round, which arises from having to compute the upper confidence bound for each action in every round.

However, UCB1 is not perfect. Indeed, it does not match the instance-dependent lower bound for all environments, which can be seen from an application of Pinsker's inequality:

$$D_{\text{KL}}(P_a \parallel P_\star) \geq \frac{\Delta_a^2}{2}.$$

Furthermore, as previously mentioned, the worst-case upper bound on its cumulative regret is loose by poly-logarithmic factors involving the horizon. Thus, significant effort has been placed on refining the optimistic estimates to achieve these goals, which we will summarise in an approximately chronological order.

Minimax Optimality. [Audibert and Bubeck \(2009\)](#) developed MOSS to fix the latter of these issues by focusing on the instance-independent cumulative regret. MOSS uses refined upper confidence bounds that are designed for the purpose of minimising the instance-independent

regret:

$$\text{MOSS}_t(a) = \hat{\mu}_a(T_a(t)) + \frac{\sqrt{\max\left\{0, \ln\left(\frac{1}{k} \cdot \frac{n}{T_a(t)}\right)\right\}}}{\sqrt{T_a(t)}}$$

where the numerator of the exploration bonus monotonically decreases as the agent approaches the known horizon. Ultimately, these upper confidence bounds allowed the authors to show that:

$$\sup_{\nu \in \mathcal{E}_{[0,1]}} R_n(\text{MOSS}, \nu) \leq 49\sqrt{nk}$$

which matches the worst-case lower bound up to the absolute constant. This algorithm requires advanced knowledge of the horizon to compute the upper confidence bounds. This can be relaxed using the doubling trick. However, the doubling trick introduces a logarithmic factor into the regret guarantee, and the resulting algorithm will no longer match the minimax lower bound. [Degenne and Perchet \(2016\)](#) provided a significantly more refined approach that does not require knowledge of the horizon. The modification made to the confidence sequence allows the algorithm to retain the minimax status, albeit with the 49 being replaced with 113.

Instance-Dependent Optimality. [Maillard et al. \(2011\)](#) and [Garivier and Cappé \(2011\)](#) concurrently and independently focussed on reconciling the upper confidence bound strategy with the asymptotic instance-dependent bound found in Equation (2.9). That is, devising a variant of the upper confidence bound strategy that features the relative entropy in the denominator of the instance-dependent bound instead of the sub-optimality gaps. [Maillard et al. \(2011\)](#) studied settings with finite support and provided a specific analysis for Bernoulli bandits. [Garivier and Cappé \(2011\)](#) focussed on environments with bounded support and distributions belonging to the one-parameter exponential family. These authors later collaborated to generalise and improve upon their original results and named the algorithm KL-UCB ([Cappé et al., 2013](#)). For Bernoulli rewards, the upper confidence bounds constructed by this algorithm have the following form:

$$\text{KL-UCB}_t(a) = \max_{\tilde{\mu}} \left\{ D_{\text{KL}}(\hat{\mu}_a(T_a(t)) \parallel \tilde{\mu}) \leq \frac{\ln(t) + 3 \ln \ln(t)}{T_a(t)} \right\}.$$

Similar upper confidence bounds can be derived for other reward distributions of interest ([Cappé et al., 2013](#)). This particular choice of upper confidence bounds yields the following instance-dependent guarantee:

$$R_n(\text{KL-UCB}, \nu) \leq \sum_{a \neq \star} \frac{\ln(n)}{D_{\text{KL}}(P_a \parallel P_\star)} + \sum_{a \neq \star} C_a$$

where C_a contains problem-dependent constants. Computing the upper confidence bounds for this algorithm requires solving an optimisation problem for each action in every round. Therefore, the implementation of KL-UCB often has a longer running time than UCB1 or MOSS, whose upper confidence bounds are readily available in closed-form. However, the optimisation problem can be solved to an arbitrary level of precision using Newton’s method or any other optimiser for one-dimensional functions. Later, [Ménard and Garivier \(2017\)](#) show that an appropriate modification of this algorithm can simultaneously achieve the minimax and instance-dependent lower bounds for all parametric bandits belonging to the one-parameter exponential family.

Other Modifications. UCB1, MOSS and KL-UCB are just a handful of examples of optimistic algorithms. [Kaufmann \(2018\)](#) develop a BayesUCB, which uses quantiles of the posterior distribution to derive the upper confidence bounds instead of concentration inequalities. Optimally Confident UCB ([Lattimore, 2016](#)) and Adaptive UCB ([Lattimore, 2018](#)) retain the minimax optimality of MOSS whilst achieving near-optimal instance-dependent guarantees. However, these algorithms rarely get extended to more complex bandit problems. This is likely because the probabilistic analyses for these algorithms are already rather complicated. Thus, UCB1 of [Auer et al. \(2002a\)](#) with an appropriate problem-specific modification is the prototypical starting point when designing an algorithm for a novel bandit problem.

2.2.2 Thompson Sampling

Thompson Sampling is an alternative algorithmic idea that looks at the multi-armed bandit problem from the Bayesian perspective. [Thompson \(1933\)](#) originally proposed the idea of placing a prior distribution over the class of Bernoulli bandits with two actions and suggested playing each action in accordance with the posterior probability that it is optimal. Therefore, exploration is guaranteed by design and exploitation will occur when the posterior distribution concentrates its mass on one of the actions. Thompson manually computed these posterior probabilities by hand, and his tabulation of these values can be used to implement the policy. [Wyatt \(1997\)](#) continued along this trajectory for the class of Bernoulli bandits with two actions and obtained a closed-form solution for the posterior probability of either action being optimal. However, Wyatt states that the time to evaluate this expression grows with the horizon, without bound. Furthermore, deriving a closed-form expression for environments with more than two actions is likely impossible ([Wyatt, 1997](#)).

Fortunately, it is not necessary to explicitly compute the posterior probability that each action is optimal to implement the algorithm. [Scott \(2010\)](#) and [Granmo \(2010\)](#) remark that it is often quicker to sample an environment from the posterior distribution and select the action with the highest expected reward in this sampled environment. Indeed, utilising this sampling procedure will select each action with the desired posterior probability. By now, this is the most common way of thinking about and implementing Thompson Sampling.

Generally, implementations of Thompson Sampling assume a particular parametric form for the reward distributions. [Granmo \(2010\)](#), in this instance, advocate for selecting the appropriate conjugate prior distribution. This choice of prior is highly beneficial because it allows for the exact computation of the posterior distribution via the likelihood of the reward distributions. Otherwise, computationally expensive procedures would be required to generate samples from the posterior distribution, such as Monte-Carlo Markov Chain ([Gelman et al., 2013](#)). Algorithm 2 presents the generic pseudo-code for the Thompson Sampling policy.

Algorithm 2 Thompson Sampling (TS)

Input. Prior Q over the class of environments.
for $t = 1, 2, \dots$ **do**
 Sample $\nu_t \sim Q(\cdot \mid A_1, X_1, \dots, A_{t-1}, X_{t-1})$
 Select $A_t = \arg \max_{a \in \mathcal{A}} \mu_a(\nu_t)$
end for

For Bernoulli bandits, the parametric form of the reward distributions only depends on the expected reward of each action. Therefore, constructing the prior distribution over the possible bandit environments amounts to placing a prior on the unknown expected rewards. For Bernoulli bandits, the most natural, convenient, and theoretically justified choice is to place an independent uniform prior over the expected reward of each action ([Chapelle and Li, 2011](#); [Agrawal and Goyal, 2012](#); [Kaufmann et al., 2012](#); [Agrawal and Goyal, 2013a](#)):

$$Q_a = \text{Beta}(1, 1)$$

for all $a \in \mathcal{A}$. This corresponds to the prior belief that each action is equally likely to be the action with the highest expected reward in the true environment. The Beta distribution is the conjugate prior distribution for the Bernoulli distribution. Therefore, computing the posterior distribution is extremely simple, and it amounts to updating the parameters of the Beta distribution for each

action:

$$Q_a(t) = \text{Beta} \left(1 + \sum_{s=1}^t X_s 1\{A_s = a\}, 1 + \sum_{s=1}^t (1 - X_s) 1\{A_s = a\} \right)$$

where the first argument counts the number of successes and the second counts the number of failures for the indexed action. For Gaussian bandits with a known variance, or any other distribution belonging to the one-parameter exponential family, a conjugate prior distribution exists. Choosing this prior distribution guarantees a closed-form expression for the posterior distribution. Thompson Sampling, when implemented with a conjugate prior distribution, is both computationally and memory efficient. Typically, it can be implemented with time and space complexity similar to that of UCB1 by [Auer et al. \(2002a\)](#).

Although Thompson did not go completely unnoticed, most of the interest in his ideas has been relatively recent. Similarly to the optimistic principle, analyses of this algorithmic approach began in the asymptotic regime. [Granmo \(2010\)](#) proved that the policy converges to selecting the optimal action with a probability arbitrarily close to one in the Bernoulli bandit environment with two actions. Additionally, [May et al. \(2012\)](#) proved asymptotic results for the setting with k actions and arbitrary reward distributions, and verified that the sequence of actions chosen by Thompson Sampling will receive rewards that converge to the maximum expected reward in the limit. [Chapelle and Li \(2011\)](#) provided empirical evidence suggesting that Thompson Sampling is near-optimal and pointed to the lack of finite time guarantees as a possible reason for its limited application in practice.

[Agrawal and Goyal \(2012\)](#) provided the first finite time analysis of Thompson Sampling for multi-armed Bernoulli bandit environments. Using the Beta-Bernoulli conjugacy, as discussed earlier in this section, the authors were able to prove that the cumulative regret can be upper bounded as follows:

$$R_n(\text{TS}, \nu) \leq 104(k-1) + \left(\sum_{a \neq \star} \left(\frac{1}{\Delta_a^2} \right)^2 1152 \ln(n) + \frac{288 \ln(n) + 192k}{\Delta_a^2} + \frac{72 \ln(n)}{\Delta_a} \right)$$

Compared to UCB1, the guarantee is substantially worse, both in terms of the absolute constants and the sub-optimality gaps. Although the guarantee depends on a strong assumption on the bandit environment, the authors of the paper show how the Beta-Bernoulli conjugacy continues to be useful and theoretically justified in the broader class of bounded bandits. [Agrawal and](#)

Goyal (2012) propose the following reduction:

- Receive $X_t \sim P_{A_t}$ with $\text{supp}(P_a) \in [0, 1]$
- Generate $\tilde{X}_t \sim \text{Bernoulli}(X_t)$
- Update Posterior with $(A_t, \tilde{X}_t)$

where $\tilde{X}_t = X_t$ if the environment consists of Bernoulli distributions. Agrawal and Goyal (2012) show that using this reduction extends the scope of the guarantee presented above to the class of environments with rewards bounded in $[0, 1]$. Subsequently, Kaufmann et al. (2012) revisited Thompson Sampling for the Bernoulli bandit, and were able to improve the finite time analysis by using distribution-specific properties:

$$R_n(\text{TS}, \nu) \leq (1 + \epsilon) \cdot \sum_{a \neq \star} \frac{\ln(n) + \ln \ln(n)}{\text{D}_{\text{KL}}(P_a \| P_\star)} + C_\epsilon(\nu)$$

where $\epsilon > 0$ and $C_\epsilon(\nu)$ is a problem-dependent quantity. Omitting the additive problem-dependent constants, this result nearly matches the asymptotic lower bound of Lai and Robbins (1985) in Equation (2.9). Furthermore, the authors provide convincing empirical evidence that Thompson Sampling is superior to UCB1 in this particular class of environment. Agrawal and Goyal (2013a) refined their previous problem-dependent analysis and presented an instance-independent upper bound on the regret of Thompson Sampling in environments with support in $[0, 1]$:

$$R_n(\text{TS}, \nu) \leq \begin{cases} \sqrt{nk \ln(n)} & \text{for Beta prior} \\ \sqrt{nk \ln(k)} & \text{for Gaussian prior} \end{cases}$$

where the authors use the same trick of simulating the Bernoulli bandit environment for the Beta prior. The guarantees presented above are for the frequentist setting. However, defining the regret as an expectation with respect to the prior distribution allows one to prove Bayesian guarantees. Russo and Roy (2014) establish a connection between optimistic and Thompson sampling algorithms, which ultimately lets them apply UCB-style tricks to control the Bayesian regret.

2.2.3 Other Principles

Optimism and Thompson Sampling are the main families of policies that are used to solve multi-armed bandit problems, and they have each received significant interest due to their simplicity.

However, these are by no means the only families of policies capable of sub-linear regret in the multi-armed bandit problem. The following paragraphs highlight some alternatives that each have their own merits.

Boltzmann Exploration. Boltzmann or Softmax exploration is another approach for balancing exploration and exploitation in sequential decision-making problems that is popular amongst practitioners. This policy guarantees exploration by playing each action with probability that is proportional to an exponential function of an estimate of its quality:

$$\pi_t(a) \propto \exp\left(\eta_t \cdot \hat{Q}_{t-1}(a)\right)$$

where $\hat{Q}_{t-1}(a)$ represents the estimated quality of action a based on the information available at the start of the t -th round and η_t denotes the step-size or learning rate. [Cesa-Bianchi et al. \(2017\)](#) analyse the popular choice of estimating the quality of each action through its empirical mean. Conversely, [Bian and Jun \(2022\)](#) propose using an empirical estimate of the sub-optimality gap. This idea was originally proposed by [Maillard \(2011\)](#). These policies have the benefit of maintaining closed-form expressions for the probability of selecting each action. Having these closed-form expressions allows data analysts to use importance-weighted estimators to de-bias any downstream analyses performed on the dataset collected by the algorithm.

Bootstrap Exploration. Bootstrapping has been highlighted as one of the most important statistical ideas of the past 50 years ([Gelman and Vehtari, 2021](#)). Loosely, the bootstrap takes in an estimator and applies it to randomly resampled datasets, and the distribution of the estimator on these randomly sampled datasets can be seen as an approximation to the sampling distribution of the data ([Efron, 1979](#)). This idea has found application in the bandit literature, where the bootstrap sub-sampling procedure is used to quantify the uncertainty in the current estimate of the expected reward for each action. BESA ([Baransi et al., 2014](#)) take this approach. The algorithm works by computing the empirical mean for each action and choosing the arg max of these estimates. To make fair comparisons, it ensures that all the empirical means are based on the same number of samples. This is done by taking the number of times the least-chosen action has been selected and using that as the sample size for all the actions. Therefore, the actions that have been played more often compute their estimates on a randomly sampled subset of the rewards. BESA comes with logarithmic instance-dependent guarantees for two-armed bandit problems, and its empirical performance is impressive. GIRO ([Kveton et al., 2019](#)) is another

algorithm that borrows ideas from the bootstrap. The authors propose adding fictitious rewards into the historical sequence of rewards for the action played in each round to increase the variance of the bootstrap estimator and induce exploration.

2.3 Linear Bandits

The multi-armed bandit problem is the simplest setting that exhibits a fundamental tension between exploration and exploitation. However, algorithms for this problem setting scale poorly with the number of actions, which makes them prohibitive in scenarios with many or an infinite number of actions. Indeed, Equations (2.10) and (2.9) demonstrate that it is impossible to avoid a dependence on the number of actions in the theoretical guarantees. However, the multi-armed bandit models the environment in a manner that is devoid of structure and the algorithms were instructed to treat each action independently.

Often, the reward distributions for each action exhibit some correlation. This correlation can be exploited to avoid having to learn about the quality of each action independently. [Robbins \(1952\)](#) imagined this as a potential extension of the multi-armed bandit model back in the 1950s. Specifically, Robbins gave the example of studying the efficacy of a treatment at varying dosages, where each dosage can be thought of as the action. Naturally, there is structure between the actions in this setting, as increasing or decreasing the dosage by a small amount should lead to similarly small changes in the expected outcomes of the patient.

Generally, structure is imposed on the bandit environment by assuming that the expected reward is a function of the chosen action. This function cannot be arbitrary and additional assumptions are required, which is usually some notion of global or local smoothness ([Kleinberg et al., 2008](#); [Bubeck et al., 2008](#); [Srinivas et al., 2010](#)).

Perhaps the most natural extension of the multi-armed bandit problem is the linear bandit problem ([Abe and Long, 1999](#); [Auer, 2002](#); [Dani et al., 2008](#); [Rusmevichientong and Tsitsiklis, 2010](#); [Abbasi-Yadkori et al., 2011](#); [Flynn et al., 2023](#)). Rather than associating each action with an index, actions are now represented by d -dimensional vectors. Each round in the linear bandit problem can have a different set of available actions. Thus, in the t -th round, the set of actions

available to the agent will be represented as follows:

$$\mathcal{A}_t \subset \mathbb{R}^d$$

for all $t \in \mathbb{N}$. Generally, $\mathcal{A}_t$ is subject to additional restrictions. [Auer \(2002\)](#), for example, assume that it is finite and spans $\mathbb{R}^d$. [Abbasi-Yadkori et al. \(2011\)](#) assume that the set of actions is chosen arbitrarily in each round and contains vectors with a bounded ℓ_2 norm.

Rather than having an independent reward distribution associated with each action, the linear bandit problem introduces structure between the actions. This structure is introduced by supposing that the expected reward for any given action is an unknown linear combination of its entries ([Abe and Long, 1999](#)). Letting $A_t \in \mathcal{A}_t$ denote the action chosen in the t -th round allows us to write down its expected reward as follows:

$$\mathbb{E}[X_t | A_t] = \langle A_t, \theta_\star \rangle$$

where $\theta_\star \in \Theta \subset \mathbb{R}^d$ is the unknown parameter vector that controls the expected reward of each action. Similarly to the multi-armed bandit setting, the agent does not get to observe the expected reward directly, and instead receives noisy samples that are centred at the expected value. Suppose $A_t \in \mathcal{A}_t$ is chosen in the t -th round by the agent, then the environment will generate a random reward with the following form:

$$X_t = \langle A_t, \theta_\star \rangle + \eta_t$$

where η_t represents conditionally zero mean noise. Typically, it is assumed that the noise satisfies the following:

$$\mathbb{E}[\exp(\lambda \eta_t) | A_1, \eta_1, \dots, A_{t-1}, \eta_{t-1}, A_t] \leq \exp\left(\frac{1}{2} \lambda^2 \sigma^2\right)$$

which states that it is conditionally σ -subgaussian. Procedure 2.2 summarises the sequential interaction between the agent and the environment in the stochastic generalised linear bandit problem.

Procedure 2.2: Linear Bandits**For** $t = 1, 2, \dots, n$:

1. **Agent.** Select $A_t \in \mathcal{A}_t$
2. **Environment.** Reveal $X_t = \langle A_t, \theta_\star \rangle + \eta_t$

Formally, the objective in this setting is to compete against the fictitious agent that selects the action with the highest expected reward in each round:

$$\hat{R}_n(\pi, \theta_\star) = \sum_{t=1}^n \max_{a \in \mathcal{A}_t} \{\langle a, \theta_\star \rangle\} - \langle A_t, \theta_\star \rangle$$

which is most similar to the pseudo-regret in Equation (2.8). The main difference is that the optimal action can change between rounds. This is to account for the possibility of changing action sets. Indeed, it is unfair to ask the agent to compete against a strategy that can select unavailable actions. The pseudo-regret for the linear bandit problem is itself a random variable. Generally, the literature tends to focus on proving upper bounds that hold with arbitrarily high probability.

2.3.1 Fundamental Limits

The linear bandit problem is an extension of the multi-armed bandit problem that allows for non-vacuous guarantees in applications with infinitely many actions. This possibility has been verified through a sequence of lower bounds, which have all been proven using action sets with different shapes.

[Dani et al. \(2008\)](#) focus on the linear bandit problem where the set of actions is the same in every round and prove the following result:

$$\mathbb{E} \left[\hat{R}_n(\pi, \nu) \right] \geq \frac{1}{2304} d \sqrt{n}$$

where $\mathcal{A}_t = \mathcal{A} = \{a \in \mathbb{R}^d : a_i^2 = a_{i+1}^2 \text{ for } i \in [1, d]\}$ and $\theta_\star$ is such that $|\langle a, \theta_\star \rangle| \leq 1$. [Rusmevichientong and Tsitsiklis \(2010\)](#) also consider a static set of actions and prove lower bounds

for infinite collection of action vectors with unit norm:

$$\mathbb{E} \left[\hat{R}_n(\pi, \nu) \right] \geq \frac{6}{1000} d\sqrt{n}$$

where $\mathcal{A}_t = \mathcal{A} = \{a \in \mathbb{R}^d : \|a\|_2 = 1\}$ and $\|\theta_\star\| = d/\sqrt{n}$. Finally, [Lattimore and Szepesvári \(2020\)](#) prove lower bounds for scenarios where the set of actions is the d -dimensional hypercube:

$$\mathbb{E} \left[\hat{R}_n(\pi, \theta_\star) \right] \geq \frac{\exp(-2)}{8} d\sqrt{n}$$

where $\mathcal{A}_t = \mathcal{A} = [-1, 1]^d$ and $\theta_\star \in \{-1/\sqrt{n}, 1/\sqrt{n}\}^d$. This particular set of actions is finite. However, the number of actions is exponentially large in the dimension: $|\mathcal{A}_t| = 2^d$.

These examples of action sets and their lower bounds illustrate that it is possible to devise algorithms whose regret is independent of the number of actions.

2.3.2 The Optimistic Principle

Numerous authors have built upon and extended the optimistic principle in the linear bandit setting ([Auer, 2002](#); [Dani et al., 2008](#); [Rusmevichientong and Tsitsiklis, 2010](#); [Abbasi-Yadkori et al., 2011](#); [Fauray et al., 2020](#); [Jun et al., 2021](#); [Abeille et al., 2021](#); [Fauray et al., 2022](#); [Flynn et al., 2023](#)). This can be implemented by maintaining a set of statistically feasible parameter vectors.

Let $\mathcal{C}_t \subset \mathbb{R}^d$ denote this set, which can be constructed using all of the available information: $A_1, X_1, \dots, A_{t-1}, X_{t-1}$. Supposing $\theta_\star \in \mathcal{C}_t$ allows the agent to construct an upper bound on the expected reward of each action as follows ([Lattimore and Szepesvári, 2020](#)):

$$\langle a, \theta_\star \rangle \leq \max_{\theta \in \mathcal{C}_t} \langle a, \theta \rangle = \text{UCB}_t(a) .$$

These upper confidence bounds can be used in exactly the same way as in Section 2.2.1. Formally, the agent would just compute the upper confidence bounds for each action and select the action according to these optimistic estimates:

$$A_t = \max_{a \in \mathcal{A}_t} \text{UCB}_{t-1}(a) .$$

Devising confidence sets that contain the unknown parameter vector is much more delicate than in the multi-armed bandit setting. This is due to the dependencies that arise due to the structure in the expected reward function. These dependencies must be accounted for if the algorithm is going to be theoretically principled.

Recall that the objective is to control the pseudo-regret with high probability. Therefore, it is sufficient to construct the sequence of confidence sets such that:

$$\mathbb{P}(\exists t \leq n : \theta_\star \notin \mathcal{C}_t) \leq \delta \quad \text{or} \quad \mathbb{P}(\exists t \in \mathbb{N} : \theta_\star \notin \mathcal{C}_t) \leq \delta$$

where $\delta \in (0, 1)$ is a user-specified confidence parameter. These confidence sets can then be used to derive upper bounds on the regret that hold with probability at least $1 - \delta$.

[Auer \(2002\)](#) focus on the setting with finitely many actions and propose the LinRel and SupLinRel algorithms. SupLinRel partitions the historical information into nearly independent sets. LinRel then uses these nearly independent datasets to build the confidence sets. This additional step of pre-processing the historical data guarantees that the confidence sets of LinRel contain the unknown parameter vector with high probability. [Auer \(2002\)](#) remark that the introduction of the pre-processing step was only introduced to handle the dependencies.

Subsequently, several authors were able to remove the pre-processing step using more sophisticated probabilistic tools based on martingales ([Dani et al., 2008](#); [Rusmevichientong and Tsitsiklis, 2010](#); [Abbasi-Yadkori et al., 2011](#)). These papers centre the confidence sets around an estimator of the unknown parameter vector. Typically, this is the regularised least squares estimator:

$$\hat{\theta}_t = V_t^{-1} \left(\sum_{s=1}^t A_s X_s \right) \quad \text{where} \quad V_t = \lambda I + \sum_{s=1}^t A_s A_s^T$$

and $\lambda \geq 0$ is the penalty on the ℓ_2 norm of the parameter vector. Traditionally, V_t is called the *design matrix*. Typically, the confidence sets have an elliptical shape, which makes it useful to define the following notation to ease the exposition:

$$\|x\|_M = \sqrt{x^T M x}$$

where $x \in \mathbb{R}^d$ and $M \in \mathbb{R}^{d \times d}$. This notation allows the elliptical confidence sets, centred on the

regularised least-squares estimator, to be written as follows (Dani et al., 2008; Rusmevichientong and Tsitsiklis, 2010; Abbasi-Yadkori et al., 2011):

$$\mathcal{C}_t = \left\{ \theta \in \Theta : \|\hat{\theta}_t - \theta\|_{V_t}^2 \leq \beta_t \right\} \quad (2.13)$$

where β_t governs the tightness of the confidence sets and varies between the papers. Gaining some intuition as to why this form of confidence set is a suitable choice comes from investigating what it implies about the upper confidence bounds. Applying Lagrange multipliers to the optimisation problem that defines the upper confidence bounds gives:

$$\text{UCB}_t(a) = \max_{\theta \in \mathcal{C}_t} \langle a, \theta \rangle = \langle a, \hat{\theta}_t \rangle + \sqrt{\beta_t} \|a\|_{V_t^{-1}}$$

which resembles the form of the upper confidence bounds presented in Section 2.2. Specifically, it is an empirical estimator of the expected reward plus an *uncertainty bonus*.

Dani et al. (2008) and Rusmevichientong and Tsitsiklis (2010) develop martingale-based confidence sets, with the latter achieving the most precise sequence of confidence sets. Subsequently, Abbasi-Yadkori et al. (2011) introduced another martingale technique that produces even tighter confidence sets, with a radius defined as:

$$\sqrt{\beta_t} = \sqrt{\lambda} \|\theta_\star\|_2 + \sqrt{d \ln \left(\frac{1 + t/\lambda}{\delta} \right)}.$$

when η_t is 1-subgaussian and the sequence of action sets all contain actions with an ℓ_2 norm upper bounded by 1. The method used to refine the confidence sets leverages results from the literature on self-normalised bounds for vector-valued martingales (de la Peña et al., 2004). Ultimately, the OFUL algorithm, which was introduced by Abbasi-Yadkori et al. (2011), achieves nearly minimax optimal guarantees on the cumulative regret:

$$\hat{R}_n(\text{OFUL}, \theta_\star) \leq \sqrt{8dn\beta_n \ln \left(\frac{d\lambda + n}{d\lambda} \right)}$$

which holds with probability at least $1 - \delta$. Recently, there has been further progress in the derivation of confidence sequences for linear bandits (Flynn et al., 2023). Reducing the radius of the confidence sets leads to improved guarantees and superior empirical performance. Indeed, Flynn et al. (2023) show that this is the case theoretically and experimentally.

2.3.3 Thompson Sampling

Section 2.2.2 introduced the Thompson Sampling algorithm for the multi-armed bandit problem. This algorithm requires a prior distribution, a way of sampling from the posterior distribution, and a way of finding the greedy action in the sampled environment. Thus, the algorithmic principle extends easily to the setting of linear bandits.

For Gaussian noise with $\sigma = 1$, the most natural choice of prior distribution is a multivariate Gaussian over the unknown parameter vector: $\mathcal{N}(0, I)$. This is the conjugate prior distribution for the Gaussian noise model and leads to closed-form solutions for the posterior distribution. Notably, the posterior distribution is also a Gaussian, and the decision-making process of the agent reduces to the following:

- Sample $\theta_t \sim \mathcal{N}(\hat{\theta}_{t-1}, r V_{t-1}^{-1})$.
- Select $A_t = \arg \max_{a \in \mathcal{A}_t} \langle a, \theta_t \rangle$

where

$$\hat{\theta}_t = V_t^{-1} \left(\sum_{s=1}^t A_s Y_s \right) \quad \text{and} \quad V_t = I + \sum_{s=1}^t A_s A_s^T$$

and $r = 1$ corresponds to the true posterior variance. This highlights how easy it can be to extend Thompson Sampling. However, the same cannot be said for the analysis in the frequentist setting.

Several authors analyse a variant of the algorithm ([Agrawal and Goyal, 2013b](#); [Abeille and Lazaric, 2017](#); [Xu et al., 2023](#)). These theoretical analyses require inflating the posterior variance by a factor of $r \approx d \ln(t/\delta)$ in each round. Intuitively, this allows the authors to argue that the algorithm is optimistic with sufficiently large probability. This modification to the posterior distribution gives the following guarantees:

$$R_n(\text{LinTS}, \theta_\star) \lesssim d^{3/2} \sqrt{n}$$

which is loose by a factor of $d^{1/2}$ when compared to lower bounds. Indeed, this is to be expected, as inflating the posterior variance introduces excessive exploration. Furthermore, adjusting the posterior distribution in this manner is not faithful to the original description of the algorithmic principle ([Thompson, 1933](#)).

2.3.4 Beyond Linearity

Often, linearity is too strong of an assumption to faithfully model the expected reward function. Consequently, there has been significant effort in extending the linear bandit framework to more expressive models for the expected reward function (Langford and Zhang, 2007; Bubeck et al., 2008; Srinivas et al., 2010; Valko et al., 2013). Of particular relevance to this thesis is the generalised linear bandit problem (Filippi et al., 2010b). Here, it is assumed that the expected rewards follow a generalised linear model. This formulation retains many of the theoretical advantages of linear models, such as the ability to employ well-understood optimisation algorithms for parameter estimation and the ability to derive well-calibrated confidence sets. This problem setting considers expected reward functions that involve a known function that links the linear predictor to the expected rewards:

$$\mathbb{E}[X_t | A_t = a] = \mu(\langle a, \theta_\star \rangle)$$

where $\mu : \mathbb{R} \rightarrow \mathbb{R}$ is known and monotonically increasing. Typically, μ is derived by assuming a specific parametric form for the likelihood of the random rewards. Filippi et al. (2010b) focus on the general case where the reward distribution belongs to the exponential family. This paper develops an optimistic algorithm and shows that the impact of introducing the link function, μ , impacts the performance of the algorithm in the worst-case. Li et al. (2017) take a similar approach and focus on cases where the number of actions is finite. Each action in each round is associated with a feature vector and these feature vectors are assumed to be drawn from distribution at the start of each round. Jun et al. (2017) focuses on retaining the theoretical guarantees and reducing computational complexity. The best-studied special case of generalised linear bandits, other than linear bandits, is when the rewards are Bernoulli random variables. This correspond to μ being the sigmoid function:

$$\mu(z) = \frac{1}{1 + \exp(-z)}$$

Often, this problem is referred to as the logistic bandit problem (Faury et al., 2020; Abeille et al., 2021; Jun et al., 2021; Faury et al., 2022). Similarly to the multi-armed bandit setting, algorithms can benefit from noise reduction when the expected reward tends to 0 or 1. Faury et al. (2020) utilise this fact to build tighter confidence sets for this specific instantiation of the generalised linear bandit problem.

Chapter 3

Reinforcement Learning

Chapter 2.1 introduces the multi-armed and linear varieties of the bandit problem, which are foundational models for sequential decision-making. These models are versatile enough to represent a wide range of real-world scenarios. However, they explicitly assume that the only consequence of selecting an action is an immediate reward drawn from a fixed but unknown distribution. This assumption is often justified in contexts where actions have an immediate and isolated effect, such as in clinical trials where each round corresponds to a new patient. Here, the agent does not need to consider long-term consequences, as it is reasonable to assume that prescribing a treatment to one patient does not influence its efficacy for another patient from the same population.

While bandit problems are well-suited to scenarios where actions have immediate and isolated consequences, many real-world problems require longer-term planning. Consider, for example, treating an individual with a chronic illness. Often, treating such illnesses involves a sequence of pharmaceutical or medical interventions over time. Each decision not only affects the patient's immediate condition but also influences their future health and treatment options. This motivates the need for a different model of the environment that captures the dependence between the immediate and long-term consequences of taking an action.

Disclaimer. The notation developed here may conflict with the notation developed in the previous chapter. This is to align with the notations that are widely adopted within literature associated with the topics of bandits and reinforcement learning. However, it will be explicitly clear from the context of all subsequent chapters which of the notations are being adopted.

3.1 Problem Setting

Reinforcement Learning considers tasks where taking an action influences the environment. This subfield of machine learning has a long history and has been of interest to the control theory (Bertsekas, 2012), operational research (Puterman, 1994) and computer science (Sutton and Barto, 2020) communities. Further, each of these fields tackle the problem from different perspectives. This makes providing a precise chronology of developments quite challenging. Therefore, this chapter focuses on providing an introduction to the problem setting and the tools that have been developed by these communities.

Generally, regardless of the research community, the environment is modelled as a Markov Decision Process (Bellman, 1957). This mathematical model introduces the notion of a state variable that evolves as a stochastic process in a manner that depends not only on the current state but also on the action chosen by the agent. Predominantly, the literature focuses on *continuing tasks* and *episodic tasks* (Sutton and Barto, 2020). This thesis focuses on episodic tasks, which use the Episodic Finite Horizon Markov Decision Process or EFH-MDP, as the mathematical model for the environment.

3.1.1 The Environment

Formally, an Episodic Finite Horizon Markov Decision Process is defined as a tuple that consists of five ingredients:

$$M = (\mathcal{S}, \mathcal{A}, H, P, r) .$$

where $\mathcal{S} = \{1, 2, \dots, S\}$ and $\mathcal{A} = \{1, 2, \dots, A\}$ represent the set of states and the set of actions available to the agent in the environment. The H denotes the episode length and can be thought of as quantifying how far ahead the agent must plan. The penultimate entry in the tuple contains the transition functions:

$$P = \{P_h(\cdot | s, a)\}_{(s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]}$$

where $P_h(\cdot | s, a)$ denotes the distribution over the next state. The transition functions characterise how the state of the environment randomly evolves throughout an episode. This random evolution depends on the current state, the chosen action, and the current stage of the interaction. Thus, the rewards that the agent can hope to receive in future stages of the episode explicitly depend on the action chosen at the current stage. The last entry contains the reward

functions:

$$r = \{r_h(s, a)\}_{(s,a,h) \in \mathcal{S} \times \mathcal{A}}$$

where $r_h(s, a) \in [0, 1]$ represents the immediate reward for taking action a in state s at the h -th stage of the episode. Often, for theoretical convenience, it is assumed that the reward function is known. This is because the main challenge in reinforcement learning lies in estimating the transition functions. Intuitively, this is because it has a higher dimension than the reward function.

Procedure 3.1: Episodic Reinforcement Learning

For $k = 1, 2, \dots, K$:

- **Environment.** Reveals $S_1^k \in \mathcal{S}$
- **For** $h = 1, 2, \dots, H$:
 - **Agent.** Selects $A_h^k \in \mathcal{A}$
 - **Agent.** Collects $R_h^k = r_h(S_h^k, A_h^k)$
 - **Environment.** Reveals $S_{h+1}^k \sim P_h(\cdot | S_h^k, A_h^k)$

Procedure 3.1 brings the above notation together and summarises how an agent interacts with the environment in an EFH-MDP. Briefly, the interaction between the agent and the environment occurs in episodes that are each made up of stages. The episode and stage are denoted by k and h , respectively. Each episode begins with the environment revealing an initial state to the agent. Subsequently, the agent will select an action, collect an immediate reward, and transition to a random next state drawn from the transition function. After H repetitions of this process, the episode terminates.

3.1.2 The Value Functions

The objective of the agent is similar to that of the multi-armed bandit problem: select the sequence of actions that maximise the expected sum of rewards within each episode. Once again, the agent will select their actions using a policy:

$$\pi^k = \{\pi_1^k, \pi_2^k, \dots, \pi_H^k\}$$

where π_h^k characterises how the agent will select their action in the h -th stage of the k -th episode. The policy can depend on everything the agent has seen so far in the episode, and it can be a distribution over the actions. However, this turns out to be unnecessary, as there is always an optimal policy that is deterministic and depends only on the current state and stage of the episode (Puterman, 1994). Therefore, the remainder of this thesis will only consider the class of policies that are deterministic mappings from each state-stage pair to an action:

$$\pi_h^k : \mathcal{S} \rightarrow \mathcal{A}$$

for all $(s, h, k) \in \mathcal{S} \times [H] \times \mathbb{N}$ where $[H] = \{1, 2, \dots, H\}$ denotes the set of stages.

Generally, the expected sum of rewards accumulated when selecting the actions from an arbitrary policy, $\pi = \{\pi_1, \dots, \pi_H\}$, in an episode is denoted by:

$$V_1^\pi(s) = \mathbb{E}^\pi \left[\sum_{h=1}^H r(S_h, A_h) \mid S_1 = s \right]. \quad (3.1)$$

The $\mathbb{E}^\pi$ notation is to be understood as an expectation with respect to states drawn from the unknown transition functions and actions being chosen by the policy: $A_h = \pi_h(S_h)$ for all $h \in [H]$. Equation (3.1) need not start the summation from $h = 1$. Indeed, it is useful to define the expected reward of following the policy from the h -th stage onwards. This is known as the *state-value function* or *value function*:

$$V_h^\pi(s) = \mathbb{E}^\pi \left[\sum_{t=h}^H r_t(S_t, A_t) \mid S_h = s \right]. \quad (3.2)$$

Another, useful quantity is the *action-value function* or *Q-function*:

$$Q_h^\pi(s, a) = \mathbb{E}^\pi \left[\sum_{t=h}^H r_t(S_t, A_t) \mid S_h = s, A_h = a \right]. \quad (3.3)$$

This function can be used by the agent to evaluate the quality of each action in every state of the environment at the h -th stage of the episode, as long as they follow their policy in all remaining stages.

The state-value and action-value functions are related. Notably, the state-value function can be

rewritten in terms of the action-value function as follows:

$$V_h^\pi(s) = Q_h^\pi(s, \pi_h(s)) \quad (3.4)$$

which follows from considering deterministic policies that depend only on the current state-stage pair. Furthermore, the action value function can be written in terms of the state value function:

$$Q_h^\pi(s, a) = r(s, a) + \sum_{s' \in \mathcal{S}} P_h(s' | s, a) V_{h+1}^\pi(s') \quad (3.5)$$

where $V_{H+1}^\pi(s) = 0$ for all $s \in \mathcal{S}$. This equality follows from an application of the tower rule, the definition of the state value function, and the fact that the next state is sampled randomly from the transition function. Together, Equations (3.4) and (3.5) are known as the Bellman Equations, which are named after Richard Bellman.

Naturally, an optimal policy is defined as the mapping from state-stage pairs to actions that maximises the state value function from any state and any stage. This means that following the optimal policy from any state at any stage will maximise the expected sum of rewards that is attainable in the remainder of the episode. Formally, the value obtained by the optimal policy will be written as follows:

$$V_h^*(s) = \max_{\pi} V_h^\pi(s) \quad \text{for all } (s, h) \in \mathcal{S} \times [H]$$

The optimal policy is not necessarily unique (Puterman, 1994). This non-uniqueness can arise for several reasons. The most obvious example is the environment possessing a redundant state-stage pair where taking any action yields the same immediate reward and has the same transition function. Trivially, all actions are equally good in this scenario and any of them can be chosen without decreasing the state value function. Thus, in this example, it is possible to define several optimal policies by having each pick a different action in this redundant state-stage pair. Although the argmax of the value function need not be unique, the value attained by each argmax is identical and cannot be improved (Puterman, 1994). Therefore, any policy that attains the maximal expected reward is called optimal and will be denoted by π^* .

3.1.3 Dynamic Programming

Equation (3.2) is used to measure the quality of the agent in the environment and the objective is to find an optimal policy. Naively, one could take a brute-force approach and calculate the expected sum of rewards for all possible sequences of state-action-stage triples. However, this quickly becomes computationally intractable. Richard Bellman showed that it is possible to do better than brute force by breaking down the optimisation problem into smaller pieces. His work showed that the optimal policy has a recursive definition that is similar to the previous subsection. These are now known as the *Bellman Optimality Equations*:

$$Q_h^*(s, a) = r_h(s, a) + \sum_{s'} P_h(s' | s, a) V_{h+1}^*(s')$$

and

$$V_h^*(s) = \max_{a \in \mathcal{A}} Q_h(s, a).$$

Therefore, optimal policies are sequences of mappings that are greedy with respect to the action value function. This representation can be used to compute the optimal policy by recursing from the last stage backwards. This technique is known as *dynamic programming* (Bertsekas, 2012).

Recall that H denotes the last stage of an episode and the interaction terminates immediately after a decision at this stage. Therefore, we can set the value function after the final stage equal to zero:

$$V_{H+1}^*(s) = 0$$

for all $s \in \mathcal{S}$ because only the first H stages of decisions matter for the value function. Consequently, achieving the maximum expected reward from the last stage of an episode is as straightforward as selecting the action with the highest expected reward:

$$Q_H^*(s, a) = r_H(s, a) \implies V_H^*(s) = \max_{a \in \mathcal{S}} r_H(s, a)$$

Computing $Q_H^*(s, a)$ and $V_H^*(s)$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$ is straightforward. Furthermore, the Bellman Optimality Equations give a recursive relationship that make these quantities useful for computing the state and action value functions at stage $H - 1$. Generally, the optimality equations say that the solution at stage $h + 1$ can be used to find the solution at the h -th stage. Thus, computing the sequence of optimal state and action value functions can be performed

backwards:

$$(Q_H^*(s, a), V_H^*(s)) \rightarrow (Q_{H-1}^*(s, a), V_{H-1}^*(s)) \rightarrow \cdots \rightarrow (Q_1^*(s, a), V_1^*(s)).$$

Once these sequence of state and action values have been computed, the Bellman Optimality Equations can be used to derive the optimal policy:

$$\pi_h(s) = \arg \max_{a \in \mathcal{A}} Q_h^*(s, a)$$

which is deterministic and depends only on the current state and stage. This observation justifies considering this restricted class of policies.

3.1.4 Reinforcement Learning

The previous section presents dynamic programming as a computationally efficient technique for finding an optimal policy in known EFH-MDPs. However, *reinforcement learning* considers the situation where the environment is unknown and can only be learnt through online interaction. Therefore, dynamic programming cannot be used out-of-the-box because it requires knowledge of the transition functions.

Similarly to Chapter 2, early work in reinforcement learning theory focuses on asymptotic guarantees for various algorithms (Sutton, 1988; Jaakkola et al., 1994; Dayan and Sejnowski, 1994). These algorithms are designed to *learn through experience* and provably find an optimal policy for the unknown environment, eventually. Subsequently, an emphasis was placed on finite-time guarantees in the late 1990s and early 2000s (Kearns and Singh, 1998; Kearns et al., 1999; Brafman and Tennenholtz, 2003; Kakade, 2003). Informally, these guarantees ask how many transitions need to be observed for each state-action pair to learn a nearly optimal policy with high probability. These kinds of guarantees continue to be of interest today (Strehl and Littman, 2005; Strehl et al., 2006; Szita and Szepesvári, 2010; Lattimore and Hutter, 2012; Dann and Brunskill, 2015; Jiang et al., 2017; Dann et al., 2018, 2019; Ménard et al., 2021). Typically, algorithms designed for this objective assume that exploration is free. Indeed, the measure of performance only asks that the policy computed using the sampled transitions is nearly optimal.

The sample complexity guarantees provide insights into the number of interactions required for

learning near-optimal policies. However, they do not necessarily capture the cumulative cost of suboptimal decisions during learning. To address this, researchers turned to regret minimisation as a measure of performance ([Auer and Ortner, 2006](#); [Jaksch et al., 2010](#); [Bartlett and Tewari, 2012](#)). This measure of performance is of primary interest in this thesis. For EFH-MDPs, the cumulative regret is defined as follows:

$$\hat{R}_K(\pi, M) = \sum_{k=1}^K V_1^*(S_1^k) - V_1^k(S_1^k) . \quad (3.6)$$

where $\pi = \{\pi^1, \pi^2, \dots, \pi^K\}$ is the sequence of policies produced by the agent and M is the unknown EFH-MDP. Equation (3.6) is a random variable because the sequence of policies depend on the random observations from the previous episodes. Again, this bears similarity to the pseudo-regret in the multi-armed bandit problem and the measure of performance in the linear bandit setting.

Fundamentally, the sequence of policies must be computed to balance exploration and exploitation. Even if the reward functions are known, the transition functions are not, and the agent only gets to observe information about the transition function associated with the chosen action. Carefully balancing exploration and exploitation was also required in the bandit problems of Chapter 2. Indeed, the algorithmic principles introduced for solving bandit problems continue to be useful in the setting of reinforcement learning. However, on top of balancing exploration and exploitation, the agent must incorporate planning into their decision-making. This element of planning is required for finding an optimal policy and for efficient exploration. [Domingues et al. \(2021\)](#) present a minimax lower bound for the cumulative regret that characterises the price for the additional complexity introduced by modelling the environment as an EFH-MDP. Formally, there exists an environment where the cumulative regret is at least:

$$\mathbb{E}[R_K(\pi, M)] \geq \frac{1}{48\sqrt{6}} H^{3/2} \sqrt{SAK} . \quad (3.7)$$

for $K \geq HSA$. Setting $S = H = 1$ recovers the lower bound for the multi-armed bandit problem up to the numeric constants ([Auer et al., 2002b](#)). Thus, there is a price to pay for increasing the planning horizon H . Recently, [Zhang et al. \(2021\)](#) consider environments where $P_h = P$ for all $h \in [H]$ and the reward functions are scaled such that $\sum_h r_h(s_h, a_h) \leq 1$ and show that it is possible to remove the dependence on the planning horizon. This does not contradict the lower

bounds of Domingues et al. (2021) as this was proven for the more general case where $P_h \neq P_{h'}$ for $h \neq h'$.

3.2 Algorithmic Principles

Equation (3.7) establishes a fundamental lower bound on what can be expected of algorithms for solving unknown EFH-MDPs. The challenge is to design algorithms that achieve cumulative regret as close as possible to these limits. Remarkably, the algorithmic principles from Chapter 2 can be extended to this more complex model of the environment. However, the number of rigorous bounds on the cumulative regret is limited in comparison to the bandit setting. Several works have proven guarantees for an extension of Thompson Sampling (Osband et al., 2013; Osband and Roy, 2017; Agrawal and Jia, 2017). Predominantly, theoretical guarantees have been proven for algorithms that build on the optimistic principle (Jaksch et al., 2010; Bartlett and Tewari, 2012; Filippi et al., 2010a; Azar et al., 2017; Dann et al., 2018; Jin et al., 2018; Fruit et al., 2020; Zhang et al., 2021).

There are several ways to implement the optimistic principle in EFH-MDPs. Broadly, there are two main categories: *model-based* and *model-free*. The model-based approaches explicitly estimate the transition functions and maintain uncertainty around these estimates to compute an optimal policy for an optimistic version of the environment. The model-free approaches do not estimate the transition functions and instead maintain an upper bound on the action-value function.

Before overviewing the literature that takes each of these approaches, it is worth noting that there is no clear reason to prefer model-based algorithms over model-free algorithms or vice versa. However, there are problem-specific reasons where preferences might arise. For example, model-free approaches require storing less information and have the benefit of being faster to compute (Jin et al., 2018). This can be advantageous in applications where there are a large number of states and actions, or there are many stages in each episode. The model-based approaches have the advantage of maintaining a model of the environment. This is why their memory footprint is larger than the model-free approaches by a multiplicative factor equal to the number of states. However, explicitly maintaining an estimate of the environment has the benefit of re-usability. For example, dynamic programming can be used to compute high-quality policies for any reward

function on the estimated transition functions. Thus, model-based approaches might be preferred when it is of interest to optimise the decision-making for many tasks in the same environment or if the reward function is expected to change between episodes. This is often the case in robotics, where specifying the reward function is a trial-and-error process (Amodei et al., 2016). This thesis predominantly builds on and compares to model-based optimistic algorithms. Therefore, the remainder of this section focuses on the class of model-based algorithms.

3.2.1 Model-Based Optimism

The model-based version of optimism is the best understood in terms of cumulative regret minimisation in reinforcement learning. There are two main ways of implementing this paradigm when designing a model-based reinforcement learning algorithm: the *model-optimistic* approach and the *value-optimistic* approach. Each of these mechanisms maintain an empirical estimate of the transition functions. For this, it is helpful to define the following counters:

$$N_{k,h}(s, a, s') = \sum_{j=1}^k 1\{S_h^j = s, A_h^k = a, S_{h+1}^j = s'\}$$

$$N_{k,h}(s, a) = \sum_{s' \in \mathcal{S}} N_{k,h}(s, a, s')$$

which count the number of transitions to state s' from s, a, h and the number of visits to s, a, h by the end of the k -th episode. Thus, an empirical estimate of the transition function is given by the ratio of these counters:

$$\hat{P}_{k,h}(s' | s, a) = \frac{N_{k,h}(s, a, s')}{N_{k,h}(s, a)}. \quad (3.8)$$

Although both model-optimistic and value-optimistic approaches construct point estimates of the transition function, they differ in how they use the estimate to compute the policy. The proceeding paragraphs discuss the model-optimistic and value-optimistic approaches of implementing the principle of optimism.

Model-Optimistic Algorithms. There are many examples of model-optimistic algorithms that are designed for regret minimisation (Jaksch et al., 2010; Filippi et al., 2010a; Bartlett and Tewari, 2012; Talebi and Maillard, 2018; Fruit et al., 2020). These algorithms all work by defining a set of statistically plausible MDPs. This is done by constructing a confidence set around the estimated transition functions of Equation (3.8). Generally, these confidence sets

have the following form:

$$\mathcal{P}_{k,h}(s, a) = \left\{ \tilde{P}_h(\cdot | s, a) \in \mathcal{P} : D\left(\tilde{P}_h(\cdot | s, a), \hat{P}_{k,h}(\cdot | s, a)\right) \leq \epsilon_{k,h}(s, a) \right\} \quad (3.9)$$

where $\mathcal{P}$ denotes the set of valid probability distributions, D is a divergence function that measures the similarity between distributions, and $\epsilon_{k,h}(s, a)$ depends on the information gathered in the previous episodes. Crucially, $\epsilon_{k,h}(s, a)$ is chosen so that the probability of the sequence of confidence sets failing to contain the true transition functions is arbitrarily small:

$$F_k = \left\{ \exists s, a, h : D\left(\hat{P}_{k,h}(\cdot | s, a), P_h(\cdot | s, a)\right) \geq \epsilon_{k,h}(s, a) \right\} \quad (3.10)$$

with $\mathbb{P}(\cup_{k=1}^{\infty} F_k) \leq \delta$ where $\delta \in (0, 1)$. With such a result, it is guaranteed that the true transition functions are within the confidence sets with probability at least $1 - \delta$. Similarly to the multi-armed and linear bandit problems, the general idea is to find the transition functions associated with a version of the environment that gives the highest expected sum of rewards:

$$\tilde{V}_h^{k+1}(s) = \max_{a \in \mathcal{A}} \left\{ \left(r_h(s, a) + \max_{\tilde{P}_h \in \mathcal{P}_{k,h}(s, a)} \sum_{s' \in \mathcal{S}} \tilde{P}_h(s' | s, a) \tilde{V}_{h+1}^{k+1}(s') \right)_h^+ \right\} \quad (3.11)$$

where $(x)_h^+ = \max\{H - h + 1, x\}$. Solving Equation (3.11) to find the next policy is usually done via *extended value iteration* (Puterman, 1994; Jaksch et al., 2010; Dann and Brunskill, 2015). Typically, the inner optimisation problem is solved iteratively. Essentially, each iteration involves placing as much probability mass on the states with the highest optimistic value at the next stage as the confidence sets permit.

Algorithm	Divergence D	Pseudo-Regret
UCRL2 (Jaksch et al., 2010)	$\ \hat{P}(\cdot) - P(\cdot)\ _1$	$H^2 S \sqrt{AK}$
UCRL2B (Fruit et al., 2020)	$\max_{s'} (\hat{P}(s') - P(s'))^2 / \hat{P}(s')$	$H^{3/2} \sqrt{STAK}$
KLUCRL (Filippi et al., 2010b)	$D_{\text{KL}}(\hat{P} \ P)$	$H^2 S \sqrt{AK}$

Table 3.1: Theoretical guarantees for model-optimistic episodic reinforcement learning algorithms. This table has been extracted from the results presented by Neu and Pike-Burke (2020). Here, $\Gamma = \max_{s,a,h} \sum_{s'} 1\{P_h(s' | s, a) > 0\}$ is an upper bound on the number of states that can be reached from any state-action-stage triple.

Changing D , which governs the shape of the confidence sets, has an impact on the theoretical guarantees. Table 3.1 presents a selection of guarantees for various choices of D , which has been

extracted from [Neu and Pike-Burke \(2020\)](#).

Value-Optimistic Algorithms. An alternative way of implementing the optimistic principle is to directly upper bound the state-value function ([Azar et al., 2017](#); [Dann et al., 2018](#)). Intuitively, the idea is to directly use the point estimate of the transition functions and apply it to an upper bound on the state-value function at the next stage. Thus, uncertainty from the latter stages is appropriately propagated backwards to the earlier stages via dynamic programming:

$$\tilde{V}_h^{k+1}(s) = \max_{a \in \mathcal{A}} \left\{ \left(r_h(s, a) + \sum_{s'} \hat{P}_{k,h}(s'|s, a) \tilde{V}_{h+1}^{k+1}(s') + B_{k,h}(s, a) \right)_h^+ \right\} \quad (3.12)$$

where $B_{k,h}(s, a)$ denotes the uncertainty bonus and $(x)_h^+ = \max\{H - h + 1, x\}$ clips the optimistic value function to the appropriate range. These bonuses are chosen to ensure that $\tilde{V}^k \geq V^*$ with probability at least $1 - \delta$ where $\delta \in (0, 1)$. Showing that this inequality holds involves a rather complex analysis. [Azar et al. \(2017\)](#) propose UCBVI-BF and employ the empirical Bernstein inequality to construct the appropriate bonuses. Ultimately, this yields an algorithm whose cumulative regret upper bound matches the minimax lower bound:

$$\hat{R}_K(\text{UCBVI-BF}, M) \lesssim H^{3/2} \sqrt{SAK} + H^2 S^2 A$$

where $\lesssim$ suppresses logarithmic factors and numeric constants. [Dann et al. \(2018\)](#) propose another value-optimistic algorithm whose cumulative regret is of the order: $H^2 \sqrt{SAK}$. This improves on many of the model-optimistic algorithms but is worse than UCBVI-BF by a factor of $\sqrt{H}$ in the leading order term.

3.2.2 Unifying Model-Based Optimism

The model-optimistic and value-optimistic algorithms look very different on the surface. Typically, the model-optimistic algorithms benefit from a cleaner theoretical analysis. Indeed, showing that the algorithm is optimistic amounts to showing that the transition functions belong to the confidence sets with high probability. However, they suffer from being more challenging to implement. Conversely, the value-optimistic algorithms are much easier to implement via dynamic programming but have more complex theoretical analyses.

Recently, [Neu and Pike-Burke \(2020\)](#) have proven that every model-optimistic algorithm has

an equivalent representation as a value-optimistic algorithm. Therefore, Equation (3.12) can be used to find the policy instead. The corresponding bonuses depend on the shape of the confidence sets through the conjugate of the divergence:

$$D_{\star}(\tilde{V} \mid \hat{P}, \epsilon) = \max_{P \in \mathcal{P}} \left\{ \langle \tilde{V}, P - \hat{P} \rangle \mid D(P, \hat{P}) \leq \epsilon \right\}.$$

Formally, [Neu and Pike-Burke \(2020\)](#) define the bonuses as follows:

$$B_{k,h}(s, a) \geq \max \left\{ D_{\star}(\tilde{V} \mid \hat{P}_{k,h}(\cdot \mid s, a), \epsilon), D_{\star}(-\tilde{V} \mid \hat{P}_{k,h}(\cdot \mid s, a), \epsilon) \right\}. \quad (3.13)$$

Running Equation (3.12) with these bonuses yields the same policy as performing extended value iteration. Propositions 1 and 2 of [Neu and Pike-Burke \(2020\)](#) establish the equivalence of the solutions to the optimisation problems for divergences that are jointly convex in both arguments and satisfy the following property:

$$D(\alpha P, \alpha P') = \alpha D(P, P')$$

for all $\alpha > 0$. These assumptions are full-filled for any norm and all f -divergences ([Liese and Vajda, 2006](#)).

3.2.3 Other Principles

The class of model-based optimistic algorithms constitutes the largest class of algorithms with theoretical guarantees on the regret. However, there are other provably efficient approaches for the problem setting, such as model-free optimism and Thompson Sampling.

Model-Free. Another way to implement the optimistic principle is with a model-free algorithm, such as Q-Learning ([Watkins, 1989](#)). Recently, [Jin et al. \(2018\)](#) proposed an optimistic version of Q-Learning for cumulative regret minimisation. Essentially, the idea is to maintain an optimistic estimate of the action-value function for each state-action-stage triple. [Jin et al. \(2018\)](#) carefully derive the learning rate and the uncertainty bonus for an optimistic version of the Q-Learning algorithm. Their specific choices guarantee that the regret is upper bounded by: $H^2 \sqrt{SAK}$ with high probability.

Thompson Sampling. Briefly, Thompson Sampling, often referred to as *posterior sampling* in this literature, extends very naturally to the setting of episodic reinforcement learning. Generally, the idea is exactly the same as in the multi-armed and linear bandit settings. Here, the appropriate prior distribution over transition functions is the Dirichlet distribution. Then, each episode consists of sampling the transition functions from the posterior distribution and computing the optimal policy for the sampled environment via dynamic programming. Typically, the guarantees are proven on the Bayesian Regret ([Osband et al., 2013](#); [Osband and Roy, 2017](#)). That is, taking the expectation of the cumulative regret with respect to the prior distribution. Frequentist guarantees for Thompson Sampling do exist. [Agrawal and Jia \(2017\)](#), for example, use the trick of inflating the posterior variance that was used in the bandit setting to obtain guarantees on the frequentist regret. For further details, the reader is referred to the excellent tutorial of [Russo et al. \(2018\)](#).

Part II

Delayed Feedback

Chapter 4

Delayed Feedback in Generalised Linear Bandits

This chapter is based on research published in AISTATS and was conducted under the supervision of Ciara Pike-Burke and Sarah Filippi ([Howson et al., 2023b](#)).

4.1 Motivation

Chapter 2 presents numerous provably efficient algorithms for multi-armed and linear bandit problems. The theoretical guarantees for these algorithms all assume that the learner observes the reward associated with their selected action immediately. However, this assumption is unrealistic in many real-world applications, such as online advertising, recommender systems, and adaptive clinical trials. There are many reasons why feedback could be delayed in these applications, and these reasons are best illustrated through examples.

Consider applying bandit algorithms to recommendation tasks. Each round in the bandit problem corresponds to a user arriving on the platform with their contextual information. The agent can use this information to recommend an item or content to the user. For streaming services, a reasonable reward signal that measures the quality of the action could be the time spent watching or listening to the recommended content. Here, the reward signal is not defined immediately. Meanwhile, the agent must continue recommending content to the users who arrive on the platform. For e-commerce or advertising applications, the ultimate goal is to recommend items that the customer will purchase. However, the time between recommending a product and a potential

purchase could be minutes, hours, days, or even longer (Chapelle, 2014).

The adaptive clinical trial is another example where the feedback is not defined immediately due to the time-consuming nature of obtaining medical feedback (Simon, 1977; Villar et al., 2015). This stems from the unfortunate fact that patients do not respond to their prescribed treatment immediately. However, the adaptive clinical trial seeks to assign patients to treatments as and when they enrol into the experiment. Therefore, the agent must continue to allocate patients to treatments with partial information about the outcomes of the previous patients.

Sometimes, practitioners even introduce delays by design (McInerney et al., 2018; Trella et al., 2024; Zhang et al., 2024). This is often due to the inability to support fully-sequential decision-making. Therefore, most real-world applications of bandit algorithms operate in the batched regime, where there is an element of randomness in the batch sizes. Here, the feedback is withheld from the agent until a pre-specified time in the future, which is when an update is scheduled to occur. This updating time is usually calendar-based. McInerney et al. (2018) and Trella et al. (2024), for example, update their algorithms on a daily and weekly cadence, respectively. This can be seen as a form of delayed feedback, where the delay associated with each decision depends on the time until the next update and the number of subsequent decisions that are made before the next update.

