UNIVERSITÀ DEGLI STUDI DI MILANO

PH.D. PROGRAM IN COMPUTER SCIENCE

(XXXVI CYCLE)

DEPARTMENT OF COMPUTER SCIENCE
"GIOVANNI DEGLI ANTONI"

A thesis submitted for the degree of

Doctor of Philosophy

Non-stationary multiarmed bandits for satiation and seasonality phenomena in music recommender systems

Subject Area: INF/01

Author:
Giulia Clerici

Supervisor:
Nicolò Cesa-Bianchi

Co-supervisor:
Pierre Laforgue

PhD Coordinator:
Roberto Sassi

Academic Year 2022-2023

"*Mi spiegò che a lei non servisse riuscire, per essere felice: il fatto stesso di fare delle esperienze anche diverse tra loro e non tutte finalizzate a un risultato le dava delle sensazioni. Belle o brutte che fossero, le riconosceva come segnali di vitalità. Questo le permetteva di sperimentarsi, vedere che effetto le facesse, scremare cosa fosse adatto a lei e cosa no. Le dava l'idea di stare scegliendo dal quasi sconfinato menu delle offerte della vita. Soffriva, molto e in solitudine - cosa che le faceva male ma non paura. In qualche modo, anche nel dolore avvertiva una dolcezza, un esistere. Prendere contatto con la crisi era per Ambra un passaggio necessario, un rilevatore di autenticità affaticante ma prezioso, che le rappresentava un'occasione per incontrare se stessa e conoscersi nel suo provarci. Prima della riuscita, nonostante la riuscita, abdicando alla riuscita.*"

Stefania Andreoli

Abstract

This thesis delves into theoretical aspects of non-stationary multiarmed bandits, motivated by their application to music recommender systems. An intrinsic challenge of such systems lies in evolving user preferences. Rather than finding a single optimal item, the objective is to craft an ordered sequence of items aligning with the user's behaviour. Indeed, these systems need to address dynamic user preferences, characterized by phenomena like satiation. Our goal is to study these phenomena in a sequential learning setting characterized by partial feedback, adopting multiarmed bandits as the approach to tackle this problem. We first introduce a novel model with finitely many arms, which handles contrasting phenomena like satiation and seasonality in a unified framework. We formalize this model, called *Last Switch Dependent Bandit*, as a non-stationary multiarmed bandit where the expected reward of an arm is determined by its state, which indicates the time elapsed since the arm last took part in a switch of actions. This model can generalize to different types of non-stationarity as we relaxed many typical assumptions. Furthermore, it can recover state-dependent bandits in the literature. In this thesis, we will discuss this bandit problem and the solution proposed to solve it, which is based on upper confidence bounds and techniques derived from combinatorial semi-bandits. We conclude this work by providing an upper bound of the regret, to assess the performance of the proposed solution.
Aware of the limitations of finite action sets, which are not always representative of real-world applications, a new linear bandit model is proposed to handle non-stationary phenomena within an infinite set of actions, posing complex challenges for cross-arm dependencies. We formalize a model called *Linear Bandit with Memory*, where the expected reward of an action is influenced by the learner's past actions in a fixed-size window, called *memory matrix*. Specifically, the two parameters m and γ, the size of the window and the exponent respectively, characterizing this memory matrix can produce two types of behaviours: rotting and rising. In our discussion, we show how our model generalizes stationary linear bandits, as well as partially recovering rested rotting and rising bandits in the limit $m \to +\infty$. We study the complex problem of modelling the interactions among arms in a linear non-stationary setting and propose a solution to solve it. Furthermore, we study the setting where m and γ are unknown and propose a meta-bandit algorithm for model selection to jointly learn the parameters and solve the bandit problem. For both models, our contributions consist in defining novel multiarmed bandit problems, proving sound theoretical guarantees, and discussing our contributions with respect to the literature.
Additionally, within the context of music streaming services, we present an additional line of research exploring the Spotify™ music credits network. We represent this music credits network as a directed graph and study the relationship between music genres and graph-related metrics. After observing interesting patterns on several centrality measures conditioned on music genre, we introduce a novel node-wise index of reciprocity as a potential index for informed recommendations.

Statement of Contribution

The content of Chapter 4 is based on the published paper: P. Laforgue, G. Clerici, N. Cesa-Bianchi, Ran Gilad-Bachrach. "*A Last Switch Dependent Analysis of Satiation and Seasonality in Bandits*". Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS). PMLR 151:971-990, 2022 ([Laforgue et al., 2022]).
Chapter 5 is based on the manuscript: G. Clerici, P. Laforgue, N. Cesa-Bianchi. "*Linear Bandits with Memory*", currently under review ([Clerici et al., 2023]).
Chapter 6 is based on the published paper: G. Clerici, M. Tiraboschi. "*Citation is not Collaboration: Music-Genre Dependence of Graph-related Metrics in a Music Credits Network*". Proceedings of the Sound and Music Computing Conference 2023. ISBN 978-91-527-7372-7 ([Clerici* and Tiraboschi*, 2023.]).

Acknowledgements

I would like to thank my supervisor, Nicolò Cesa-Bianchi, and co-supervisor, Pierre Laforgue, for everything they taught me during the last three years. I would also like to thank the reviewers for kindly agreeing to review this thesis and for their helpful feedback.

Thank you to all the people who are, or were, part of the LAILA lab and everyone I met during these years. I would also like to thank the LIM lab for their continued support during all my academic years. You have always shown me unmatched encouragement and uplifting support, especially when I needed it the most.

I would also like to thank Women In Machine Learning (WiML) for the amazing opportunities they offered me. I could have only dreamt about taking part and organizing these events on the other side of the world to bring women in ML together. I cannot describe the joy and fulfilment it gave me to help grow this community.

But my biggest thank you goes to my family and friends. Only you know how hard and challenging these years have been. Thank you for all your love and support, and for listening to all my complaints. I cannot express how grateful I am for all the kindness you have shown me during the hardest moments. Thank you for the laughs and happy moments we shared during our family Sunday lunches, improvised dinners with friends, last-moment sushi nights, and friend trips. These moments were an extremely important source of joy and peace during the hardships of the last few years. It sounds obvious to say, but thank you for loving me for who I am, and not for what I do or for the results I achieve. I love you immensely.

Lastly, I would like to thank Matteo. You have seen all the ups and downs of the past three years, more than anyone else. You might be the only one who truly understood how tough it has been during certain times. During this period, and much before that, you have always shown me an incredible amount of patience, support, and love. Whenever I felt everything was wrong and my self-esteem was so low, you were always there to show me how much I had accomplished and to remind me I could achieve everything I wanted. You always know how to make everything right. Every day I spend with you is the most precious gift.

The hardships of the past few years have taught me a lot about myself. But I think the greatest lesson I learned, more than any bandits theorem, is that we're all humans, we're all fallible, and we're not our jobs. It taught me the importance of taking care of yourself before everything else, that life is full of possibilities if you're willing to grasp them and demand them, and that perfection is the ugliest lie we tell ourselves. I cheer for all the beautiful mistakes I'll make in the future, with optimism in the face of uncertainty.

Contents

List of Figures

Notation

Bandits notation

$\mathbb{N}$	set of natural numbers $\mathbb{N} = \{0, 1, 2, \dots\}$
$\mathbb{N}^+$	set of positive natural numbers
$\mathbb{R}$	set of real numbers
$\mathbb{Z}$	set of integer numbers
$\mathbb{Z}^-$	set of negative integer numbers $\mathbb{Z}^- = \{-1, -2, \dots\}$
B^*	a set of finite sequences over B, $B^* = \cup_{i=0}^{\infty} B_i$
d	dimension
T	temporal horizon of the sequential game, $T \in \mathbb{N}$
t	index for time steps $t = 1, \dots, T$
$\mathcal{A}$	set of actions, also called arms
K	cardinality of the set of actions, when the setting considers a finite set of actions
a	a general action, also called arm
$a^{(i)}$	action where the index $i = 1, \dots, K$ indicates the specific action in the set $\mathcal{A}$
a_t	action played at time step t
a^*	optimal fixed action
a_t^*	optimal action at time t
$P_{a^{(i)}}$	reward distribution associated with action $a^{(i)}$
p	set of rewards distributions associated to actions, $p = (P_a : a \in \mathcal{A})$
X_t	reward obtained by the learner at time t
X_i	for $i = 1, \dots, K$, reward obtained by the learner after playing arm a_i
$\mu_{a^{(i)}}$, μ_i	mean reward of action $a^{(i)}$
H_t	history of past actions and rewards $H_t = \{a_1, X_1, \dots, a_{t-1}, X_{t-1}\}$
R_T	regret over the horizon T
$\mathbb{E}$	expectation
π	a policy, also referred to as strategy
π^*	optimal policy, also called OPT
v	vector of actions, the size is specified in the context
Y_t	sum of rewards collected at time t by a combinatorial bandit
$\boldsymbol{a}$	a general super-arm in a combinatorial bandit

$\boldsymbol{a}(i)$	the i-th element of a super-arm in a combinatorial bandit

Bachmann-Landau notation

Given the functions $f, g : \mathbb{N} \to [0, \infty)$:

$$
\begin{aligned}
f(T) = \mathcal{O}(g(T)) &\leftrightarrow \lim_{T\to\infty} \sup \frac{f(T)}{g(T)} < \infty \\
f(T) = o(g(T)) &\leftrightarrow \lim_{T\to\infty} \frac{f(T)}{g(T)} = 0 \\
f(T) = \Omega(g(T)) &\leftrightarrow \lim_{T\to\infty} \inf \frac{f(T)}{g(T)} > 0 \\
f(T) = \omega(g(T)) &\leftrightarrow \lim_{T\to\infty} \inf \frac{f(T)}{g(T)} = \infty \\
f(T) = \Theta(g(T)) &\leftrightarrow f(T) = \mathcal{O}(g(T)) \text{ and } f(T) = \Omega(g(T))
\end{aligned}
$$

Chapter 1

Introduction

This thesis aims at studying theoretical models of multiarmed bandits to address non-stationary phenomena in music recommender systems. In particular, we are inspired by the research problems related to the non-stationarity of users' preferences towards songs in music recommender systems and chose multiarmed bandits as a method to tackle this problem. Although we mention this application, we clarify that our research on non-stationary multiarmed bandits for recommender systems is purely theoretical and can be applied to a variety of applications. Before diving into the details of our work, we introduce the setting and some motivations useful to understand the following chapters in this manuscript.

1.1 Motivation

Since the Nineties, the World Wide Web has witnessed a remarkable expansion, moving from its origins in scientific and academic institutions to not only companies but also everyday individuals, thus becoming accessible to an ever-growing number of people. This medium's popularity grew rapidly and became an outlet for business activities and e-commerce, showing a rapid growth of online retail platforms. The number of available products and services exploded and users started to face the challenge of finding relevant items from large catalogues. Traditional search engines provided a means to navigate the web, but they were not well-suited for personalized recommendations. In the meantime, these circumstances led to the observation of the so-called long tail phenomenon [Rajaraman and Ullman, 2011]. This phenomenon describes a distribution of the popularity of items in e-commerce, where there is a small number of items whose popularity is very high and a large quantity of products whose number of sold items is very low. This is one of the situations where an e-commerce manager could exploit the benefits of a recommender system. Indeed, these systems could provide suggestions tailored to the individual preferences of specific users who could appreciate those low-popularity items. Overall, since an online platform could gather thousands or millions of items, a much larger number with respect to catalogues in a physical store, it is necessary to adopt a technology able to direct the user towards certain sets of items of their interest. From the provider's perspective, recommender systems are a great tool to compensate for the possibility that the user is overwhelmed by the plethora of possible choices and leaves the platform. Thus, the emergence of recommender systems can be attributed to the recognition that leveraging user's preferences could greatly enhance the online shopping experience. Providing a personalized service can be critical, especially in a world characterized by an exponentially increasing data flood [Rydning et al., 2018]. For this reason, recommender systems are employed in a broad range of applications, spanning various industries and sectors from e-commerce, and online advertisement to entertainment. They can be a valuable resource for e-commerce by exploiting the users' browsing history, purchase behaviour, and preferences to enhance the shopping experience and increase sales. In streaming services, they can provide personalized recommendations for movies, TV shows, songs, podcasts, and other multimedia content based on the user's preferences, ratings, and viewing history. They can also be used for news and content aggregation, suggesting relevant articles, news stories, blogs, and other material based on user interests, and reading habits; as well as for personalized advertising, by targeting users with relevant advertisements and promotions based on their demographics, browsing behaviour, and purchase history. Other industries where these systems can be useful include social media platforms, job portals, financial services, gaming, food and restaurant delivery, personalized medicine, healthcare interventions, and many others.

Recommender systems are essentially responsible for providing suggestions to a user on an online platform. In such platforms, the system provides a catalogue containing a vast amount of products, called items, from which the user can select one to buy or enjoy, whether they are an e-commerce or an entertainment platform. After consuming the product, they can provide their level of satisfaction by rating the item, usually with a numeric value on a certain scale. Using the user's past ratings and clicks, and sometimes combining them with the knowledge gained through other users and the features associated with the items, the system learns the user's taste and recommends items they most likely would enjoy. The fundamental goal here is to find the best product, or the best set of products, for a particular user. When a user joins a platform for the first time, the system has no information concerning that person's taste and preferences and has to deal with the problem of being able to suggest an item immediately, without having a history of their preferences. This is called the cold-start problem. In entertainment platforms, this is usually solved by presenting the first-time user with a list of items and asking them to explicitly rate some items or select the products they prefer, in order to collect some information about their taste. For example, when joining a movie streaming service, the user may be presented with a list of movies and asked to pick their favourites. By doing so, the system can collect some initial information about the user's preferences. However, this is not always possible depending on the type of platform. This means that the system is required to process data sequentially, gaining more information with each interaction between the user and the platform. It is from these types of settings, where the intelligent system needs to learn patterns given a sequential stream of data, that online machine learning finds some of its most convincing applications. Online learning is the

most suited class of methods to deal with environments where the learner is not given a dataset but needs to learn sequentially with each data sample presented one at a time.
In the most simple settings, the user's preferences remain unchanged. However, realistic scenarios are characterized by the fact that the user's predilection towards an item may change over time depending on the history of the user's fruition of items. This is the case, for example, of music streaming platforms. For instance, after a user listens to several jazz songs, he might get bored of listening to the same genre and would want to move to some rock songs. When we observe this pattern, we notice that the user's taste is not fixed, but the level of satisfaction with a product changes with respect to the items he previously experienced. Furthermore, another important property of these platforms is the fact that the system observes an explicit rating only for those items that have been consumed by the user. In practical terms, it means that the system has no knowledge of the exact rating that the user would have assigned to other items if they had made a different choice at that particular time. This peculiarity, which seems reasonable and natural in practical applications of everyday life, is an important matter when choosing the techniques used to implement the recommender system and constitutes a major difficulty for the theoretical study of these models. As a result of the phenomena we mentioned, the demand for new cutting-edge algorithms has been rising recently.
While several techniques can be adopted to develop a recommendation engine, a relevant approach that has been prospering in the last few decades and that is the focus of this research is the multiarmed bandit model. We focus on partial feedback, meaning that the system receives feedback only for the specific item that has been chosen, and not for all the other items in the set if they were chosen instead. The multiarmed bandit model is purposely designed to deal with partial feedback settings in online learning. Moreover, this class of models proves to be appropriate when dealing with data that is combinatorially large and often incomplete, which is the case of recommendation systems that are characterized by vast online catalogues and by the impossibility of a user to express their preference for all available items. Combining these aspects, multiarmed bandits are best suited for this environment since they encourage a good balance between exploitation and exploration, which is fundamental in this type of setting. In the rest of this thesis, we will use the terminology "multiarmed bandit" and "bandit" interchangeably.

1.2 Multiarmed Bandits

Before introducing multiarmed bandits, it is appropriate to introduce the broader class of methods to which they belong, which is online learning. Statistical machine learning methods are usually given a dataset, which can be annotated or not, right from the beginning and use it to build a model able to predict labels of new never-seen-before data points. On the other hand, online learning is a sequential learning protocol, which means that it is characterized by the fact that the learner does not have access to the entire training dataset but accesses data in a sequential fashion, processing one example at a time step.

This framework unveils its practicality when dealing with environments characterized by sensor data, user-interaction data, and financial data, for example. The general framework of online learning consists of a sequential game with a temporal horizon T, which can be finite or infinite, known or unknown. Given an initial predictor $h_1 \in \mathcal{H}$, where $\mathcal{H}$ is a class of predictors, in each time step $t = 1, \dots, T$ the learner observes a data point (x_t, y_t), receives a reward $r(h_t(x_t), y_t)$, or a loss $l_t(h_t(x_t), y_t) = 1 - r(h_t(x_t), y_t)$, which for simplicity can be assumed to be bounded in $[0, 1]$. This reward is used to update the predictor $h_t \to h_{t+1}$. So given an algorithm A adopted by the learner, this protocol generates a sequence of models $h_1, \dots, h_t$ where we have a different predictor at each time step. Each one of these predictors differs from the previous one for the incremental knowledge gained as the game proceeds. To measure the performance of an algorithm, we introduce the regret at horizon T as the following quantity $R_T = \sum_{t=1}^{T} r_t(h_t^*) - \sum_{t=1}^{T} r_t(h_t)$, where $h_t^* = \text{argmax}_{h \in \mathcal{H}} \sum_{s=1}^{t} r_t(h_t)$. Essentially, the regret is defined as the difference between the cumulative rewards collected by the optimal policy and the total rewards collected by the learner over the horizon T. We use the notation h_t^* to indicate the best predictor at time t, the one that maximizes the cumulative rewards up to time t. The goal of the learner is to exhibit a sublinear regret such that $R_T = o(T)$, given that, if the rewards are bounded in $[0, 1]$, the regret cannot grow more than linearly. The reader should note how the optimal policy is not always known, which means that it is not always possible to measure the regret in practical applications. However, the regret remains an important measure to study and estimate the quality of the algorithm from a theoretical point of view. Among the different variants that online learning can take, the current work focuses on the special case of partial feedback settings, where the learner has access to a finite set of actions and the models are probability distributions over this set. We assume that at each time step the learner can play one action and only observe the reward for the chosen action and not for all the possible actions in the set. The family of algorithms that addresses this type of setting is called multiarmed bandits.

The bandit problem is a mathematical model depicting the situation where a subject is faced with the dilemma of making a decision in the face of uncertainty on the payoffs resulting from the available choices. Assuming the user makes some decisions and can observe the payoffs of his actions, he is faced with another dilemma: Is it better to exploit the action that returns the highest payoff? Or is it better to move to another one, that might return a higher reward in future steps? This tradeoff between exploitation and exploration, called *exploration-exploitation dilemma*, deeply characterizes the bandit problem.

The multiarmed bandit problem consists of a sequential game between a learner and an environment over the horizon T, which is a positive natural number representing the total number of rounds. In each round $t \in \{1, \dots, T\}$ the learner is faced with a set of available actions, also called arms, $A = \{a^{(1)}, \dots, a^{(K)}\}$, where each one is associated with a reward unknown to the learner. In each round t the learner must choose an action a_t and at the end of the round the reward $X_t \in \mathbb{R}$ associated with the chosen action is revealed. The objective of the learner is to maximize the reward collected over the horizon. The future is

unknown to the learner, which at time t can only base their choices on the information collected in the past history, defined as $H_{t-1} = \{a_1, X_1, \dots, a_{t-1}, X_{t-1}\}$. The decisions of the learner are determined by a strategy, also called policy, which is a mapping from histories to actions or probability distributions over actions in case of randomized strategies. The learner must decide what strategy to follow in order to maximize the cumulative reward. This policy is chosen from a set of policies, called competitor class. The performance of the learner is measured by the regret, which is the difference between the reward collected by the optimal policy in the competitor class and the expected reward collected by the policy chosen from the same class and played by the learner.
Different assumptions lead to different types of bandits. Stochastic bandits assume that the environment draws the rewards in response to each action from an unknown distribution specific to that action and independent from the reward distributions of other actions. Therefore, a stochastic bandit consists of a set of distributions $v = (P_a : a \in A)$ such that at time $t \in \{1, \dots, T\}$ the reward X_{a_t} of the chosen action a_t is sampled from the distribution P_{a_t}.
When these distributions remain unchanged in each round, we are talking of stationary stochastic bandits. However, there is another category of bandits that assumes that the reward distributions of the arms change over time. This set is called non-stationary stochastic bandits.
Furthermore, there is a different class of bandits for which the previous assumptions drop and the environment can secretly choose the rewards for each action after examining the code of the algorithm proposed by the learner. This model where the environment has complete control over the payoffs is called adversarial multiarmed bandit [Auer et al., 2002] since the environment acts as an adversary to the learner.
In the context of recommendation systems, additional information is usually presented in association with users and items. However, the models mentioned above do not use this kind of data. Ignoring this side information would not be advisable since the quality of the predictions can greatly benefit from this data. Hence, a particular class of bandits emerges from this opportunity and is called contextual multiarmed bandits ([Li et al., 2010], [Woodroofe, 1979]).
We will provide a more in-depth discussion on multiarmed bandits in section 2. This introduction serves the purpose of initiating the discussion on multiarmed bandits as the main topic of this thesis. The research contribution we will present is purely theoretical but is focused on the applications of these models in the setting of music recommender systems.
Although this is the setting we focus on, we specify how music recommender systems are not the only application for these models. Indeed, the multiarmed bandit models we formalize can be employed to address non-stationary in other settings like social media platforms, news and jobs portals, online educational platforms, financial services, gaming industries, personalized medicine, and healthcare interventions.

Problem Formulation. The high-level challenge of this thesis is to address non-stationary phenomena in music recommender systems by introducing novel models of multiarmed bandits able to generalize to different behaviours.

The research effort of this dissertation aims to address the following questions:

- Given the different bandit models in the literature, is it possible to formalize a multiarmed bandit model able to generalise to different forms of non-stationarity? If so, which theoretical guarantees can be proven?
- Is it possible to aim at the same objective in the linear setting, where the learner deals with an infinite set of actions? In this case, what are the guarantees on the regret that can be formalized?

1.3 The Spotify™ network as a directed graph

Besides the research on non-stationary multiarmed bandits, we also discuss a contribution to the research analysis of music credits networks in a music streaming platform. We base our work on the data provided by Spotify™, which is used to build a Spotify™ music credits network. This network is modeled as a directed graph where nodes represent artists and arcs collaborations between them. Using this data and its representation, we investigate the relationship between some graph-related metrics and music genre. Specifically, we are interested in the analysis of the Spotify™ graph to better understand the crediting patterns revealed in this network, with the objective of studying this data and seeing if there could be some exploitable information that can be used when providing recommendations in music streaming platforms.

Problem Formulation. This line of research is concerned with the analysis of several centrality measures on this directed graph and aims to find an explanation for certain phenomena observed in this analysis. While a previous work on the undirected graph proposed a model to interpret these phenomena, we showed that it is not able to illustrate the different types of collaborations between two artists.
Given these premises, the research question here is: Can we better motivate the centrality of certain music genres by looking at the directed graph of the credits network? Can we propose a new index to distinguish between two different crediting patterns such as citation and collaboration?

1.4 Contributions and Publications

The contributions presented in this manuscript are the following:

- We propose a new model for the class of non-stationary stochastic multiarmed bandits with finitely many arms aimed at addressing research problems that arise from music recommender systems. In particular, we introduce a bandit model able to generalise to different forms of non-stationarity in a single framework. We call this model the Last Switch Dependent Bandit, study this setting, and provide algorithmic guarantees on the regret.

- We contribute to the research of linear bandits by introducing a new model able to generalize to different behaviours of non-stationarity in the linear setting with an infinite set of actions. When presenting this work, we show how it can recover previous multiarmed bandit models from the literature, showing its ability to generalise to standard linear, rotting, and rising bandits. We call this model Linear Bandits with Memory.

- In our solution to Linear Bandits with Memory, we propose an algorithm based on the knowledge of two parameters. In order to overcome this limitation, we propose a bandit model selection algorithm to adapt our solution to the setting where the learner has no access to the true values of these parameters. We analyse the meta-bandit and provide guarantees on its regret.

- Lastly, we present a study of the Spotify™ music credits network, analysing the relationship between music-genre and graph-related metrics, specifically centrality measures. We introduce a node-wise index of reciprocity to better explain the patterns observed in our analysis.

The contributions of this thesis are presented in the following publications:

- P. Laforgue, G. Clerici, N. Cesa-Bianchi, Ran Gilad-Bachrach. "*A Last Switch Dependent Analysis of Satiation and Seasonality in Bandits*". Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS). PMLR 151:971-990, 2022 [Laforgue et al., 2022]

- G. Clerici, M. Tiraboschi. "*Citation is not Collaboration: Music-Genre Dependence of Graph-related Metrics in a Music Credits Network*". Proceedings of the Sound and Music Computing Conference 2023. ISBN 978-91-527-7372-7 [Clerici* and Tiraboschi*, 2023.]

and the following manuscript:

- G. Clerici, P. Laforgue, N. Cesa-Bianchi. "*Linear Bandits with Memory*". Currently under review. [Clerici et al., 2023].

1.5 Structure of the thesis

The thesis is structured into seven chapters and an appendix. Here, we briefly describe the content of each chapter.

- In Chapter 1, we introduce and motivate the work presented in this manuscript. We briefly present the topics we are discussing in the following chapters, and state the research objectives, the contributions, and the publications.

- Chapter 2 is focused on the basic theory of multiarmed bandits necessary to better understand the content of this dissertation. In this chapter, we introduce multiarmed bandits and present different environments, mentioning those that are going to be discussed in the rest of this thesis.

In particular, we discuss stochastic, non-stationary, combinatorial, linear, and contextual bandits. In this chapter, we will also introduce the mathematical notation used in this dissertation.