4.2 Problem Setting

The introductory paragraphs highlight a fundamental shortcoming of the canonical bandit model of sequential decision-making: immediate feedback. This assumption even fails to hold in the original motivating example of clinical trials (Simon, 1977; Villar et al., 2015). Furthermore, real-world implementations of bandit algorithms tend to update their algorithms on randomly sized batches of data (McInerney et al., 2018; Trella et al., 2024). The common theme is that the agent must continue to make decisions without observing the rewards associated with many of the previous actions. Procedure 4.1 models this real-world issue by introducing a delay between playing an action and receiving the corresponding feedback.

Procedure 4.1: Delayed Feedback in Bandit Problems

For $t = 1, 2, \dots, n$:

1. **Agent.** Receives $\mathcal{A}_t \subset \mathbb{R}^d$ containing the available actions.
2. **Agent.** Selects $A_t \in \mathcal{A}_t$ to play in the bandit environment.
3. **Environment.** Generates $\tau_t \in \mathbb{N}$ the delay in secret.
4. **Environment.** Generates $X_t \in \mathbb{R}$ the reward.
5. **Agent.** Receives $\{(s, A_s, X_s) : s + \tau_s = t\}$.

Following Procedure 2.2, the t -th round begins with the agent selecting an action from the set of available actions. Subsequently, the environment will generate a delay and then a random reward. This feedback will arrive at the end of round $t + \tau_t$. Finally, the agent receives feedback from previous rounds $s : s + \tau_s = t$.

This chapter of the thesis focuses on the impact of delays in the generalised linear bandit problem, which was introduced by [Filippi et al. \(2010b\)](#). Formally, this problem setting assumes that the conditional distribution of the rewards given the action belongs to the *exponential family* ([McCullagh and Nelder, 1989](#)). Thus, the density of the reward associated with the action chosen in the t -th round has the following parametric form:

$$\mathbb{P}(X_t | \theta_*, A_t) = \exp \left(\frac{X_t \langle A_t, \theta_* \rangle - b(\langle A_t, \theta_* \rangle)}{a(\phi)} + c(X_t, \phi) \right) \quad (4.1)$$

where $\theta^* \in \Theta \subset \mathbb{R}^d$ is an unknown parameter vector; a , b and c are known distribution-specific functions; and ϕ is a known constant that is often referred to as the *dispersion parameter*. For distributions belonging to the exponential family, it is well-known that the expected value can be found as follows:

$$\mathbb{E}[X_t | A_t] = \dot{b}(\langle A_t, \theta_* \rangle) := \mu(\langle A_t, \theta_* \rangle)$$

where $\dot{b}$ denotes the 1-st derivative of the function with respect to its argument. The second equality serves as the definition for what is commonly referred to as the *link function*.

Ultimately, the objective of the agent is to minimise their pseudo-regret. Mathematically, the pseudo-regret in the generalised linear bandit setting is defined as follows:

$$\hat{R}_n(\pi, \theta_*) = \sum_{t=1}^n \max_{a \in \mathcal{A}_t} \{\mu(\langle a, \theta_* \rangle)\} - \mu(\langle A_t, \theta_* \rangle) \quad (4.2)$$

where $\mathcal{A}_t \subset \mathbb{R}^d$ is the set of actions available to the agent in the t -th round. Therefore, each action chosen by the agent is measured against the highest possible expected reward in every round.

4.2.1 Additional Assumptions

Throughout, there will be several assumptions imposed on the sets of actions, the noise, and the link function. These assumptions, or variants thereof, are standard in the literature on linear and generalised linear bandits (Dani et al., 2008; Rusmevichientong and Tsitsiklis, 2010; Filippi et al., 2010b; Abbasi-Yadkori et al., 2011).

Assumption 4.1 (Sets of Actions). *Each $\mathcal{A}_t \subset \mathbb{R}^d$ can be chosen arbitrarily subject to the condition that:*

$$\max_{a \in \bigcup_{t=1}^{\infty} \mathcal{A}_t} \{\|a\|_2\} \leq 1.$$

Assumption 4.2 (The Environment). *The moment-generating function of the noise distribution conditional on the observed information must satisfy the following inequality:*

$$\mathbb{E}[\exp(\gamma \eta_t) | A_1, A_2, \dots, A_t, \tau_1, \tau_2, \dots, \tau_t, \eta_{s:s+\tau_s} < t] \leq \exp\left(\frac{1}{2} \gamma^2 R^2\right)$$

for all $\gamma \in \mathbb{R}$. Furthermore, the unknown parameter vector that governs the expected reward function is assumed to belong to a restricted subset of $\mathbb{R}^d$:

$$\theta_* \in \Theta := \{\theta \in \mathbb{R}^d : \|\theta\|_2 \leq m\}$$

Both R and $m > 0$ are known a-priori.

Assumption 4.3 (Link Function). *The link function $\mu : \mathbb{R} \rightarrow \mathbb{R}$ is known a-priori and satisfies the following conditions:*

1. $|\dot{\mu}(z)| \leq L_\mu$, and
2. $\kappa := \inf \{\dot{\mu}(\langle a, \theta \rangle) : (a, \theta) \in \mathcal{A} \times \Theta\} > 0$ where $\mathcal{A} := \bigcup_{t=1}^{\infty} \mathcal{A}_t$.

Assumption 4.4 (Delays). *The delays τ_t are non-negative, independent and identically distributed random variables such that:*

$$\mathbb{P}(\exists t \geq 1 : \tau_t \geq \epsilon_t(\tau)) \leq \delta$$

where $\epsilon_t(\tau) \leq \epsilon_{t+1}(\tau)$ is a non-decreasing and deterministic function of the rounds.

Assumption 4.1 is mainly for mathematical convenience. Essentially, the environment can present the agent with sets of actions in any manner of their choosing, as long as the norm of each element can be upper bounded by one. This includes the special cases of having the same set of actions in every round or a set of actions generated by an unknown distribution with support on the unit ball.

Assumption 4.2 implies that the noise distribution has light tails. This is standard in almost all of the bandit literature. However, it does preclude some distributions that belong to the exponential family, such as the Exponential and Poisson distributions. Nevertheless, making a light-tailed assumption on the noise is typical in the generalised linear bandit literature (Filippi et al., 2010b; Li et al., 2017; Jun et al., 2017). Assumptions 4.1 and 4.2 together imply that the linear predictor is bounded:

$$|\langle a, \theta_\star \rangle| \leq \|a\|_2 \|\theta_\star\|_2 \leq m.$$

Assumption 4.3 implies that the link function is L_μ -Lipschitz continuous and is monotonically increasing in its argument. Furthermore, the condition on κ can be understood as guaranteeing that it is possible to distinguish between two actions whose expected rewards are arbitrarily close.

Assumption 4.4 dictates that the delays are independent and identically distributed random variables with a finite expected value. Further, it states that there exists a tail bound that holds uniformly across the rounds. Notably, existing works in the delayed feedback literature make similar assumptions. Zhou et al. (2019) and Blanchet et al. (2020), for example, make a sub-exponential assumption on the tails of the delay distribution. Assumption 4.4 is satisfied in this case and the value of $\epsilon_t(\tau)$ can be derived from existing results (Wainwright, 2019).

4.2.2 Useful Notation

The linear structure of the environment means the tools from linear algebra are highly applicable in the theoretical analysis. Therefore, it is helpful to introduce some notation that will ease future expositions. Let $M, M_1, M_2 \in \mathbb{R}^{d \times d}$ denote arbitrary d -dimensional matrices.

- Weighted Norm: $\|x\|_M := \sqrt{x^T M x}$ for all $x \in \mathbb{R}^d$.
- Positive Definite: $M \succ 0 \iff x^T M x > 0$ for all $x \in \mathbb{R}^d \setminus \{\vec{0}\}$.
- Positive Semi-Definite: $M_1 \succeq 0 \iff x^T M x \geq 0$ for all $x \in \mathbb{R}^d$.
- Löwner Ordering: $M_1 \succeq M_2 \iff x^T (M_1 - M_2) x \geq 0$ for all $x \in \mathbb{R}^d$.

Additionally, $\lambda_i(M)$ and $\sigma_i(M)$ will denote the i -th largest eigenvalue and the i -th largest singular value of matrix M , respectively.

There are several problem-specific notations that need to be defined before presenting the algorithm and the theoretical analysis. Throughout, the number of rounds with missing feedback at the end of the t -th round will be denoted as follows:

$$G_t = \sum_{s=1}^t 1\{s + \tau_s > t\}$$

Further $\lambda > 0$ will denote a regularisation parameter and it is used to define the total, observed and missing design matrices:

$$\bar{V}_t = \lambda I + \sum_{s=1}^t A_s A_s^T \quad (4.3)$$

$$\bar{W}_t = \lambda I + \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s A_s^T \quad (4.4)$$

$$Z_t = \sum_{s=1}^t 1\{s + \tau_s > t\} A_s A_s^T, \quad (4.5)$$

respectively. Briefly, $\bar{V}_t$ is the total design matrix and contains information relating to all past choices. Whereas $\bar{W}_t$ and Z_t include information about actions with and without observed rewards, respectively. Equation (4.3), the total design matrix, can be rewritten as the sum of observed and missing design matrices:

$$\bar{V}_t = \bar{W}_t + Z_t. \quad (4.6)$$

Thus, when there are no delays, the total and observed design matrices are equivalent to each other, and the missing design matrix has all of its entries equal to zero.

4.3 Related Work

Originally, [Eick \(1988\)](#) introduced delayed feedback into the two-armed bandit problem with the specific application of clinical trials in mind. Naturally, his objective was to design a policy that maximises the survival time of each patient. Therefore, the delay serves as the reward signal for the agent. The reward signal in this setting has a special character in that it permits intermediate observations directly related to the reward. Namely, the agent is provided with the survival indicator from all previously treated patients at the end of every round. This is not the case for the problem setting considered in this chapter, where the agent only gets to observe the reward after the delay has elapsed.

Theoretical progress in the sub-field of sequential decision-making with delays stalled until the early 2000s, when interest revived in the context of online learning problems ([Weinberger and Ordentlich, 2002](#); [Mesterharm, 2005](#); [Langford and Zhang, 2007](#); [Agarwal and Duchi, 2011](#)). The online learning problem bears similarity to the bandit problem discussed in Chapter 2, with a key distinction: the agent receives full-information feedback. This means that the rewards for actions not chosen are also observed. Interestingly, this line of research reveals differences in what is theoretically achievable under varying feedback assumptions ([Mesterharm, 2005](#); [Agarwal and Duchi, 2011](#)). Generally, algorithms suffer an additive penalty on top of their non-delayed guarantees when the feedback is stochastic. Conversely, this penalty scales the non-delayed guarantees when an adversary chooses the feedback.

Research into the impact of delayed feedback in bandit problems restarted around two decades after their introduction by [Eick \(1988\)](#). [Dudik et al. \(2011\)](#) re-initiated this research direction and investigated stochastic contextual multi-armed bandit problems with a constant delay of τ rounds. They develop an elimination-style algorithm that uses all of the received feedback to remove sub-optimal policies and prove that the pseudo-regret is upper bounded by:

$$\hat{R}_n(\pi, \nu) \leq 16\sqrt{2kn \ln\left(\frac{4mn^2}{\delta}\right)} + 16\tau\sqrt{2k \ln\left(\frac{4mn^2}{\delta}\right)}$$

with probability at least $1 - \delta$. The m in this bound denotes the number of policies in the policy class. However, implementing the algorithm requires perfect knowledge of the distribution over contexts and maintaining the set of feasible policies.

Subsequently, [Joulani et al. \(2013\)](#) revisited the multi-armed bandit problem and analysed the regret of UCB1 and KL-UCB with randomly delayed feedback. For UCB1 of [Auer et al. \(2002a\)](#), they show that using only the observed feedback suffers expected regret that can be upper bounded as follows:

$$R_n(\text{UCB1}, \nu) \leq \sum_{a \neq \star} \frac{8 \ln(n)}{\Delta_a} + \sum_{a \neq \star} \Delta_a \mathbb{E}[\tau_a]$$

where $\nu \in \mathcal{E}_{[0,1]}$ and $\tau_a \stackrel{\text{i.i.d.}}{\sim} D_a$. Therefore, UCB1 suffers only an additive penalty relative to the immediate feedback setting, and this penalty is a weighted sum of the expected delays. For KL-UCB of [Cappé et al. \(2013\)](#), the analysis is somewhat more complicated. Nevertheless, the result is the same and the impact of the delays is cleanly separated from the horizon.

[Joulani et al. \(2013\)](#) also address the partial monitoring problem with stochastically delayed feedback. This framework is rich enough to model numerous sequential decision-making problems, and includes online learning and multi-armed bandits as special cases ([Lattimore and Szepesvári, 2020](#)). For this problem setting, the authors propose a black-box algorithm called QPM-D. This black-box algorithm allows any multi-armed bandit algorithm to be applied in delayed feedback settings. QPM-D works by maintaining queues of rewards for each action. Every time the environment reveals a reward for an action, it is appended to the corresponding queue. Each round consists of the chosen bandit algorithm selecting an action. When there is a reward in the queue for this action, the reward is given to the bandit algorithm, who proceeds to select another action. This process is repeated until an action is selected that has an empty queue. Once such an action has been found, this action will be played in the bandit environment in every round until a reward becomes available that can be used to update the bandit algorithm. Intuitively, QPM-D is creating a simulated non-delayed version of the environment for the bandit algorithm by maintaining the queues. Ultimately, [Joulani et al. \(2013\)](#) prove that this black-box algorithm increases the regret of the chosen algorithm by an additive term involving the expected delay:

$$R_n(\text{QPM-D}(\pi), \nu) \leq R'_n(\pi, \nu) + \sum_{a \neq \star} \Delta_a \mathbb{E}[\tau_a]$$

where $R'_n(\pi, \nu)$ is the regret of running π for n rounds in the non-delayed version of the environment and $\tau_a \stackrel{\text{i.i.d.}}{\sim} D_a$ denotes a random delay associated with playing the a -th action. Thus, QPM-D with UCB1 or KL-UCB yields similar guarantees as running the algorithm with just the observed feedback. [Mandel et al. \(2015\)](#) build on the queuing method to address two main issues: high memory requirements and repeatedly selecting the same action while waiting for feedback. They introduce carefully designed randomness to tackle these problems. Theoretically, their method maintains the same additive penalty as the original queuing algorithm ([Joulani et al., 2013](#)). However, empirical evidence shows significant performance improvements from the introduction of randomness.

[Desautels et al. \(2014\)](#) focus on both the batched and delayed version of bandit optimisation for smooth expected reward functions that are assumed to be drawn from a Gaussian Process prior. Their motivation lies in experimental settings with numerous workstations where experiments are run continuously. However, experiments that start at the same time need not finish simultaneously. Consequently, some workstations become available earlier, requiring the experimenter to choose the next configuration with partial feedback. Their approach builds on GP-UCB of [Srinivas et al. \(2010\)](#). Specifically, they adjust the width of the upper confidence bounds using the ratio of two posterior variances: one with all past information and one with only observed past information. This ratio depends on the delays and ends up scaling the regret of the standard GP-UCB algorithm.

[Zhou et al. \(2019\)](#) and [Blanchet et al. \(2020\)](#), with the latter presenting follow-up work, focus on developing an optimistic algorithm for contextual k -armed generalised linear bandit problems with delayed feedback. The contextual information is drawn from a fixed, but unknown, distribution D . This distribution is assumed to satisfy the following condition:

$$\lambda_d \left(\mathbb{E}_{\mathcal{A}_t \sim D} \left[\frac{1}{k} \sum_{a \in \mathcal{A}_t} aa^T \right] \right) > 0$$

This condition is required for an initialisation period to terminate. Once this initialisation period has finished, the authors propose selecting the action that maximises the following upper confidence bound:

$$\text{UCB}_t(a) = \mu \left(\langle a, \hat{\theta}_{t-1} \rangle \right) + \sqrt{\beta_t + G_t} \cdot \|a\|_{V_{t-1}^{-1}}.$$

Notably, the total design matrix is used to calculate the exploration bonus, which means that

the elliptical potential lemma can be applied. Ultimately, the usage of these upper confidence bounds leads to pseudo-regret that is upper bounded as follows (Blanchet et al., 2020):

$$\hat{R}_n(\pi, \theta_\star) \lesssim \sqrt{dn(\beta_n + \mathbb{E}[\tau])}$$

with probability at least $1 - \delta$. This guarantee does not cleanly separate the delay-related quantities from the leading order term and suggests that the impact of the delay increases with the horizon.

Delayed, Aggregated and Anonymous Feedback. Pike-Burke et al. (2018) revisit the multi-armed bandit problem with delays. However, they assume that the feedback is aggregated and anonymous, which means that the agent cannot associate each reward with an action. To tackle this challenge, the authors propose a phase-based elimination algorithm that is designed to avoid switching actions too frequently. Pike-Burke et al. (2018) prove that this strategy is capable of achieving: $\sqrt{kn} + k\mathbb{E}[\tau]$ when the algorithm has knowledge of the expected delay.

Delayed and Censored Feedback. There have been several papers that focus on the setting where the rewards are delayed and censored (Vernade et al., 2017, 2020). Here, the rewards are binary and the agent cannot distinguish between a reward of zero and a reward that has not yet returned. Vernade et al. (2017) show that the delay-dependent quantities necessarily scale the leading-order terms in the regret bound. Nevertheless, UCB1 and KL-UCB can be adapted to nearly match this asymptotic lower bound, providing that the cumulative density function of the delay distribution is known. Vernade et al. (2020) consider a similar problem in the linear bandit setting and prove a minimax lower bound of the same character. The authors develop an algorithm for this problem setting. However, the upper bound does not exhibit the same scaling in the delay-dependent quantities as their lower bound.

Batched Bandits. Perchet et al. (2015) introduced the related problem of batched bandits where the learner cannot leverage rewards from their decisions until a pre-defined and user-specified time in the future. This setting is common in real-world applications. The authors propose an action-elimination algorithm and pay specific attention to the batching scheme, showing that it is possible to achieve nearly minimax optimal guarantees with very few batches (Perchet et al., 2015). Requiring very few batches to obtain nearly minimax optimal guarantees has been observed in contextual generalised linear bandits too (Ren et al., 2021; Sawarni et al., 2024).

4.4 Delayed OFU-GLM

Section 4.3 seems to suggest that the penalty for delayed feedback in k -armed problems is an additive term involving the expected delay (Dudik et al., 2011; Joulani et al., 2013; Mandel et al., 2015; Pike-Burke et al., 2018). Conversely, this penalty is multiplicative for bandit problems with structure (Desautels et al., 2014; Zhou et al., 2019; Blanchet et al., 2020). The solutions for structured problems all utilise the optimistic principle and choose to increase the size of the confidence sets to account for the missing information caused by the delays. Eventually, this allows the authors to apply standard algebraic techniques to control the pseudo-regret. For generalised linear bandits, this corresponds to an application of Lemma 4.5.

However, taking this approach leads to guarantees where delay-dependent quantities scale the horizon, n . This suggests that the impact of delays never disappears in structured problems, regardless of how well the expected reward function is estimated. This contrasts with the intuition that the delays should become irrelevant as the estimator of the reward function converges. Therefore, alternative approaches are required if the optimistic principle is to recover the additive dependence that has been observed in the multi-armed setting.

Here, we propose a natural alternative to the existing approaches. Namely, constructing the confidence sets using only the observed information. Algorithm 3, Delayed OFU-GLM, provides the pseudo-code for our algorithm, which only utilises the historical information with observed feedback.

Algorithm 3 Delayed OFU-GLM

Input: $m, \kappa, a(\phi), \alpha > 0$, and $\delta \in (0, 1)$.

Initialisation: $\lambda = \alpha a(\phi)/\kappa, \bar{W}_0 = \lambda I, \theta_0 = \hat{\theta}_0 = \vec{0}$ and $\mathcal{C}_0 = \Theta$

for $t = 1, 2, \dots, n$ **do**

 Select A_t where:

$$(A_t, \tilde{\theta}_t) = \arg \max_{(a, \theta) \in \mathcal{A}_t \times \mathcal{C}_{t-1}} \mu(a^T \theta) \quad \text{and} \quad \mathcal{C}_{t-1} = \{\theta \in \Theta : \|\theta - \theta_{t-1}\|_{\bar{W}_{t-1}} \leq \sqrt{\beta_{t-1}}\}$$

 Receive $\{(s, A_s, X_s) : s + \tau_s = t\}$.

 Update $\bar{W}_t = \bar{W}_{t-1} + \sum_{s: s+\tau_s=t} A_s A_s^T$

 Update $\hat{\theta}_t$ by minimising the penalised negative log-likelihood in Equation (4.7).

 Compute θ_t by projecting $\hat{\theta}_t$ into Θ via Equation (4.10).

end for

4.4.1 Input Parameters

Algorithm 3 requires several parameters as input. Notably, none of these input parameters require knowledge of the delays and are relatively straightforward to select. However, some of them require prior knowledge of the problem instance. These problem-dependent parameters are the following:

- Firstly, $m > 0$ is an upper bound of the norm of the unknown parameter vector that features in Assumption 4.2. This input parameter is required by many algorithms for linear and generalised linear bandits (Abbasi-Yadkori et al., 2011; Jun et al., 2017; Faury et al., 2020).
- Secondly, κ is the lower bound found in Assumption 4.3. Computing appropriate values is straightforward. For $\mu(z) = z$, the derivative of the link function is constant and $\kappa = 1$. For $\mu(z) = 1/(1 + \exp(-z))$, it suffices to notice that the derivative of the link function is symmetric around zero and decreasing. Therefore $\kappa \geq \exp(-m)$ in this case.
- Thirdly, $a(\phi) > 0$ is a known function of the dispersion parameter of the reward distribution. The dispersion parameter is intrinsically linked to the variance of the rewards (McCullagh and Nelder, 1989):

$$\mathbb{V}[X_t | A_t, \theta_\star] = \dot{\mu}(\langle A_t, \theta_\star \rangle) a(\phi).$$

Therefore, an upper bound on this parameter is required by all optimistic algorithms for deriving valid confidence sets. For Bernoulli and Gaussian rewards: $a(\phi) = 1$ and $a(\phi) = R^2$, respectively.

Finally, $\alpha > 0$ and $\delta \in (0, 1)$ are the user-specified hyperparameters that correspond to the regularisation strength and the confidence level, respectively. The confidence parameter represents the probability that there exists a round where the unknown parameter vector falls outside confidence sets.

4.4.2 Estimating the Unknown Parameters

Algorithm 3 selects actions optimistically. Specifically, it maintains an estimate of the unknown parameter vector and constructs confidence sets on the space of parameter vectors. This estimator is found via maximum likelihood estimate, which is equivalent to minimising the negative

log-likelihood:

$$\begin{aligned} L_t(\theta) &= \frac{\alpha}{2} a(\phi) \|\theta\|_2^2 - a(\phi) \sum_{s=1}^t 1\{s + \tau_s \leq t\} \log(\mathbb{P}(X_s | A_s, \theta_*)) \\ &= \frac{\alpha}{2} a(\phi) \|\theta\|_2^2 - \sum_{s=1}^t 1\{s + \tau_s \leq t\} (X_s \langle A_s, \theta \rangle - b(\langle A_s, \theta \rangle)) + \text{constant}. \end{aligned} \quad (4.7)$$

where $\alpha > 0$ is the regularisation parameter that is input for Algorithm 3. Rescaling the negative log-likelihood by $a(\phi)$ does not change the minimiser but is done for mathematical convenience. Minimising Equation (4.7) can be done efficiently with gradient-based optimisation algorithms, such as gradient descent or Newton's method. Equations (4.8) and (4.9) provide the necessary quantities required by these optimisation methods:

$$\nabla L_t(\theta) = \alpha a(\phi) \theta + \sum_{s=1}^t 1\{s + \tau_s \leq t\} \mu(\langle A_s, \theta \rangle) A_s - \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s X_s \quad (4.8)$$

$$\nabla^2 L_t(\theta) = \alpha a(\phi) I + \sum_{s=1}^t 1\{s + \tau_s \leq t\} \dot{\mu}(\langle A_s, \theta \rangle) A_s A_s^T \quad (4.9)$$

where $\dot{\mu}$ denotes the first derivative of the link function with respect to its argument. Notably, $\nabla^2 L_t(\theta) \succeq 0$ whenever $\alpha a(\phi) \geq 0$. Therefore, the negative log-likelihood is convex and this implies that the maximum likelihood estimator is the unique vector satisfying the following:

$$\nabla L_t(\hat{\theta}_t) = \vec{0}$$

Sometimes $\hat{\theta}_t \notin \Theta$ and in such cases, the maximum likelihood estimator will be projected back into the set of admissible parameter vectors. This projection step is standard in the literature and involves solving the following (Filippi et al., 2010b; Faury et al., 2020):

$$\theta_t = \arg \min_{\theta \in \Theta} \left\{ \|g_t(\hat{\theta}_t) - g_t(\theta)\|_{\bar{W}_t^{-1}} \right\} \quad (4.10)$$

where

$$g_t(\theta) = \alpha a(\phi) \theta + \sum_{s=1}^t 1\{s + \tau_s \leq t\} \mu(\langle A_s, \theta \rangle) A_s. \quad (4.11)$$

Notably, Equation (4.10) projects the maximum likelihood estimator onto itself whenever it is in the admissible set of vectors. This follows from the fact that the weighted norm is strictly positive unless the input vector contains all zeros.

4.5 The Confidence Sets

Currently, $\mathcal{C}_t$ is defined as an elliptical confidence set that is centred around the projected maximum likelihood estimator and has a shape that is defined by the observed design matrix. Therefore, it depends only on the observed information. However, β_t , the width of these confidence sets, has been left unspecified.

Here, we will prove that the following confidence sets contain the unknown parameter vector with arbitrarily high probability:

$$\mathcal{C}_t = \left\{ \theta \in \Theta : \|\theta - \theta_t\|_{\bar{W}_t} \leq \sqrt{\beta_t} \right\} \quad (4.12)$$

where

$$\sqrt{\beta_t} = 2m \sqrt{\frac{\alpha a(\phi)}{\kappa}} + \frac{2R}{\kappa} \sqrt{2 \log \left(\frac{\det(\bar{W}_t)^{1/2}}{\delta \lambda^{d/2}} \right)} \quad (4.13)$$

and θ_t is the projected maximum likelihood estimator.

Theorem 4.1. *Under Assumptions 4.2 and 4.3, it holds that:*

$$\mathbb{P}(\exists t \geq 1 : \theta_\star \notin \mathcal{C}_t) \leq \delta$$

where $\mathcal{C}_t$ and β_t are defined as in Equations (4.12) and (4.13), respectively.

Proving Theorem 4.1 requires three preliminary results. Lemmas 4.1, 4.2 and 4.3. In Section 4.5.1, we will state and prove these intermediate results. Subsequently, in Section 4.5.2, we will utilise these results to prove Theorem 4.1.

4.5.1 Preliminary Results for Theorem 4.1

Lemma 4.1 is a corollary of the mean-value theorem and we will use this result to relate the form of the confidence sets to quantities that are used in the estimation procedure.

Lemma 4.1. *Let $\theta_1 \in \Theta$ and $\theta_2 \in \Theta$ be arbitrary vectors, and $\lambda = \alpha a(\phi)/\kappa$. Then, the following inequality holds:*

$$\|\theta_1 - \theta_2\|_{\bar{W}_t} \leq \frac{1}{\kappa} \|g_t(\theta_1) - g_t(\theta_2)\|_{\bar{W}_t^{-1}}$$

Proof. Assumption 4.3 guarantees that the link function is differentiable. Therefore, an applica-

tion of the mean-value theorem gives:

$$\mu(\langle A_s, \theta_1 \rangle) - \mu(\langle A_s, \theta_2 \rangle) = \dot{\mu}(z_s) (\langle A_s, \theta_1 \rangle - \langle A_s, \theta_2 \rangle)$$

where $z_s \in [\langle A_s, \theta_1 \rangle, \langle A_s, \theta_2 \rangle]$. Therefore, $g_t(\theta_1) - g_t(\theta_2)$ can be rewritten by directly previous display into the expression for the difference as follows:

$$\begin{aligned} g_t(\theta_1) - g_t(\theta_2) &= \alpha a(\phi) (\theta_1 - \theta_2) + \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s (\mu(\langle A_s, \theta_1 \rangle) - \mu(\langle A_s, \theta_2 \rangle)) \\ &= \alpha a(\phi) (\theta_1 - \theta_2) + \sum_{s=1}^t 1\{s + \tau_s \leq t\} \dot{\mu}(z_s) A_s A_s^T (\theta_1 - \theta_2) \\ &= \left(\alpha a(\phi) I + \sum_{s=1}^t 1\{s + \tau_s \leq t\} \dot{\mu}(z_s) A_s A_s^T \right) (\theta_1 - \theta_2) \\ &:= H_t \cdot (\theta_1 - \theta_2) \end{aligned}$$

where the last equality serves as the definition for the matrix H_t . Recall $z_s \in [\langle A_s, \theta_1 \rangle, \langle A_s, \theta_2 \rangle]$ can be expressed as follows:

$$z_s = p \langle A_s, \theta_1 \rangle + (1 - p) \langle A_s, \theta_2 \rangle = \langle A_s, p \theta_1 + (1 - p) \theta_2 \rangle = \langle A_s, \theta_p \rangle$$

where $\theta_p = p \theta_1 + (1 - p) \theta_2$ and $p \in [0, 1]$. Since $\theta_1 \in \Theta$ and $\theta_2 \in \Theta$, it follows that $\theta_p \in \Theta$ and this gives a lower bound on the derivative of the link function in the s -th position of the summation:

$$\dot{\mu}(z_s) = \dot{\mu}(\langle A_s, \theta_p \rangle) \geq \kappa$$

which follows from the fact that $A_s \in \mathcal{A}_s \subseteq \cup_{t=1}^{\infty} \mathcal{A}_t$ almost surely and then applying the definition of κ in Assumption 4.3. Consequently, H_t satisfies the following Löwner partial ordering:

$$\begin{aligned} H_t &= \alpha a(\phi) I + \sum_{s=1}^t 1\{s + \tau_s \leq t\} \dot{\mu}(z_s) A_s A_s^T \\ &\succeq \alpha a(\phi) I + \kappa \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s A_s^T \\ &= \kappa \left(\frac{\alpha a(\phi)}{\kappa} I + \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s A_s^T \right) \\ &= \kappa \left(\lambda I + \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s A_s^T \right) \\ &= \kappa \bar{W}_t \end{aligned}$$

where the penultimate equality follows from the definition of λ and the final equality follows from Equation (4.4). Further, the Löwner ordering reverses when the matrices are inverted, which gives:

$$H_t^{-1} \preceq \frac{1}{\kappa} \bar{W}_t^{-1}.$$

Bringing all of the above together gives the result via the following steps:

$$\begin{aligned} \|\theta_1 - \theta_2\|_{\bar{W}_t} &= \frac{1}{\sqrt{\kappa}} \|\theta_1 - \theta_2\|_{\kappa \bar{W}_t} \\ &\leq \frac{1}{\sqrt{\kappa}} \|\theta_1 - \theta_2\|_{H_t} \\ &= \frac{1}{\sqrt{\kappa}} \|H_t^{1/2} (\theta_1 - \theta_2)\|_2 \\ &= \frac{1}{\sqrt{\kappa}} \|H_t^{-1/2} (g_t(\theta_1) - g_t(\theta_2))\|_2 \\ &= \frac{1}{\sqrt{\kappa}} \|g_t(\theta_1) - g_t(\theta_2)\|_{H_t^{-1}} \\ &\leq \frac{1}{\sqrt{\kappa}} \|g_t(\theta_1) - g_t(\theta_2)\|_{\frac{1}{\kappa} \bar{W}_t^{-1}} \\ &= \frac{1}{\kappa} \|g_t(\theta_1) - g_t(\theta_2)\|_{\bar{W}_t^{-1}} \end{aligned}$$

as required. $\square$

Lemma 4.2 is the second technical result that is used in the proof of Theorem 4.1. The purpose of this result is to separate the noise-related terms from the terms that are measurable once the agent has made their decision and the delay has been generated.

Lemma 4.2. *Let $\lambda = \alpha a(\phi)/\kappa$ and $S_t = \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s \eta_s$. Then, it holds that:*

$$\frac{1}{\kappa} \|g_t(\hat{\theta}_t) - g_t(\theta^*)\|_{\bar{W}_t^{-1}} \leq \sqrt{\lambda} \|\theta^*\|_2 + \frac{1}{\kappa} \|S_t\|_{\bar{W}_t^{-1}}.$$

Proof. Recall $\hat{\theta}_t$ is the maximum likelihood estimator and satisfies the following:

$$0 = \nabla L_t(\hat{\theta}_t) = g_t(\hat{\theta}_t) - \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s X_s$$

Since $X_s = \mu(\langle A_s, \theta_* \rangle) + \eta_s$, it follows that:

$$g_t(\hat{\theta}_t) = \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s X_s$$

$$\begin{aligned}
&= \sum_{s=1}^t 1\{s + \tau_s \leq t\} \mu(\langle A_s, \theta_\star \rangle) A_s + \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s \eta_s \\
&= -\alpha a(\phi) \theta_\star + g_t(\theta_\star) + \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s \eta_s \quad (\text{Equation (4.11) with } \theta_\star) \\
&= -\alpha a(\phi) \theta_\star + g_t(\theta_\star) + S_t
\end{aligned}$$

where $S_t := \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s \eta_s$ gives the last equality. Therefore, it follows that:

$$\begin{aligned}
\frac{1}{\kappa} \|g_t(\hat{\theta}_t) - g_t(\theta_\star)\|_{\bar{W}_t^{-1}} &= \frac{1}{\kappa} \|-\alpha a(\phi) \theta_\star + S_t\|_{\bar{W}_t^{-1}} \\
&\leq \frac{1}{\kappa} \|\alpha a(\phi) \theta_\star\|_{\bar{W}_t^{-1}} + \frac{1}{\kappa} \|S_t\|_{\bar{W}_t^{-1}} \quad (\text{Triangle Inequality}) \\
&= \frac{\alpha a(\phi)}{\kappa} \|\theta_\star\|_{\bar{W}_t^{-1}} + \frac{1}{\kappa} \|S_t\|_{\bar{W}_t^{-1}} \\
&= \lambda \|\theta_\star\|_{\bar{W}_t^{-1}} + \frac{1}{\kappa} \|S_t\|_{\bar{W}_t^{-1}} \quad (\text{Since } \lambda = \alpha a(\phi)/\kappa) \\
&\leq \lambda \|\theta_\star\|_{\frac{1}{\lambda} I} + \frac{1}{\kappa} \|S_t\|_{\bar{W}_t^{-1}} \quad (\text{Since } \bar{W}_t \succeq \lambda I) \\
&= \sqrt{\lambda} \|\theta_\star\|_2 + \frac{1}{\kappa} \|S_t\|_{\bar{W}_t^{-1}}
\end{aligned}$$

as required. $\square$

Lemma 4.3 is the last technical result used in the proof of Theorem 4.1. This result is used to establish properties of the noise-related term, which allows for the application of standard concentration results for vector-valued martingales. By Fenchel Duality, the noise terms can be rewritten as follows:

$$\begin{aligned}
\frac{1}{2} \|S_t\|_{\bar{W}_t^{-1}}^2 &= \max_{x \in \mathbb{R}^d} \left\{ \langle x, S_t \rangle - \frac{1}{2} \|x\|_{\bar{W}_t}^2 \right\} \\
&= \max_{x \in \mathbb{R}^d} \left\{ \log \left(\exp \left(\langle x, S_t \rangle - \frac{1}{2} \|x\|_{\bar{W}_t}^2 \right) \right) \right\} \\
&= \log \left(\max_{x \in \mathbb{R}^d} \left\{ \exp \left(\langle x, S_t \rangle - \frac{1}{2} \|x\|_{\bar{W}_t}^2 \right) \right\} \right) \quad (4.14)
\end{aligned}$$

For Lemma 9 and Theorem 1 of [Abbasi-Yadkori et al. \(2011\)](#) to apply, it suffices to show that the following random variable is a non-negative supermartingale:

$$M_t(x) = \exp \left(\frac{1}{R} \langle x, S_t \rangle - \frac{1}{2} \|x\|_{\bar{W}_t}^2 \right) \quad (4.15)$$

where $W_t = \bar{W}_t - \lambda I = \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s A_s^T$.

However, due to the delayed feedback, we cannot rely on the standard arguments to verify this property holds. Therefore, additional care is required to construct a recursive definition that can be used to verify the properties required for $M_t(x)$.

Lemma 4.3. *Let $M_t(x)$ be defined as in Equation (4.15) and define ω to be a stopping time with respect to the filtration $\{\mathcal{F}_t\}_{t=0}^\infty$. Then, $M_\omega(x)$ is almost surely well-defined and $\mathbb{E}[M_\omega(x)] \leq 1$.*

Proof. Rewriting Equation (4.15) by expanding out the definitions of S_t and W_t reveals the following:

$$\begin{aligned}
M_t(x) &= \exp \left(\frac{1}{R} \langle x, S_t \rangle - \frac{1}{2} \|x\|_{W_t}^2 \right) \\
&= \exp \left(\sum_{s=1}^t \frac{1\{s + \tau_s \leq t\} \langle x, A_s \rangle \eta_s}{R} - \frac{1}{2} x^T \left(\sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s A_s^T \right) x \right) \\
&= \exp \left(\sum_{s=1}^t \frac{1\{s + \tau_s \leq t\} \langle x, A_s \rangle \eta_s}{R} - \frac{1}{2} \sum_{s=1}^t 1\{s + \tau_s \leq t\} \langle x, A_s \rangle^2 \right) \\
&= \exp \left(\sum_{s=1}^t \frac{1\{s + \tau_s \leq t\} \langle x, A_s \rangle \eta_s}{R} - \frac{1}{2} \sum_{s=1}^t 1\{s + \tau_s \leq t\} \langle x, A_s \rangle^2 \right) \\
&= \exp \left(\sum_{s=1}^t 1\{s + \tau_s \leq t\} \left(\frac{\langle x, A_s \rangle \eta_s}{R} - \frac{1}{2} \langle x, A_s \rangle^2 \right) \right).
\end{aligned}$$

The summation inside the exponential can be partitioned using the indicator variables. Notably, performing a case-by-case analysis reveals that:

$$1\{s + \tau_s \leq t\} = 1\{s + \tau_s \leq t - 1\} + 1\{s + \tau_s = t\}.$$

Leveraging this observation and continuing the sequence of equalities gives a recursive definition:

$$\begin{aligned}
M_t(x) &= \exp \left(\sum_{s=1}^t (1\{s + \tau_s \leq t - 1\} + 1\{s + \tau_s = t\}) \left(\frac{\langle x, A_s \rangle \eta_s}{R} - \frac{1}{2} \langle x, A_s \rangle^2 \right) \right) \\
&= M_{t-1}(x) \exp \left(\sum_{s=1}^t \frac{1\{s + \tau_s = t\} \langle x, A_s \rangle \eta_s}{R} \right) \exp \left(- \sum_{s=1}^t \frac{1\{s + \tau_s = t\} \langle x, A_s \rangle^2}{2} \right) \\
&:= M_{t-1}(x) D_t(x) \exp \left(- \sum_{s=1}^t \frac{1\{s + \tau_s = t\} \langle x, A_s \rangle^2}{2} \right)
\end{aligned}$$

where the penultimate equality follows from properties of exponents and the definition of $M_{t-1}(x)$.

The final equality serves as the definition for $D_t(x)$. Now, the natural filtration is the σ -algebra

generated by the sequence of random variables:

$$\mathcal{F}_{t-1} = \sigma(A_1, A_2, \dots, A_t, \tau_1, \tau_2, \dots, \tau_t, \eta_s \text{ for } s + \tau_s < t) .$$

Therefore, $D_t(x)$ is the only term containing quantities that are not $\mathcal{F}_{t-1}$ -measurable. Assumption 4.2 guarantees that the noise is conditionally subgaussian and this allows the expectation to be bounded:

$$\begin{aligned} \mathbb{E}[M_t(x) | \mathcal{F}_{t-1}] &= M_{t-1}(x) \mathbb{E}[D_t(x) | \mathcal{F}_{t-1}] \exp\left(-\sum_{s=1}^t \frac{1\{s + \tau_s = t\} \langle x, A_s \rangle^2}{2}\right) \\ &\leq M_{t-1}(x) \exp\left(\sum_{s=1}^t \frac{1\{s + \tau_s = t\} \langle x, A_s \rangle^2 R^2}{2R^2}\right) \exp\left(-\sum_{s=1}^t \frac{1\{s + \tau_s = t\} \langle x, A_s \rangle^2}{2}\right) \\ &= M_{t-1}(x) . \end{aligned}$$

where Assumption 4.2 guarantees that the noise variables depend only on the observable history and utilises the subgaussian property with $\gamma = 1\{s + \tau_s = t\} \langle x, A_s \rangle / R$. Therefore, $\{M_t(x)\}_{t=1}^\infty$ is a non-negative supermartingale, where the non-negativity property follows from the exponential function producing strictly positive values. The law of total expectation guarantees the following:

$$\begin{aligned} \mathbb{E}[M_t(x)] &= \mathbb{E}[\mathbb{E}[M_t(x) | \mathcal{F}_{t-1}]] \\ &\leq \mathbb{E}[M_{t-1}(x)] = \mathbb{E}[\mathbb{E}[M_{t-1}(x) | \mathcal{F}_{t-2}]] \\ &\vdots \\ &\leq \mathbb{E}[M_0(x)] \\ &= 1 \end{aligned}$$

where the final line follows from the convention that empty summations evaluate to zero. Furthermore, the convergence theorem for non-negative supermartingales:

$$M_\infty(x) = \lim_{t \rightarrow \infty} M_t(x)$$

is almost surely well-defined. Hence, $M_\omega(x)$ is almost surely well-defined, regardless of whether the stopping time ω is finite or infinite. Fatou's Lemma guarantees that:

$$\mathbb{E}[M_\omega(x)] = \mathbb{E}\left[\liminf_{t \rightarrow \infty} M_{\min\{t, \omega\}}(x)\right] \leq \liminf_{t \rightarrow \infty} \mathbb{E}[M_{\min\{t, \omega\}}(x)]$$

Combining the right-hand side of the above with the law of total expectation:

$$\begin{aligned}
\mathbb{E} [M_{\min\{t,\omega\}}(x)] &= \mathbb{E} [\mathbb{E} [M_{\min\{t,\omega\}}(x) \mid \mathcal{F}_{t-1}]] \\
&\leq \mathbb{E} [M_{\min\{t-1,\omega\}}(x)] = \mathbb{E} [\mathbb{E} [M_{\min\{t-1,\omega\}}(x) \mid \mathcal{F}_{t-1}]] \\
&\vdots \\
&\leq \mathbb{E} [M_{\min\{0,\omega\}}(x)] = \mathbb{E} [M_0(x)] = 1
\end{aligned}$$

for any $t \geq 0$. Therefore, $\mathbb{E}[M_\omega(x)] \leq 1$, as required. $\square$

4.5.2 Proof of Theorem 4.1

Section 4.5.1 presents all of the intermediate results that we will require to prove the validity of the confidence sets in Equation (4.12). Therefore, all that remains is carefully piecing these results together.

Proof. From Assumption 4.2 and Equation (4.10), $\theta_\star \in \Theta$ and $\theta_t \in \Theta$. Therefore, Lemma 4.1 guarantees the following:

$$\|\theta_\star - \theta_t\|_{\bar{W}_t} \leq \frac{1}{\kappa} \|g_t(\theta_\star) - g_t(\theta_t)\|_{\bar{W}_t^{-1}}.$$

This inequality relates the weighted norm of the unknown parameter vector and the projected maximum likelihood estimator to quantities involved in the estimation procedure. Generally, not much can be said about the projected maximum likelihood estimator. Therefore, the right-hand side of the previous display is related to the maximum likelihood estimator:

$$\begin{aligned}
\frac{1}{\kappa} \|g_t(\theta_\star) - g_t(\theta_t)\|_{\bar{W}_t^{-1}} &= \frac{1}{\kappa} \|g_t(\theta_\star) - g_t(\hat{\theta}_t) + g_t(\hat{\theta}_t) - g_t(\theta_t)\|_{\bar{W}_t^{-1}} \\
&\leq \frac{1}{\kappa} \|g_t(\theta_\star) - g_t(\hat{\theta}_t)\|_{\bar{W}_t^{-1}} + \frac{1}{\kappa} \|g_t(\hat{\theta}_t) - g_t(\theta_t)\|_{\bar{W}_t^{-1}} && \text{(Triangle Inequality)} \\
&= \frac{1}{\kappa} \|g_t(\theta_\star) - g_t(\hat{\theta}_t)\|_{\bar{W}_t^{-1}} + \frac{1}{\kappa} \min_{\theta \in \Theta} \|g_t(\hat{\theta}_t) - g_t(\theta)\|_{\bar{W}_t^{-1}} && \text{(Definition of } \theta_t) \\
&\leq \frac{1}{\kappa} \|g_t(\theta_\star) - g_t(\hat{\theta}_t)\|_{\bar{W}_t^{-1}} + \frac{1}{\kappa} \|g_t(\hat{\theta}_t) - g_t(\theta_\star)\|_{\bar{W}_t^{-1}} \\
&= \frac{2}{\kappa} \|g_t(\hat{\theta}_t) - g_t(\theta_\star)\|_{\bar{W}_t^{-1}} && (4.16)
\end{aligned}$$

where the last equality follows from the absolute homogeneity property of norms. Applying Lemma 4.2 to Equation (4.16) allows for the separation of the noise-related terms and the bias

introduced by penalising the negative log-likelihood function:

$$\begin{aligned} \frac{2}{\kappa} \|g_t(\hat{\theta}_t) - g_t(\theta_*)\|_{\bar{W}_t^{-1}} &\leq 2\sqrt{\lambda} \|\theta_*\|_2 + \frac{2}{\kappa} \|S_t\|_{\bar{W}_t^{-1}} \\ &\leq 2m\sqrt{\lambda} + \frac{2}{\kappa} \|S_t\|_{\bar{W}_t^{-1}} \end{aligned} \quad (\text{Assumption 4.2})$$

where $\lambda := \alpha a(\phi)/\kappa$ and $S_t := \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s \eta_s$ as-per the statement of Lemma 4.2.

Lemma 4.3 confirms that $\|S_t\|_{\bar{W}_t^{-1}}$ is a non-negative supermartingale under the information structure created by the delays. Furthermore, it verifies the conditions that are required to apply existing results (de la Peña et al., 2004; Abbasi-Yadkori et al., 2011). Applying Theorem 1 of Abbasi-Yadkori et al. (2011) reveals that:

$$\mathbb{P} \left(\exists t \geq 0 : \|S_t\|_{\bar{W}_t^{-1}} \geq R \sqrt{2 \log \left(\frac{\det(\bar{W}_t)^{1/2}}{\delta \lambda^{d/2}} \right)} \right) \leq \delta.$$

Bringing everything together gives:

$$\begin{aligned} \mathbb{P}(\exists t \geq 0 : \theta_* \notin \mathcal{C}_t) &= \mathbb{P}(\exists t \geq 0 : \|\theta_* - \theta_t\|_{\bar{W}_t} \geq \sqrt{\beta_t}) \\ &\leq \mathbb{P} \left(\exists t \geq 0 : 2m\sqrt{\lambda} + \frac{2}{\kappa} \|S_t\|_{\bar{W}_t^{-1}} \geq \sqrt{\beta_t} \right) \\ &= \mathbb{P} \left(\exists t \geq 0 : \|S_t\|_{\bar{W}_t^{-1}} \geq R \sqrt{2 \ln \left(\frac{\det(\bar{W}_t)^{1/2}}{\delta \lambda^{d/2}} \right)} \right) \\ &\leq \delta \end{aligned}$$

as required. □

4.6 The Delayed Elliptical Potential

Theorem 4.1 establishes that the sequence of confidence sets contain the unknown parameter vector with arbitrarily high probability. Lemma 4.4 applies this fact to provide a preliminary upper bound on the pseudo-regret of Algorithm 3.

Lemma 4.4. *Under Assumptions 4.1, 4.2 and 4.3, the pseudo-regret of Delayed OFU-GLM with any $\lambda = \alpha a(\phi)/\kappa \geq 1$ satisfies:*

$$\hat{R}_n(\pi, \theta_\star) \leq 2L_\mu \sqrt{\beta_n} \sum_{t=1}^n \|A_t\|_{W_{t-1}^{-1}}$$

with probability at least $1 - \delta$.

Proof. From Theorem 4.1, it is easy to verify that:

$$\mathbb{P}(\theta_\star \in \mathcal{C}_t \text{ for all } t \geq 0) = 1 - \mathbb{P}(\exists t \geq 0 : \theta_\star \notin \mathcal{C}_{t-1}) \geq 1 - \delta.$$

Therefore, $\theta_\star \in \mathcal{C}_t$ for all $t \geq 0$ will be used throughout the proof with the understanding that the resulting upper bound will hold with probability at least $1 - \delta$. Now, we will focus on getting an upper bound on the per-round pseudo-regret on the event that the unknown parameter lives within the confidence set.

Define the optimal action in the t -th round as follows:

$$a_t^\star = \arg \max_{a \in \mathcal{A}_t} \mu(\langle a, \theta_\star \rangle).$$

Since $\theta_\star \in \mathcal{C}_{t-1}$, it follows that the expected reward for the optimal action in the t -th round can be controlled as follows:

$$\mu(\langle a_t^\star, \theta_\star \rangle) \leq \max_{\theta \in \mathcal{C}_{t-1}} \mu(\langle a_t^\star, \theta \rangle) = \text{UCB}_{t-1}(a_t^\star)$$

Algorithm 3 selects the action with the highest upper confidence bound. Therefore, the upper confidence bound for the optimal action can be upper bounded by the upper confidence bound of the action that was selected by the agent in the t -th round:

$$\text{UCB}_{t-1}(a_t^\star) \leq \max_{a \in \mathcal{A}_t} \text{UCB}_{t-1}(a) = \text{UCB}_{t-1}(A_t).$$

Define $\tilde{\theta}_t = \arg \max_{\theta \in \mathcal{C}_{t-1}} \mu(\langle A_t, \theta \rangle)$ as the parameter vector that gives the chosen action the highest expected reward. Then, chaining the previous displays together gives an upper bound

on the per-round pseudo regret incurred by the agent in the t -th round:

$$\begin{aligned}
\mu(\langle a_t^*, \theta_* \rangle) - \mu(\langle A_t, \theta_* \rangle) &\leq \mu(\langle A_t, \tilde{\theta}_t \rangle) - \mu(\langle A_t, \theta_* \rangle) \\
&\leq L_\mu \left(\langle A_t, \tilde{\theta}_t - \theta_* \rangle \right) \\
&= L_\mu \left(\langle A_t, \tilde{\theta}_t - \theta_{t-1} + \theta_{t-1} - \theta_* \rangle \right)
\end{aligned}$$

where the penultimate line follows from Assumption 4.3. Specifically, the gradient of the link function is bounded by L_μ , which implies it is L_μ -Lipschitz by the mean-value theorem. The final line follows from adding and subtracting the projected maximum likelihood estimator. This is done because it is this vector that is used to centre the confidence sets. Choosing $\theta \in \mathcal{C}_{t-1}$ arbitrarily gives the following upper bound:

$$\begin{aligned}
|\langle A_t, \theta - \theta_{t-1} \rangle| &= \langle \bar{W}_{t-1}^{-1/2} A_t, \bar{W}_{t-1}^{1/2} (\theta - \theta_{t-1}) \rangle \\
&\leq \|\bar{W}_{t-1}^{-1/2} A_t\|_2 \|\bar{W}_{t-1}^{1/2} (\theta_{t-1} - \theta)\|_2 && \text{(Cauchy-Schwarz)} \\
&= \|A_t\|_{\bar{W}_{t-1}^{-1}} \|\theta_{t-1} - \theta\|_{\bar{W}_{t-1}} \\
&\leq \sqrt{\beta_{t-1}} \|A_t\|_{\bar{W}_{t-1}^{-1}} && \text{(Since } \theta \in \mathcal{C}_{t-1}\text{)} \\
&\leq \sqrt{\beta_n} \|A_t\|_{\bar{W}_{t-1}^{-1}}
\end{aligned}$$

where $\beta_{t-1} \leq \beta_n$ is the width of the confidence sets as-per Equation (4.13). Recall $\tilde{\theta}_t, \theta_* \in \mathcal{C}_{t-1}$, where the former is by definition and the latter is on the event that the unknown parameter vector is within the confidence set in every round. Therefore, the instantaneous regret can be upper bounded as follows:

$$\mu(\langle a_t^*, \theta_* \rangle) - \mu(\langle A_t, \theta_* \rangle) \leq 2L_\mu \sqrt{\beta_n} \|A_t\|_{\bar{W}_{t-1}^{-1}}.$$

Therefore, summing the instantaneous sub-optimality of the decisions made by the agent and using the upper bound above gives the following:

$$\begin{aligned}
\hat{R}_n(\pi, \theta_*) &= \sum_{t=1}^n \mu(\langle a_t^*, \theta_* \rangle) - \mu(\langle A_t, \theta_* \rangle) \\
&\leq 2L_\mu \sqrt{\beta_n} \sum_{t=1}^n \|A_t\|_{\bar{W}_{t-1}^{-1}}
\end{aligned} \tag{4.17}$$

as required. $\square$

Lemma 4.4 reveals that the last term that needs to be controlled is Equation (4.17). Summations of this form arise in various applications involving linear functions (Lai and Wei, 1982; Abbasi-Yadkori et al., 2011; Cesa-Bianchi and Lugosi, 2006). Thus, there is a well-known technical tool for controlling this summation, now known as the *elliptical potential lemma*. This algebraic tool dates back to at least Lai and Wei (1982). The elliptical potential lemma is presented below for ease of reference.

Lemma 4.5 (Elliptical Potential Lemma). *Let $\{A_t\}_{t=1}^\infty$ be an arbitrary sequence of d -dimensional vectors such that $\|A_t\|_2^2 \leq 1$. Define $V_0 = \lambda I$, $V_t = V_0 + \sum_{s=1}^t A_s A_s^T$. Then,*

$$\sum_{t=1}^n \|A_t\|_{V_{t-1}^{-1}}^2 \leq 2 \log \left(\frac{\det(V_n)}{\det(V_0)} \right) \leq 2d \log \left(\frac{d\lambda + n}{d\lambda} \right)$$

for $\lambda \geq 1$.

Lemma 4.5 is used to control the regret of algorithms in linear and generalised linear bandit problems without delays (Filippi et al., 2010b; Abbasi-Yadkori et al., 2011). Furthermore, this algebraic tool provides a tight upper bound on the term in question (Carpentier et al., 2020).

However, the elliptical potential lemma requires that the learner updates the design matrix at the end of every round with the most recent action. This is not the case for the summation in Equation (4.17), as the feedback associated with the most recent action is not necessarily observable immediately. Therefore, it may not be used to increment the observed design matrix. Moreover, there will likely be rounds where no feedback arrives at all and rounds where multiple feedbacks return to the agent.

Therefore, different technical tools are required, which we will develop in this section. Lemma 4.6 is the end result. Briefly, it establishes an elliptical potential lemma for the setting of delayed feedback to control Equation (4.17). This result relates the elliptical potential with the observed design matrix to the standard elliptical potential of Lemma 4.5.

Lemma 4.6. *Let $\{A_t : \|A_t\|_2 \leq 1\}_{t=1}^\infty$ and $\{\tau_t : \tau_t \geq 0\}_{t=1}^\infty$ be an arbitrary sequence of d -dimensional vectors and non-negative scalars, respectively. Then, it holds that:*

$$\sum_{t=1}^n \|A_t\|_{\bar{W}_{t-1}^{-1}}^2 \leq \sum_{t=1}^n \|A_t\|_{\bar{V}_{t-1}^{-1}}^2 + \sum_{t=1}^n \frac{1 + G_t + \tau_t}{2} \|A_t\|_{\bar{V}_{t-1}^{-1}}^2$$

where $\lambda \geq 1$, $\bar{V}_t$, $\bar{W}_t$ and Z_t are defined as-per Equations (4.3), (4.4) and (4.5), respectively.

Proving Lemma 4.6 requires two intermediate results. Specifically, Lemmas 4.7 and 4.8. These are stated and proven in Section 4.6.1.

4.6.1 Preliminary Results for the Proof of Lemma 4.6

Lemma 4.7. *Let $A, B \in \mathbb{R}^{d \times d}$ be symmetric positive semi-definite matrices. Then, AB and $A^{1/2}BA^{1/2}$ share the same set of eigenvalues and these eigenvalues are all non-negative.*

Proof. Theorem 1.3.22 of [Horn and Johnson \(2013\)](#) shows that the characteristic polynomials of AB and BA are equivalent when at least one of A or B are non-singular. Since A is a symmetric and positive semi-definite matrix, it follows that:

$$\lambda(AB) = \lambda(A^{1/2}A^{1/2}B) = \lambda(A^{1/2}BA^{1/2}).$$

where $\lambda(\cdot)$ denotes the spectrum of the matrix given as input. Furthermore, $A^{1/2}BA^{1/2}$ is a symmetric matrix. Therefore, it follows that:

$$x^T A^{1/2}BA^{1/2}x = y^T B y \geq 0$$

where $y = A^{1/2}x$ because $B \succeq 0$. Therefore, $\lambda(A^{1/2}BA^{1/2}) = \lambda(AB)$ and $\lambda_i(A^{1/2}BA^{1/2}) \geq 0$, which implies that AB has non-negative eigenvalues, as required. $\square$

Lemma 4.8. *Let $\{A_t : \|A_t\|_2 \leq 1\}_{t=1}^\infty$ and $\{\tau_t : \tau_t \geq 0\}_{t=1}^\infty$ be an arbitrary sequence of d -dimensional vectors and non-negative scalars, respectively. Then, it holds that:*

$$\frac{G_t}{\lambda} \bar{V}_t^{-1} \succeq \bar{V}_t^{-1} Z_t \bar{W}_t^{-1}$$

where $\bar{V}_t$, $\bar{W}_t$ and Z_t be defined as-per Equations (4.3), (4.4) and (4.5), respectively.

Proof. Recall $A \succeq B \iff A - B \succeq 0$. Therefore, it is equivalent to prove that the difference between the two matrices is positive semi-definite:

$$D_t = \frac{G_t}{\lambda} \bar{V}_t^{-1} - \bar{V}_t^{-1} Z_t \bar{W}_t^{-1} = \bar{V}_t^{-1} \left(\frac{G_t}{\lambda} I - Z_t \bar{W}_t^{-1} \right). \quad (4.18)$$

This proof takes a four-stepped approach.

1. Equation (4.18) is a symmetric matrix.
2. Equation (4.18) has the same set of eigenvalues as another matrix.

3. This similar matrix has non-negative eigenvalues.
4. Finally, we chain the above three steps in reverse order and recall basic facts about symmetric matrices to prove the claim.

Step 1. From Equation (4.19), it is possible to rewrite the matrix in question as follows:

$$D_t = \frac{G_t}{\lambda} \bar{V}_t^{-1} + \bar{V}_t^{-1} - \bar{W}_t^{-1} = \left(\frac{G_t}{\lambda} I + I \right) \bar{V}_t^{-1} - \bar{W}_t^{-1}$$

which is just the sum of symmetric matrices. Therefore, D_t is symmetric and this establishes the claim for Step 1. Leveraging the eigendecomposition for this symmetric matrices reveals that:

$$x^T D_t x = x^T U \Lambda U^T x = \sum_{i=1}^d \lambda_i(D_t) x^T u_i u_i^T x = \sum_{i=1}^d \lambda_i(D_t) \langle x, u_i \rangle^2.$$

Consequently, $D_t \succeq 0 \iff x^T D_t x \geq 0$ for all $x \in \mathbb{R}^d$ if and only if all of its eigenvalues are non-negative.

Step 2. Define the following matrix:

$$\tilde{D}_t := \bar{V}_t^{-1/2} \left(\frac{G_t}{\lambda} I - Z_t \bar{W}_t^{-1} \right) \bar{V}_t^{-1/2}$$

Applying Lemma 4.7 with $A = \bar{V}_t^{-1}$ and $B = (G_t/\lambda)I - Z_t \bar{W}_t^{-1}$ reveals that $D_t = AB$ and $\tilde{D}_t = A^{1/2} B A^{1/2}$ share the same set of eigenvalues.

Step 3. Showing $\tilde{D}_t \succeq 0$ proves it must have non-negative eigenvalues, as this is a necessary condition for the associated quadratic form to be non-negative: $x^T \tilde{D}_t x \geq 0$. Recall that every matrix can be decomposed into the sum of symmetric and anti-symmetric matrices:

$$\tilde{D}_t = \frac{1}{2} (\tilde{D}_t + \tilde{D}_t^T) + \frac{1}{2} (\tilde{D}_t - \tilde{D}_t^T).$$

Let $x \in \mathbb{R}^d$ be an arbitrary vector. Then, $\tilde{D}_t - \tilde{D}_t^T$ has a quadratic form that evaluates to the following:

$$\begin{aligned} y &= x^T (\tilde{D}_t - \tilde{D}_t^T) x \\ &= \left(x^T (\tilde{D}_t - \tilde{D}_t^T) x \right)^T \end{aligned}$$

$$\begin{aligned}
&= -x^T \tilde{D}_t x + x^T \tilde{D}_t^T x \\
&= -x^T (\tilde{D}_t - \tilde{D}_t^T) x \\
&= -y
\end{aligned}$$

which holds if and only if $y = 0$. Therefore, the anti-symmetric component of the matrix does not contribute to the evaluation of the quadratic form. Applying this fact gives:

$$\begin{aligned}
x^T \tilde{D}_t x &= \frac{1}{2} x^T (\tilde{D}_t + \tilde{D}_t^T) x \\
&= \frac{1}{2} x^T \left(\bar{V}_t^{-1/2} \left(\frac{2G_t}{\lambda} I - Z_t \bar{W}_t^{-1} - \bar{W}_t^{-1} Z_t \right) \bar{V}_t^{-1/2} \right) x \\
&= \frac{G_t}{\lambda} x^T \bar{V}_t^{-1} x - \frac{1}{2} x^T \bar{V}_t^{-1/2} (Z_t \bar{W}_t^{-1} + \bar{W}_t^{-1} Z_t) \bar{V}_t^{-1/2} x \\
&= \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \frac{1}{2} x^T \bar{V}_t^{-1/2} (Z_t \bar{W}_t^{-1} + \bar{W}_t^{-1} Z_t) \bar{V}_t^{-1/2} x
\end{aligned}$$

Now, $A = Z_t \bar{W}_t^{-1} + \bar{W}_t^{-1} Z_t$ is a real-value symmetric matrix. Therefore, its eigendecomposition is given by $A = Q \Lambda Q^T$ where Λ is a diagonal matrix containing the eigenvalues and Q is an orthogonal matrix whose columns contain the unit eigenvectors.

$$\begin{aligned}
x^T \tilde{D}_t x &= \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \frac{1}{2} x^T \bar{V}_t^{-1/2} Q \Lambda Q^T \bar{V}_t^{-1/2} x \\
&= \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \frac{1}{2} \tilde{x}^T \Lambda \tilde{x} \quad (\tilde{x} := Q^T \bar{V}_t^{-1/2} x) \\
&= \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \frac{1}{2} \sum_{i=1}^d \lambda_i \tilde{x}_i^2
\end{aligned}$$

where λ_i is the i -th largest eigenvalue of matrix $A = Z_t \bar{W}_t^{-1} + \bar{W}_t^{-1} Z_t$. Clearly, A is symmetric because it is the sum of a matrix and its transpose. Therefore, A has singular values that are the absolute values of its eigenvalues:

$$\sigma_i(A) = |\lambda_i(A)|$$

where $\sigma_i(A)$ denotes the i -th largest singular value of the matrix. Then, it follows that:

$$\begin{aligned}
x^T \tilde{D}_t x &\geq \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \frac{1}{2} \sum_{i=1}^d \sigma_i(A) \tilde{x}_i^2 & (\lambda_i(A) \leq \sigma_i(A)) \\
&\geq \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \frac{1}{2} \sigma_1(A) \sum_{i=1}^d \tilde{x}_i^2 & (\sigma_i(A) \leq \sigma_1(A))
\end{aligned}$$

$$\begin{aligned}
&= \frac{G_t}{\lambda_*} \|x\|_{V_t(\lambda_*)^{-1}}^2 - \frac{1}{2} \sigma_1(A) \|\tilde{x}\|_2^2 && \text{(Definition of } \|\cdot\|_2\text{)} \\
&= \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \frac{1}{2} \sigma_1(A) \|x\|_{\bar{V}_t^{-1}}^2 && \text{(Definition of } \tilde{x}\text{)} \\
&= \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \frac{1}{2} \|A\|_2 \|x\|_{\bar{V}_t^{-1}}^2 && \text{(For } A \in \mathbb{R}^{d \times d}: \sigma_1(A) = \|A\|_2\text{)}
\end{aligned}$$

Therefore, to lower bound the quadratic form, an upper bound the matrix norm of $A = Z_t \bar{W}_t^{-1} + \bar{W}_t^{-1} Z_t$ is required. Proceeding with this step and utilising the sub-additive and sub-multiplicative properties of matrix norms gives:

$$\begin{aligned}
\|A\|_2 &= \|Z_t \bar{W}_t^{-1} + \bar{W}_t^{-1} Z_t\|_2 \\
&\leq \|Z_t \bar{W}_t^{-1}\|_2 + \|\bar{W}_t^{-1} Z_t\|_2 && \text{(Sub-additive)} \\
&\leq \|Z_t\|_2 \|\bar{W}_t^{-1}\|_2 + \|\bar{W}_t^{-1}\|_2 \|Z_t\|_2 && \text{(Sub-multiplicative)} \\
&= 2 \|Z_t\|_2 \|\bar{W}_t^{-1}\|_2
\end{aligned}$$

From Equation (4.4): $\|\bar{W}_t^{-1}\|_2 = 1/\lambda_d(\bar{W}_t) \leq 1/\lambda$. Further, Equation 4.5 can be combined with the absolute homogeneity, sub-additive and sub-multiplicative properties of the spectral norm to yield:

$$\begin{aligned}
\|Z_t\|_2 &= \left\| \sum_{s=1}^t 1\{s + \tau_s > t\} A_s A_s^T \right\|_2 \\
&\leq \sum_{s=1}^t \|1\{s + \tau_s > t\} A_s A_s^T\|_2 && \text{(Sub-additive)} \\
&= \sum_{s=1}^t 1\{s + \tau_s > t\} \|A_s A_s^T\|_2 && \text{(Abs. Hom)} \\
&\leq \sum_{s=1}^t 1\{s + \tau_s > t\} && \text{(rank}(A_s A_s^T) = 1\text{)} \\
&= G_t
\end{aligned}$$

Substituting this into the above gives:

$$\begin{aligned}
x^T \tilde{D}_t x &\geq \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \|Z_t\|_2 \|\bar{W}_t^{-1}\|_2 \|x\|_{\bar{V}_t^{-1}}^2 \\
&\geq \frac{G_t}{\lambda} \|x\|_{\bar{V}_t^{-1}}^2 - \frac{2G_t}{2\lambda} \|x\|_{\bar{V}_t^{-1}}^2 \\
&= 0
\end{aligned}$$

Step 4. Now, Step 3 shows $\tilde{D}_t$ is positive semi-definite and this implies that all of its eigenvalues

are non-negative. Therefore, Step 2 shows D_t has non-negative eigenvalues. Finally, Step 1 shows D_t is a symmetric matrix. Since D_t is symmetric, non-negative eigenvalues implies positive semi-definiteness. Finally, $D_t \succeq 0$ implies that:

$$\frac{G_t}{\lambda} \bar{V}_t^{-1} \succeq \bar{V}_t^{-1} Z_t \bar{W}_t = M_t,$$

as required. $\square$

4.6.2 Proof of Lemma 4.6

With Lemmas 4.7 and 4.8, we are now ready to prove Lemma 4.6. This result controls the delayed elliptical potential that features in Equation (4.17).

Proof. Since $\lambda > 0$, $\bar{V}_t$ and $\bar{W}_t$ are invertible. Therefore, the relationship between the inverse of the observed design matrix and the inverse of the total design matrix can be established as follows:

$$\begin{aligned} \bar{W}_t^{-1} &= \bar{V}_t^{-1} + \bar{W}_t^{-1} - \bar{V}_t^{-1} \\ &= \bar{V}_t^{-1} + \bar{V}_t^{-1} \bar{V}_t \bar{W}_t^{-1} - \bar{V}_t^{-1} \bar{W}_t \bar{W}_t^{-1} \\ &= \bar{V}_t^{-1} + \bar{V}_t^{-1} (\bar{V}_t - \bar{W}_t) \bar{W}_t^{-1} \\ &= \bar{V}_t^{-1} + \bar{V}_t^{-1} Z_t \bar{W}_t^{-1} \end{aligned} \tag{4.19}$$

where the final equality rearranges Equation (4.6). Therefore, focussing on the t -th element of the summation gives:

$$\begin{aligned} \|A_t\|_{\bar{W}_{t-1}^{-1}} &= \|A_t\|_{\bar{V}_{t-1}^{-1} + \bar{V}_{t-1}^{-1} Z_{t-1} \bar{W}_{t-1}^{-1}} \\ &\leq \|A_t\|_{\bar{V}_{t-1}^{-1}} + \|A_t\|_{\bar{V}_{t-1}^{-1} Z_{t-1} \bar{W}_{t-1}^{-1}} \end{aligned}$$

where the first term is what appears in the standard elliptical potential lemma. Therefore, it suffices to focus on the second term. Expanding out the definition gives:

$$\begin{aligned} \|A_t\|_{\bar{V}_{t-1}^{-1} Z_{t-1} \bar{W}_{t-1}^{-1}} &= \sqrt{A_t^T \bar{V}_{t-1}^{-1} Z_{t-1} \bar{W}_{t-1}^{-1} A_t} \\ &= \sqrt{\text{Tr} \left(A_t^T \bar{V}_{t-1}^{-1} Z_{t-1} \bar{W}_{t-1}^{-1} A_t \right)} \\ &= \sqrt{\text{Tr} \left(\bar{V}_{t-1}^{-1} Z_{t-1} \bar{W}_{t-1}^{-1} A_t A_t^T \right)} \end{aligned}$$

$$\begin{aligned}
&\leq \sqrt{\text{Tr}(\bar{V}_{t-1}^{-1} Z_{t-1}) \text{Tr}(\bar{W}_{t-1}^{-1} A_t A_t^T)} && \text{(von Neumann Trace)} \\
&\leq \frac{1}{2} \text{Tr}(\bar{W}_{t-1}^{-1} A_t A_t^T) + \frac{1}{2} \text{Tr}(\bar{V}_{t-1}^{-1} Z_{t-1}) && \text{(AM-GM Inequality)}
\end{aligned}$$

From Equation (4.5), $Z_t = \sum_{s=1}^t 1\{s + \tau_s > t\} A_s A_s^T$. Applying properties of trace allows the previous display to be written as follows:

$$\begin{aligned}
\frac{1}{2} \text{Tr}(\bar{W}_{t-1}^{-1} A_t A_t^T) + \frac{1}{2} \text{Tr}(\bar{V}_{t-1}^{-1} Z_{t-1}) &= \frac{1}{2} \|A_t\|_{\bar{W}_{t-1}}^2 + \frac{1}{2} \sum_{s=1}^t 1\{s + \tau_s > t-1\} \|A_s\|_{\bar{V}_{t-1}}^2 \\
&= \frac{1}{2} \|A_t\|_{\bar{V}_{t-1}}^2 + \frac{1}{2} \|A_t\|_{\bar{V}_{t-1} Z_t \bar{W}_{t-1}}^2 + \frac{1}{2} \sum_{s=1}^t 1\{s + \tau_s > t-1\} \|A_s\|_{\bar{V}_{t-1}}^2
\end{aligned}$$

where the last equality follows from Equation (4.19). Finally, Lemma 4.8 gives:

$$\|A_t\|_{\bar{V}_{t-1} Z_t \bar{W}_{t-1}}^2 \leq \frac{G_t}{\lambda} \|A_t\|_{\bar{V}_{t-1}}^2 \leq G_t \|A_t\|_{\bar{V}_{t-1}}^2$$

where the last inequality follows from assuming $\lambda \geq 1$. Bringing everything together and introducing the outer summation gives:

$$\begin{aligned}
\sum_{t=1}^n \|A_t\|_{\bar{W}_{t-1}}^2 &\leq \sum_{t=1}^n \|A_t\|_{\bar{V}_{t-1}}^2 + \frac{1}{2} \sum_{t=1}^n \frac{1+G_t}{2} \|A_t\|_{\bar{V}_{t-1}}^2 + \frac{1}{2} \sum_{t=1}^n \sum_{s=1}^t 1\{s + \tau_s > t-1\} \|A_s\|_{\bar{V}_{t-1}}^2 \\
&\leq \sum_{t=1}^n \|A_t\|_{\bar{V}_{t-1}}^2 + \sum_{t=1}^n \frac{1+G_t}{2} \|A_t\|_{\bar{V}_{t-1}}^2 + \frac{1}{2} \sum_{t=1}^n \sum_{s=1}^t 1\{s + \tau_s > t-1\} \|A_s\|_{\bar{V}_{s-1}}^2 \quad (\bar{V}_t \succeq \bar{V}_{t-1}) \\
&= \sum_{t=1}^n \|A_t\|_{\bar{V}_{t-1}}^2 + \sum_{t=1}^n \frac{1+G_t}{2} \|A_t\|_{\bar{V}_{t-1}}^2 + \frac{1}{2} \sum_{t=1}^n \tau_t \|A_t\|_{\bar{V}_{t-1}}^2
\end{aligned}$$

where the last equality follows from expanding the two summations to reveal that the indicator ensures that each term contributes to the summation τ_t times. $\square$

4.7 The Pseudo-Regret of Delayed OFU-GLM

Lemma 4.6 upper bounds the delayed elliptical potential by the standard elliptical potential and an additive lower-order term that involves the delays. Furthermore, it only requires that the delays are an arbitrary sequence of non-negative integers. Therefore, this result is quite generic and can be applied to other problem settings.

From Lemmas 4.4 and 4.6, it follows that the last step in obtaining an upper bound on the regret of Algorithm 3 is to control the delays and the number of missing feedbacks in each round.

Subsequently, Lemma 4.5 can be applied in the usual manner to get an upper bound on the pseudo-regret. Assumption 4.4 gives high probability control on the delays:

$$\tau_t \leq \epsilon_t(\tau) \leq \epsilon_n(\tau)$$

Therefore, it is just the number of missing feedbacks that needs to be controlled to remove the round dependence and allow for a smooth application of the elliptical potential lemma. Luckily, [Joulani et al. \(2013\)](#) have developed an upper bound on the number of missing feedbacks when the delays are independent and identically distributed random variables. Lemma 4.9 differs from [Joulani et al. \(2013\)](#) only in the choice of the confidence parameter, which is because the desired result in this setting is a guarantee that holds with high probability for all $t \in \mathbb{N}$.

Lemma 4.9. *Define $G_t = \sum_{s=1}^t 1\{s + \tau_s > t\}$ and let $\{\tau_t\}_{t=1}^\infty$ be a sequence of independent and identically distributed random variables with a finite expectation: $\mathbb{E}[\tau] < \infty$. Then, it holds that:*

$$\mathbb{P} \left(\exists t \geq 1 : G_t \geq \mathbb{E}[\tau] + \log \left(\frac{t(t+1)}{\delta} \right) + \sqrt{2\mathbb{E}[\tau] \log \left(\frac{t(t+1)}{\delta} \right)} \right) \leq \delta.$$

Proof. [Joulani et al. \(2013\)](#) prove this result by noticing that the number of missing feedbacks in the t -th round is bounded. Therefore, Bernstein's inequality gives them the following guarantee:

$$\begin{aligned} \delta_t &\geq \mathbb{P} \left(G_t - \mathbb{E}[G_t] \geq \log \left(\frac{1}{\delta_t} \right) + \sqrt{2\mathbb{V}[G_t] \log \left(\frac{1}{\delta_t} \right)} \right) \\ &\geq \mathbb{P} \left(G_t - \mathbb{E}[\tau] \geq \log \left(\frac{1}{\delta_t} \right) + \sqrt{2\mathbb{E}[\tau] \log \left(\frac{1}{\delta_t} \right)} \right) \end{aligned} \quad (\text{Bernstein})$$

where $\delta_t \in (0, 1)$ denotes the confidence parameter for the t -th round. The last inequality follows from the fact that the expected value and the variance of the number of missing feedbacks can be bounded by the expected delay:

$$\mathbb{E}[G_t] \leq \mathbb{E}[\tau]$$

$$\mathbb{V}[G_t] \leq \mathbb{E}[\tau]$$

Setting $\delta_t = \delta/t(t+1)$ and taking the union bound gives a time-uniform bound:

$$\mathbb{P} \left(\exists t \geq 1 : G_t - \mathbb{E}[\tau] \geq \frac{2}{3} \log \left(\frac{t(t+1)}{\delta} \right) + 2\sqrt{\mathbb{E}[\tau] \log \left(\frac{t(t+1)}{\delta} \right)} \right) \leq \sum_{t=1}^{\infty} \delta_t \leq \delta$$

as required. $\square$

Theorem 4.2 proves a high probability upper bound on the pseudo-regret of Algorithm 3, Delayed OFU-GLM.

Theorem 4.2. *Under Assumptions 4.1, 4.2, 4.3 and 4.4, Delayed OFU-GLM with any regularisation parameter $\lambda = \alpha a(\phi)/\kappa \geq 1$ has pseudo-regret that satisfies:*

$$\hat{R}_n(\pi, \theta_\star) \lesssim \frac{RL_\mu}{\kappa} d\sqrt{n} + \frac{RL_\mu}{\kappa} d^{3/2}(\epsilon_n(\tau) + \mathbb{E}[\tau])$$

with probability at least $1 - 3\delta$.

Proof. Define the following *bad* event:

$$B = B_1 \cup B_2 \cup B_3$$

$$B_1 = \{\exists t \in \mathbb{N} : \theta_\star \notin \mathcal{C}_t\}$$

$$B_2 = \{\exists t \in \mathbb{N} : \tau_t \geq \epsilon_t(\tau)\}$$

$$B_3 = \{\exists t \in \mathbb{N} : G_t \geq \mathbb{E}[\tau] + \epsilon_t(G)\}.$$

where

$$\epsilon_t(G) = \log\left(\frac{t(t+1)}{\delta}\right) + \sqrt{2\mathbb{E}[\tau] \log\left(\frac{t(t+1)}{\delta}\right)}$$

From Theorem 4.1, Assumption 4.4 and Lemma 4.9, it follows that:

$$\mathbb{P}(B) \leq \mathbb{P}(B_1) + \mathbb{P}(B_2) + \mathbb{P}(B_3) \leq 3\delta.$$

Therefore, the proceeding analysis assumes that the complement of the bad event B . Consequently, the set membership and inequalities reverse and the guarantee will hold with probability at least $1 - 3\delta$. From Lemma 4.4, where the pseudo-regret was analysed on a subset of the complement of the bad event, it follows that:

$$\begin{aligned} \hat{R}_n(\pi, \theta_\star) &\leq 2L_\mu \sqrt{\beta_n} \sum_{t=1}^n \|A_t\|_{\bar{W}_{t-1}^{-1}} \\ &\leq 2L_\mu \sqrt{\beta_n} \sum_{t=1}^n \|A_t\|_{\bar{V}_{t-1}^{-1}} + L_\mu \sqrt{\beta_n} \sum_{t=1}^n (1 + \tau_t + G_t) \|A_t\|_{\bar{V}_{t-1}^{-1}}^2 \quad (\text{Lemma 4.6}) \\ &\leq 2L_\mu \sqrt{\beta_n} \sum_{t=1}^n \|A_t\|_{\bar{V}_{t-1}^{-1}} + L_\mu \sqrt{\beta_n} \sum_{t=1}^n (1 + \mathbb{E}[\tau] + \epsilon_n(\tau) + \epsilon_n(G)) \|A_t\|_{\bar{V}_{t-1}^{-1}}^2 \end{aligned}$$

$$\leq 2L_\mu \sqrt{\beta_n} \sqrt{n \sum_{t=1}^n \|A_t\|_{\bar{V}_{t-1}^{-1}}^2} + (1 + \mathbb{E}[\tau] + \epsilon_n(\tau) + \epsilon_n(G)) L_\mu \sqrt{\beta_n} \sum_{t=1}^n \|A_t\|_{\bar{V}_{t-1}^{-1}}^2$$

where the penultimate inequality utilises the fact that none of the bad events have occurred, and $\epsilon_t(\tau) \leq \epsilon_n(\tau)$ and $\epsilon_t(G) \leq \epsilon_n(G)$. The last inequality utilises Cauchy-Schwarz. The remaining summations depend only on the elliptical potential. Applying Lemma 4.5 gives:

$$\begin{aligned} \hat{R}_n(\pi, \theta_*) &\leq 2L_\mu \sqrt{\beta_n} \sqrt{2dn \ln \left(\frac{d\lambda + n}{d\lambda} \right)} + 2dL_\mu \sqrt{\beta_n} (1 + \epsilon_n(\tau) + \epsilon_n(G) + \mathbb{E}[\tau]) \ln \left(\frac{d\lambda + n}{d\lambda} \right) \\ &= 2L_\mu \sqrt{\beta_n} \left(\sqrt{2dn \ln \left(\frac{d\lambda + n}{d\lambda} \right)} + d(1 + \epsilon_n(\tau) + \epsilon_n(G) + \mathbb{E}[\tau]) \ln \left(\frac{d\lambda + n}{d\lambda} \right) \right) \end{aligned}$$

From Theorem 4.1, the widths of the confidence sets have the following function form:

$$\begin{aligned} \sqrt{\beta_n} &= 2m\sqrt{\lambda} + \frac{2R}{\kappa} \sqrt{2 \log \left(\frac{\det(\bar{W}_n)^{1/2}}{\delta \lambda^{d/2}} \right)} \\ &= 2m\sqrt{\lambda} + \frac{2R}{\kappa} \sqrt{\log \left(\frac{\det(\bar{W}_n)}{\lambda^d} \right) + 2 \ln \left(\frac{1}{\delta} \right)}. \end{aligned}$$

The determinant of the observed design matrix can be upper bounded utilising the AM-GM inequality:

$$\det(\bar{W}_n) = \prod_{i=1}^d \lambda_i(\bar{W}_n) \leq \left(\frac{1}{d} \sum_{i=1}^d \lambda_i(\bar{W}_n) \right)^d = \left(\frac{1}{d} \text{Tr}(\bar{W}_n) \right)^d$$

Furthermore, Assumption 4.1 allows the trace of the observed design matrix to be upper bounded by using the linearity and cyclic properties of trace:

$$\begin{aligned} \text{Tr}(\bar{W}_n) &= \text{Tr}(\lambda I + \sum_{t=1}^n 1\{t + \tau_t \leq n\} A_t A_t^T) \\ &= d\lambda + \sum_{t=1}^n 1\{t + \tau_t \leq n\} \text{Tr}(A_t A_t^T) \\ &= d\lambda + \sum_{t=1}^n 1\{t + \tau_t \leq n\} \|A_t\|_2^2 \\ &\leq d\lambda + \sum_{t=1}^n 1\{t + \tau_t \leq n\} \\ &\leq d\lambda + n \end{aligned}$$

Combining these inequalities gives an upper bound on the width of the confidence sets at the

end of the final round:

$$\begin{aligned}
\sqrt{\beta_n} &= 2m\sqrt{\lambda} + \frac{2R}{\kappa} \sqrt{\log \left(\frac{\det(\bar{W}_n)}{\lambda^d} \right) + 2 \ln \left(\frac{1}{\delta} \right)} \\
&\leq 2m\sqrt{\lambda} + \frac{2R}{\kappa} \sqrt{\log \left(\frac{\left(\frac{1}{d} \text{Tr}(\bar{W}_n) \right)^d}{\lambda^d} \right) + 2 \ln \left(\frac{1}{\delta} \right)} \\
&\leq 2m\sqrt{\lambda} + \frac{2R}{\kappa} \sqrt{d \log \left(\frac{\text{Tr}(\bar{W}_n)}{d\lambda} \right) + 2 \ln \left(\frac{1}{\delta} \right)} \\
&\leq 2m\sqrt{\lambda} + \frac{2R}{\kappa} \sqrt{d \log \left(\frac{d\lambda + n}{d\lambda} \right) + 2 \ln \left(\frac{1}{\delta} \right)}
\end{aligned}$$

Therefore, omitting logarithmic factors and numeric constants, the pseudo-regret is upper bounded as follows:

$$\hat{R}_n(\pi, \theta_\star) \lesssim \frac{RL_\mu}{\kappa} d\sqrt{n} + \frac{RL_\mu}{\kappa} d^{3/2}(\epsilon_n(\tau) + \mathbb{E}[\tau])$$

as required. $\square$

4.7.1 Discussion of the Result

Theorem 4.2 demonstrates that it is possible to devise an algorithm with an additive dependence on the delay in structured settings. Previously, theoretical guarantees in structured settings had a multiplicative dependence on the delay. Specifically, [Zhou et al. \(2019\)](#) and [Blanchet et al. \(2020\)](#) devise an algorithm whose pseudo-regret is upper bounded by:

$$\hat{R}_n(\pi, \theta_\star) \lesssim \frac{RL_\mu}{\kappa} d\sqrt{n} + \sqrt{dn(\mathbb{E}[\tau] + M_\tau)}$$

with probability at least $1 - 5\delta$. Here, M_τ is a positive constant such that the tails of the delay distribution beyond this value are approximately subgaussian. Therefore, Algorithm 3 and Theorem 4.2 offers an improvement over the existing guarantees ([Zhou et al., 2019](#); [Blanchet et al., 2020](#)). This improvement is quite substantial when $\mathbb{E}[\tau] \approx \sqrt{n}$. Here, Theorem 4.2 would yield a horizon-dependence of the order $\sqrt{n}$. Conversely, [Zhou et al. \(2019\)](#) and [Blanchet et al. \(2020\)](#) would suffer a sub-optimal horizon-dependence that is of the order $n^{3/4}$.

Assumption 4.4 is relatively standard in the literature and requires deriving an appropriate tail bound for the random delays. Furthermore, this tail bound is only needed within the theoretical analysis and it is not a parameter that is required as input for the algorithm. Generally, the

subexponential class of distributions is of practical relevance (Chapelle, 2014). For this particular setting, the value that appears in the bound is the following (Wainwright, 2019):

$$\epsilon_t(\tau) = \mathbb{E}[\tau] + \min \left\{ \sqrt{2v^2 \log \left(\frac{t(t+1)}{\delta} \right)}, 2b \log \left(\frac{t(t+1)}{\delta} \right) \right\}$$

where v and b are the parameters that govern the heaviness of the tails. For applications where the delay is almost surely bounded, it is sufficient to plug in this uniform upper bound on the delays and have the guarantees hold with probability at least $1 - 2\delta$.

However, Assumption 4.4 could be relaxed significantly. Notably, Lemma 4.6 did not require any assumptions on the delays, other than non-negativity. Further, Theorem 4.1 holds under the broader assumption of the noise being conditionally subgaussian. This conditioning allows for the noise to depend on the historical sequence of actions and delays, as well as the historical values of the noise associated with rounds whose feedback has been observed. Therefore, a natural line of enquiry would be to look at the possibility of delays depending on the chosen action. This would not influence the confidence sets, which currently allow for this possibility, or the algebraic analysis. Only Lemma 4.9 and the tail bound in Assumption 4.4 would need changing to control the pseudo-regret of Delayed OFU-GLM for action-dependent delays.

Finally, it is expected that an additive delay dependence will hold for other optimistic algorithms that have been designed for the immediate feedback setting. Indeed, at the heart of the probabilistic analysis is a high probability upper bound on the following quantity:

$$\left\| \sum_{s=1}^t 1\{s + \tau_s \leq t\} A_s \eta_s \right\|_{\bar{W}_t^{-1}}.$$

Lemma 4.3 verifies that this quantity is a non-negative supermartingale under the information structure that is imposed on the agent by the introduction of the delays. Quantities similar to this martingale have been upper bounded, with high probability, with various techniques in the immediate feedback setting (Filippi et al., 2010b; Jun et al., 2017; Li et al., 2017; Fauray et al., 2020; Abeille et al., 2021; Jun et al., 2021; Fauray et al., 2022). Generally, it is expected that these probabilistic analyses will remain valid in the delayed feedback setting. Notably, Fauray et al. (2020) utilise a different weighting for the norm to significantly improve the κ -dependence for Bernoulli rewards. This improvement in problem-dependent constants is expected to transfer

naturally to the delayed feedback setting and could yield significant improvements in practice.

4.8 Experimental Evaluation

Theorem 4.2 suggests that Delayed OFU-GLM has sub-linear worst-case guarantees. This subsection presents experimental evidence to support the theoretical derivations. This is done by simulating the generalised linear bandit environment with Gaussian rewards. Specifically, the rewards were sampled as follows:

$$X_t = \langle A_t, \theta_\star \rangle + \eta_t$$

where $\eta_t \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$ is Gaussian noise with unit variance. The experiments were conducted with the following unknown parameter vector:

$$\theta_\star = \frac{\vec{1}}{\sqrt{d}}$$

where $d \in \{5, 10, 20\}$ represents the dimensionality of the problem. This parameter vector was kept fixed across 30 independent runs, which allowed for the computation of upper and lower bounds on the empirical performance. Each independent run consists of $n = 10,000$ rounds and the agent was presented with $k = 100$ randomly sampled action vectors from the d -dimensional unit ball in each round. This was chosen to satisfy the assumptions made by [Zhou et al. \(2019\)](#) and [Blanchet et al. \(2020\)](#). This assumption is exploited within their algorithm to derive tighter confidence sets that exploit the independence of the decision sets. Thus, the comparison is not necessarily fair, as their algorithm is designed to exploit this additional information about the environment. Finally, constant, uniform, Poisson and geometric delays were considered. Each of these delays were parametrised with four different settings such that:

$$\mathbb{E}[\tau] = \{100, 250, 500, 1000\}.$$

Algorithm 3, Delayed OFU-GLM, is compared to the DUCB-GLCB ([Zhou et al., 2019](#); [Blanchet et al., 2020](#)). For Delayed OFU-GLM, $\alpha = \kappa = a(\phi) = 1$ were chosen according to the density of the reward distribution, as per the discussion in Section 4.4. For DUCB-GLCB of [Zhou et al. \(2019\)](#) and [Blanchet et al. \(2020\)](#), the same parameters were used. This algorithm also requires an initial period of exploration, which was performed until the minimum eigenvalue of the observed design matrix surpassed 1; the smallest value that is permitted by their theory.

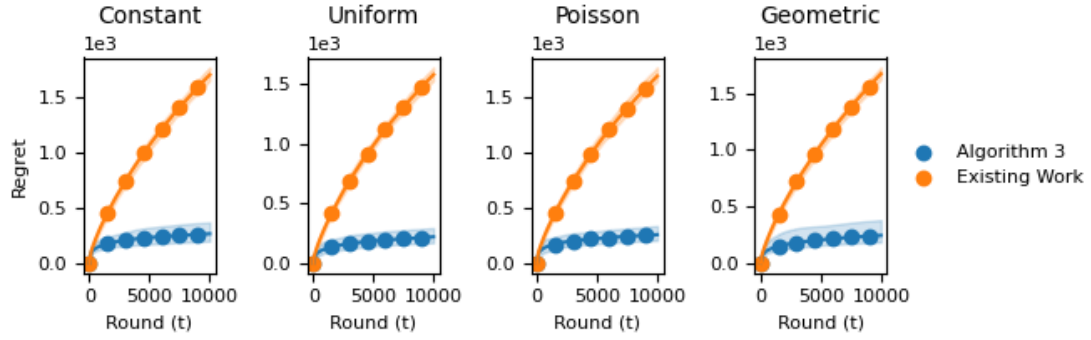


Figure 4.1: Experimental results for $d = 5$ and $\mathbb{E}[\tau] = 100$. The boundary of the shaded regions represent the minimum and maximum pseudo-regret across the 30 independent runs.

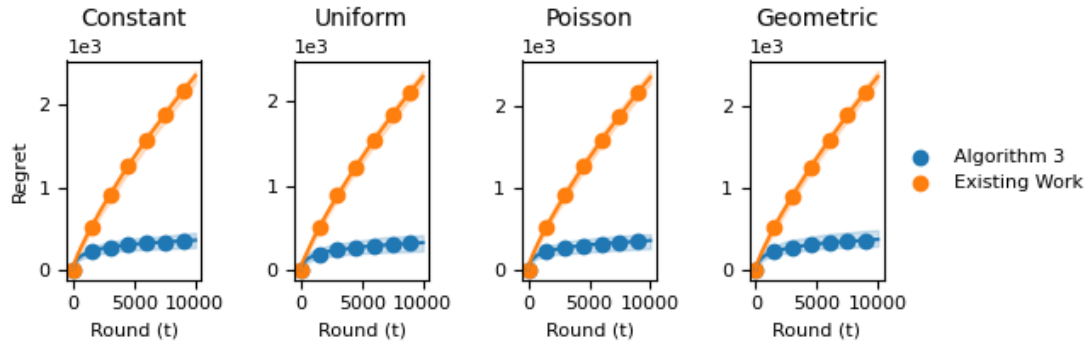


Figure 4.2: Experimental results for $d = 10$ and $\mathbb{E}[\tau] = 100$. The boundary of the shaded regions represent the minimum and maximum pseudo-regret across the 30 independent runs.

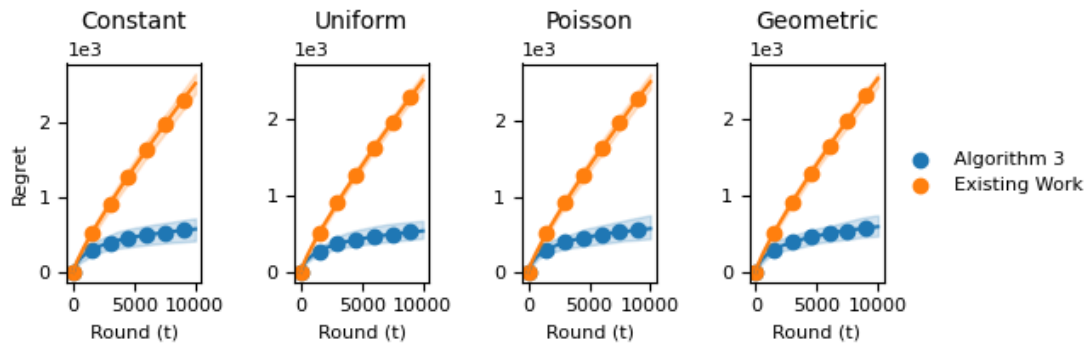


Figure 4.3: Experimental results for $d = 20$ and $\mathbb{E}[\tau] = 100$. The boundary of the shaded regions represent the minimum and maximum pseudo-regret across the 30 independent runs.

Figures 4.1, 4.2 and 4.3 display the pseudo-regret as a function of the number of rounds for $d \in \{5, 10, 20\}$, respectively. These empirical findings suggest that removing the delay dependence from the width of the confidence sets has substantial empirical benefits, which aligns with the well-established fact that tighter confidence sets yield better guarantees and improved performance.

From Figures 4.1, 4.2 and 4.3, it is difficult to assess the impact of the delay distribution on the performance of the algorithms. Further, they display the empirical results for $\mathbb{E}[\tau] = 100$ only. Figure 4.4 visualises the empirical evidence to better display how the pseudo-regret at the end of the final round is impacted by an increase in the expected delay for Algorithm 3 with $d \in \{5, 10, 20\}$.

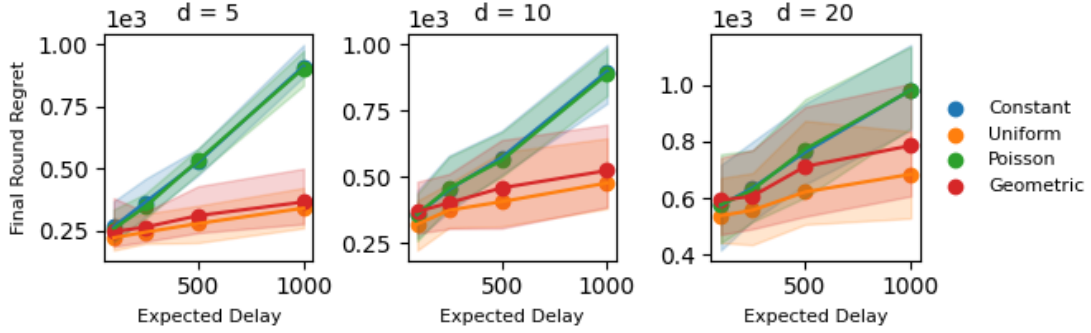


Figure 4.4: Experimental results visualising the impact of the expected value and distribution of the delays on the final-round pseudo-regret of Algorithm 3. The boundary of the shaded regions represent the minimum and maximum pseudo-regret across the 30 independent runs.

Generally, Figure 4.4 is in agreement with the theoretical guarantees presented in Theorem 4.2. Namely, that the relationship between the final-round pseudo-regret and the expected delay is linear. However, the empirical results suggest that the performance depends on other properties of the delays. For Poisson and constant delays, the increase in the pseudo-regret appears to be greater, and these distributions happen to have the lowest variances of the distributions considered. This suggests that it is not only the expected value of the delays that influence the performance and that other moments of their distributions have an impact on the pseudo-regret of Algorithm 3.

4.9 Conclusion

This chapter has studied the impact of delayed feedback on algorithms for generalised linear bandits. Under Assumption 4.4, that the delays have an appropriate tail bound, it was shown that it is possible to recover the additive penalty for the delays from the multi-armed bandit setting. These theoretical results significantly improve on prior work, where existing algorithms suffer a multiplicative penalty and require a-priori knowledge of the delay distribution as input. Reducing the delay dependence from multiplicative to additive was possible by introducing a novel technique to carefully separate the delays from the difficulty of the learning problem. Doing so allowed for the derivation of tighter confidence sets, which is well-known to yield better theoretical guarantees and superior empirical performance. Indeed, the theoretical techniques introduced in this chapter might be useful in other bandit problems with complex feedback structures.

Generally, Theorem 4.2 matches the intuition that the delays should become irrelevant once the agent has a reasonably accurate estimate of the unknown parameter vector. Furthermore, it recovers the additive delay penalty from the multi-armed bandit problem, despite the additional complexities that are introduced by the structure of generalised linear models. However, there are several direct extensions of the work presented in this chapter that would be of theoretical and practical relevance.

- **Thompson Sampling.** Experimental evidence has shown that the explicit randomness that is present in the decision-making of Thompson Sampling is helpful in the setting of delayed feedback (Chapelle and Li, 2011; Chapelle, 2014). Intuitively, for fixed sets of actions, the randomness allows for additional exploration while the agent is waiting for feedback to return. Conversely, optimistic algorithms would repeatedly select the same action until a reward from a previous round becomes observable. Combining the techniques found in Chapter 4 with those of Russo and Roy (2014) will likely yield an additive delay-dependence on the Bayesian regret.
- **Logistic Bandits.** Theorem 4.2 does have an undesirable dependency on κ in the leading-order term and in the additive delay-dependent term. This κ -dependence has been the subject of interest in the special case of Bernoulli rewards, which corresponds to the logistic bandit problem (Fauray et al., 2020; Abeille et al., 2021; Fauray et al., 2022). This line of work has almost removed the κ dependence entirely. Combining the techniques developed

in these papers with those introduced in this chapter should translate the improved κ -dependence to the setting of delayed feedback.

- Dimension-Dependence.** Theorem 4.2 has $d^{3/2}$ scaling the expected delay. However, for unstructured action sets, it is possible to achieve a scaling factor of d in the multi-armed bandit setting (Joulani et al., 2013). Following Howson et al. (2023b), the paper associated with Chapter 4, there has been positive progress on answering this question. Vakili et al. (2023), in the kernelised bandit setting, were able to remove the dimension dependence entirely and obtain an additive delay dependence of just the expected delay. This was possible by combining pure exploration and action elimination with an interesting batching scheme. However, in this case, prior knowledge of the expected delay is required to appropriately choose the number of rounds within each batch. Whether it is possible to obtain similarly positive results without knowledge of the delays remains unclear.
- Tail Assumptions on the Delays.** Theorem 4.2 requires an assumption on the tails of the delay distribution. This assumption is standard in the literature (Zhou et al., 2019; Blanchet et al., 2020; Vakili et al., 2023). However, assumptions on the tails of the delay distribution are not required in the multi-armed bandit setting (Joulani et al., 2013; Mandel et al., 2015; Pike-Burke et al., 2018). Chapter 5 shows that such assumptions are not required in the episodic reinforcement learning problem either. Therefore, is it possible to obtain high probability bounds on the pseudo-regret that only require that the delays have a finite expected value or variance? Tackling this problem will likely require adjustments to the technical tools developed in this chapter. Alternatively, this could be tackled by changing the structure of the algorithm. Chapter 5 shows that reducing the updating frequency is a useful algorithmic technique for separating the delays from the difficulty of the learning problem. Integrating this technique into the generalised linear bandit problem might be able to remove the tail assumption entirely.

Chapter 5

Delayed Feedback in Episodic Reinforcement Learning

This chapter is based on research published in AISTATS and was conducted under the supervision of Ciara Pike-Burke and Sarah Filippi ([Howson et al., 2023a](#)).

5.1 Introduction

Chapter 4 focuses on quantifying the impact of delayed feedback in the generalised linear bandit problem, where the agent only needs to consider maximising their immediate expected rewards. However, this model for sequential decision-making is inadequate for applications where taking an action influences the environment. Therefore, a natural next step is to investigate the impact of delayed feedback in the episodic reinforcement learning problem.

Chapter 4 highlights many reasons why feedback may not be immediate in many real-world applications. These reasons persist in the more complex setting of reinforcement learning. For example, an application in healthcare might involve prescribing a treatment protocol for a population of patients with a chronic illness. Here, the medical practitioner can be thought of as the algorithm who prescribes an adaptive treatment plan. This treatment plan is then given to, and executed by, the patient over a period of time. Feedback about the quality of the plan is provided upon the patient's return. Meanwhile, the medical practitioner will continue prescribing treatment plans to other patients within the population, without the valuable feedback from the

patients that are currently undergoing treatment. Other examples include wearable technology and autonomous vehicles, where the heavy computation involved in policy updating is likely to occur on a separate machine. This forces the communication of information, which naturally introduces a delay between the agent collecting feedback and the policy updater.

5.1.1 Existing & Related Work

The episodic reinforcement learning problem is relatively well-understood under immediate feedback setting, and there are numerous provably efficient algorithms (Jaksch et al., 2010; Filippi et al., 2010a; Fruit et al., 2020; Azar et al., 2017; Dann et al., 2018; Jin et al., 2018; Zhang et al., 2021). These algorithms were all discussed in Chapter 3. However, the guarantees for all of these algorithms depend on immediate feedback. Despite being of practical importance, there is limited literature on stochastically delayed feedback in reinforcement learning.

Previous work has considered stochastic delays in observing the current state and executing an action in a known MDP (Katsikopoulos and Engelbrecht, 2003). In either case, the authors show that it is possible to create an extended MDP, which involves increasing the size of the state space. This increase in the size of the state space depends on the maximum delay. Therefore, the computational complexity of computing the optimal policy in the delayed environment is necessarily greater.

Lancewicki et al. (2022) considered adversarially delayed feedback in MDPs. Here, the authors adapt the optimistic proximal policy optimisation algorithm (Cai et al., 2020; Shani et al., 2020). They show that the regret of this algorithm scales multiplicatively with the delays. However, this guarantee is for the adversarial setting, where the delay-dependence is multiplicative, even in the simpler multi-armed bandit problem. Lancewicki et al. (2022) also consider stochastic MDPs. For this setting, the authors propose an optimistic algorithm. The authors state that the pseudo-regret for an unspecified optimistic algorithm whose pseudo-regret is upper bounded by:

$$\hat{R}_K(\pi, M) \leq H^2 S \sqrt{AK} + H^2 S \tau_{\max}$$

where H is the number of decisions the learner must make per episode, K is the number of episodes, S is the number of states in the environment, A is the number of actions and $\tau_k \leq \tau_{\max}$ is a uniform upper bound on the delays. However, the leading order term in their regret bound is

loose. This can be seen by setting $\tau_{\max} = 0$ in their upper bound and noting that the lower bound for immediate feedback is: $\Omega(H^{3/2}\sqrt{SAK})$. Their approach also requires *a-priori* knowledge of the maximal delay to define a phase of explicit exploration; this quantity is often unknown in many practical applications. Further, the algorithm accrues linear regret in this exploration phase, and the maximal delay can be prohibitively large.

Following [Howson et al. \(2023a\)](#), the publication associated with this chapter, there has been continued interest in delayed feedback in reinforcement learning problems. [Kuang et al. \(2023\)](#), for example, utilise the tools developed in Chapter 4 to prove regret bounds for a variant of Thompson Sampling for linear MDPs. [Yang et al. \(2023\)](#) propose a generic wrapper algorithm for the various problem settings with delayed feedback. This wrapper algorithm takes in any immediate feedback algorithm that updates itself in batches and uses this structure to upper bound its regret in the delayed feedback setting. Therefore, the guarantees do not hold for the algorithms considered in this chapter of this thesis and the results are complementary.

5.2 Problem Setting

This chapter of the thesis focuses on quantifying the impact of delayed feedback on algorithms for episodic reinforcement learning. Chapter 3 introduces the episodic reinforcement learning problem with immediate feedback. However, it is helpful to recap the notation.

Formally, the environment is modelled as an episodic finite horizon Markov decision process:

$$M = (\mathcal{S}, \mathcal{A}, H, P, r)$$

where $\mathcal{S} = \{1, 2, \dots, S\}$ and $\mathcal{A} = \{1, 2, \dots, A\}$ denote the set of states and the set of actions, respectively. H denotes the horizon and gives the number of steps per episode. $P = \{P_h(\cdot|s, a)\}_{h,s,a}$ denotes the set of probability distributions over the next state, which will also be referred to as the transition functions. Finally, $r = \{r_h(s, a) \in [0, 1]\}_{(h,s,a) \in [H] \times \mathcal{S} \times \mathcal{A}}$ denotes the set of reward functions, which are assumed to be known.

Chapter 3 introduces how an agent interacts with this model of the environment. Specifically, the interaction occurs in a sequence of episodes: $k = 1, 2, \dots, K$. This chapter assumes that the algorithm computes a deterministic policy $\pi^k : \mathcal{S} \times [H] \rightarrow \mathcal{A}$ at the start of each episode. This

restriction on the class of policies does not preclude the optimal policy, which is known to be deterministic (Puterman, 1994). Once the algorithm has computed a policy, an agent uses said policy to sample feedback from the environment by: selecting an action, $A_h^k = \pi_h^k(S_h^k)$; receiving a reward, $R_h^k = r_h(S_h^k, A_h^k)$; and transitioning to the next state, $S_{h+1}^k \sim P_h(\cdot | S_h^k, A_h^k)$; for each $h = 1, \dots, H$. The feedback associated with the k -th episode is given by:

$$\{(S_h^k, A_h^k, R_h^k, S_{h+1}^k)\}_{h=1}^H. \quad (5.1)$$

The quality of a policy is measured via the value function. This is just the expected sum of rewards from the current stage of the episode to the end of the episode when starting in a particular state:

$$V_h^\pi(s) = \mathbb{E}^\pi \left[\sum_{h'=h}^H R_{h'}^k \middle| S_h^k = s \right]. \quad (5.2)$$

Throughout, $V_h^*(s) = \max_\pi \{V_h^\pi(s)\}$ will denote the maximum expected sum of rewards when starting in state s at the h -th stage. Following the immediate feedback literature, this chapter will measure the performance of an algorithm via the pseudo-regret:

$$\hat{R}_K(\pi, M) = \sum_{k=1}^K V_1^*(S_1^k) - V_1^{\pi_k}(S_1^k) := \sum_{k=1}^K \Delta_1^k \quad (5.3)$$

which is the cumulative sub-optimality of the sequence of policies computed by the algorithm when compared to an optimal policy. Domingues et al. (2021) show that the lower bound for the regret in the standard episodic reinforcement learning setting with stage-dependent transitions is: $\Omega(H^{3/2}\sqrt{SAK})$.

5.2.1 Delayed Feedback

Following Chapter 4 the symbol $\tau_k \in \mathbb{N}$ will denote the delay associated with the k -th episode. Throughout, it is assumed that the delays have the following property.

Assumption 5.1. *The delays are positive, independent and identically distributed random variables with a finite expected value: $\mathbb{E}[\tau] = \mathbb{E}[\tau_k] < \infty$ for all $k \in \mathbb{N}$.*

The sequence of states, actions and rewards associated with the policy executed in the k -th episode will now arrive at the end of round: $k + \tau_k$. Again, $\tau_k = 0$ recovers the immediate feedback setting, which is reasonably well understood. Procedure 5.1 introduces the sequential decision-making procedure with the introduction of the delays. Further, it introduces some

important terminology that draws a distinction between computing and executing the policy. Specifically, an algorithm computes the policy and this policy is given to an agent to execute in the environment.

Procedure 5.1: Delayed Feedback in Reinforcement Learning

For $k = 1, 2, \dots, K$:

- **Algorithm.** Computes $\pi^k = \{\pi_1^k, \pi_2^k, \dots, \pi_H^k\}$ with the observed feedback.
- **Agent.** Executes π^k in $M = (\mathcal{S}, \mathcal{A}, H, P, r)$.
- **Environment.** Reveals $\{(S_h^i, A_h^i, R_h^i, S_{h+1}^i)_{h=1}^H : i + \tau_i = k\}$.

The introduction of delays causes the feedback associated with an episode to return at some unknown time in the future, $k + \tau_k$. As a result, the algorithm may not be able to update its policy using feedback from episode k at the start of episode $k + 1$. Instead, it can only use feedback it has observed, and this corresponds to episodes where $i : i + \tau_i < k$. Consequently, it is helpful to introduce the observed and missing visitation counters:

$$N'_{k,h}(s, a) = \sum_{i: i + \tau_i < k} 1 \{(S_h^i, A_h^i) = (s, a)\} \quad (5.4)$$

$$N''_{k,h}(s, a) = \sum_{i: i + \tau_i \geq k} 1 \{(S_h^i, A_h^i) = (s, a)\}. \quad (5.5)$$

Equation (5.4) counts the number of observed visits to each state-action-stage triple. Equation (5.5) counts the number of visits to each state-action-stage triple that are missing and unknown to the algorithm. These are related to the total visitation counter, which is defined as:

$$N_{k,h}(s, a) = \sum_{i=1}^k 1 \{(S_h^i, A_h^i) = (s, a)\} = N'_{k,h}(s, a) + N''_{k,h}(s, a). \quad (5.6)$$

Chapter 3 introduced numerous provably efficient algorithms for episodic reinforcement learning. Therefore, this chapter of the thesis seeks to develop meta-algorithms that can modify any existing algorithm to work with delayed feedback. However, the theoretical analysis will focus on the class of optimistic model-based algorithms. The reasons for this particular focus are two-fold. Firstly, it is the largest class of algorithms for the immediate feedback setting (Jaksch et al., 2010; Filippi et al., 2010a; Talebi and Maillard, 2018; Azar et al., 2017; Dann et al., 2018; Fruit

et al., 2020; Zhang et al., 2021). Secondly, this class of algorithms contains nearly minimax optimal solutions (Azar et al., 2017).

Since the feedback is delayed, optimistic model-based algorithms must compute their bonuses and any required estimators using the observed visitation counters. This is because agents that have not yet returned still hold crucial information needed to determine the total number of times a state-action-stage has been visited. Moreover, even if the algorithm knew each agent's starting state, reconstructing the exact sequence of state-action-stage triples they encountered would introduce significant uncertainty due to the environment's stochastic transitions. Therefore, the remainder of this chapter will suppose that the algorithm utilises the following estimator and bonuses:

$$\hat{P}_{k,h}(s' | s, a) = \frac{\sum_{i:i < k} 1 \{ (S_h^i, A_h^i, S_{h+1}^i) = (s, a, s') \}}{\max\{1, N'_{k,h}(s, a)\}} \quad (5.7)$$

$$B_{k,h}(s, a) = \min \left\{ H - h + 1, \left(\frac{\beta_{k,h}^{(1)}}{\sqrt{N'_{k,h}(s, a)}} + \frac{\beta^{(2)}}{N'_{k,h}(s, a)} \right) \right\} \quad (5.8)$$

where $\beta_{k,h}^{(1)}$ and $\beta^{(2)}$ are algorithm-dependent quantities that depend only on the observed feedback. The proceeding list gives some examples of numerators for the exploration bonuses that are associated with algorithms, omitting logarithmic factors.

- UCRL2 (Jaksch et al., 2010) has $\beta_{k,h}^{(1)} = H\sqrt{S}$ and $\beta^{(2)} = 0$.
- UCBVI-BF (Azar et al., 2017) has $\beta_{k,h}^{(1)} = \sqrt{\mathbb{V}_{s' \sim \hat{P}_{k,h}(\cdot | s, a)} [\tilde{V}_{h+1}^k(s')]}$ and $\beta^{(2)} = H$, where $\tilde{V}_{h+1}^k(s')$ denotes an optimistic estimate of the next-stage value function computed via Equation (5.9).

Neu and Pike-Burke (2020) show that all optimistic model-based algorithms from the immediate feedback setting compute their policy utilising the following optimistic dynamic programming procedure:

$$\tilde{V}_h^k(s) = \min \left\{ H - h + 1, \max_{a \in \mathcal{A}} \left(r_h(s, a) + \langle \hat{P}_{k,h}(\cdot | s, a), \tilde{V}_{h+1}^k(\cdot) \rangle + B_{k,h}(s, a) \right) \right\} \quad (5.9)$$

where $\pi_h^k(s) := \arg \max r_h(s, a) + \langle \hat{P}_{k,h}(\cdot | s, a), \tilde{V}_{h+1}^k(\cdot) \rangle + B_{k,h}(s, a)$. For our theoretical results to hold, the following assumption is required.

Assumption 5.2. *The exploration bonus satisfies the following:*

$$\mathbb{P} \left(\exists (k, h) \in \mathbb{N} \times [H] : \langle (\hat{P}_{k,h} - P_h)(\cdot | s, a), V_{h+1}^*(\cdot) \rangle \geq B_{k,h}(s, a) \right) \leq \delta.$$

Assumption 5.2 is satisfied for the sub-class of value-optimistic algorithms, such as the UBEV (Dann et al., 2018) and UCBVI (Azar et al., 2017) algorithms. These algorithms satisfy Assumption 5.2 by design. Further, model-optimistic algorithms implicitly maintain bonuses that satisfy this assumption implicitly (Neu and Pike-Burke, 2020). Therefore, Assumption 5.2 captures a wide range of optimistic model-based algorithms. For our analysis, it will be helpful to define the following algorithm-dependent variable:

$$C = \begin{cases} 1 & \text{if } B_{k,h}(s, a) < \langle (\hat{P}_{k,h} - P_h)(\cdot | s, a), \tilde{V}_{h+1}^{\pi^k}(\cdot) \rangle \\ 0 & \text{otherwise} \end{cases} \quad (5.10)$$

where $B_{k,h}(s, a)$ satisfies Assumption 5.2. Intuitively, $C = 1$ corresponds to algorithms whose bonuses sit somewhere between the estimation error and the difference between the expectation of the optimistic value function under the estimated and true transition function. Since these bonuses must sit within a specific interval, which is potentially narrow, they are tighter.

5.3 Decomposing the Pseudo-Regret

Before proving guarantees for the class of optimistic model-based algorithms with delayed feedback, it is helpful to develop a regret decomposition. Theorem 5 of Neu and Pike-Burke (2020) provides a decomposition for the class of algorithms that explicitly maintain uncertainty around an estimate of the unknown transition functions. However, this may not include the class of algorithms that directly compute upper bounds on the value function. Indeed, the bonuses for these algorithms could be substantially smaller. Therefore, Lemma 5.1 seeks to provide a regret decomposition suitable for all optimistic model-based algorithms.

Lemma 5.1. *Under Assumption 5.2, with probability $1 - 4\delta$, the pseudo-regret of any algorithm can be upper bounded by:*

$$\hat{R}_K(\pi, M) \leq 6(H + C) \sqrt{KH \ln \left(\frac{K(K+1)}{\delta} \right)} + 6 \sum_{k=1}^K \sum_{h=1}^H \left(B_{k,h}(S_h^k, A_h^k) + \frac{CH^2SL}{N'_{k,h}(s_h^k, a_h^k)} \right)$$

where $L = \log(S^2AH\pi^2/6\delta')$ and C is defined as in Equation (5.10).

5.3.1 Preliminary Results for Lemma 5.1

From Equations (5.8), (5.8) and (5.9), the difference between the optimistic state value and the actual state value of the executed policy can be rewritten as follows:

$$\begin{aligned}
\tilde{\Delta}_h^k(S_h^k) &:= \tilde{V}_h^{\pi^k}(S_h^k) - V_h^{\pi^k}(S_h^k) \\
&= B_{k,h}(S_h^k, A_h^k) + \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} \rangle - \langle P_h(\cdot | S_h^k, A_h^k), V_{h+1}^{\pi^k} \rangle \\
&= B_{k,h}(S_h^k, A_h^k) + \langle (\hat{P}_{k,h} - P_h)(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} \rangle + \langle P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} - V_{h+1}^{\pi^k} \rangle \\
&= B_{k,h}(S_h^k, A_h^k) + \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} \rangle + \langle P_h(\cdot | S_h^k, A_h^k), \tilde{\Delta}_{h+1}^k \rangle \\
&= \tilde{\Delta}_{h+1}^k(S_{h+1}^k) + B_{k,h}(S_h^k, A_h^k) + \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} \rangle + \zeta_{k,h} \\
&= \tilde{\Delta}_{h+1}^k(S_{h+1}^k) + B_{k,h}(S_h^k, A_h^k) + \gamma_{k,h} + \zeta_{k,h}
\end{aligned} \tag{5.11}$$

where

$$\begin{aligned}
\gamma_{k,h} &:= \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} \rangle \\
\zeta_{k,h} &:= \langle P_h(\cdot | S_h^k, A_h^k), \tilde{\Delta}_{h+1}^k \rangle - \tilde{\Delta}_{h+1}^k(S_{h+1}^k).
\end{aligned}$$

From Theorem 5 of [Neu and Pike-Burke \(2020\)](#), many optimistic model-based algorithms can use bonuses such that $C = 0$, where C is defined in Equation (5.10). Therefore, for these algorithms, we immediately have the following:

$$(1 - C) \gamma_{k,h} \leq (1 - C) B_{k,h}(S_h^k, A_h^k)$$

where the inequality holds under the complement of the event given in Assumption 5.2. When $C = 1$, we require additional steps, as the bonuses may not be large enough to guarantee that the inequality holds with high probability. Therefore, Lemma 5.2 utilises other probabilistic arguments to control $\gamma_{k,h}$ for algorithms with $C = 1$. These arguments are just a feature of the theoretical analysis and they do not change the underlying algorithm. Therefore, they could be applied to algorithms with $C = 0$, but this would give a looser upper bound on the regret.

Lemma 5.2. Let $\gamma_{k,h} = \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} \rangle$. Then, it holds that:

$$\gamma_{k,h} \leq \frac{\tilde{\Delta}_{h+1}^k(S_{h+1}^k)}{H} + B_{k,h}(S_h^k, A_h^k) + 2\bar{\zeta}_{k,h} + \frac{(2HS + 4H^2S) \ln(S^2 AHk(k+1)/\delta)}{N'_{k,h}(S_h^k, A_h^k)}$$

where

$$\bar{\zeta}_{k,h} = \sqrt{\frac{\ln(S^2 AHk(k+1)/\delta)}{N'_{k,h}(S_h^k, A_h^k)}} \left[\left(\sum_{s'} P_h(s' | S_h^k, A_h^k) \frac{\tilde{\Delta}_{h+1}^k(s')}{\sqrt{P_h(s' | S_h^k, A_h^k)}} \right) - \frac{\tilde{\Delta}_{h+1}^k(S_{h+1}^k)}{\sqrt{P_h(S_{h+1}^k | S_h^k, A_h^k)}} \right]$$

with probability at least $1 - \delta$.