- In Chapter 3 we present the state of the art for what concerns the main topic of this thesis: non-stationary multiarmed bandits. We identify two main classes of non-stationarity, exogenous and endogenous, and we discuss the models in the literature belonging to these two classes.

- Chapter 4 presents the work concerned with the first contribution, the Last Switch Dependent Bandit model. We define the model, discuss the hardness and approximation results, along with the solution we provide and its theoretical guarantees. The chapter ends with some experiments to show the empirical results of our methods.

- Chapter 5 focuses on the second contribution, presenting the Linear Bandit with Memory model. We introduce the model and discuss the results obtained in the linear setting. We prove regret guarantees for our algorithm and show some experiments. In this chapter, we also discuss the bandit selection algorithm for Linear Bandit with Memory, illustrating the dynamics involved in the model selection process in the non-stationarity setting presented in the previous chapter. As for the previous contributions, we prove regret bounds and show empirical validation of our solution.

- Chapter 6 discusses the contribution concerning the Spotify™ music credits network, analysing several centrality measures on the directed graph and proposing a new node-wise index of reciprocity to better understand intrinsic phenomena observed in the analysis.

- Finally, in Chapter 7 we present the conclusions of this work and discuss future developments.

- In the Appendix A the reader is provided with all the proofs omitted in the main body of this thesis.

Chapter 2

The theory behind multiarmed bandits: classes of bandits and popular approaches to solve them

Contents

Introduced in 1933 by [Thompson, 1933] and then further studied in 1953 by [Bush and Mosteller, 1953], multiarmed bandits were first considered for applications in medical trials and animal learning. The initial goal was to provide a mathematical model to study the behaviour of mice when placed in a T-shaped maze. In particular, the concept of these experiments consisted of studying how these mice would face the dilemma of choosing between two directions where only one would lead to a reward. This type of study was extended to humans and their behaviour when facing a situation characterized by a similar dilemma. Specifically, further studies aimed to analyze the behaviour of a subject when confronted with a choice: continuing to select an option whose outcome is known or taking a risk and picking an uncertain alternative for which the outcome is unknown. This dilemma between a safe and known choice and an alternative and uncertain direction is called the *exploration-exploitation dilemma*. Multiarmed bandits illustrate this situation and started to gain popularity decades later. They have been extensively studied only in the last twenty years ([Lattimore and Szepesvári, 2020]), finding many practical applications in industries such as advert placement, network routing, resource allocation, dynamic pricing, and, of course, recommendation services.
The name of this class of models comes from the lever-operated slot machines where the player needs to pull an arm and a reward is revealed. This type

of machine is usually called *bandit*. This term has been adopted and changed to multiarmed bandit to refer to a variant of these machines with multiple arms to pull. One can specifically talk about K-armed bandits where K is the number of arms. For instance, in a bandit where the action set consists of only two possible actions, therefore $K = 2$, one can refer to the model as 2-armed bandits. However, when $K \geq 2$ it is common practice to simply call these models *multiarmed* bandits.
These slot machines embody the exploration-exploitation dilemma and are an example of an online learning application characterized by partial feedback. The exploration-exploitation dilemma in bandits can be summed up in a question: when a player needs to decide which action to take, is it better to exploit those actions that he knows are more remunerative, or is it better to explore new unknown actions that could potentially hold a higher reward? This question ponders if it is better to make a safe choice for which you know the outcome or to risk and select a new unknown action without knowing if it will improve or worsen your performance. This question is modelled by the multiarmed bandit problem.
The multiarmed bandit problem consists of a sequential game between a learner and an environment over a horizon $T \in \mathbb{N}$, which is a positive natural number representing the total number of rounds. In each time step $t = 1, \ldots, T$ the learner is faced with a set $A = \{a^{(1)}, \ldots, a^{(K)}\}$ of available actions, also called arms. When it is clear from the context, we use a to refer to a general action in $\mathcal{A}$. Each action is associated with a reward unknown to the learner. Depending on the type of environment, the set of actions can be finite or infinite. We start by presenting the simpler setting where the set of arms is finite and K is the cardinality of this set, $|\mathcal{A}| = K$. We will investigate bandits with infinite arms in Sections 2.3 to 2.5. In each round t the learner must select an action a_t to play. Once the arm is pulled, the environment reveals to the learner the reward X_t resulted from having played the chosen arm a_t at time step t. The environment is unknown to the learner, who only knows that the true environment belongs to a set called environment class. As well as the environment, the future is also unknown to the learner, which at time t can only base its choices on the information collected in the past history, defined as $H_{t-1} = \{a_1, X_1, \ldots, a_{t-1}, X_{t-1}\}$. The objective of the learner is to maximize the reward collected over the horizon. The learner must decide what strategy to follow in order to maximize the cumulative reward at time T, which is defined as the sum of the rewards collected by the learner over the entire game. The learner's strategy, also called policy and indicated by π, is a map from histories to actions and it is chosen from a set of policies called competitor class. The performance of the learner is measured by the regret, which is defined as the difference between the cumulative reward collected by the best policy in the competitor class and the expected reward collected by the learner's policy. One can also consider the competitor class as large as possible to include the best policy for all possible environments. In this case, the regret is measured against the so-called optimal policy, also referred to as OPT. The regret can be written

as follows:

$$R_T = \sum_{t=1}^{T} X_t^* - \sum_{t=1}^{T} X_t, \tag{2.1}$$

where we use X_t^* to indicate the reward observed by the optimal policy at time t. For the rest of this manuscript the asterisk symbol $*$ associated to some term will signify that the term refers to the optimal policy.
Different assumptions about the environment lead to different types of bandits. For the rest of this chapter, we are introducing different classes of bandit problems, each one characterized by different assumptions on the environment, on the action set, and the reward function. We will start by introducing the difference between a stochastic and an adversarial environment. Then, we will proceed by presenting the prominent class of linear and contextual bandits, which introduces some assumptions that better match practical scenarios. Then, we will move to the class of combinatorial bandits, which can be seen as a specific subgroup of linear bandits. Finally, we introduce non-stationary bandits, which is the family of bandits characterized by non-stationary environments. Due to their relevance with respect to the contributions of this thesis, a more detailed discussion on non-stationary bandits is postponed to Chapter 3, which will focus on this particular class.

2.1 Stochastic K-armed bandits

Stochastic multiarmed bandits are the foundation on which many other models are built. This class of bandits assumes that the environment draws the rewards in response to each action from an unknown distribution specific to that action and independent from the reward distributions of other actions. Therefore, a stochastic multiarmed bandit consists of a set of distributions $p = (P_a : a \in \mathcal{A})$ such that at time t for $t = 1, \dots, T$ the reward X_t from playing arm a_t is sampled from the distribution P_{a_t}. When these distributions remain unchanged in each round, we are talking about stationary stochastic bandits. We provide a simple example of a stochastic Bernoulli bandit, where T is the horizon, $A = \{a^{(1)}, \dots, a^{(K)}\}$ is the action set, and the Bernoulli specification means that the rewards can take the values zero or one, $X_t \in \{0, 1\}$. We can characterize the actions using a vector $\mu \in [0, 1]^K$, where each element of the vector, μ_i for $i = 1, \dots, K$, indicates the probability that the reward $X_{a^{(i)}}$ observed by the learner when choosing action $a^{(i)}$ is 1. We use the notation $\mu_i = \mu_{a^{(i)}}$ to indicate the mean value of action $a^{(i)}$ and $\widehat{\mu}_i(t)$ to indicate the empirical average of μ_i at time t. Recall that the setting is stochastic and stationary, therefore the reward distributions are fixed throughout the entire horizon and do not depend on previous actions and rewards. This means that the actions are characterized by their mean values μ_i and that the competitor class is the set of constant policies always playing the same action in each time step. Consequently, the optimal policy is to play the fixed best action $a^* = \operatorname{argmax}_{a \in \mathcal{A}} \mu_a$ in every time step. In this case, one can write the expected regret as:

$$R_T = T\mu_{a^*} - \mathbb{E}\left[\sum_{t=1}^{T} X_t\right], \tag{2.2}$$

where the first term is the cumulative reward collected by the policy π^* which plays arm a^* in every time step and it is the best possible performance for this specific environment. The second term is the cumulative reward of the policy played by the learner, with the expectation being with respect to the randomness of the environment and the policy. The strategy of a learner translates in practice into an algorithm played during the duration of the game. The analysis of this algorithm adopted by the learner is essential to prove some guarantees on the performance of the algorithm in the environment considered. By looking closely at the strategy played by the learner one can estimate the regret incurred by the algorithm, proving an upper and lower bound. As one can imagine, it is of primary importance to prove an upper bound on the regret in order to guarantee that the worst performance of the algorithm in this environment cannot be worse than the upper bound. A performance is considered good if the regret is sublinear, such that $R_T = o(T)$. However, one aims at getting a regret as small as possible, for instance, a regret which scales as $R_T = \mathcal{O}(\sqrt{T})$ or $R_T = \mathcal{O}(\log(T))$. Studying the regret means proving guarantees on the performance of your algorithm.
One of the first algorithms proposed to solve this bandit problem is `Explore-Then-Commit`, which can be abbreviated in `ETC`. The concept behind this algorithm is to explore each arm equally for a fixed number of times, look at which arm performed the best during this exploration phase, and then commit to this arm by playing it for the rest of the game. We denote with m the number of times each action is played during the exploration phase. For the first mK time steps, the learner plays action $a_t = (t \bmod K) + 1$ to explore the different arms in $\mathcal{A}$. For the following $T - mK$ time steps, the learner plays action $a_t = \max_{a^{(i)} \in \mathcal{A}} \widehat{\mu}_i(mK)$ for $t = mK + 1, \ldots, T$. If we denote with $\Delta_i = \mu^* - \mu_i$ the suboptimality gap of the mean of action $a^{(i)}$ with respect to the mean of the optimal arm, we can define the regret obtained by the ETC algorithm using this quantity. When $1 \leq m \leq T/K$, the performance of ETC is upper bounded by:

$$R_T \leq m \sum_{t=1}^{T} \Delta_i + (T - mK) \sum_{i=1}^{K} \Delta_i \exp\Big(- m\Delta_i^2 \Big). \tag{2.3}$$

One can notice that the larger m, meaning that the learner explores for longer, the bigger the first term gets. On the contrary, if m is too small, the policy might select the wrong arm to play for the remaining time steps, which will increase the second term. With some computations [Lattimore and Szepesvári, 2020], it is possible to show that a 2-armed bandit with an optimally tuned m, the regret bound yields a worst-case bound

$$R_T \leq \min\left\{ T\Delta_2, \Delta_2 + \frac{1 + \log(T\Delta_2^2)}{\Delta_2} \right\} \lesssim \sqrt{T \log(T)}. \tag{2.4}$$

However, the tuning of m depends on Δ_i and T which are rarely known in practical applications. In this case, when T is known but Δ is unknown, the parameter m can be tuned as $m = (T/K)^{2/3}(\ln T)^{1/3}$ and the regret becomes $R_T = \mathcal{O}(T^{2/3}(K \ln T)^{1/3})$. Notice that ETC depends on the knowledge of T so it is not an anytime algorithm, which is a denomination for those algorithms which do not require the knowledge of T.

Another possible solution, similar to `ETC`, would be to use *elimination-based strategies*. This type of algorithms consists of exploring the arms and adding an elimination phase to remove an action from the set of arms based on a test of increasing sensitivity. This strategy can prevent unnecessary plays of suboptimal actions but still shares some limitations with `ETC`. We present one specific example of an `elimination-based algorithm`. We denote with l the index of the phase and assume that the horizon T is known. For each phase $l = 1, 2, 3, \dots,$ the learner plays each arm $a_i \in \mathcal{A}_l$ for m_l times and computes the average reward $\widehat{\mu}_{i(l)}$ of each arm only counting the rewards obtained in the current phase. At the end of the phase, the learner updates the set of active arms as $\mathcal{A}_{l+1} = \{a^{(i)} : \widehat{\mu}_i(l) + 2^{-l} \geq \max_{a^{(j)} \in \mathcal{A}_l} \widehat{\mu}_j(l)\}$. This is repeated as long as the horizon is reached. It has been studied [Lattimore and Szepesvári, 2020] that for an appropriate universal constant $C > 0$, the regret of this algorithm is upper bounded by

$$R_T \leq \sum_{i=0}^{K} \Delta_i + C\sqrt{TK \log(K)}. \tag{2.5}$$

Among the plethora of algorithms in the literature, one has become the base of most popular solutions, whose variants have been developed and adapted to many other classes of bandits due to its efficiency. We are talking about the `Upper Confidence Bound Algorithm` [Lai and Robbins, 1985; Auer et al., 2002], which is based on the principle of *optimism in the face of uncertainty*. This algorithm is based on the index called upper confidence bound (UCB), which is an upper bound that overestimates the mean reward of an arm based on the data collected so far in the game. The idea behind this tool is to overestimate the actions' mean reward using this index and play the action with the highest UCB. Thus, an action is played only if its UCB index is higher than the UCB index of the optimal arm. Before defining the UCB index, we start by stating that for a sequence of independent random variables $X_1, \dots, X_T$ bounded in $[0, 1]$ with mean μ and $\widehat{\mu} = \frac{1}{T}\sum_{t=1}^{T} X_t$, then for any $\varepsilon \geq 0$:

$$\mathbb{P}(\widehat{\mu} \geq \mu + \varepsilon) \leq \exp\Big(-2T\varepsilon^2\Big) \quad \text{and} \quad \mathbb{P}(\widehat{\mu} \leq \mu - \varepsilon) \leq \exp\Big(-2T\varepsilon^2\Big). \tag{2.6}$$

We can use this statement and write

$$\mathbb{P}\left(\mu \geq \widehat{\mu} + \sqrt{\frac{\log(1/\delta)}{2T}}\right) \leq \delta \tag{2.7}$$

for all $\delta \in (0, 1)$. Here, δ is an upper bound on the probability of the event that the quantity $\widehat{\mu} + \sqrt{\frac{2\log(1/\delta)}{T}}$ is an underestimate of the true mean μ. We can use this knowledge to define the UCB index. When the game starts, the UCB index is set to infinity for every arm, $UCB_i = \infty$ for $i = 1, \dots, K$. Then, as soon as an arm a_i is played once, the UCB index will be equal to:

$$UCB_i(t-1) = \widehat{\mu}_i(t-1) + \sqrt{\frac{2\log(1/\delta)}{T_{t-1}(i)}}, \tag{2.8}$$

where $T_t(i)$ indicates how many times the action $a^{(i)}$ has been played so far, meaning that the bandit has collected $T_i(t)$ samples from arm a_i so far. The `Upper Confidence Bound Algorithm` consists of playing in each time step t the arm $a_t = \max_{a^{(i)} \in \mathcal{A}} UCB_i(t-1)$ with the current highest UCB index. After each round, the UCB indices are updated, and the process is repeated.
Notice that the term under the square root in the definition of the UCB index in Equation (2.8) is called exploration bonus since it forces the learner to consider those arms that have not been played many times. In fact, when the arm has not been frequently played, and $T_i(t-1)$ is small, the exploration bonus dominates the index. On the contrary, the more an action is played, the more its exploration bonus decreases and its empirical mean controls the index. This dynamic is useful in the algorithm when the learner has to decide which action to play. Indeed, this strategy makes sure that at the beginning each action is explored sufficiently enough. Then, the more an arm is played and its mean appears to be low, the less it will be played in the future. However, if an arm's index is high, it will still be played often. This is coherent with the fact that, even if an arm is not the optimal one, it is still played as long as its performance is close to the one of the optimal arm. The UCB algorithm has been extensively studied [Lattimore and Szepesvári, 2020; Auer et al., 2002] and it has been shown that for any horizon T, if $\delta = 1/T^2$, then the regret of a 1-sub-Gaussian stochastic K-armed bandits is upper bounded by:

$$R_T \leq 3\sum_{t=1}^{T} \Delta_i + \sum_{i:\Delta_i>0} \frac{16\log(T)}{\Delta_i}. \tag{2.9}$$

Rewriting the regret so that it does not depend on the reciprocal of the suboptimality gaps and for any 1-sub-Gaussian environment with K arms, it becomes

$$R_T \leq 8\sqrt{TK\log(T)} + 3\sum_{i=1}^{K} \Delta_i.$$

The UCB algorithm has been proven to be near minimax-optimal, meaning that its upper bound matches the lower bound $\Omega(\sqrt{TK})$ except for the logarithmic factor. This algorithm and its analysis are fundamental in the development of many other algorithms in the literature, as we will see in the rest of this dissertation.
So far, we focused on the class of stochastic stationary bandits. However, some of the assumptions we presented about the environment are not always realistic

when we think about real scenarios and practical applications. In the following paragraphs, we will build on the complexity of the environments and present different classes of bandits.

2.2 Adversarial multiarmed bandits

We start by introducing *adversarial multiarmed bandits* ([Auer et al., 1995]). In contrast with the previous class of stochastic bandits, there is a separate class of bandits for which the previous assumptions on the environment drop. In adversarial bandits the environment is seen as an adversary since it can secretly choose the rewards for each action after examining the code of the algorithm proposed by the learner. In this model the environment has complete control over the payoffs and acts as an adversary to the learner. In this manuscript we will not touch this subject; however, we will briefly introduce this class of algorithms to better comprehend the entire family of multiarmed bandits. Due to the fundamentally different assumptions on the generation of the rewards, we take a moment to introduce the framework.
An adversarial multiarmed bandit is described by an arbitrary sequence of vectors $\{x_1, \dots, x_T\}$, where $x_t \in [0,1]^K$ for $t = 1, \dots, T$, which are secretly chosen by the adversarial environment. In each round t, the learner chooses a distribution over the actions $P_t \in \mathcal{P}_{K-1}$ and samples from P_t an action a_t from the set of actions $\mathcal{A} = \{a^{(1)}, \dots, a^{(K)}\}$ to play in the current round. After the action is played, the learner observes a reward $x_t(a_t)$. In this specific setting, the policy is defined as $\pi : (\mathcal{A} \times [0,1])^* \to \mathcal{P}_{K-1}$, which is a function that maps the sequences of history to the distributions over the actions. Similarly to the stochastic setting, the performance of a policy π in environment x is measured by the expected regret, which is the difference between the expected total reward of the best fixed action in hindsight and the policy played by the learner. It is formally defined as follows

$$R_T(\pi, x) = \sum_{t=1}^{T} x_t(a^*) - \mathbb{E}\left[\sum_{t=1}^{T} x_t(a_t)\right], \tag{2.10}$$

where $a^* = \max_{a^{(i)} \in \mathcal{A}} \sum_{t=1}^{T} x_t(a^{(i)})$ and the expectation is with respect to the randomness of the learner's policy. We will abbreviate this notation $R_T(\pi, x)$ by simply writing R_T when it is clear from the context. Using this definition, it is possible to write the worst-case regret as

$$R_T^* = \sup_{x \in [0,1]^{(T \times K)}} R_T(\pi, x), \tag{2.11}$$

which is the maximum regret over all possible environments. Indeed, a key element in successful strategies for adversarial environments is the randomness introduced by the learner when selecting the actions to play. Without randomisation, the environment could easily take advantage of the learner's deterministic policy, which could not provide a sublinear regret. Here, we illustrate an algorithm for adversarial bandits called Exp3 [Auer et al., 1995]. This algorithm uses importance-weighted estimators in order to estimate the rewards of the arms that have not been played. It consists of computing for each arm $a^{(i)} \in \mathcal{A}$ the sampling distribution $P_{i,t}$:

$$P_{i,t} = \frac{\exp(\eta \widehat{S}_{i,t-1})}{\sum_{j=1}^{K} \exp((\eta \widehat{S}_{j,t-1})}, \tag{2.12}$$

where $\eta > 0$ is the learning rate parameter, $\widehat{S}_{i,t} = \sum_{s=1}^{t} \widehat{y}_t(a^{(i)})$ is the estimate of the total reward at the end of round t, and $\widehat{y}_t(a^{(i)})$ is the importance-weighted estimator of the reward $x_t(a^{(i)})$. After computing the distribution P_t, the learner samples the actions to play from this distribution and receives a reward $x_t(a_t)$, which we denote also using X_t. The reward is then used to update the estimates $\widehat{S}_{i,t}$ as:

$$\widehat{S}_{i,t} = \widehat{S}_{i,t-1} + \frac{\mathbb{I}\{a_t = a^{(i)}\}(1 - X_t)}{P_{i,t}}. \tag{2.13}$$

In this solution, exponential weighting is used to map the estimates of the total rewards to probabilities and plays actions with higher estimates with higher probability. It is possible to notice how the parameter η in Equation (2.12) governs the balance between exploration and exploitation. When η is small, the distribution P_t becomes more uniform and it encourages exploration. On the contrary, as η becomes larger, the distribution focuses on the arm with the highest estimate, promoting an exploitative behaviour. The regret of this algorithm has been proven to be upper bounded by:

$$R_T \leq 2\sqrt{TK \log(K)}, \tag{2.14}$$

for $\eta = \sqrt{\log(K)/(TK)}$. Although this bound is sublinear, it is proven with expectation and it is possible to show that its variance is large. Therefore, it is worth mentioning that with some modifications it is possible to overcome this problem. Specifically, modifying the reward estimates, the algorithm EXP3-IX [Kocák et al., 2014] presents a high-probability bound well concentrated around its mean, reaching a regret upper bound of $R_T = \mathcal{O}(\sqrt{TK \log(K/\delta)})$ with probability at least $1 - \delta$ for $\delta \in (0, 1)$. Although the environments of adversarial and stochastic bandits are quite distant from each other, one can notice how the regret for this algorithm for adversarial bandits is close to the regret presented for stochastic bandits.

In the models presented above, there is no feature associated with the arms. However, in practical applications, such as recommender systems, additional information is usually given in association with users and items. For instance, in a movie recommender system, each item is typically described by a set of features such as genre, year, cast members, director, and so on. Likewise, users may be described by their demographic information or a history of previously watched movies. In this case, ignoring this side information would not be advisable. In fact, the quality of predictions can greatly benefit from this data. Hence, a particular class of bandits emerges from this opportunity and it is called contextual multiarmed bandits ([Li et al., 2010], [Woodroofe, 1979]). To present this class, we must first introduce a class of bandits closely related to contextual bandits, the family of linear multiarmed bandits.

2.3 Adversarial linear multiarmed bandits

Linear bandits can be declined both in the adversarial and in the stochastic setting. We start by introducing *adversarial linear bandits* and then we will move to the stochastic environment.
This setting is characterized by the presence of an adversarial environment that chooses the reward vectors $(x_t)_{t=1}^T$ such that $x_t \in [0,1]^K$. We use the term $X_{i,t}$ to indicate the single reward of arm $a^{(i)}$ at time t. Moreover, the environment chooses the sequence of context, one for each time step, which we indicate by $(c_t)_{t=1}^T$ such that $c_t \in \mathcal{C}$ where $\mathcal{C}$ is the set of possible contexts. The protocol is the following:

- The adversarial environment secretly decides the reward vectors $(x_t)_{t=1}^T$,
- The adversarial environment secretly picks the contexts $(c_t)_{t=1}^T$.

Then, in each round $t = 1, \dots, T$:

- The learner observes the context $c_t \in \mathcal{C}$,
- The learner selects a distribution $P_t \in p$ and samples the arm to play a_t from P_T,
- The learner observes the reward of the pulled arm $X_t(a_t)$ which depends on the observed context c_t.

In the setting of contextual bandits, the regret is formally defined as:

$$R_T = \mathbb{E}\left[\sum_{c\in\mathcal{C}} \max_{a^{(i)}\in\mathcal{A}=\{a^{(1)},\dots,a^{(K)}\}} \sum_{t\in\{1,\dots,T\}:c_t=c} \Big(X_{i,t} - X_t\Big)\right], \tag{2.15}$$

which is the sum of rewards collected by the best policy for each context in hindsight and the reward observed by the policy played by the learner.
In case the set of contexts $\mathcal{C}$ is finite, a simplistic approach would be to instantiate a bandit algorithm for each context. In this case, the final regret would be impacted by the regret of the algorithm weighted by the cardinality of the set of contexts. However, if the set of contexts is too big, using a bandit per context would not be an efficient solution, since the regret would scale with $|\mathcal{C}|$. An alternative solution would be to partition the set of contexts into disjoint parts $P \in \mathcal{P}$ with $\mathcal{P} \subset 2^{\mathcal{C}}$ such that $\cup_{P\in\mathcal{P}} P = \mathcal{C}$. Then, for each part in $\mathcal{P}$ there is a function from $\mathcal{C}$ to the set of actions $\{1, \dots, K\}$ which is constant for that part $P \in \mathcal{P}$. For each part, the learner can run an instance of a bandit algorithm and eventually the regret would depend on the number of parts $|\mathcal{P}|$ instead of the number of contexts $|\mathcal{C}|$. Among all the possible approaches, it is necessary to mention the framework of bandits with expert advice. In this adversarial environment, the learner is equipped with M experts. At the beginning of each round, the experts announce which actions are the most rewarding, providing a probability distribution over the actions. The experts are represented by $E_t \in [0,1]^{M\times K}$. We indicate with $E_t(j)$ with $j \in \{1, \dots, M\}$ the prediction made by expert j at time t. The learner selects an expert $E_t(j_t)$ and plays

the action recommended by the expert by sampling the action $a_t \sim P_t$, where the distribution is defined as $P_t = Q_t E_t$ and Q_t is the distribution over the experts. Then, once the reward is observed by the learner, it is propagated to the experts, who can update their statistics using importance-weighted estimators and exponential weighting similarly to EXP3. With the additional presence of experts, this algorithm is called EXP4 [Auer et al., 2002]. Using this algorithm in this setting, the learner obtains a regret upper bound of $R_T \leq \sqrt{2TK\log(M)}$ when $\eta = \sqrt{2\log(M)/(TK)}$.
Here, we briefly mentioned the environment of adversarial linear bandits for completeness. Now we move to the stochastic counterpart.