Proof. Firstly, $\gamma_{k,h}$ can be rewritten by adding and subtracting the optimal value function at the next stage:

$$\begin{aligned} \gamma_{k,h} &= \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} \rangle \\ &= \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), V_{h+1}^* \rangle + \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} - V_{h+1}^* \rangle \\ &\leq B_{k,h}(S_h^k, A_h^k) + \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} - V_{h+1}^* \rangle \end{aligned}$$

where the inequality holds with high probability by Assumption 5.2. Next, define the following quantity:

$$c_{k,h} := \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} - V_{h+1}^* \rangle$$

which is the term involved in the upper bound of $\gamma_{k,h}$. The remainder of the analysis involves controlling $c_{k,h}$ in a manner that does not influence the algorithm:

$$\begin{aligned} c_{k,h} &:= \langle \hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k), \tilde{V}_{h+1}^{\pi^k} - V_{h+1}^* \rangle \\ &\leq \langle |\hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k)|, \tilde{V}_{h+1}^{\pi^k} - V_{h+1}^* \rangle \\ &\leq \langle |\hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k)|, \tilde{V}_{h+1}^{\pi^k} - V_{h+1}^{\pi^k} \rangle && \text{(Since } V_{h+1}^* \geq V_{h+1}^{\pi^k} \text{)} \\ &= \langle |\hat{P}_{k,h}(\cdot | S_h^k, A_h^k) - P_h(\cdot | S_h^k, A_h^k)|, \tilde{\Delta}_{h+1}^k \rangle && \text{(Definition of } \tilde{\Delta}_{h+1}^k \text{)} \end{aligned}$$

Applying Lemma A.2, Bernstein's inequality, to the absolute difference between the estimated and true transition functions at each state reveals the following:

$$\mathbb{P} \left(\exists (k, h, s, a, s') : |\hat{P}_{k,h}(s' | s, a) - P_h(s' | s, a)| \geq 2\epsilon_{k,h}(s, a) \right) \leq \delta \quad (5.12)$$

where

$$\epsilon_{k,h}(s, a) := \frac{\ln(S^2 AHk(k+1)/\delta)}{N'_{k,h}(s, a)} + \sqrt{\frac{P_h(s' | s, a) \ln(S^2 AHk(k+1)/\delta)}{N'_{k,h}(s, a)}}.$$

Throughout, $L := \ln(S^2 AHk(k+1)/\delta)$ will denote the logarithmic factors introduced by the application of Bernstein's inequality.

Now, on the complement of the event inside the probability, it holds that:

$$\begin{aligned} c_{k,h} &\leq 2 \sum_{s'} \left(\frac{L}{N'_{k,h}(S_h^k, A_h^k)} + \sqrt{\frac{P_h(s' | S_h^k, A_h^k) L}{N'_{k,h}(S_h^k, A_h^k)}} \right) \tilde{\Delta}_{h+1}^k(s') \\ &\leq \frac{2HSL}{N'_{k,h}(S_h^k, A_h^k)} + 2 \sum_{s'} \sqrt{\frac{P_h(s' | S_h^k, A_h^k) L}{N'_{k,h}(S_h^k, A_h^k)}} \tilde{\Delta}_{h+1}^k(s') \end{aligned}$$

where the last inequality splits the summation and utilises the fact that $\tilde{\Delta}_{h+1}^k(s') \leq H$ due to the clipping that is present in the optimistic dynamic programming procedure of Equation (5.9). Next, it is useful to partition the state-action-stage triples of the k -th episode into those that transition into good states, which are defined as follows:

$$\mathcal{G}_{k,h} = \{s' : P_h(s' | S_h^k, A_h^k) N'_{k,h}(S_h^k, A_h^k) \geq 4H^2 \ln(S^2 AHk(k+1)/\delta)\}.$$

Notably, [Dann et al. \(2018\)](#) and [Azar et al. \(2017\)](#) apply the same idea of classifying states as either *good* or *bad* in their analyses. Now, we will proceed to handle the *good* and *bad* in a manner that is similar to [Azar et al. \(2017\)](#).

When $s' \notin \mathcal{G}_{k,h}$, $P_h(s' | S_h^k, A_h^k) \leq 4H^2 L / N'_{k,h}(S_h^k, A_h^k)$ and this can be plugged into the summation over states that are not in the set of good states to get the following:

$$\begin{aligned} \sum_{s' \notin \mathcal{G}_{k,h}} \sqrt{\frac{P_h(s' | S_h^k, A_h^k) L}{N'_{k,h}(S_h^k, A_h^k)}} \tilde{\Delta}_{h+1}^k(s') &\leq \sum_{s' \notin \mathcal{G}_{k,h}} \frac{2HL}{N'_{k,h}(S_h^k, A_h^k)} \tilde{\Delta}_{h+1}^k(s') \\ &\leq \frac{2H^2 SL}{N'_{k,h}(S_h^k, A_h^k)} \end{aligned}$$

where the last line follows from the clipping in Equation (5.9) and the fact that the value function non-negative because $r_h(s, a) \in [0, 1]$ for all $(h, s, a) \in [H] \times \mathcal{S} \times \mathcal{A}$.

When $s' \in \mathcal{G}_{k,h}$, we can add and subtract the random version of the term that we want to control

to give:

$$\begin{aligned}
\sum_{s' \in \mathcal{G}_{k,h}} \sqrt{\frac{P_h(s' | S_h^k, A_h^k) L}{N'_{k,h}(S_h^k, A_h^k)}} \tilde{\Delta}_{h+1}^k(s') &= \frac{\sqrt{L}}{\sqrt{N'_{k,h}(S_h^k, A_h^k)}} \sum_{s' \in \mathcal{G}_{k,h}} \sqrt{P_h(s' | S_h^k, A_h^k)} \tilde{\Delta}_{h+1}^k(s') \\
&= \frac{\sqrt{L}}{\sqrt{N'_{k,h}(S_h^k, A_h^k)}} \left[\left(\sum_{s' \in \mathcal{G}_{k,h}} P_h(s' | S_h^k, A_h^k) \frac{\tilde{\Delta}_{h+1}^k}{\sqrt{P_h(s' | S_h^k, A_h^k)}} \right) \pm \frac{\tilde{\Delta}_{h+1}^k(S_{h+1}^k) 1\{S_{h+1}^k \in \mathcal{G}_{k,h}\}}{\sqrt{P_h(S_{h+1}^k | S_h^k, A_h^k)}} \right] \\
&\leq \frac{\sqrt{L} 1\{S_{h+1}^k \in \mathcal{G}_{k,h}\} \tilde{\Delta}_{h+1}^k(S_{h+1}^k)}{\sqrt{P_h(S_{h+1}^k | S_h^k, A_h^k) N'_{k,h}(S_h^k, A_h^k)}} + \bar{\zeta}_{k,h}
\end{aligned}$$

where

$$\bar{\zeta}_{k,h} = \sqrt{\frac{L}{N'_{k,h}(S_h^k, A_h^k)}} \left[\left(\sum_{s' \in \mathcal{G}_{k,h}} \frac{P_h(s' | S_h^k, A_h^k) \tilde{\Delta}_{h+1}^k(s')}{\sqrt{P_h(s' | S_h^k, A_h^k)}} \right) - \frac{1\{S_{h+1}^k \in \mathcal{G}_{k,h}\} \tilde{\Delta}_{h+1}^k(S_{h+1}^k)}{\sqrt{P_h(S_{h+1}^k | S_h^k, A_h^k)}} \right].$$

Thanks to the indicator variable, we can utilise the definition of the *good* states. Specifically, $P_h(S_{h+1}^k | S_h^k, A_h^k) N'_{k,h}(S_h^k, A_h^k) \geq 4H^2L$ for $S_{h+1}^k \in \mathcal{G}_{k,h}$. Plugging this into the previous display gives:

$$\begin{aligned}
\sum_{s' \in \mathcal{G}_{k,h}} \sqrt{\frac{P_h(s' | S_h^k, A_h^k) L}{N'_{k,h}(S_h^k, A_h^k)}} \tilde{\Delta}_{h+1}^k(s') &\leq \frac{\tilde{\Delta}_{h+1}^k(S_{h+1}^k) 1\{S_{h+1}^k \in \mathcal{G}_{k,h}\}}{2H} + \bar{\zeta}_{k,h} \\
&\leq \frac{\tilde{\Delta}_{h+1}^k(S_{h+1}^k)}{2H} + \bar{\zeta}_{k,h}
\end{aligned}$$

where the last inequality follows from the fact that the optimistic value function upper bounds the true value function by optimism. Bringing everything together gives:

$$\gamma_{k,h} \leq \frac{\tilde{\Delta}_{h+1}^k(S_{h+1}^k)}{H} + B_{k,h}(S_h^k, A_h^k) + 2\bar{\zeta}_{k,h} + \frac{(2HS + 4H^2S) \ln(S^2 AHk(k+1)/\delta)}{N'_{k,h}(S_h^k, A_h^k)}$$

as required. $\square$

Combining Equation (5.11) and Lemma 5.2 allows the difference between the optimistic state value and the actual state value of the policy executed in the k -th episode to be upper bounded as follows:

$$\begin{aligned}
\tilde{\Delta}_h^k(S_h^k) &\leq \left(1 + \frac{C}{H}\right) \tilde{\Delta}_{h+1}^k(S_{h+1}^k) + \zeta_{k,h} + 2C\bar{\zeta}_{k,h} + 2B_{k,h}(S_h^k, A_h^k) + \frac{(2HS + 4H^2S) CL}{N'_{k,h}(S_h^k, A_h^k)} \\
&\leq \left(1 + \frac{C}{H}\right) \tilde{\Delta}_{h+1}^k(S_{h+1}^k) + \zeta_{k,h} + 2C\bar{\zeta}_{k,h} + 2B_{k,h}(S_h^k, A_h^k) + \frac{6CH^2SL}{N'_{k,h}(S_h^k, A_h^k)}
\end{aligned}$$

where $L = \ln(S^2 AHk(k+1)/\delta)$ and the inequality holds on the complement of the events in Assumption 5.2 and Equation (5.12) of Lemma 5.2. Starting at $h = H$ and unravelling the recursive relationship by repeatedly substituting in the upper bound above gives the following:

$$\begin{aligned} \tilde{\Delta}_1^k(S_1^k) &\leq \left(1 + \frac{C}{H}\right)^H \left(\sum_{h=1}^H \zeta_{k,h} + 2C\bar{\zeta}_{k,h} + 2B_{k,h}(S_h^k, A_h^k) + \frac{6CH^2SL}{N'_{k,h}(S_h^k, A_h^k)} \right) \\ &\leq \exp(1) \left(\sum_{h=1}^H \zeta_{k,h} + 2C\bar{\zeta}_{k,h} + 2B_{k,h}(S_h^k, A_h^k) + \frac{6CH^2SL}{N'_{k,h}(S_h^k, A_h^k)} \right) \\ &< 3 \left(\sum_{h=1}^H \zeta_{k,h} + 2C\bar{\zeta}_{k,h} + 2B_{k,h}(S_h^k, A_h^k) + \frac{6CH^2SL}{N'_{k,h}(S_h^k, A_h^k)} \right) \end{aligned} \quad (5.13)$$

which is the expression that will be used to prove Lemma 5.1.

5.3.2 Proof of Lemma 5.1

Proof. From Assumption 5.2, the optimistic value of the policy executed in each episode upper bounds the optimal value with high probability. Therefore, it follows that:

$$\begin{aligned} \Delta_1^k(S_1^k) &= V_1^*(S_1^k) - V_1^{\pi^k}(S_1^k) \\ &\leq \tilde{V}_1^{\pi^k}(S_1^k) - V_1^{\pi^k}(S_1^k) \\ &\leq 3 \left(\sum_{h=1}^H \zeta_{k,h} + 2C\bar{\zeta}_{k,h} + 2B_{k,h}(S_h^k, A_h^k) + \frac{6CH^2SL}{N'_{k,h}(S_h^k, A_h^k)} \right) \end{aligned} \quad (\text{Equation (5.13)})$$

where $L = \ln(S^2 AHk(k+1)/\delta)$ and the inequality holds on the complement of the events from Assumption 5.2 or Equation (5.12) of Lemma 5.2 occurring. From Equation (5.3), the definition of the pseudo-regret, and the previous display, it follows that:

$$\hat{R}_K(\pi, M) \leq 3 \sum_{k=1}^K \sum_{h=1}^H \zeta_{k,h} + 6C \sum_{k=1}^K \sum_{h=1}^H \bar{\zeta}_{k,h} + 6 \sum_{k=1}^K \sum_{h=1}^H \left(B_{k,h}(S_h^k, A_h^k) + \frac{CH^2SL}{N'_{k,h}(S_h^k, A_h^k)} \right)$$

where

$$\begin{aligned} \zeta_{k,h} &= \langle P_h(\cdot | S_h^k, A_h^k), \tilde{\Delta}_{h+1}^k \rangle - \tilde{\Delta}_{h+1}^k(S_{h+1}^k) \\ \bar{\zeta}_{k,h} &= \sqrt{\frac{L}{N'_{k,h}(S_h^k, A_h^k)}} \left[\left(\sum_{s' \in \mathcal{G}_{k,h}} P_h(s' | S_h^k, A_h^k) \frac{\tilde{\Delta}_{h+1}^k(s')}{\sqrt{P_h(s' | S_h^k, A_h^k)}} \right) - \frac{1\{S_{h+1}^k \in \mathcal{G}_{k,h}\} \tilde{\Delta}_{h+1}^k(S_{h+1}^k)}{\sqrt{P_h(S_{h+1}^k | S_h^k, A_h^k)}} \right] \\ \mathcal{G}_{k,h} &= \{s' : P_h(s' | s_h^k, a_h^k) N'_{k,h}(S_h^k, A_h^k) \geq 4H^2L\}. \end{aligned}$$

Notably, $|\zeta_{k,h}| \leq 2H$ and $|\bar{\zeta}_{k,h}| \leq 1$, and their expectations conditional on the observed information and the current state-action-stage are equal to zero (Azar et al., 2017):

$$\mathbb{E} [\zeta_{k,h} \mid (S_h^i, A_h^i, R_h^i, S_{h+1}^i)_{i:i+\tau_i < k} \cup (S_h^k, A_h^k)] = 0$$

$$\mathbb{E} [\bar{\zeta}_{k,h} \mid (S_h^i, A_h^i, R_h^i, S_{h+1}^i)_{i:i+\tau_i < k} \cup (S_h^k, A_h^k)] = 0.$$

which follows from the fact that each term is an expected value of the random variable at the next stage minus the value of the random variable at the next stage. Therefore, $\zeta_{k,h}$ and $\bar{\zeta}_{k,h}$ are martingale difference sequences. The Azuma-Hoeffding inequality can be applied to control these kinds of random variables (Azar et al., 2017; Neu and Pike-Burke, 2020):

$$\mathbb{P} \left(\exists K \in \mathbb{N} : \sum_{k=1}^K \sum_{h=1}^H \zeta_{k,h} \geq H \sqrt{2KH \ln \left(\frac{K(K+1)}{\delta} \right)} \right) \leq \delta \quad (5.14)$$

$$\mathbb{P} \left(\exists K \in \mathbb{N} : \sum_{k=1}^K \sum_{h=1}^H \bar{\zeta}_{k,h} \geq \sqrt{KH \ln \left(\frac{K(K+1)}{\delta} \right)} \right) \leq \delta \quad (5.15)$$

Therefore, it follows that the pseudo-regret can be upper bounded by

$$\hat{R}_K(\pi, M) \leq 6(H+C) \sqrt{KH \ln \left(\frac{K(K+1)}{\delta} \right)} + 6 \sum_{k=1}^K \sum_{h=1}^H \left(B_{k,h}(S_h^k, A_h^k) + \frac{CH^2SL}{N'_{k,h}(s_h^k, a_h^k)} \right)$$

on the complement of the union of the events defined in Assumption 5.2, and Equations (5.12), (5.14) and (5.15). The union of these events occurs with probability at most 4δ . Therefore, the upper bound on the pseudo-regret holds with probability at least $1 - 4\delta$, as required. $\square$

5.4 Active Updating

Chapter 4 highlighted that utilising all of the available information is an effective strategy in the generalised linear bandit problem with delayed feedback. Therefore, the same strategy is worth investigating in the more complex setting of episodic reinforcement learning. Algorithm 4 presents this approach and is called *active updating*. The meta-algorithm is quite straightforward. First, it is given an immediate feedback algorithm as input, which we will refer to as **Base**. **Base** takes in trajectories as input and outputs a policy. Then, Algorithm 4 uses **Base** to update the policy as soon as new feedback arrives. Notably, **Base** is assumed to return the same arbitrary policy when presented with an empty set of trajectories.

Algorithm 4 Active Updating

Input. Base (an immediate feedback algorithm).
Initialise. $\mathcal{X} = \emptyset$ (an empty dataset).
for $k = 1$ **to** K **do**
 Provide $\pi_k = \text{Base}(\mathcal{X})$ to an agent to sample an episode.
 Update $\mathcal{X} = \mathcal{X} \cup \{(S_h^i, A_h^i, R_h^i, S_{h+1}^i)_{h=1}^H : i + \tau_i = k\}$ with newly observed trajectories.
end for

Theorem 5.1 provides a high probability upper bound on the pseudo-regret for the class of optimistic model-based algorithms when they are paired with active updating.

Theorem 5.1 (Active Updating). *Under Assumptions 5.2 and 5.1, with at least probability $1 - 5\delta$, the regret of any model-based algorithm under delayed feedback with active updating is upper bounded by:*

$$\hat{R}_K(\pi, M) \lesssim \beta H \sqrt{SAK} + \max\{\beta, \beta^{(2)}, CH^2S\} HSAE[\tau]$$

where $\lesssim$ suppresses numeric constants and logarithmic factors, and $\beta \geq \beta_{k,h}^{(1)}$ is a uniform upper bound on the leading-order term in the numerator of the exploration bonus.

Proof. From Lemma 5.1, it follows that bounding the summation of the exploration bonuses will control the pseudo-regret. However, the exploration bonuses will depend on the observed visitation counters. Therefore, the standard arguments no longer apply. Section 5.4.1 presents the tools required to control the bonuses of Equation (5.8). See Section 5.4.2 for full proof of the claim. $\square$

5.4.1 Preliminary Results for Theorem 5.1

Lemma 5.3 and 5.4 develop the technical tools to control the exploration bonuses, which depend on the observed number of visits to each state-action-stage.

Lemma 5.3. *Let $Z^p = \sum_{n=0}^N 1/(1 \vee n)^p$. Then, $Z^{1/2}$ and Z^1 have the following upper bounds:*

$$Z_n^p \leq \begin{cases} 2\sqrt{N} & \text{if } p = \frac{1}{2} \\ \log(8N) & \text{if } p = 1 \end{cases}$$

Proof. Removing the first two terms from the summation and upper bounding the remaining

terms by an integral gives:

$$Z^p = 2 + \sum_{n=2}^N \frac{1}{n^p} \leq 2 + \int_1^N \frac{1}{n^p} dn \leq 2 + \begin{cases} 2\sqrt{N} - 2 & \text{if } p = \frac{1}{2} \\ \log(N) & \text{if } p = 1 \end{cases} \leq \begin{cases} 2\sqrt{N} & \text{if } p = \frac{1}{2} \\ \log(8N) & \text{if } p = 1 \end{cases}$$

as required. $\square$

Lemma 5.4. Let $Z_K^p = \sum_{k=1}^K \sum_{h=1}^H 1/(N'_{k,h}(S_h^k, A_h^k))^p$. Under Assumption 5.1, it holds that:

$$Z_T^p \leq \begin{cases} 4H\sqrt{SAK} + 2\sqrt{2}HSA\psi_K & \text{if } p = \frac{1}{2} \\ 2HSA \log\left(\frac{8K}{SA}\right) + HSA\psi_K \log(16\psi_K) & \text{if } p = 1 \end{cases}$$

where

$$\psi_K = \mathbb{E}[\tau] + \ln\left(\frac{K(K+1)}{\delta}\right) + \sqrt{2\mathbb{E}[\tau] \ln\left(\frac{K(K+1)}{\delta}\right)}$$

with probability at least $1 - \delta$.

Proof. Throughout, $N_{k,h}(s, a) = 1 \vee N_{k,h}(s, a)$ and $N'_{k,h}(s, a) = 1 \vee N'_{k,h}(s, a)$ for notational convenience. Next, Equation (5.6) is used to partition the summation into terms that do and do not depend on the delays:

$$\begin{aligned} Z_K^p &= \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1}{N'_{k,h}(S_h^k, A_h^k)} \right)^p = \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1}{N_{k,h}(S_h^k, A_h^k)} \right)^p \left(\frac{N_{k,h}(S_h^k, A_h^k)}{N'_{k,h}(S_h^k, A_h^k)} \right)^p \\ &= \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1}{N_{k,h}(S_h^k, A_h^k)} \right)^p \left(\frac{N'_{k,h}(S_h^k, A_h^k) + N''_{k,h}(S_h^k, A_h^k)}{N'_{k,h}(S_h^k, A_h^k)} \right)^p \\ &= \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1}{N_{k,h}(S_h^k, A_h^k)} \right)^p \left(1 + \frac{N''_{k,h}(S_h^k, A_h^k)}{N'_{k,h}(S_h^k, A_h^k)} \right)^p \\ &\leq \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1}{N_{k,h}(S_h^k, A_h^k)} \right)^p \left(1 + \left(\frac{N''_{k,h}(S_h^k, A_h^k)}{N'_{k,h}(S_h^k, A_h^k)} \right)^p \right) \\ &= \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1}{N_{k,h}(S_h^k, A_h^k)} \right)^p \tag{5.16} \end{aligned}$$

$$+ \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1}{N_{k,h}(S_h^k, A_h^k)} \frac{N''_{k,h}(S_h^k, A_h^k)}{N'_{k,h}(S_h^k, A_h^k)} \right)^p \tag{5.17}$$

where $(1+x)^p \leq 1+x^p$ for $x \geq 0$ and $p \in \{1/2, 1\}$ was applied to get the inequality.

Controlling Equation (5.16) is straightforward and applies standard arguments:

$$\begin{aligned}
(5.16) &= \sum_{s,a,h} \sum_{k=1}^K 1\{S_h^k = s, A_h^k = a\} \left(\frac{1}{N_{k,h}(s,a)} \right)^p \\
&= \sum_{s,a,h} \sum_{n=1}^{N_{K,h}(s,a)} \left(\frac{1}{n} \right)^p \\
&\leq \begin{cases} \sum_{s,a,h} 2\sqrt{N_{K,h}(s,a)} & \text{if } p = 1/2 \\ \sum_{s,a,h} \ln(8N_{K,h}(s,a)) & \text{if } p = 1 \end{cases} \quad (\text{Lemma 5.3}) \\
&\leq \begin{cases} 2\sqrt{HSA \sum_{s,a,h} N_{K,h}(s,a)} & \text{if } p = 1/2 \\ HSA \ln \left(\frac{8 \sum_{s,a,h} N_{K,h}(s,a)}{HSA} \right) & \text{if } p = 1 \end{cases} \quad (\text{Cauchy-Schwarz/Jensen's}) \\
&= \begin{cases} 2H\sqrt{SAK} & \text{if } p = 1/2 \\ HSA \ln \left(\frac{8K}{SA} \right) & \text{if } p = 1 \end{cases} \quad (\text{Since } \sum_{s,a,h} N_{K,h}(s,a) = KH)
\end{aligned}$$

Controlling Equation (5.17) requires more care because it involves the observed and missing visitation counters. The trick to controlling this term is to first upper bound the missing visitation counter:

$$N''_{k,h}(s,a) = \sum_{i: i+\tau_i \geq k} 1\{S_h^i = a, A_h^i = a\} \leq \sum_{i=1}^k 1\{i + \tau_i \geq k\} := G_k$$

where G_k counts the number of episodes with missing feedback. From Chapter 4, Lemma 4.9 applies Assumption 5.1 to show that:

$$\mathbb{P} \left(\exists k \in \mathbb{N} : G_k \geq \mathbb{E}[\tau] + \ln \left(\frac{k(k+1)}{\delta} \right) + \sqrt{2\mathbb{E}[\tau] \ln \left(\frac{k(k+1)}{\delta} \right)} \right) \leq \delta.$$

Therefore, with probability at least $1 - \delta$, it follows that:

$$N''_{k,h}(s,a) \leq \mathbb{E}[\tau] + \ln \left(\frac{k(k+1)}{\delta} \right) + \sqrt{2\mathbb{E}[\tau] \ln \left(\frac{k(k+1)}{\delta} \right)} := \psi_k \leq \psi_K$$

for all $k \in \mathbb{N}$. Rewriting Equation (5.17) with the indicator that the observed visitation counter exceeds the upper bound on the number of missing episodes gives:

$$(5.17) \leq \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1\{N'_{k,h}(S_h^k, A_h^k) \geq \psi_K\} \psi_K}{N_{k,h}(S_h^k, A_h^k) N'_{k,h}(S_h^k, A_h^k)} \right)^p + \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1\{N'_{k,h}(S_h^k, A_h^k) \leq \psi_K\} \psi_K}{N_{k,h}(S_h^k, A_h^k) N'_{k,h}(S_h^k, A_h^k)} \right)^p$$

$$\begin{aligned}
&\leq \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1 \{N'_{k,h}(S_h^k, A_h^k) \geq \psi_K\}}{N_{k,h}(S_h^k, A_h^k)} \right)^p + \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1 \{N'_{k,h}(S_h^k, A_h^k) \leq \psi_K\} \psi_K}{N_{k,h}(S_h^k, A_h^k) N'_{kh}(S_h^k, A_h^k)} \right)^p \\
&\leq \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1}{N_{k,h}(S_h^k, A_h^k)} \right)^p + \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1 \{N'_{k,h}(S_h^k, A_h^k) \leq \psi_K\} \psi_K}{N_{k,h}(S_h^k, A_h^k) N'_{kh}(S_h^k, A_h^k)} \right)^p \\
&= (5.16) + \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1 \{N_{k,h}(S_h^k, A_h^k) \leq \psi_K + N''_{k,h}(S_h^k, A_h^k)\} \psi_K}{N_{k,h}(S_h^k, A_h^k)} \right)^p \quad (\star) \\
&\leq (5.16) + \sum_{k=1}^K \sum_{h=1}^H \left(\frac{1 \{N_{k,h}(S_h^k, A_h^k) \leq 2\psi_K\} \psi_K}{N_{k,h}(S_h^k, A_h^k)} \right)^p \quad (\text{Since } N''_{k,h} \leq \psi_K) \\
&\leq (5.16) + \psi_K \sum_{s,a,h} \sum_{n=1}^{2\psi_K} \left(\frac{1}{n} \right)^p \\
&\leq \begin{cases} 2H\sqrt{SAK} + 2\sqrt{2}HSA\psi_K & \text{if } p = 1/2 \\ HSA \ln(\frac{8K}{SA}) + \psi_K HSA \ln(16\psi_K) & \text{if } p = 1. \end{cases}
\end{aligned}$$

where $(\star)$ uses $N'_{k,h}(s, a) = N_{k,h}(s, a) - N''_{k,h}(s, a)$ to rewrite the term inside the indicator and the last inequality applies Lemma 5.3. Adding the upper bounds on Equations (5.16) and (5.17) completes the proof. $\square$

5.4.2 Proof of Theorem 5.1

Proof. From Lemma 5.1, the pseudo-regret of the optimistic model-based algorithm provided as input can be upper bounded as follows:

$$\begin{aligned}
R_K(\pi, M) &\leq 6(H + C) \sqrt{KH \log \left(\frac{K\pi}{6\delta'} \right)} + 6 \sum_{k=1}^K \sum_{h=1}^H B_{k,h}(s_h^k, a_h^k) + 6 \sum_{k=1}^K \sum_{h=1}^H \frac{CH^2SL}{N'_{kh}(s_h^k, a_h^k)} \\
&= 6(H + C) \sqrt{KH \log \left(\frac{K\pi}{6\delta'} \right)} + 6 \sum_{k=1}^K \sum_{h=1}^H \frac{\beta_{k,h}^{(1)}}{\sqrt{N'_{k,h}(s, a)}} + 6 \sum_{k=1}^K \sum_{h=1}^H \frac{\beta_{k,h}^{(2)} + CH^2SL}{N'_{k,h}(s, a)}
\end{aligned}$$

with probability at least $1 - 4\delta$. For many algorithms, $\beta_{k,h}^{(1)}$ depends polynomially on quantities related to the environment, such as H and S . For such algorithms, a direct application of Lemma 5.4 is able to separate the expected delay from the number of episodes. This is in line with the intuition that the impact of delay becomes negligible once the algorithm has a reasonable model of the environment. However, for other algorithms (Fruit et al., 2020; Neu and Pike-Burke, 2020; Azar et al., 2017):

$$\beta_{k,h}^{(1)} \approx \sqrt{\mathbb{V}_{s' \sim \hat{P}_{kh}(\cdot | s, a)} \left(\tilde{V}_{h+1}^\pi(s') \right)}$$

Typically, Cauchy-Schwarz is applied to separate the terms involving the variance from those involving the counters, which gives (Fruit et al., 2020; Neu and Pike-Burke, 2020; Azar et al., 2017):

$$\begin{aligned} \sum_{k,h} \sqrt{\frac{\text{Var}_{s' \sim \hat{P}_h}(\tilde{V}_{h+1}(s'))}{N'_{kh}(s_h^k, a_h^k)}} &\leq \sqrt{\sum_{k,h} \text{Var}_{s' \sim \hat{P}_h}(\tilde{V}_{h+1}(s'))} \sum_{k,h} \frac{1}{N'_{kh}(s_h^k, a_h^k)} \\ &\lesssim \sqrt{H^2 K \sum_{k,h} \frac{1}{N'_{kh}(s_h^k, a_h^k)}} \end{aligned}$$

Lemma 5.4 shows that applying this technique would lead to the delays multiplying the number of episodes. Fortunately, $\tilde{V}_h(s') \in [0, H]$ almost surely, thanks to the clipping that is present in the dynamic programming procedure of Equation (5.9). Therefore, Popoviciu's inequality can be applied to upper bound the variance term by setting:

$$\beta = \max\{\beta_{k,h}^{(1)}, H/2\}$$

which avoids the expected delay scaling the number of episodes for algorithms that have empirical variances in their exploration bonuses. Applying this uniform upper bound to the summation of the terms involving the observed visitation counters gives:

$$\begin{aligned} \sum_{k=1}^K \sum_{h=1}^H \frac{\beta_{k,h}^{(1)}}{\sqrt{N'_{kh}(s_h^k, a_h^k)}} + \frac{\beta^{(2)} + CH^2 SL}{N'_{kh}(s_h^k, a_h^k)} &\leq \sum_{k=1}^K \sum_{h=1}^H \frac{\beta}{\sqrt{N'_{kh}(s_h^k, a_h^k)}} + \frac{\beta^{(2)} + CH^2 SL}{N'_{kh}(s_h^k, a_h^k)} \\ &\leq \sum_{k=1}^K \sum_{h=1}^H \frac{\beta}{\sqrt{N'_{kh}(s_h^k, a_h^k)}} + \frac{2 \max\{\beta^{(2)}, CH^2 SL\}}{N'_{kh}(s_h^k, a_h^k)} \\ &\leq 4\beta H \sqrt{SAK} + 2\sqrt{2}\beta HSA\psi_K + 2HSA \max\{\beta^{(2)}, CH^2 SL\} \left(2 \ln \left(\frac{K}{SA} \right) + \psi_K \ln(16\psi_K) \right) \end{aligned}$$

where

$$\psi_K = \mathbb{E}[\tau] + \ln \left(\frac{K(K+1)}{\delta} \right) + \sqrt{2\mathbb{E}[\tau] \ln \left(\frac{K(K+1)}{\delta} \right)}$$

and final inequality holds with probability at least $1 - \delta$ and follows from an application of Lemma 5.4. Therefore, omitting numeric constants and logarithmic factors, the pseudo-regret is upper bounded by:

$$\hat{R}_K(\pi, M) \lesssim (H + C) \sqrt{KH} + \beta H \sqrt{SAK} + \max\{\beta, \beta^{(2)}, CH^2 S\} HSA \mathbb{E}[\tau]$$

which holds with probability at least $1 - 5\delta$ in totality. $\square$

5.5 Lazy Updating

Theorem 5.1 is somewhat cumbersome to apply in practice, as it requires substituting various algorithm-dependent quantities to obtain a high-probability upper bound on the pseudo-regret. Additionally, for algorithms that use empirical variances in their exploration bonuses, it was necessary to apply a uniform upper bound to ensure an additive dependence on the expected delay. However, this comes at the expense of degrading the leading-order term in the guarantee by a factor of $\sqrt{H}$. This introduces looseness into the bound for the algorithms that are expected to perform the best (Azar et al., 2017).

Therefore, Algorithm 5 takes an alternative approach, which we call *lazy updating*, to address these limitations. The idea behind lazy updating is to wait until there is enough additional information before performing an update to the policy. Intuitively, grouping the episodes into batches where each agent is given an identical policy ensures that the updates are based on a more stable and representative set of observations. Further, by carefully defining an updating rule, we will show how to handle the exploration bonuses and the delays separately, while only paying an additional logarithmic factor in the guarantee.

Algorithm 5 Lazy Updating

Input. *Base* (an immediate feedback algorithm) and $\alpha \geq 1$ (an activity hyperparameter).
Initialise. $\mathcal{X} = \emptyset$ (an empty dataset), $j = 1$ and $k_j = 1$ (epoch counter and corresponding episode, respectively).
 Compute $\pi_1 = \text{Base}(\mathcal{X})$
for $k = 1$ **to** K **do**
 Provide π_k to an agent to sample an episode.
 Update $\mathcal{X} = \mathcal{X} \cup \{(S_h^i, A_h^i, R_h^i, S_{h+1}^i)_{h=1}^H : i + \tau_i = k\}$
 Compute $N'_{k,h}(s, a)$ via Equation (5.4).
 if $\exists (s, a, h) : N'_{k,h}(s, a) \geq (1 + 1/\alpha)N'_{k_j,h}(s, a)$ **then**
 Update $j = j + 1$ and $k_j = k + 1$
 Compute $\pi_{k+1} = \text{Base}(\mathcal{X})$
 else
 Set $\pi_{k+1} = \pi_k$
 end if
end for

Algorithm 5 presents the pseudo-code for *lazy updating*, which works in batches of episodes that will be referred to as epochs: $j = 1, 2, \dots, J$. At the start of the j -th epoch, lazy updating uses the base algorithm to compute a policy using all the available information. The meta-algorithm uses this policy in every episode until the next epoch begins. Therefore, each epoch is just a set of episodes where the lazy updating algorithm uses the same policy.

Epoch $j + 1$ begins as soon as there is an (s, a, h) whose observed visitation counter reaches $1 + 1/\alpha$ times the observed visits at the start of the j -th epoch, where $\alpha \in [1, \infty)$. Setting $\alpha = 1$ will wait until there exists a state-action-stage triple whose observed visitation count doubles. Setting $\alpha > 1$ will cause the algorithm to update more frequently. This is similar to, but not the same as, the doubling trick (Besson and Kaufmann, 2018). The difference is that *lazy updating* does not discard information at the end of an epoch, whereas the doubling trick performs a full reset.

Once the observed visitation counter triggers this condition, a new epoch begins, and lazy updating will use **Base** to update the policy. Formally, epoch $j + 1$ begins in episode k_{j+1} , which occurs when:

$$\begin{aligned} k_{j+1} &= \arg \min_{k > k_j} \left\{ \exists (s, a, h) : N'_{k,h}(s, a) \geq \left(1 + \frac{1}{\alpha}\right) N'_{k_j,h}(s, a) \right\} \\ &= \arg \min_{k > k_j} \left\{ \exists (s, a, h) : n_{k_j,h}^k(s, a) \geq \frac{1}{\alpha} N'_{k_j,h}(s, a) \right\} \end{aligned} \quad (5.18)$$

where

$$n_{k,h}^l(s, a) = \sum_{i=k}^{l-1} 1 \{ (s_h^i = s, a_h^i = a), i + \tau_i \leq l \} \quad (5.19)$$

counts the observed number of visits between episodes k and l for $l > k$. Intuitively, this updating scheme forces the number of samples needed for any particular (s, a, h) to trigger an update to increase exponentially quickly. Therefore, the total number of epochs should grow logarithmically in K . Theorem 5.2 utilises this fact to establish a high probability upper bound on the pseudo-regret of any optimistic model-based algorithm.

Theorem 5.2 (Lazy Updating). *Let $K \geq SA$ and $\alpha \geq 1$. Under Assumption 5.2 and 5.1, with probability at least $1 - 5\delta$, the regret of any model-based algorithm under delayed feedback is upper bounded by:*

$$\hat{R}_K(\pi, M) \lesssim \left(1 + \frac{1}{\alpha}\right) \tilde{R}_K(\text{Base}, M) + \frac{H^2 S A \mathbb{E}[\tau]}{\log(1 + \frac{1}{\alpha})}$$

where $\lesssim$ suppresses numeric constants and logarithmic factors that are independent of Algorithm 5, and $\tilde{R}_K(\text{Base}, M)$ denotes an upper bound on the regret of the chosen base algorithm under immediate feedback.

5.5.1 Preliminary Results for Theorem 5.2

Proving Theorem 5.2 relies on the properties introduced by the updating scheme. In this subsection, we will state and prove these properties. First, we seek to establish an upper bound on the number of epochs. For this, we will need the following result.

Lemma 5.5. *Let $\alpha \in [1, \infty)$. Then*

$$\sum_{i=0}^n \left(1 + \frac{1}{\alpha}\right)^i \geq \left(1 + \frac{1}{\alpha}\right)^{n+1} - \frac{1}{\alpha}$$

Proof. Trivially, the statement is true for $n = 0$, because $(1 + 1/\alpha)^0 = 1$ and $(1 + 1/\alpha)^1 - 1/\alpha = 1$. Thus, the remainder of the proof will proceed by induction. Suppose that the inequality holds for some $n \in \mathbb{N}$:

$$\sum_{i=0}^n \left(1 + \frac{1}{\alpha}\right)^i \geq \left(1 + \frac{1}{\alpha}\right)^{n+1} - \frac{1}{\alpha}$$

Then, it holds that:

$$\begin{aligned} \sum_{i=0}^{n+1} \left(1 + \frac{1}{\alpha}\right)^i &= \left(1 + \frac{1}{\alpha}\right)^{n+1} + \sum_{i=0}^n \left(1 + \frac{1}{\alpha}\right)^i \\ &\geq \left(1 + \frac{1}{\alpha}\right)^{n+1} + \left(1 + \frac{1}{\alpha}\right)^{n+1} - \frac{1}{\alpha} \\ &= 2 \left(1 + \frac{1}{\alpha}\right)^{n+1} - \frac{1}{\alpha} \\ &\geq \left(1 + \frac{1}{\alpha}\right) \left(1 + \frac{1}{\alpha}\right)^{n+1} - \frac{1}{\alpha} && \text{(Since } 2 \geq 1 + 1/\alpha\text{)} \\ &= \left(1 + \frac{1}{\alpha}\right)^{n+2} - \frac{1}{\alpha} \end{aligned}$$

which proves the claim. □

Lemma 5.6 uses Lemma 5.5 to prove that the total number of epochs is logarithmic in the number of episodes. This will be leveraged within the theoretical analysis to control the episodes with long delays.

Lemma 5.6. *Algorithm 5 ensures that the number of epochs has the following upper bound:*

$$J \leq \frac{HSA \log\left(\frac{\alpha K}{SA} + 1\right)}{\log\left(1 + \frac{1}{\alpha}\right)}$$

for all $K \geq SA$ and $\alpha \geq 1$.

Proof. From Equation (5.18), a direct computation allows the observed visitation counter at the start of epoch $j + 1$ to be related to the sum of the visits within each of the previous epochs:

$$\begin{aligned}
N'_{k_{j+1},h}(s,a) &= \sum_{i=1}^{k_{j+1}-1} 1 \{ (S_h^i, A_h^i) = (s,a), i + \tau_i < k_{j+1} \} \\
&= \sum_{l=1}^j \sum_{i=k_l}^{k_{l+1}-1} 1 \{ (S_h^i, A_h^i) = (s,a), i + \tau_i < k_{l+1} \} \\
&= \sum_{l=1}^j \sum_{i=1}^{k_{j+1}} 1 \{ (S_h^i, A_h^i) = (s,a), k_l \leq i + \tau_i < k_{l+1} \} \\
&= \sum_{l=1}^j n_{k_l,h}^{k_{l+1}}(s,a)
\end{aligned}$$

where the second equality follows from the fact that an epoch is a disjoint set of episodes and the final equality follows from the definition of the between episodes visitation counter. From the above, it is easy to see that:

$$N'_{k_{j+1},h}(s,a) = n_{k_j,h}^{k_{j+1}}(s,a) + \sum_{l=1}^{j-1} n_{k_l,h}^{k_{l+1}}(s,a,h) = n_{k_j,h}^{k_{j+1}}(s,a) + N'_{k_j,h}(s,a,h)$$

Thus, Equation (5.18) can be rewritten as follows:

$$\begin{aligned}
k_{j+1} &= \arg \min_{k > k_j} \left\{ \exists s, a, h : N'_{k,h}(s,a) \geq \left(1 + \frac{1}{\alpha}\right) N'_{k_j,h}(s,a) \right\} \\
&= \arg \min_{k > k_j} \left\{ \exists s, a, h : N'_{k,h}(s,a) - N'_{k_j,h}(s,a) \geq \frac{1}{\alpha} N'_{k_j,h}(s,a) \right\} \\
&= \arg \min_{k > k_j} \left\{ \exists s, a, h : n_{k_j,h}^{k_{j+1}}(s,a) \geq \frac{1}{\alpha} N'_{k_j,h}(s,a) \right\}
\end{aligned}$$

whenever the state-action-stage has been seen once. The case for epochs where the observed visitation count is zero is handled at the end of the proof. Therefore, at the end of each epoch, there is a state-action-stage triple such that $n_{k_j,h}^{k_{j+1}}(s,a) \geq N'_{k_j,h}(s,a)/\alpha$.

Suppose $N'_{(K+1),h}(s,a) > 0$ for a fixed $(s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]$. Define $J(s,a,h)$ as the number of epochs with $n_{k_j,h}^{k_{j+1}}(s,a) \geq N'_{k_j,h}(s,a)/\alpha$. Or, equivalently, the number of epochs with $N_{k_{j+1},h}(s,a) \geq (1 + 1/\alpha)N'_{k_j,h}(s,a)$. Then, we have that:

$$N'_{K+1,h}(s,a) = \sum_{j=1}^J n_{k_j,h}^{k_{j+1}}(s,a)$$

$$\begin{aligned}
&\geq 1 + \sum_{j: n_{k_j, h}^{k_j+1}(s, a) \geq N'_{k_j, h}(s, a)/\alpha} n_{k_j, h}^{k_j+1}(s, a) \\
&\geq 1 + \frac{1}{\alpha} \sum_{j: n_{k_j, h}^{k_j+1}(s, a) \geq N'_{k_j, h}(s, a)/\alpha} N'_{k_j, h}(s, a) \\
&\geq 1 + \frac{1}{\alpha} \sum_{j=1}^{J(s, a, h)} \left(1 + \frac{1}{\alpha}\right)^j
\end{aligned}$$

The first inequality follows from focusing only on the epochs where we update due to (s, a, h) and the $+1$ accounts for the first update due to observing the given state-action-step triple. The second inequality follows from the condition in the subscript of the summation, which is over the epochs where the update was triggered by (s, a, h) . The final inequality follows from Equation (5.18) and because we update $J(s, a, h)$ times due to (s, a, h) . Since $\alpha \in [1, \infty)$, Lemma 5.5 applies. Rearranging terms reveals that:

$$\sum_{j=1}^{J(s, a, h)} \left(1 + \frac{1}{\alpha}\right)^j \geq \left(1 + \frac{1}{\alpha}\right)^{J(s, a, h)+1} - \left(1 + \frac{1}{\alpha}\right)$$

Therefore, for $N'_{K+1}(s, a, h) > 0$:

$$\begin{aligned}
N'_{K+1}(s, a, h) &\geq 1 - \frac{1}{\alpha} \left(1 + \frac{1}{\alpha}\right) + \frac{1}{\alpha} \left(1 + \frac{1}{\alpha}\right)^{J(s, a, h)+1} \\
&> \frac{1}{\alpha} \left(1 + \frac{1}{\alpha}\right)^{J(s, a, h)+1} - \frac{1}{\alpha} \left(1 + \frac{1}{\alpha}\right).
\end{aligned}$$

When $N'_{K+1, h}(s, a) = 0$ it follows that this state-action-stage triple never triggers the update rule. Therefore, $J(s, a, h) = 0$ too. Notably, we can modify the expression we had for case where the observed number of visits is positive to include the case where we never observe any visits:

$$N'_{K+1, h}(s, a) = \frac{1}{\alpha} \left(1 + \frac{1}{\alpha}\right)^{J(s, a, h)+1} - \frac{1}{\alpha} \left(1 + \frac{1}{\alpha}\right) = 0.$$

Thus, for all possible values of the observed visitation counter, the following holds:

$$N'_{K+1}(s, a, h) \geq \frac{1}{\alpha} \left(1 + \frac{1}{\alpha}\right)^{J(s, a, h)+1} - \frac{1}{\alpha} \left(1 + \frac{1}{\alpha}\right). \quad (5.20)$$

where the inequality is to include cases where we never observe a visit. Using the above inequality:

$$KH = \sum_{s, a, h} N_{K+1, h}(s, a)$$

$$\begin{aligned}
&\geq \sum_{s,a,h} N'_{K+1,h}(s,a) \\
&\geq \sum_{s,a,h} \left(\frac{1}{\alpha} \left(1 + \frac{1}{\alpha} \right)^{J(s,a,h)+1} - \frac{1}{\alpha} \left(1 + \frac{1}{\alpha} \right) \right) \quad (\text{Equation (5.20)}) \\
&= -\frac{HSA}{\alpha} \left(1 + \frac{1}{\alpha} \right) + \sum_{s,a,h} \frac{1}{\alpha} \left(1 + \frac{1}{\alpha} \right)^{J(s,a,h)+1} \\
&\geq -\frac{HSA}{\alpha} \left(1 + \frac{1}{\alpha} \right) + \frac{HSA}{\alpha} \left(1 + \frac{1}{\alpha} \right)^{\frac{HSA + \sum_{s,a,h} J(s,a,h)}{HSA}} \quad (\text{Jensen's Inequality}) \\
&\geq -\frac{HSA}{\alpha} \left(1 + \frac{1}{\alpha} \right) + \frac{HSA}{\alpha} \left(1 + \frac{1}{\alpha} \right)^{\frac{J}{HSA}}
\end{aligned}$$

where $J \leq HSA + \sum_{s,a,h} J(s,a,h)$ because it is not necessarily the case that every state-action-stage will be visited. Rearranging this gives:

$$\frac{K\alpha}{SA} + 1 \geq \left(1 + \frac{1}{\alpha} \right)^{\frac{J}{HSA}}$$

Taking logs of both sides and rearranging one last time gives:

$$J \leq \frac{HSA \ln \left(\frac{K\alpha}{SA} + 1 \right)}{\ln \left(1 + \frac{1}{\alpha} \right)}$$

as required. $\square$

Now, we have an upper bound on the number of epochs, which is what we will use to control the episodes with long delays. Lemma 5.7 will be used to control the summation of the bonuses for the episodes with short delays.

Lemma 5.7. *If $n_0, n_1, \dots, n_J$ are an arbitrary sequence of real-valued numbers satisfying $N_0 := 0$ and $0 \leq n_j \leq \frac{1}{\alpha} N_{j-1}$ with $N_{j-1} = \max\{1, \sum_{i=0}^{j-1} n_i\}$ for all $j \leq J$, then*

$$Z_J = \sum_{j=1}^J \frac{n_j}{N_{j-1}} \leq \left(1 + \frac{1}{\alpha} \right) + \left(1 + \frac{1}{\alpha} \right) \ln(N_J)$$

Proof. The proof of this claim will utilise induction on J . Suppose $\sum_{j=1}^{J-1} n_j \leq 1$ which immediately tells us that $N_j = 1$ for all $j \leq J-1$. Therefore, the summation can be controlled as follows:

$$Z_J = \sum_{j=1}^J \frac{n_j}{N_{j-1}} = \sum_{j=1}^J n_j = n_J + \sum_{j=1}^{J-1} n_j$$

$$\begin{aligned}
&\leq 1 + n_J \leq 1 + \frac{1}{\alpha} && (\text{Since } n_J \leq N_{J-1}/\alpha = 1/\alpha) \\
&\leq \left(1 + \frac{1}{\alpha}\right) + \left(1 + \frac{1}{\alpha}\right) \ln(N_J) && (\text{Since } N_J \geq 1)
\end{aligned}$$

This result gives the desired result whenever $\sum_{j=1}^{J-1} n_j \leq 1$. Suppose $\sum_{j=1}^J n_j > 1$ and assume that the claim holds for some $J \geq 1$:

$$Z_J \leq \left(1 + \frac{1}{\alpha}\right) + \left(1 + \frac{1}{\alpha}\right) \ln(N_J) \quad (\text{Inductive Hypothesis})$$

For $J + 1$, we can apply the inductive hypothesis to show that:

$$\begin{aligned}
Z_{J+1} &= Z_J + \frac{n_{J+1}}{N_J} \\
&\leq \left(1 + \frac{1}{\alpha}\right) + \left(1 + \frac{1}{\alpha}\right) \ln(N_J) + \frac{n_{J+1}}{N_J} && (\text{Inductive Hypothesis}) \\
&\leq \left(1 + \frac{1}{\alpha}\right) + \left(1 + \frac{1}{\alpha}\right) \ln(N_J) + \left(1 + \frac{n_{J+1}}{N_J}\right) \ln\left(1 + \frac{n_{J+1}}{N_J}\right) && (*) \\
&\leq \left(1 + \frac{1}{\alpha}\right) + \left(1 + \frac{1}{\alpha}\right) \ln(N_J) + \left(1 + \frac{1}{\alpha}\right) \ln\left(1 + \frac{n_{J+1}}{N_J}\right) && (\text{Since } n_{J+1} \leq N_J/\alpha) \\
&= \left(1 + \frac{1}{\alpha}\right) + \left(1 + \frac{1}{\alpha}\right) \ln\left(N_J \left(1 + \frac{n_{J+1}}{N_J}\right)\right) \\
&= \left(1 + \frac{1}{\alpha}\right) + \left(1 + \frac{1}{\alpha}\right) \ln(N_J + n_{J+1}) \\
&= \left(1 + \frac{1}{\alpha}\right) + \left(1 + \frac{1}{\alpha}\right) \ln(N_{J+1})
\end{aligned}$$

where $(*)$ uses the fact that $x \leq (1+x) \ln(1+x)$ for $x \geq 0$ and this gives the stated result. $\square$

Lemma 5.8. *Algorithm 5 ensures that the summation of the counters across the episodes where no update occurs has the following upper bounds:*

$$\sum_{s,a,h} \sum_{j=1}^J \frac{n_{k_j+1,h}^{k_j+1}(s,a)}{N'_{k_j h}(s,a)} \leq 4 \left(1 + \frac{1}{\alpha}\right) HSA \ln\left(\frac{K}{SA}\right)$$

where the final inequality holds for $K/SA \geq \exp(1)$.

Proof. Let $N'_j(s,a,h)$ denote the number of observed visits to (s,a,h) at the start of the final epoch. Then, we have that:

$$\sum_{s,a,h} \sum_{j=1}^J \frac{n_{k_j+1,h}^{k_j+1}(s,a)}{N'_{k_j h}(s,a)} \leq \sum_{s,a,h} \left(\left(1 + \frac{1}{\alpha}\right) + \left(1 + \frac{1}{\alpha}\right) \log(N'_j(s,a,h)) \right) \quad (\text{Lemma 5.7})$$

$$\begin{aligned}
&= \left(1 + \frac{1}{\alpha}\right) HSA + \left(1 + \frac{1}{\alpha}\right) \sum_{s,a,h} \log(N'_J(s, a, h)) && \text{(Expand Summation)} \\
&\leq \left(1 + \frac{1}{\alpha}\right) HSA + \left(1 + \frac{1}{\alpha}\right) HSA \log\left(\frac{\sum_{s,a,h} N'_J(s, a, h)}{HSA}\right) && \text{(Jensen's)} \\
&\leq \left(1 + \frac{1}{\alpha}\right) HSA + \left(1 + \frac{1}{\alpha}\right) HSA \log\left(\frac{KH}{HSA}\right) && (\sum_{s,a,h} N'_J(s, a, h) \leq KH) \\
&= \left(1 + \frac{1}{\alpha}\right) HSA + \left(1 + \frac{1}{\alpha}\right) HSA \log\left(\frac{K}{SA}\right) \\
&\leq 2 \left(1 + \frac{1}{\alpha}\right) HSA \log\left(\frac{K}{SA}\right)
\end{aligned}$$

for $K/SA \geq \exp(1)$, as required. $\square$

5.5.2 Proof of Theorem 5.2

Proof. Let $\tilde{\Delta}_h^k(s) = \tilde{V}_h^{\pi^k}(s) - V_h^{\pi^k}(s)$ denote the difference between the optimistic and actual value of policy π^k from state s and stage h of the k -th episode. On the complement of the event in Assumption 5.2, it holds that:

$$\begin{aligned}
\hat{R}_K(\pi, M) &= \sum_{k=1}^K \left(V_1^*(S_1^k) - V_1^{\pi^k}(S_1^k) \right) \\
&\leq \sum_{k=1}^K \left(\tilde{V}_1^{\pi^k}(S_1^k) - V_1^{\pi^k}(S_1^k) \right) = \sum_{k=1}^K \tilde{\Delta}_1^k(S_1^k) = \sum_{j=1}^J \sum_{k=k_j}^{k_{j+1}-1} \tilde{\Delta}_1^k(S_1^k) \\
&= \sum_{j=1}^J \tilde{\Delta}_1^{k_j}(S_1^{k_j}) \tag{5.21}
\end{aligned}$$

$$+ \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \tilde{\Delta}_1^k(S_1^k) \mathbf{1}\{k + \tau_k \geq k_{j+1}\} \tag{5.22}$$

$$+ \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \tilde{\Delta}_1^k(S_1^k) \mathbf{1}\{k + \tau_k < k_{j+1}\} \tag{5.23}$$

where the inequality follows from optimism, the penultimate equality follows from epochs consisting of disjoint sets of episodes, and the final equality follows from splitting the episodes into three disjoint sets. Notably, Equation (5.21) is the set of episodes when a policy update occurs, Equation (5.22) is the set of episodes belonging to the j -th epoch with a long delay, and Equation (5.23) is the set of episodes within the j -th epoch with a short delay.

First, Equation (5.21) is easily controlled using Equation (5.9) and Lemma 5.6:

$$(5.21) \leq HJ$$

Second Equation (5.22) can be controlled via Lemma 4.9, which states that:

$$\mathbb{P} \left(\exists k \in \mathbb{N} : \sum_{i=1}^k 1\{i + \tau_i > k\} \geq \mathbb{E}[\tau] + \ln \left(\frac{k(k+1)}{\delta} \right) + \sqrt{2\mathbb{E}[\tau] \ln \left(\frac{k(k+1)}{\delta} \right)} \right) \leq \delta \quad (5.24)$$

Combining Equation (5.24) with Lemma 5.6 and the fact that the regret of any episode is at most H allows us to upper bound Equation (5.22) as follows:

$$\begin{aligned} (5.22) &= \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \tilde{\Delta}_1^k(S_1^k) 1\{k + \tau_k > k_{j+1} - 1\} \\ &\leq H \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} 1\{k + \tau_k > k_{j+1} - 1\} \quad (\tilde{\Delta}_h^k(s) \leq H) \\ &\leq H \sum_{j=1}^J \sum_{k=1}^{k_{j+1}-1} 1\{k + \tau_k > k_{j+1} - 1\} \\ &\leq HJ \left(\mathbb{E}[\tau] + \ln \left(\frac{K(K+1)}{\delta} \right) + \sqrt{2\mathbb{E}[\tau] \ln \left(\frac{K(K+1)}{\delta} \right)} \right) \end{aligned}$$

where the last inequality holds with probability at least $1 - \delta$.

Third, Equation (5.23) can be controlled by utilising the assumed independence of the delays and an application of Lemma 5.1:

$$\begin{aligned} (5.23) &\leq 6(H+C) \sqrt{T \log \left(\frac{K\pi}{6\delta'} \right)} \\ &\quad + 6 \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}} \sum_{h=1}^H \left(B_{k,h}(S_h^k, A_h^k) + \frac{CH^2SL}{N'_{k,h}(S_h^k, A_h^k)} \right) 1\{k + \tau_k < k_{j+1}\} \\ &= 6(H+C) \sqrt{T \log \left(\frac{K\pi}{6\delta'} \right)} \\ &\quad + 6 \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \sum_{h=1}^H \frac{\beta^{(2)} + CH^2SL}{N'_{k,j,h}(S_h^k, A_h^k)} 1\{k + \tau_k < k_{j+1}\} \quad (5.25) \end{aligned}$$

$$+ 6 \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \sum_{h=1}^H \frac{\beta_{k,h}^{(1)}}{\sqrt{N'_{k,j,h}(S_h^k, A_h^k)}} 1\{k + \tau_k < k_{j+1}\} \quad (5.26)$$

where the last equality follows from the form of the bonuses and the fact that the bonuses do not change within an epoch due to no update occurring. Thus, Equation (5.23) can be controlled by upper bounding Equations (5.25) and (5.26).

Lemma 5.8 can be applied to control Equation (5.25):

$$\begin{aligned}
(5.25) &= 6(\beta^{(2)} + CH^2SL) \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \sum_{h=1}^H \frac{1 \{k + \tau_k < k_{j+1}\}}{N'_{k_j h}(s_h^k, a_h^k)} \\
&= 6(\beta^{(2)} + CH^2SL) \sum_{s,a,h} \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \frac{1 \{S_h^k = s, A_h^k = a, k + \tau_k < k_{j+1}\}}{N'_{k_j h}(s, a)} \\
&= 6(\beta^{(2)} + CH^2SL) \sum_{s,a,h} \sum_{j=1}^J \frac{\sum_{k=k_j+1}^{k_{j+1}-1} 1 \{S_h^k = s, A_h^k = a, k + \tau_k < k_{j+1}\}}{N'_{k_j h}(s, a)} \\
&= 6(\beta^{(2)} + CH^2SL) \sum_{s,a,h} \sum_{j=1}^J \frac{n_{k_j+1,h}^{k_{j+1}}(s, a)}{N'_{k_j h}(s, a)} \quad (\text{Definition in Equation (5.19)}) \\
&\leq 12(\beta^{(2)} + CH^2SL) \left(\left(1 + \frac{1}{\alpha}\right) HSA \ln \left(\frac{K}{SA} \right) \right) \quad (\text{By Lemma 5.8}) \\
&\leq 24 \max\{\beta^{(2)} HSA, CH^3 S^2 AL\} \left(1 + \frac{1}{\alpha}\right) \ln \left(\frac{K}{SA} \right)
\end{aligned}$$

Bounding Equation (5.26) requires some care due to the various forms of $\beta_{k,h}^{(1)}$, such as those that remain constant and those that utilise variance reduction techniques. However, Lemma 5.7 verifies that the summation of the visitation counters no longer depend on the delay. Therefore, Cauchy-Schwarz can be applied without introducing delays into the leading-order terms:

$$\begin{aligned}
(5.26) &= 6 \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \sum_{h=1}^H \frac{\beta_{k,h}^{(1)}}{\sqrt{N'_{k_j,h}(S_h^k, A_h^k)}} 1 \{k + \tau_k < k_{j+1}\} \\
&\leq 6 \sqrt{\sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \sum_{h=1}^H (\beta_{k,h}^{(1)})^2 1 \{k + \tau_k < k_{j+1}\} \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \sum_{h=1}^H \frac{1 \{k + \tau_k < k_{j+1}\}}{N'_{k_j,h}(S_h^k, A_h^k)}} \\
&= 6 \sqrt{\sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \sum_{h=1}^H (\beta_{k,h}^{(1)})^2 1 \{k + \tau_k < k_{j+1}\} \sum_{s,a,h} \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \frac{1 \{S_h^k = s, A_h^k = a, k + \tau_k < k_{j+1}\}}{N'_{k_j,h}(S_h^k, A_h^k)}} \\
&= 6 \sqrt{\sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \sum_{h=1}^H (\beta_{k,h}^{(1)})^2 1 \{k + \tau_k < k_{j+1}\} \sum_{s,a,h} \sum_{j=1}^J \frac{n_{k_j+1,h}^{k_{j+1}}(s, a)}{N'_{k_j,h}(S_h^k, A_h^k)}} \\
&\leq 6 \sqrt{2 \left(1 + \frac{1}{\alpha}\right) HSA \ln \left(\frac{K}{SA} \right) \sum_{j=1}^J \sum_{k=k_j+1}^{k_{j+1}-1} \sum_{h=1}^H (\beta_{k,h}^{(1)})^2 1 \{k + \tau_k < k_{j+1}\}} \\
&\leq 6 \sqrt{2 \left(1 + \frac{1}{\alpha}\right) \ln \left(\frac{K}{SA} \right)} \tilde{R}_K(\text{Base}, M)
\end{aligned}$$

where $\tilde{R}_K(\text{Base}, M)$ denotes an upper bound on the regret of the algorithm in the immediate feedback setting. The penultimate line in the above is simply the summation involving the nu-

erator of the exploration bonuses for the chosen algorithm under immediate feedback. However, it is scaled by a logarithmic factor, which is introduced by the slower updating. For $\beta_{k,h}^{(1)} = \beta$ that remains constant across all episode-stage pairs, the summation under the square root can be bounded by: $KH\beta^2$, which is tight up to logarithmic factors. When $\beta_{k,h}^{(1)}$ involves some form of empirical variance term, one can use the techniques outlined by (Neu and Pike-Burke, 2020; Azar et al., 2017; Fruit et al., 2020) to bound the summation under the square root by: $\approx H^2K$; once again this too is tight up to logarithmic factors. Therefore, (5.26) can be replaced with the upper bound on the regret in an immediate feedback environment multiplied by the extra logarithmic factors that arise from the slower updating. This leverages the fact that the summation of the bonuses are the leading term in the regret bound (Azar et al., 2017; Dann et al., 2018; Neu and Pike-Burke, 2020).