2.4 Stochastic contextual multiarmed bandits

The adversarial setting we mentioned is in contrast with the stochastic contextual bandits. This class of bandit generalises the setting of linear bandits, which is a prominent class of bandits. We start by defining stochastic *contextual bandits* and then move to linear bandits. The protocol that frames stochastic contextual bandits reflects the one presented for adversarial contextual bandits but in a stochastic environment. So, removing the presence of an adversarial environment, the steps are the following. In each round $t = 1, \dots, T$:

- The environment presents a context $c_t \in \mathcal{C}$, which may be random or not,
- The learner observes the context c_t,
- The learner selects an arm to play $a_t \in \{1, \dots, K\}$
- The learner observes the reward $X_t \in [0, 1]$ which if obtained by a_t and depends on the observed context c_t.

Apart from the fact that there is no adversarial component in this protocol, the stages of this framework are similar to the adversarial counterpart except that the current setting presents a key assumption on the reward function. In fact, the reward function is now defined as:

$$X_t = r(c_t, a_t) + \eta_t, \tag{2.16}$$

where $r : \mathcal{C} \times \{1, \dots, K\} \to \mathbb{R}$ is the reward function that takes as inputs the context at time t and the arm pulled at time t and returns a real number, and η_t is the noise, which can be assumed to be conditionally 1-sub-Gaussian for simplicity. Since the learner observes the context at the beginning of the round, if the function r were known, then the best strategy would be to play the arm $a_t^* = \text{argmax}_{a^{(i)} \in \mathcal{A} = \{a^{(1)}, \dots, a^{(K)}\}} r(c_t, a^{(i)})$ that maximizes the reward in the current time step, which depends on the given context c_t. However, the reward function r is typically unknown. One may define the regret as:

$$R_T = \mathbb{E}\left[\sum_{t=1}^{T} \max_{a^{(i)} \in \mathcal{A}} r(c_t, a^{(i)}) - \sum_{t=1}^{T} X_t\right]. \tag{2.17}$$

Nevertheless, since we do not know how r may be defined, this formalization of the regret could be misleading. As we mentioned for the adversarial counterpart, one approach could be to estimate the reward function for each context-arm pair $(c, a) \in \mathcal{C} \times \{1, \dots, K\}$. But, as we said, it would not be efficient since the regret would be weighted by the number of context-arm pairs. Among the possible approaches, one interesting choice is to assume the linearity of the reward function defined as:

$$r(c, a) = \langle \theta^*, \psi(c, a) \rangle \tag{2.18}$$

for every context-arm pair $(c, a) \in \mathcal{C} \times \{1, \dots, K\}$ and where $\theta^* \in \mathbb{R}^d$ is the unknown parameter vector. In this definition of the reward function, the learner has access to $\psi : \mathcal{C} \times \{1, \dots, K\} \to \mathbb{R}$ which is a feature map that maps the context-arm pair to a d-size vector of real numbers. The concept is close to the definition of a feature map in Neural Networks, where the aim is to extract important features from the input data. By introducing this linear assumption on the reward function, one can approach stochastic contextual bandits by viewing them as a linear bandit where the reward function is defined as the dot product between the unknown parameter θ^* and a vector of size d. Here, this vector is interpreted as an action $a_t \in \mathbb{R}^d$ and can be seen as the vector resulting from $\psi(c, a)$. In the next paragraph, we are going to present this class of bandits, which is one of the most relevant bandit problems in the literature.

2.5 Stochastic linear multiarmed bandits

Stochastic linear bandits are a family of bandit problems where the learner has access to a set of actions $\mathcal{A}_t \subset \mathbb{R}^d$, typically infinite, and the reward function of an arm $a_t \in \mathcal{A}$ is given by:

$$X_t = \langle \theta^*, a_t \rangle + \eta_t, \tag{2.19}$$

where $\theta^* \in \mathbb{R}^d$ is the unknown parameter vector and η_t is the 1-sub-Gaussian noise. In every round, $t = 1, \dots, T$, the learner selects an action a_t to play, and obtains a reward X_t. The performance of the learner is measured by the regret, which is defined as:

$$R_T = \mathbb{E}\left[\sum_{t=1}^{T} \max_{a \in \mathcal{A}_t} \langle \theta^*, a \rangle - \sum_{t=1}^{T} X_t\right]. \tag{2.20}$$

One can also refer to the definition of random pseudo-regret defined as:

$$\widehat{R}_T = \sum_{t=1}^{T} \max_{a \in \mathcal{A}_t} \langle \theta^*, a - a_t \rangle, \tag{2.21}$$

so that one can define $R_T = \mathbb{E}\left[\widehat{R}_T\right]$.
One way to characterize this setting is to make assumptions on the set of actions $\mathcal{A}_t$. In fact, stochastic linear bandits are a generalisation of stochastic K-armed

bandits, which can be recovered when the set of actions is reduced to unit vectors, $\mathcal{A}_t = \{e_1, \dots, e_d\}$. On the other hand, assuming that $\mathcal{A}_t \subseteq \{0,1\}^d$ reduces the model to combinatorial bandits.
We analyze the most popular algorithm for stochastic linear bandits, called LinUCB or OFUL [Abbasi-Yadkori et al., 2011]. This solution is an adaptation to the Upper Confidence Bound algorithm to the linear setting. To define the UCB index for the linear setting, it is necessary to define what we call a confidence set. The confidence set $\mathcal{C}_t \subset \mathbb{R}^d$ is an ellipsoid which defines the set to which the unknown parameter θ^* should belong with high probability. We will define the construction of this confidence set later on, but for now it is sufficient to know that it is constructed based on the history of pulled arms and rewards obtained $(a_1, X_1, \dots, a_{t-1}, X_{t-1})$ in the previous time steps. Before defining the confidence set, we start by providing the idea behind the linear UCB index. It is defined as:

$$UCB_a(t) = \max_{\theta \in \mathcal{C}_t} \langle \theta, a \rangle, \tag{2.22}$$

where $a \in \mathcal{A}_t$. This definition acts as an upper bound on the mean reward of the dot product $\langle \theta^*, a \rangle$ since it is computed with the best possible $\theta \in \mathcal{C}_t$ in the confidence set according to the action a. The algorithm works similarly as to the finite arm case. In each round, the learner selects the action with the highest UCB:

$$a_t = \operatorname*{argmax}_{a \in \mathcal{A}_t} UCB_a(t), \tag{2.23}$$

obtains the reward $X_t \in \mathbb{R}$, and uses this reward to update the estimate of the unknown parameter, $\widehat{\theta}_t$, and the confidence set. The unknown parameter θ^* is, in fact, estimated using the regularised least-squares estimator:

$$\widehat{\theta}_t = \operatorname*{argmin}_{\theta \in \mathbb{R}^d} \Big(\sum_{s=1}^{t} (X_s - \langle \theta, a_s \rangle)^2 + \lambda \|\theta\|_2^2 \Big), \tag{2.24}$$

where $\lambda \geq 0$ is the so-called penalty factor. Through some simple computations, it is possible to compute the estimate of θ^* as :

$$\widehat{\theta}_t = V_t^{-1} \sum_{s=1}^{t} a_s X_s, \tag{2.25}$$

where

$$V_0 = \lambda I, \quad V_t = V_0 + \sum_{s=1}^{t} a_s a_s^\top, \tag{2.26}$$

where $V_t \in \mathbb{R}^{d \times d}$ and I is the identity matrix. This allows us to better define the confidence set. It is typically assumed to be $\mathcal{C}_t \subseteq \mathcal{E} = \{\theta \in \mathbb{R}^d : \|\theta - \widehat{\theta}_{t-1}\|^2_{V_{t-1}} \leq \beta_t\}$ with an increasing sequence of constants $\beta_1 \geq 1$, as it is defined for the works discussed in this thesis. As we said, it is an ellipsoid centred at $\widehat{\theta}_{t-1}$ and its shape is defined by the history of past played actions and rewards. The axes of the ellipsoid are defined by the eigenvectors of V_t and the lengths dictated by the reciprocal of the eigenvalues so that with time and with growing eigenvalues of matrix V_t, the confidence set shrinks based on the actions played in the past. Before presenting the regret of LinUCB, it is necessary to specify which assumptions must hold for the regret analysis to hold. The proof requires that:

- $1 \leq \beta_1 \leq \cdots \leq \beta_T$,
- $\max_t \sup_{a,b \in \mathcal{A}_t} \langle \theta^*, a - b \rangle \leq 1$,
- $\|a\|_2 \leq L, \forall a \in \cup_{t=1}^{T} \mathcal{A}_t$,
- $\exists \delta \in (0,1)$ such that with probability $1-\delta$, $\forall t \in \{1,\ldots,T\}$ $\theta^* \in \mathcal{C}_t$.

If these assumptions hold, then with probability $1-\delta$ the regret of LinUCB is upper bounded by:

$$R_T \leq \sqrt{8dT\beta_T \log\left(\frac{d\lambda + TL^2}{d\lambda}\right)}, \tag{2.27}$$

which can be written in expectation as:

$$R_T \leq Cd\sqrt{T}\log(TL), \tag{2.28}$$

with $\delta = 1/T$ and $C > 0$ an appropriate universal constant.
OFUL proves to be an important contribution to the bandit literature since it adapts the use of UCB indices in a linear setting, which is at the foundation of later works. Due to the importance of OFUL and its proof in one of our contributions, we provide the proof of this regret bound in Appendix A.1.

2.6 Combinatorial (semi)-bandits

The next class of bandits we present is *combinatorial multiarmed bandits* ([Cesa-Bianchi and Lugosi, 2012; Gai et al., 2012; Chen et al., 2013; Kveton et al., 2015]). Combinatorial bandits are characterized by the fact that in each time step the learner plays a so-called *super-arm* from an action set defined as

$$\mathcal{A} \subseteq \{\boldsymbol{a} \in \{0,1\}^L : \|\boldsymbol{a}\|_1 \leq m\}, \tag{2.29}$$

where L is a positive natural number defining the size of the action vector, which therefore indicates the cardinality of the ground set of base actions, and m is the maximum number of elements taking the values 1 in the vector, therefore the maximum number of chosen items. In some settings, we have that $\mathcal{A} \subset \{0,1\}^L$ due to the fact that the combinatorial structure does not allow every possible combination. One can think of the element $\boldsymbol{a}(i)$ of the super-arm to be on (1) or off (0), activated or not. One can possibly interpret this combinatorial action $\boldsymbol{a}$ as a vector of size K ($L = K$) where each element of the vector $\boldsymbol{a}(i)$ corresponds to one of the K base arms in the initial set of actions $\mathcal{A} = \{a^{(1)}, \ldots, a^{(K)}\}$. With this interpretation, the component $\boldsymbol{a}(i)$ indicates if the arm a_i is activated or

not. In this case, combinatorial bandits essentially allow the learner to play more actions in a single time step.
It is also possible to consider this class of models as a subset of linear bandits where the norm of an action of size d is bounded by m. To remain as general as possible we will consider the definition in Equation (2.29).
In the definition of combinatorial bandits, each component of the vector $\boldsymbol{a}$ is characterized by a probability distribution $\boldsymbol{a}(i) \sim P_i$. In each time step, the learner plays an action vector $\boldsymbol{a}$ and observes the sum of the rewards the vector obtains. The reward of the vector is observed as the sum of the rewards of the activated elements. We can define the compound reward collected at time t as $\boldsymbol{X}_t = \sum_{i=1}^{L} X_t(v(i)) \cdot \boldsymbol{a}(i)$.
However, from the way the learner observes the reward, we can distinguish between two types of combinatorial bandits. On one hand, we have combinatorial bandits where the learner observes only the sum of the rewards collected by the actions activated in the vector $\boldsymbol{a}_t$, without knowing the specific rewards obtained by the different elements of the super-arm. On the other hand, we have *combinatorial semi-bandits*, where the learner can observe not only the sum of the rewards but also the individual rewards obtained by each component $\boldsymbol{a}(i)$. In both cases, it is possible to note that if the assumption of the boundedness of the rewards $X_t \in [0, 1]$ remains true, then we have that the reward of the super-arm $\boldsymbol{a}$ is bounded as $\boldsymbol{X}_t \in [0, m]$. Of course, one can consider combinatorial bandits in the stochastic setting, as well as in the adversarial or non-stationary setting, depending on the probability distributions P_i associated with the components $\boldsymbol{a}(i)$ and the assumptions on the environment.
Since this dissertation will focus on stochastic environments, we focus on stochastic combinatorial semi-bandits and mention an algorithm to solve this problem based on upper confidence bounds. The algorithm is called `CombUCB1` ([Gai et al., 2012], [Kveton et al., 2015]). It adopts the UCBs strategy and adjusts it for the combinatorial semi-bandit setting. This is done by defining the UCB index of a super-arm $\boldsymbol{a}$ as:

$$UCB_{\boldsymbol{a}}(t) = \sum_{i=1}^{L} UCB_{\boldsymbol{a}(i)}(t), \tag{2.30}$$

which is the sum of the UCB indices of the arms in $\boldsymbol{a}$ and where the UCB of an element $\boldsymbol{a}(i)$ is defined as:

$$UCB_{\boldsymbol{a}(i)}(t) = \widehat{\mu}_{\boldsymbol{a}(i)}(t) + \sqrt{\frac{1.5 \log(t)}{T_i(t)}}, \tag{2.31}$$

where $T_i(t)$ counts the number of times action $a^{(i)}$ has been played. This definition slightly differs from Equation (2.8) for the term inside the logarithm, but the concept behind UCBs remains the same. In each round, the algorithm plays the super-arm:

$$\boldsymbol{a}_t = \operatorname*{argmax}_{\boldsymbol{a} \in \mathcal{A}} UCB_{\boldsymbol{a}}(t) = \operatorname*{argmax}_{\boldsymbol{a} \in \mathcal{A}} \sum_{i=1}^{L} UCB_{\boldsymbol{a}(i)}(t). \tag{2.32}$$

Then, once the learner observes the sum of the rewards $\boldsymbol{X}_t$ as well as the single rewards $X_{i,t}$, it updates the statistics, such as μ_i and $T_i(t)$ for $i = 1, \ldots, L$, before starting a new iteration. In [Kveton et al., 2015], the authors prove an expected regret upper bound for `CombUCB1` of

$$R_T = \mathcal{O}(47\sqrt{mKT \log T} + mK)$$

and a lower bound of $R_T \geq (1/20) \min(\sqrt{mKT}, mT)$. This algorithm and the proof of its regret are crucial for understanding the contributions of Chapter 4 and Chapter 5, therefore we provide details on its analysis in Appendix A.2.

2.7 Non-stationary multiarmed bandits

In the previous paragraphs, we anticipated that there is another category of stochastic bandits, which is based on the assumption that the reward distributions of the arms change over time. This is the class of *non-stationary stochastic multiarmed bandits*, which is the primary class we are focusing on in this research. In the next chapter, we will describe several non-stationary bandit models useful to understand our contributions. Specifically, we will present the state of the art for this class of models. For the moment, we characterize this class of algorithms by the fact that the reward function depends on time t. This means the arm is not characterized by the mean reward $\mu_{a^{(i)}}$ for the entire game. Instead, each arm is characterized by mean reward $\mu_{a^{(i)}}(t)$, meaning that the average reward of the arm $a^{(i)}$ changes depending on t. The different levels and forms of non-stationarity depend on the assumptions on the reward functions and the environment, such as a variation budget on the reward function, change points used to define the type of non-stationarity, and other assumptions. Regardless of the specifics of the model, these characteristics suggest one important aspect of this class: the optimal policy here does not consist of a single optimal arm. Instead, the optimal policy is given by an optimal ordered sequence of actions, where the order, the set of actions, and its cardinality depend on the specific assumptions on the environment. To measure the performance of an algorithm π, we introduce a new measure called *dynamic regret*:

$$R_T = \underset{a_1,\ldots,a_T \in \mathcal{A}}{\operatorname{argmax}} \sum_{t=1}^{T} r_t(\pi^*, a_t) - \sum_{t=1}^{T} r_t(\pi, a_t). \tag{2.33}$$

This definition differs from the static regret defined in Equation (2.2) since the cumulative reward is measured against the one obtained by the optimal sequence of actions, not the optimal arm, dictated by π^*. As one can imagine, the assumptions on the non-stationarity greatly characterize the problem, its difficulty, and the strategy used to solve it. For this reason, we dedicate the next chapter to the introduction and analysis of different non-stationary settings.

Chapter 3

The state of the art of non-stationary bandits

Contents

This chapter dives into the current state of the art of non-stationary multi-armed bandit models. We consider different types of non-stationarity and different environments. Since the contributions of this dissertation focus on the standard K-armed setting and the linear setting, we present state-of-the-art non-stationary bandit models for these environments.
First, we make a distinction between two types of non-stationarity: *exogenous* and *endogenous*. With the term *exogenous* we indicate those models where the non-stationarity of the rewards is due to external factors, which are independent of the learner's actions throughout the sequential game. This class is commonly referred to as *restless bandits*. This category might be seen as a subset of adversarial bandits since the non-stationarity is dictated by the environment, although there is no explicit adversary. Most of the models that study this setting introduce some assumptions to bound the non-stationarity, such as defining a piece-wise stationary or bandits with variation budget. We will discuss these models in Section 3.1.
On the other hand, we denote a type of non-stationarity as *endogenous* if the learner's actions have an impact on future rewards, meaning that the non-stationarity of these rewards is given by internal causes. This second category is seen as a ramification from the stochastic bandit models since the non-stationary distribution of an arm follows a specific function that depends on the previous

history of actions. Inside this class, we make a distinction between *rested bandits* and *state-dependent bandits*. The first category identifies those bandits where the reward of the pulled arm changes differently with respect to the rewards of those arms that have not been pulled. Oftentimes, the reward functions of these models depend on the number of pulls of an action. On the other hand, we classify as *state-dependent bandits* those models which are neither restless nor rested, where the reward of an arm depends on the time elapsed since it was last pulled. We use the term *state-dependent* since these models often use the concept of an arm state to keep track of how many time steps elapsed since this arm was last played.
The contributions of this dissertation focus on endogenous types of non-stationarity, both rested and state-dependent, where the learner's actions influence future rewards and these rewards are governed by specific functions. Before diving into that, in the rest of this chapter we introduce the prominent works in the literature that focus on both exogenous and endogenous non-stationary bandits, before presenting our contributions in the following chapter. Figure 3.1 presents an overview of the different classes and models of non-stationary bandits presented in this chapter, along with our contributions.

3.1 Exogenous non-stationarity

As we mention, we classify as *exogenous* the type of non-stationarity which is caused by external factors. The change in the rewards does not depend on the policy adopted by the learner and their actions but on time, and it is intrinsic to the environment. We use the term *restless bandits* to denote the bandits that model this behaviour.
Restless bandits were first introduced by [Whittle, 1988] as those bandits where the mean rewards of the actions change over time independently of the learner's actions. This class of bandits, characterized by an exogenous type of non-stationarity, can be divided into different groups depending on the nature of this non-stationary behaviour. We identify two main subclasses: *piece-wise stationary bandits* (or *switching bandits*), which can be further divided into *abruptly-changing* and *slowly-changing* environments, and *bandits with variation budget*. In the next sections, we present the most relevant works in these areas that help understand the contributions of this dissertation.

3.1.1 Piece-wise stationary bandits: abruptly-changing and slowly-changing environments

One category is the class of bandit models whose non-stationary environment can be defined as *piece-wise stationary*. This class is also addressed as *switching bandits* since [Garivier and Moulines, 2011]. The setting of these models is characterized by the fact that there are *change-points* (also called *breakpoints*) throughout the horizon where the reward distributions of the arms change. The peculiarity of this model is that between two change-points, the distributions of the arms remain unchanged. It is important to notice how these breakpoints do not depend on the policy of the learner or on the rewards obtained but

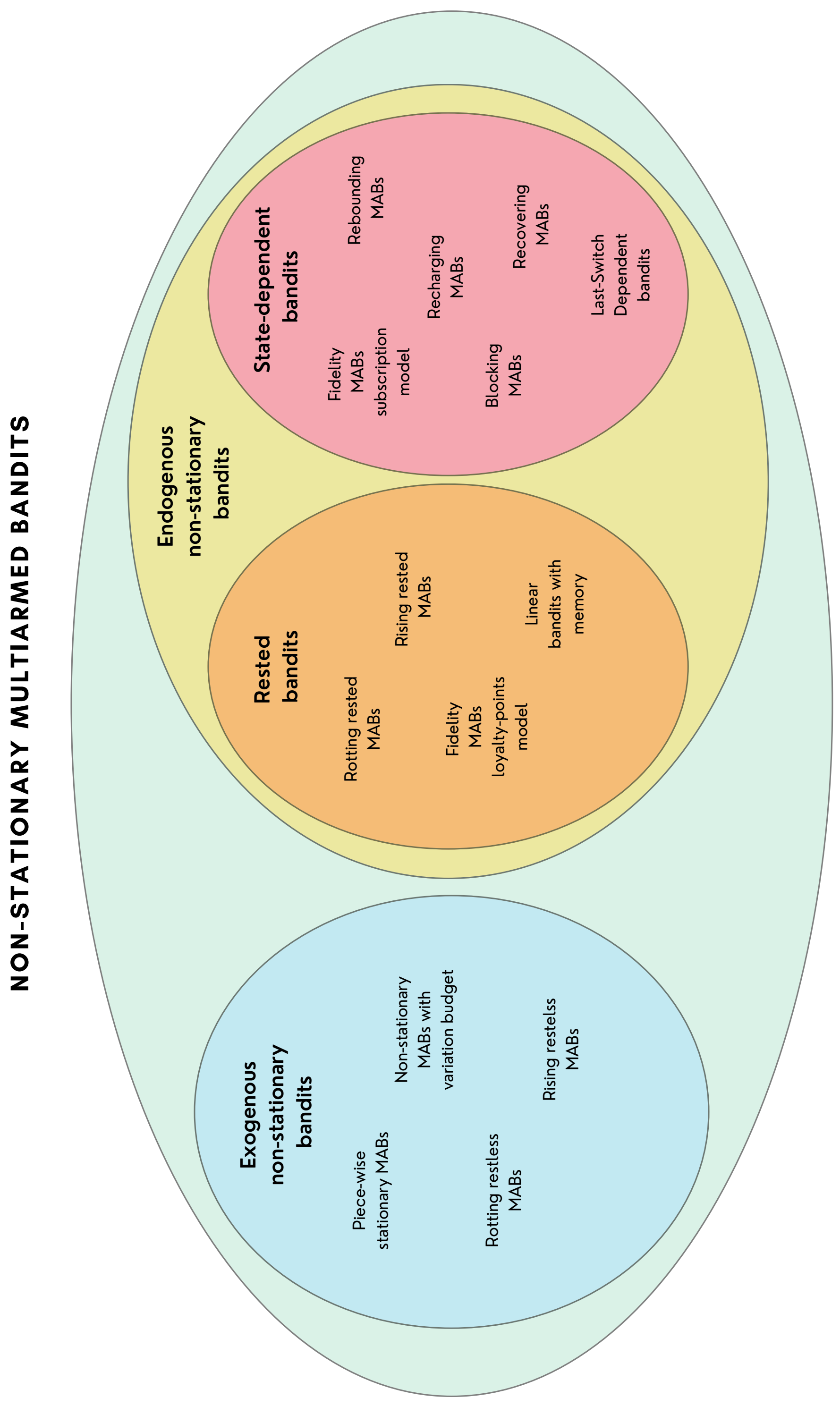
NON-STATIONARY MULTIARMED BANDITS
Exogenous non-stationary bandits
Piece-wise stationary MABs
Non-stationary MABs with variation budget
Rotting restless MABs
Rising restelss MABs
Endogenous non-stationary bandits
Rested bandits
Rotting rested MABs
Rising rested MABs
Fidelity MABs loyalty-points model
Linear bandits with memory
State-dependent bandits
Fidelity MABs subscription model
Rebounding MABs
Recharging MABs
Blocking MABs
Recovering MABs
Last-Switch Dependent bandits

Figure 3.1: This figure summarises the state of the art of non-stationary bandits, identifying the different classes and the models belonging to them.

are fixed and predefined by the environment at the beginning of the game. In between these two points, the bandit problem could potentially be solved using an algorithm for stationary bandits. However, the algorithms proposed to solve this type of bandits must take care of these abrupt changes in order to provide a sublinear regret.
In this type of environment ([Kocsis and Szepesvari, 2006], [Hartland et al., 2007], [Yu et al., 2009], [Garivier and Moulines, 2011], [Alami et al., 2017], [Wei and Srivatsva, 2018], [Besson and Kaufmann, 2019]), where $\mathcal{A} = \{a^{(1)}, \dots, a^{(K)}\}$ is the finite set of arms, the non-stationarity is usually defined by the following assumption:

$$\Upsilon_T = \sum_{t=1}^{T-1} \mathbb{I}\{\exists a^{(i)} \in \mathcal{A} : \mu_i(t) \neq \mu_i(t+1)\} \tag{3.1}$$

where i is an index used to indicate arm $a^{(i)} \in \mathcal{A}$ and the term Υ_T limits the number of change points that can appear during the sequential game. The j-th breakpoint in the game is denoted by $\tau_{(j)} = \inf\{t > \tau_{(j-1)} : \exists a^{(i)} : \mu_i(t) \neq \mu_i(t+1)\}$. This definition explains that between two breakpoints $\tau_{(j)}$ and $\tau_{(j+1)}$ the mean rewards of the arms are stationary and do not change. However, when there is a breakpoint, there is at least one arm whose mean reward changes. Some of the algorithms proposed to solve this problem combine algorithms for stationary bandits and techniques for detecting change points, overwriting the mean rewards in each stationary interval between two change points. Some other algorithms operate so that the past average rewards are gradually forgotten by implementing sliding windows or discounting factors to ensure that the algorithm considers recent rewards as more relevant. In the rest of this section, we present some works in the literature.
This class of exogenous bandit was first proposed by [Kocsis and Szepesvari, 2006] and [Yu et al., 2009], which initiated a plethora of works concerning switching bandits. In [Garivier and Moulines, 2011], the authors propose two algorithms that they call `discounted UCB` and `sliding-window UCB`. The first algorithm, `discounted UCB`, consists of playing each arm once and then selecting the arm that maximizes the discounted empirical average reward summed to the exploration bonus $c_t(\gamma, a^{(i)})$, which both depend on a discount factor $\gamma \in (0, 1)$:

$$a_t = \operatorname*{argmax}_{a^{(i)} \in \mathcal{A}} \bar{X}_t(\gamma, a^{(i)}) + c_t(\gamma, a^{(i)}), \tag{3.2}$$

where the discounted mean reward is defined as

$$\bar{X}_t(\gamma, a^{(i)}) = \frac{1}{N_t(\gamma, a^{(i)})} \sum_{s=1}^{t} \gamma^{t-s} X_s(a^{(i)}) \mathbb{I}\{a_s = a^{(i)}\}$$

$$\text{where} \quad N_t(\gamma, a^{(i)}) = \sum_{s=1}^{t} \gamma^{t-s} \mathbb{I}\{a_s = a^{(i)}\}.$$