Bringing everything together gives:

$$\begin{aligned}\hat{R}_K(\pi, M) &\leq (5.21) + (5.22) + (5.25) + (5.26) \\ &\lesssim \sqrt{\left(1 + \frac{1}{\alpha}\right)} \tilde{R}_K(\text{Base}, M) + HJ\mathbb{E}[\tau] \\ &\lesssim \sqrt{\left(1 + \frac{1}{\alpha}\right)} \tilde{R}_K(\text{Base}, M) + \frac{H^2SA\mathbb{E}[\tau]}{\log(1 + \frac{1}{\alpha})}\end{aligned}$$

where $\lesssim$ suppresses numeric constants and logarithmic factors that do not depend on the parameters of Algorithm 5. Finally, these inequalities hold on the complement of the events defined in Assumption 5.2, and Equations (5.14), (5.15), (5.12) and (5.24). Therefore, the bound holds with probability at least $1 - 5\delta$. $\square$

5.6 Discussion of Active & Lazy Updating

Table 5.6 presents a selection of algorithms that fit into our framework and their accompanying theoretical guarantees when using the active and lazy updating meta-algorithms to handle delayed feedback. In particular, acting in delayed environments causes an additive increase in regret for almost all combinations of optimistic algorithms and meta-algorithms considered. This result mirrors what is seen in the bandit setting where algorithms incur an additive regret penalty involving $\mathbb{E}[\tau]$ (Joulani et al., 2013).

Base	C	$\tilde{R}_K(\text{Base}, M)$	Active Updating	Lazy Updating
UBEV (Dann et al., 2018)	1	$H^{3/2}\sqrt{SAT}$	$\tilde{R}_K(\text{Base}, M) + H^3 S^2 A\mathbb{E}[\tau]$	$\sqrt{(1 + \frac{1}{\alpha})} \tilde{R}_K(\text{Base}, M) + \frac{H^2 S A\mathbb{E}[\tau]}{\log(1 + \frac{1}{\alpha})}$
UCBVI-CH (Azar et al., 2017)	1	$H^{3/2}\sqrt{SAT}$	$\tilde{R}_K(\text{Base}, M) + H^3 S^2 A\mathbb{E}[\tau]$	$\sqrt{(1 + \frac{1}{\alpha})} \tilde{R}_K(\text{Base}, M) + \frac{H^2 S A\mathbb{E}[\tau]}{\log(1 + \frac{1}{\alpha})}$
UCRL2 (Jaksch et al., 2010)	0	$H^{3/2}S\sqrt{AT}$	$\tilde{R}_K(\text{Base}, M) + H^2 S^{3/2} A\mathbb{E}[\tau]$	$\sqrt{(1 + \frac{1}{\alpha})} \tilde{R}_K(\text{Base}, M) + \frac{H^2 S A\mathbb{E}[\tau]}{\log(1 + \frac{1}{\alpha})}$
KL-UCRL (Filippi et al., 2010a)	0	$H^{3/2}S\sqrt{AT}$	$\tilde{R}_K(\text{Base}, M) + H^2 S^{3/2} A\mathbb{E}[\tau]$	$\sqrt{(1 + \frac{1}{\alpha})} \tilde{R}_K(\text{Base}, M) + \frac{H^2 S A\mathbb{E}[\tau]}{\log(1 + \frac{1}{\alpha})}$
UCRL2B (Fruit et al., 2020)	0	$H\sqrt{STAT}$	$\sqrt{H} \tilde{R}_K(\text{Base}, M) + H^2 S^2 A\mathbb{E}[\tau]$	$\sqrt{(1 + \frac{1}{\alpha})} \tilde{R}_K(\text{Base}, M) + \frac{H^2 S A\mathbb{E}[\tau]}{\log(1 + \frac{1}{\alpha})}$
χ^2 -UCRL (Neu and Pike-Burke, 2020)	0	$HS\sqrt{AT}$	$\sqrt{H} \tilde{R}_K(\text{Base}, M) + H^2 S^2 A\mathbb{E}[\tau]$	$\sqrt{(1 + \frac{1}{\alpha})} \tilde{R}_K(\text{Base}, M) + \frac{H^2 S A\mathbb{E}[\tau]}{\log(1 + \frac{1}{\alpha})}$
UCBVI-BF (Azar et al., 2017)	1	$H\sqrt{SAT}$	$\sqrt{H} \tilde{R}_K(\text{Base}, M) + H^3 S^2 A\mathbb{E}[\tau]$	$\sqrt{(1 + \frac{1}{\alpha})} \tilde{R}_K(\text{Base}, M) + \frac{H^2 S A\mathbb{E}[\tau]}{\log(1 + \frac{1}{\alpha})}$

Table 5.1: A selection of upper bounds on the pseudo-regret of various immediate feedback algorithms when paired with Algorithms 4 and 5 in the delayed feedback setting.

For active updating and some base algorithms, the additive delay dependence comes at the price of a penalty to the leading order term in the regret bound. Namely, an extra $\sqrt{H}$. This extra penalty multiplying the leading order term is a feature of the theoretical analysis. Another important factor influencing the impact of the delays when using active updating is the parameter C . The penalty for delayed feedback is higher when $C = 1$. The worsened delay dependence for these algorithms is due to the introduction of lower-order terms in the probabilistic analysis under immediate feedback, which allows for the derivation of tighter bonuses. Unfortunately, these lower-order terms become dependent on the delays and lead to a worse delay dependence.

To rectify the undesirable penalty to the leading order terms and the dependence on C , an alternative approach called lazy updating was considered. This approach achieves the same additive delay dependence for all optimistic model-based algorithms that satisfy Assumption 5.2. The price for this improved delay-dependence is a logarithmic factor. Algorithm 5 works by introducing an additional hyperparameter that controls how frequently the algorithm updates its policy. This hyperparameter is denoted by α and is named the *activity parameter*.

Theorem 5.2 indicates that there is a trade-off when selecting α . On the one hand, a larger value of α will minimise the penalty to the leading order term, which arises from the slower updating. On the other hand, the penalty introduced by the delays is a strictly increasing function of α , making large values undesirable. As $\alpha \rightarrow \infty$, lazy updating tends to active updating; at this limiting value, lazy updating will update as soon as it receives new feedback, just like active updating. Thus, the empirical performance of lazy updating should get closer to active updating as α increases. Section 5.7 demonstrates this empirically and shows that it is possible to get

most of the benefits of active updating with a relatively modest value of α , which has better worst-case regret bounds in the delayed feedback setting.

Comparatively, Theorems 5.1 and 5.2 significantly improve the regret bounds for many algorithms in the delayed feedback setting. [Lancewicki et al. \(2022\)](#) state bounds on the pseudo-regret of the form: $H^2 S \sqrt{AK} + H^2 S \tau_{\max}$ for all optimistic algorithms. Except for UCRL2 and KL-UCRL, the leading order term in their regret bound is loose in either H , S or both. Conversely, Theorems 5.1 and 5.2 are tight for all algorithms when utilising lazy updating and are only loose by a factor of $\sqrt{H}$ for a few algorithms when utilising active updating. Furthermore, $\mathbb{E}[\tau] \ll \tau_{\max}$ in almost all scenarios. As a result, the delay-dependence in the guarantees for Algorithms 4 and 5 are much tighter. Furthermore, these algorithms remove the need for *a-priori* knowledge of the maximal delay.

5.7 Experimental Results

This section investigates the impact of delayed feedback on the pseudo-regret of active and lazy updating in the chain environment of [Osband and Roy \(2017\)](#). Briefly, this environment consists of a sequence of S states arranged side-by-side. The learner starts in the left-most state and has to decide between $A = 2$ actions, head left or right. Each episode consists of $H = S$ stages and the only state with a reward is the right-most state. Thus, the optimal policy is to head right at every step. Heading left is always successful. However, heading right is successful with probability $1 - 1/S$. If unsuccessful, the learner moves one state to the left. Notably, any inefficient exploration strategy will take at least 2^S episodes to learn the optimal policy ([Osband and Roy, 2017](#)).

Chains with $H = S \in \{5, 10, 20, 30\}$ are considered and UCBVI-BF was used as the base algorithm in all experiments due to its superior theoretical guarantees under immediate feedback. For the lazy updating approach, several values for the activity hyperparameter are selected: $\alpha \in \{1, 10, 100\}$. In all experiments, the confidence parameter of the base algorithm is set to ensure that the regret bounds hold with probability 0.95. Additionally, the meta-algorithms are compared to the explicit exploration procedure proposed by [Lancewicki et al. \(2022\)](#). This procedure requires prior knowledge of the maximum delay, which is provided by generating all the delays before the first episode and taking the maximum. In practice, the maximum delay is often unknown and possibly infinite, making this approach infeasible.

The experiments consider: Constant, Geometric, Poisson, and Uniform delays. For each of these distributions, the following expected delays are examined: $\mathbb{E}[\tau] \in \{0, 100, 200, 300, 400, 500\}$. For the uniformly distributed delays, the lower and upper limits are set to 0 and $2\mathbb{E}[\tau]$, respectively. All results are averaged over 30 independent runs, and the shaded regions in all figures contain 95% of the empirical results.

Figure 5.1 displays the results for experiments in the chain environment with $S = 30$ and $\mathbb{E}[\tau] = 100$. The results for other chain lengths and values for the expected delay are presented in Appendix B.1. Empirically, active updating achieves the best performance of all three meta-algorithms. However, experimental results suggest that near-identical performance can be obtained with lazy updating by setting α to a sufficiently large constant. Both active and lazy updating offer superior performance to the explicit exploration approach of [Lancewicki et al. \(2022\)](#) in all experiments, despite their meta-algorithm having prior knowledge of the delays. In some cases, the meta-algorithms have converged to the optimal policy before the explicit exploration procedure finishes.

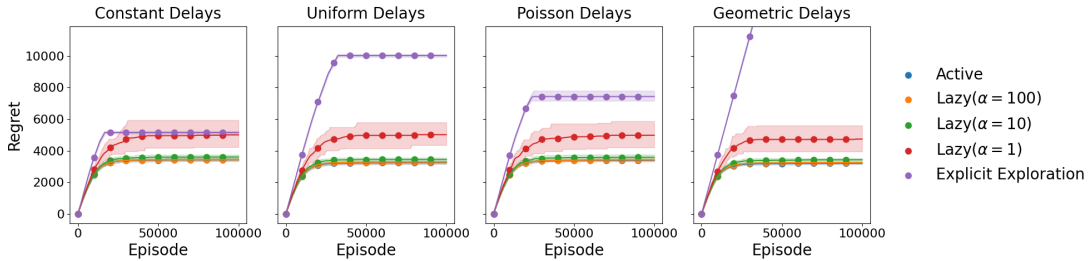


Figure 5.1: Experimental results visualising the growth of the pseudo-regret as a function of the episode. These results correspond to the chain environment with $H = S = 30$.

Next, the impact of different delay distributions on the regret of the meta-algorithms is examined. Empirically, Figure 5.2 shows that the regret penalty of delays at the end of the final episode is linear in the expected delay for active and lazy updating, as predicted by the theoretical analysis. For lazy updating, the gradient of this linear relationship decreases with α , which aligns with expectations based on the $\log(1 + 1/\alpha)$ term in the denominator of the delay-dependent terms in the regret bounds.

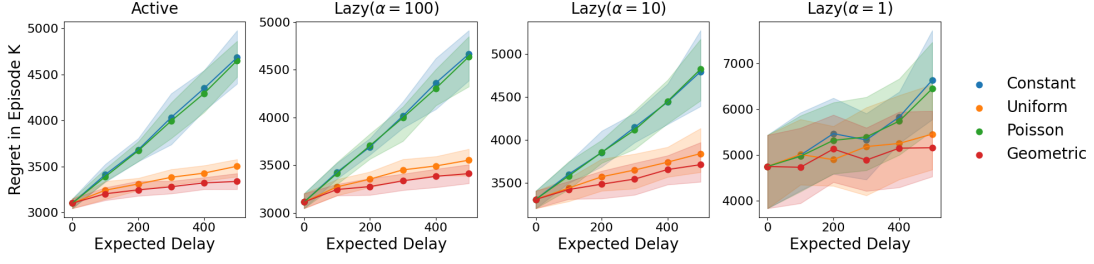


Figure 5.2: Experimental results depicting the growth of the pseudo-regret at the end of the final episode as a function of the expected delay. These results correspond to the chain environment with $H = S = 30$.

Interestingly, lazy updating with $\alpha = 1$ is the most robust to the delay distribution. This can be attributed to forcing **Base** to wait for long periods between updates. Intuitively, if the epochs are long enough, most information within an epoch will be received before an update, leading to little loss of information. Further investigation into this effect presents an interesting avenue for future work.

5.8 Conclusion

This chapter explored the impact of delayed feedback in episodic reinforcement learning and introduced two general-purpose meta-algorithms: active updating and lazy updating. These meta-algorithms enable existing optimistic model-based reinforcement learning algorithms to handle stochastic delays while maintaining theoretical guarantees under mild assumptions on the delays. The analysis demonstrated that the penalty for delayed feedback is an additive term involving the expected delay, independent of the number of episodes. This additive penalty aligns with findings from the multi-armed bandit setting, despite the additional complexities of reinforcement learning.

Many algorithms designed to match minimax-optimal regret guarantees exhibit a higher dependence on delay when paired with active updating. This arises from lower-order terms introduced in the analysis, which become delay-dependent. However, this effect can be mitigated by updating less frequently as episodes progress, which is the core idea behind lazy updating. By controlling the update frequency, lazy updating provides a more robust approach, ensuring that regret bounds remain tight while preserving the advantages of efficient exploration.

Several extensions and improvements could be built on the work presented in this chapter. The focus here was on quantifying the impact of delayed feedback across a broad class of algorithms, leading to inflated leading-order terms for some when using active updating. A more refined algorithm-specific analysis may remove this artefact. Finally, while the theoretical results apply to optimistic model-based algorithms, developing a truly black-box approach that applies to a wider range of reinforcement learning methods would be a valuable direction for future research. For this goal, the idea of lazy updating seems to be the most promising of the two methods proposed in this chapter.

Part III

Cooperation & Pairwise Comparisons

Chapter 6

Cooperative Multi-Armed Bandits

This chapter is based on research published in AISTATS and was conducted under the supervision of Sarah Filippi and Ciara Pike-Burke ([Howson et al., 2025](#)).

6.1 Motivation

Chapters 4 and 5 focus on delayed feedback in sequential decision-making processes involving an isolated agent interacting with an environment. However, many real-world decision-making systems consist of multiple agents operating in the same environment, creating opportunities for cooperation to accelerate the learning process. Applications requiring distributed decision-making arise in domains such as sensor networks and large-scale recommender systems ([Tran-Thanh et al., 2011](#); [Tekin et al., 2014](#)). In these settings, multiple decision-makers interact with an environment and, ideally, share knowledge to improve their collective performance.

For example, consider a large-scale recommender system where each server hosts a decision-maker. Every time a user arrives at one of the servers, the corresponding decision-maker selects an item to recommend, and the user provides a reward signal in response. By sharing this information with neighbouring servers, the network of decision-makers can enhance their recommendations across the system, improving the overall user experience. This cooperation is particularly desirable when all decision-makers operate under a shared objective, such as maximising engagement or revenue for the same company.

Naturally, cooperation can be achieved through communication, but in most real-world systems, information exchange is not immediate. This introduces an interesting connection between distributed decision-making and the delayed feedback problems studied in Chapters 4 and 5. These chapters studied the impact of delayed feedback in the single-agent setting, where the delay is imposed on the feedback of the agent's own decisions. Here, the delays slow down the learning process and increase the regret of the agent by an additive penalty. However, in multi-agent systems, each agent receives their own feedback immediately while also obtaining additional information from other agents, subject to a deterministic communication delay. This chapter explores how agents can effectively incorporate delayed information from others and whether such cooperation can ultimately compensate for or even surpass the learning speed of isolated agents.

6.2 Problem Setting

The cooperative multi-agent version of the stochastic multi-armed bandit consists of the same ingredients presented in Chapter 2.1. Formally, $\mathcal{A} = \{1, 2, \dots, k\}$ denotes the set of k actions, and each action has an unknown probability distribution over the rewards:

$$\nu = \{P_1, P_2, \dots, P_k\}$$

where ν is the bandit environment. However, rather than having an agent interact with the environment in isolation, there is now a network of agents interacting with the environment. This network will be modelled by a graph, which is a collection of vertices and edges:

$$G = (V, E) \text{ where } V = \{1, 2, \dots, m\} \text{ and } E \subseteq \{(i, j) \in V \times V : i \neq j\}.$$

Each vertex will represent an agent, and each edge will represent a channel of communication between a pair of agents. Therefore, in totality, there are m agents. The purpose of introducing an underlying graph is to represent connections between agents and to model scenarios where each agent is only aware of a subset of the other agents. This subset of other agents is known as their *neighbourhood*:

$$N_i = \{j \in V : (i, j) \in E\}$$

which is the set of agents that the i -th agent is connected to via an edge in the underlying graph. Briefly, the interaction between the network of agents occurs over rounds. Each round involves every agent simultaneously selecting an action and receiving an independent draw from the corresponding probability distribution. After each agent receives their reward, and before the start of the next round, they may communicate information with their neighbours. Procedure 6.1 presents an outline of the decision-making process for ease of reference.

Procedure 6.1: Multi-Agent Multi-Armed Bandits

For $t = 1, 2, \dots, n$:

1. Each $i \in V$ simultaneously selects: $A_t^i \in \mathcal{A}$
2. Each $i \in V$ simultaneously receives: $X_t^i \in \mathbb{R}$
3. Each $i \in V$ simultaneously communicates with: $j \in N_i$.

Step 3 of Procedure 6.1 makes it explicit that each agent can only directly communicate with the agents within their neighbourhood. Therefore, any pair of agents that are not adjacent must communicate by passing information through their neighbours.

Throughout, we will require some assumptions on the bandit environment and the graph, which are presented in Assumptions 6.1 and 6.2.

Assumption 6.1 (Assumptions on the Bandit). *The multi-armed bandit environment satisfies the following conditions:*

- (i) $\mu_a(\nu) = \int x dP_a(x) < \infty$ for all $a \in \mathcal{A}$
- (ii) $X_t^i \stackrel{\text{iid}}{\sim} P_a$ whenever $A_t^i = a$ for all $(a, i, t) \in \mathcal{A} \times V \times \mathbb{N}$

Assumption 6.2 (Assumptions on the Network). *Let $d_{i \rightarrow j}$ denote the length of the shortest path between agents $(i, j) \in V \times V$. Then, the network must satisfy the following conditions:*

- (i) Messages from $i \in V$ to $j \in V$ can be sent in $d_{i \rightarrow j}$ rounds.
- (ii) The diameter of the graph is finite: $d = \max_{(i,j) \in V \times V} d_{i \rightarrow j} < \infty$

Assumption 6.1 is relatively mild, as it does not constrain the set of distributions to take on a specific parametric form or have certain tail properties. The only requirement is that the rewards

are conditionally independent given the action, which is a standard assumption in multi-armed bandits.

Assumption 6.2 is standard in the literature on the cooperative multi-agent multi-armed bandit problem (Landgren et al., 2016; Martínez-Rubio et al., 2019; Wang et al., 2020; Dubey and Pentland, 2020a; Lalitha and Goldsmith, 2021; Raveh et al., 2024). Essentially, it says that the graph is connected so that there is a path of finite length between any pair of vertices.

6.2.1 Measuring Performance

Naturally, it is of interest to design algorithms that provide guarantees on the performance of the network as a whole. Equation (2.4) provides a natural measure of performance for each agent individually. Summing these individual performances gives rise to what is known as the *group regret*:

$$R_n(G, \nu) = mn\mu_\star(\nu) - \mathbb{E} \left[\sum_{i=1}^m \sum_{t=1}^n X_t^i \right] \quad (6.1)$$

where G is the communication graph. Thus, the task is to design algorithms that minimise the group regret. Intuitively, the idea is that having the agents cooperate within their neighbourhoods could improve the decision-making of each agent individually and the decision-making of the network as a whole. This should be possible by having each agent communicate information about the bandit environment with the other agents in their neighbourhood. Indeed, Assumption 6.1 says that each agent is interacting with the same bandit environment. Therefore, the information received by each agent is useful for all of the other agents in the network.

6.2.2 Lower Bounds for Edge Cases

Procedure 6.1 without any communication is identical to running m independent copies of the standard multi-armed bandit problem in parallel. Therefore, we can obtain a lower bound on the group regret without communication from the lower bounds of Auer et al. (2002b):

$$R_n(G, \nu) = \sum_{i=1}^m \left(\mu_\star(\nu) - \mathbb{E} \left[\sum_{t=1}^n X_t^i \right] \right) \geq \frac{m}{20} \sqrt{n(k-1)}$$

where $G = (V, \emptyset)$ due to the supposed absence of communication and the inequality applies Equation (2.10) to each element of the summation. Thus, the lower bound on the group regret without communication immediately follows from summing the lower bound on the individual

regret of each agent, which all operate independently because the network structure forbids communication.

Conversely, the network where every agent is connected to all other agents allows for the maximum amount of cooperation. Therefore, we can reduce the cooperative multi-agent multi-armed bandit problem to single-agent bandits with m plays per round. This implies the following lower bound in the cooperative setting:

$$R_n(K_m, \nu) \geq \frac{1}{20} \sqrt{mn(k-1)} \quad (6.2)$$

where $K_m = (V, \{(i, j) \in V \times V : i \neq j\})$ denotes the complete graph. Notably, the complete graph allows for the maximum amount of communication. Therefore, this lower bound applies to the cooperative multi-agent bandit problem for arbitrary network topologies. However, it may not be tight for graphs with a particular structure. Nevertheless, these lower bounds for networks that permit no communication and full communication highlight that cooperation can reduce the group regret by a factor of $\sqrt{m}$, which is substantial for large networks.

6.3 Literature Review

Section 6.2 presents the stochastic cooperative multi-armed bandit problem that is considered in this chapter. This problem and numerous variants have recently received attention in the literature. Section 6.3.3 presents a selection of such variants that are most relevant to the problem setting we study in this chapter.

Broadly, algorithms for the cooperative multi-agent multi-armed bandit problem, as described in Section 6.2, can be categorised into two styles of algorithms, namely *gossip-based* algorithms (Landgren et al., 2016; Martínez-Rubio et al., 2019; Lalitha and Goldsmith, 2021) and *leader-follower* algorithms (Bar-On and Mansour, 2019; Dubey and Pentland, 2020a; Wang et al., 2020). Sections 6.3.1 and 6.3.2 reviews the literature that take these approaches separately.

6.3.1 Gossip-Based Solutions

The gossip-based solutions are fully decentralised and have each agent approximate a centralised version of the chosen bandit algorithm. Here, centralised refers to a fictitious algorithm that has

access to full-network information. Approximating the centralised agent is achieved using what is known as a *gossip algorithm*, which is widely applied in the distributed computing literature (Boyd et al., 2005; Shah, 2009). Essentially, the gossip algorithm is an iterative procedure that seeks to establish a consensus amongst a network of agents. Perhaps the most common sub-problem in this field is distributed averaging (Xiao and Boyd, 2004). Here, each agent in the network possesses a value. The objective is to have each agent compute an approximation for the network-wide average of these values. This can be achieved by having each agent maintain an approximation, and making each agent repeatedly compute a weighted average of their own approximation with the approximations from their neighbours.

Duchi et al. (2012) present an adaptation for settings where the quantity being averaged is evolving over time. This adaptation of the gossip algorithm is what is applied in the multi-agent bandit setting (Landgren et al., 2016; Martínez-Rubio et al., 2019; Lalitha and Goldsmith, 2021).

Formally, $W \in \mathbb{R}^{m \times m}$ will denote the gossip matrix. This is the matrix that will be used to compute the weighted average of each agent's approximation of the network average with their neighbour's approximation. Thus, the gossip matrix must satisfy the following properties to ensure that each iteration consists of computing a weighted average amongst neighbouring agents:

- W has rows and columns that sum to one so that each iteration is indeed a weighted average, and
- $W_{ij} \geq 0$ with $W_{ij} = 0$ if $(i, j) \notin E$ so that agents can only average their current approximations with their immediate neighbours in each iteration.

Xiao and Boyd (2004) show that the gossip matrix must satisfy another necessary and sufficient condition for the averaging of values to take place:

$$1 = \lambda_1(W) > |\lambda_2(W)| \geq |\lambda_3(W)| \geq \dots \geq |\lambda_m(W)|$$

where $\lambda_i(W)$ denotes the eigenvalue with the i -th largest magnitude. While Xiao and Boyd (2004) focuses on static averaging, Duchi et al. (2012) investigate an extension to dynamic settings. This extension allows each agent to add values to the approximation in each round. Formally, the

iterative averaging procedure uses the gossip matrix as follows (Duchi et al., 2012):

$$z(t) = W z(t-1) + y(t) \quad (6.3)$$

where $z(t) \in \mathbb{R}^m$ and $y(t) \in \mathbb{R}^m$ are vectors. Entries of $z(t)$ represent each agent's current approximation of the network average at the end of the t -th iteration. Entries of $y(t)$ represent the value each agent wants to include in the t -th approximation. Formally, Xiao and Boyd (2004) show that the previously stated conditions on the gossip matrix guarantee the following:

$$\lim_{t \rightarrow \infty} W^t \rightarrow \frac{1}{m} \mathbf{1}\mathbf{1}^T.$$

Unrolling Equation (6.3) with $z(0) = \vec{0}$ reveals that:

$$z(t) = \sum_{s=1}^t W^{t-s} y(s)$$

which guarantees that the values each agent adds to the gossip algorithm will get averaged in the limit. Therefore, Equation (6.3) leaves the i -th agent with an approximation of the network average of the values provided by themselves and all the other agents in the network:

$$z_i(t) \approx \frac{1}{m} \sum_{j=1}^m \sum_{s=1}^t y_j(s)$$

Choosing W , the gossip matrix, is of great importance for gossip-based algorithms. Even when the network is known, choosing this matrix is a non-trivial task, and it is common to use heuristics (Xiao and Boyd, 2004; Duchi et al., 2012). The heuristics based on the graph Laplacian are the most widely adopted in the bandit setting (Landgren et al., 2016; Martínez-Rubio et al., 2019; Lalitha and Goldsmith, 2021):

$$W = I - \frac{1}{1 + \max_{i \in V} |N_i|} (D - M) = I - \frac{L}{1 + \max_{i \in V} |N_i|} \quad (6.4)$$

where $L = D - M$ denotes the graph Laplacian, and $M \in \{0, 1\}^{m \times m}$ and $D \in \mathbb{N}^{m \times m}$ are graph-related matrices. Formally, M denotes the adjacency matrix and D denotes the degree matrix.

These matrices can be defined element-wise as follows:

$$M_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E \\ 0 & \text{if } (i, j) \notin E \end{cases} \quad \text{and} \quad D_{ij} = \begin{cases} |N_i| & \text{if } i = j \\ 0 & \text{if } i \neq j. \end{cases}$$

[Landgren et al. \(2016\)](#) utilise Equation (6.3) with the gossip matrix of Equation (6.4) to extend the upper confidence bound algorithm to the multi-agent setting. Intuitively, the idea is to use the gossip algorithm to approximate what the upper confidence bound algorithm would do if it had access to full network information to construct its optimistic estimates. Letting $\zeta_a^i(t) = 1\{A_t^i = a\}$ and $\hat{X}_a^i(t) = X_t^i \zeta_a^i(t)$ gives ([Landgren et al., 2016](#)):

$$\hat{T}_a(t) = W \hat{T}_a(t-1) + \zeta_a(t)$$

$$\hat{S}_a(t) = W \hat{S}_a(t-1) + \hat{X}_a(t)$$

which are approximations of the network average plays and the network average sum of rewards for the a -th action at the end of the t -th round. Therefore, $\hat{S}_a(t)/\hat{T}_a(t)$ gives an approximation of the empirical mean computed with full network information. Showing that this estimator concentrates around the true expected rewards for each action requires a more careful analysis than the single-agent variant of the upper confidence bound algorithm. Intuitively, this is because each agent will have differing levels of uncertainty based on how well-connected they are to other agents in the network. This fact is reflected in the upper confidence bounds and in their guarantee on the group regret ([Landgren et al., 2016](#)):

$$R_n(G, \nu) \leq \sum_{a \neq \star} \sum_{i=1}^m \frac{1}{m} \frac{8\gamma(1+\epsilon_i) \ln(mn)}{\Delta_a(\nu)} + m \left(\frac{\gamma}{\gamma-1} + \sqrt{m} \sum_{i=2}^m \frac{|\lambda_i(W)|}{1-|\lambda_i(W)|} \right) \sum_{a \neq \star} \Delta_a(\nu)$$

where ϵ_i is a measure of the connectivity for the i -th agent that depends on spectral properties of the underlying graph and $\gamma > 1$ is an exploration hyperparameter. This guarantee exhibits an undesirable scaling of the leading-order term with graph-dependent quantities. Nevertheless, the gossiping algorithm can be applied to extend other single-agent bandit algorithms. [Lalitha and Goldsmith \(2021\)](#), for example, leverage the same idea to approximate what Thompson Sampling would do with full network information in bandit environments with Bernoulli reward distributions.

[Martínez-Rubio et al. \(2019\)](#) propose an intuitive idea that follows from looking at an upper

bound on the approximation error of using the gossip algorithm:

$$\left\| W^{t-s} y(s) - \frac{1}{m} \bar{1} \bar{1}^T y(s) \right\|_2 \leq |\lambda_2(W)|^{t-s} \|y(s)\|_2.$$

Specifically, this upper bound suggests that values added more recently are subject to larger deviations from their network averages. Thus, [Martínez-Rubio et al. \(2019\)](#) propose truncating the summation in Equation (6.3). The number of rounds to remove from the summation is chosen so that the approximation error that is introduced into the network averages is negligible. [Martínez-Rubio et al. \(2019\)](#) present another, more complex, version that builds on the same intuition but uses an accelerated gossiping algorithm. Theoretically, this more complex version offers the best guarantees on the group regret, which have the following form:

$$R_n(G, \nu) \leq \sum_{a \neq \star} \frac{8\eta \left(1 + \frac{\epsilon}{2}\right) \ln(mn)}{\Delta_a} + \left(\left\lceil \frac{\ln\left(\frac{2m}{\epsilon}\right)}{\sqrt{2 \ln\left(\frac{1}{|\lambda_2(W)|}\right)}} \right\rceil + m + 4 \right) \sum_{a \neq \star} \Delta_a$$

where $\eta > 1$ and $\epsilon \in (0, \eta^{-1}/7(\eta+1))$ are hyperparameters that control the amount of exploration and the approximation error, respectively. The main difference in the theoretical guarantee between the non-accelerated and accelerated gossiping algorithms is found in the additive constant. Empirically, the difference is not statistically significant ([Martínez-Rubio et al., 2019](#)). Section 6.7 replicates these findings and suggests that the benefits of the additional complexity are diminishing.

6.3.2 Leader-Follower Solutions

Another style of algorithm uses the leader-follower framework, which involves at least one agent who has the responsibility of coordinating the decision-making process. Essentially, the leader-follower solutions involve partitioning the network of agents into subsets. These subsets each contain an agent that is the nominated leader and numerous other agents known as followers. The leader is responsible for coordinating the decision-making within their subset of the network. There are several degrees of freedom in the design of leader-follower algorithms, such as how to partition the network of agents, how to select the leader, and how the leader coordinates the decision-making of their followers. Regardless of these specific details, communication between the leader of each component and the followers is required for coordination. This is implemented through *message passing*; a natural procedure that has agents aggregate their message with their

neighbours before passing the aggregated message on to their neighbours.

[Bar-On and Mansour \(2019\)](#) develop a leader-follower algorithm for the non-stochastic multi-agent multi-armed bandit problem. They propose an automatic procedure that partitions the network into subsets of agents, where each agent in each subset knows their leader. The leader of each subset follows the well-known exponential weights algorithm and communicates their distribution over actions to the followers in their subset of the network. The followers utilise this distribution to select their decisions, and send several quantities back to their leader: the distribution, the action, and the feedback from the environment. Using this information allows the leader to update their distribution over actions in an appropriate manner. [Bar-On and Mansour \(2019\)](#) prove that the regret of the i -th agent in the network, who is following policy π_i as described by their algorithm, can be upper bounded as follows:

$$R_n(\pi_i, \nu) \lesssim \sqrt{n + \frac{kn}{|N_i|}}$$

where $\lesssim$ denote inequality up to absolute constants and π_i denotes the policy being followed by the i -th agent. This contrasts with all of the algorithms presented in the previous subsection, where all known guarantees hold at the group level. Naturally, the individual regret can be used to obtain bounds on the group regret and the resulting guarantee matches a direct extension of the exponential weights to the multi-agent setting ([Cesa-Bianchi et al., 2016](#)). The group regret of this approach scales with the independence number of the underlying graph, which does not match the lower bound in Section 6.2. However, the problem is non-stochastic, and the relationship between cooperative multi-armed bandits and delayed feedback suggests that this multiplicative graph dependence is unavoidable.

[Wang et al. \(2020\)](#) propose a different instantiation of the leader-follower algorithm for the stochastic version of the multi-agent multi-armed bandit problem, which is the primary focus of this chapter. The authors focus on simultaneously minimising the cumulative regret and the expected communication. Therefore, the network of agents elects exactly one leader who proceeds to utilise the KL-UCB algorithm ([Cappé et al., 2013](#)). Notably, KL-UCB is updated using only the leader's action-reward pairs. The followers, which is the set containing the remaining $m - 1$ agents, all play the index of the action that maximises the empirical means of the leader. The index of this action is communicated by the leader, and the message passing algorithm is used

to propagate this information throughout the graph. Thus, after at most d rounds, every agent will have received the updated index of the greedy action. Importantly, the followers do not send any information back to the leader. This is a sacrifice made to minimise the total amount of communication. Thus, the leader is the only exploring agent. Wang et al. (2020) call this technique *parsimonious exploration*, due to the lack of network-wide exploration, and show that the group regret has the following upper bound:

$$R_n(G, \nu) \leq \sum_{a \neq \star} \frac{\Delta_a}{D_{\text{KL}}(\mu_a + \delta \parallel \mu_\star - \delta)} (\ln(n) + 4 \ln \ln(n)) + 9dkm (29 + k\delta^{-2})$$

where $\delta \geq \min_{a, a'} \{\mu_a - \mu_{a'} \mid \text{such that } \mu_a > \mu_{a'}\}$.

Dubey and Pentland (2020a) design another leader-follower algorithm for the stochastic cooperative multi-armed bandit problem with heavy tails, which is a superset of the standard bandit problem. Similarly to Bar-On and Mansour (2019), they propose electing numerous leaders, and the number of leaders depends on a hyperparameter, γ . Intuitively, γ can be interpreted as the communication radius, and it is the maximum number of edges a message can traverse before it expires. Setting $\gamma = d$, the diameter of the graph, would allow a message to propagate across the entire network. Conversely, $\gamma = 1$ only allows messages to be passed between neighbouring agents. Thus, choosing the leaders and followers heavily depends on the value of this hyperparameter. Specifically, Dubey and Pentland (2020a) define an extended graph, also referred to as the γ -th power of the graph, as follows:

$$G_\gamma = (V, E_\gamma) \text{ where } E_\gamma = 1\{(i, j) \in V \times V : d_{i \rightarrow j} \leq \gamma\}$$

which adds an edge to the original graph if there exists a shortest path between a pair of agents that is smaller than the communication radius, γ . Then, the leaders are assumed to be chosen such that they form a maximal independence set of the extended graph:

$$\alpha(G_\gamma) = \max_{V' \subseteq V} \{(i, j) \in V' \times V' : (i, j) \notin E_\gamma\}$$

which means that no pair of leaders are adjacent in the extended graph and the number of leaders is as large as possible. Followers are assumed to be connected to just one of the leaders. Each leader uses the Robust UCB algorithm (Bubeck et al., 2013a). Specifically, Dubey and Pentland (2020a) focus on the variant of the algorithm that uses the trimmed mean estimator

(Bickel, 1965). After every decision, the leader will propagate the chosen action to the followers in their subset of the network via message passing, and the followers are instructed to copy the most recent action given to them by the leader. The followers send the resulting action-reward pair back to the leader, who incorporates this information into their upper confidence bounds as soon as it is received. Ultimately, Dubey and Pentland (2020a) prove upper bounds on the individual and group regret, which are stated explicitly in Section 6.5. These guarantees indicate that even the poorly connected agents can substantially improve upon the performance that they could obtain by running an independent copy of the Robust UCB algorithm. Raveh et al. (2024) apply the same leader-follower approach in the multi-agent duelling bandit setting, and have each leader use the Relative UCB algorithm (Zoghi et al., 2014). Again, further details are provided in Section 6.5.

6.3.3 Related Problems

There are many variants of the multi-agent multi-armed bandit problem, which is in part due to the many degrees of freedom introduced by considering multiple agents. The variant studied in this chapter has served as a springboard for numerous extensions. These extensions modify the standard multi-agent multi-armed bandit problem to capture salient features present in certain distributed decision-making tasks. The proceeding paragraphs highlight some of these extensions and serves the purpose of providing a brief overview of the broader literature on multi-agent decision-making tasks.

Communication Constraints. Sections 6.3.1 and 6.3.2 presents several algorithms designed under the assumption that each agent can communicate around k values to their neighbours with arbitrary precision in each round. However, this is a bottleneck in some applications. Aforementioned, Wang et al. (2020) address this challenge by placing a single agent in charge of all the exploration. They show that the expected number of bits of communication is independent of the number of rounds. Agarwal et al. (2022) devise another algorithm where all agents explore and communicate. They propose an action-elimination algorithm that operates in epochs and have the agents communicate at the end of each epoch. Thus, they are able to control the communication complexity by controlling the number of epochs.

Asynchronous Settings. Another assumption that we make in this chapter is that the agents play their actions at exactly the same time. This is unlikely to occur in real-world distributed

systems. For example, recommender systems with geographically distributed servers will likely have different periods of peak activity due to differing time zones. [Sankararaman et al. \(2019\)](#) assume that each agent makes decisions at time points that follow a Poisson Process. [Yang et al. \(2021\)](#) focus on settings where agents make decisions at known frequencies. [Chen et al. \(2023\)](#) relax this assumption to unknown and arbitrary points in time.

Heterogenous Environments. Generally, the literature assumes that each agent interacts with the same bandit environment. However, this need not be the case in real-world applications, where the set of reward distributions that characterise the bandit environment are different for each agent. Naturally, this leads to different definitions of what is deemed optimal. [Shahrampour et al. \(2017\)](#) average the expected values for each action across the network. Thus, the optimal policy in their setting will pick the action whose network-averaged expected reward is the highest in every round. However, this could result in some agents being forced to pick actions with low or the lowest expected reward in their own environment. [Hossain et al. \(2021\)](#) aim to fix this issue by learning a distribution over actions that maximises the Nash Social Welfare. This measure of quality enforces fairness because the optimal policy maximises the product of expected rewards over the agents. Thus, it avoids scenarios where a minority of agents are forced to pick a sub-optimal action to benefit the majority.

Imperfect Communication. Sometimes, communication channels fail and this feature is not captured in any of the above models of multi-agent problems. [Madhushani et al. \(2021\)](#) relax this assumption and suppose that the communication channels fail with a fixed probability. Generally, the theoretical guarantees worsen as the failure probability increases, as would be expected. Additionally, the authors consider the setting where the rewards can be arbitrarily corrupted in the communication process. [Dubey and Pentland \(2020b\)](#) consider a similar setting where some of the agents randomly communicate incorrect information from an unknown distribution.

Multiplayer Settings. Perhaps the best-studied setting involves agents receiving degraded or zero rewards for playing the same action at the same time. This is in contrast to the model considered in this chapter, where each agent receives their own reward, regardless of the decisions made by the other agents in the network. The literature on this topic is vast. Recently, an extensive survey has been published that covers the algorithms and techniques that are relevant to this problem setting ([Boursier and Perchet, 2024](#)).

6.4 Black-Box Reduction

Section 6.3 presents an overview of the existing approaches for designing cooperative multi-armed bandit algorithms. These works adapt specific single-agent algorithms to the multi-agent setting. Rather than taking this approach, we propose a general-purpose black-box reduction that takes any single-agent algorithm as input and adapts it to work in cooperative multi-agent settings.

Algorithm 6 QuACK

```

Input: Leader  $i \in V$ 
Input for Leader: Bandit Algorithm  $\pi$ 
Initialisation for Leader: Empty Queue  $Q_a = \emptyset$  for all  $a \in \mathcal{A}$ 
for  $t = 1, 2, \dots, n$  do
  Leader ( $i$ )
    Receive messages via shortest path:
       $\mathcal{M}_t = \{(A_{t-d_{i \rightarrow j}}^j, X_{t-d_{i \rightarrow j}}^j)\}_{j \neq i}$ 
    Append messages to the queues:
       $Q_a = Q_a \cup \{x : (a', x) \in \mathcal{M}_t \text{ and } a' = a\}$ 
    Select  $a$  with  $\pi$ 
    while  $Q_a \neq \emptyset$  do
      Remove  $x$  from  $Q_a$ 
      Update  $\pi$  with  $(a, x)$ 
      Select  $a$  with  $\pi$ 
    end while
    Play  $A_t^i = a$ 
    Observe  $X_t^i \sim P_{A_t^i}$ 
    Send  $A_t^i$  to all  $j \neq i$  via shortest path
  Follower ( $j \neq i$ )
    if  $t > d_{i \rightarrow j}$  then
      Receive  $A_{t-d_{i \rightarrow j}}^i$  from  $i$  via shortest path
      Play  $A_t^j = A_{t-d_{i \rightarrow j}}^i$ 
    else
      Play  $A_t^j$  uniformly at random
    end if
    Observe  $X_t^j \sim P_{A_t^j}$ 
    Send  $(A_t^j, X_t^j)$  to  $i$  via shortest path.
end for

```

Algorithm 6 presents the pseudo-code for our black-box reduction: QuACK, *a queuing algorithm for cooperative k -armed bandits*. This algorithm uses the leader-follower paradigm and builds on the queuing approach that was developed for the setting of delayed feedback (Joulani et al., 2013; Mandel et al., 2015). For clarity, the algorithm is presented assuming that:

- the network of agents know the identity of the leading agent, and
- the shortest path between each follower and the chosen leader is known.

Section 6.6 shows how to use well-known algorithms to nominate the leader of the network and

compute the shortest paths that are required for message passing. These algorithms are all distributed, and this means that the agents only need to know the structure of the graph locally. That is to say that each agent is only required to know their neighbours.

Essentially, QuACK takes an arbitrary *leader* in the network and provides them with a single-agent bandit algorithm to select actions for the entire network. The leader begins by initialising an empty queue for each action. These queues will store the rewards that other agents observe from the environment and have been passed to the leader via message passing. Intuitively, the leader can use the reward samples in these queues instead of playing actions in the real environment.

Specifically, in each round, the leader begins by asking their bandit algorithm for an action. Whenever there is a reward in the queue for that action, the leader will take the reward from the queue and use this to update the bandit algorithm. Importantly, the leader does not play this action in the environment. This process of requesting an action and taking a reward from the queue continues until the bandit algorithm suggests playing an action whose queue is empty. Once this occurs, the leader will play the action in the environment and will use the observed reward to update the bandit algorithm. The leader then communicates this decision to the other, *follower*, agents. Once a follower receives the decision from the leader, they will proceed to play the instructed action and communicate the reward back to the leader. Once the leader receives rewards from their followers, they will place the rewards in the corresponding queues, and begin the process again. Notably, the process of sending decisions and rewards can take several rounds because the message needs to travel across the network.

6.4.1 Message-Passing

The message a follower $j \neq i$ sends to the leader i at the end of the t -th round contains only the action and the reward:

$$(A_t^j, X_t^j).$$

Sending such simple messages is possible since there is a known fixed route between the leader and any follower. Although multiple routes with the minimum path length might exist, the algorithm determines exactly one and adheres to it throughout. Computing these routes can be done in a distributed manner before the start of the interaction. Specifically, this amounts to

constructing the shortest path tree of the graph, which can be done by solving the single-source shortest-path problem before interacting with the bandit environment (Ahuja et al., 1993). Figure 6.1 illustrates the shortest path tree for the grid graph, where the black vertex represents the leader.

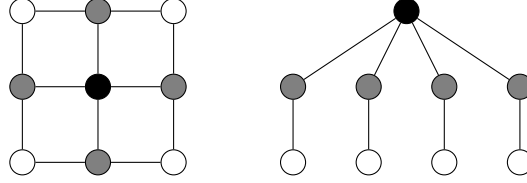


Figure 6.1: The Grid Graph and its Shortest Path Tree when the leader is the central vertex.

The Distributed Bellman-Ford algorithm can solve the single-source shortest-path problem exactly and it does so using approximately m iterations, whilst only requiring that each agent knows their neighbours and has a unique identifier (Ford, 1956; Bellman, 1958; Moore, 1959; Elkin, 2020). Section 6.6 presents full details of this algorithm for completeness.

Sending instructions from the leader to the followers using the shortest path tree is straightforward. The leader communicates the action they played to their children, and the children send this action to their children at the end of the next round, and so forth. Using the shortest path tree also makes passing messages to the leader straightforward. Each follower will forward messages from their children to their parent, and send their own message to their parent. Let, $\mathcal{C}_j$ and $\mathcal{P}_j$ denote the children and the parent of agent j on the shortest path tree, respectively. Then, in the t -th round, each follower j will perform the following communication:

- Forward $A_t^j = A_{t-d_{i \rightarrow j}}^i$ to each $z \in \mathcal{C}_j$ where i is the index of the leader.
- Send $M_t^j = (A_t^j, X_t^j) \cup \{M_{t-1}^z\}_{z \in \mathcal{C}_j}$ to $\mathcal{P}_j$.

The leader i only needs to send the action they choose in each round to their children, which corresponds to just the first item in the above list. Thus, the communication complexity, in terms of the number of messages sent in each round, for each agent j is upper bounded by the diameter of the graph.

Note that we could use any other message-passing protocol that satisfies Assumption 6.2. For example, each agent could forward everything they have been sent in the past m rounds. However,

in this case, all messages would have to contain additional information about who originally sent the message and the time the message was sent, such as a unique agent identifier and the round. This extra information is required so that the followers can identify the most recent action sent to them by the leader. Furthermore, this additional information can be used by the leader to ensure that each message is added to the queue only once.

6.4.2 Equivalence Between the Queues and the Environment

The primary objective is to use the black-box algorithm as a theoretical tool that abstracts away the nuances of specific algorithms. Ultimately, the goal is to express the group regret in terms of the regret of the single-agent algorithm operating in the environment. However, the single-agent algorithm is operating in the environment defined by the queues and not the bandit environment faced by the network of agents. Therefore, this section focuses on proving that operating an algorithm in the queues is equivalent to running it in the bandit environment. The minimal requirement is that the expectations are identical. This subsection seeks to prove this equivalence.

Intuitively, the equivalence should hold for the following reasons. First, Assumption 6.1 states that the rewards for each action are independent and identically distributed. Consequently, the order of these rewards can be permuted arbitrarily, provided that the permutation does not depend on the value of the random variable itself. Algorithm 6 effectively reorders the rewards by permuting the indices through a one-to-one mapping based on the lengths of the shortest paths. Therefore, drawing a reward from the queue of an action should simulate playing that action in the environment. Second, the leader is the sole decision-maker and instructs all the other agents in the network to play their most recent action. Therefore, the followers are delayed copies of the leader and do not use any information in their decision-making process that has not already been accounted for by the leader. This design choice eliminates statistical dependencies that could arise in the queues if the followers were to utilise their own decision-making mechanisms.

Proving the equivalence of the expectations in the queued version of the bandit environment and the bandit environment itself requires the introduction of some additional notation. Let $\tilde{A}_s$ denote the s -th action chosen by the single-agent bandit algorithm and define the total number of times it plays each action up to and including the τ -th round as follows:

$$T'_a(\tau) = \sum_{s=1}^{\tau} 1\{\tilde{A}_s = a\} \quad (6.5)$$

for all $(a, \tau) \in \mathcal{A} \times \mathbb{N}$. Recall $\nu = \{P_1, \dots, P_k\}$ is the set of reward distributions that characterise the multi-armed bandit environment faced by the network of agents, as in Assumption 6.1. Then, the cumulative regret of the bandit algorithm after directly interacting with the bandit environment for τ rounds can be defined in the usual manner:

$$R'_\tau(\pi, \nu) = \sum_{a \neq \star} \Delta_a \mathbb{E}_\nu[\tilde{T}_a(\tau)],$$

where $\mathbb{E}_\nu$ is used to make it explicit that the bandit algorithm is interacting directly with the multi-armed bandit environment.

Lemma 6.1 proves that the expectation with respect to the probability measure induced by interacting with the queues is equivalent to the expectation with respect to the probability measure induced by interacting directly with the bandit environment. Or, more simply, the leader creates a simulated single-player version of the environment for their bandit algorithm by maintaining the queues.

Lemma 6.1. *Under Assumption 6.1, QuACK guarantees, for all $\tau \in \mathbb{N}$, that:*

$$R'_\tau(\pi, \nu) := \sum_{a \neq \star} \Delta_a \mathbb{E}_\nu[T'_a(\tau)] = \sum_{a \neq \star} \Delta_a \mathbb{E}[T'_a(\tau)]$$

where $\mathbb{E}$ denotes expectation with respect to the interaction between the network of agents and the multi-armed bandit environment.

Proof. Define $\tilde{\nu} = \{\tilde{P}_1, \tilde{P}_2, \dots, \tilde{P}_k\}$ as the set of distributions that the bandit algorithm faces when taking rewards from the queues. Following the notation of [Lattimore and Szepesvári \(2020\)](#), it is straightforward to write down the density of the bandit algorithm interacting with ν and $\tilde{\nu}$:

$$p_\pi(a_1, x_1, \dots, a_\tau, x_\tau) = \prod_{s=1}^{\tau} \pi(a_s | a_1, x_1, \dots, a_{s-1}, x_{s-1}) p_{a_s}(x_s)$$

$$\tilde{p}_\pi(a_1, x_1, \dots, a_\tau, x_\tau) = \prod_{s=1}^{\tau} \pi(a_s | a_1, x_1, \dots, a_{s-1}, x_{s-1}) \tilde{p}_{a_s}(x_s)$$

where p_a and $\tilde{p}_a$ denote the Radon-Nikodym derivatives of P_a and $\tilde{P}_a$, respectively.

Let $h = (x_1, a_1, \dots, a_\tau, x_\tau) \in \mathcal{H}_\tau = (\mathcal{A} \times \mathbb{R})^\tau$ represent a possible sequence of action-reward pairs that the bandit algorithm could produce within a τ -round interaction in each environment.

Then, the expectations in the statement of the result can be defined more precisely as follows:

$$\mathbb{E}_\nu [T'_a(\tau)] = \int_{h \in \mathcal{H}_\tau} f(h) p_\pi(h) dh$$

and

$$\mathbb{E} [T'_a(\tau)] = \int_{h \in \mathcal{H}_\tau} f(h) \tilde{p}_\pi(h) dh.$$

where $f(h) := \sum_{(a_s, x_s) \in h} 1\{a_s = a\}$. By Assumption 6.1, the rewards for each action-agent pair are independent and identically distributed random variables. Furthermore, Algorithm 6 appends the rewards from the followers to the queues in an order that depends on their shortest path distance. Therefore, the queue for each action contains a re-ordered sequence of independent and identically distributed random variables and, by exchangeability, it follows that:

$$P_a \stackrel{d}{=} \tilde{P}_a$$

The Radon-Nikodym Theorem tells us that the p_a is unique, see Theorem 4.2.2 of [Cohn \(2013\)](#).

Combining $P_a = \tilde{P}_a$ with this well-known result implies that:

$$p_a(x) = \tilde{p}_a(x)$$

for all $x \in \mathbb{R}$. Therefore, $p_\pi(h) = \tilde{p}_\pi(h)$ for all $h \in \mathcal{H}_\tau = (\mathcal{A} \times \mathbb{R})^\tau$ because the policy operating in each bandit environment is identical. Consequently, it follows that:

$$\mathbb{E}_\nu [T'_a(\tau)] = \int_{h \in \mathcal{H}_\tau} f(h) p_\pi(h) dh = \int_{h \in \mathcal{H}_\tau} f(h) \tilde{p}_\pi(h) dh = \mathbb{E} [T'_a(\tau)].$$

Finally $P_a = \tilde{P}_a$ implies that the expected rewards of each action are equivalent in the two environments. Therefore, the sub-optimality gaps are also equivalent. Finally, multiplying each side of the previous display by the sub-optimality gaps and summing over the actions completes the proof. $\square$

6.4.3 Guarantees on the Group Regret

Recall that group regret is defined in terms of the number of times the network of agents plays each action. Thus, Lemma 6.1 is only useful if we can relate the number of times the network of agents plays each action to the number of times the bandit algorithm plays each action in the queues.

Algorithm 6 suggests that the bandit algorithm, operated by the leader, is primarily updated using rewards from the queues. Occasionally, the leader must play an action in the bandit environment if the corresponding queue is empty. Therefore, Equation (6.5) accounts for both the number of times the leader plays this action in the bandit environment and the number of times it takes a reward from the queue for their bandit algorithm.

Suppose $i \in V$ is the index of the leader and let s_t denote the number of simulated rounds the bandit algorithm has completed in the queued version of the environment by the end of the t -th round. Algorithm 6 specifies that the action played by the leader in the t -th round of its interaction with the bandit environment corresponds to the action suggested by the bandit algorithm in the corresponding simulated round:

$$\tilde{A}_{s_t} = A_t^i.$$

Thus, by the end of the final round, the number of times the single-agent bandit algorithm chooses each action can be expressed as follows:

$$T'_a(s_n) = \sum_{t=1}^n \sum_{s=s_{t-1}+1}^{s_t} 1\{\tilde{A}_s = a\}. \quad (6.6)$$

Lemma 6.2 uses Equation (6.6) to relate the number of times the network of agents plays each action to the number of times the bandit algorithm chooses each action.

Lemma 6.2. *Running QuACK with an arbitrary $i \in V$ as the leader guarantees, for all $t \in \mathbb{N}$, that:*

$$\sum_{j=1}^m T_{aj}(t - d_{i \rightarrow j}) \leq T'_a(s_t) + 2 \sum_{w=1}^m d_{i \rightarrow j}$$

where $T_{aj}(t - d_{i \rightarrow j}) = \sum_{s=1}^{t-d_{i \rightarrow j}} 1\{A_s^j = a\}$ denotes the total number of times follower j has played action a by the end of round $t - d_{i \rightarrow j}$.

Proof. Define the last round that the leading agent plays action a in the bandit environment as follows:

$$z = \max \{s \leq t : A_s^i = a\}.$$

Recall s_t denotes the pseudo round counter when the leading agent makes their t -th decision in the bandit environment. Using Equation (6.6) and leveraging the fact that the round in the

previous display is associated with a play of action a gives the following:

$$\begin{aligned}
T'_a(s_t) &= \sum_{s=1}^{s_t} 1\{\tilde{A}_s = a\} \\
&= \sum_{s=1}^{s_z} 1\{\tilde{A}_s = a\} + \sum_{s=s_z+1}^{s_t} 1\{\tilde{A}_s = a\} \\
&= T'_a(s_z) + \sum_{s=s_z+1}^{s_t} 1\{\tilde{A}_s = a\} \\
&\stackrel{(*)}{=} \sum_{j=1}^m T_{aj}(z - d_{i \rightarrow j}) + \sum_{s=s_z+1}^{s_t} 1\{\tilde{A}_s = a\} \\
&\geq \sum_{j=1}^m T_{aj}(z - d_{i \rightarrow j})
\end{aligned} \tag{6.7}$$

where $(*)$ follows from the fact that the leader plays action a in round z and this can only occur if the queue has been emptied. Therefore, the number of feedbacks that the bandit algorithm has received for this action must be equivalent to the group plays.

Now, Equation (6.7) relates the number of times the bandit algorithm has played action a to the number of times the network has chosen this action. Additionally, Algorithm 6 tells us that the j -th agent will play action a in round:

$$z_j = z + d_{i \rightarrow j}$$

where z is the last round that the leader plays action a in the bandit environment. The definition and properties of z_j facilitate an upper bound on the number of plays for any non-leading agent j as follows:

$$\begin{aligned}
T_{aj}(t - d_{i \rightarrow j}) &= \sum_{s=1}^{t-d_{i \rightarrow j}} 1\{A_s^j = a\} \\
&= \sum_{s=1}^{z-d_{i \rightarrow j}} 1\{A_s^j = a\} + \sum_{s=z-d_{i \rightarrow j}+1}^{t-d_{i \rightarrow j}} 1\{A_s^j = a\} \\
&\stackrel{(*)}{\leq} T_{aw}(z - d_{i \rightarrow j}) + \sum_{s=z-d_{i \rightarrow j}+1}^{z+d_{i \rightarrow j}} 1\{A_s^j = a\} \\
&\leq T_{aj}(z - d_{i \rightarrow j}) + 2d_{i \rightarrow j}
\end{aligned} \tag{6.8}$$

Notably, $(*)$ follows from analysing the two possible cases.

- When $t - d_{i \rightarrow j} \geq z_j = z + d_{i \rightarrow j}$, the inequality is actually an equality because z_j is the last round follower j plays this action up to and including the t -th round.
- Otherwise, $t - d_{i \rightarrow j} < z_j = z + d_{i \rightarrow j}$ and extending the summation only include more rounds gives the inequality.