The exploration bonus in this setting is defined as

$$c_t(\gamma, a^{(i)}) = 2B\sqrt{\xi \log(n_t(\gamma))/N_t(\gamma, a^{(i)})}$$

where $n_t(\gamma) = \sum_{i=1}^{K} N_t(\gamma, a^{(i)})$, $\xi > 0$ is an appropriate constant, and B is an upper-bound on the rewards. When $\gamma = 1$, the algorithm recovers the standard UCB algorithm. However, when $\gamma < 1$, the computation of the mean reward $\bar{X}_t(\gamma, a^{(i)})$ and the exploration bonus ensures that recent rewards have more weight in the computation of the average with respect to the ones far back in the past. The authors show that `discounted UCB` obtains a regret upper bound of $\mathcal{O}(\sqrt{\Upsilon_T T}\log(T))$, for $\gamma = 1 - (4B)^{-1}\sqrt{\Upsilon/T}$ and specific choices of the parameters to optimize this quantity.
The second algorithm the authors proposed in [Garivier and Moulines, 2011] is `sliding-window UCB`. This algorithm expects the learner to play each action once and then select the action that maximizes the following quantity:

$$a_t = \operatorname*{argmax}_{a^{(i)} \in \mathcal{A}} \bar{X}_t(\tau, a^{(i)}) + c_t(\tau, a^{(i)}), \tag{3.3}$$

where $c_t(\tau, a^{(i)}) = B\sqrt{\xi \log(\min\{t, \tau\})/N_t(\tau, a^{(i)})}$. The difference here with respect to Equation (3.2) is in the definition of the exploration bonus, which does not depend on γ anymore but on τ. Here, τ indicates the last plays that the algorithm considers when computing the average reward of an action. The definitions of the average reward and the number of plays of an action become:

$$\bar{X}_t(\tau, a^{(i)}) = \frac{1}{N_t(\tau, a^{(i)})} \sum_{s=t-\tau+1}^{t} \gamma^{t-s} X_s(a^{(i)}) \mathbb{I}\{a_s = a^{(i)}\} \tag{3.4}$$

$$\text{where} \quad N_t(\tau, a^{(i)}) = \sum_{s=t-\tau+1}^{t} \mathbb{I}\{a_s = a^{(i)}\}. \tag{3.5}$$

The idea is to use a sliding window of size τ which forgets the reward accumulated outside of the window. The goal is to keep track only of recent rewards in the hope that the sliding window is able to avoid the breakpoints. This algorithm attains a regret upper bound of

$$\mathcal{O}(\sqrt{\Upsilon_T T \log(T)}\log(T))$$

for $\tau = 2B\sqrt{T\log(T)/\Upsilon}$ and specific choices of the parameters. Therefore, `sliding-window UCB` shows a better upper bound just by a $\sqrt{\log(T)}$ factor.

The setting we presented in Equation (3.1) and addressed up to this moment is called *per-arm (or local) abruptly-changing environment*, where the change-point occurs if there is at least one arm whose mean reward changes. However, proceeding chronologically, it is interesting to cite the work of [Alami et al., 2017], where the authors proposed a new solution to the switching bandit problem with Bernoulli arms distributions in a *global switching bandit*. The *global* adjective describes the environment where all the arms $a^{(i)} \in \mathcal{A}$ suffer a change in the mean reward μ_i when a breakpoint occurs. This differs from previous works we discuss, where a change-point indicates that the mean reward of one or more arms has changed. Here, if a breakpoint occurs, it indicates that the mean reward of all arms is altered. To address this problem, the authors propose a solution based on Thompson Sampling (TS) [Agrawal and Goyal, 2012;

Russo et al., 2018], which is called Global Switching TS with Bayesian Aggregation `Global-STS-BA`. The algorithms assumes a Beta prior distribution $\pi^{(Beta)}_{a^{(i)},1} = Beta(\alpha_0, \beta_0)$ where $\alpha_0, \beta_0 > 0$. At every time step, the learner samples $\theta_{a^{(i)},t}$ from each distribution $\pi^{(Beta)}_{a^{(i)},t}$ and selects the arm to play following $a_t = \text{argmax}_{a^{(i)} \in \mathcal{A}} \theta_{a^{(i)},t}$. Once the arm is played and the learner collects the reward, the algorithm updates the posterior distribution, such that

$$\pi^{(Beta)}_{a^{(i)},t} = Beta\Big(\alpha_{a^{(i)},t} = \alpha_0 + \mathbb{I}\{\sum_{s=1}^{t} a_s = a^{(i)}\}\mathbb{I}\{\sum_{s=1}^{t} r_s = 1\},$$
$$\beta_{i,t} = \beta_0 + \mathbb{I}\{\sum_{s=1}^{t} a_s = a^{(i)}\}\mathbb{I}\{\sum_{s=1}^{t} r_s = 0\}\Big).$$

To face the non-stationarity of the model, the authors combine the Thompson Sampling approach to a Bayesian online change-point detector, which works by tracking the optimal expert. Unfortunately, the authors did not provide a regret analysis of the algorithm.

A subsequent work, which goes back to the local abruptly-changing environment, is studied in [Liu et al., 2018]. Here, the authors further assume that:

- the distributions of all arms are Bernoulli distributions,
- the shortest stationary interval between two consecutive change-points is greater than KM, where K is the cardinality of the set of actions $\mathcal{A}$ and $M \in \mathbb{Z}$ is an integer,
- there exists a known parameter $\epsilon > 0$ such that $\forall a^{(i)} \in \mathcal{A}$ and $\forall t \leq T-1$, if $\mu_i(t) \neq \mu_i(t+1)$, then $\|\mu_i(t) - \mu_i(t+1)\| \geq 3\epsilon$. This ensures that when a breakpoint occurs, the shift suffered by the mean reward of the arm involved is not infinitesimal.

Similarly to other works, the authors combine an upper confidence bound strategy with a change-point detection, focusing their efforts on the latter. The upper confidence bound strategy is exactly the UCB algorithm previously mentioned. The change-point detector is used to tell the UCB algorithm when to reset the estimates of a certain arm for which the change-point is detected. The method used to detect the breakpoints is called `Two-sided CUSUM` and it is based on the concept of a random walk. It considers a function of the reward sampled at time t and uses it to make a step in a random walk. If no change in the mean rewards is detected, then the random walk has a negative mean drift. Otherwise, if a change-point is detected, then the random walk has a positive mean drift. The algorithm signals a change in the mean reward of an arm if the random walk crosses a positive threshold. The algorithm is called "two-sided" because it instantiates two random walks with a negative mean drift and a positive mean drift respectively before and after a change-point. This is instantiated for every arm $a^{(i)} \in \mathcal{A}$. This means that there are K `Two-sided CUSUM` algorithms running in parallel to detect changes in each arm. The algorithm that combines UCB with `Two-sided CUSUM` is called `CUSUM-UCB`. The authors show that when the horizon T and the number of breakpoints Υ_T are known and the parameters are appropriately tuned, this algorithm has been proven to reach a regret upper bound of

$$\mathcal{O}\left(\frac{\Upsilon_T \log(T)}{(\Delta_{\mu_i(T)})^2} + \sqrt{T\gamma_T \log(\frac{T}{\Upsilon_T})}\right),$$

where $\Delta_{\mu_i(T)} = \min\{\mu_{a^*}(t) - \mu_{a^{(i)}}(t)\}$ is the minimum difference between the expected reward of the best arm a^* and a suboptimal arm $a^{(i)}$ over all the time slots.

In the context of abruptly-changing environments, [Wei and Srivatsva, 2018] proposed two algorithms, `Limited Memory Deterministic Sequencing of Exploration and Exploitation (LM-DSEE)` and `Sliding-Window Upper Confidence Bound (SW-UCB#)`. With their work, the authors address the same setting of abruptly-changing environments where they assume the mean reward to be bounded $\mu_i(t) \in [0,1]$, as well as the number of breakpoints $\Upsilon_T \in \mathcal{O}(T^\nu)$ where $\nu \in [0,1)$. The first algorithm, called `LM-DSEE`, proposed by the authors to solve this bandit problem consists of alternating between exploration and exploitation phases. The algorithm splits the horizon into epochs. In the i-th exploration phase, it samples each action $L(i) = \lceil \gamma \ln(i^\rho l \beta) \rceil$ times and collects the reward it receives to update the arms' estimates. In the i-th exploitation phase, the algorithm plays for $\lceil \alpha i^\rho l \rceil - KL(i)$ times the action that resulted to be the best in the previous phase, namely the one which reached the highest average reward in the i-th exploration phase. To better understand these quantities, the authors specify how to tune the parameters γ, l, and ρ depending on the environment as:

$$\begin{aligned}&\gamma \geq \frac{2}{\Delta^2_{\text{min}}},\\ &l \in \{K/\alpha\lceil\gamma \ln l\beta\rceil, \ldots, +\infty\},\\ &\rho = \frac{1-\nu}{1+\nu},\end{aligned}$$

where $\nu \in [0,1), \Delta_{min} \in (0,1), \alpha \in \mathbb{R}^+ - \{0\}, \beta \in (0,1]$. This algorithm gets a regret upper bound of $\mathcal{O}(T^{\frac{1+\nu}{2}} \ln T)$ in abruptly-changing environments with the assumption that $\Upsilon_T \in \mathcal{O}(T^\nu)$ with $\nu \in [0,1)$ known a priori.
For the same type of environment and the same assumption on Υ_T, the authors proposed the `SW-UCB#` algorithm, which is an adaptation of the `SW-UCB` algorithm proposed in [Garivier and Moulines, 2011]. It is still based on the concept of using a sliding window to keep track of the arms' estimates. However, while [Garivier and Moulines, 2011] defines a fixed-sized sliding window which depends on the a priori knowledge of T, here [Wei and Srivatsva, 2018] defines the window to be of size $\tau(t,\alpha) = \min\{\lceil\gamma t^\alpha\rceil, t\}$ where $\alpha \in (0,1]$ and $\gamma \in \mathbb{R}^+ \cup \{+\infty\}$. This allows the window to have a time-varying size and to not depend on the knowledge of T. The mean rewards of the arms are computed following Equation (3.5) using only the rewards collected in the sliding window of

size $\tau(t,\alpha)$. The algorithm consists of playing each arm once and then selecting the arm with the highest UCB index similarly as Equation (3.3) with the only difference that here the confidence bonus is defined as $c_i(t,\alpha) = \sqrt{\frac{(1+\alpha)\ln t}{N_t(\tau(t,\alpha),a^{(i)})}}$. Recalling the assumption that $\Upsilon_T \in \mathcal{O}(T^\nu)$ with known $\nu \in [0,1)$, the regret of the SW-UCB# algorithm is upper bounded by $\mathcal{O}(T^{\frac{1+\nu}{2}}\ln(T))$, the same regret obtained by LM-DSEE.

More recently, [Besson and Kaufmann, 2019] proposed a new algorithm which combines approaches based on upper confidence bounds and a change-point detector. The authors study the piece-wise stationary setting characterized by the parameter Υ_T as defined in Equation (3.1). The authors proposed an algorithm called GLR-klUCB, which is the first one where it is not required to know the smallest magnitude of change between two change-points. The peculiarity of this algorithm is that when selecting the arm with the highest UCB index to play, the UCB index is not the one defined in Equation (2.8) but it is an upper confidence bound defined on the Kullbak-Leibler (KL) divergence, which is a measure of similarity between distributions. Typically, the KL-UCB standard algorithm selects the arm to play using the following criterion:

$$\begin{aligned} a_t &= \operatorname*{argmax}_{a^{(i)}\in\mathcal{A}} \max\left\{\widetilde{\mu}\in[0,1] : d(\widehat{\mu}_i(t-1),\widetilde{\mu}) \le \frac{\log(f(t))}{T_{(t-1)}(i)}\right\} \\ \text{where} \quad & f(t) = 1 + t\log^2(t). \end{aligned}$$

The GLR-klUCB algorithm uses this bandit algorithm combined with a change-point detector based on a Generalized Likelihood Ratio test, which detects any change in the reward distribution of any arm and communicates this to the algorithm. The authors provide two different versions of the algorithm: *local restart* and *global restart*. The Local Restart resets the mean reward of the arm for which the change-point has been detected. The Global Restart resets the mean reward for all the arms even if the change-point has been detected only for one of them. In both cases, once the mean reward of an arm is restarted, the algorithm starts recomputing the mean reward from the samples collected after the change-point. We use $\tau(i)$ to indicate the time step where arm $a^{(i)}$ was last restarted. After playing each arm once at the beginning of the game, the algorithm selects the action to play a_t using the KL-divergence and adapting the criterion to:

$$\begin{aligned} a_t &= \operatorname*{argmax}_{a^{(i)}\in\mathcal{A}} \max\{\widetilde{\mu}\in[0,1] : N_t(i)\times d(\widehat{\mu}_i(t-1),\widetilde{\mu}) \le f(t-\tau(i))\} \\ \text{where} \quad & f(t) = \ln(t) + 3\ln(\ln(t)), \end{aligned} \tag{3.6}$$

where $N_t(i)$ is the number of times that action $a^{(i)}$ has been pulled after the last restart. Based on the type of algorithm, either GLR-klUCB with Local Restart or GLR-klUCB with Global restart, the authors prove that with a certain tuning of the parameters, the regret upper bound is $\mathcal{O}(K\sqrt{\Upsilon_T T\ln(T)})$ and $\mathcal{O}(K\sqrt{C_T T\ln(T)})$ respectively, where $C_T \le K\Upsilon_T$ is the total number of change points occurred during the game. When $\Upsilon_T = C_T$, the analysis shows that it is preferable to adopt GLR-klUCB with Local Restart.

The last work we mention on switching bandits is the one by [Auer et al., 2019]. In this work, the authors analyse the local abruptly-changing environment when the number of change-points is unknown. They propose an algorithm called `AdSwitch` which does not require knowing Υ_T in advance. `AdSwitch` works on intervals $l = 1, 2, \ldots$ and starts a new interval when a change in the mean rewards is detected. It starts by considering all arms as *good*. The more the actions are played and the rewards collected, the algorithm labels an arm as *bad* if it is proven to be suboptimal at time t using as a condition a confidence bound computed on the current time interval:

$$\max_{a' \in good_t} \widehat{\mu}_{[s,t],a'} - \widehat{\mu}_{[s,t],a} > \sqrt{\frac{C_1 \log(T)}{n_{[s,t]}(a) - 1}}, \tag{3.7}$$

where s is the time index used inside the interval, C_1 is a suitable constant, and $n_{[s,t]}(a)$ indicates the number of times the action a has been played inside the current interval. Similarly, $\widehat{\mu}_{[s,t],a}$ indicates the mean reward of arm a computed using the rewards collected in the current interval. If an arm a satisfies the condition and is labelled as *bad*, then the mean reward of arm a and its gap to arm a' in Equation (3.7) are saved and arm a is removed from the set of *good arms*. The algorithm also checks for changes in the *good arms*, by using the condition:

$$|\widehat{\mu}_{[s1,s2],a} - \widehat{\mu}_{[s,t],a}| > \sqrt{\frac{2\log(T)}{n_{[s1,s2]}(a)}} + \sqrt{\frac{2\log(T)}{n_{[s,t]}(a)}}, \tag{3.8}$$

where s is in the current interval and $s1 \leq s2$. The algorithm also forces the examination of *bad* arms following some sampling obligations in order to select this type of arms only rarely to avoid an impact on the regret. The condition used in this case is the following:

$$|\widehat{\mu}_{[s1,s2],a} - \widetilde{\mu}_a(l)| > \frac{\widetilde{\Delta}_l(a)}{4} + \sqrt{\frac{2\log(T)}{n_{[s,t]}(a)}}, \tag{3.9}$$

where l is the index of the interval where the arm was evicted from the *good arms*, $\widetilde{\mu}_a(l)$ is the mean reward the arm had, and $\widetilde{\Delta}_l(a) = \max_{a' \in good_t} \widehat{\mu}_{[s,t],a'} - \widehat{\mu}_{[s,t],a}$ the gap between arm a and arm a' when it was evicted. At time step t, `AdSwitch` selects the arm in the set of *good arms* that has been played least recently, selecting *good arms* in a round-robin fashion. Sometimes, following the sampling obligations, the algorithm plays some *bad arms* to check if their mean reward has changed using Equation (3.9). When a change in mean rewards is detected, the algorithm starts a new interval and proceeds to examine the same conditions. The authors analyzed the regret of this algorithm and proved that its regret is upper bounded by $\mathcal{O}(C\sqrt{K\Upsilon_T T \log(T)})$, where C is a suitable constant and there is no knowledge of Υ_T in advance.

While the previous works focused only on settings with finite sets of actions, the next one studies linear d-dimensional environments.
In [Wei and Srivatsva, 2018] the authors propose `LM-DSEE` and `SW-UCB#` not

only for the abruptly-changing environment but also for the *slowly-varying* environment. In this setting, the mean reward of an arm measured in two subsequent time steps suffers a small shift upper bounded by $\epsilon_T \in \mathcal{O}(T^{-m})$ with $m \in \mathbb{R}^+ - \{0\}$ known a priori. We note how for m close to zero, the changes in the mean rewards become higher. For this environment, the authors proposed the exact same algorithms proposed for the abruptly-changing environment, but the specificity of this setting influences the regret bounds. Assuming $\epsilon = \mathcal{O}(T^{-m})$, the `LM-DSEE` algorithm obtains a regret bound of $\mathcal{O}(T^{\frac{3+2\rho}{3+3\rho}} \ln T)$ for $\rho = \frac{3\min\{m, m_{max}\}}{4-3\min\{m,m_{max}\}}$ and $m_{max} \in (0, 4/3)$, while the `SW-UCB#` algorithm is upper bounded by $\mathcal{O}(T^{1-\frac{\alpha}{3}} \ln T)$ for $\alpha = \min\{1, \frac{3m}{4}\}$.

3.1.2 Non-stationary bandits with variation budget

In the class of exogenous bandits, where the non-stationarity does not depend on the actions of the learner, we include another subset called bandits with *variation budget*, or *variation bound*. This family of bandits is still characterized by a horizon T and a set of actions $\mathcal{A}$, which we assumed to have finite cardinality K for the moment. The peculiarity of this model is the assumption that the reward of each arm can change at any point $t \in \{1, \dots, T\}$ but that throughout the horizon the total variation of the expected reward of an arm is bounded by the following quantity, called variation budget

$$\sum_{t=1}^{T-1} \sup_{a^{(i)} \in \mathcal{A}} \|\mu_i(t) - \mu_i(t+1)\| \leq V_T. \tag{3.10}$$

The variation budget indicates how much the mean reward of an arm can change between two consecutive time steps. A particular aspect of this setting is that it includes both abruptly-changing and slowly-changing environments. In [Besbes et al., 2014] and [Besbes et al., 2019], the authors present a bandit problem with *variation budget* as the one defined. The authors study the setting where the variation budget V_T is known and provide a lower bound for the setting where $T \geq 1, K \geq 2$, and $V_T \in [K^{-1}, K^{-1}T]$, stating that $R_T \geq C(KV_T)^{1/3}T^{2/3}$ where $C > 0$ is an absolute constant. The authors present an algorithm called `Rexp3` as a solution to this bandit problem. The algorithm sets a batch size Δ_T and divides the horizon in T/Δ_T epochs. In each one, it employs a new instance of a subroutine based on the `Exp3` algorithm. They show that for $T \geq 1, K \geq 2, V_T \in [K^{-1}, K^{-1}T]$ and setting the batch size as $\Delta_T = \lceil (K \log K)^{1/3}(T/V_T)^{2/3} \rceil$, `Rexp3` obtains a regret upper bound of $\bar{C}(K \log KV_T)^{1/3}T^{2/3}$ for some absolute constant $\bar{C}$ and by appropriately tuning the parameters.

In [Cheung et al., 2019] the authors analyze the bandit with variation budget problem in a linear setting of dimension d. They study a linear bandit problem as presented in Section 2.5 for the non-stationary setting. They assume that the reward function is the inner product between the action $a_t \in D_t \subseteq \mathbb{R}^d$ where D_t is a decision set chosen by an oblivious adversary, and the unknown parameters vector θ_t^*. The subscript indicates that the unknown parameter, which dictates the best action the learner could pull, changes with every time step. In this setting, the authors introduced the variation budget as

$$\sum_{t=1}^{T-1} \|\theta^*_{t+1} - \theta^*_t\| \leq V_T, \tag{3.11}$$

assuming that $V_T = \Theta(T^\rho)$ where $\rho \in (0,1)$. However, they differ from the previous work since they allow the adversary to choose the different θ^*_t. To face this bandit problem, the authors present an algorithm called `SW-UCB` where they combine the use of UCB index with a sliding window. The main step of `SW-UCB` consists of selecting in each time step the action $a_t \in D_t$ which maximizes the UCB index defined as:

$$a_t = \operatorname*{argmax}_{a \in D_t} \left\{ a^\top \widehat{\theta}_t + \|a\|_{V^{-1}_{t-1}} \left[R\sqrt{d \ln\left(\frac{1 + wL^2/\lambda}{\delta}\right)} + \sqrt{\lambda} S \right] \right\}, \tag{3.12}$$

which is the solution to the optimization problem where $\widehat{\theta}_t$ is the estimation of θ_t using the regularized least squares estimator (Equation (2.24) in Section 2.5), w is the window size, R is the variance proxy of the noise terms, and L is the upper bound of all the actions' l_2 norms s.t. $\|a\| \leq L$. Adopting this algorithm, the learner is able to achieve a regret bound of $\mathcal{O}(d^{2/3} V_T^{1/3} T^{2/3})$, when V_T is known and $w = \mathcal{O}((dT)^{2/3} V_T^{-2/3})$. To overcome the knowledge of V_T, the authors propose a `Bandit-over-bandit (BOB)` algorithm. It consists of dividing the horizon into $\lceil T/H \rceil$ blocks of length H. In each block, a window size w_i is drawn from a set $J \subseteq \{1, \ldots, H\}$ and used to run an instance of `SW-UCB` with window size $w = w_i$. After H rounds, once the block is over, the learner sends the average of the rewards collected in the block as the reward to a meta-bandit, which is an instance of the EXP3 algorithm to control the selection of the window size w_i. In this setting, where V_T is unknown, `BOB` achieves a regret of $\widetilde{\mathcal{O}}(d^{2/3}(V_T + 1)^{1/4} T^{3/4})$.

Another work which analyses a bandit problem with variation budget is [Russac et al., 2019]. Here, the authors consider a linear setting as well. The reward function is defined as $X_t = \langle a_t, \theta^*_t \rangle$, where θ^*_t is the unknown parameter which changes with every time step. The authors assume that every action is bounded by $\|a\|_2 \leq L$, the unknown parameter is bounded by $\forall t, \|\theta^*_t\|_2 \leq S$ in each time step, and the reward is also bounded by $\langle a_t, \theta^*_t \rangle \leq 1$. The authors propose a weighted regularized least-squares estimator, which is a modified version compared to the one present in Equation (2.24) and defined as follows:

$$\widehat{\theta}_t = \operatorname*{argmin}_{\theta \in \mathbb{R}^d} \left(\sum_{s=1}^{t} w_s (X_s - \langle \theta, a_s \rangle)^2 + \lambda_t \|\theta\|_2^2 \right), \tag{3.13}$$

where $\forall s \in \{1, \ldots, t\}, w_s$ are positive weights. Developing the RLSE with weights, we get the following equations:

$$\widehat{\theta}_t = V_t^{-1} \sum_{s=1}^{t} w_s a_s X_s, \tag{3.14}$$

where

$$V_0 = \lambda_t I, \quad V_t = V_0 + \sum_{s=1}^{t} w_s a_s a_s^\top, \tag{3.15}$$

where $V_t \in \mathbb{R}^{d \times d}$ and I is the identity matrix. They also introduce the matrix:

$$\widetilde{V}_t = \mu(t) I_d + \sum_{s=1}^{t} w_s^2 a_s a_s^\top, \tag{3.16}$$

where $\forall t, \mu(t)$ are positive parameters and $\widetilde{V}_t$ is related to the variance of $\widehat{\theta}_t$. These equations are used to solve the bandit problem, following the `D-LinUCB` algorithm and setting $w_{t,s} = \gamma^{t-s}$, where γ is interpreted as a discount factor. The algorithm plays the action which maximizes the UCB index, which is defined as:

$$UCB(a) = a^\top \widehat{\theta} + \beta_{t-1} \sqrt{a^\top V^{-1} \widetilde{V} V^{-1} a}, \tag{3.17}$$

where $\beta_{t-1} = \sqrt{\lambda} S + \sigma \sqrt{2 \log(1/\delta) + d \log\left(1 + \frac{L^2(1-\gamma^{2t})}{\lambda d (1-\gamma^2)}\right)}$ defines the confidence set $C_t = \{\theta : \|\theta - \widehat{\theta}_{t-1}\|_{V_{t-1}\widetilde{V}_{t-1}^{-1}V_{t-1}} \leq \beta_{t-1}\}$, using $\mu(t) = \gamma^{-2t}$. After selecting the action, the algorithm uses the reward to update the following quantities:

$$\begin{aligned}
&V = \gamma V + a_t a_t^\top + (1-\gamma)\lambda I \\
&\widetilde{V} = \gamma^2 \widetilde{V} + a_t a_t^\top + (1-\gamma^2)\lambda I \\
&\widehat{\theta} = V^{-1} b \\
&b = \gamma b + X_t a_t.
\end{aligned}$$

The authors analysed the performance of the algorithm assuming a variation budget $\sum_{s=1}^{T-1} \|\theta_s^* - \theta_{s+1}^*\|_2 \leq V_T$ and proved that, when $\gamma = 1 - (\frac{V_T}{dT})^{2/3}$ and $T \to \infty$, with high probability `D-LinUCB` reaches an asymptotic regret upper bound of $\mathcal{O}(d^{2/3} V_T^{1/3} T^{2/3})$. Note that this result requires T to be known in order to tune the discount factor γ.

3.1.3 Rotting restless bandits

Finally, we cite the work done by [Seznec et al., 2020] who studied both the switching bandits and bandits with variation budget only when the rewards are arbitrary non-increasing functions which depend on time or number of pulls. For both settings, they define rotting restless bandits as a class of bandits with a finite set of arms $|\mathcal{A}| = K$ where each arm $a^{(i)}$ is associated with a reward function $\mu_i(t)$ which depends on time t. The environment is based on the assumption that $\forall a^{(i)} \in \mathcal{A}$ the reward functions $\mu_i(t)$ are non-increasing. In this work and in the ones which will follow in the next sections, we will refer to this last assumption with the terms *rotting*, other than *decreasing* or *decaying*. Another characteristic of this setting is the assumption of a variation budget on the mean rewards functions $\mu_i : \mathbb{N}^\star \to [-V_T, 0]$:

$$\sum_{t=1}^{T} \sup_{a^{(i)} \in \mathcal{A}} \Big(\mu_i(t) - \mu_i(t+1) \Big) \leq V_T, \tag{3.18}$$

where the variation budget V_T is a positive constant. The authors show that adapting the proof in [Besbes et al., 2014], under these assumptions and with a variation budget $V_T \geq \sigma\sqrt{K/(8T)}$, the expected regret of any strategy would be lower bounded by:

$$R_T \geq \frac{1}{16\sqrt{2}} (\sigma^2 V_T K T^2)^{1/3}. \tag{3.19}$$

The algorithm proposed by authors to solve this bandit problem is called `Rotting Adaptive Window Upper Confidence Bound (Raw-UCB)`. It is built upon the proof that the greedy policy, which is the strategy that plays the arm $a_t = \text{argmax}_{a^{(i)} \in \mathcal{A}} \mu_i(t)$ that maximizes the reward at current time t, is optimal in the rotting restless setting. They defined the following quantities:

$$\begin{aligned} \widehat{\mu}_i^m(t) &= \frac{1}{m} \sum_{s=1}^{t-1} \mathbb{I}\{a_s = a^{(i)} \wedge N_{i,s} > N_{i,t-1} - m\} X_s \\ \bar{\mu}_t^m(i) &= \frac{1}{m} \sum_{s=1}^{t-1} \mathbb{I}\{a_s = a^{(i)} \wedge N_{i,s} > N_{i,t-1} - m\} \mu_i((s, N_{i,s-1})), \end{aligned} \tag{3.20}$$

where $N_{i,t}$ indicates the number of times action $a^{(i)}$ has been pulled up to time t. The quantity $\widehat{\mu}_i^m(t)$ defines the mean reward of arm $a^{(i)}$ computed on its last m observations at time t, while $\bar{\mu}_i^m(t)$ defines the average of the associated means. They use these quantities to define the event under which the confidence bound is true:

$$\forall t, \delta = 2t^{-\alpha} \quad \xi_t^\alpha = \{\forall a^{(i)} \in \mathcal{A}, \forall n \leq t-1, \forall h \leq n, \quad |\widehat{\mu}_i^m(t) - \bar{\mu}_t^m(i)| \leq c(m, \delta_t)\} \tag{3.21}$$

where $c(m, \delta_t) = \sqrt{\frac{2\sigma^2 \log(2/\delta_t)}{m}}$ and $\forall t > K$, after pulling each arm once, $\mathbb{P}[\bar{\xi_t^\alpha}] \leq Kt^{2-\alpha}$. After defining these quantities, it is easier to define the steps of the algorithm. After pulling each arm once, `Raw-UCB` selects the arm which maximizes the following variant of the UCB index:

$$UCB_{Raw,i} = \min_{h \leq N_{i,t-1}} \widehat{\mu}_i^m(t) + c(m, \delta_t), \tag{3.22}$$

where $\delta_t = 2/t^\alpha$. The algorithm selects in an adaptive way the window to compute the tightest UCB_{Raw} index for each arm, since the rotting behaviour makes sure that this index is an upper confidence bound on the reward in the next step. The authors show that their proposed algorithm matches the lower bound in Equation (3.19) up to poly-logarithmic factors and without the knowledge of T nor V_T:

$$R_T = 4\Big(C^2\sigma^2 V_T K T^2 \log(T)\Big)^{1/3} + \widetilde{\mathcal{O}}\Big((\sigma V_T^2 K^2 T)^{1/3}\Big). \tag{3.23}$$

In the very same work, the authors investigate the piece-wise stationary restless bandits as well. In this particular setting, they still assume that the reward functions are bounded as defined in Equation (3.18), but they add another assumption, which defines the piece-wise stationarity of the model:

$$\Upsilon_T - 1 \geq \sum_{t=1}^{T-1} \mathbb{I}\{\exists a^{(i)} \in \mathcal{A} : \mu_i(t) \neq \mu_i(t+1)\}, \tag{3.24}$$

where Υ_T is a positive integer and the assumption states that there are at most $\Upsilon_T - 1$ change-points throughout the horizon T. The authors propose the same algorithm `Raw-UCB` to solve this problem and prove that in this particular piece-wise stationary setting the regret upper bound of `Raw-UCB` is:

$$R_T \leq C\sigma\sqrt{\log(T)}\Big(\sqrt{\Upsilon_T K T} + \Upsilon_T K\Big) + 6KV_T, \tag{3.25}$$

which matches its lower bound up to poly-logarithmic factors and without the knowledge of T or $\Upsilon_T - 1$, where the expected regret lower bound for $\Upsilon_T \leq \left(\frac{32V_T^2 T}{K\sigma^2}\right)$ is:

$$R_T \geq \frac{\sigma}{32}\sqrt{\Upsilon_T K T}. \tag{3.26}$$

3.1.4 Rising restless bandits

Finally, we conclude the discussion on exogenous non-stationarity presenting rising bandits by [Metelli et al., 2022], which does not fit neither into the piece-wise stationary nor in the variation budget bandits. The work published in the cited paper addresses both the restless and rested setting. We defer rising rested bandits to the next chapter and we explore here the rising restless bandit model. Each action $a^{(i)} \in \mathcal{A}$ with $|\mathcal{A}| = K$ is characterized by a reward function $\mu_i(t)$ which is non-decreasing and concave. The monotonicity is defined as $\gamma_i = \mu_i(n+1) - \mu_i(n) \geq 0$ and the concavity as $\gamma_i(n+1) - \gamma_i(n) \leq 0$. They characterize the problem by introducing the definition of *cumulative increment*:

$$\Gamma_\mu(s, q) = \max_{a^{(i)} \in \mathcal{A}} \left\{ \sum_{l=1}^{s-1} \gamma_i(l)^q \right\} \tag{3.27}$$

for $q \in [0, 1]$ and $s \in \{1, \dots, T\}$. In this setting, the optimal policy is oracle greedy, which at every time step t selects the arm with the highest expected reward. They start by analyzing the deterministic setting, for which they propose an algorithm that at every time step t selects the arm that maximizes the exploration index $B_i(t)$ which is set equal to:

$$\bar{\mu}_t^{R-less}(i) = \mu_i(t_{i,N_{i,t-1}}) + (t - t_{i,N_{i,t-1}})\frac{\mu_i(t_{i,N_{i,t-1}}) - \mu_i(t_{i,N_{i,t-1}-1})}{t_{i,N_{i,t-1}} - t_{i,N_{i,t-1}-1}}, \tag{3.28}$$

where $\mu_i(t_{i,N_{i,t-1}})$ represents the most recent reward and $\frac{\mu_i(t_{i,N_{i,t-1}}) - \mu_i(t_{i,N_{i,t-1}-1})}{t_{i,N_{i,t-1}} - t_{i,N_{i,t-1}-1}}$ the most recent increment. Using this algorithm in the deterministic setting, for every $q \in [0,1]$ the algorithm reaches a regret bound of:

$$R_T \leq 2K + KT^{\frac{q}{q+1}}\Gamma_\mu\left(\left\lceil\frac{T}{K}\right\rceil, q\right)^{\frac{1}{q+1}}. \tag{3.29}$$

From this result, the authors move to the stochastic setting and propose a solution based on a d-sized window with $d \in [N_{i,t-1}]$ where the size d governs the bias-variance trade-off between using a few recent payoffs and many past rewards. In the algorithm, they propose they define:

$$\widehat{\mu}_i^{R-less}(t) = \frac{1}{d}\sum_{n=N_{i,t-1}-d+1}^{N_{i,t-1}}\left(\widehat{\mu}_{i,n}(t) + (t-n)\frac{\widehat{\mu}_{i,n}(t) - \widehat{\mu}_{i,n-d}(t)}{d}\right), \tag{3.30}$$

and adapt the exploration index to be $B_i(t) = \widehat{\mu}_{i,d}^{R-less}(t) + \beta_{i,d}^{R-less}(t)$ where $\beta_{i,d}^{R-less}(t,\delta_t) = \sigma(t - N_{i,t-1} + d_{i,t} - 1)\sqrt{\frac{10\log(1/\delta_t)}{d_{i,t}^3}}$. So the algorithm selects in each time step the arm which maximizes the index $B_i(t)$.
The authors show that for $q \in [0,1]$, $d_{i,t} = \lfloor \varepsilon N_{i,t-1} \rfloor$ for $\varepsilon \in (0, 1/2)$, $\delta_t = t^{-\alpha}$ for $\alpha > 2$, the algorithm achieves an expected regret bound of:

$$R_T = \mathcal{O}\left(\frac{K}{\varepsilon}(\sigma T)^{2/3}(\alpha\log(T))^{1/3} + \frac{KT^{\frac{2q}{1+q}}}{\varepsilon(1-2\varepsilon)}\Gamma_\mu\left(\left\lceil(1-2\varepsilon)\frac{T}{K}\right\rceil, q\right)^{\frac{1}{1+q}}\right). \tag{3.31}$$

3.2 Endogenous non-stationarity: rested and state-dependent bandits

In the previous sections, we investigated different classes of restless bandits, all united by the fact that their non-stationarity is intrinsically related to the environment. In this section, we depart from the family of restless bandits to focus on the class of bandits with endogenous non-stationarity. The peculiarity of this family is that the non-stationarity depends on the previous history of actions. Here, the learner can directly influence the rewards of future actions based on the choices he makes from the start of the game. Different models in the literature study different effects of the history of actions on the rewards. As mentioned at the beginning of this chapter, we distinguish between *rested* and *state-dependent bandits*. *Rested bandits* identify a class of bandits where the expected reward of an arm changes only when this arm has been pulled. In this case, the non-stationarity of an arm's distribution at the current time step

t depends on the number of times the arm has been pulled up to time t. Inside this class, we are going to analyze models that rely on different assumptions: we mainly distinguish between a *rotting*, or *decaying*, and *rising*, or *increasing*, behaviour. In *rotting* bandits, the expected reward of the arm decreases with the number of pulls, while in *rising* bandits this reward increases with the number of pulls. We will detail each model when we discuss it.
On the other hand, we classify as *state-dependent bandits* all those bandit models where the non-stationarity of the distribution of an arm depends on the time elapsed since it was last pulled or how many times the arm has been consecutively pulled. We use the term *state-dependent* since these models oftentimes rely on the concept of *state*, or *delay*, to keep track of how many time steps have passed since the arm was last pulled or how many times it has been consecutively pulled. Inside this class, we are also able to distinguish between bandits with *decreasing* and *increasing* reward functions, recalling the dichotomy between *rotting* and *rising* bandits that we found for restless and rested bandits. However, as we will see in the rest of this section, we will switch to the term *recovering*, or *recharging*, when we refer to those bandits whose reward function is increasing with the state but will go back to zero once the action is played. The different ways in which the rewards are affected by these behaviours are going to be specified for each model we are going to present. We will make every distinction clear when presenting each work.

3.2.1 Rested bandits

Rotting rested bandits

We start by presenting those works in the literature that focus on *rotting*, or *decaying, decreasing*, behaviours, which are namely those bandits characterized by non-increasing or decreasing reward functions. We consider all those bandit models where the rewards of the actions tend to decrease the more they are played.
We start by introducing rotting bandits, which is one of the prominent works in this class. They were first introduced in [Gittins, 1979], [Gittins et al., 2011] and further studied in [Heidari et al., 2016]. In the latter, the authors define the rotting bandit problem as a multiarmed bandit with a finite set of actions of cardinality K where each action $a^{(i)}$ is associated with an arm-dependent reward function $f_i : \mathbb{N}^{\geq 1} \to [0, 1]$, which receives as input the number of times $T_i(t)$ the action $a^{(i)}$ has been played up to time t and returns the reward $f_i(T_i(t))$ obtained by the arm at time t, which is bounded in $[0, 1]$. We emphasize how the cumulative reward of a policy does not depend on the specific order of the pulls but only on the number of pulls of each arm. This is an important distinction between *rested* and *state-dependent bandits*, where the performance of the latter depends on the specific order of actions pulled by the learner. The authors study both increasing and decreasing reward functions for rested bandits. Here, we analyze the decreasing setting, where the reward function is decreasing, and defer the discussion about the increasing setting in the next section. For the offline setting of this problem, they prove that oracle greedy, which at every round t pulls the arm with the highest instantaneous reward, is

the optimal policy. The authors adapt this algorithm to the online setting by proposing a greedy policy that pulls each arm once and then selects the arm with the highest most recent instantaneous reward. Since the reward functions are decreasing, the player is sure that the next pull of any arm $a^{(i)}$ will not entail a higher reward than its last one. This algorithm shows a constant regret of $K + \varepsilon T$, where $\varepsilon > 0$ is a bound on the magnitude of the corruption of the rewards.

The same rotting setting was also studied in [Levine et al., 2017], where the reward function of any arm is decreasing and depends on the number of pulls. As in the work of [Heidari et al., 2016], the authors prove that the optimal policy for the rotting bandit problem is greedy, which selects in each time step t the action which maximizes the most recent instantaneous reward, $a_t^* = \text{argmax}_{a^{(i)} \in \mathcal{A}} \{\mu_i(T_i(t) + 1)\}$. The novelty of this work is to propose two different models: a non-parametric and a parametric rotting bandit. In the non-parametric case, the only assumption is that the reward functions are decreasing, with no prior knowledge of the expected rewards. For this setting, they propose a `Sliding-Window Average (SWA)` algorithm. In the beginning, the player pulls each arm m times in a round-robin fashion. Then, he selects the arm $a_t = \text{argmax}_{a^{(i)} \in \mathcal{A}} \left\{ \frac{1}{m} \sum_{n=T_i(t)-m+1}^{T_i(t)} X_{i,n} \right\}$ where $X_{i,n}$ is the reward collected by the learner when playing arm $a^{(i)}$ at its n-th pull. Once the arm is played, the algorithm increases by 1 the number of pulls of the selected arm and stores $X_{i,t}$ as its last reward. The algorithm follows a similar idea compared to the previous work, but instead of selecting the arm that maximizes the most recent instantaneous reward, the learner adapts it so that the condition does not rely simply on the last reward but on the average of its last m rewards. `SWA` achieves a regret upper bound of:

$$R_T \leq \left(\alpha \max_{a^{(i)} \in \mathcal{A}} \mu_i(1) + \alpha^{-1/2} \right) 4^{2/3} \alpha^{2/3} K^{1/3} T^{2/3} \ln^{1/3}(\sqrt{2}T) + 3K \max_{a^{(i)} \in \mathcal{A}} \mu_i(1) \tag{3.32}$$

where $\alpha > 0$ is used to tune $m = \lceil \alpha 4^{2/3} \alpha^{2/3} K^{-2/3} T^{2/3} \ln^{1/3}(\sqrt{2}T) \rceil$ and the upper bound is minimized when $\alpha = (2 \max_{a^{(i)} \in \mathcal{A}} \mu_i(1))^{-2/3}$. This is the bound for the case where the horizon is known. In case it is not, the authors show that, by using the doubling trick, the same regret is worsened only by a $\log_2(T)$ factor.

On the other hand, the same work analyzes the parametric case where it is assumed that the player knows that the reward function is the product of the sum of an unknown constant part and a rotting part belonging to a set of models. So, the reward function can be written as $\mu(n) = \mu_i^c + \mu(n, \theta_i^*)$, where the collection of $\{\theta_i^*\}_{i=1}^K$ constitutes the set of models Θ^*. The parametric setting can be further split into two cases: the *asymptotically vanishing case*, where $\forall a^{(i)} : \mu^c(i) = 0$, and the *asymptotically non-vanishing case*, where $\forall a^{(i)} : \mu^c(i) \in \mathbb{R}$. For both cases, the setting relies on the assumption that $\forall a^{(i)}, \theta \in \Theta \quad \mu_i(n, \theta)$ is positive, non-increasing in n, and $\mu_i(n, \theta) \in \wr(1)$, $\forall \theta \in \Theta$ where Θ is a discrete known set. In the vanishing case, the authors propose an algorithm called `Closest to Origin (CTO)`. In this algorithm, after playing each arm once, the learner computes an estimate $\widehat{\theta}_i$ from the detection of the true rotting model. This detection is performed using the *proximity to origin* rule, which looks at what model best fits the rewards collected in the past. This is done by defining:

$$Y(i,t,\Theta) = \left\{ \sum_{n=1}^{T_i(t)} X_{i,n} - \sum_{n=1}^{T_i(t)} \mu(n,\theta) \right\}_{\theta\in\Theta} \tag{3.33}$$

and computing the estimate as:

$$\widehat{\theta}_i(t) = \operatorname*{argmin}_{\theta\in\Theta}\{|Y(i,t,\theta)|\}. \tag{3.34}$$

Then, the algorithm uses this estimate to select greedily in each time step t the arm $a_t = \mathrm{argmax}_{a^{(i)}\in\mathcal{A}}\, \mu(T_i(t)+1, \widehat{\theta}_i)$ which maximizes the most recent reward and uses the new reward obtained by playing a_t to update the estimates. The authors show that this approach still achieves a regret upper bound of $o(1)$.
For the asymptotically non-vanishing case, the authors propose a similar method, which they call `Differences Closest to Origin (D-CTO)`. This algorithm implements the detection of both the rotting model and the constant term. The detection of the rotting model is similar to the one performed in the `CTO` algorithm but, instead of minimizing $|Y(i,t,\theta)|$ in Equation (3.34), here the algorithm minimizes $|Z(i,t,\theta)|$, which is the norm of the following function:

$$\begin{aligned} Z(i,t,\Theta) = &\Big(\sum_{n=1}^{\lfloor T_i(t)/2\rfloor} X_{i,n} - \sum_{n=\lfloor T_i(t)/2\rfloor+1}^{T_i(t)} X_{i,n} \Big) \\ &- \Big(\sum_{n=1}^{\lfloor T_i(t)/2\rfloor} \mu(n,\theta) - \sum_{n=\lfloor T_i(t)/2\rfloor+1}^{T_i(t)} \mu(n,\theta) \Big). \end{aligned} \tag{3.35}$$

The idea is to apply the same concept of proximity to origin but on the differences between the two halves of the sequences of rewards for both parts of the reward function. On these definitions, the authors build the `D-CTO` algorithm which plays every action for a certain optimized amount of time m^*_{diff}in a round-robin fashion, uses the criterion of the proximity rule with $Z(i,t,\theta)$ to detect $\widehat{\theta}_i(t)$, and then selects the arm to play using the following UCB criterion:

$$a_t = \operatorname*{argmax}_{a^{(i)}\in\mathcal{A}} \left[\widehat{\mu}^c_i(t) + \mu_(T_i(t)+1, \widehat{\theta}_i(t)) + c_{t,T_i(t)} \right] \tag{3.36}$$

where

$$\widehat{\mu}^c_i(t) = \frac{\sum_{n=1}^{T_i(t)} \Big(X_{i,n} - \mu(n, \widehat{\theta}_i(t)) \Big)}{T_i(t)} \qquad \text{and} \qquad c_{t,s} = \sqrt{\frac{8\ln(t)\sigma^2}{s}}. \tag{3.37}$$

In order to better understand the regret results, we define the following notation: given a function $f : \mathbb{N} \to \mathbb{R}$, a new function $f^{\star\downarrow} : \mathbb{R} \to \mathbb{N}\bigcup\{\infty\}$ is defined as the function which takes $\zeta \in \mathbb{R}$ as an argument and returns the smallest natural number $N \in \mathbb{N}$ such that $\forall n \leq N : f(n) \leq \zeta$ if N exists, otherwise $f(n) \leq \infty$.

The regret of the D-CTO algorithm has been proven to be upper bounded with probability $1-\delta$ by:

$$R_t \leq \sum_{a^{(i)} \in \mathcal{A}: a^{(i)} \neq a^*} \left[\max\left\{ m^*_{diff}(\delta/K), \mu^{\star\downarrow}(\varepsilon_i, \theta^*_i), \frac{32\sigma^2 \ln(T)}{(\Delta_i - \varepsilon_i)^2} \right\} \times (\Delta_i + \mu(1, \theta^*_{a^*})) \right] + C(\Theta^*, \{\mu^c(i)\}) \tag{3.38}$$

for any sequence $\varepsilon_i \in (0, \Delta_i)$ where $\Delta_i = \mu^c(a^*) - \mu^c(i)$ is the suboptimal gap on the constant terms, and $C(\Theta^*, \{\mu^c(i)\}) = \sum_{a^{(i)} \neq a^*} \sum_{n=1}^{\mu^{\star\downarrow}(\Delta_i, \theta^*_i)} \mu_i(n) + \sum_{a^{(i)} \neq a^*} \frac{\pi^2+3}{3}(\Delta_i + \mu(1, \theta^*_{a^*}))$.

Another result in the literature which analyzes decaying bandits is [Bouneffouf and Féraud, 2016]. The environment consisted of a K-armed bandit with rewards bounded in $[0,1]$ and a non-stationary reward function defined as $f_i(t) = X_t \cdot D(T_i(t))$, where X_t is the stationary reward associated with action $a^{(i)}$ at time t with expected mean reward μ_i, $T_i(t)$ the number of pulls associated with arm $a^{(i)}$ up to time t, and $D(T_i(t))$ is the known trend reward function which depends on the number of pulls. They measure the regret against an optimal policy defined as the strategy which always selects arm $a^* = \text{argmax}_{a^{(i)} \in \mathcal{A}} \{\mu_i D(T_i(t))\}$. The authors propose the algorithm A-UCB where the player selects at each time t action $a_t = \text{argmax}_{a^{(i)} \in \mathcal{A}} (\widehat{\mu}_i + c(i)) \cdot D(T_i(t))$, where $\widehat{\mu}_i$ is the empirical average reward of arm $a^{(i)}$ and the confidence bonus is $c(i) = \sqrt{(2\log(t))/T_i(t)}$. In the paper it is proved that this strategy achieves an expected regret upper bound of:

$$R_T \leq \max_{a^{(i)} \in \mathcal{A}} \mu_i D_{max} \sum_{a^{(i)}: T^*_i(T) > T_i(t)} \frac{8\ln(T)}{\Delta'^2_i} + K\frac{\pi^2}{3}, \tag{3.39}$$

where $\Delta'_i = (D_{min}/D_{max})\mu^* - \mu_i$, and D_{min}, D_{max} are two Lipschitz constants.