Summing Equation (6.8) over all agents and using Equation (6.7) gives:

$$\begin{aligned} \sum_{j=1}^m T_{aj}(t - d_{i \rightarrow j}) &\leq \sum_{j=1}^m T_{aj}(z - d_{i \rightarrow j}) + 2 \sum_{j=1}^m d_{i \rightarrow j} && \text{(By Equation (6.8))} \\ &\leq T'_a(s_t) + 2 \sum_{w=1}^m d_{i \rightarrow j} && \text{(By Equation (6.7))} \end{aligned}$$

as required. $\square$

Lemmas 6.1 and 6.2 provide crucial insights into the properties of Algorithm 6. Namely, that the queuing mechanism simulates the bandit environment and that the decision-making of the bandit algorithm in the queues can be used to upper bound the number of times the entire network has selected each action. Given these results, it is possible to upper bound the group regret by the regret of the bandit algorithm in the queues, plus an additive graph-dependent quantity that measures the speed of information flow to and from the leader.

Theorem 6.1. *Under Assumptions 6.1 and 6.2, the group regret of QuACK run with single-player bandit algorithm π and leading agent $i \in V$ is bounded by:*

$$R_n(G, \nu) \leq R'_{mn}(\pi, \nu) + \left(3 \sum_{j=1}^m d_{i \rightarrow j} \right) \sum_{a=1}^k \Delta_a.$$

Proof. Firstly, we upper bound the group plays for action $a \in \mathcal{A}$ at the end of the final round as:

$$\begin{aligned} \sum_{j=1}^m T_{aj}(n) &= T_{ai}(n) + \sum_{j \neq i} T_{aj}(n) \\ &= T_{ai}(n) + \sum_{j \neq i} \sum_{t=1}^n 1\{A_t^j = a\} \\ &= T_{ai}(n) + \sum_{j \neq i} \sum_{t=1}^{n-d_{i \rightarrow j}} 1\{A_t^j = a\} + \sum_{j \neq i} \sum_{t=n-d_{i \rightarrow j}+1}^n 1\{A_t^j = a\} \\ &= T_{ai}(n) + \sum_{j \neq i} T_{aj}(n - d_{i \rightarrow j}) + \sum_{j \neq i} \sum_{t=n-d_{i \rightarrow j}+1}^n 1\{A_t^j = a\} \\ &\leq T_{ai}(n) + \sum_{j \neq i} T_{aj}(n - d_{i \rightarrow j}) + \sum_{j \neq i} d_{i \rightarrow j} \end{aligned}$$

$$= \sum_{j=1}^m T_{aj} (n - d_{i \rightarrow j}) + \sum_{j \neq i} d_{i \rightarrow j} \quad (6.9)$$

where the last equality follows from the fact that $d_{i \rightarrow i} = 0$. Algorithm 6 has the leader append each message to the queue exactly once, which implies that $s_n < mn$ almost surely. Combining Equation (6.9) with Lemma 6.2 and $s_n < mn$ gives an upper bound on the number of times the entire network plays action a by the number of times π has played this action in the queues:

$$\begin{aligned} \sum_{j=1}^m T_{aj} (n) &\leq \sum_{j=1}^m T_{aj} (n - d_{i \rightarrow j}) + \sum_{j \neq i} d_{i \rightarrow j} \\ &\leq T'_a (mn) + 3 \sum_{j \neq i} d_{i \rightarrow j}. \end{aligned} \quad (6.10)$$

Plugging Equation (6.10) into the definition of the group regret gives:

$$\begin{aligned} R_n (G, \nu) &= \sum_{a \neq \star} \Delta_a \mathbb{E} \left[\sum_{j=1}^m T_{aj} (n) \right] \\ &\leq \sum_{a \neq \star} \Delta_a \mathbb{E} [T'_a (mn)] + \left(3 \sum_{j \neq i} d_{i \rightarrow j} \right) \sum_{a \neq \star} \Delta_a \\ &= R'_{mn} (\pi, \nu) + \left(3 \sum_{j \neq i} d_{i \rightarrow j} \right) \sum_{a \neq \star} \Delta_a \end{aligned}$$

where the final line follows from Lemma 6.1. □

Theorem 6.1 suggests that the best-positioned leader is the agent minimising the sum of shortest paths:

$$i_\star = \arg \min_{i \in V} \sum_{j=1}^m d_{i \rightarrow j}.$$

which is the graph median. This intuitively makes sense. Indeed, minimising the sum of the shortest paths will minimise the total delay in sending and receiving instructions and feedback from the followers. Section 6.6 presents an approach for finding this agent using well-known distributed algorithms. Nevertheless, Assumption 6.2 guarantees that the worst-case graph-dependence is always upper bounded by:

$$\sum_{j=1}^m d_{i \rightarrow j} \leq d(m-1)$$

which holds for any choice of leading agent. Theorem 6.1 holds for any choice of bandit algorithm

and makes weak assumptions on the rewards. This is in contrast to existing works, where it is common to analyse a specific bandit algorithm and make stronger assumptions on the reward distributions, such as having a particular parametric form or subgaussian tails. Section 6.5 shows that the guarantees presented in this subsection are competitive with the case-specific analyses that exist in the literature when providing QuACK with the corresponding single-agent bandit algorithm.

6.5 Instances of QuACK

Theorem 6.1 holds under the assumption that the rewards are conditionally independent given the actions and the expected reward for each of the actions is finite. Therefore, we can apply our reduction in numerous bandit environments by choosing an appropriate single-agent algorithm. This section presents a non-exhaustive selection of bandit problems that satisfy Assumption 6.1.

Algorithm 6 will be provided with a specific bandit algorithm for each of the considered bandit problems. Then, Theorem 6.1 will be applied to obtain an upper bound on the group regret. The specific choices of bandit algorithms are made to facilitate a comparison with the existing literature. However, any other provably efficient single-player algorithm could be combined with our reduction to get similar results.

6.5.1 Standard Bandits

The standard multi-armed bandit problem consists of playing an action and receiving a reward drawn independently from a distribution that depends only on the chosen action (Lattimore and Szepesvári, 2020). Typically, it is assumed that each distribution has bounded support or subgaussian tails. Thus, Assumption 6.1 holds. For ease of presentation, it will be assumed that the bandit environment contains distributions with support bounded in $[0, 1]$. Corollaries 6.1 and 6.2 present the group regret bound for our algorithm when the leader uses UCB1 (Auer et al., 2002a) and MOSS (Audibert and Bubeck, 2009), respectively.

Corollary 6.1. *Running QuACK with UCB1 of Auer et al. (2002a) and an arbitrary leader $i \in V$ guarantees that:*

$$R_n(G, \nu) \leq \sum_{a \neq \star} \frac{8 \ln(mn)}{\Delta_a} + \left(3 + 3 \sum_{j=1}^m d_{i \rightarrow j} \right) \sum_{a \neq \star} \Delta_a.$$

Proof. Theorem 1 of [Auer et al. \(2002a\)](#) provides an upper bound on the regret of the upper confidence bound algorithm over a τ -round interaction with the bandit environment:

$$R'_\tau(\text{UCB1}, \nu) \leq \sum_{a \neq \star} \frac{8 \ln(\tau)}{\Delta_a} + 3 \sum_{a \neq \star} \Delta_a. \quad (6.11)$$

Combining this upper bound with Theorem 6.1 and taking $\tau = mn$ gives the result. $\square$

Corollary 6.2. *Running QuACK with MOSS of [Audibert and Bubeck \(2009\)](#) and an arbitrary leader $i \in V$ guarantees that:*

$$R_n(G, \nu) \leq 49\sqrt{mnk} + \left(3 \sum_{j=1}^m d_{i \rightarrow j} \right) \sum_{a \neq \star} \Delta_a.$$

Proof. Theorem 5 of [Audibert and Bubeck \(2009\)](#) provides an upper bound on the regret of the MOSS algorithm over a τ -round interaction with the bandit environment:

$$R'_\tau(\text{MOSS}, \nu) \leq 49\sqrt{\tau k}.$$

Combining this upper bound with Theorem 6.1 and taking $\tau = mn$ gives the result. $\square$

Equation (6.2) presents the minimax lower bound for the multi-agent version of the standard multi-armed bandit problem. Corollary 6.2 shows that QuACK is capable of achieving this minimax rate up to an additive term that is independent of the horizon. Conversely, prior works miss the minimax lower bound by at least a poly-logarithmic factor involving the horizon and the number of agents ([Landgren et al., 2016](#); [Martínez-Rubio et al., 2019](#); [Lalitha and Goldsmith, 2021](#)). Note that to achieve this minimax rate, the leader must know the horizon n and the number of agents m in the network. However, we could still achieve the minimax rate without prior knowledge of these parameters. This can be done by using the anytime version of the MOSS algorithm ([Degenne and Perchet, 2016](#)).

[Martínez-Rubio et al. \(2019\)](#) analyse a multi-agent extension of UCB1 ([Auer et al., 2002a](#)). This is not the only extension of this well-known algorithm to the multi-agent setting ([Landgren et al., 2016](#)). However, [Martínez-Rubio et al. \(2019\)](#) offer the best theoretical guarantees, and prove

the upper bound on the group regret:

$$R_n(G, \nu) \leq \sum_{a \neq \star} \frac{8\eta \left(1 + \frac{\epsilon}{2}\right) \ln(mn)}{\Delta_a} + C_G \sum_{a \neq \star} \Delta_a$$

where $\eta > 1$ and $\epsilon \in (0, 1)$ are tunable hyperparameters of their algorithm. Further, C_G is a graph-dependent quantity with the following definition:

$$C_G = \left\lceil \frac{6m \ln\left(\frac{2m}{\epsilon}\right)}{\sqrt{2 \ln|\lambda|^{-1}}} \right\rceil + m + 4$$

where λ is the second largest singular value of the communication matrix they use for the gossiping procedure. Corollary 6.1 shows a strict improvement in the leading-order term. However, comparing the graph-dependence is a little more challenging. Therefore, we will focus on special cases. For each of the special cases, we will utilise the gossip matrix specified in Equation (6.4), as recommended ([Martínez-Rubio et al., 2019](#)).

Star Graphs. The star graph consists of $m - 1$ vertices connected to a single central vertex. Here, it is possible to evaluate the spectrum of the communication matrix exactly. Notably, the spectrum of $L := D - M$ is known for this graph ([Chung, 1997](#)):

$$\lambda(D - M) = \begin{cases} m & \text{with multiplicity 1} \\ 1 & \text{with multiplicity } m - 2 \\ 0 & \text{with multiplicity 1.} \end{cases}$$

Therefore, the spectrum of the communication matrix W can be found by plugging in these quantities:

$$\lambda(W) = 1 - \frac{\lambda(D - M)}{m} = \begin{cases} 0 & \text{with multiplicity 1} \\ \frac{m-1}{m} & \text{with multiplicity } m - 2 \\ 1 & \text{with multiplicity 1} \end{cases}$$

which gives us $\lambda = \lambda_2(W) = m^{-1}/m$. Plugging this into the graph-dependent term of [Martínez-Rubio et al. \(2019\)](#) yields:

$$C_G + m + 4 = \left\lceil \frac{6m \ln\left(\frac{m}{\epsilon}\right)}{\sqrt{2 \ln\left(\frac{m}{m-1}\right)}} \right\rceil + m + 4$$

$$\begin{aligned}
&\geq \frac{6m \ln(m)}{\sqrt{2 \ln\left(\frac{m}{m-1}\right)}} + m + 4 \\
&> 7m + 4
\end{aligned}$$

where the final inequality follows from the fact that $m \geq 2$ is required for the multi-agent setting. Running Algorithm 9 from Section 6.6.3, which selects the agent who is best-positioned to minimise the group regret according to Theorem 6.1, will choose the central vertex as the leading agent. Therefore, the length of the shortest path from every follower to the leader is 1 and this will make the graph dependence in our algorithm:

$$3 + 3 \sum_{j=1}^m d_{i \rightarrow j} = 3 + 3(m-1) = 3m$$

which is strictly smaller than that of [Martínez-Rubio et al. \(2019\)](#). Without optimising the leader, the graph-dependence in our algorithm would become $6m - 6$, which demonstrates that the graph-dependence of our algorithm is better even with a poor initialisation.

Regular Graphs. Graphs are called δ -regular if every agent has exactly δ neighbours, e.g. $|N_v| = \delta$ for all $v \in V$. For these graphs, the expression for W simplifies and allows us to obtain an expression for the spectral term found in the graph-dependent term of [Martínez-Rubio et al. \(2019\)](#):

$$W = \frac{I + M}{1 + \delta} \implies \lambda := \lambda_2(W) = \frac{1 + \lambda_2(M)}{1 + \delta}$$

[Chung \(1989\)](#) provide an upper bound on the diameter d for regular graphs, which we can substitute into the graph-dependent term in the group regret of QuACK-UCB:

$$\begin{aligned}
3 + 3 \sum_{w=1}^m d_{vw} &\leq 3 + 3d(m-1) \\
&\leq 3 + \left\lceil \frac{3(m-1) \ln(m-1)}{\ln\left(\frac{1+\delta}{|1+\lambda_2(M)|}\right)} \right\rceil \\
&= 3 + \left\lceil \frac{3(m-1) \ln(m-1)}{\ln|\lambda|^{-1}} \right\rceil \\
&= 3 + \left\lceil \frac{6(m-1) \ln(m-1)}{\sqrt{2 \ln|\lambda|^{-1}} \sqrt{2 \ln|\lambda|^{-1}}} \right\rceil \\
&\stackrel{(a)}{<} 3 + \left\lceil \frac{6(m-1) \ln\left(\frac{2m}{\epsilon}\right)}{\sqrt{2 \ln|\lambda|^{-1}} \sqrt{2 \ln|\lambda|^{-1}}} \right\rceil
\end{aligned}$$

$$\begin{aligned}
&= 3 + \frac{C_G - 6 \ln(m-1)}{\sqrt{2 \ln|\lambda|^{-1}}} \\
&\leq 3 + \frac{C_G}{\sqrt{2 \ln|\lambda|^{-1}}}
\end{aligned}$$

where (a) follows from the fact that $\ln(2m/\epsilon) > \ln(m-1)$ for $m \geq 2$ and $\epsilon \in (0, 1)$. This establishes that we can upper bound the graph-dependence in our theoretical guarantees by a function involving the graph-dependent term found in that of [Martínez-Rubio et al. \(2019\)](#). Notably, better dependence for our algorithm is guaranteed whenever:

$$\sqrt{2 \ln|\lambda|^{-1}} \geq 1 \implies |\lambda| \geq \exp\left(-\frac{1}{2}\right)$$

Generally, this suggests that the graph-dependence of Algorithm 6 is better for δ -regular graphs where the communication matrix has large enough values for $|\lambda|$.

6.5.2 Heavy-Tailed Bandits

[Bubeck et al. \(2013a\)](#) introduced the single-agent heavy-tailed bandit problem. The decision-making process is exactly the same as the standard bandit setting. However, the reward distributions only have finite moments of order $1 + \epsilon$ where $\epsilon > 0$ controls the heaviness of the tails. Therefore, Assumption 6.1 holds because the expected values of each distribution are finite. Thus, QuACK can be used for the multi-agent heavy-tailed bandit problem.

[Bubeck et al. \(2013a\)](#) propose the *robust upper confidence bound algorithm* as a generic solution for the single-agent setting. This algorithm replaces the empirical mean with a robust estimator of the expected value, such as the truncated mean, median-of-means, or Catoni's M ([Bickel, 1965](#); [Alon et al., 1999](#); [Catoni, 2012](#)). Indeed, QuACK can be used with any instantiation of the robust upper confidence bound algorithm. Corollary 6.3 presents a guarantee on the group regret when the leading agent uses the truncated mean estimator.

Corollary 6.3. *Running QuACK with Robust UCB of [Bubeck et al. \(2013a\)](#) using the truncated mean and an arbitrary leader guarantees that:*

$$R_n(G, \nu) \leq \sum_{a \neq \star} 8 \left(\frac{4\sigma}{\Delta_a} \right)^{\frac{1}{\epsilon}} \ln(mn) + \left(5 + 3 \sum_{j=1}^m d_{i \rightarrow j} \right) \sum_{a \neq \star} \Delta_a$$

where $\mathbb{E}_{X \sim P_a}[|X|^{1+\epsilon}] \leq \sigma$ for all $a \in \mathcal{A}$.

[Dubey and Pentland \(2020a\)](#) study the multi-agent heavy-tailed bandit problem and devise two algorithms based on the truncated-mean estimator. Their best algorithm achieves the following guarantee:

$$R_n(G, \nu) \lesssim \alpha(G_\gamma) \sum_{a \neq \star} \left(\frac{\sigma}{\Delta_a} \right)^{\frac{1}{\epsilon}} \ln(n) + (m\gamma \cdot \alpha(G_\gamma) + \alpha(G_\gamma) + m) \sum_{a \neq \star} \Delta_a$$

where $\lesssim$ removes numeric constants, γ controls the communication radius, and $\alpha(G_\gamma) \in [1, m]$ denotes the independence number of the γ -th power of the graph. Thus, we obtain a smaller leading order term whenever:

$$n^{\alpha(G_\gamma)-1} \geq m$$

which occurs if $\alpha(G_\gamma) > 1$ and $m < n$ so that the number of agents is smaller than the number of rounds. Specifying $\gamma \geq d$ as the input parameter will yield an independence number of 1 in the bound of [Dubey and Pentland \(2020a\)](#). However, their analysis is specific to the robust upper confidence bound algorithm and the truncated mean estimator, whereas Theorem 6.1 is strictly more general.

Next, we will compare the additive graph-dependent terms. [Dubey and Pentland \(2020a\)](#) have a term that displays a trade-off between the input parameter γ and the independence number. Generally, increasing γ will decrease the independence number and decreasing γ will increase the independence number. Therefore, we expect their additive term to be comparable to ours in many cases. For example, choosing $\gamma = d$ will minimise the leading order term in the regret bound and yields an additive graph dependence of:

$$m\gamma \cdot \alpha(G_\gamma) + \alpha(G_\gamma) + m = md + m + 1$$

Conversely, Corollary 6.3 shows that our additive graph-dependence can be upper bounded as follows:

$$5 + 3 \sum_{j=1}^m d_{i \rightarrow j} \leq 5 + 3md - 3m$$

which is more or less of the same order.

6.5.3 Duelling Bandits

[Yue et al. \(2009\)](#) introduced the single-agent duelling bandit problem. Here, feedback from the environment is the outcome of a duel between two actions. Modifying Algorithm 6 for the multi-agent duelling bandit problem amounts to maintaining a queue for each pair of actions: $\mathcal{A}_+ = \mathcal{A} \times \mathcal{A}$. Now, each agent will communicate the pair of actions and the outcome of the duel to their neighbours.

Assumption 6.1 holds since the outcomes of the duels are assumed to depend only on the pair of actions chosen in each round. Recall that we define the group regret relative to a fixed optimal action. Therefore, Theorem 6.1 applies to solution concepts, such as the Condorcet, Copeland, and Borda winners ([Zoghi et al., 2014, 2015a; Jamieson et al., 2015](#)). Corollary 6.4 presents an upper bound on the group regret when we pair QuACK with the Relative Upper Confidence Bound algorithm ([Zoghi et al., 2014](#)), which is designed under the assumption that the Condorcet winner exists.

Corollary 6.4. *Running QuACK with the Relative Upper Confidence Bound algorithm of [Zoghi et al. \(2014\)](#) and an arbitrary leader $i \in V$ guarantees that:*

$$R_n(G, \nu) \lesssim \sum_{a \neq \star} \frac{\ln(mn)}{\tilde{\Delta}_a} + \left(\sum_{j=1}^m d_{i \rightarrow j} \right) \sum_{a \neq \star} \tilde{\Delta}_a$$

where $\lesssim$ omits numeric constants, $\tilde{\Delta}_a = P(a_\star \succ a) - 1/2$ denotes the sub-optimality gap of action a in the duelling bandit setting and the summation is over the actions $\mathcal{A}$.

[Raveh et al. \(2024\)](#) develop several algorithms for the multi-agent duelling bandit problem. Their best algorithm builds on the relative upper confidence bound algorithm and achieves the following upper bound on the group regret:

$$R_n(G, \nu) \lesssim \chi(G_\gamma) \sum_{a \neq \star} \frac{\ln(n)}{\tilde{\Delta}_a} + m^2 k^2 (1 + \ln(1 + |N_\star^\gamma|)) \tilde{\Delta}_{\max} + (1 + \gamma) km \tilde{\Delta}_{\max}$$

where $\chi(G_\gamma) \in [1, m]$ denotes the clique covering number of the γ -th power of the graph and γ is a hyperparameter that controls the communication radius. Comparing the leading-order terms reveals that we obtain a strictly better leading-order term whenever:

$$n^{\chi(G_\gamma)-1} \geq m$$

Therefore, our leading-order term is smaller whenever $\chi(G_\gamma) > 1$. Specifying $\gamma \geq d$ as the input parameter will yield a clique covering number of 1 in the bound of [Raveh et al. \(2024\)](#). However, their analysis is specific to the relative upper confidence bound algorithm, whereas Theorem 6.1 is strictly more general. [Raveh et al. \(2024\)](#) have an additive graph-dependent term that is strictly larger. Notably, their graph dependence is always quadratic in the network size, whereas ours scales as md , where $d < m$ in connected graphs.

6.5.4 Local Differential Privacy

Privacy is a topic of growing interest in the machine learning community and seeks to protect private information within datasets. [Ren et al. \(2020\)](#) study *local differential privacy* (LDP) in the bandit setting, where the environment uses a mechanism to make the rewards private before sending them to the agent. Formally, in the multi-agent setting, the only difference is that the rewards each agent receives are privatised:

$$X_t^v = f_\epsilon(X) \text{ for } X \stackrel{\text{iid}}{\sim} P_{A_t^v}$$

where f_ϵ denotes the privatising mechanism that ensures ϵ -LDP of the rewards before handing them to the agent. Procedures for privatising rewards include adding Bernoulli or Laplace noise to the rewards, and proving the mechanism provides the specified level of privacy is done independently of the bandit algorithm. See Lemmas 2 and 5 of [Ren et al. \(2020\)](#). Therefore, it is possible to directly use these mechanisms in the multi-agent setting and guarantee that the rewards are ϵ -LDP. Furthermore, Assumption 6.1 is satisfied in this setting because the curator adds independent and identically distributed noise to the rewards.

Corollary 6.5. *Running QuACK with LDP-UCB-L of [Ren et al. \(2020\)](#) and an arbitrary leader is ϵ -LDP and guarantees:*

$$R_n(G, \nu) \lesssim \sum_{a \neq \star} \frac{1}{\epsilon^2} \frac{\ln(mn)}{\Delta_a} + \left(\sum_{j=1}^m d_{i \rightarrow j} \right) \sum_{a \neq \star} \Delta_a.$$

where $\lesssim$ removes numeric constants.

Notably, Corollary 6.5 provides the first guarantees for multi-agent multi-armed bandits with local differential privacy. Indeed, the same idea can be used to derive multi-agent algorithms for other bandit problems too, as long as the environment satisfies Assumption 6.1.

6.6 Implementation Details

Section 6.4 presents QuACK assuming that the leader has been elected and a fixed shortest path has been established for the message passing protocol. However, these quantities need not be known in advance and can be computed in a distributed manner using well-known algorithms.

6.6.1 Shortest Path Tree

Algorithm 6 requires that each agent knows their parents and children on the shortest path tree, and is presented assuming that these have been pre-computed. Constructing the shortest path tree rooted at a fixed vertex amounts to solving the single-source shortest path problem. Traditionally, this problem is solved using the Bellman-Ford algorithm in non-distributed systems (Ford, 1956; Bellman, 1958). However, using this algorithm would require the existence of an agent who possesses knowledge of the entire network topology. Therefore, the distributed variant of this algorithm is presented (Bertsekas and Gallager, 1992). Here, each agent knows only their neighbours. Algorithm 7 presents the pseudo-code for the Distributed Bellman-Ford algorithm. Briefly, this algorithm takes a source agent as input, and each agent keeps track of their distances from the source and their parent on the shortest path tree.

Algorithm 7 Distributed Bellman-Ford (DBF)

Input. Index of a source agent $i \in V$
Set $d_{i \rightarrow i}^1 = 0$ and $d_{i \rightarrow j}^1 = \infty$ for $j \neq i$
Set $\mathcal{P}_j = \emptyset$ for each $j \in V$ to initialise their parent.
for $t \in \{1, 2, \dots, m\}$ **do**
 Each $j \in V$ sends $d_{i \rightarrow j}^t$ to $z \in N_j$
 Each $j \in V$ receives $d_{i \rightarrow z}^t$ from $z \in N_j$
 Each $j \in V$ updates their distance from the source

$$d_{i \rightarrow j}^{t+1} = \min_{z \in N_j \cup \{j\}} \{d_{i \rightarrow z}^t + 1\{z \neq j\}\}$$

Each $j \in V$ updates their parent:

$$\mathcal{P}_j = \begin{cases} \arg \min_{z \in N_j} d_{i \rightarrow z}^t + 1 & \text{if } d_{i \rightarrow j}^{t+1} < d_{i \rightarrow j}^t \\ \mathcal{P}_j & \text{if } d_{i \rightarrow j}^{t+1} = d_{i \rightarrow j}^t \end{cases}$$

end for

Each $j \in V$ sends their identifier j to their parent $\mathcal{P}_j$.

Each $j \in V$ receives the identifiers from their children and creates their set of children:

$$\mathcal{C}_j = \{z \in V : \mathcal{P}_z = j\}$$

After Algorithm 7 terminates, each agent will know their children and parent on the shortest path tree rooted at the index of the input agent. These are exactly the quantities required to implement the message-passing scheme described in Section 6.4.1.

6.6.2 Leader Election

Leader Election (LE) is a well-known problem in distributed computing (Santoro, 2006; Fokkink, 2013). Here, we want the network to unanimously agree upon a single agent as a leader. Generally, there are impossibility results for anonymous networks (Fokkink, 2013). Therefore, it is assumed that each agent has a unique identifier. Wang et al. (2020) devise a simple leader election scheme that requires at most m iterations, where m is the number of agents. Algorithm 8 presents a slight modification of their procedure that will ease the exposition of finding the optimal leader in the next subsection.

Algorithm 8 Leader Election (LE)

```

Each  $i \in V$  sets  $f_i$  arbitrarily.
Section 6.6.3 has each agent set  $f_i$  to their sum of shortest paths.
Each  $i \in V$  saves their initial value  $I_i = (i, f_i)$ 
Each  $i \in V$  creates their state value  $S_i^1 = (i, f_i)$ 
for  $t \in \{1, 2, \dots, m\}$  do
  Each  $i \in V$  receives  $S_j^t$  from  $j \in N_i$ 
  if  $\exists j \in N_i$  such that  $f_j < f_i$  then
    Update  $S_i^{t+1} = (j, f_j)$ 
  else if  $\exists j \in N_i$  such that  $f_j = f_i$  then
    Update  $S_i^{t+1} = (\min\{i, j\}, f_i)$ 
  else
     $S_i^{t+1} = S_i^t$ 
  end if
end for
if  $I_i = S_i^{m+1}$  then
  Agent  $i$  is the leader.
end if

```

Running Algorithm 8 requires at most m iterations because every agent will have seen the identifier of every other agent in the graph at the end of this iteration. Furthermore, Algorithm 8 will always terminate with exactly one leader. To see this, it is sufficient to consider two possible cases.

- *Unique Minimum.* Here, there exists exactly one agent i who possesses the smallest f_i value. Therefore, they never get to update their state, and the network elects them as the leader.
- *Non-Unique Minimum.* Let $f = \min_{i \in V} f_i$ and suppose multiple agents attain this value.

Algorithm 8 implements an index-based tie-breaking rule to handle this case. Notably, agents with $f_i = f$ will eventually receive an $f_j = f$ and they will proceed to check if their index is minimal. Since each agent is assumed to have a unique index, there will be exactly one agent who possesses $f_i = f$ with the minimum index. This agent will never update their state, and the network will elect them as the leader.

6.6.3 Optimising the Leader

Combining Sections 6.6.1 and 6.6.2 allows the best-positioned agent in the network to be found. Algorithm 9 presents the pseudo-code for finding and electing the leader.

Algorithm 9 Optimise the Leader

```

for  $i \in V = \{1, 2, \dots, m\}$  do
  Compute Sum of Shortest Paths
  Execute DBF( $i$ )
    Each agent now knows  $d_{i \rightarrow j}$  (distance from source  $i$ )
    Each agent now knows  $\mathcal{P}_j$  (their parent on the shortest path tree with source  $i$ )
  Each  $j \in V$  creates message  $M_j^1 = (j, d_{i \rightarrow j})$ 
  for  $t \in \{1, 2, \dots, m\}$  do
    Each  $j \in V$  sends  $M_j^t$  to  $\mathcal{P}_j$ 
    Each  $j \in V$  updates  $M_j^{t+1} = M_j^t \cup \{M_z^t\}_{z \in \mathcal{C}_j}$ 
  end for
  Agent  $i$  calculates  $f_i = \sum_j d_{i \rightarrow j}$  from the messages.
end for
Use Sum of Shortest Paths as  $f_i$  in Leader Election
  Each agent now knows their value for  $f_i = \sum_{w=1}^m d_{i \rightarrow j}$ 
  Execute LE to elect the leader with minimal sum of shortest paths.

```

Briefly, this procedure starts with each agent calling Algorithm 7. Afterwards, several rounds of message passing are required so that agent i can calculate the sum of shortest paths: f_i . Finally, Algorithm 8 performs leader election with f_i equal to the sum of shortest paths.

6.7 Experimental Results

Theorem 6.1 and its corollaries suggest that our algorithm will be competitive with existing multi-agent approaches when paired with a comparable single-agent bandit algorithm. Naturally, we seek to verify these theoretical insights with experimental evidence.

6.7.1 Standard Bandits

Here, the bandit environment consists of ten actions where each reward distribution is Bernoulli with $\mu_1 = 0.5$ and $\mu_2 = \dots = \mu_{10} = 0.45$. Following existing works, we deployed each of the

algorithms on the cycle and grid graphs. Additionally, the star graph is included on top of the standard settings to verify our theoretical insights. The results are averaged over 100 independent runs and we represent uncertainty by shading between the 2.5-th and 97.5-th quantiles.

QuACK is initialised with UCB1 (Auer et al., 2002a) and Thompson Sampling (Thompson, 1933). This allows for a fair comparison with existing methods that extend these single-agent algorithms to the multi-agent setting. Specifically, QuACK with UCB1 against Coop-UCB (Landgren et al., 2016) and DDUCB (Martínez-Rubio et al., 2019), and QuACK with Thompson Sampling against DecTS (Lalitha and Goldsmith, 2021).

QuACK is initialised as described in Section 6.6. Landgren et al. (2016), Martínez-Rubio et al. (2019) and Lalitha and Goldsmith (2021) all present gossip-based algorithms. Thus, they all require the specification of a gossip matrix, and the general consensus amongst the authors is to select the matrix based on Equation (6.4). These algorithms have other hyperparameters and the specific values used in our experiments are listed below.

- Coop-UCB (Landgren et al., 2016): $\gamma = 1.01$ controls the exploration and was chosen based on the empirical results of existing work (Martínez-Rubio et al., 2019; Lalitha and Goldsmith, 2021).
- DDUCB (Martínez-Rubio et al., 2019): $\eta = 2$ and $\epsilon = 1/22$ control the exploration and the approximation error, respectively. These values were chosen based on Theorem 3.2 of Martínez-Rubio et al. (2019).
- DecTS (Lalitha and Goldsmith, 2021): $\eta = m$ is the learning rate and was chosen based on Theorem 1 of Lalitha and Goldsmith (2021).

Figure 6.2 compares the group regret of each algorithm across various graph structures with $m = 196$ agents. For UCB-based algorithms, we observe that QuACK significantly outperforms Coop-UCB and DDUCB. These empirical findings support the claims that were made based on the theoretical results in Section 6.5. For Thompson Sampling algorithms, QuACK is competitive with existing work on the cycle and grid structures, and performs significantly better on the star structure, also supporting our theoretical claims. These empirical findings are consistent over the various graph sizes. See Figures B.15-B.17 of Appendix B.2.

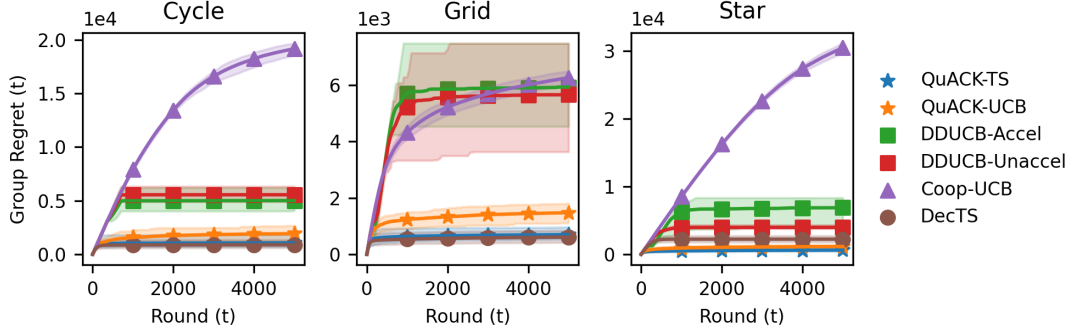


Figure 6.2: The group regret of various cooperative multi-armed bandit algorithms on the cycle, grid and star graphs with $m = 196$ agents.

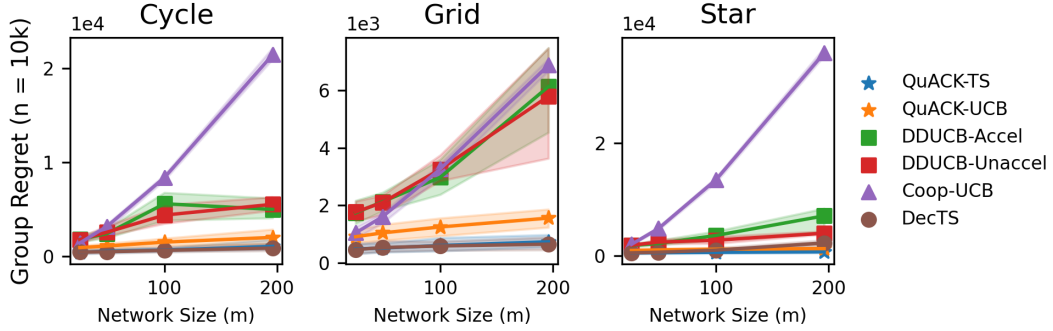


Figure 6.3: The final round group regret as a function of the network size in the cycle, grid and star graphs.

Figure 6.3 visualises the group regret at the end of the final round as a function of the network size for each graph structure. Notably, it shows that Coop-UCB and DDUCB have group regrets that display an unpleasant scaling with the number of agents in the network. Conversely, the group regret of QuACK-UCB, QuACK-TS and Dec-TS have a shallower gradient, indicating a better dependence on the network size and topology.

6.7.2 Heavy-Tailed Bandits

Following [Dubey and Pentland \(2020a\)](#), the experiments in this subsection utilise standard α -stable densities for the reward distributions. Specifically, $\alpha = 1.9$ and the location of the densities were chosen such that:

$$\mu_a = \begin{cases} 0.7 & \text{if } a = 1 \\ 0.4 & \text{if } a \neq 1 \end{cases}$$

where $\mathcal{A} = \{1, 2, \dots, 5\}$.

QuACK was initialised as described in Section 6.6. Further, the single-agent bandit algorithm was chosen to be the Robust UCB algorithm with the truncated mean estimator (Bubeck et al., 2013a) to facilitate a fair comparison with existing algorithms. Dubey and Pentland (2020a) obtain the best theoretical guarantees for CMP-UCB and this algorithm partitions the graph into disjoint subsets. Each of these subsets has one leader that uses the robust upper confidence bound algorithm with the truncated mean estimator. Following their guidance, the leaders were selected so that they form the maximal weighted independent set of each graph. CMP-UCB was ran with 1, 2 and 4 leaders on each graph. Notably, with 1 leader CMP-UCB should outperform QuACK. With ≥ 2 leaders, QuACK is expected to perform best.

Our experiments in the heavy-tailed setting consider only the cycle network structure with $m \in \{16, 36, 64\}$ agents. These experiments use a smaller number of agents and fewer rounds because the computational complexity of computing the truncated mean estimator grows linearly with the number of reward samples for each action. This implicitly depends on the network size in the multi-agent setting.

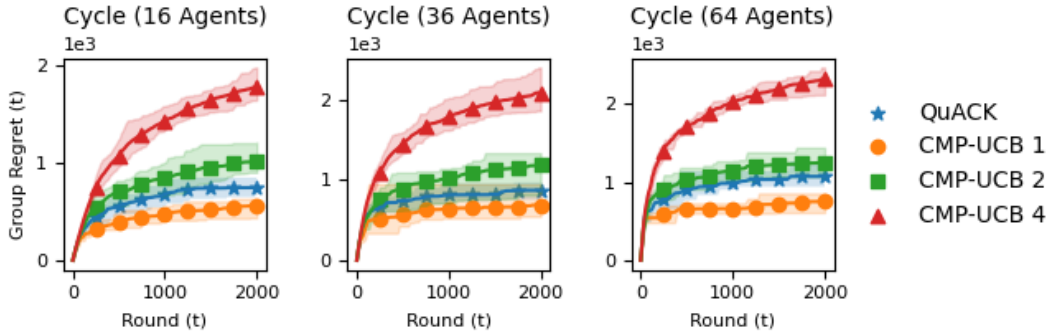


Figure 6.4: The final round group regret as a function of the network size on cycle graphs of varying sizes.

Figure 6.4 displays the group regret for the various cycle and grid graphs. From Section 6.5, QuACK is expected to perform best when CMP-UCB has more than one leader. Notably, the experimental results align exactly with the theoretical predictions.

6.8 Conclusion

This chapter presents a generic black-box framework for extending any single-agent bandit algorithm to the multi-agent setting. Under mild assumptions on the bandit environment, we show that our black-box algorithm preserves the theoretical guarantees of the chosen algorithm, seamlessly transferring its regret bounds from the single-agent case to the multi-agent scenario. Furthermore, this generality does not appear to sacrifice empirical performance.

For the standard bandit setting, Algorithm 6 can be combined with any minimax-optimal single-agent algorithm to achieve minimax-optimal performance in the multi-agent case, up to an additive graph-dependent term. However, this lower bound is tight only in the complete graph setting, where each agent can communicate with every other agent in the network. This limitation highlights an important gap: the current lower bound does not fully capture the effect of graph structure and communication delays. Establishing graph-dependent lower bounds remains an open problem.

Finally, an interesting extension of Algorithm 6 would be to allow each agent to run an independent bandit algorithm alongside the queuing mechanism. However, establishing theoretical guarantees for this variant would require addressing the intricate dependencies introduced by information sharing. In particular, the resulting probability measure of the chosen algorithm may deviate from those in the original bandit environment. A natural first step in this direction would be to analyse the case where all agents employ the same bandit algorithm.

Chapter 7

Pairwise Comparisons

This chapter contains preliminary research that came out of an academic visit to Tübingen University. This was conducted under the supervision of Claire Vernade, Ciara Pike-Burke and Sarah Filippi.

7.1 Introduction

Traditionally, bandit and reinforcement learning problems rely on reward-based feedback. Here, the rewards assign each action or sequence of actions a numerical value that guides the decision-maker toward optimal behaviour. Providing the agent with reward-based feedback has led to the development of algorithms that are highly successful across various domains, such as data-centre cooling ([Luo et al., 2022](#)), game-playing ([Silver et al., 2016](#)), healthcare ([Trella et al., 2024](#)) and e-commerce ([Zhang et al., 2024](#)), just to name a few.

However, reward-based feedback is often impractical or inefficient in real-world applications. This is particularly true in robotics, where rewards often require human input. Here, the reward for selecting an action is often hand-engineered to encourage specific behaviours. Designing these reward functions is labour-intensive and prone to unintended consequences. These unintended consequences occur when the agent selects actions in a way that technically maximises their reward, but the actions deviate from the intended behaviour. This is commonly referred to as reward hacking ([Amodei et al., 2016](#); [Skalse et al., 2022](#); [Knox et al., 2023](#)).

Providing reward-based feedback is not the only signal that an agent can use to learn near-optimal behaviour. One alternative is to have the agent pick a pair of actions and ask the environment which of these actions they prefer (Yue et al., 2009). For example, this could involve the agent selecting a pair of movies and asking the environment, which could be a human in this case, for their preference. Eventually, with enough action-action-preference feedback, the agent should be able to learn the preferable actions.

Recently, there has been a resurgence of interest in learning from pairwise comparisons due to its success in fine-tuning large language models (Christiano et al., 2017; Ziegler et al., 2020; Bai et al., 2022). The main idea behind relative feedback is to address the issues that make providing reward-based feedback challenging. For instance, how should you assign a numerical score to a sequence of text? Providing this form of feedback is difficult and time-consuming. Conversely, it is much easier to read two pieces of text and state a preference.

This chapter proposes an algorithm that sequentially learns to maximise the preferences of the environment from pairwise comparisons. Section 7.2 formally introduces the problem setting under consideration. Then, Section 7.3 summarises the related work. Subsequently, Section 7.4 introduces the proposed algorithm and provides guarantees on its worst-case performance. Finally, Section 7.5 presents a discussion and highlights possible improvements.

7.2 Problem Setting

Throughout, $\mathcal{A} = \{1, 2, \dots, k\}$ will denote the set of actions. Further, $P_\star \in [0, 1]^{k \times k}$ will denote the preference matrix, where each entry corresponds to the unknown probability of the environment preferring each action to another action. The only assumption imposed on the preference matrix is that it satisfies the following equality:

$$P_\star + P_\star^T = \mathbf{1}\mathbf{1}^T.$$

Concretely, $P_\star(a \succ a')$ will represent the probability that the environment prefers action a to action a' and is the entry in the a -th row and a' -th column of the matrix $P_\star$. The assumptions imposed on the preference matrix guarantees the following:

$$P_\star(a \succ a') = 1 - P_\star(a' \succ a) \tag{7.1}$$

for all $(a, a') \in \mathcal{A} \times \mathcal{A}$.

Just like the multi-armed bandit problem, the interaction between the agent and the environment occurs at discrete points in time. However, the agent cannot ask for direct feedback on the quality of each action. Instead, they can only ask for a preference between pairs of actions. Therefore, the agent must select two actions that they would like to compare in every round. Suppose $(A_t, A'_t) \in \mathcal{A} \times \mathcal{A}$ were chosen in the t -th round. Then, the environment is assumed to randomly generate their preference as follows:

$$X_t \stackrel{\text{i.i.d.}}{\sim} \text{Bernoulli}(P_*(A_t \succ A'_t)) .$$

Throughout, it will be assumed that the agent can use a pair of policies to select their actions. These policies can depend on past information and can use randomisation. Formally, this means that:

$$\begin{aligned} A_t &\sim \pi_t(\cdot) \text{ where } \pi_t(a) = \mathbb{P}(A_t = a \mid A_1, A'_1, X_1, \dots, A_{t-1}, A'_{t-1}, X_{t-1}) \\ A'_t &\sim \pi'_t(\cdot) \text{ where } \pi'_t(a) = \mathbb{P}'(A'_t = a \mid A_1, A'_1, X_1, \dots, A_{t-1}, A'_{t-1}, X_{t-1}) . \end{aligned}$$

Generally, π_t and π'_t do not have to be identical. Furthermore, this could be relaxed by letting the agent sample pairs of actions from a joint distribution or allowing A'_t to be chosen with knowledge of the value of A_t . This possibility will not be explored here.

Typically, real-world applications have access to two quantities, namely a reference policy and an existing dataset (Christiano et al., 2017; Ziegler et al., 2020; Bai et al., 2022). This chapter focuses on problems where an agent is given access to these two sources of information and can leverage them to guide their decision-making. First, π_{ref} will denote the reference policy:

$$\pi_{\text{ref}} : \mathcal{A} \rightarrow (0, 1]^k \text{ such that } 1 = \sum_{a \in \mathcal{A}} \pi_{\text{ref}}(a) .$$

The reference policy represents a known distribution over the actions that is either currently used in practice or has been demonstrated to have desirable properties, such as high user engagement. This is often the case in applications involving large-language models (OpenAI, 2019).

Second, $H_{n_{\text{off}}}$ will denote an offline dataset, which we will assume has been collected by following

the reference policy:

$$H_{n_{\text{off}}} = \{(A_t, A'_t, X_t) : A_t \sim \pi_{\text{ref}}(\cdot), A'_t \sim \pi_{\text{ref}}(\cdot), X_t \sim \text{Bernoulli}(P_\star(A_t \succ A'_t))\}_{t=1}^{n_{\text{off}}} \quad (7.2)$$

where $n_{\text{off}} \in \mathbb{N}$ denotes the number of pairwise comparisons in the existing dataset. This dataset is available before any interaction with the environment takes place and is motivated by what is done in practice (Ziegler et al., 2020).

Finally, we will constrain the agent to updating the pair of policies at the end of pre-determined rounds. The rounds where updating is permitted is often referred to as the *grid* in the batched bandit literature (Perchet et al., 2015; Gao et al., 2019; Ren et al., 2021; Sawarni et al., 2024):

$$\mathcal{T} = \{t_1, t_2, \dots, t_m\} \quad \text{with} \quad n_{\text{off}} + 1 = t_1 < t_2 < \dots < t_m = n. \quad (7.3)$$

Here, t_j denotes the round associated with the start of the j -th batch and t_m denotes the round associated with the final decision. Equation (7.3) does not preclude the fully-online setting, where $t_{j+1} = t_j + 1$. However, fully online decision-making is difficult to execute in real-world applications, especially when collecting feedback involves humans (Kittur et al., 2008). Procedure 7.1 summarises the sequential decision-making process under consideration.

Procedure 7.1: Batched Learning with Pairwise Comparisons

Input.

- Existing Dataset: $H_0 = H_{n_{\text{off}}}$ of the form given in Equation (7.2).
- The Grid: $\mathcal{T}$ of the form given in Equation (7.3).

For $j \in \{1, 2, \dots, m-1\}$:

- **Agent.** Computes π_{t_j} and π'_{t_j}
- **For** $t \in \{t_j, t_j + 1, \dots, t_{j+1} - 1\}$
 1. **Agent.** Selects $A_t \sim \pi_{t_j}(\cdot)$ and $A'_t \sim \pi'_{t_j}(\cdot)$
 2. **Environment.** Reveals $X_t = 1\{A_t \succ A'_t\} \sim \text{Bernoulli}(P_\star(A_t \succ A'_t))$
- **Agent.** Updates $H_{t_{j+1}-1} = H_{t_j-1} \cup \{A_t, A'_t, X_t\}_{t=t_j}^{t_{j+1}-1}$

To summarise, the offline dataset and batching scheme are given as input. Subsequently, each batch begins with the agent computing a pair of policies, which are then used to collect additional preference data. Finally, the preference dataset is updated at the end of the batch.

7.2.1 Measuring Performance

Procedure 7.1 is missing crucial details, such as: how should the agent compute the policies at the start of each batch? Naturally, this depends on the objective, which will be introduced in this subsection.

This chapter focuses on the *identity preference optimisation* objective (Azar et al., 2024). Informally, this objective asks the agent to find a policy that generates actions that are preferred to those generated by the reference policy, but is not too far away from the reference policy. Formalising this objective requires some definitions.

Let π and π' denote a pair of policies, which represent distributions over the set of actions. Then, the probability of preferring action a to policy π' is defined as follows:

$$P_{\star}(a \succ \pi') = \sum_{a' \in \mathcal{A}} P_{\star}(a \succ a') \pi'(a'). \quad (7.4)$$

Next, the probability of preferring policy π to policy π' has the following definition:

$$P_{\star}(\pi \succ \pi') = \sum_{a \in \mathcal{A}} P_{\star}(a \succ \pi') \pi(a) = \sum_{a \in \mathcal{A}} \sum_{a' \in \mathcal{A}} P_{\star}(a \succ a') \pi(a) \pi'(a').$$

Azar et al. (2024) introduced the idea of identity preference optimisation. Formally, the quality of policy π is measured as the regularised probability of beating the reference policy π_{ref} . Mathematically, this is defined as follows (Azar et al., 2024):

$$Q_{\star}(\pi, \pi_{\text{ref}}) = P_{\star}(\pi \succ \pi_{\text{ref}}) - \eta D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) \quad (7.5)$$

Here, $D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) := \sum_{a \in \mathcal{A}} \pi(a) \ln(\pi(a)/\pi_{\text{ref}}(a))$ is the usual definition of the relative entropy between distributions. Further, $\eta > 0$ denotes the strength of the regularisation and penalises policies that are too far away from the reference policy, as measured by the relative entropy.

Naturally, the optimal policy is defined as the maximiser of Equation (7.5):

$$\pi_{\star} = \arg \max_{\pi \in \Pi} Q_{\star}(\pi, \pi_{\text{ref}})$$

where Π is the set of all probability distributions over the actions. [Azar et al. \(2024\)](#) show that the optimal policy for the preference matrix $P_{\star}$ and reference policy π_{ref} can be computed in closed-form:

$$\pi_{\star}(a) = \frac{\pi_{\text{ref}}(a) \exp\left(\frac{P_{\star}(a \succ \pi_{\text{ref}})}{\eta}\right)}{Z_{\star}(\pi_{\text{ref}})} \quad (7.6)$$

where

$$Z_{\star}(\pi_{\text{ref}}) = \sum_{a=1}^k \pi_{\text{ref}}(a) \exp\left(\frac{1}{\eta} P_{\star}(a \succ \pi_{\text{ref}})\right)$$

is referred to as the *partition function*. Throughout this chapter, we will define the regret as the cumulative sub-optimality of the sequence of policies computed by the agent:

$$\hat{R}_n(\pi, P_{\star}) = \sum_{t=n_{\text{off}}+1}^n (Q_{\star}(\pi_{\star}, \pi_{\text{ref}}) - \max\{Q_{\star}(\pi_t, \pi_{\text{ref}}), Q_{\star}(\pi'_t, \pi_{\text{ref}})\}) \quad (7.7)$$

where $\pi = \{\pi_t, \pi'_t\}_{t=n_{\text{off}}+1}^n$ denotes the sequence policies that are used to select the actions in each round of the interaction. Equation (7.7) measures the difference between the regularised preferences of the optimal policy and the better of policies π_t and π'_t . Naturally, we want to minimise the regret. This objective is natural when $P_{\star}$ represents the unknown preference matrix for a human or population of humans. For instance, in recommender systems, user satisfaction depends only on the best presented option, rather than the individual quality of all presented choices. What ultimately matters is whether at least one of the policies produces actions that are close to those generated by the optimal policy.

7.3 Related Work

The shift from absolute to relative feedback presents practical advantages and interesting theoretical challenges. This section presents an overview of the existing literature on the topic of learning from pairwise comparisons, with a particular emphasis placed on methods that come with performance guarantees.

7.3.1 Duelling Bandits

[Yue et al. \(2009\)](#) were the first to consider learning from pairwise comparisons with bandit feedback, which is commonly referred to as *duelling bandits*. The literature on this topic is vast. Therefore, only a brief summary will be provided here. See [Bengs et al. \(2021\)](#) for an extensive survey of the literature on duelling bandits.

Briefly, the problem setting is very similar to Procedure 7.1. However, there are some notable differences. First, $H_{\text{off}} = \emptyset$ and $\mathcal{T} = \{1, 2, \dots, n\}$ so that the agent can change their distribution over the actions every round, as was the case in the standard multi-armed bandit problem from Procedure 2.1 of Chapter 2. Second, there is no reference policy. There are numerous variants of the duelling bandit problem. These variants depend on the assumptions that are made on the preference matrix $P_\star$.

Typically, it is assumed that there exists an action that is preferred to all other actions:

$$\exists a \in \mathcal{A} \text{ such that } P_\star(a \succ a') > \frac{1}{2} \text{ for all } a' \neq a.$$

Here, modifications of the algorithmic principles for the multi-armed bandit have been proposed. For example, authors have built on the upper confidence bound algorithm ([Zoghi et al., 2014, 2015b](#)), Thompson Sampling ([Wu and Liu, 2016](#); [Li et al., 2020](#)) and successive elimination ([Yue et al., 2009](#); [Yue and Joachims, 2011](#)), just to name a few. [Jamieson et al. \(2015\)](#) highlight that an action that dominates all other actions need not exist. This motivated the introduction of various other problem settings and algorithms ([Jamieson et al., 2015](#); [Zoghi et al., 2015a](#); [Dudik et al., 2011](#)).

7.3.2 Reward-Based Preferences

Recently, there has been a renewed interest in learning from pairwise comparisons. Similarly to the duelling bandit problem, it is common to place additional assumptions on preference matrix $P_\star$. The most common is to assume the existence of an underlying reward function ([Zhu et al., 2023](#); [Xiong et al., 2024](#); [Zhan et al., 2024a](#); [Xie et al., 2025](#)):

$$P_\star(a \succ a') = \sigma(r_\star(a) - r_\star(a')) \tag{7.8}$$

where $r_\star : \mathcal{A} \rightarrow \mathbb{R}$ can be interpreted as an unknown latent expected reward function and $\sigma : \mathbb{R} \rightarrow [0, 1]$ is a monotonically increasing link function that is known in advance. The reward function is latent in the sense that rewards cannot be observed directly. Equation (7.8) is known as the Bradley-Terry assumption (Zermelo, 1929; Bradley and Terry, 1952).

The existing literature partitions into methods developed for the offline and online settings. Chapters 4-6 all focus on problems in the online setting, where the agent must carefully balance exploration and exploitation. However, we have not discussed the offline setting. Briefly, the offline learning setting assumes access to a static dataset of pairwise comparisons and does not permit the collection of any more information. Therefore, the methods developed for this setting focus on learning near-optimal policies using only $H_{n_{\text{off}}}$. Typically, algorithms developed for the offline reward-based setting estimate the latent reward function via maximum likelihood and implement the pessimistic principle (Zhu et al., 2023; Xiong et al., 2024; Zhan et al., 2024a). The pessimistic principle is similar to, but different from, the optimistic principle introduced in Chapter 2. For learning from pairwise comparisons, the pessimistic and optimistic principles would both adjust an estimate of the latent reward for each action based on uncertainty. However, they do so in opposite directions: pessimism subtracts the uncertainty, whereas optimism adds the uncertainty. This may seem symmetric, but there is a subtle difference that makes optimism unappealing for the setting of offline learning. Section 2.2 of Chapter 2 explains that optimism encourages exploration of the lesser-chosen actions by over-estimating the rewards. However, the setting of offline learning does not permit interaction with the environment, meaning that optimism can lead to learning policies that assign a high probability to under-explored actions that only appear promising due to a lack of information. Conversely, pessimism penalises the estimated expected reward of each action based on whether there is sufficient evidence to trust their estimated value. This encourages the learning policies that place a high probability on actions that have a high latent reward and are well-represented in the offline dataset.

Offline Learning. Currently, the existing algorithms developed for the offline reward-based setting estimate the latent reward function via maximum likelihood and implement the pessimistic principle (Zhu et al., 2023; Xiong et al., 2024; Zhan et al., 2024a). Here, the offline dataset is used to compute the maximum likelihood estimator and to construct the confidence sets. Zhu et al. (2023) and Xiong et al. (2024) both assume that the latent reward function is

linear:

$$r_{\star}(a) = \langle a, \theta_{\star} \rangle,$$

where $\theta_{\star} \in \mathbb{R}^d$ is an unknown parameter vector. This assumption, along with assuming that the offline dataset was collected as per Equation (7.2), leads them to devise pessimistic estimators which are then used to select a policy. [Zhu et al. \(2023\)](#) focus on using the offline dataset to learn the policy that maximises:

$$J(\pi) = \sum_{a \in \mathcal{A}} r_{\star}(a) \pi(a)$$

which is the expected latent reward. The authors propose selecting the policy that maximises a pessimistic estimate of the latent reward function:

$$\hat{\pi} = \arg \max_{\pi \in \Pi} \min_{\theta \in \Theta_{n_{\text{off}}}} \sum_{a \in \mathcal{A}} \langle a, \theta \rangle \pi(a)$$

where Π denotes a class of policies and $\Theta_{n_{\text{off}}}$ denotes a set of statistically feasible parameter vectors that was constructed using an offline dataset containing n_{off} pairwise comparisons. [Xiong et al. \(2024\)](#) focus on finding the policy that maximises the regularised expected reward: $J(\pi) - \eta D_{\text{KL}}(\pi \parallel \pi_{\text{ref}})$, where $\eta > 0$ is a hyperparameter controlling the amount of regularisation. Both [Zhu et al. \(2023\)](#) and [Xiong et al. \(2024\)](#) utilise existing concentration results presented in [Hsu et al. \(2012\)](#) to construct their confidence sets. [Zhan et al. \(2024a\)](#) also consider applying the pessimistic principle in the regularised setting and extend the theoretical results to handle scenarios where $r_{\star}$ is potentially non-linear.

Online Learning. The other problem setting that has been considered in the literature allows the algorithm to interact with the environment. This is the online learning setting. [Xiong et al. \(2024\)](#) develop an algorithm for minimising the regret against the maximiser of: $J(\pi) - \eta D_{\text{KL}}(\pi \parallel \pi_{\text{ref}})$. The algorithm uses an asymmetric algorithmic design where one action is drawn from a policy that is greedy with respect to the maximum likelihood estimator. The other action is drawn from a distribution that carefully balances exploration and exploitation on the set of promising policies. [Xie et al. \(2025\)](#) consider the pure exploration setting, where there is an existing reference policy and the objective is to use online interactions to produce a single policy at the end that is nearly optimal. This setting does not consider the sub-optimality of any policies that were produced within the interaction. [Xie et al. \(2025\)](#) directly optimise the parameters of the policy using the log-likelihood; an idea introduced by [Rafailov et al. \(2023\)](#). However,

they introduce an optimistic bonus into the log-likelihood objective to encourage exploration. The policy that is outputted by their procedure is the one that maximises performance on a validation dataset, and this is the policy that comes with theoretical guarantees.

Limitations. The widely used Bradley-Terry model, as expressed in Equation (7.8), rests on a strong assumption: the existence of a latent reward function. This modelling assumption implies increasing the latent reward will increase the probability of preference. Consequently, the reward function can be used to order the actions in terms of their preferences. Furthermore, this implies that it is limited to learning transitive preferences:

$$P_{\star}(a \succ a') \geq \frac{1}{2} \text{ and } P_{\star}(a' \succ a'') \geq \frac{1}{2} \implies P_{\star}(a \succ a'') \geq \frac{1}{2}.$$

There are many real-world problems where humans have been shown to violate this modelling assumption, such as law, biology and economics (Tversky, 1969; Klimenko, 2015). This is particularly concerning because the main attraction of pairwise comparisons is the promise of replacing the task of designing, or eliciting, reward-based feedback with stating a preference. Yet, the most-studied model is restricted to learning models that may not capture the nuances of the humans that are providing the preference-based feedback.

7.3.3 Reward-Free Preferences

The other category of literature does not impose any assumptions on the preference matrix $P_{\star}$. This approach was first tackled in the duelling bandit literature, where several alternative notions of optimality were proposed (Dudik et al., 2011; Jamieson et al., 2015; Zoghi et al., 2015a). Among these, a variant of the *von Neumann Winner* has been studied in the regularised setting:

$$\pi_{\star} = \arg \max_{\pi} \min_{\pi'} \sum_{a, a'} P_{\star}(a \succ a') \pi(a) \pi(a') - \eta (\text{D}_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) - \text{D}_{\text{KL}}(\pi' \parallel \pi_{\text{ref}})). \quad (7.9)$$

Equation (7.9) defines the optimal policy as the distribution over actions that is at least as preferable to any other distribution over actions.

Offline Learning. Ye et al. (2024) consider the reward-free setting with the objective of learning the policy defined in Equation (7.9). The proposed method uses maximum likelihood estimation to estimate the unknown preference probabilities and implements pessimism by constructing

confidence sets around the estimator. Their approach selects the following policy:

$$\hat{\pi} = \arg \max_{\pi} \left\{ \min_{\pi'} \left\{ \min_{P \in \mathcal{P}_{n_{\text{off}}}} \left\{ \sum_{a, a'} P(a \succ a') \pi(a) \pi(a') - \eta D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) + \eta D_{\text{KL}}(\pi' \parallel \pi_{\text{ref}}) \right\} \right\} \right\}$$

where $\mathcal{P}_{n_{\text{off}}}$ denotes the set of statistically feasible preference matrices that is constructed with the offline dataset.