Rotting bandits are also studied in [Seznec et al., 2019], where the authors investigate a setting comparable to the non-parametric case presented in [Levine et al., 2017]. The environment is characterized by a finite set of actions $|\mathcal{A}| = K$ and by reward functions which decay with the number of pulls. Specifically, when at time t the learner plays an arm $a^{(i)}$, he gets a noisy σ^2-sub-Gaussian reward with mean $\mu_i(T_i(t))$, where $T_i(t)$ is the number of times arm $a^{(i)}$ has been pulled up to time t. It is necessary to clarify a small distinction between this model and the one presented in [Levine et al., 2017]: while in the latter $\mu_i(n)$ indicated the expected mean reward of arm $a^{(i)}$ for the n-th pull, here in [Seznec et al., 2019] $\mu_i(n)$ indicates the expected mean reward of arm $a^{(i)}$ *after* n pulls. Due to the rotting nature of this setting, the reward function $\mu_i : \mathbb{N}^+ \to \mathbb{R}$ are assumed to be non-increasing. The peculiarity of this environment is the assumption that the decay of these reward functions μ_i is bounded as $-L \leq \mu_i(n+1) - \mu_i(n) \leq 0$ with $\mu_i(0) \in [0, L]$. Firstly, they prove a lower bound for the deterministic ($\sigma = 0$) rotting setting (as the one in [Heidari et al., 2016]) with bounded decay equal to $L(K-1)$. Then, they address this bandit problem by proposing the `Filtering on Expanding Window Average (FEWA)` algorithm. This algorithm shares the idea behind previous methods for which the last reward(s) collected by an arm is the most informative. However, considering only one or a few samples is producing estimates with higher variance. In order to balance this trade-off, `FEWA` employs sliding windows of increasing length in order to discard suboptimal arms. Specifically, the algorithm starts with a window size equal to $m = 1$. In each round it computes a threshold equal to $c(m, \sigma, \delta_t) = \sqrt{(2\sigma^2/m)\log(1/\delta_t)}$. An arm $a^{(i)}$ is discarded if the difference between the empirical average of the reward of arm $a^{(i)}$ and the best empirical average among all the active arms is greater than $2c(m, \sigma, \delta_t)$. Once suboptimal arms are filtered out, the algorithm plays the arm $a_t = \operatorname{argmin}_{a^{(i)} \in \mathcal{A}_t} T_i(t)$ which has been pulled the least among the remaining arms $\mathcal{A}_t$. Finally, the reward obtained by the arm is used to update the statistics. The authors show that in this setting `FEWA` gets an expected regret upper bound of

$$R_T \leq 13\sigma(\sqrt{KT} + K)\sqrt{\log(KT)} + KL.$$

The authors show that applying `FEWA` to the rotting bandit setting studied in [Levine et al., 2017] would reach a regret upper bound of $\widetilde{\mathcal{O}}(\sqrt{KT})$, an improvement compared to the regret of `SWA` which achieved an upper bound of $\widetilde{\mathcal{O}}(\mu_{max}^{1/3} K^{1/3} T^{2/3})$. The authors also studied the problem-dependent regret. They first defined:

$$\begin{aligned}
&m_{i,T}^{+} = \left\{ m \leq 1 + \frac{32\alpha\sigma^2 \log(KT)}{\Delta_{i,m-1}^2} \right\} \\
&\text{where } \Delta_{i,m} = \min_{a^{(j)} \in \mathcal{A}} \mu(j)(T_j^*(t) - 1) - \bar{\mu}_{(T_i^*(t)+m)}^{m}(i) \\
&\text{and } \bar{\mu}_n^m(i) = \frac{1}{n}\sum_{j=1}^{m} \mu_i(n-j).
\end{aligned}$$

Then, they used these definitions to state the upper bound, which is equal to:

$$R_T \leq \sum_{a^{(i)} \in \mathcal{A}} \left(\frac{C_5 \log(KT)}{\Delta_{i, m_{i,T}^{+} - 1}} + \sqrt{C_5 \log(KT)} + L \right) \tag{3.40}$$

for $\delta_t = 1/(Kt^5)$ and $C_\alpha = 32\alpha\sigma^2$.

A subsequent work is proposed in [Seznec et al., 2020], where some of the same authors from [Seznec et al., 2019] further study the rotting environment, both in the restless and rested setting. We presented the restless rotting bandit problem in the previous section when discussing restless bandits. Here, we focus on the rotting rested environment and discuss the relationship between the two. First, we recall the difference between restless and rested rotting bandits and show some results for the combined setting. While in the restless case the reward function of any arm $a^{(i)}$ decays with time irrespective of the learner's actions, namely $\mu_i(t)$, in the rested case the reward function of any arm $a^{(i)}$ decreases with the number of pulls, namely $\mu_i(T_i(t))$. In [Seznec et al., 2020] the authors study the possibility of a combined setting for a rotting bandit with both restless and rested arms. They prove that the worst-case regret suffered in this specific setting is linear, showing that oracle greedy suffers a regret lower bounded by

$R_T \geq \lfloor T/4 \rfloor$ and that any learning strategy suffers $R_T \geq \lfloor T/8 \rfloor$. These results show that learning a bandit problem with both restless and rested rotting arms is difficult and achieves linear regret, but by separating the restless and rested arms allows a near-optimal regret guarantee.

Since the restless case has been discussed before, here we focus on the rested rotting bandit problem. The authors propose the `RAW-UCB` algorithm. It starts by pulling each arm once and then selects the arm which maximizes the UCB index as defined in Equation (3.22) with the mean reward defined in the first of the two equations in Equation (3.22). The UCB index is defined as the average reward on the last h observations summed to a confidence bound defined in Equation (3.21). The authors analyze the regret of `Raw-UCB` applied in the rested rotting environment and prove, for $\alpha \geq 5$, a problem independent expected regret upper bound of:

$$R_T \leq C\sigma\sqrt{\log(T)}\left(\sqrt{KT} + K\right) + 6KL \tag{3.41}$$

and a problem-dependent bound of:

$$R_T \leq \sum_{a^{(i)} \in \mathcal{A}} \left(\frac{C^2\sigma^2 \log(T)}{\Delta_{i,m^+_{i,T}-1}} + C\sigma\sqrt{\log(T)} + 6L \right) \tag{3.42}$$

where $C = 2\sqrt{2\alpha}$ is a universal constant which depends on α, $m^+_{i,T} = \max\{m \leq 1 + \frac{C^2\sigma^2 \log(T)}{\Delta^2_{i,m-1}}\}$, and $\Delta_{i,m} = \min_{a^{(i)} \in \mathcal{A}}(N^*_{j,T} - 1) - \bar{\mu}^m_{(N^*_{i,T}+h)}(i)$.

We mention two more works which are addressing two different environments compared to the works mentioned so far. We start by presenting the rotting bandit problem with infinitely many arms [Kim et al., 2022]. In this setting, the player is faced with an infinite set of actions. Although in bandits literature infinite set of actions often imply linear assumptions on the reward function, here the authors do not make this assumption. They assume that $\mu_a(t)$ is the mean reward of action a at time t. The reward obtained at time t for pulling arm a is defined as $X_t = \mu_{a_t}(t) + \eta_t$ where η_t denotes 1-sub-Gaussian noise. At time $t = 1$ the reward is bounded in $[0, 1]$, but it decreases with a rotting rate of ρ_t where $0 \leq \rho_t \leq \rho$ and $\rho = o(1)$. In rounds $t \geq 1$, the rotting behaviour follows the formula $\mu_a(t+1) = \mu_a(t) - \rho_t$ for the arm that has been pulled in the previous step. Therefore, at time t we can define the mean reward as:

$$\mu_a(t) = \mu_a(1) - \sum_{s=1}^{t-1} \rho_s \mathbb{I}(a_s = a). \tag{3.43}$$

The performance of an algorithm is measured against an optimal arm which returns a reward of 1 at every time step, justified by the infinite set of actions available. The authors provide a lower bound of the expected regret equal to $R_T = \Omega(\max\{\rho^{1/3}T, \sqrt{T}\})$. When the rotting rate is $\rho \leq 1/T^{3/2}$, the lower bound achieves $\Omega(\sqrt{T})$. Otherwise, when the rotting rate is larger, $\rho > 1/T^{3/2}$, the lower bound is $\Omega(\rho^{1/3}T)$. The authors propose two algorithms: one for when the maximum rotting rate is known and one for unknown maximum rotting rates. The first one called `UCB-Threshold Policy (UCB-TP)` is based on the definition of an estimator for the initial mean reward of an arm a defined as:

$$\widetilde{\mu}^o_a(t) = \frac{\sum_{s=1}^{t-1}(X_s + \rho T_a(s))\mathbb{I}(a_s = a)}{T_a(t)}, \tag{3.44}$$

where $T_a(t)$ is the number of times arm a has been pulled. The algorithm updates this estimator in each round and checks if this arm is good with a condition based on a UCB index defined as:

$$UCB_a(t) = \widetilde{\mu}^o_a(t) - \rho T_a(t) + \sqrt{8\log(T)/T_a(t)}. \tag{3.45}$$

If the UCB index of arm a satisfies a threshold-based condition, $UCB_a(t) \geq 1-\delta$ where δ is an input of the algorithm, then it keeps pulling arm a. If the condition is not satisfied anymore, the player removes the action from the set of actions and pulls another arm. Setting $\delta = \max\{\rho^{1/3}, 1/\sqrt{T}\}$ and having $\rho = o(1)$, the expected regret upper bound is equal to $R_T = \widetilde{\mathcal{O}}(\max\{\rho^{1/3}T, \sqrt{T}\})$. The second algorithm works when the maximum rotting rate ρ is unknown. To deal with this missing information, the authors propose an algorithm of bandit-over-bandits, called `Adaptive UCB-TP (AUCB-TP)`, where a main bandit is running the EXP3 algorithm on a bandit instance of `UCB-TP`, the algorithm proposed for a known maximum rotting rate. The difference consists in the fact that here the main bandit divides the horizon T into blocks of size M and at the beginning of each block picks from a pool of possible candidates an estimator $\widetilde{\beta}$ for the unknown maximum rotting rate. Then, the bandit instance of `UCB-TP` is run with $\widetilde{\beta}$ as the maximum rotting rate and $\delta = \widetilde{\beta}^{1/3}$. In this case, the regret upper bound is proved to be $\widetilde{\mathcal{O}}(\max\{\rho^{1/3}T, T^{3/4}\})$.

Finally, we mention a result studied in [Seznec, 2020] where the author proves that the problem of linear rested rotting bandit is impossible to learn. In this analysis, the authors consider a setting with an infinite set of actions $\mathcal{A} \subset \mathbb{R}^d$ and where the reward function is linear, meaning that the reward obtained at time t after playing arm a_t is $X_t = \langle \mu, a_t \rangle + \eta_t$ where η_t is noise with mean reward equal to zero and $\mu \in \mathbb{R}^d$ is an unknown reward vector which can be seen as the typical parameter θ^* presented in the linear bandit section of Section 2.5. Instead of having a single latent parameter θ^*, this model is characterized by d unknown parameters $\mu \in \mathbb{R}^d$. The rested rotting bandit environment is modelled by introducing d functions $\mu_i : \mathbb{R}^+ \to \mathbb{R}$ which are assumed to be non-increasing and L-Lipschitz. Here, each function $\mu(1), \ldots, \mu(d)$ corresponds to one of the d parameters in μ. While in the K-armed rested rotting bandit the reward function of an arm depended on the number of pulls, here in the linear setting we measure the impact of pulling arm a on the next rounds defining the quantity $T_{i,t} = \sum_{s=1}^{t} a_{s,i}$. This means that the pull of a certain arm with non-zero components in dimension $i \in \{1, \ldots, d\}$ will impact the future rewards obtained from pulling actions with non-zero components at index i. We denote the cumulative pulling intensity as $N_T = \sum_{i=1}^{d} T_{i,T}$. When restraining the action set $\mathcal{A} \subset \mathbb{R}^{+d}$ to be the positive quadrant of the ball, we can show how the model can recover rested rotting bandits as well as standard linear bandits when the reward functions are constant. On the other hand, when $\mathcal{A}$ is the actions set formed by the d canonical basis vector, it is possible to recover rested rotting bandits with finitely many actions. However, when these restrictions are relaxed and we consider $\mathcal{A} \subset \mathbb{R}$, the author shows that oracle greedy is not an optimal policy anymore and that for the simple case where $d = 2$, oracle greedy can suffer a regret of $(L(T-2))/8$. This happens because oracle greedy can be tricked into pulling the arm with the higher mean reward right from the beginning even in settings where the higher reward would be achieved when playing the suboptimal arm first. Indeed, there are cases where the optimal policy needs to avoid playing the optimal arm early on as its reward function could decrease also the suboptimal arm due to the cross-arm effects and the dependencies caused by the non-orthogonality of the actions. For this reason, Proposition 4.7.2 and Corollary 4.7.3 in [Seznec, 2020] show that for any learning policy with $T \geq 23$, there exists a linear rested rotting bandit problem with lower bound $R_T \geq LT/20$, showing that any learning policy gets a worst-case linear regret, even when removing noise from the setting.

Rising rested bandits

Among the class of rested bandits, we now consider the family of rising rested bandits. We use the terms rising, increasing, and recharging to indicate the broad family of bandits where the reward function of an arm increases with its number of pulls. With each work we are going to present, we will clarify the specific assumptions and environment proposed by the authors.
Rising bandits in [Metelli et al., 2022] address a rested bandit setting where the reward of every action $a^{(i)} \in \mathcal{A}$ with $|\mathcal{A}| = K$ depends on the number of times the action has been pulled before. Specifically, the payoffs are bounded in $[0,1]$, have σ^2-sub-Gaussian noise, and $\mu_i(N_{i,t})$, where μ_i is a non-decreasing and concave function and $N_{i,t}$is the number of times arm $a^{(i)}$ has been pulled up to time t. As for the rising restless bandit model, the monotonicity is defined as $\gamma_i = \mu_i(n+1) - \mu_i(n) \geq 0$, the concavity as $\gamma_i(n+1) - \gamma_i(n) \leq 0$, and they characterize the problem by introducing the definition of *cumulative increment* $\Gamma_\mu(s,q) = \max_{a^{(i)} \in \mathcal{A}} \left\{ \sum_{l=1}^{s-1} \gamma_i(l)^q \right\}$ for $q \in [0,1]$ and $s \in \{1, \dots, T\}$. The first result presented in [Metelli et al., 2022] is to show that the optimal policy for rising rested bandits is what they call *oracle constant policy*, which consists in pulling constantly the same arm which maximizes the cumulative reward over the horizon, namely $a^* = \operatorname{argmax}_{a^{(i)} \in \mathcal{A}} \sum_{t=1}^{T} \mu_i(t) \quad \forall t \geq T$. The authors exploit this knowledge and propose an algorithm `R-ed-UCB` for the deterministic setting which at each time step selects the arm with the highest exploration index $B_i(t)$ which is set equal to $\bar{\mu}_t^{R-ed}(i)$ defined as:

$$\bar{\mu}_t^{R-ed}(i) = \mu_i(N_{i,t-1}) + (t - N_{i,t-1})\gamma_i(N_{i,t-1} - 1). \tag{3.46}$$

This index can be seen as a UCB index, where the most recent payoff is summed to the term $(t - N_{i,t-1})\gamma_i(N_{i,t-1} - 1)$, which upper bounds the sum of future increments with the most recent increment received. Using this definition and for $q \in [0, 1]$, `R-ed-UCB` achieves an expected regret upper bound equal to:

$$R_T \leq 2K + KT^q \Gamma_\mu \Big(\lceil \frac{T}{K} \rceil, q \Big), \tag{3.47}$$

where q can be selected to make the bound tighter. Note that the optimal value depends on the cumulative increment $\Gamma_\mu(T, q)$ which is a function of T. Starting from this result, the authors move to the stochastic setting, where equation Equation (3.46) cannot be used. To tackle this setting, the authors propose a solution based on a d-sized window whose size governs the bias-variance trade-off between using a few recent payoffs and many past rewards. Here, the exploration bonus for $d \in [N_{i,t-1}]$ becomes:

$$\widehat{\mu}_i^{R-ed}(t) = \frac{1}{d} \sum_{n=N_{i,t-1}-d+1}^{N_{i,t-1}} \Big(\widehat{\mu}_{i,n}(t) + (t-n) \frac{\widehat{\mu}_{i,n}(t) - \widehat{\mu}_{i,n-d}(t)}{d} \Big), \tag{3.48}$$

where $\widehat{\mu}_{i,n}(t)$ is the estimated reward and the term $(t-n)\frac{\widehat{\mu}_{i,n}(t)-\widehat{\mu}_{i,n-d}(t)}{d}$ accounts for the estimated increment. In the algorithm proposed for the stochastic setting, the exploration index is set to $B_i(t) = \widehat{\mu}_{i,d}^{R-ed}(t) + \beta_{i,d}^{R-ed}(t)$ where $\beta_{i,d}^{R-ed}(t, \delta_t) = \sigma(t - N_{i,t-1} + d_{i,t} - 1)\sqrt{\frac{10 \log(1/\delta_t)}{d_{i,t}^3}}$. The algorithm selects in each time step the arm which maximizes this index. For $q \in [0, 1]$, $d_{i,t} = \lfloor \varepsilon N_{i,t-1} \rfloor$ for $\varepsilon \in (0, 1/2)$, $\delta_t = t^{-\alpha}$ for $\alpha > 2$, the algorithm achieves an expected regret bound of:

$$R_T = \mathcal{O}\left(\frac{K}{\varepsilon}(\sigma T)^{2/3}(\alpha \log(T))^{1/3} + \frac{KT^q}{1-2\varepsilon}\Gamma_\mu\left(\left\lceil (1-2\varepsilon)\frac{T}{K} \right\rceil, q \right) \right). \tag{3.49}$$

Fidelity bandits

Fidelity bandits [Lugosi et al., 2021] is a K-armed bandit which models several non-stationary behaviours. Although the authors present both a stochastic and an adversarial variant of this model, we will focus on the former as it is relevant to the current discussion. Furthermore, the authors propose a bandit which can represents two different models of non-stationarity, which they call *loyalty-points* and *subscription* model. Both these models are then further analyzed for decreasing, increasing, and *coupon* reward functions. We start by discussing the loyalty-points model, as it can be classified as a rested bandit. At the beginning of the next section, when presenting state-dependent bandits, we will discuss the subscription model, as it belongs to this other class.
Since the environment is stochastic, every arm $a^{(i)} \in \mathcal{A}$ is generating rewards from a reward distribution with expected mean μ_i. The rewards are also assumed to be bounded in $[0, 1]$. The horizon T is assumed to be known. What characterizes this bandit model is the fact that the reward obtained from playing arm $a^{(i)}$ at time t is given by:

$$Y_{t,i} = X_{t,i} + \phi_i(\mathcal{H}_{t-1}), \tag{3.50}$$

Two main parts build the reward. The first is $X_{t,i}$, which is the base reward obtained from playing arm $a^{(i)}$ at time t. The second term is $\phi_i(\mathcal{H}_{t-1})$, which is a known arm-specific function which depends on the history of past actions $\mathcal{H}_{t-1}$ played by the learner up to time t and models two different behaviours of fidelity rewards. As we said, the authors distinguish between the *loyalty-points* model and the *subscription* model. In this section, we discuss the former. In this case, the fidelity rewards are generated by a known function $f : \mathcal{N} \rightarrow [0, 1]$ which depends on the number of times $N_{i,t-1}$ the action has been played in the past, so $\phi_i(\mathcal{H}_{t-1}) = f_i(N_{i,t-1})$. The arm-specific function f_i is known and the authors analyze three different cases based on the assumptions on f_i: when f_i is non-decreasing, non-increasing and the *coupon setting*, where the player receives a fidelity reward of $r_i \in [0, 1]$ after every ρ_i plays, which are not necessarily consecutive. We start by analyzing the loyalty-points model in the setting where the fidelity rewards functions $f_i \forall a^{(i)} \in \mathcal{A}$ are non-decreasing functions in $N_{i,t-1}$. For this setting, the authors provide an adaptation of the UCB algorithm. After playing each arm once and using the rewards obtained to update the estimates, the player selects at each time step t the arm:

$$a_t = \underset{a^{(i)} \in \mathcal{A}}{\operatorname{argmax}} \, UCB_i(t-1), \tag{3.51}$$

where the UCB index is defined as:

$$UCB_i(t) = \bar{X}_{i,t} + \frac{1}{T} F_i(T) + \sqrt{\frac{2\log(KT)}{N_{i,t}}}, \tag{3.52}$$

where the term under the square root is the exploration bonus and the first two terms constitute the modified reward, where $F_i(T) = \sum_{t=1}^{T} f_i(t)$. Note that the computation of this modified reward is possible since both T and the fidelity functions $f_i \forall a^{(i)} \in \mathcal{A}$ are known. The expected regret obtained by this algorithm in this specific loyalty-points setting with increasing fidelity functions is upper bounded by the following quantity:

$$R_T \leq \sum_{i:a^{(i)} \neq a^*}^{K} \frac{16\log(KT)}{\widetilde{\Delta}_i^2} \left(\mu^* - \mu_i + f^*(T) - f_0(i)\right) + \frac{1}{K}, \tag{3.53}$$

where $\widetilde{\Delta}_i = \mu^* - \mu_i + (F^*(T) - F_T(i))/T$, meaning that the worst case regret is of order $\mathcal{O}(T^{2/3}(K\log(T))^{1/3})$.
For the loyalty-points model with non-increasing fidelity functions, the authors refer to the model proposed by [Seznec et al., 2020] with regret of order $\widetilde{\mathcal{O}}(\sqrt{KT\log(T)})$, where the authors show that this bound matches the lower bound for the standard bandit problem up to logarithmic factors. For the loyalty-points model with coupon rewards, the authors propose a variant of the UCB algorithm using the following augmented rewards

$$\widehat{Y}_{i,t} = X_{i,t} + \frac{r_i}{\rho_i} \tag{3.54}$$

and show that there exists a strategy which gets an expected regret bounded by:

$$R_T \leq \sum_{i:\Delta_i>0}^{K} \frac{16\log(T)}{\Delta_i} + 4K. \tag{3.55}$$

where $\Delta_i = \max_{a^{(j)}\in\mathcal{A}} \left(\mu(j) + \frac{r_j}{\rho_j}\right) - \left(\mu_i + \frac{r_i}{\rho_i}\right)$, so the worst case regret is of order $\mathcal{O}(\sqrt{KT\log(T)})$.

3.2.2 State-dependent bandits

After discussing rested models, we switch to *state-dependent bandits*, where the reward function of an arm depends either on the last time the arm has been pulled or on the number of consecutive pulls of the arm. We will specify the assumptions on the reward functions when presenting each model. We stress that one important characterization of state-dependent models is the fact that the cumulative reward of a policy does not depend on the number of pulls of each arm, as in rested bandits, but on the order of the pulls.

After analyzing the loyalty-points model at the end of the previous section, we start by introducing the *subscription model* of fidelity bandits. As we said, there is a stochastic environment with $|\mathcal{A}| = K$, known horizon T, and where the reward of each arm is generated from a reward distribution with expected mean μ_i, and rewards bounded in $[0,1]$. When the learner plays action $a^{(i)}$ at time t, the reward is given by:

$$Y_{t,i} = X_{t,i} + \phi_i(\mathcal{H}_{t-1}), \tag{3.56}$$

where $X_{t,i}$ is the base reward obtained from playing arm $a^{(i)}$ at time t, $\phi_i(\mathcal{H}_{t-1})$ is a known arm-specific function which depends on the history of past actions $\mathcal{H}_{t-1}$ played by the learner up to time t. In the subscription model we have that $\phi_i(\mathcal{H}_{t-1}) = f_{(Q_{i,t})}(i)$, where the fidelity rewards depend on the current number of consecutive pulls of the arm $Q_{i,t} = \mathbb{I}\{t - \max\{s \leq t : a_s = a^{(i)}, a_{s-1} \neq a^{(i)}\}\}$ where $Q_{i,t} = 0 \ \forall a^{(i)} \neq a_{t-1}$. When the fidelity function $f : \mathbb{N} \to [0,1]$ is non-decreasing in $Q_{i,t}$, the authors mention that strategies employed for best-arm identification can be directly used in this setting achieving an expected regret of $R_T \leq CT^{2/3}(K\log K)^{1/3}$ where C is a numerical constant. When the fidelity reward function is non-increasing, the authors propose the following strategy. After an initial phase of pure exploration of length t_0 where each arm is played $\lfloor t_0/K \rfloor$, the policy uses the empirical average rewards to select the two arms $a_{\widehat{i}}$ and $a_{\widehat{j}}$ with the highest values of $\widehat{\mu}_i + f_0(i)$ such that $\widehat{\mu}_{\widehat{i}} + f_{\widehat{i}}(0) \geq \widehat{\mu}_{\widehat{j}} + f_{\widehat{j}}(0)$. Then, the learner computes the following quantity:

$$\widehat{m} = \min\{m : \widehat{\mu}_{\widehat{i}} + f_{\widehat{i}}(m+1) < \widehat{\mu}_{\widehat{j}} + f_{\widehat{j}}(0)\}. \tag{3.57}$$

After time step t_0, the strategy consists in pulling arm $a_{\widehat{i}}$ for $\widehat{m}$ times, pulling $a_{\widehat{j}}$ once, and then coming back to arm $a_{\widehat{i}}$ again, repeating this process until T is reached. The choice of $\widehat{i}$, $\widehat{j}$, and $\widehat{m}$ can be written as the maximization of the following quantity:

$$W_T(\mu_i, \mu(j), m) = \left(T - \left\lfloor \frac{T}{m+1} \right\rfloor\right) \mu_i + \left\lfloor \frac{T}{m+1} \right\rfloor \mu(j) \\ + \left\lfloor \frac{T}{m+1} \right\rfloor (F_m(i) + F_j(1)) + F_i \left(T - (m+1) \left\lfloor \frac{T}{m+1} \right\rfloor\right).$$

The authors prove that for $t_0 = (2T)^{2/3}(K \log K)^{1/3}$ this strategy gets an expected regret upper bounded by $R_T \leq 3T^{2/3}(K \log K)^{1/3}$.
Finally, the authors analyze the subscription model with *coupon rewards*. This characterization means that the player only receives a fidelity reward of $r_i \in [0, 1]$ only after every ρ_i plays, not necessarily consecutive. For this setting, they propose a batch variant of the classic UCB algorithm. The strategy consists of selecting arm a_t which maximizes the UCB index, defined as:

$$UCB_i(t) = \bar{X}_{i,t} + \frac{r_i}{\rho_i} + \sqrt{\frac{2\log(KT)}{N_{i,t}}}, \tag{3.58}$$

where the $\bar{X}_{i,t}$ is the empirical average reward computed over $N_{i,t}$ samples up to time t, and ρ_i are known since the fidelity reward functions are known. Each time a_t is selected, it is played ρ_i times to guarantee that the fidelity reward is being collected. The expected regret of this strategy in the subscription model with coupon rewards is upper bounded by the following quantity:

$$R_T \leq \sum_{i:a^{(i)} \neq a^*} \frac{16\log(TK)}{\widetilde{\Delta}_i} + \sum_{i:a^{(i)} \neq a^*} \rho_i \widetilde{\Delta}_i + \left(1 + \frac{2}{K}\right) \sum_{i:a^{(i)} \neq a^*} \widetilde{\Delta}_i + 2\bar{\rho}, \tag{3.59}$$

where $\widetilde{\Delta}_i = \max_{a^{(j)} \in \mathcal{A}}(\mu(j) + r_j/\rho_j) - (\mu_i + r_i/\rho_i)$ and $\bar{\rho}$ is the least common multiple of $\rho_1, \ldots, \rho - K$, getting a worst case regret bound of $\mathcal{O}(\sqrt{KT\log(T)} + \sum_{i=1}^{K} \rho_i + \bar{\rho})$.

We proceed by mentioning the model called *recharging bandits* by [Kleinberg and Immorlica, 2018]. In this work, the environment is stochastic and deals with a finite set of actions, $|\mathcal{A}| = K$. The expected reward of an arm depends on how far back in the past it was last pulled. Specifically, for any arm $a^{(i)} \in \mathcal{A}$ the expected reward at time t is given by a weakly concave and weakly increasing arm-specific reward function $H_i(\tau_t)$, where τ_t indicates how many steps in the past the action was last pulled and $H_i(0) = 0$. In their work, the authors prove that greedy is suboptimal. First, they studied the problem with known payoffs to understand what is the optimal strategy in an offline setting. They show that greedy can be arbitrarily close to a (1/2)-approximation, but never less than that. To solve this problem, they propose a concave program to upper bound the cumulative reward of the optimal rounding scheme of arms and use it to show a constant approximation for what they call *interleave rounding*. This idea, which will be used in the proposed algorithm, is to schedule the pull of an arm in a continuous time with a specific frequency, using the frequency of how much this arm would be pulled by the optimal schedule, and adding a random offset to avoid arms colliding in the same spot. This is then translated into discrete time by preserving the order. This scheme achieves a general $(1 - 1/2e)$-approximation and it is shown to be optimal for the setting where $K = 2$. Exploiting concavity, the authors prove that for every $0 < \varepsilon < 1$ there exists a periodic schedule of length $T \geq K/\varepsilon$ whose asymptotic average reward is at least $(1-\varepsilon)$-OPT. They propose an algorithm for unknown reward functions based on UCBs. The algorithm divides the horizon into epochs and computes the upper confidence bounds on every $H_i(x)$, which is defined as $\bar{H}_i(x) = q/n + \tau_i\sqrt{(4\ln(KnT))/n}$ in order to be concave, where τ_i is the delay, n is the number of samples of the arm-delay pair and q their sum. Then, it uses these UCBs to run an approximation algorithm to plan for the scheduling of arms in the current epoch. Once the epoch is ended, it uses the rewards obtained to update the estimates of the arms and plan for the following epoch. This algorithm achieves an expected regret upper bounded by

$$R_T = \mathcal{O}\Big(\frac{K^2\log(K)}{\varepsilon^{\mathcal{O}(1/\varepsilon)}}\sqrt{\frac{\log(T)}{T}}\Big). \tag{3.60}$$

A related work is the one by [Pike-Burke and Grunewalder, 2019]. The authors study the finite-arm stochastic bandit environment, where the reward of an action follows an unknown recovery function and depends on the time passed since the arm was last pulled. The number of rounds since an arm $a^{(i)}$ was last pulled is denoted by $Z_{i,t} \in \mathcal{Z} = \{0, \dots, z_{max}\}$ for a finite $z_{max} \in \mathbb{N}$. This state is updated as follows:

$$Z_{i,t} = \begin{cases} 0 & \text{if } a_t = a^{(i)} \\ \min\{z_{max}, Z_{i,t}+1\} & \text{if } a_t \neq a^{(i)}, \end{cases} \tag{3.61}$$

meaning that if the arm has been pulled more than z_{max} rounds ago, the state will remain equal to z_{max}. The reward obtained by the learner is given by:

$$X_t = f_{i_t}(Z_{i_t,t}) + \varepsilon_t \tag{3.62}$$

where f_i is the unknown recovery function and ε_t are i.i.d. $\mathcal{N}(0, \sigma^2)$ random variables with known σ. The difference compared to the previous work is that the recovery function is assumed to be sampled from a Gaussian process with mean 0 and known kernel. To measure the performance of their solution, the authors propose the *full horizon regret*, the *instantaneous regret*, and the *d-step lookahead regret*, but focus on the last one. While the *full horizon regret* matches the definition of regret employed so far, the one where the comparator is the best sequence of T actions, it is necessary to define the two other definitions of regret. *Instantaneous regret* is defined as the regret between the learner's policy and oracle greedy which selects the best arm at the states $\boldsymbol{\tau}_t = \{\tau_{1,t}, \dots, \tau_{K,t}\}$. On the other hand, the *d-step lookahead regret* measures the difference between the cumulative reward of the learner's strategy and the optimal sequence of actions for the current delays $\boldsymbol{\tau}_t$ considering subsequent epochs of size $d \geq 1$. It is important to distinguish between the full horizon regret and the d-step lookahead regret, as the latter is not measured against the optimal sequence of actions over the horizon but the optimal sequence of actions in the current d-sized epoch. As for the previous bandit problem, the authors show the intractability of this problem even in the offline setting. To solve it, the authors propose a d-step lookahead UCB algorithm and Thompson Sampling. The idea behind both algorithms is to construct a d-step lookahead tree with edges representing arms played with updated delay. A leaf on the tree represents a sequence of d arms, whose expected reward is the sum of the expected rewards collected by the actions played from the origin to the leaf. The idea for the use of UCBs is to select the leaf with the largest UCB index at time t which is defined as:

$$UCB_i(t) = \sum_{l=0}^{d-1} \mu_{(J_{t+l}}(t) + \alpha_t \sum_{l,q=0}^{d-1} cov_t(f_{J_{t+l}}(Z_{J_{t+l},t+l}), f_{J_{t+q}}(Z_{J_{t+q},t+q})) \tag{3.63}$$

where $\{J_{t+l}\}_{l=0}^{d-1}$ is the sequence of arms, $\{Z_{J_{t+l},t+l}\}_{l=0}^{d-1}$ is the sequence of delays, and

$$\begin{aligned} &cov_t(f_{J_{t+l}}(Z_{J_{t+l},t+l}), f_{J_{t+q}}(Z_{J_{t+q},t+q})) \\ &= \mathbb{I}\{J_{t+l} = J_{t+q}\} k_{J_{t+l}}(Z_{J_{t+l},t+l}, Z_{J_{t+q},t+q}, N_{J_{t+l}}(t)) \end{aligned} \tag{3.64}$$

where $k_{J_{t+l}}(Z_{J_{t+l},t+l}, Z_{J_{t+q},t+q}, N_{J_{t+l}}(t))$ is the covariance related to the kernel defined by the Gaussian process (see Section 3 in [Pike-Burke and Grunewalder, 2019]). The authors analyze the two algorithms in two different settings. In the first setting, they allow for a single play of any arm $a^{(i)}$ in the d-step. In this case, they show that the expected d-step lookahead regret of both the UCB strategy and the Thompson Sampling strategy is upper bounded by the following quantity:

$$R_T = \mathcal{O}(\sqrt{KT\gamma_T \log(TK|\mathcal{Z}|)}), \tag{3.65}$$

where $\mathcal{Z} = \{0, \tau_{max}\}$ is the set of possible states and γ_T is the maximal information gain from T samples, where $\gamma_T = \mathcal{O}(\log(T))$ for linear kernels and $\gamma_T = \mathcal{O}(\log^2(T))$ for squared exponential kernels. In the second setting, they allow for multiple pulls of any arm $a^{(i)}$ in the same d-step. In this case, the expected d-step lookahead regret is upper bounded by:

$$R_T = \mathcal{O}(\sqrt{KT\gamma_T \log((K|\mathcal{Z}|)^d T}), \tag{3.66}$$

for both the UCB and Thompson Sampling strategies. The authors also analyse the instantaneous regret, showing that in this case both algorithms are upper bounded by:

$$R_T = \mathcal{O}(\sqrt{KT\gamma_T \log(TK|\mathcal{Z}|)}), \tag{3.67}$$

the same obtained from the d-step lookahead regret in the single-play setting.

Stochastic bandits with delay-dependent payoffs (B2DEP) is a non-stationary bandit model proposed in [Cella and Cesa-Bianchi, 2020]. The environment is characterized by a finite set of actions $|\mathcal{A}| = K$ where each one has a state τ_i which denotes the number of rounds since arm $a^{(i)}$ was last pulled. Each arm $a^{(i)}$ is associated with an unknown arm-specific delay-parameter $d_i > 0$. The expected reward $\mu_i(\tau)$ of an arm $a^{(i)}$ is a function of τ_i. When $1 \leq \tau_i \leq d_i$, then $\mu_i(\tau) \leq \mu_i$. Otherwise, if $\tau_i > d_i$, then $\mu_i(\tau_i) = \mu_i$. More specifically, each arm is associated with an unknown baseline expected reward μ_i from a fixed and unknown reward distribution. The reward function of an arm $a^{(i)}$ with state τ_i is a function bounded between $[0,1]$, non-decreasing and defined as:

$$\mu_i(\tau) = (1 - f(\tau)\{0 < \tau \leq d_i\})\mu_i, \tag{3.68}$$

where $f : \mathbb{N} \to [0,1]$ is an unknown non-increasing function. This reward function means that when you first play any action $a^{(i)} \in \mathcal{A}$ the expected reward will match its baseline expected reward. Then, whenever $a^{(i)}$ is pulled, its expected reward will decrease to zero. At this point, τ_i denotes how many time steps have passed since its last pull. The more τ_i grows the more the expected reward will increase until it matches the baseline expected reward once $\tau_i > d_i$. After showing that finding the optimal policy is NP-hard, the authors tackle this bandit problem by considering the class of periodic ranking policies. This class is made of policies of different lengths which play arms in decreasing order of highest expected payoff. If we assume that $\mu(1), \ldots, \mu(K)$, then class $\Pi_K = \{\pi_m : m \in \{1, \ldots, K\}\}$, where policy π_m cycles over the arms $\{a^{(1)}, \ldots, a^{(m)}\}$. They denote with π_{ghost} the best policy in this class and show that the expected cumulative regret of π_{ghost} is close to the optimal policy with an approximation error of:

$$G_T(\pi_{ghost} \geq (1 - f(r_0))G_T(\pi^*) + \mathcal{O}(1), \tag{3.69}$$

where j_0 is the largest arm index j such that $\mu_i > \max_{l=1,\ldots,i-1} \mu_{(i-l)}(l) \quad i = 2, \ldots, j$ and $j_0 = 1$ if $\mu_2 \leq \mu_1(1)$. Therefore, there are two different objectives: learning the ordering of the arms and learning the best periodic ranking policy. For the first goal, the authors propose an algorithm called `Bandit Ranker` based on action elimination. It consists in sampling each arm, computing the average rewards, and keeping them sorted in decreasing order. Once the confidence interval of one arm does not overlap with others, the arm is removed from the set of arms. Then, the arm that is removed is placed at the root of a tree, with the left and right nodes containing arms which have respectively a bigger and lower average reward. This process is iterated until there is no arm whose average reward overlaps with any other. In the paper, the authors show that this is attained with probability $1 - \delta$ after a number of pulls of order:

$$\sum_{i=1}^{K-1} \frac{1}{\Delta_i^2} \ln \frac{1}{\delta \Delta_i}, \tag{3.70}$$

where the suboptimality gap is defined as:

$$\Delta_i = \begin{cases} \Delta_{1,2} & \text{if } i = 1 \\ \min\{\Delta_{i-1,i}, \Delta_{i,i+1}\} & \text{if } < i < K \\ \Delta_{K-1,K} & \text{if } i = K. \end{cases}$$

After learning the ordering of the arms, the next objective is to learn the best ranking policy. This is done using the π_{low} algorithm. Here, the algorithm consists in considering all periodic ranking policies and playing each one for $T_s/(m|\mathcal{A}_s|) + 1$ times with $T_s = T^{1-2^{-s}}$ and where $\mathcal{A}_s$ is the set of active policies at time s. After discarding the first sample due to calibration, the subsequent samples are collected and used to compute the average reward of each ranking policy and select the best one. Then, the average reward of the best one is used to remove all policies whose average reward is significantly smaller, using a Chernoff-Hoeffding bound equal to $\sqrt{\frac{K}{2T_s}\ln\frac{2KS}{\delta}}$ where $S0\{j \in \mathbb{N} : \sum_{s=1}^{j}(|\mathcal{A}_s| + T_s) \geq T\}$. The authors compute the regret of π_{low} not against the optimal policy in general, but against the best policy among the periodic ranking policies, π_{ghost}, and prove that this regret is of order:

$$G_T(\pi_{ghost}) - G_T(\pi_{low}) = \mathcal{O}\Big(K^2 \ln\ln T + \sqrt{KT\Big(\ln\frac{K}{\delta} + \ln\ln\ln T\Big)}\Big). \quad (3.71)$$

with probability at least $1-\delta$.

Another work related to the previous two is the one proposed in [Simchi-Levi et al., 2021]. The peculiarity of this work compared to the previous ones is that in this setting the authors assume that in each time step the learner can play at most N actions out of the K arms in $\mathcal{A}$. This characteristic makes it close to combinatorial bandits. Each action $a^{(i)}$ is associated with a non-decreasing recovery function $R_i(t-t')$, where t' denotes the time step when arm $a^{(i)}$ was last pulled and $R_i(0) = 0$ for any arm $a^{(i)}$. As for the previous two works, this means that the more time has passed since the last pull of action $a^{(i)}$, the higher its reward will be. The reward in each time step is bounded by R_{max}, which is a known constant. The authors show that oracle greedy, which selects in each time step the N arms with the highest expected rewards, is arbitrarily bad. The solution proposed in this work is to consider purely periodic policies which pull arm $a^{(i)}$ with a certain frequency d_i. The authors show that this offline problem achieves an approximation ratio of $1-\mathcal{O}(1/\sqrt{K})$, which is asymptotically optimal for $k\to\infty$. To tackle the online problem, they propose an algorithm based on UCBs called `Online Purely Periodic Learning`. It consists in dividing the horizon into epochs of fixed lengths. At the beginning of each epoch, the learner runs an offline oracle which returns a scheduling of arms to be played in that phase using the UCB indexes of the arms, which are defined as follows:

$$UCB_{i,j}(\tau) = \min\left\{\bar{R}_{i,j-1(\tau)}, R_{max}\sqrt{\frac{2\log(KT)}{\max\{n_{i,j-1}(\tau), 1\}}}, R_{max}\right\}, \quad (3.72)$$

where i indicates the arm and j the epoch, $\bar{R}_{i,j-1(\tau)}$ is the empirical mean of $R_i(\tau)$ prior to epoch j, and $n_{i,j-1}(\tau)$ is the number of samples collected for $R_i(\tau)$ prior to phase j. At the end of this epoch, the learner uses the rewards collected during the phase to update the estimates of the arms and their UCB indexes, before starting a new epoch. The authors measure the performance of this algorithm against the offline benchmark and prove that its expected regret for an epoch length of $\phi = \Theta(\sqrt{T/(\log(N+1))})$ is:

$$\mathcal{O}(\max\{K, K^{1/2}N^{3/4}\}\sqrt{T}) \tag{3.73}$$

with respect to $\gamma_N \cdot UB[K,N] \cdot T$, where $UB[K,N]$ is the objective value of the offline oracle and $\gamma_N = \max_{a^{(i)} \in \mathcal{A}: \tau_i > 0} \frac{a^{(i)}}{a^{(i)}+1} \cdot \frac{N}{N+a^{(i)}}$ in the offline oracle.

Rebounding bandits

Rebounding bandits in [Leqi et al., 2021] model an environment with a finite set of actions of size K, where an arm $a^{(i)}$ for $i = 1, \dots, K$ has bounded base rewards b_i for $i = 1, \dots, K$. Every arm has a *satiation level* defined as:

$$s_{i,t} = \gamma_i(s_{i,t-1} + u_{i,t-1}) + z_{i,t-1} \quad \forall t > t'_i \tag{3.74}$$

where $\gamma_i \in [0,1)$ is a *satiation retention factor* of arm $a^{(i)}$, $u_{i,t} = \{a_t = a^{(i)}\}$ indicates if arm $a^{(i)}$ has been pulled at time t, $z_{i,t-1}$ is a identically distributed Gaussian noise with zero mean and σ^2 variance, and $t'_i = \min\{t : a_t = a^{(i)}\}$ is the first time arm $a^{(i)}$ was pulled. Any arm has a satiation level equal to zero before being played for the first time, $s_{i,0} = \dots = s_{i,t'_i} = 0 \; \forall a^{(i)} \in \mathcal{A}$. Whenever an arm $a^{(i)}$ is pulled at time t, the reward obtained by the learner is equal to $\mu_i(t) = b_i - \lambda_i s_{i,t}$. Therefore, the expected reward of arm $a^{(i)}$ at time t is given by the difference between its base rewards b_i and the product between its satiation level $s_{i,t}$ and a bounded *exposure influence factor* $\lambda_i \geq 0$. The authors call the product $\lambda_i s_{i,t}$ the *satiation influence*. The behaviour modelled by this reward function is the following: the more the action is pulled the more its reward will decrease, while it rebounds back to its base reward when it is not been pulled for a while since the satiation level decreases. The authors first analyze the simpler setting with deterministic dynamic, removing the noise term $z_{i,t-1}$ setting $\sigma = 0$ from the formula of the satiation level. In this setting, they show that greedy is not optimal in general. For the case where the satiation effect is always zero, so for instance if the satiation retention factor $\gamma_i = 0 \; \forall i = 1, \dots, K$, greedy always pulls the arm with the highest instantaneous expected reward and it is optimal. Greedy is also optimal when all the actions have the same properties $\{\gamma_i, \lambda_i, b_i\} \; \forall i = 1, \dots, K$, playing all arms cyclically. However, it is not optimal in all other cases, which may be more representative of real applications. To solve this problem, they propose the algorithm called d-lookahead policy where d is the size of a window and the strategy is to select arms that maximize the total reward over the following d rounds, before repeating the process for the next d steps. The authors use this knowledge to face this problem in the original stochastic setting, reintroducing the noise $z_{i,t-1}$ with $\sigma > 0$, and with unknown satiation dynamics, where satiation levels are not observable by the learner. They represent this problem as a continuous state partially observable Markov Decision Process. The algorithm proposed to solve this problem is `Explore-Estimate-Plan (EEP)`. The strategy consists in collecting samples from the arms by playing them a fixed number of times, using these samples to estimate the parameters $\gamma_i, b_i, \lambda_i\gamma_i$. Then, it plans for the actions to play in the window of size d by using the strategy proposed in the d-lookahead policy. They adopt the d-lookahead regret definition from [Pike-Burke and Grunewalder, 2019], where the performance of the learner is analyzed window by window of size d and measured against the optimal policy given the initial state caused by the learner's past pulls in the previous window. The authors prove that there exists a constant T_0 such that $\forall T > T_0$ and $d \leq T^{2/3}$, this algorithm achieves a d-step lookahead regret of:

$$R_T = \mathcal{O}(K^{1/2}T^{2/3}\log(T)). \tag{3.75}$$

Blocking bandits

We end this chapter by presenting blocking bandits [Basu et al., 2019]. This bandit problem models a non-stationary stochastic setting with a finite set of actions of cardinality $|\mathcal{A}| = K$ at the beginning of the game. In this environment, the reward distributions of the arms are stochastic and bounded in $[0, 1]$. However, when the arm is pulled, it becomes unavailable for a deterministic and known number of rounds after that. The number of rounds during which the arm is temporarily unavailable is arm-dependent and called delay, $D_i \in \mathbb{N}^+$. If at time t we play arm $a^{(i)}$, arm $a^{(i)}$ will be blocked for the following $(D_i - 1) \geq 0$ time steps, so it will be unavailable in the time interval $\{t+1, \dots, t+D_i-1\}$. The authors use the notation $\tau_{i,t}$ to indicate the state of arm $a^{(i)}$ stating that $\tau_{i,t} = (D_i + \max_{t' \leq t}\{a_{t'} = a^{(i)}\} - t)$ and an arm is available when $\tau_{i,t} \leq 0$. Using these definitions, we can define the set of available actions at time t as $\mathcal{A}_t = \{a^{(i)} \in \mathcal{A} : \tau_{i,t} \leq 0\}$. As always, the goal is to maximize the cumulative reward or minimize the total regret. Firstly, they reduce their bandit problem to a pinwheel scheduling instance to prove its NP-hardness. We present this reduction since we will use a similar variant to prove the NP-hardness of the LSD bandit problem in Chapter 4. The pinwheel scheduling problem considers K arms with delays, s.t. each arm $a^{(i)}$ is associated with its delay D_i. The problem consists of deciding if there exists a schedule of these arms such that each arm $a^{(i)}$ appears at least once every D_i time steps. A pinwheel scheduling instance is called *dense* if the sum of the reciprocal of the delays sums to 1, $\sum_{i=1}^{K} 1/D_i = 1$. It has been proven in the literature that the pinwheel scheduling problem does not allow any pseudo-polynomial time algorithm unless the randomized exponential time hypothesis is false. The authors map the blocking bandit problem to the pinwheel scheduling on dense instances to prove that solving it is just as hard (Theorem 3.1, Corollary 3.2 in [Basu et al., 2019]). After showing the NP-hardness of this problem, the authors prove that greedy is asymptotically $(1 - 1/e)$ optimal for this problem and propose the `UCB Greedy` algorithm. This algorithm starts by playing each arm once, updating the statistics about each arm, and then selecting at each time step t the arm $a_t = \text{argmax}_{a^{(i)} \in \mathcal{A}_t}\left(\widehat{\mu}_i + \sqrt{\frac{8\log(t)}{N_i}}\right)$, where N_i is the number of times arm $a^{(i)}$ has been played up to time t. So, at each time step t the algorithm pulls the arm a_t with the highest UCB index and a_t will be blocked for a deterministic number of time steps such that $\mathcal{A}_{t+1} = \mathcal{A}_t \backslash a_t$. The authors prove a regret upper bound for this algorithm with respect to oracle greedy with known rewards, showing that for any $\varepsilon > 0$

$$R_T^{(1-1/e)} = \mathcal{O}\Big(\frac{1}{\varepsilon}\log(\frac{1}{\varepsilon})\Big) + \frac{32K_g(K-K_\varepsilon^*)}{\min_{a^{(i)}\in[K_\varepsilon^*,\dots,K_g]}\Delta_{i,i+1}}\log(T), \tag{3.76}$$

where K_g is the arm with the lowest mean reward strictly positive played by oracle greedy, $K_\varepsilon^* = \min(K \cup \{k : \sum_{i=1}^{(k-1)} 1/D_k \geq 1-\varepsilon\})$ for any $\varepsilon \geq 0$, and $\Delta(i,j) = \mu_i - \mu(j)$.
From the finite-arm variant, several models were proposed for blocking bandits in different settings, such as contextual blocking bandits, adversarial blocking bandits, and combinatorial blocking bandits. In this dissertation we analyze the first and the last variants, omitting the adversarial variant due to the focal points of this dissertation.

We start by discussing *contextual blocking bandits* [Basu et al., 2021]. This setting is stochastic and characterized by the observation of a context c_t at the beginning of each round. At the beginning of each round $t = 1,\dots,T$, the environment samples a context $c_t \in \mathcal{C}$ where $|\mathcal{C}| = m$. The player observes the context c_t and selects an action to play $a_t \in \mathcal{A}_t$. As in blocking bandits, each action is associated with a known delay $D_i \in \mathbb{N}^+$ which indicates how many rounds the action will be blocked once it is pulled. Specifically, when an action $a^{(i)}$ is played, it will be blocked for the following $(D_i - 1) \geq 0$ time steps, irrespective of the contexts observed in the next rounds. This means that the learner needs to select an arm knowing that the arm pulled at time t will be blocked for $(D_i - 1)$ time steps across all contexts. Once the action is played, the player observes the reward X_t. The reward X_t is sampled from a stochastic distribution with mean μ_{c_t,a_t}, which depends both on the arm and on the context. This reward is used to update the estimates of the selected arm, which will be removed from the set of available actions $\mathcal{A}_{t+1}$ and will be available again only from time step $t + D_i$. The authors designed the `UCB algorithm for contextual blocking bandits (UCB-CBB)` algorithm, which stores the UCB indices for every arm-context pair. The UCB is here defined as:

$$\bar{\mu}_{t,c_t}(i) = \min\left\{\widehat{\mu}_{i,c_t}(T_{i,c_t}(t)) + \sqrt{\frac{3\ln(t)}{2T_{i,c_t}(t)}}\right\}, \tag{3.77}$$

where $T_{i,c_t}(t)$ indicates the number of times the action $a^{(i)}$ was played together with context c_t and $\widehat{\mu}_{i,c_t}(T_{i,c_t}(i))$ is the empirical estimate of $\mu_{c_t,i}$ computed using the $T_{i,c_t}(t)$ samples. The algorithm selects the arm a_t by solving a linear program (LP) using the UCB indices. The LP is defined as:

$$\begin{aligned} \max \quad & \sum_{a^{(i)}\in\mathcal{A}}\sum_{c_t\in\mathcal{C}} \bar{\mu}_{t,c_t}(i)_{i,c_t} z_{i,c_t} \\ s.t. \quad & \sum_{c_t\in\mathcal{C}} z_{i,c_t} \leq \frac{1}{D_i} \quad \forall a^{(i)} \in \mathcal{A} \end{aligned}$$

$$\sum_{a^{(i)} \in \mathcal{A}} z_{i,c_t} < f_{c_t} \quad \forall c_t \in \mathcal{C}$$
$$z_{i,c_t} \geq 0 \quad \forall a^{(i)} \in \mathcal{A}_t, \forall c_t \in \mathcal{C}.$$

As one can see, the solution to the LP is linked to the sampling of the arms in the previous steps. To overcome this, the authors introduce the concept of *delayed exploitation*, which consists of considering at time t the indices of the actions from far in the past. In their algorithm, the authors allow for skipping, so they allow the player to play no arm in some rounds. To strike a balance between the availability of the arms and the skipping, they introduce non-skipping probabilities for each arm at every time step, which controls the rate of availability of one arm to ensure a balance between availability and skipping. The authors measure the performance of the algorithm using the approximate regret, also called α-regret, which is defined as:

$$\alpha R_T(\pi) = \alpha Rwd_T(\pi^*) - Rwd_T(\pi), \tag{3.78}$$

where π^* is the optimal policy and π is the policy followed by the player. Using this definition, the authors proved a α-regret upper bound for `UCB-CBB` with $\alpha = \frac{D_{max}}{2D_{max}-1}$ of:

$$\begin{aligned} \left(\frac{D_{max}}{2D_{max}-1}\right) - R_T \leq \sum_{a^{(i)} \in \mathcal{A}} \sum_{c_t \in \mathcal{C}} \frac{C(K+m)\log(T)}{\Delta_{min}^{i,c_t}} \\ + \frac{\pi^2}{6} \sum_{a^{(i)} \in \mathcal{A}} \sum_{c_t \in \mathcal{C}} \log\left(\frac{2(K+m)}{\Delta_{min}^{i,c_t}}\Delta_{max} + 6D_{max}\right), \end{aligned} \tag{3.79}$$

where $C > 0$ is a universal constant and m is the cardinality of the set of contexts $\mathcal{C}$.