Online Learning. Ye et al. (2024) also study the online learning setting. Specifically, the authors consider a batched deployment strategy, where each batch consists of a fixed number of samples. Their theoretical analysis aims to establish sample complexity bounds, determining the total number of samples required to obtain an ϵ -optimal policy. Their approach is similar to the approach taken in the reward-based setting (Xiong et al., 2024). Specifically, Ye et al. (2024) propose an asymmetric algorithm that computes a pair of policies at the start of each batch. The first policy is greedy with respect to the maximum likelihood estimator, which requires solving the same min-max problem as in the offline setting. The second policy is designed to generate actions that maximally reduce the uncertainty.

Limitations. Even if $P_{\star}$ were known, exact computation of the optimal policy in Equation (7.9) is generally intractable. Munos et al. (2024) proposed a mirror-descent style algorithm that approximately solves the minimax problem for a given preference matrix. This has last-iterate convergence, which is especially useful in the setting of large-language models. Nevertheless, the methods analysed by Ye et al. (2024) are still challenging to implement due to the inner minimisation over the set of statistically plausible preference matrices. Consequently, approximations are required, and the theoretical guarantees they provide need not hold.

7.4 An Algorithm for Identity Preference Optimisation

Here, we seek to design an algorithm for the problem setting introduced in Section 7.2. Our main aim is to develop an algorithm that can be implemented efficiently and keeps the regret of Equation (7.7) as small as possible. This means that, in each round, we only need one of the policies to converge to the optimal policy of Equation (7.6). This is because the regret depends only on the policy that generates the highest regularised probability of beating π_{ref} .

Equation (7.6) is particularly instructive and it provides a closed-form expression for the optimal

policy. This closed-form expression can be used to derive the optimal policy that corresponds to an arbitrary preference matrix. Therefore, a natural strategy is to use all of the existing information to estimate $P_\star$. Generally, as long as this estimator converges to the unknown preference probabilities sufficiently fast, it should be possible to obtain sub-linear guarantees on the regret. Previous works, which consider different problem settings, have relied on the maximum likelihood estimator for estimating preferences (Zhu et al., 2023; Ye et al., 2024; Xiong et al., 2024; Zhan et al., 2024a,b; Xie et al., 2025). However, importance-weighted estimators offer an alternative that has been under-explored in the context of learning from pairwise comparisons. The idea of importance weighting goes back to at least Horvitz and Thompson (1952). More recently, it has been widely used in offline contextual bandits (Dudík et al., 2011; Swaminathan and Joachims, 2015; Wang et al., 2024) and adversarial bandits (Auer et al., 1995; Stoltz, 2005; Auer et al., 2002b).

Traditionally, importance-weighted estimators are used to obtain an unbiased estimate of the relevant quantity by re-weighting the dataset based on the probability of selecting each action. However, for pairwise comparisons, each round involves selecting two actions. Therefore, the standard importance-weighted estimator would have to re-weight the dataset using the probabilities of selecting both actions in each round. This strategy would increase the variance of the estimator. Fortunately, Equations (7.4) and (7.6) suggest that the quantity that needs to be estimated is the probability of preferring each action to the reference policy. Indeed, we can get samples from this distribution by sampling one of the actions from the reference policy. Therefore, we propose setting:

$$\pi'_t(\cdot) = \pi_{\text{ref}}(\cdot)$$

for all $t > n_{\text{off}}$. This algorithmic choice reduces the number of quantities that need estimating from k^2 to k . Consequently, we propose the following re-weighting of the information collected in the t -th round:

$$\hat{X}_t(a) = \frac{1\{A_t = a\} X_t}{\pi_t(a)} = \frac{1\{A_t = a\} 1\{A_t \succ A'_t\}}{\pi_t(a)} \quad (7.10)$$

where π_t denotes the policy that was used to select A_t and $A'_t \sim \pi'_t(\cdot) = \pi_{\text{ref}}$. In Section 7.4.1, we will show that using $\pi'_t(\cdot) = \pi_{\text{ref}}(\cdot)$ guarantees that the expected value of $\hat{X}_t(a)$ is $P_\star(a \succ \pi_{\text{ref}})$. Then, $P_\star(a \succ \pi_{\text{ref}})$ will be estimated using all of the available information at the start of the

j -th batch:

$$\hat{P}_j(a \succ \pi_{\text{ref}}) = \frac{1}{t_j - 1} \sum_{t=1}^{t_j-1} \hat{X}_t(a). \quad (7.11)$$

Algorithm 10 provides the pseudo-code for our algorithm that uses Equation (7.11) for the task of identity preference optimisation.

Algorithm 10 Importance Sampling for Identity Preference Optimisation

Input. $H_{n_{\text{off}}} = \{A_t, A'_t, X_t\}_{t=1}^{n_{\text{off}}}$ the offline dataset, $\mathcal{T} = \{1 + n_{\text{off}} = t_1 < t_2 < \dots < t_m = n\}$ the batching scheme, π_{ref} the reference policy and $\eta > 0$.

for $j = 1, 2, \dots, m$ **do**

Compute $\hat{P}_j(a \succ \pi_{\text{ref}})$ via Equation (7.11)

Compute $\pi_{t_j} \propto \pi_{\text{ref}}(a) \exp\left(\frac{\hat{P}_j(a \succ \pi_{\text{ref}})}{\eta}\right)$

for $t = t_j, t_j + 1, \dots, t_{j+1} - 1$ **do**

Set $\pi_t(\cdot) = \pi_{t_j}(\cdot)$.

Select $A_t \sim \pi_t(\cdot)$ and $A'_t \sim \pi_{\text{ref}}(\cdot)$

Receive $X_t = 1\{A_t \succ A'_t\}$ and update $H_t = H_{t-1} \cup \{A_t, A'_t, X_t\}$

end for

end for

Briefly, Algorithm 10 selects π_{t_j} greedily by plugging in the estimated probability of preferring each action to the reference policy into the form of the optimal policy. This policy is stochastic, and this guarantees exploration. Further, always playing the reference policy ensures that the preference dataset collected in each batch can be used to accurately estimate $P_\star(a \succ \pi_{\text{ref}})$. Theorem 7.1 provides an upper bound on the regret for Algorithm 10.

Theorem 7.1. *Let $H_{n_{\text{off}}}$ be the offline dataset that has been provided as input and collected as per Equation (7.2). Then, Algorithm 10 with an arbitrary predetermined grid $\mathcal{T} = \{t_1, t_2, \dots, t_m\}$ satisfies the following with probability at least $1 - 2\delta$:*

$$\hat{R}_n(\pi, P_\star) \leq \frac{\sqrt{2 \ln\left(\frac{k m (m+1)}{\delta}\right)}}{\alpha} \sum_{j=1}^{m-1} \frac{(t_{j+1} - t_j)}{\sqrt{t_j - 1}} - \eta \sum_{j=1}^{m-1} (t_{j+1} - t_j) D_{KL}(\pi_\star \| \pi_{t_j})$$

where

$$\alpha := \min_{(a \times t) \in \mathcal{A} \times \mathcal{T}} \pi_t(a) > 0$$

denotes a positive lower bound on the probability of selecting an action.

Proof. See Section 7.4.1. □

Theorem 7.1 provides a high probability upper bound of the regret of Algorithm 10. This guarantee is stated for an arbitrary predetermined grid, and it can be used to compute an upper bound on the regret for various special cases, such as the *arithmetic* and *geometric* grids (Perchet

et al., 2015). These grids are defined as follows:

$$\begin{aligned}\mathcal{T}_{\text{arithmetic}}^B &= \{t_1 = n_{\text{off}} + 1, t_2 = t_1 + B, \dots, t_m = t_{m-1} + B = n\} \\ \mathcal{T}_{\text{geometric}}^B &= \{t_1 = n_{\text{off}} + B^0, t_2 = t_1 + B^1, \dots, t_m = t_{m-1} + B^{m-1} = n\}\end{aligned}$$

where B is a parameter that controls the number of samples within each batch. Corollaries 7.1 and 7.2 present the upper bound on the cumulative regret for $\mathcal{T}_{\text{arithmetic}}^B$ and $\mathcal{T}_{\text{geometric}}^B$, respectively.

Corollary 7.1. *Let $H_{n_{\text{off}}}$ be the offline dataset that has been provided as input and collected as-per Equation (7.2). Then, Algorithm 10 with $\mathcal{T} = \mathcal{T}_{\text{arithmetic}}^B$ satisfies the following:*

$$\hat{R}_n(\pi, \nu) \leq \frac{\sqrt{2 \ln \left(\frac{k m (m+1)}{\delta} \right)}}{\alpha} \left(\frac{B}{\sqrt{n_{\text{off}}}} + 2B\sqrt{n - n_{\text{off}} - (1 + B)} \right) - \eta B \sum_{j=1}^{m-1} D_{KL}(\pi_{\star} \parallel \pi_{t_j})$$

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

Proof. Since $\mathcal{T}_{\text{arithmetic}}^B$, it follows that the number of samples collected within every batch is identical:

$$B = t_{j+1} - t_j$$

for all $j \in \{1, 2, \dots, m-1\}$. Since the number of rounds within each batch is constant, it is straightforward to calculate the total number of samples used to compute the policy in each batch:

$$\begin{aligned}t_j &= B + t_{j-1} = 2B + t_{j-2} = \dots = t_1 + (j-1)B \\ &= 1 + n_{\text{off}} + (j-1)B\end{aligned}\tag{7.12}$$

From Theorem 7.1, it follows that the remaining step is to control the summation:

$$S_m = \sum_{j=1}^{m-1} \frac{(t_{j+1} - t_j)}{\sqrt{t_j - 1}} = B \sum_{j=1}^{m-1} \frac{1}{\sqrt{t_j - 1}}$$

This summation is straightforward to control and can be upper bounded by an integral:

$$S_m = \frac{B}{\sqrt{n_{\text{off}}}} + B \sum_{j=2}^{m-1} \frac{1}{\sqrt{t_j - 1}}$$

$$\begin{aligned}
&= \frac{B}{\sqrt{n_{\text{off}}}} + B \sum_{j=2}^{m-1} \frac{1}{\sqrt{n_{\text{off}} + (j-1)B}} \\
&\leq \frac{B}{\sqrt{n_{\text{off}}}} + B \int_0^{m-2} \frac{dx}{\sqrt{n_{\text{off}} + Bx}} \\
&= \frac{B}{\sqrt{n_{\text{off}}}} + 2B\sqrt{B(m-2) + n_{\text{off}}} - 2B\sqrt{n_{\text{off}}} \\
&\leq \frac{B}{\sqrt{n_{\text{off}}}} + 2B\sqrt{B(m-2)} \quad (\sqrt{a-b} \geq \sqrt{a} - \sqrt{b} \text{ for } a \geq b > 0) \\
&= \frac{B}{\sqrt{n_{\text{off}}}} + 2B\sqrt{t_m - (1 + n_{\text{off}} + B)} \quad (\text{By Equation (7.12)}) \\
&= \frac{B}{\sqrt{n_{\text{off}}}} + 2B\sqrt{n - n_{\text{off}} - (1 + B)} \quad (\text{Since } t_m = n.)
\end{aligned}$$

Plugging this upper bound on the summation into Theorem 7.1 gives the desired result. $\square$

Corollary 7.2. *Let $H_{n_{\text{off}}}$ be the offline dataset that has been provided as input and collected as-per Equation (7.2). Then, Algorithm 10 with $\mathcal{T} = \mathcal{T}_{\text{geometric}}^B$ and $B \geq 2$ satisfies the following:*

$$\hat{R}_n(\pi, \nu) \leq \frac{\sqrt{2 \ln \left(\frac{k m (m+1)}{\delta} \right)}}{\alpha} \left(\frac{B}{\sqrt{n_{\text{off}}}} + \frac{2B-1}{\ln(B)} \sqrt{n - n_{\text{off}} - 1} \right) - \eta \sum_{j=1}^{m-1} B^j D_{KL}(\pi_{\star} \parallel \pi_{t_j})$$

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

Proof. Since $\mathcal{T}_{\text{geometric}}^B$, it follows that the number of samples collected within every batch grows according to a geometric progression:

$$t_{j+1} - t_j = B^j.$$

Furthermore, it is straightforward to calculate the total number of samples used to compute the policy in each batch:

$$\begin{aligned}
t_j &= t_{j-1} + B^{j-1} = t_{j-2} + B^{j-2} + B^{j-1} = \dots = t_1 + \sum_{i=1}^{j-1} B^i \\
&= 1 + n_{\text{off}} + \sum_{i=1}^{j-1} B^i \\
&= 1 + n_{\text{off}} + \frac{B^j - B}{B - 1}
\end{aligned} \tag{7.13}$$

where the last equality follows from the summation being a geometric series. Notably, $B \geq 2$ in the statement of the corollary. Therefore, Equation (7.13) is not infinite. From Theorem 7.1, it

follows that the remaining step is to control the summation:

$$S_m = \sum_{j=1}^{m-1} \frac{(t_{j+1} - t_j)}{\sqrt{t_j - 1}} = \sum_{j=1}^{m-1} \frac{B^j}{\sqrt{n_{\text{off}} + \frac{B^j - B}{B-1}}}$$

where the equality follows from plugging in $t_{j+1} - t_j = B^j$ and Equation (7.13). This summation is straightforward to control and can be upper bounded by an integral:

$$\begin{aligned} S_m &= \frac{B}{\sqrt{n_{\text{off}}}} + \sum_{j=2}^{m-1} \frac{B^j}{\sqrt{n_{\text{off}} + \frac{B^j - B}{B-1}}} \\ &\leq \frac{B}{\sqrt{n_{\text{off}}}} + \int_1^{m-1} \frac{B^x}{\sqrt{n_{\text{off}} + \frac{B^x - B}{B-1}}} dx \\ &= \frac{B}{\sqrt{n_{\text{off}}}} + \frac{2B-1}{\ln(B)} \left[\sqrt{\frac{B^{m-1} - B}{B-1}} + n_{\text{off}} - \sqrt{n_{\text{off}}} \right] \\ &\leq \frac{B}{\sqrt{n_{\text{off}}}} + \frac{2B-1}{\ln(B)} \sqrt{\frac{B^{m-1} - B}{B-1}} \quad (\sqrt{a-b} \geq \sqrt{a} - \sqrt{b} \text{ for } a \geq b > 0) \\ &\leq \frac{B}{\sqrt{n_{\text{off}}}} + \frac{2B-1}{\ln(B)} \sqrt{\frac{B^m - B}{B-1}} \\ &= \frac{B}{\sqrt{n_{\text{off}}}} + \frac{2B-1}{\ln(B)} \sqrt{t_m - (1 + n_{\text{off}})} \quad (\text{By Equation (7.13)}) \\ &= \frac{B}{\sqrt{n_{\text{off}}}} + \frac{2B-1}{\ln(B)} \sqrt{n - n_{\text{off}} - 1} \quad (\text{Since } t_m = n) \end{aligned}$$

Plugging this upper bound on the summation into Theorem 7.1 gives the desired result. $\square$

Corollaries 7.1 and 7.2 display how changing the grid impacts the performance. For $\mathcal{T}_{\text{arithmetic}}^B$ and $\mathcal{T}_{\text{geometric}}^B$, the regret decreases as the offline dataset increases in size. Additionally, the guarantees suggest that increasing the regularisation strength makes the problem easier. Intuitively, this is because the optimal policy is defined in terms of the reference policy. Equation (7.6) shows that as $\eta \rightarrow \infty$, $\pi_\star \rightarrow \pi_{\text{ref}}$. Therefore, increasing the regularisation strength has the effect of restricting the exploration to a smaller neighbour of policies around the reference policy.

Comparing $\mathcal{T}_{\text{arithmetic}}^B$ and $\mathcal{T}_{\text{geometric}}^B$ directly is challenging due to the relative entropy term. Indeed, more work is required to establish the rate at which the sequence of policies converges to the optimal policy. This would be informative in selecting the batching scheme. Nevertheless, the geometric grid does have a better leading-order term, which is especially pronounced when considering larger values of B .

7.4.1 Preliminary Results for Theorem 7.1

This subsection will prove some useful results related to Algorithm 10. First, we will verify that the estimator in Equation (7.10) is unbiased. Subsequently, Lemmas 7.2 and 7.3 verify that the estimated probability of preferring each action to the reference policy converges sufficiently fast. Finally, we will establish a fact about policies of the form given in Equation (7.6). These results will all be used within the proof of Theorem 7.1.

Lemma 7.1. *Equation (7.10) is an unbiased estimate of the unknown preference matrix:*

$$\mathbb{E} \left[\hat{P}_j(a \succ \pi_{\text{ref}}) \right] = P_{\star}(a \succ \pi_{\text{ref}})$$

for all $(a, j) \in \mathcal{A} \times \{1, 2, \dots, m-1\}$.

Proof. From Algorithm 10, $\pi_{t_j} = \pi_{t_j+1} = \dots = \pi_{t_{j+1}-1}$ and π_{t_j} is computed using the entire historical dataset:

$$H_{t_j-1} = \{A_t, A'_t, X_t\}_{t=1}^{t_j-1}.$$

Consider $t \in [t_j, t_{j+1})$ and note that the preference indicator, X_t , depends only on the pair of actions that were chosen by the agent. Further, A_t and A'_t are drawn from π_{t_j} and π_{ref} , respectively. Therefore, the probability of observing a particular action-action-preference triple, given the history, can be written as follows:

$$\mathbb{P}(A_t = a, A'_t = a', X_t = x \mid H_{t_j-1}) = \mathbb{P}(X_t = x \mid A_t = a, A'_t = a') \pi_{t_j}(a) \pi_{\text{ref}}(a')$$

which follows from the fact that the preference indicator depends only on the pair of actions that were chosen in the t -th round. Consequently, taking the expectation of $\hat{X}_t(a)$ gives:

$$\begin{aligned} \mathbb{E} \left[\hat{X}_t(a) \mid H_{t_j-1} \right] &= \mathbb{E} \left[\frac{1\{A_t = a\} X_t}{\pi_t(a)} \mid H_{t_j-1} \right] \\ &= \mathbb{E} \left[\frac{1\{A_t = a\} X_t}{\pi_{t_j}(a)} \mid H_{t_j-1} \right] && (\pi_t = \pi_{t_j} \text{ for } t \in [t_j, t_{j+1})) \\ &= \frac{1}{\pi_{t_j}(a)} \mathbb{E} [1\{A_t = a\} X_t \mid H_{t_j-1}] \\ &= \frac{1}{\pi_{t_j}(a)} \sum_{a_1} \sum_{a_2} \sum_{x \in \{0,1\}} x 1\{a_1 = a\} \mathbb{P}(A_t = a_1, A'_t = a_2, X_t = x \mid H_{t_j-1}) \\ &= \frac{1}{\pi_{t_j}(a)} \sum_{a_2} \mathbb{P}(A_t = a, A'_t = a_2, X_t = 1 \mid H_{t_j-1}) \\ &= \frac{1}{\pi_{t_j}(a)} \sum_{a_2} P_{\star}(a \succ a_2) \pi_{t_j}(a) \pi_{\text{ref}}(a_2) \end{aligned}$$

$$= P_{\star}(a \succ \pi_{\text{ref}})$$

where the last equality follows from the definition given in Equation (7.4). Therefore, the law of total expectation immediately reveals that Equation (7.11) inherits this unbiased property:

$$\begin{aligned} \mathbb{E} \left[\hat{P}_j(a \succ \pi_{\text{ref}}) \right] &= \frac{1}{t_j - 1} \sum_{i=1}^{j-1} \sum_{t=t_i}^{t_{i+1}-1} \mathbb{E} \left[\hat{X}_t(a) \right] \\ &= \frac{1}{t_j - 1} \sum_{i=1}^{j-1} \sum_{t=t_i}^{t_{i+1}-1} \mathbb{E} \left[\mathbb{E} \left[\hat{X}_t(a) \mid H_{t_i-1} \right] \right] \\ &= \frac{1}{t_j - 1} \sum_{i=1}^{j-1} \sum_{t=t_i}^{t_{i+1}-1} P_{\star}(a \succ \pi_{\text{ref}}) \\ &= P_{\star}(a \succ \pi_{\text{ref}}) \end{aligned}$$

which completes the proof. $\square$

Next, we will use the fact that the estimator is unbiased to show that it converges sufficiently fast. Notably, $\hat{X}_t(a)$ of Equation (7.10) is itself a random variable and it is non-negative. Furthermore, we make use of the following symbol:

$$\alpha := \min_{(t,a) \in \mathcal{T} \times \mathcal{A}} \pi_t(a) > 0 \quad (7.14)$$

which defines α as a positive lower bound on the probability of selecting an action. This does not need to be known for implementing the algorithm and it is only utilised for the theoretical analysis. From Equation (7.14) it follows that:

$$0 \leq \hat{X}_t(a) \leq \frac{1}{\alpha}$$

almost surely. These almost sure upper and lower bounds can be used to derive high probability guarantees on how far $\hat{P}_j(a \succ \pi_{\text{ref}})$ deviates from $P_{\star}(a \succ \pi_{\text{ref}})$.

Lemma 7.2. *Fix an action-batch pair $(a, j) \in \mathcal{A} \times \{1, 2, \dots, m\}$. Let $H_{n_{\text{off}}}$ be the offline dataset collected as-per Equation (7.2). Then, it holds that:*

$$\begin{aligned} \mathbb{P} \left(\hat{P}_j(a \succ \pi_{\text{ref}}) - P_{\star}(a \succ \pi_{\text{ref}}) \geq \epsilon \right) &\leq \exp(-2(t_j - 1)\alpha^2\epsilon^2) \\ \mathbb{P} \left(P_{\star}(a \succ \pi_{\text{ref}}) - \hat{P}_j(a \succ \pi_{\text{ref}}) \geq \epsilon \right) &\leq \exp(-2(t_j - 1)\alpha^2\epsilon^2) \end{aligned}$$

for $\epsilon > 0$ and α defined as per Equation (7.14).

Proof. Applying the Cramér-Chernoff method, as in Hoeffding's inequality, gives the following (Hoeffding, 1963):

$$\begin{aligned} \mathbb{P} \left(\hat{P}_j(a \succ \pi_{\text{ref}}) - P_\star(a \succ \pi_{\text{ref}}) \geq \epsilon \right) &= \mathbb{P} \left(\sum_{t=1}^{t_j-1} \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) \geq (t_j - 1) \epsilon \right) \\ &= \mathbb{P} \left(\exp \left(\lambda \sum_{t=1}^{t_j-1} \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) \right) \geq \exp(\lambda(t_j - 1) \epsilon) \right) \\ &\leq \exp(-\lambda(t_j - 1) \epsilon) \mathbb{E} \left[\exp \left(\lambda \sum_{t=1}^{t_j-1} \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) \right) \right] \end{aligned}$$

where λ is a free parameter that will be tuned later in the analysis. Next, define the following:

$$M_{t_j} = \exp \left(\lambda \sum_{t=1}^{t_j-1} \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) \right)$$

which is the quantity inside the expectation of the previous display. From Algorithm 10 and Procedure 7.1, it follows that samples within the same batch are independent, given the information that was available at the end of the previous batch. Therefore, to start controlling this term, we will partition the summation into the batches and apply the law of total expectation:

$$\begin{aligned} \mathbb{E} [M_{t_j}] &= \mathbb{E} \left[\exp \left(\lambda \sum_{t=1}^{t_j-1} \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) + \lambda \sum_{t=t_{j-1}}^{t_j-1} \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) \right) \right] \\ &= \mathbb{E} \left[M_{t_{j-1}} \exp \left(\lambda \sum_{t=t_{j-1}}^{t_j-1} \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) \right) \right] \\ &= \mathbb{E} \left[\mathbb{E} \left[M_{t_{j-1}} \exp \left(\lambda \sum_{t=t_{j-1}}^{t_j-1} \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) \right) \mid H_{t_{j-1}-1} \right] \right] \quad (\text{Law Tot. Expectation}) \\ &= \mathbb{E} \left[M_{t_{j-1}} \mathbb{E} \left[\exp \left(\lambda \sum_{t=t_{j-1}}^{t_j-1} \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) \right) \mid H_{t_{j-1}-1} \right] \right] \quad (\star) \\ &= \mathbb{E} \left[M_{t_{j-1}} \prod_{t=t_{j-1}}^{t_j-1} \mathbb{E} \left[\exp \left(\lambda \left(\hat{X}_t(a) - P_\star(a \succ \pi_{\text{ref}}) \right) \right) \mid H_{t_{j-1}-1} \right] \right] \quad (\text{Indep. within Batch}) \\ &\leq \mathbb{E} [M_{t_{j-1}}] \exp \left(\frac{\lambda^2(t_j - t_{j-1})}{8\alpha^2} \right) \quad (\text{Hoeffding's Lemma}) \end{aligned}$$

where $(\star)$ uses the fact that $M_{t_{j-1}}$ is $H_{t_{j-1}-1}$ -measurable and the final line utilises the unbiased property of $\hat{X}_t(a)$ in conjunction with Equation (7.14) to apply Hoeffding's Lemma. Repeatedly

applying these steps to the remaining batches gives:

$$\begin{aligned}
\mathbb{E}[M_{t_j}] &\leq \mathbb{E}[M_{t_{j-1}}] \exp\left(\frac{\lambda^2(t_j - t_{j-1})}{8\alpha^2}\right) \\
&\leq \mathbb{E}[M_{t_{j-2}}] \exp\left(\frac{\lambda^2(t_{j-1} - t_{j-2})}{8\alpha^2}\right) \exp\left(\frac{\lambda^2(t_j - t_{j-1})}{8\alpha^2}\right) \\
&\vdots \\
&\leq \exp\left(\frac{\lambda^2 n_{\text{off}}}{8\alpha^2}\right) \exp\left(\frac{\lambda^2(t_2 - t_1)}{8\alpha^2}\right) \cdots \exp\left(\frac{\lambda^2(t_j - t_{j-1})}{8\alpha^2}\right) \\
&= \exp\left(\frac{\lambda^2 n_{\text{off}}}{8\alpha^2}\right) \exp\left(\frac{\lambda^2(t_2 - n_{\text{off}} - 1)}{8\alpha^2}\right) \cdots \exp\left(\frac{\lambda^2(t_j - t_{j-1})}{8\alpha^2}\right) \\
&= \exp\left(\frac{\lambda^2}{8\alpha^2} (n_{\text{off}} + t_2 - n_{\text{off}} - 1 + t_3 - t_2 + \cdots + t_j - t_{j-1})\right) \\
&= \exp\left(\frac{\lambda^2}{8\alpha^2} (t_j - 1)\right)
\end{aligned}$$

Bringing everything together gives:

$$\mathbb{P}\left(\hat{P}_j(a \succ \pi_{\text{ref}}) - P_\star(a \succ \pi_{\text{ref}}) \geq \epsilon\right) \leq \exp\left(-(t_j - 1) \left(\lambda \epsilon - \frac{\lambda^2}{8\alpha^2}\right)\right).$$

Choosing $\lambda = 4\alpha^2\epsilon^2$ to minimise the upper bound completes the proof. The second inequality in the statement of the result follows from the same arguments. $\square$

Lemma 7.2 holds for a fixed action-batch pair. Therefore, it is necessary to apply a union bound to obtain guarantees that hold for all action-batch pairs.

Lemma 7.3. *Let $H_{n_{\text{off}}}$ be the offline dataset collected as-per Equation (7.2). Then, it holds that:*

$$\begin{aligned}
&\mathbb{P}\left(\exists (a, j) \in \mathcal{A} \times \mathbb{N} : \hat{P}_j(a \succ \pi_{\text{ref}}) - P_\star(a \succ \pi_{\text{ref}}) \geq \frac{1}{\alpha} \sqrt{\frac{\ln\left(\frac{k j (j+1)}{\delta}\right)}{2(t_j - 1)}}\right) \leq \delta \\
&\mathbb{P}\left(\exists (a, j) \in \mathcal{A} \times \mathbb{N} : P_\star(a \succ \pi_{\text{ref}}) - \hat{P}_j(a \succ \pi_{\text{ref}}) \geq \frac{1}{\alpha} \sqrt{\frac{\ln\left(\frac{k j (j+1)}{\delta}\right)}{2(t_j - 1)}}\right) \leq \delta
\end{aligned}$$

where α is defined as per Equation (7.14).

Proof. From Lemma 7.2, it follows that:

$$\mathbb{P}\left(\hat{P}_j(a \succ \pi_{\text{ref}}) - P_\star(a \succ \pi_{\text{ref}}) \geq \frac{1}{\alpha} \sqrt{\frac{\ln\left(\frac{k}{\delta_j}\right)}{2(t_j - 1)}}\right) \leq \frac{\delta_j}{k}$$

$$\mathbb{P} \left(P_{\star}(a \succ \pi_{\text{ref}}) - \hat{P}_j(a \succ \pi_{\text{ref}}) \geq \frac{1}{\alpha} \sqrt{\frac{\ln \left(\frac{k}{\delta_j} \right)}{2(t_j - 1)}} \right) \leq \frac{\delta_j}{k}$$

Setting $\delta_j = \delta/j(j+1)$ and applying the union bound gives:

$$\begin{aligned} \mathbb{P} \left(\exists (a, j) \in \mathcal{A} \times \mathbb{N} : \hat{P}_j(a \succ \pi_{\text{ref}}) - P_{\star}(a \succ \pi_{\text{ref}}) \geq \frac{1}{\alpha} \sqrt{\frac{\ln \left(\frac{k}{\delta_j} \right)}{2(t_j - 1)}} \right) &\leq \sum_{a \in \mathcal{A}} \sum_{j=1}^{\infty} \frac{\delta_j}{k} \\ &= \delta \sum_{j=1}^{\infty} \frac{1}{j(j+1)} \\ &\leq \delta \end{aligned}$$

where the last inequality uses the fact that the summation is telescopic. Applying the same sequence of arguments to the second inequality gives an identical result and completes the proof. $\square$

Finally, Lemma 7.4 establishes a fact about policies of the form given in Equation (7.6). Specifically, it shows that the difference in regularised preference between an optimal policy for a given preference preference and any other policy is equivalent to the relative entropy between the policies.

Lemma 7.4. *Let $P_1 \in \mathbb{R}^{k \times k}$ and $\pi_2 : \mathcal{A} \rightarrow (0, 1]^k$ denote an arbitrary matrix and an arbitrary policy, respectively. Define the regularised preference function for each of these matrices as follows:*

$$Q_1(\pi, \pi_{\text{ref}}) = P_1(\pi \succ \pi_{\text{ref}}) - \eta D_{KL}(\pi \parallel \pi_{\text{ref}})$$

Then, it holds that:

$$Q_1(\pi_1, \pi_{\text{ref}}) - Q_1(\pi_2, \pi_{\text{ref}}) = \eta D_{KL}(\pi_2 \parallel \pi_1)$$

where $\pi_1 = \arg \max_{\pi} Q_1(\pi, \pi_{\text{ref}})$.

Proof. Expanding out the definitions of the regularised preference functions gives the following:

$$\begin{aligned} Q_1(\pi_1, \pi_{\text{ref}}) - Q_1(\pi_2, \pi_{\text{ref}}) &= P_1(\pi_1 \succ \pi_{\text{ref}}) - P_1(\pi_2 \succ \pi_{\text{ref}}) \\ &\quad + \eta D_{KL}(\pi_2 \parallel \pi_{\text{ref}}) - \eta D_{KL}(\pi_1 \parallel \pi_{\text{ref}}) \end{aligned} \tag{7.15}$$

From Equation (7.6) and recalling that π_1 is the maximiser of $Q_1(\pi, \pi_{\text{ref}})$ allows us to obtain the

following closed-form expression for these policies:

$$\pi_1(a) = \frac{\pi_{\text{ref}}(a) \exp\left(\frac{1}{\eta} P_1(a \succ \pi_{\text{ref}})\right)}{Z_P(\pi_{\text{ref}})}$$

where $Z_1(\pi_{\text{ref}}) = \sum_{a'} \pi_{\text{ref}}(a') \exp\left(\frac{1}{\eta} P_1(a' \succ \pi_{\text{ref}})\right)$. Rearranging this equality gives an expression for $P_1(a \succ \pi_{\text{ref}})$ in terms of π_1 :

$$P_1(a \succ \pi_{\text{ref}}) = \eta \ln(Z_1(\pi_{\text{ref}})) + \eta \ln\left(\frac{\pi_1(a)}{\pi_{\text{ref}}(a)}\right)$$

Plugging this into the preference probabilities of each policy over the reference policy allows us to show that these probabilities can be rewritten in terms of relative entropies:

$$\begin{aligned} P(\pi_1 \succ \pi_{\text{ref}}) - P(\pi_2 \succ \pi_{\text{ref}}) &= \sum_{a \in \mathcal{A}} \pi_1(a) P_1(a \succ \pi_{\text{ref}}) - \sum_{a \in \mathcal{A}} \pi_2(a) P_1(a \succ \pi_{\text{ref}}) \\ &= \sum_{a \in \mathcal{A}} \pi_1(a) \left(\eta \ln(Z_1(\pi_{\text{ref}})) + \eta \ln\left(\frac{\pi_1(a)}{\pi_{\text{ref}}(a)}\right) \right) - \sum_{a \in \mathcal{A}} \pi_2(a) \left(\eta \ln(Z_1(\pi_{\text{ref}})) + \eta \ln\left(\frac{\pi_1(a)}{\pi_{\text{ref}}(a)}\right) \right) \\ &= \eta \sum_{a \in \mathcal{A}} \pi_1(a) \ln\left(\frac{\pi_1(a)}{\pi_{\text{ref}}(a)}\right) - \eta \sum_{a \in \mathcal{A}} \pi_2(a) \ln\left(\frac{\pi_1(a)}{\pi_{\text{ref}}(a)}\right) \\ &= \eta D_{\text{KL}}(\pi_1 \parallel \pi_{\text{ref}}) - \eta \sum_{a \in \mathcal{A}} \pi_2(a) \ln\left(\frac{\pi_1(a)}{\pi_{\text{ref}}(a)}\right) \quad (\text{Definition of } D_{\text{KL}}(\cdot \parallel \cdot)) \\ &= \eta D_{\text{KL}}(\pi_1 \parallel \pi_{\text{ref}}) - \eta \sum_{a \in \mathcal{A}} \pi_2(a) \ln\left(\frac{\pi_1(a)}{\pi_{\text{ref}}(a)} \frac{\pi_2(a)}{\pi_2(a)}\right) \\ &= \eta D_{\text{KL}}(\pi_1 \parallel \pi_{\text{ref}}) - \eta \sum_{a \in \mathcal{A}} \pi_2(a) \ln\left(\frac{\pi_2(a)}{\pi_{\text{ref}}(a)}\right) - \eta \sum_{a \in \mathcal{A}} \pi_2(a) \ln\left(\frac{\pi_1(a)}{\pi_2(a)}\right) \\ &= \eta D_{\text{KL}}(\pi_1 \parallel \pi_{\text{ref}}) - \eta D_{\text{KL}}(\pi_2 \parallel \pi_{\text{ref}}) + \eta \sum_{a \in \mathcal{A}} \pi_2(a) \ln\left(\frac{\pi_2(a)}{\pi_1(a)}\right) \\ &= \eta D_{\text{KL}}(\pi_1 \parallel \pi_{\text{ref}}) - \eta D_{\text{KL}}(\pi_2 \parallel \pi_{\text{ref}}) + \eta D_{\text{KL}}(\pi_2 \parallel \pi_1) \end{aligned}$$

Plugging this equality into Equation (7.15) gives:

$$\begin{aligned} Q_1(\pi_1, \pi_{\text{ref}}) - Q_1(\pi_2, \pi_{\text{ref}}) &= P_1(\pi_1 \succ \pi_{\text{ref}}) - P_1(\pi_2 \succ \pi_{\text{ref}}) + \eta D_{\text{KL}}(\pi_2 \parallel \pi_{\text{ref}}) - \eta D_{\text{KL}}(\pi_1 \parallel \pi_{\text{ref}}) \\ &= \eta D_{\text{KL}}(\pi_1 \parallel \pi_{\text{ref}}) - \eta D_{\text{KL}}(\pi_2 \parallel \pi_{\text{ref}}) + \eta D_{\text{KL}}(\pi_2 \parallel \pi_1) + \eta D_{\text{KL}}(\pi_2 \parallel \pi_{\text{ref}}) - \eta D_{\text{KL}}(\pi_1 \parallel \pi_{\text{ref}}) \\ &= \eta D_{\text{KL}}(\pi_2 \parallel \pi_1) \end{aligned}$$

as required. $\square$

7.4.2 Proof of Theorem 7.1

Proof. Define the estimated regularised preference function corresponding to the j -th batch as follows:

$$\hat{Q}_j(\pi, \pi_{\text{ref}}) = \hat{P}_j(\pi \succ \pi_{\text{ref}}) - \eta D_{\text{KL}}(\pi \| \pi_{\text{ref}}).$$

for any $\pi \in \Pi$ where Π denotes the set of all probability distributions over actions. Further, define the following *bad* events:

$$B_1 = \left\{ \exists (a, j) \in \mathcal{A} \times \mathbb{N} : \hat{P}_j(a \succ \pi_{\text{ref}}) - P_*(a \succ \pi_{\text{ref}}) \geq \frac{1}{\alpha} \sqrt{\frac{\ln \left(\frac{k j (j+1)}{\delta} \right)}{2(t_j - 1)}} \right\}$$

$$B_2 = \left\{ \exists (a, j) \in \mathcal{A} \times \mathbb{N} : P_*(a \succ \pi_{\text{ref}}) - \hat{P}_j(a \succ \pi_{\text{ref}}) \geq \frac{1}{\alpha} \sqrt{\frac{\ln \left(\frac{k j (j+1)}{\delta} \right)}{2(t_j - 1)}} \right\}$$

which is the event where the estimated probability of preferring each action to the ground deviates too far from the true preferences. From Lemma 7.3, it follows that the probability of either of the bad events occurring can be controlled:

$$\mathbb{P}(B_1 \cup B_2) \leq \mathbb{P}(B_1) + \mathbb{P}(B_2) \leq 2\delta$$

Therefore, the probability that either of these bad events occurs can be controlled with arbitrarily high probability. The remainder of the analysis will assume that the *good* event holds:

$$G = (B_1 \cup B_2)^C$$

where $\mathbb{P}(G) = 1 - \mathbb{P}(B_1 \cup B_2) \geq 1 - 2\delta$. This gives the lower bound on the probability with which the guarantees hold.

From Algorithm 10, $\pi_j = \arg \max_{\pi} \hat{Q}_j(\pi, \pi_{\text{ref}})$ is chosen to be the maximiser of the estimated regularised preference function. Expanding the per-round regret that is incurred by the policy executed within the j -th batch gives the following:

$$\begin{aligned} Q_*(\pi_*, \pi_{\text{ref}}) - Q_*(\pi_{t_j}, \pi_{\text{ref}}) &= Q_*(\pi_*, \pi_{\text{ref}}) - Q_*(\pi_{t_j}, \pi_{\text{ref}}) \pm \hat{Q}_j(\pi_*, \pi_{\text{ref}}) \pm \hat{Q}_j(\pi_{t_j}, \pi_{\text{ref}}) \\ &= \left[Q_*(\pi_*, \pi_{\text{ref}}) - \hat{Q}_j(\pi_*, \pi_{\text{ref}}) \right] \end{aligned} \quad (7.16)$$

$$+ \left[\hat{Q}_j(\pi_{t_j}, \pi_{\text{ref}}) - Q_*(\pi_{t_j}, \pi_{\text{ref}}) \right] \quad (7.17)$$

$$+ \left[\hat{Q}_j(\pi_\star, \pi_{\text{ref}}) - \hat{Q}_j(\pi_{t_j}, \pi_{\text{ref}}) \right] \quad (7.18)$$

Equation (7.16) can be interpreted as the error associated with estimating the regularised preference function for the optimal policy. Equation (7.17) can be interpreted as the error associated with estimating the regularised preference function for the policy executed in the j -th batch. Finally, Equation (7.18) can be thought of as the optimisation error. These three terms will be controlled separately.

Expanding Equation (7.16) and using the definition of preferences between policies gives:

$$\begin{aligned} (7.16) &= P_\star(\pi_\star \succ \pi_{\text{ref}}) - \hat{P}_j(\pi_\star \succ \pi_{\text{ref}}) - \eta [\text{D}_{\text{KL}}(\pi_\star \parallel \pi_{\text{ref}}) - \text{D}_{\text{KL}}(\pi_\star \parallel \pi_{\text{ref}})] \\ &= P_\star(\pi_\star \succ \pi_{\text{ref}}) - \hat{P}_j(\pi_\star \succ \pi_{\text{ref}}) \\ &= \sum_{a \in \mathcal{A}} \left[P_\star(a \succ \pi_{\text{ref}}) - \hat{P}_j(a \succ \pi_{\text{ref}}) \right] \pi_\star(a) \\ &\leq \frac{1}{\alpha} \sqrt{\frac{\ln \left(\frac{k j (j+1)}{\delta} \right)}{2(t_j - 1)}} \sum_{a \in \mathcal{A}} \pi_\star(a) \quad (\text{Since } G \text{ holds}) \\ &= \frac{1}{\alpha} \sqrt{\frac{\ln \left(\frac{k j (j+1)}{\delta} \right)}{2(t_j - 1)}} \end{aligned}$$

For Equation (7.17), the same steps yield an identical upper bound:

$$\begin{aligned} (7.17) &= \hat{P}_j(\pi_{t_j} \succ \pi_{\text{ref}}) - P_\star(\pi_j \succ \pi_{\text{ref}}) - \eta [\text{D}_{\text{KL}}(\pi_{t_j} \parallel \pi_{\text{ref}}) - \text{D}_{\text{KL}}(\pi_{t_j} \parallel \pi_{\text{ref}})] \\ &= \sum_{a \in \mathcal{A}} \left[\hat{P}_j(a \succ \pi_{\text{ref}}) - P_\star(a \succ \pi_{\text{ref}}) \right] \pi_{t_j}(a) \quad (\text{Equation (7.4)}) \\ &\leq \frac{1}{\alpha} \sqrt{\frac{\ln \left(\frac{k j (j+1)}{\delta} \right)}{2(t_j - 1)}} \sum_{a \in \mathcal{A}} \pi_\star(a) \quad (\text{Since } G \text{ holds}) \\ &= \frac{1}{\alpha} \sqrt{\frac{\ln \left(\frac{k j (j+1)}{\delta} \right)}{2(t_j - 1)}} \end{aligned}$$

Finally, Equation (7.18) can be controlled by noticing that the policy that is executed in the j -th batch is the maximiser of the estimated regularised preference probabilities. Therefore, Lemma 7.4 applies and this gives the following:

$$(7.18) = -\eta \text{D}_{\text{KL}}(\pi_\star \parallel \pi_{t_j}).$$

Bringing everything together yields an upper bound on the regret incurred in each round of the j -th batch:

$$\begin{aligned}
Q_\star(\pi_\star, \pi_{\text{ref}}) - Q_\star(\pi_{t_j}, \pi_{\text{ref}}) &= (7.16) + (7.17) + (7.18) \\
&\leq \frac{2}{\alpha} \sqrt{\frac{\ln\left(\frac{k j (j+1)}{\delta}\right)}{2(t_j - 1)}} - \eta \text{D}_{\text{KL}}(\pi_\star \parallel \pi_{t_j}) \\
&\leq \frac{2}{\alpha} \sqrt{\frac{\ln\left(\frac{k m (m+1)}{\delta}\right)}{2(t_j - 1)}} - \eta \text{D}_{\text{KL}}(\pi_\star \parallel \pi_{t_j})
\end{aligned}$$

Finally, summing across the rounds gives:

$$\begin{aligned}
\hat{R}_n(\pi, \nu) &= \sum_{t=n_{\text{off}}+1}^n Q_\star(\pi_\star, \pi_{\text{ref}}) - \max\{Q_\star(\pi_t, \pi_{\text{ref}}), Q_\star(\pi_{\text{ref}}, \pi_{\text{ref}})\} \\
&\leq \sum_{t=n_{\text{off}}+1}^n Q_\star(\pi_\star, \pi_{\text{ref}}) - Q_\star(\pi_t, \pi_{\text{ref}}) \\
&= \sum_{j=1}^{m-1} \sum_{t=t_j}^{t_{j+1}-1} Q_\star(\pi_\star, \pi_{\text{ref}}) - Q_\star(\pi_t, \pi_{\text{ref}}) \\
&= \sum_{j=1}^{m-1} \sum_{t=t_j}^{t_{j+1}-1} Q_\star(\pi_\star, \pi_{\text{ref}}) - Q_\star(\pi_{t_j}, \pi_{\text{ref}}) \quad (\pi_t = \pi_{t_j} \text{ for } t \in [t_j, t_{j+1})) \\
&= \sum_{j=1}^{m-1} (t_{j+1} - t_j) (Q_\star(\pi_\star, \pi_{\text{ref}}) - Q_\star(\pi_{t_j}, \pi_{\text{ref}})) \\
&\leq \frac{\sqrt{2 \ln\left(\frac{k m (m+1)}{\delta}\right)}}{\alpha} \sum_{j=1}^{m-1} \frac{(t_{j+1} - t_j)}{\sqrt{t_j - 1}} - \eta \sum_{j=1}^{m-1} (t_{j+1} - t_j) \text{D}_{\text{KL}}(\pi_\star \parallel \pi_{t_j})
\end{aligned}$$

with probability at least $1 - 2\delta$, as required. $\square$

7.5 Conclusion

This chapter, to the best of our knowledge, presents the first algorithm that comes with guarantees on the regret in the setting of identity preference optimisation. Algorithm 10 is computationally efficient and based on importance-weighted estimators. Furthermore, it does not require strong assumptions on the form of the unknown preferences and the theoretical guarantees hold in settings of practical interest.

Theorem 7.1 shows that an appropriately chosen importance-weighted estimator yields sub-linear

guarantees on the regret. These guarantees hold irrespective of the batching scheme. Corollaries 7.1 and 7.2 investigate constant-size batches and geometrically increasing batch sizes.

However, the guarantees depend on α , which denotes the minimum probability of selecting an action in any of the rounds. The best-case scenario, which arises from maximising the minimum action probability, would give:

$$\alpha \leq \max_{\pi \in \Pi} \left\{ \min_{a \in \mathcal{A}} \pi(a) \right\} = \frac{1}{k}.$$

where Π is the set containing all probability distributions over $\mathcal{A} = \{1, 2, \dots, k\}$. Therefore, Theorem 7.1 has an implicit dependence on the number of actions. This dependence, in the best case, scales the leading-order term by a factor of k . Generally, $\alpha \ll 1/k$ and the precise value will depend on the problem instance and the policies computed by the algorithm. Therefore, it is of theoretical and practical interest to remove this dependence in the guarantees and potentially obtain a better scaling with the number of actions.

There are several ways to push the dependence on α to a lower-order term that have been explored in the offline contextual bandit literature, where a term similar to α appears. For example, Bernstein-like arguments could have been used within the analysis. However, α would still feature in the regret bound. Typically, the offline contextual bandit literature addresses the issue of policies assigning a low probability to an action via clipping (Bottou et al., 2013; Swaminathan and Joachims, 2015). For Algorithm 10, this would correspond to using the following re-weighting of the existing preference dataset:

$$\hat{X}_t(a) = \frac{1\{A_t = a\}X_t}{\max\{\gamma, \pi_t(a)\}}$$

where $\gamma > 0$ is a tunable hyperparameter. This guarantees $\alpha \geq \gamma$. However, the clipped version of the estimator is not easy to analyse due to the max in the denominator. Alternatively, the adversarial bandit literature has developed the *implicit exploration estimator*, which can be seen as a smoothed version of clipping (Kocák et al., 2014; Neu, 2015). Gabbianelli et al. (2024) adapt this estimator for the offline contextual bandit problem. For Algorithm 10, this would correspond to using the following re-weighting of the existing preference dataset:

$$\hat{X}_t(a) = \frac{1\{A_t = a\}X_t}{\gamma + \pi_t(a)}$$

where $\gamma > 0$ is a tunable hyperparameter. Again, $\alpha \geq \gamma$ is guaranteed to hold for this estimator too. Future research could utilise either of these adjustments to the importance-weighted estimator within Algorithm 10. This will likely improve the α -dependence within the regret bound of Theorem 7.1.

Chapter 8

Conclusion

This thesis has studied several extensions of classical sequential decision-making models. These extensions allow for the possibility of delayed feedback, cooperation, and feedback in the form of pairwise comparisons. By studying these extensions, progress has been made towards bridging the gap between the theoretical literature and practical applications. Section 8.1 summarises the contributions the thesis makes to the existing literature. Section 8.2 presents avenues for future research on the topics of delayed feedback, multi-agent bandit problems, and learning from pairwise comparisons. Finally, Section 8.3 presents closing remarks.

8.1 Summary of Contributions

Chapter 4 focuses on the generalised linear bandit problem. Typically, algorithms for this problem setting are designed and analysed under the assumption of immediate feedback. However, feedback is rarely immediate in practical applications ([Chapelle, 2014](#); [Villar et al., 2015](#); [Trella et al., 2024](#); [Zhang et al., 2024](#)). Prior work has investigated the impact of stochastic delays in multi-armed bandit problems and has proven that stochastic delays increased the regret of the agent by an additive term involving the expected delay ([Joulani et al., 2013](#); [Mandel et al., 2015](#); [Pike-Burke et al., 2018](#)). However, this result had not been replicated in bandit problems with structure, where existing algorithms suffered a multiplicative penalty caused by the delays ([Zhou et al., 2019](#); [Blanchet et al., 2020](#)). Therefore, we proposed a new algorithm for generalised linear bandits with delayed feedback: Delayed OFU-GLM. Theoretically, this algorithm is able to recover the additive delay-dependence from the multi-armed bandit setting. These findings substantially improve upon the best-known regret bounds for the problem setting. This

improvement was possible due to the construction of tighter confidence sets for the setting of delayed feedback and the introduction of new analysis tools for separating the delays from the difficulty of the learning problem.

Chapter 5 further investigated the impact of delayed feedback, but in the setting of episodic reinforcement learning. Here, the existing literature was limited. Furthermore, it was unclear whether it was possible to achieve regret bounds that separated the impact of the delays from the difficulty of the episodic reinforcement learning problem (Lancewicki et al., 2022). Therefore, we proposed two meta-algorithms: active updating and lazy updating. Rather than focusing our analysis on a single algorithm, we built on prior work to obtain theoretical guarantees for the class of optimistic model-based algorithms. For active updating, a strategy that updates the policy as soon as new feedback arrives, we found that an additive delay-dependence can be achieved without degrading the leading-order term for some algorithms. However, this was not possible for the algorithms with better theoretical guarantees in the immediate feedback setting. This arises because the algebraic parts of the analysis introduced delay-dependencies into what usually become logarithmic terms in the regret bounds. Therefore, lazy updating was proposed to avoid this penalty. This meta-algorithm introduces an algorithmic technique for handling delayed feedback: waiting before updating the policy. We show that increasing the waiting period geometrically can be used to separate the difficulty introduced by delayed feedback from the difficulty of the episodic reinforcement learning problem. Consequently, when running lazy updating with any optimistic model-based algorithm, we proved that the regret increases by an additive term that involves the expected delay.

Chapter 6 was motivated by the growing need for distributed algorithms for decision-making and explores the cooperative multi-agent multi-armed bandit problem. Previously, the literature addressing this problem all focussed on extending specific algorithms from the standard multi-armed bandit problem (Landgren et al., 2016; Martínez-Rubio et al., 2019; Lalitha and Goldsmith, 2021). This chapter introduced a novel black-box approach that uses a queue-based technique to simulate a single-agent environment for any input bandit algorithm. Consequently, the regret suffered by the network of agents can be related to that of the single-agent algorithm in the queued version of the environment. This result holds for any multi-armed bandit algorithm. Furthermore, it was shown that combining the black-box reduction with a minimax optimal algorithm gives the first algorithm that achieves minimax optimality, up to an additive

graph-dependent constant, in the multi-agent setting. Despite being more general, the guarantees are competitive with or better than the existing case-by-case analyses. These guarantees hold under mild assumptions about the bandit environment. Consequently, it is straightforward to derive guarantees for numerous multi-agent extensions of other bandit problems. This was demonstrated by investigating heavy-tailed bandits and duelling bandits, which were previously of independent interest (Dubey and Pentland, 2020a; Raveh et al., 2024). Additionally, the black-box reduction can be used to obtain the first theoretical guarantees for a multi-agent extension of bandits with local differential privacy.

Finally, Chapter 7 explores learning from pairwise comparisons. Typically, prior work makes use of the Bradley-Terry assumption (Zhu et al., 2023; Xiong et al., 2024; Xie et al., 2025). This assumption restricts the learnable preference structures and, as such, may fail to hold in settings where human-generated preferences are involved. This chapter advocates for the identity preference optimisation framework (Azar et al., 2024). For this framework, a problem-specific importance-weighted estimator of the preference probabilities was proposed. This estimator was used to develop an algorithm that can be warm-started with an offline dataset and comes with theoretical guarantees on the regret. These guarantees hold in settings where the updating frequency is constrained, including constant and geometrically increasing batch sizes, which better aligns with practical implementations. These guarantees establish initial positive results for regret minimisation within the identity preference optimisation framework.

8.2 Future Directions

8.2.1 Delayed Feedback

There are several promising directions for extending the study of delayed feedback in structured bandit problems. One natural approach is to build on the existing framework of Chapter 4, exploring refinements and theoretical improvements under the same set of assumptions. These extensions have been discussed in Section 4.9. Beyond these direct extensions, another avenue for future research is to introduce additional complexities into the problem settings. These additional complexities should consider relaxing the assumptions to develop practical algorithms for real-world problems. For instance, the following modifications could be considered.

- **Intermediate Rewards.** Often, practical applications allow for a partial observation of the rewards before they are fully observed. Eick (1988) considers such a problem in the

context of clinical trials. Here, the agent is trying to identify the action that maximises the expected survival time. Therefore, the delay and the reward are equivalent. For applications in clinical trials, the agent may be permitted to observe whether each of the previously treated patients has survived until the end of the previous round (Eick, 1988). This additional information, which is directly related to the reward, could then be used to improve the decision-making for future patients. Here, a natural approach would be to combine the algorithmic principles from the bandit literature with the modelling tools from survival analysis (Cox, 1984).

- **Intermediate Observations.** Esposito et al. (2023) consider a related problem of getting to observe an intermediate state variable immediately after playing an action in the k -armed adversarial bandit problem. This is different from the intermediate reward setting, where new reward-related information becomes available for each of the previously chosen actions at the end of each round. The single intermediate observation has been shown to improve the theoretical guarantees in the adversarial multi-armed bandit problem (Esposito et al., 2023). Investigating this problem setting within structured bandits would be of practical relevance to online advertising and recommender systems. Here, an example of an intermediate signal could be clicking the advert, and the reward could represent a purchase. Recently, Zhang et al. (2025) have made steps towards understanding the potential improvements that intermediate observations can offer and propose a variant of Thompson Sampling for settings where the expected reward for each action is a linear function.
- **Reward-Dependent Delays.** Other real-world applications might have delays that are correlated with random values of the rewards. Conversely, Chapter 4 assumed that the delays and the rewards were independent. However, failing to account for existing correlations could lead to poorly-calibrated uncertainty quantification and lead to sub-optimal decision-making. For instance, a negative correlation would have the effect of withholding information for sub-optimal actions. Consequently, algorithms failing to account for this correlation would keep selecting these actions in an attempt to reduce their uncertainty. Ultimately, this would lead to worse performance. The reverse would be true if the correlation between the delays and the rewards were positive. Here, the confidence sets will need to be modified. Further, it is expected that efficient algorithms will exploit the correlation between the delays and the rewards. This problem has already received attention in the

multi-armed bandit setting (Lancewicki et al., 2021). However, it remains an interesting problem that is unexplored in structured settings.

8.2.2 Multi-Agent Bandits

Section 6.8 of Chapter 6 outlines some specific extensions of the algorithm that we proposed for the cooperative multi-agent bandit problem. Furthermore, it highlights the absence of graph-dependent lower bounds for the multi-agent multi-armed bandit problem. However, the model for sequential decision-making is quite far removed from reality. Future work could focus on extending the ideas developed in Chapter 6 to problem settings that are closer to real-world applications, some of which are listed below.

- Asynchronous Settings.** One obvious pitfall with the model considered in Chapter 6 is that all agents in the network make decisions at the same time. However, it is unrealistic to expect this to be the case in real-world applications. For example, recommender systems with geographically distributed servers will likely have different periods of peak activity due to differing time zones. Here, the exact points in time where agents make their decisions need not be the same. Prior work has explored communication strategies in such settings (Sankararaman et al., 2019; Yang et al., 2021; Chen et al., 2023). The queuing approach proposed in Chapter 6 might be useful in this setting. For example, the queues could be used by the agents between making decisions. Investigating exactly how to adapt the queuing idea to the asynchronous setting would be an interesting direction for future research that would better model real-world applications.
- Constrained and Unreliable Communication.** Chapter 6 assumes unrestricted and perfect communication among the agents. This assumption is standard in the literature that considers the same problem setting (Landgren et al., 2016; Martínez-Rubio et al., 2019; Lalitha and Goldsmith, 2021). However, communicating information can be costly and unreliable in real-world networks. For instance, we assumed that sending a message was always successful. This need not be true in practice, as communication channels might break and fail to deliver the message to the intended recipient with some probability. Madhushani et al. (2021) have developed an algorithm based on the optimistic principle for this problem setting. Investigating whether Algorithm 6 is robust to such scenarios would be valuable for real-world problems, especially if degradation in performance can be quantified in the regret bound. Alternatively, the agents might be limited in the number of

messages they can send, or constrained in the amount of information they can put in the messages. One approach to reducing the amount of communication would be to integrate successive elimination strategies into Algorithm 6. These kinds of bandit algorithms have shown effectiveness in communication-limited settings ([Agarwal et al., 2022](#)).

- **Heterogenous Environments.** Chapter 6 assumes a shared bandit environment across all agents. However, in geographically distributed systems, the reward distributions may vary between locations. Here, there are several interesting questions that could be asked, as it is not clear what would constitute the optimal policy. [Shahrampour et al. \(2017\)](#) focus on the setting where it is of interest to identify the action with the highest network-averaged reward. However, some agents could end up playing an action with a low expected reward in their own environment and be sacrificed to ensure better group performance. The downstream consequences of this could lead to customers dis-engaging with the platform due to receiving bad recommendations. Alternatively, [Hossain et al. \(2021\)](#) seek to learn a distribution over actions that maximises the Nash Social Welfare. Currently, algorithms for this objective appear to be difficult to implement and devising computationally efficient alternatives would be of practical interest. Finally, one could allow for correlations between the rewards generated by each of the bandit environments. Here, the focus could be placed on analysing the regret of each agent individually and devising algorithms that exploit the correlation to utilise feedback from the other agents to improve their own decision-making.

8.2.3 Pairwise Comparisons

Section 7.5 of Chapter 7 outlines some direct extensions of Algorithm 10. However, there are other avenues for future research that appear promising.

- **Large Action Sets.** Chapter 7 assumes a finite number of actions. This is standard in the literature and it is needed to ensure that computing the policy in closed-form is computationally feasible ([Zhu et al., 2023](#); [Xiong et al., 2024](#); [Ye et al., 2024](#)). This computational challenge arises from the difficulty of evaluating the normalising constant when the action space is infinite. Indeed, this is somewhat analogous to estimating the normalising constant in Bayesian inference. However, [Rafailov et al. \(2023\)](#) circumvent this problem under the Bradley-Terry assumption. They achieve this by considering a parametric class of policies and reformulating the closed-form solution to express preference probabilities directly in terms of the policy. This transformation allowed them to optimise policy parameters via

maximum likelihood without ever having to explicitly compute the normalising constant. [Xie et al. \(2025\)](#) build on this idea and analyse a variant of the approach for the pure exploration setting. Nevertheless, extending theoretical guarantees for such an approach without the Bradley-Terry assumption remains an open problem. Notably, [Azar et al. \(2024\)](#) derive a similar loss function for the identity preference optimisation setting that was considered in Chapter 7. This loss function could be used to optimise the parameters of the policy directly. A promising direction for future research could take this approach and look to develop provably efficient algorithms that scale to large or infinite action sets.

- **Strong Convexity.** The relative entropy regulariser is strongly convex whenever the policy assigns positive probability to each action. This can be seen by computing the Hessian. Recently [Lan \(2022\)](#) have shown that strongly convex regularisers can lead to improved theoretical guarantees in reinforcement learning problems. Furthermore, strongly convex objective functions are known to give faster rates of convergence in optimisation ([Garrigos and Gower, 2024](#)). However, this fact is yet to be exploited when designing algorithms for the setting of learning from preferences. Future research could look to exploit the strong convexity of the relative entropy regulariser within the context of identity preference optimisation objective function to devise faster algorithms.

8.3 Closing Remarks

The study of bandit and reinforcement learning problems has significantly advanced our understanding of sequential decision-making in uncertain environments. This has allowed us to progress beyond the classical multi-armed bandit and episodic reinforcement learning problems to obtain theoretical guarantees in problem settings that better reflect real-world applications. The work conducted in this thesis shows promising results in simulated environments. However, they are yet to be applied in a real-world application, and it would be interesting to assess their effectiveness in such scenarios. Furthermore, deploying the algorithms in a practical application would help to identify where the mathematical models of the environment fail to account for real-world complexities and would generate interesting and practically meaningful directions for future research.

Generally, the theoretical community has been highly successful in providing a rigorous understanding of isolated components of certain challenges that might arise in real-world problems.

However, it is often the case that such problems involve combinations of many of these isolated parts, and combining these methods is usually non-trivial. A more general direction for future research could consider problem settings that are combinations of these isolated parts. This would lead to the development of algorithms that better align with problems that often arise in real-world applications. Despite being challenging, this approach has already found some success. For instance, the challenge of balancing exploration and exploitation was initially addressed in isolation ([Thompson, 1933](#); [Auer et al., 2002a](#)). Subsequently, these algorithmic principles were integrated with long-term planning methods to develop provably efficient reinforcement learning algorithms ([Jaksch et al., 2010](#); [Russo and Roy, 2014](#)). It would be exciting to see more success stories where innovative and principled algorithms are derived for combinations of existing problem settings.

Beyond this, there are other differences between what works in theory and what is applied in practice. Generally, Thompson Sampling and the optimistic principle serve as the starting points for developing provably efficient algorithms. Yet, implementing these strategies in practice can be computationally intractable, especially when the unknown expected reward function is non-linear. Therefore, real-world applications frequently rely on different exploration strategies, such as ϵ -greedy or gradient-based approaches. While computationally feasible, ϵ -greedy is far from theoretically optimal in scenarios where key problem-dependent parameters are unknown. Additionally, research on gradient-based methods for exploration is limited. Thus, further research on the theoretical properties of scalable exploration mechanisms would be valuable. Recently, there has been progress on this front ([Mei et al., 2020, 2023](#)). These results are promising but are so far limited to the multi-armed bandit problem. Generalising these results to structured bandit problems could yield insights that are immediately useful in practice.

Bibliography

- Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved Algorithms for Linear Stochastic Bandits. In *Advances in Neural Information Processing Systems*, volume 24, 2011.
- N. Abe and P. M. Long. Associative Reinforcement Learning using Linear Probabilistic Concepts. In *Proceedings of the 16th International Conference on Machine Learning*, page 3–11, San Francisco, CA, USA, 1999.
- M. Abeille and A. Lazaric. Linear Thompson Sampling Revisited. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 176–184, 2017.
- M. Abeille, L. Faury, and C. Calauzenes. Instance-Wise Minimax-Optimal Algorithms for Logistic Bandits. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130, pages 3691–3699, 2021.
- A. Agarwal and J. C. Duchi. Distributed Delayed Stochastic Optimization. In *Advances in Neural Information Processing Systems*, volume 24, 2011.
- M. Agarwal, V. Aggarwal, and K. Azizzadenesheli. Multi-Agent Multi-Armed Bandits with Limited Communication. *Journal of Machine Learning Research*, 23(212):1–24, 2022.
- R. Agrawal. Sample Mean Based Index Policies with $O(\log n)$ Regret for the Multi-Armed Bandit Problem. *Advances in Applied Probability*, 27(4):1054–1078, 1995.
- S. Agrawal and N. Goyal. Analysis of Thompson Sampling for the Multi-armed Bandit Problem. In *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23, pages 39.1–39.26, 2012.
- S. Agrawal and N. Goyal. Further Optimal Regret Bounds for Thompson Sampling. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, volume 31, pages 99–107, 2013a.

- S. Agrawal and N. Goyal. Thompson Sampling for Contextual Bandits with Linear Payoffs. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pages 127–135, 2013b.
- S. Agrawal and R. Jia. Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 1184–1194, 2017.
- R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, 1993.
- N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1):137–147, 1999.
- D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete Problems in AI Safety, 2016.
- F. J. Anscombe. Sequential Medical Trials. *Journal of the American Statistical Association*, 58(302):365–383, 1963.
- J. Audibert and S. Bubeck. Minimax Policies for Adversarial and Stochastic Bandits. In *Proceedings of the 22nd Annual Conference on Computational Learning Theory*, pages 217–226, 2009.
- P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3:397–422, 2002.
- P. Auer and R. Ortner. Logarithmic Online Regret Bounds for Undiscounted Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 19, 2006.
- P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 322–331, 1995.
- P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 2002a.
- P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The Nonstochastic Multiarmed Bandit Problem. *SIAM Journal on Computing*, 32(1):48–77, 2002b.

- M. G. Azar, I. Osband, and R. Munos. Minimax Regret Bounds for Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 263–272, 2017.
- M. G. Azar, Z. D. Guo, B. Piot, R. Munos, M. Rowland, M. Valko, and D. Calandriello. A General Theoretical Paradigm to Understand Learning from Human Preferences. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238, pages 4447–4455, 2024.
- Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, N. Joseph, S. Kadavath, J. Kernion, T. Conerly, S. El-Showk, N. Elhage, Z. Hatfield-Dodds, D. Hernandez, T. Hume, S. Johnston, S. Kravec, L. Lovitt, N. Nanda, C. Olsson, D. Amodei, T. Brown, J. Clark, S. McCandlish, C. Olah, B. Mann, and J. Kaplan. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback, 2022.
- Y. Bar-On and Y. Mansour. Individual Regret in Cooperative Nonstochastic Multi-Armed Bandits. In *Advances in Neural Information Processing Systems*, 2019.
- A. Baransi, O. Maillard, and S. Mannor. Sub-sampling for multi-armed bandits. In *Machine Learning and Knowledge Discovery in Databases*, pages 115–131, 2014.
- P. L. Bartlett and A. Tewari. REGAL: A Regularization based Algorithm for Reinforcement Learning in Weakly Communicating MDPs, 2012.
- R. Bellman. A Markovian Decision Process. *Journal of Mathematics and Mechanics*, 6(5): 679–684, 1957.
- R. Bellman. On a Routing Problem. *Quarterly of Applied Mathematics*, 1958.
- V. Bengs, R. Busa-Fekete, A. Mesaoudi-Paul, and E. Hüllermeier. Preference-based Online Learning with Dueling Bandits: A Survey. *Journal of Machine Learning Research*, 2021.
- D. Bertsekas. *Dynamic Programming and Optimal Control: Approximate Dynamic Programming*. Athena Scientific optimization and computation series. Athena Scientific, 2012.
- D. P. Bertsekas and R. G. Gallager. *Data Networks*. Prentice-Hall, 1992.
- L. Besson and E. Kaufmann. What Doubling Tricks Can and Can’t Do for Multi-Armed Bandits, 2018.

- J. Bian and K. Jun. Maillard Sampling: Boltzmann Exploration Done Optimally. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, 2022.
- P. J. Bickel. On Some Robust Estimates of Location. *The Annals of Mathematical Statistics*, 36(3):847–858, 1965.
- J. Blanchet, R. Xu, and Z. Zhou. Delay-Adaptive Learning in Generalized Linear Contextual Bandits. *arXiv*, abs/2003.05174, 2020.
- L. Bottou, J. Peters, J. Quiñonero-Candela, D. X. Charles, D. M. 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(101):3207–3260, 2013.
- E. Boursier and V. Perchet. A Survey on Multi-player Bandits. *Journal of Machine Learning Research*, 2024.
- S. Boyd, A. Ghosh., B. Prabhakar, and D. Shah. Gossip Algorithms: Design, Analysis and Applications. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 3, pages 1653–1664, 2005.
- R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 1952.
- R. I. Brafman and M. Tennenholtz. R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213–231, 2003.
- S. Bubeck and N. Cesa-Bianchi. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.
- S. Bubeck, G. Stoltz, C. Szepesvári, and R. Munos. Online Optimization in X-Armed Bandits. In *Advances in Neural Information Processing Systems*, volume 21, 2008.
- S. Bubeck, N. Cesa-Bianchi, and G. Lugosi. Bandits With Heavy Tail. *IEEE Transactions on Information Theory*, 2013a.
- S. Bubeck, V. Perchet, and P. Rigollet. Bounded regret in stochastic multi-armed bandits. In *Proceedings of the 26th Annual Conference on Learning Theory*, volume 30, pages 122–134, 2013b.

- Q. Cai, Z. Yang, C. Jin, and Z. Wang. Provably efficient exploration in policy optimization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1283–1294, 2020.
- O. Cappé, A. Garivier, O. Maillard, R. Munos, and G. Stoltz. Kullback–Leibler upper confidence bounds for optimal sequential allocation. *The Annals of Statistics*, 41(3), 2013.
- A. Carpentier, C. Vernade, and Y. Abbasi-Yadkori. The Elliptical Potential Lemma Revisited, 2020.
- O. Catoni. Challenging the empirical mean and empirical variance: A deviation study. *Annales de l’Institut Henri Poincaré, Probabilités et Statistiques*, 48(4):1148 – 1185, 2012.
- N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.
- N. Cesa-Bianchi, C. Gentile, Y. Mansour, and A. Minora. Delay and Cooperation in Nonstochastic Bandits. In *29th Annual Conference on Learning Theory*, volume 49, pages 605–622, 2016.
- N. Cesa-Bianchi, C. Gentile, G. Lugosi, and G. Neu. Boltzmann exploration done right. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, 2017.
- O. Chapelle. Modeling Delayed Feedback in Display Advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 1097–1105, 2014.
- O. Chapelle and L. Li. An Empirical Evaluation of Thompson Sampling. In *Advances in Neural Information Processing Systems*, volume 24, 2011.
- Y. J. Chen, L. Yang, X. Wang, X. Liu, M. Hajiesmaili, J. C. S. Lui, and D. Towsley. On-Demand Communication for Asynchronous Multi-Agent Bandits. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, 2023.
- P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 4302–4310, 2017.
- F. R. K. Chung. Diameters and Eigenvalues. *Journal of the American Mathematical Society*, 1989.

- F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.
- D. L. Cohn. *Measure Theory*. Birkhäuser/Springer, New York, 2013.
- D. R. Cox. *Analysis of Survival Data*. Chapman and Hall, 1984.
- V. Dani, T. P. Hayes, and S. M. Kakade. Stochastic Linear Optimization under Bandit Feedback. In *Proceedings of the 21st Annual Conference on Learning Theory*, pages 355–366, 2008.
- C. Dann and E. Brunskill. Sample Complexity of Episodic Fixed-Horizon Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 28, 2015.
- C. Dann, T. Lattimore, and E. Brunskill. Unifying PAC and Regret: Uniform PAC Bounds for Episodic Reinforcement Learning, 2018.
- C. Dann, L. Li, W. Wei, and E. Brunskill. Policy Certificates: Towards Accountable Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 1507–1516, 2019.
- P. Dayan and T. J. Sejnowski. TD(λ) Converges with Probability 1. *Machine Learning*, 1994.
- V. H. de la Peña, M. J. Klass, and T. L. Lai. Self-normalized processes: exponential inequalities, moment bounds and iterated logarithm laws. *The Annals of Probability*, 32(3):1902 – 1933, 2004.
- R. Degenne and V. Perchet. Anytime optimal algorithms in stochastic multi-armed bandits. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48, pages 1587–1595, 2016.
- T. Desautels, A. Krause, and J. W. Burdick. Parallelizing Exploration-Exploitation Tradeoffs in Gaussian Process Bandit Optimization. *Journal of Machine Learning Research*, 15(119):4053–4103, 2014.
- O. D. Domingues, P. Ménard, E. Kaufmann, and M. Valko. Episodic Reinforcement Learning in Finite MDPs: Minimax Lower Bounds Revisited. In *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pages 578–598, 2021.
- A. Dubey and A. Pentland. Cooperative Multi-Agent Bandits with Heavy Tails. In *Proceedings of the 37th International Conference on Machine Learning*, 2020a.

- A. Dubey and A. Pentland. Private and Byzantine-Proof Cooperative Decision-Making. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, page 357–365, 2020b.
- J. C. Duchi, A. Agarwal, and M. J. Wainwright. Dual Averaging for Distributed Optimization: Convergence Analysis and Network Scaling. *IEEE Transactions on Automatic Control*, 2012.
- M. Dudík, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang. Efficient Optimal Learning for Contextual Bandits. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, UAI’11, page 169–178, 2011.
- M. Dudík, J. Langford, and L. Li. Doubly Robust Policy Evaluation and Learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, 2011.
- B. Efron. Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7(1):1 – 26, 1979.
- S. G. Eick. The Two-Armed Bandit with Delayed Responses. *The Annals of Statistics*, 16(1): 254–264, 1988.
- M. Elkin. Distributed Exact Shortest Paths in Sublinear Time. *J. ACM*, 2020.
- E. Esposito, S. Masoudian, H. Qiu, D. Van Der Hoeven, N. Cesa-Bianchi, and Y. Seldin. Delayed Bandits: When Do Intermediate Observations Help? In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 9374–9395, 2023.
- L. Faury, M. Abeille, C. Calauzenes, and O. Fercoq. Improved Optimistic Algorithms for Logistic Bandits. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 3052–3060, 2020.
- L. Faury, M. Abeille, K. Jun, and C. Calauzenes. Jointly Efficient and Optimal Algorithms for Logistic Bandits. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151, pages 546–580, 2022.
- S. Filippi, O. Cappe, and A. Garivier. Optimism in reinforcement learning and Kullback-Leibler divergence. *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2010a.

- S. Filippi, O. Cappé, A. Garivier, and C. Szepesvári. Parametric Bandits: The Generalized Linear Case. In *Advances in Neural Information Processing Systems*, volume 23, 2010b.
- H. Flynn, D. Reeb, M. Kandemir, and J. Peters. Improved Algorithms for Stochastic Linear Bandits Using Tail Bounds for Martingale Mixtures. In *Proceedings of the 37th Annual Conference on Neural Information Processing Systems*, 2023.
- W. Fokkink. *Distributed Algorithms: An Intuitive Approach*. The MIT Press, 2013.
- L. Ford. *Network Flow Theory*. RAND Corporation, 1956.
- R. Fruit, M. Pirotta, and A. Lazaric. Improved Analysis of UCRL2 with Empirical Bernstein Inequality, 2020.
- G. Gabbianelli, G. Neu, and M. Papini. Importance-Weighted Offline Learning Done Right. In *Proceedings of The 35th International Conference on Algorithmic Learning Theory*, volume 237 of *Proceedings of Machine Learning Research*, pages 614–634, 2024.
- Z. Gao, Y. Han, Z. Ren, and Z. Zhou. Batched multi-armed bandits problem. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.
- A. Garivier and O. Cappé. The KL-UCB Algorithm for Bounded Stochastic Bandits and Beyond. In Sham M. Kakade and Ulrike von Luxburg, editors, *Proceedings of the 24th Annual Conference on Learning Theory*, volume 19, pages 359–376, 2011.
- G. Garrigos and R. M. Gower. *Handbook of Convergence Theorems for (Stochastic) Gradient Methods*, 2024.
- A. Gelman and A. Vehtari. What are the most important statistical ideas of the past 50 years?, 2021.
- A. Gelman, J.B. Carlin, H.S. Stern, D.B. Dunson, A. Vehtari, and D.B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis, 2013.
- O. Granmo. Solving two-armed Bernoulli bandit problems using a Bayesian learning automaton. *International Journal of Intelligent Computing and Cybernetics*, 3(2):207–234, 2010.
- W. Hoeffding. Probability Inequalities for Sums of Bounded Random Variables. *Journal of the American Statistical Association*, 58:13–30, 1963.
- R. A. Horn and C. R. Johnson. *Matrix Analysis: Second Edition*. Cambridge University Press, 2013.

- D. G. Horvitz and D. J. Thompson. A Generalization of Sampling Without Replacement From a Finite Universe. *Journal of the American Statistical Association*, 1952.
- S. Hossain, E. Micha, and N. Shah. Fair Algorithms for Multi-Agent Multi-Armed Bandits. In *Advances in Neural Information Processing Systems*, 2021.
- B. Howson, C. Pike-Burke, and S. Filippi. Optimism and Delays in Episodic Reinforcement Learning. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, volume 206, pages 6061–6094, 2023a.
- B. Howson, C. Pike-Burke, and S. Filippi. Delayed Feedback in Generalised Linear Bandits Revisited. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, volume 206, pages 6095–6119, 2023b.
- B. Howson, S. L. Filippi, and C. Pike-Burke. QuACK: A Multipurpose Queuing Algorithm for Cooperative k -Armed Bandits. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025.
- D. Hsu, S. Kakade, and T. Zhang. A Tail Inequality for Quadratic Forms of Subgaussian Random Vectors. *Electronic Communications in Probability*, 2012.
- T. Jaakkola, M. I. Jordan, and S. P. Singh. On the Convergence of Stochastic Iterative Dynamic Programming Algorithms. *Neural Computation*, 1994.
- T. Jaksch, R. Ortner, and P. Auer. Near-optimal Regret Bounds for Reinforcement Learning. *Journal of Machine Learning Research*, 11(51):1563–1600, 2010.
- K. Jamieson, S. Katariya, A. Deshpande, and R. Nowak. Sparse Dueling Bandits. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, 2015.
- N. Jiang, A. Krishnamurthy, A. Agarwal, J. Langford, and R. E. Schapire. Contextual decision processes with low Bellman rank are PAC-learnable. In *Proceedings of the 34th International Conference on Machine Learning*, page 1704–1713, 2017.
- C. Jin, Z. Allen-Zhu, S. Bubeck, and M. I. Jordan. Is Q-learning Provably Efficient? In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 4868–4878, 2018.
- P. Joulani, Andras A. György, and C. Szepesvári. Online Learning under Delayed Feedback. In *Proceedings of the 30th International Conference on Machine Learning*, 2013.

- K. Jun, A. Bhargava, R. Nowak, and R. Willett. Scalable Generalized Linear Bandits: Online Computation and Hashing. In *Advances in Neural Information Processing Systems*, volume 30, 2017.
- K. Jun, L. Jain, B. Mason, and H. Nassif. Improved Confidence Bounds for the Linear Logistic Model and Applications to Bandits. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 5148–5157, 2021.
- S. M. Kakade. *On the Sample Complexity of Reinforcement Learning*. PhD thesis, University College London, 2003.
- M. N. Katehakis and H. Robbins. Sequential choice from several populations. *Proceedings of the National Academy of Sciences*, 92(19):8584–8585, 1995.
- K. V. Katsikopoulos and S. E. Engelbrecht. Markov decision processes with delays and asynchronous cost collection. *IEEE Transactions on Automatic Control*, 48(4):568–574, 2003.
- E. Kaufmann. On Bayesian Index Policies For Sequential Resource Allocation. *The Annals of Statistics*, 46(2):842–865, 2018.
- E. Kaufmann, N. Korda, and R. Munos. Thompson Sampling: An Asymptotically Optimal Finite-Time Analysis. In *Algorithmic Learning Theory*, pages 199–213, 2012.
- M. Kearns and S. Singh. Finite-Sample Convergence Rates for Q-Learning and Indirect Algorithms. In *Advances in Neural Information Processing Systems*, volume 11, 1998.
- M. Kearns, Y. Mansour, and A. Ng. Approximate Planning in Large POMDPs via Reusable Trajectories. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.
- A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with Mechanical Turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, page 453–456. Association for Computing Machinery, 2008.
- R. Kleinberg, A. Slivkins, and E. Upfal. Multi-Armed Bandits in Metric Spaces. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, page 681–690. Association for Computing Machinery, 2008.
- A. Klimentenko. Intransitivity in Theory and in the Real World. *Entropy*, 17(6):4364–4412, 2015.

- W. B. Knox, A. Allievi, H. Banzhaf, F. Schmitt, and P. Stone. Reward (Mis)design for autonomous driving. *Artificial Intelligence*, 316, 2023.
- T. Kocák, G. Neu, M. Valko, and R. Munos. Efficient learning by implicit exploration in bandit problems with side observations. In *Advances in Neural Information Processing Systems*, volume 27, 2014.
- N. L. Kuang, M. Yin, M. Wang, Y. Wang, and Y. Ma. Posterior Sampling with Delayed Feedback for Reinforcement Learning with Linear Function Approximation. In *Advances in Neural Information Processing Systems*, volume 36, 2023.
- B. Kveton, C. Szepesvári, S. Vaswani, Z. Wen, T. Lattimor, and M. Ghavamzadeh. Garbage In, Reward Out: Bootstrapping Exploration in Multi-Armed Bandits. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.
- T. L. Lai. Adaptive treatment allocation and the multi-armed bandit problem. *The Annals of Statistics*, 15(3):1091–1114, 1987.
- T. L. Lai and H. Robbins. Asymptotically Efficient Adaptive Allocation Rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.
- T. L. Lai and C. Z. Wei. Least Squares Estimates in Stochastic Regression Models with Applications to Identification and Control of Dynamic Systems. *The Annals of Statistics*, 10(1):154–166, 1982.
- A. Lalitha and A. Goldsmith. Bayesian Algorithms for Decentralized Stochastic Bandits. *IEEE Journal on Selected Areas in Information Theory*, 2021.
- G. Lan. Policy mirror descent for reinforcement learning: linear convergence, new sampling complexity, and generalized problem classes. *Mathematical Programming*, 198:1059–1106, 2022.
- T. Lucewicz, S. Segal, T. Koren, and Y. Mansour. Stochastic Multi-Armed Bandits with Unrestricted Delay Distributions. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 5969–5978, 2021.
- Tal Lucewicz, Aviv Rosenberg, and Yishay Mansour. Learning Adversarial Markov Decision Processes with Delayed Feedback. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):7281–7289, 2022.

- P. Landgren, V. Srivastava, and N. E. Leonard. Distributed Cooperative Decision-Making in Multi-Armed Bandits: Frequentist and Bayesian Algorithms. In *IEEE 55th Conference on Decision and Control*, 2016.
- J. Langford and T. Zhang. The Epoch-Greedy algorithm for contextual multi-armed bandits. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, page 817–824, 2007.
- T. Lattimore. Optimally Confident UCB: Improved Regret for Finite-Armed Bandits, 2016.
- T. Lattimore. A Scale Free Algorithm for Stochastic Bandits with Bounded Kurtosis. In *Advances in Neural Information Processing Systems*, volume 30, 2017.
- T. Lattimore. Refining the Confidence Level for Optimistic Bandit Strategies. *Journal of Machine Learning Research*, 19(20):1–32, 2018.
- T. Lattimore and M. Hutter. PAC Bounds for Discounted MDPs. In *Algorithmic Learning Theory*, pages 320–334, 2012.
- T. Lattimore and C. Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020.
- C. Li, I. Markov, M. de Rijke, and M. Zoghi. MergeDTS: A Method for Effective Large-Scale Online Ranker Evaluation. *ACM Transactions on Information Systems*, 38(4), 2020.
- L. Li, Y. Lu, and D. Zhou. Provably Optimal Algorithms for Generalized Linear Contextual Bandits. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2071–2080, 2017.
- F. Liese and I. Vajda. On Divergences and Informations in Statistics and Information Theory. *IEEE Transactions on Information Theory*, 52(10):4394–4412, 2006.
- J. Luo, C. Paduraru, O. Voicu, Y. Chervonyi, S. Munns, J. Li, C. Qian, P. Dutta, J. Q. Davis, N. Wu, X. Yang, C. Chang, T. Li, R. Rose, M. Fan, H. Nakhost, T. Liu, B. Kirkman, F. Altamura, L. Cline, P. Tonker, J. Gouker, D. Uden, W. Bryan, J. Law, D. Fatiha, N. Satra, J. Rothenberg, M. Waraich, M. Carlin, S. Tallapaka, S. Witherspoon, D. Parish, P. Dolan, C. Zhao, and D. J. Mankowitz. Controlling Commercial Cooling Systems Using Reinforcement Learning, 2022.
- U. Madhushani, A. Dubey, N. Leonard, and A. Pentland. One More Step Towards Reality: Cooperative Bandits with Imperfect Communication. In *Advances in Neural Information Processing Systems*, 2021.

- O. Maillard. *APPRENTISSAGE SÉQUENTIEL : Bandits, Statistique et Renforcement*. PhD thesis, Université des Sciences et Technologie de Lille - Lille I, 2011.
- O. Maillard, R. Munos, and G. Stoltz. A Finite-Time Analysis of Multi-armed Bandits Problems with Kullback-Leibler Divergences. In Sham M. Kakade and Ulrike von Luxburg, editors, *Proceedings of the 24th Annual Conference on Learning Theory*, volume 19, pages 497–514, 2011.
- T. Mandel, Y. Liu, E. Brunskill, and Z. Popović. The Queue Method: Handling Delay, Heuristics, Prior Data, and Evaluation in Bandits. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 2015.
- D. Martínez-Rubio, V. Kanade, and P. Rebeschini. Decentralized Cooperative Stochastic Bandits. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.
- B. C. May, N. Korda, A. Lee, and D. S. Leslie. Optimistic Bayesian Sampling in Contextual-Bandit Problems. *Journal of Machine Learning Research*, 13(67):2069–2106, 2012.
- P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Routledge, 1989.
- J. McInerney, B. Lackner, S. Hansen, K. Higley, H. Bouchard, A. Gruson, and R. Mehrotra. Explore, Exploit, and Explain: Personalizing Explainable Recommendations with Bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems*, page 31–39, 2018.
- J. Mei, C. Xiao, C. Szepesvári, and D. Schuurmans. On the Global Convergence Rates of Softmax Policy Gradient Methods. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 6820–6829, 2020.
- J. Mei, Z. Zhong, B. Dai, A. Agarwal, C. Szepesvári, and D. Schuurmans. Stochastic Gradient Succeeds for Bandits. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.
- P. Ménard and A. Garivier. A minimax and asymptotically optimal algorithm for stochastic bandits. In *Proceedings of the 28th International Conference on Algorithmic Learning Theory*, pages 223–237, 2017.
- P. Ménard, O. D. Domingues, A. Jonsson, E. Kaufmann, E. Leurent, and M. Valko. Fast active learning for pure exploration in reinforcement learning. In *Proceedings of the 38th International*

- Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 7599–7608, 2021.
- C. Mesterharm. On-line Learning with Delayed Label Feedback. In *Algorithmic Learning Theory*, pages 399–413, 2005.
- E. Moore. *The Shortest Path Through a Maze*. Bell Telephone System., 1959.
- R. Munos, M. Valko, D. Calandriello, M. G. Azar, M. Rowland, Z. D. Guo, Y. Tang, M. Geist, T. Mesnard, C. Fiegel, A. Michi, M. Selvi, S. Girgin, N. Momchev, O. Bachem, D. J. Mankowitz, D. Precup, and B. Piot. Nash Learning from Human Feedback. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.
- G. Neu. Explore no more: improved high-probability regret bounds for non-stochastic bandits. In *Proceedings of the 29th International Conference on Neural Information Processing Systems*, page 3168–3176, 2015.
- G. Neu and C. Pike-Burke. A Unifying View of Optimism in Episodic Reinforcement Learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020.
- OpenAI. Language Models are Unsupervised Multitask Learners, 2019. Technical Report.
- I. Osband and B. Van Roy. Why is Posterior Sampling Better than Optimism for Reinforcement Learning? In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2701–2710, 2017.
- I. Osband, D. Russo, and B. Van Roy. (More) Efficient Reinforcement Learning via Posterior Sampling. In *Advances in Neural Information Processing Systems*, volume 26, 2013.
- O. P. O’Sullivan. The Neural Basis of Always Looking on the Bright Side. *Dialogues in Philosophy, Mental and Neuro Sciences*, 8(1):11–15, 2015.
- V. Perchet, P. Rigollet, S. Chassang, and E. Snowberg. Batched Bandit Problems. In *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pages 1456–1456, 2015.
- C. Pike-Burke, S. Agrawal, C. Szepesvári, and S. Grünewälder. Bandits with Delayed, Aggregated Anonymous Feedback. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 4105–4113, 2018.

- M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.
- R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741, 2023.
- O. Raveh, J. Honda, and M. Sugiyama. Multi-Player Approaches for Dueling Bandits. Preprint, 2024.
- W. Ren, X. Zhou, J. Liu, and N. B. Shroff. Multi-Armed Bandits with Local Differential Privacy. Preprint, 2020.
- Z. Ren, Z. Zhou, and J. R. Kalagnanam. Batched Learning in Generalized Linear Contextual Bandits with General Decision Sets. In *2021 American Control Conference*, pages 4958–4963, 2021.
- H. Robbins. Some Aspects of the Sequential Design of Experiments. *Bulletin of the American Mathematical Society*, 58(5):527 – 535, 1952.
- P. Rusmevichientong and J. N. Tsitsiklis. Linearly Parameterized Bandits. *Mathematics of Operations Research*, 35(2):395–411, 2010.
- D. Russo and B. Van Roy. Learning to Optimize via Posterior Sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014.
- D. J. Russo, B. Van Roy, A. Kazerouni, I. Osband, and Z. Wen. A Tutorial on Thompson Sampling. *Foundations and Trends Machine Learning*, 11:1–96, 2018.
- A. Sankararaman, A. Ganesh, and S. Shakkottai. Social Learning in Multi Agent Multi Armed Bandits. *Proc. ACM Meas. Anal. Comput. Syst.*, 3(3), 2019.
- N. Santoro. *Design and Analysis of Distributed Algorithms*. John-Wiley & Sons, 2006.
- A. Sawarni, N. Das, S. Barman, and G. Sinha. Generalized Linear Bandits with Limited Adaptivity. In *The 38th Annual Conference on Neural Information Processing Systems*, 2024.
- S. L. Scott. A modern Bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6):639–658, 2010.
- D. Shah. Gossip Algorithms. *Foundations and Trends in Networking*, 3(1):1–125, 2009.

- S. Shahrampour, A. Rakhlin, and A. Jadbabaie. Multi-Armed Bandits in Multi-Agent Networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017.
- L. Shani, Y. Efroni, A. Rosenberg, and S. Mannor. Optimistic policy optimization with bandit feedback. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 8604–8613, 2020.
- T. Sharot. The optimism bias. *Current Biology*, 21(23):R941–R945, 2011.
- D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.
- R. Simon. Adaptive Treatment Assignment Methods and Clinical Trials. *Biometrics*, 33(4):743–749, 1977.
- J. Skalse, N. Howe, D. Krashennnikov, and D. Krueger. Defining and Characterizing Reward Gaming. In *Advances in Neural Information Processing Systems*, volume 35, pages 9460–9471, 2022.
- A. Slivkins. Introduction to Multi-Armed Bandits. *Foundations and Trends in Machine Learning*, 12(1-2):1–286, 2019.
- N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, page 1015–1022, 2010.
- G. Stoltz. *Incomplete information and internal regret in prediction of individual sequences*. PhD thesis, Université Paris Sud-Paris XI, 2005.
- A. L. Strehl and M. L. Littman. A theoretical analysis of Model-Based Interval Estimation. In *Proceedings of the 22nd International Conference on Machine Learning*, page 856–863, 2005.
- A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman. PAC model-free reinforcement learning. In *Proceedings of the 23rd International Conference on Machine Learning*, page 881–888, 2006.
- R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 1988.

- R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2020.
- A. Swaminathan and T. Joachims. Counterfactual Risk Minimization: Learning from Logged Bandit Feedback. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 814–823, 2015.
- I. Szita and C. Szepesvári. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *Proceedings of the 27th International Conference on Machine Learning*, page 1031–1038, 2010.
- M. S. Talebi and O. Maillard. Variance-Aware Regret Bounds for Undiscounted Reinforcement Learning in MDPs. In *Proceedings of Algorithmic Learning Theory*, volume 83, pages 770–805, 2018.
- C. Tekin, S. Zhang, and M. van der Schaar. Distributed Online Learning in Social Recommender Systems. *IEEE Journal of Selected Topics in Signal Processing*, 2014.
- W. 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.
- L. Tran-Thanh, A. Rogers, and N. Jennings. Long-Term Information Collection with Energy Harvesting Wireless Sensors: A Multi-Armed Bandit Based Approach. *Autonomous Agents and Multi-Agent Systems*, 2011.
- A. L. Trella, K. W. Zhang, H. Jajal, I. Nahum-Shani, V. Shetty, F. Doshi-Velez, and S. A. Murphy. A Deployed Online Reinforcement Learning Algorithm In An Oral Health Clinical Trial, 2024. URL <https://arxiv.org/abs/2409.02069>.
- A. Tversky. Intransitivity of preferences. *Psychological Review*, 1969.
- S. Vakili, D. Ahmed, A. Bernacchia, and C. Pike-Burke. Delayed feedback in kernel bandits. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.
- M. Valko, N. Korda, R. Munos, I. Flaounas, and N. Cristianini. Finite-time analysis of kernelised contextual bandits. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, page 654–663, 2013.
- C. Vernade, O. Cappé, and V. Perchet. Stochastic Bandit Models for Delayed Conversions. In *Conference on Uncertainty in Artificial Intelligence*, 2017.

- C. Vernade, A. Carpentier, T. Lattimore, G. Zappella, Beyza Ermis, and M. Brückner. Linear Bandits with Stochastic Delayed Feedback. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9712–9721, 2020.
- S. S. Villar, J. Bowden, and J. Wason. Multi-armed Bandit Models for the Optimal Design of Clinical Trials: Benefits and Challenges. *Statistical Science*, 30(2):199 – 215, 2015.
- M. J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019.
- L. Wang, A. Krishnamurthy, and A. Slivkins. Oracle-Efficient Pessimism: Offline Policy Optimization In Contextual Bandits. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2024.
- P. Wang, A. Proutiere, K. Ariu, Y. Jedra, and A. Russo. Optimal Algorithms for Multiplayer Multi-Armed Bandits. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, 2020.
- C. Watkins. *Learning from Delayed Rewards*. PhD thesis, University of Cambridge, 1989.
- M. J. Weinberger and E. Ordentlich. On Delayed Prediction of Individual Sequences. *IEEE Transactions on Information Theory*, 48(7):1959–1976, 2002.
- H. Wu and X. Liu. Double Thompson Sampling for Dueling Bandits. In *Advances in Neural Information Processing Systems*, 2016.
- J. Wyatt. *Exploration and Inference in Learning from Reinforcement*. PhD thesis, University of Edinburgh, 1997.
- L. Xiao and S. Boyd. Fast Linear Iterations for Distributed Averaging. *Systems & Control Letters*, 2004.
- T. Xie, D. J Foster, A. Krishnamurthy, C. Rosset, A. H. Awadallah, and A. Rakhlin. Exploratory Preference Optimization: Harnessing Implicit Q^* -Approximation for Sample-Efficient RLHF. In *The 13th International Conference on Learning Representations*, 2025.
- W. Xiong, H. Dong, C. Ye, Z. Wang, H. Zhong, H. Ji, N. Jiang, and T. Zhang. Iterative preference learning from human feedback: Bridging theory and practice for RLHF under KL-constraint. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 54715–54754, 2024.

- R. Xu, Y. Min, and T. Wang. Noise-Adaptive Thompson Sampling for Linear Contextual Bandits. In *Advances in Neural Information Processing Systems*, volume 36, pages 23630–23657, 2023.
- L. Yang, Y. J. Chen, S. Pasteris, M. Hajiesmaili, J. C. S. Lui, and D. Towsley. Cooperative Stochastic Bandits with Asynchronous Agents and Constrained Feedback. In *Advances in Neural Information Processing Systems*, 2021.
- Y. Yang, H. Zhong, T. Wu, B. Liu, L. Wang, and S. S. Du. A Reduction-based Framework for Sequential Decision Making with Delayed Feedback. In *Advances in Neural Information Processing Systems*, 2023.
- C. Ye, W. Xiong, Y. Zhang, H. Dong, N. Jiang, and T. Zhang. Online Iterative Reinforcement Learning from Human Feedback with General Preference Model. In *Advances in Neural Information Processing Systems*, volume 37, pages 81773–81807, 2024.
- Y. Yue, J. Broder, R. Kleinberg, and T. Joachims. The k -armed duelling bandits problem. In *Proceedings of the 22nd Conference on Learning Theory*, 2009.
- Yisong Yue and Thorsten Joachims. Beat the mean bandit. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, 2011.
- E. Zermelo. Die Berechnung der Turnier-Ergebnisse als ein Maximumproblem der Wahrscheinlichkeitsrechnung. *Mathematische Zeitschrift*, 1929.
- W. Zhan, M. Uehara, N. Kallus, J. D. Lee, and W. Sun. Provable Offline Preference-Based Reinforcement Learning. In *The 12th International Conference on Learning Representations*, 2024a.
- W. Zhan, M. Uehara, W. Sun, and J. D. Lee. Provable Reward-Agnostic Preference-Based Reinforcement Learning. In *The Twelfth International Conference on Learning Representations*, 2024b.
- J. S. Zhang, B. Howson, P. Savva, and E. Loh. DISCO: An End-to-End Bandit Framework for Personalised Discount Allocation. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track*, pages 33–49, 2024.
- K. W. Zhang, T. Baldwin-McDonald, K. Ciosek, L. Maystre, and D. Russo. Impatient Bandits: Optimizing for the Long-Term Without Delay, 2025.

- Z. Zhang, X. Ji, and S. Du. Is Reinforcement Learning More Difficult Than Bandits? A Near-optimal Algorithm Escaping the Curse of Horizon. In *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134, pages 4528–4531, 2021.
- Z. Zhou, R. Xu, and J. Blanchet. Learning in Generalized Linear Contextual Bandits with Stochastic Delays. In *Advances in Neural Information Processing Systems*, volume 32, 2019.
- B. Zhu, M. I. Jordan, and J. Jiao. Principled Reinforcement Learning with Human Feedback from Pairwise or K-wise Comparisons. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.
- D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-Tuning Language Models from Human Preferences, 2020.
- M. Zoghi, S. Whiteson, R. Munos, and M. Rijke. Relative Upper Confidence Bound for the K-Armed Dueling Bandit Problem. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.
- M. Zoghi, Z. Karnin, S. Whiteson, and M. de Rijke. Copeland Dueling Bandits. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, 2015a.
- M. Zoghi, S. Whiteson, and M. de Rijke. MergeRUCB: A Method for Large-Scale Online Ranker Evaluation. In *Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, WSDM '15, page 17–26. Association for Computing Machinery, 2015b.

Appendix A

Useful Results

A.1 Concentration of Measure

Lemma A.1 (Hoeffding's Inequality ([Hoeffding, 1963](#))). *Let $X_1, X_2, \dots, X_n \stackrel{iid}{\sim} P$ where each $X_t \in [0, 1]$ almost surely. Then, for any $\epsilon \geq 0$:*

$$\mathbb{P}(\hat{\mu}(n) - \mu \geq \epsilon) \leq \exp(-2n\epsilon^2)$$

$$\mathbb{P}(\mu - \hat{\mu}(n) \geq \epsilon) \leq \exp(-2n\epsilon^2)$$

where $\hat{\mu}(n) = \frac{1}{n} \sum_{t=1}^n X_t$ denotes the empirical mean computed with n samples.

Lemma A.2 (Bernstein's Inequality ([Cesa-Bianchi and Lugosi, 2006](#))). *Let $\{X_t\}_{t=1}^n$ be a sequence of independent and identically distributed σ_t -subgaussian random variables. Define $S_n = X_1 + X_2 + \dots + X_n$. Then,*

$$\mathbb{P}\left(S_n - \mathbb{E}[S_n] \geq \frac{2}{3} \log\left(\frac{1}{\delta}\right) + 2\sqrt{\mathbb{V}[S_n] \log\left(\frac{1}{\delta}\right)}\right) \leq \delta$$

Appendix B

Additional Experimental Results

B.1 Chapter 5

Here, we present additional experimental results for the chain environments with $H = S \in \{5, 10, 20\}$ and $\mathbb{E}[\tau] \in \{100, 300, 500\}$. In all combinations of chain length and expected delay, our updating procedures give better empirical performance, especially for the delay distributions with higher variances. For all expected delays, active updating gives the best performance. However, our experiments indicate that lazy updating with $\alpha = 10$ or 100 is comparable, as one would expect based on the intuition that it is an approximation to active updating that converges in the limit as $\alpha \rightarrow \infty$.

B.1.1 Chain Environment with $H = S = 5$

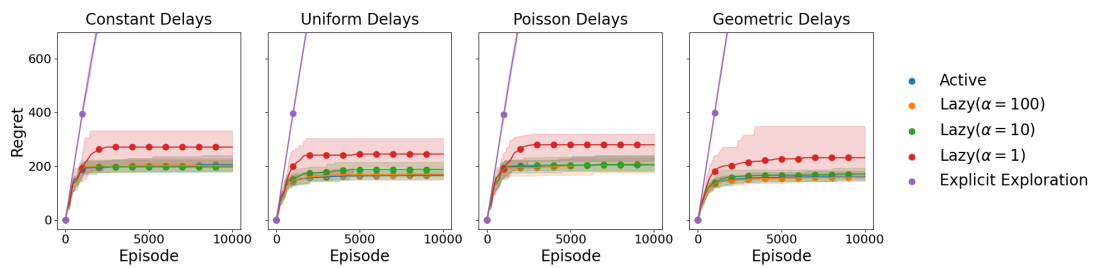
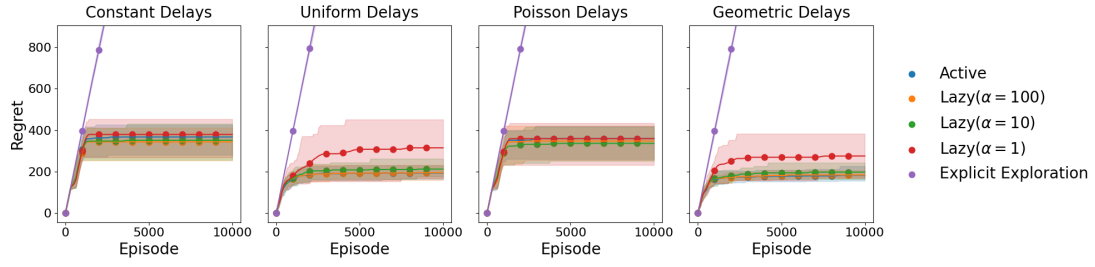
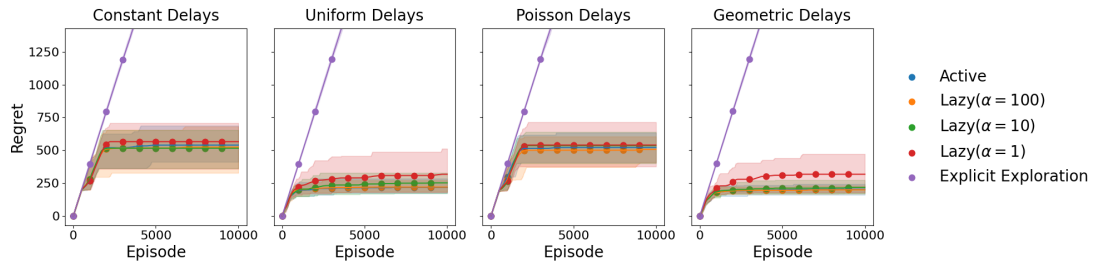
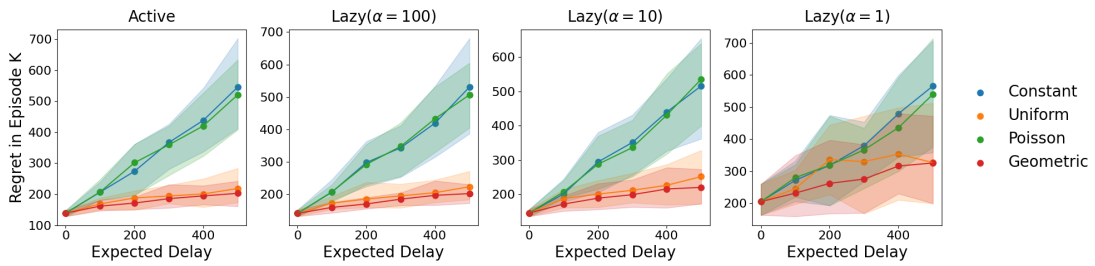
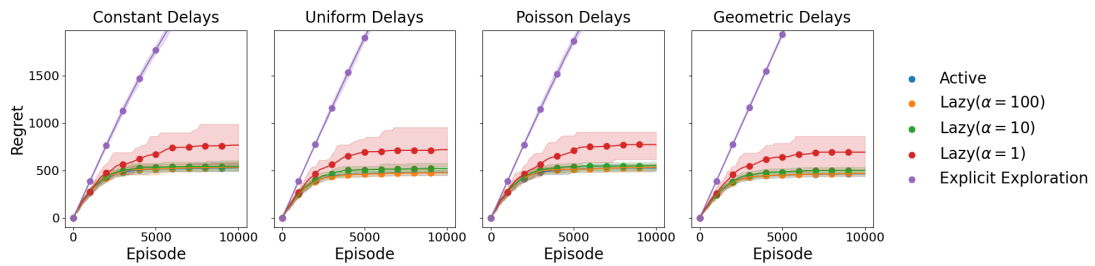
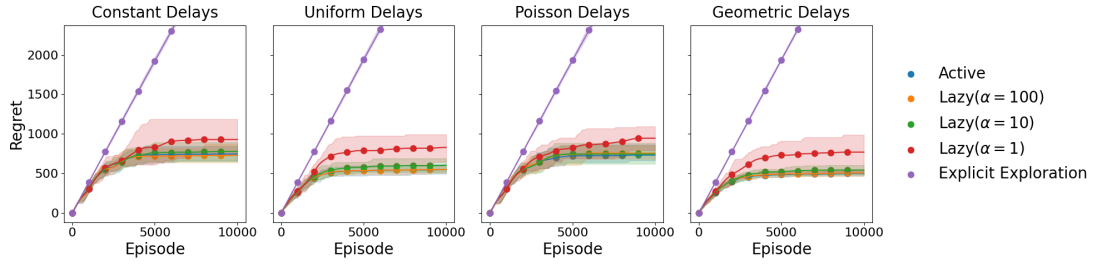
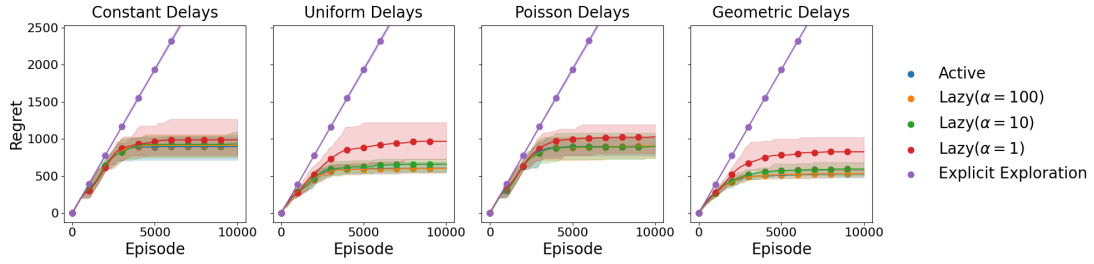
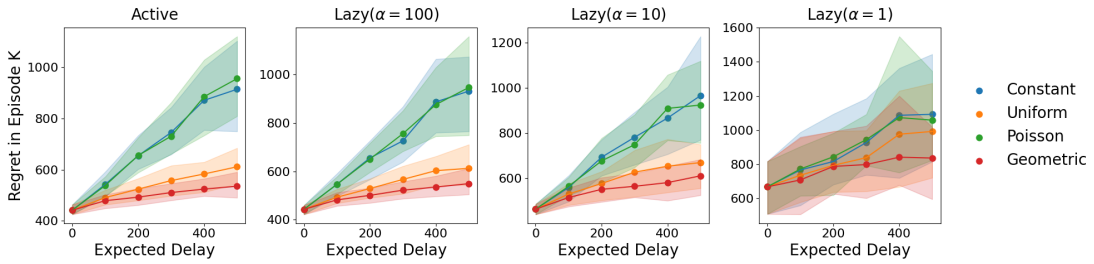


Figure B.1: Cumulative Regret ($S = 5, \mathbb{E}[\tau] = 100$).

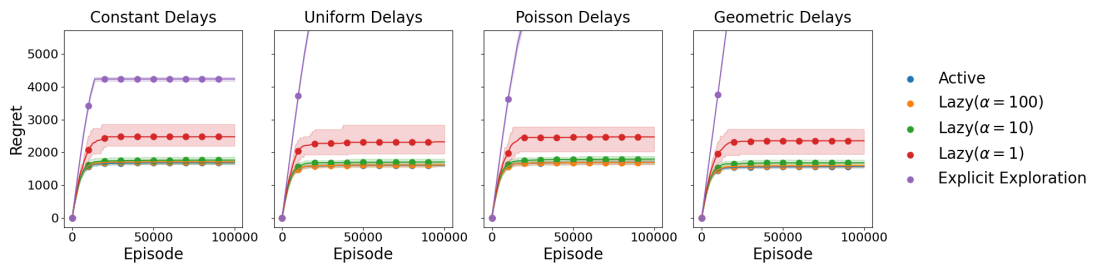
Figure B.2: Cumulative Regret ($S = 5, \mathbb{E}[\tau] = 300$).Figure B.3: Cumulative Regret ($S = 5, \mathbb{E}[\tau] = 500$).Figure B.4: Delay Dependence ($S = 5$)

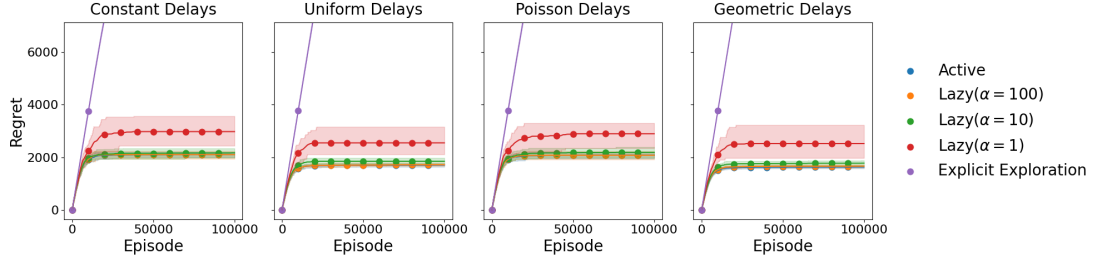
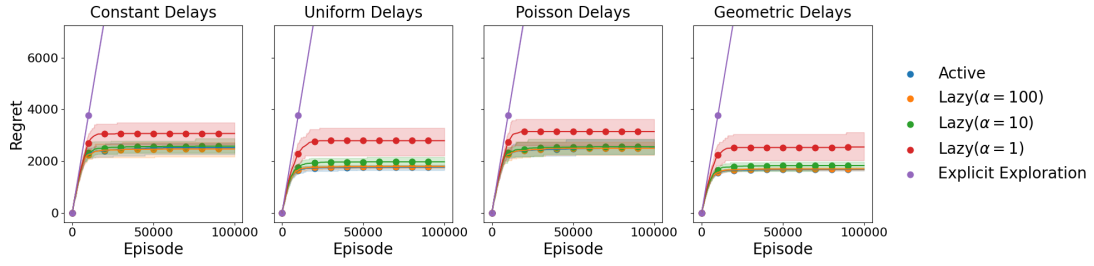
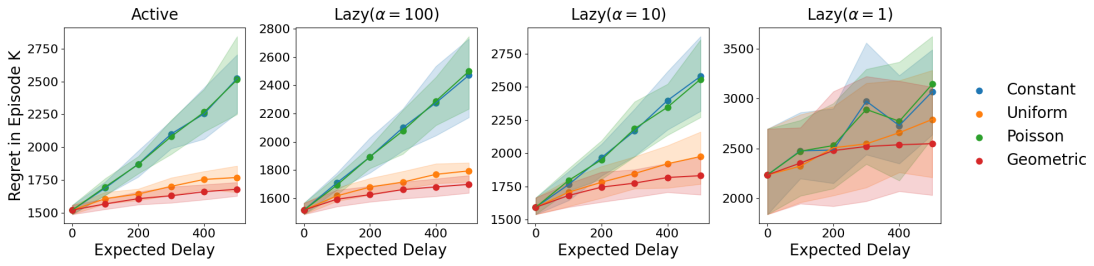
B.1.2 Chain Environment with $H = S = 10$

Figure B.5: Cumulative Regret ($S = 10, \mathbb{E}[\tau] = 100$).

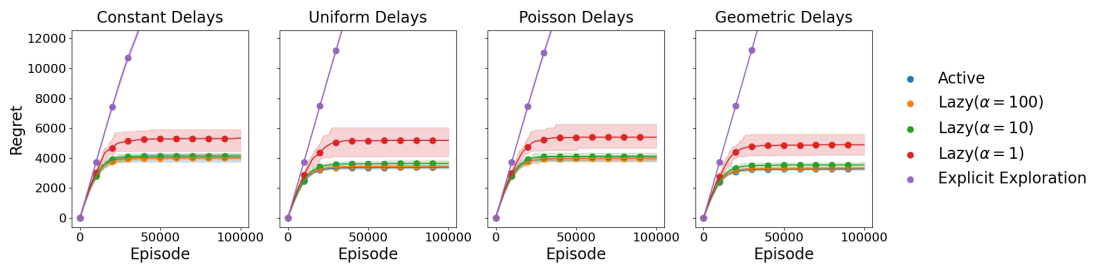
Figure B.6: Cumulative Regret ($S = 10, \mathbb{E}[\tau] = 300$).Figure B.7: Cumulative Regret ($S = 10, \mathbb{E}[\tau] = 500$).Figure B.8: Delay Dependence ($S = 10$)

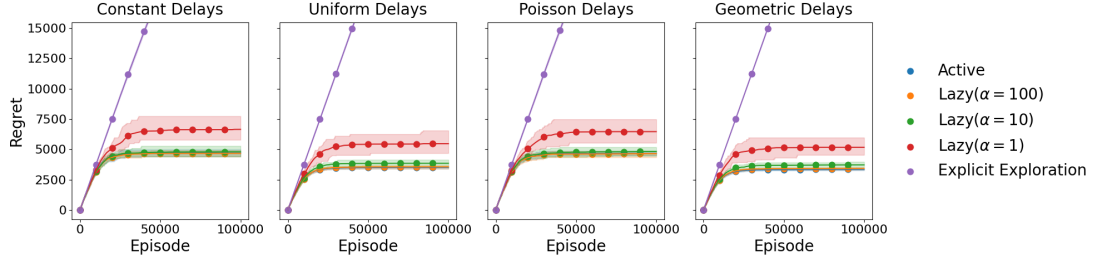
B.1.3 Chain Environment with $H = S = 20$

Figure B.9: Cumulative Regret ($S = 20, \mathbb{E}[\tau] = 100$).

Figure B.10: Cumulative Regret ($S = 20, \mathbb{E}[\tau] = 300$).Figure B.11: Cumulative Regret ($S = 20, \mathbb{E}[\tau] = 500$).Figure B.12: Delay Dependence ($S = 20$)

B.1.4 Chain Environment with $H = S = 30$

Figure B.13: Cumulative Regret ($S = 30, \mathbb{E}[\tau] = 300$).

Figure B.14: Cumulative Regret ($S = 30, \mathbb{E}[\tau] = 500$).

B.2 Chapter 6

B.2.1 Cycle Graph

Figure B.15 displays the group regret for cycle graphs with varying numbers of agents. Here, QuACK-UCB outperforms the other UCB-based algorithms on all network sizes. Furthermore, QuACK-TS is competitive with DecTS, with the performance gap decreasing as the size of the network increases.

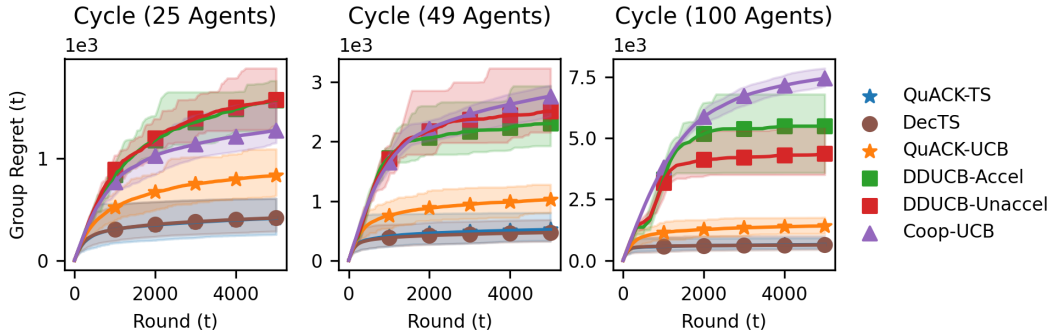


Figure B.15: The group regret of various cooperative multi-armed bandit algorithms on cycle graphs of varying sizes.

B.2.2 Grid Graph.

Figure B.16 displays the group regret for grid graphs with varying numbers of agents. Here, QuACK-UCB outperforms the other UCB-based algorithms on all network sizes. Furthermore, QuACK-TS is competitive with DecTS.

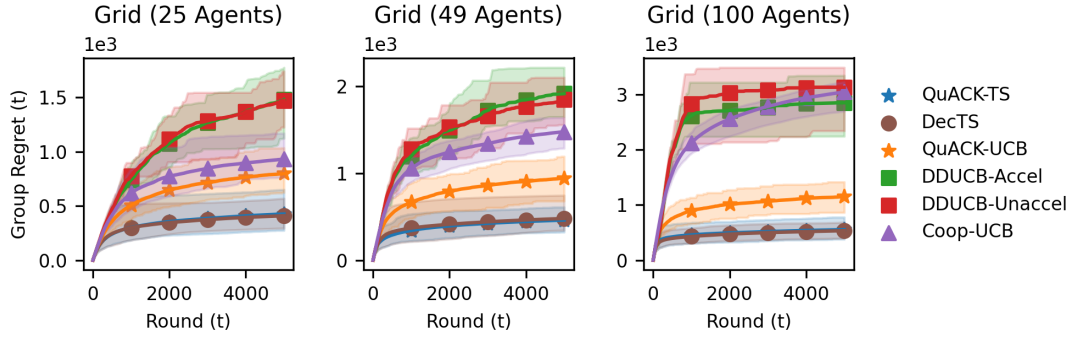


Figure B.16: The group regret of various cooperative multi-armed bandit algorithms on grid graphs of varying sizes.

B.2.3 Star Graph.

The star graph is a common sub-structure that appears in social networks. Figure B.16 displays the group regret for star graphs with varying numbers of agents. Here, QuACK-UCB outperforms the other UCB-based algorithms on all network sizes. Furthermore, QuACK-TS is competitive with DecTS on the smallest star network. However, QuACK-TS achieves smaller group regret for larger star networks.

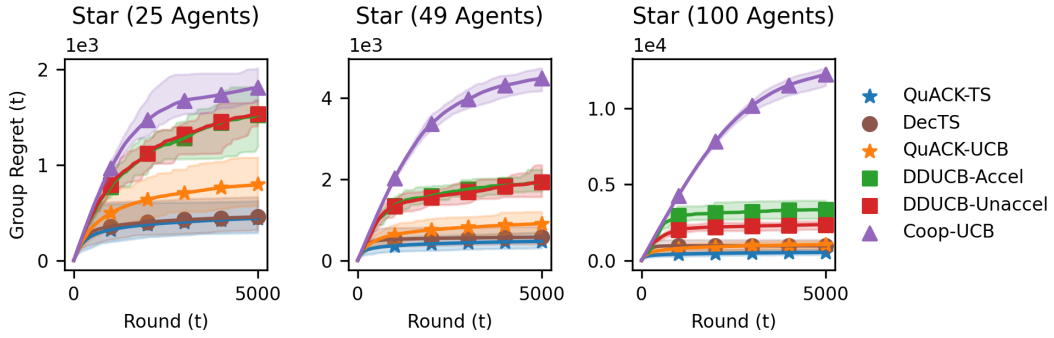


Figure B.17: The group regret of various cooperative multi-armed bandit algorithms on star graphs of varying sizes.