Another work related to the family of blocking bandits is *combinatorial blocking bandits with stochastic delays* [Atsidakou et al., 2021]. The setting is similar to the ones presented before but adapted to a combinatorial setting. The environment is characterized by a finite set of actions $|\mathcal{A}| = K$, where each action $a^{(i)}$ is characterized by a mean reward μ_i. In the blocking setting, each action is associated with a delay D_i, which indicates that the action will be blocked for $(D_i - 1) \geq 0$ rounds, as defined in the previous two models. What differs in this combinatorial model is that the delay $D_{i,t}$ is a random variable sample from an arm-dependent distribution $\mathcal{D}_i$ with mean $d_i = \mathbb{E}[D_i, t]$, $\forall t \in \{1, \ldots, T\}$ and bounded in $[1, d_{max}]$, where d_{max} is the maximum delay defined as $d_{max} = \max_{a^{(i)} \in \mathcal{A}}\{d_i^{max}\}$. The reward X_t obtained by the learner is also bounded in $[0, 1]$ and, together with the delay, it is sampled independently from the joint distribution $\mathcal{X}_i, \mathcal{D}_i$, so that the reward and the delay are allowed to be correlated. The distributions of the delays are initially unknown to the player, who will infer $D_{i,t}$ by observing for how many rounds the arm will be blocked after being pulled. Due to the combinatorial nature of this setting, in each time step the player plays a super-arm of maximum size m as defined in

Equation (2.29). This means that an arm can be selected in a super-arm only if it is not blocked. This setting is semicombinatorial since the player can observe the rewards obtained by each action played in the super-arm and not only the cumulative reward of the super-arm. The authors propose the `CBBSD-UCB` algorithm, which is based on upper confidence bounds similarly to the solutions proposed for the previous models. The algorithm collects the rewards obtained by the arms when they are pulled and uses them to compute the UCB index, as defined in Equation (3.77). Then, it chooses the actions to play in the super-arm by solving an (α, β)-oracle [Atsidakou et al., 2021], which takes as input the subset of feasible arms and their UCB indices and outputs the maximum weight feasible set, which is a α-approximate solution. The algorithm plays the subset of arms from the set of available arms which maximizes the expected reward. The authors measure the performance of this algorithm using a $\rho(\alpha, \beta)$-regret where $\rho = \rho(\alpha, \beta) = \frac{\alpha\beta}{1+\alpha\beta}$ and prove the following distribution-independent bound:

$$\rho R_T \leq \frac{14\sqrt{KmT\ln(T)}}{1+\alpha\beta} + K(2 + \frac{\pi^2}{3}\Delta_{max}) + \mathcal{O}(d_{max}K). \tag{3.80}$$

Chapter 4

A novel non-stationary bandit model with finitely many arms

Contents

In the previous chapter, we analyzed several non-stationary bandit models in the literature. We started by distinguishing between two different types of non-stationarity: exogenous and endogenous. Inside these categories, we considered both restless, rested and neither restless nor rested bandits, along with different assumptions on the reward functions and how the history of past actions influences future rewards. In the current chapter, we present our contribution which fits into the literature of state-dependent bandits. We focus on a non-stationary bandit model called *Last Switch Dependent* (LSD) *bandit* presented in Laforgue et al. [2022]. This model does not fit neither in the restless nor the rested class of bandits, since the reward of the arm that is pulled changes differently from the reward of all the other arms that are not pulled. Furthermore, we can notice how most of the works presented in the previous chapter focus on one type of behaviour: either rotting or rising. The model we are going to present addresses different behaviours in a single model, aiming to be as general as possible. Before jumping into the technicalities of the model, we discuss the motivation and applications of this work. Afterwards, we present the definition of our model and some hardness results. We present our proposed approach, together with approximation and estimation errors. Finally, we are going to show some experiments to benchmark the performance of our solution against relevant algorithms proposed in the literature.

4.1 Motivations and applications

As in many works in the non-stationary bandits literature, the straightforward example of application is recommender systems where the system cannot select a single best item to suggest to a user. Indeed, in certain systems it is not sufficient to identify a single best item to guarantee a good performance and satisfy the user's preferences. Instead, in these applications the goal is to identify a set of items which could satisfy the non-stationary preferences of the user. This is true when the level of satisfaction of a user with an item strongly depends on the items they accessed in the past. We consider music streaming platforms where the items are music genres and the goal of a recommender system is to propose songs of certain genres. We imagine a sequential setting where in each time step the user listens to a song and returns a value from a predetermined scale which indicates how much they enjoyed it. In music recommender systems, it is plausible to assume that the behaviour of the user's preferences is not stationary (Dimitrijević et al. [1972], Kovacs et al. [2018]) and that different users have different behaviours. The non-stationarity can take different forms. The straightforward consequence caused by recommender systems providing a single suggestion to the user is that listening to the same genre repeatedly would very probably cause some users to feel bored or even annoyed. We use the term satiation to refer to the event when the user's level of satisfaction with respect to an item decreases the more the same item is submitted to the user. This phenomenon has been greatly studied in the literature (Kunaver and Požrl [2017], Leqi et al. [2021]). One can imagine how listening to jazz music all the time might cause the user to get bored. On the other hand, another form of non-stationarity can be identified in seasonality. We refer to seasonality as what happens when the user's level of satisfaction with respect to an item shows some peaks with a certain frequency Schedl et al. [2018]. We explain this phenomenon with the simple example of Christmas songs. We can imagine how a user may very much enjoy Christmas songs or Summer Hits once a year, but this level of satisfaction may decrease in other months of the year. This is an example of how certain items may be greatly appreciated by the users only with a certain frequency, after a certain period of time. Furthermore, how fast and how much the user gets bored or enthusiastic about an item depends on the specific user's inclinations. One may prefer certain music genres compared to others, and furthermore the characteristics of this preference greatly depend on the individual user's predilection. A user might get bored of listening to the same genre after a few plays, while another user could spend hours listening to the same genre before getting bored of it.
We discussed music recommender systems since it is one of the easiest examples to mention and the application which we focus on in this thesis. However, non-stationary behaviours such as satiation and seasonality may also occur in medical domains. We consider the context of psychological interventions for mental disorders, such as anxiety, depression, and post-traumatic stress disorder. When suffering from these conditions, the patient should be supported by a professional psychotherapist who treats a mental illness using psychological therapies, such as Cognitive Behavioural Therapy or Dialectical Behavioral Therapy. The

therapist's approach may suggest to the patient some micro-interventions such as positive psychology exercises, mindfulness, and some other activities which can improve the daily life of the patient. Some studies (Meinlschmidt et al. [2016], Owen et al. [2018], Schroeder et al. [2018], Fuller-Tyszkiewicz et al. [2019]) analyze this approach. Different activities may be suggested with different frequencies during the patient's treatment. For instance, a therapy session may occur once a week. On the other hand, walks and workouts may be planned multiple times during the week. The seasonality phenomenon explained before can model the frequencies of such activities. However, in Paredes et al. [2014] the authors show that when the same exercise is applied repeatedly its effect might decrease, following the satiation behaviour we encountered in the previous example. As for music recommender systems, in this setting it is also plausible that different patients have different needs and the exercises proposed by a therapist may be suggested with different frequencies based on the severity of each patient's condition.
These effects were considered when formalizing the model we propose in the rest of this chapter.

4.2 The Last Switch Dependent bandit model

Motivated by these applications, we propose a multiarmed bandit model able to address these two types of non-stationarity, satiation and seasonality, in a single framework. We define our model as a non-stationary bandit problem where the set of actions $\mathcal{A} = \{a^{(1)}, \dots, a^{(K)}\}$ is finite with cardinality $|\mathcal{A}| = K$. For all time steps $t = 1, \dots, T$, the learner selects an action $a_t \in \mathcal{A}$ to play and receives a reward X_t, which is used to update the information about the arms. Each arm $a \in \mathcal{A}$ is equipped with a state $\tau_a(t)$ which depends on t. In the rest of this chapter, we will use the notation τ_a, dropping the dependence on t whenever it is clear from the context. As we will see, the state of an arm fully determines its expected reward. Before discussing the reward function, we take some time to clarify the concept of state, how it is defined, and how to interpret it.
The state is used to indicate the last time the arm took part in a *switch of actions*. With *switch of actions* we refer to the tuple $[a_{t-1}, a_t]$ which indicates the switch between action a_{t-1} played in the previous round $t-1$ and action a_t played in the current round t. More specifically, we define the state as:

$$\tau_a(0) = 1 \quad \text{and} \quad \tau_a(t+1) = \delta_a(\tau_a(t), a_t), \tag{4.1}$$

where δ_a is the transition function given by:

$$\Delta_a(\tau, a') = \begin{cases} \tau - 1 & \text{if } a' = a,\ \tau \leq 0 \\ -1 & \text{if } a' = a,\ \tau \geq 0 \\ 1 & \text{if } a' \neq a,\ \tau \leq 0 \\ \tau + 1 & \text{if } a' \neq a,\ \tau \geq 0. \end{cases} \tag{4.2}$$

To better understand the previous equations, we present Figure 4.1. Assisted by Figure 4.1, we show how the states evolve. At the beginning of the sequential

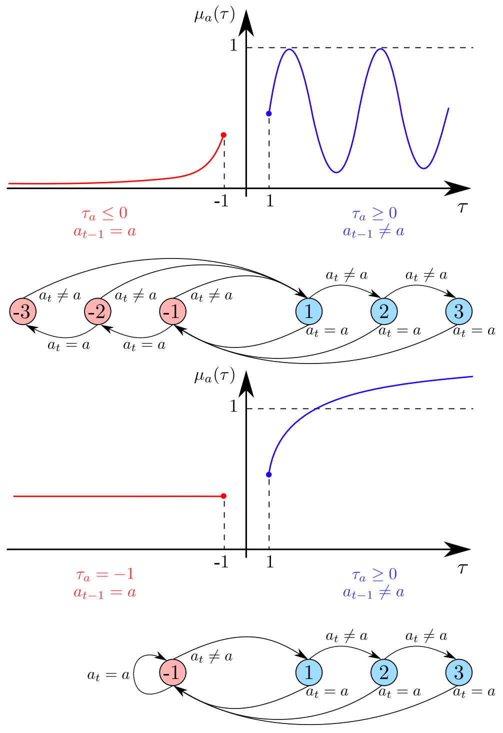

FIGURE 4.1: In this figure we show examples of an expected reward function μ_a and transition mechanisms (reduced to 6 and 4 states) for LSD Bandits (left) and models with delay-dependent rewards [Kleinberg and Immorlica, 2018, e.g.] (right).

game, when $t = 0$, each action has a state equal to 1, $\tau_a(0) = 1 \; \forall a \in \mathcal{A}$. Then, in the following round, if arm $a^{(i)}$ is played, its state will get to $\tau_{a^{(i)}}(1) = -1$. For all the other actions in $\mathcal{A}$ which are not played, the state will be incremented by 1, having $\tau_{a^{(j)}}(1) = 2$ for $a^{(j)} \neq a_1, a^{(j)} \in \mathcal{A}$. If at time t arm $a^{(i)}$ has been consecutively played for the last n time steps, then its state will be $\tau_{a^{(i)}}(t) = -n$. Otherwise, if any arm $a^{(j)}$ has not been played for the last n time steps, then its state will be $\tau_{a^{(j)}}(t) = n$. This can be simply understood by saying that whenever the state $\tau_a(t)$ of the arm is negative, the absolute value of the state $\tau_a(t)$ indicates how many times steps this action has been consecutively pulled by the learner. Otherwise, if the state is positive, its value $\tau_a(t) = n$ indicates that the action has not been played for the last n rounds. This mechanism is represented by the plot on the left of Figure 4.1. Using these definitions, it is possible to model both satiation and seasonality. The first is modelled by the negative part of the reward function μ_a using negative states, the second is represented by the positive part of the reward function related to positive states. On the right of Figure 4.1, we compare our mechanism with the one presented in other delay-dependent rewards (e.g. Kleinberg and Immorlica [2018]). As we can notice, many models also use the concept of states to indicate how many time steps in the past the action was last pulled. However, these works usually consider only positive states to indicate how many rounds have passed since the last pull of an arm and consider a single state, equal to 0 or -1, to indicate that the action played at time t is the same as the one played in the previous step $(t-1)$. Thus, these mechanisms do not keep track of the number of consecutive

plays of an arm, which is one of the strengths of this model. Indeed, it is because of this feature that it is possible to model satiation and seasonality in a detailed way.
We use these concepts of switch of actions and state $\tau_a(t)$ to define the expected reward of the actions. For this reason, we call our model *Last Switch Dependent (LSD) bandit*. As we anticipated, in a LSD bandit the expected reward of an arm a is fully determined by its state $\tau_a(t)$. More specifically, for every action $a \in \mathcal{A}$ there exists an unknown function $\mu_a : \mathbb{Z} \to [0,1]$. There are only two assumptions on the reward functions: to be bounded in $[0,1]$ and non-decreasing on $\mathbb{Z}^-$. The expected reward of arm a at time t is given by $\mu_a(\tau_a)$, where τ_a is the state of arm a at time t defined in Equation (4.1) and Equation (4.2).

Definition 4.1 (LSD Bandit)**.** *A stochastic bandit with action set $\mathcal{A}$ is a LSD bandit if for every action $a \in \mathcal{A}$ there exists an (unknown) function $\mu_a \colon \mathbb{Z} \to [0,1]$, nondecreasing on $\mathbb{Z}^-$, such that the expected reward of arm a is given by $\mu_a(\tau_a)$, where τ_a is the* last switch *state of a, as defined in* (4.1) *and* (4.2)*.*

For $t = 1, \dots, T$ the sequential protocol will be the following:

1. the learner selects an action $a_t \in \mathcal{A}$ to play,
2. the learner receives a stochastic reward X_t with an expected value $\mathbb{E}[X_t] = \mu_{a_t}(\tau_{a_t}(t))$,
3. the states of every $a \in \mathcal{A}$ will be updated: $\forall a \in \mathcal{A}$, $\tau_a(t+1) = \delta_a(\tau_a(t), a_t)$.

The goal is to maximize the expected cumulative reward, measuring the performance of an algorithm by the regret defined as:

$$R_t = \sum_{t=1}^{T} \mu_{a_t^*}(\tau_{a_t^*}(t)) - \mathbb{E}\Big[\sum_{t=1}^{T} \mu_{a_t}(\tau_{a_t}(t))\Big], \tag{4.3}$$

where the expected cumulative regret of the learner's policy is compared to the cumulative reward of $[a_1^*, \dots, a_T^*]$, which is the optimal sequence of actions, the sequence of actions which maximizes the expected sum of rewards over the horizon $1, \dots, T$. We highlight how we measure the performance of our algorithm against the best possible trajectory, instead of a restricted class of competitors, such as in the instantaneous regret, or a d-step lookahead regret, as in other non-stationary bandits discussed in Chapter 3.
The primary aspect of our model is that our definition of the states allows us to represent two different behaviours, satiation and seasonality, in a single framework. Furthermore, this is done by dropping many assumptions on the reward function, such as concavity, Lipschitzness, and monotonicity, which are assumptions exploited by many other works in the literature (e.g. Kleinberg and Immorlica [2018], Metelli et al. [2022], Seznec et al. [2020], and many other works presented in Chapter 3). Compared to other works, our model is neither restless nor rested. In fact, here the changes depend on the policy played by the learner, but, at the same time, the reward of an arm changes whether it is pulled or not. Out of the different categories of non-stationary bandits

we described in Chapter 3, the LSD bandit model best fits into the category of state-dependent bandits (Section 3.2.2). Nonetheless, LSD bandits differ from the model presented in Section 3.2.2. As we said, in LSD bandits the assumptions of concavity, Lipschitzness, and increasing monotonicity on the reward function, proposed in Kleinberg and Immorlica [2018], are dropped. The fact that we relax the concavity assumption has a drastic effect on our model. Indeed, for concave reward functions, oracle greedy achieves a 1/2-approximation of the optimal policy Kleinberg and Immorlica [2018], but when this assumption is dropped we show in Example 4.1 that oracle greedy can be arbitrarily bad in our setting. This shows how LSD bandits can be more difficult than recharging bandits. Furthermore, we do not assume a specific form of the reward function, unlike Cella and Cesa-Bianchi [2020], or that this function is drawn from Gaussian Process with known kernel, unlike Pike-Burke and Grunewalder [2019]. The arm-dependent nature of our reward function enables us to model different behaviours associated with different arms within the same bandit instance, without limiting a single bandit instance to represent only one type of non-stationarity. We also rely on the standard assumption that in each time step the learner is allowed to play a single action, whereas in Simchi-Levi et al. [2021] the learner selects more arms in one round. Unlike Basu et al. [2019], we do not block actions and remove them from the actions set, in every time steps every action a is always available, with a different reward function depending on its arm-specific reward function μ_a and its state $\tau_a(t)$. We can state that LSD bandits generalize state-dependent bandits (e.g. Kleinberg and Immorlica [2018], Pike-Burke and Grunewalder [2019], Cella and Cesa-Bianchi [2020]). The positive branch of the reward function in LSD bandits can recover the behaviour of other state-dependent models which consider the delay between the current time step and the last time step when the action was played. On the other side, the LSD bandit model is also able to generalize these state-dependent bandits when an action is consecutively pulled. Indeed, in state-dependent bandits, such as Kleinberg and Immorlica [2018], the consecutive plays of an arm retrieve the same expected reward, independently of the number of plays of that arm. This is represented in the plot on the right in Figure 4.1, where this behaviour is represented as a constant function over the negative states. However, LSD bandits are also able to represent progressive satiation, which is a behaviour that other models did not consider. In fact, on the negative branch $\mathbb{Z}^-$ of the function μ_a LSD bandits can model a non-decreasing trend over the negative states, where the function can take different values depending on the specific number of consecutive plays of the arm as long as it satisfies the non-decreasing assumption. While in other models such as Kleinberg and Immorlica [2018] the expected reward for playing arm a consecutively for 1, 10 or 100 times remains the same, in LSD bandits the model can distinguish between the expected reward of arm a played consecutively for 1, 10 or 100 times. We realize that there is still an assumption of monotonicity on $\mathbb{Z}^-$ which does not allow the reward to increase as the number of pulls increases. However, we believe that this assumption is natural in most scenarios, as it seems reasonable to think that the more an item is presented to the learner the more its reward will decrease. We justify this assumption by considering

the applications that frame this model and the necessity of this assumption to derive non-vacuous approximations (see 4.1).
Before showing some results about the difficulty of our setting, we present an example where we show how oracle greedy is not optimal in LSD bandits.

Example 4.1. *Consider the LSD bandit defined by the following reward functions: $\mu_1(\tau) = \epsilon + (1-\epsilon)\mathbb{I}\{\tau \geq 1\}$, and $\mu_2(\tau) = 0\ \forall \tau$. An oracle greedy strategy, which pulls at each time step the arm with highest expected reward (assuming the knowledge of the μ_a) would always pull arm $a^{(1)}$ obtaining a reward of $1 + (T-1)\epsilon$ after T rounds. Instead, the optimal policy alternates between arms $a^{(1)}$ and $a^{(2)}$ and gets a $T/2$ overall reward. By making ϵ arbitrary close to 0, we can thus make oracle greedy arbitrarily bad. We conclude with a remark on why this example would be ruled out by concavity. In Kleinberg and Immorlica [2018], concavity is actually defined with respect to the origin, such that the reward obtained in case of consecutive pulls (here it is ϵ) is considered as an increment from 0. Concavity then prevents the next increments from being bigger, while here it is equal to $1-\epsilon$, which is greater than ϵ as soon as $\epsilon < 1/2$. And for $\epsilon \geq 1/2$, one can note that oracle greedy is indeed a $1/2$-approximation of the optimal strategy, as revealed by the above computations.*

With the previous example, we show that our problem can be more difficult than recharging bandits, as oracle greedy can perform arbitrarily bad. Starting from this evidence, we studied the hardness of solving LSD bandits. Building on the proof used in Theorem 3.1 of Basu et al. [2019] and discussed in Section 3.2.2, we prove Proposition 4.2.

Proposition 4.2. *Computing the optimal policy for LSD bandits is NP-hard.*

Before presenting the proof of Proposition 4.2, we define some notation which will be useful for the rest of this chapter. We use the term $B = [a_1, \dots, a_d]$ to define a sequence of actions, also called block of actions, and $\boldsymbol{\tau} \in \mathbb{Z}^K$ to denote the array of states of the actions, where the i-th element of $\boldsymbol{\tau}$ is $\tau_i, \forall i \in \{1, \dots, K\}$. When we write $r(B|\tau_{init}) = \sum_{t=1}^{d} \mu_{a_t}(\tau_{a_t}(t))$ we refer to the sum of the expected rewards obtained by playing the sequence of actions B when the states of the actions before the first play of a_1 in B are initialized to τ_{init}, so $\tau(1) = \tau_{init}$.

Proof. First of all, we consider reward functions μ_i that are constant functions on $\mathbb{Z}^-$, since it is enough to prove NP-hardness. In this proof, we show a more general result than the statement of Proposition 4.2. Specifically, we prove that for any $\tau_{init} \in \mathbb{N}^K$, the following computation:

$$B^* = \underset{|B|=d}{\operatorname{argmax}}\, r(B|\tau_{init}) \tag{4.4}$$

is NP-hard as soon as d is greater than a certain value d_0, which will be defined later on. Note that when $\tau_{init} = \mathbf{1}$ and $d = T$, solving Equation (4.4) corresponds to finding the best policy. When T is large enough, we get $d \geq d_0$ and computing the optimal policy is NP-hard. The interesting aspect of this result is that it highlights that the inner optimization problem of the `CombUCB1` approach

(which will be explained later on) is itself NP-hard. The same result holds for the optimization problem in our algorithm `ISI-CombUCB1`, since the steps we are showing work when one substitutes r with $\widetilde{r}$, which will be defined later on in Equation (4.22). As in Basu et al. [2019], we consider the Pinwheel Scheduling Problem (PSP) Holte et al. [1989]. The problem considers a set of actions $\{a^{(1)}, \dots, a^{(K)}\}$, each one associated with a delay $d_i \in \mathbb{N}$ for $i = 1, \dots, K$, and consists in deciding if it is possible or not to design a schedule for the actions, formally a mapping $\mathbb{N} \to \{1, \dots, K\}$, such that each arm $a^{(i)}$ appears at least in any sequence of d_i consecutive steps. When a PSP instance allows for this schedule, then it is called a YES instance. Otherwise, in case it is not possible to provide a schedule which respects this constraint, it is called a NO instance. Furthermore, when a PSP instance is called dense when $\sum_{i=1}^{K} 1/d_i = 1$. This means that if a PSP instance is a YES instance and is also dense, it implies that each arm appears exactly every d_i steps. In Bar-Noy et al. [2002] the authors show that solving a dense PSP instance is NP-complete. Here, we show that PSP on dense instances reduces to particular instances of our problem, proving the hardness of LSD bandits.
To show this reduction, we consider an instance of LSD bandits with K actions and where each action is associated with a reward function:

$$\mu_i(\tau_i) = \mathbb{I}\{\tau_i \geq d_i\}. \tag{4.5}$$

We introduce an additional arm $K + 1$ such that $\mu_{K+1}(\tau_{K+1}) = 0, \forall \tau$. Given an initial state τ_{init}, the aim is to identify the best block of actions $B^* = \text{argmax}_{|B|=d}\, r(B|\tau_{init})$. There are two possibilities:

- if the PSP is a YES instance, the PSP schedule obtains a reward of 1 at each time step. Since the initial state τ_{init} might not exactly suit the schedule, we obtain a reward $r(B^*|\tau_{init}, YES) \geq d - K$.
- if the PSP is a NO instance, we can use the argument proposed in Basu et al. [2019] and prove that $r(B^*|\tau_{init}, NO) \geq d - \lfloor \frac{d}{\Pi_{i=1}^{K} d_i} \rfloor$.

Hence, for $d \geq d_0 := (K + 1)\Pi_{i=1}^{K} d_i$, if we were able to compute the best block B^*, and therefore $r(B^*|\tau_{init})$, this would mean we could discriminate between a YES and a NO instance, depending on whether $r(B^*|\tau_{init}) \geq d - K$ or not. Thus, solving the PSP on dense instances reduces to solving Equation (4.4) for the particular choice of reward functions defined in Equation (4.5), therefore proving the NP-hardness of LSD bandits.
As we anticipated, our proof is largely based on the one presented in Basu et al. [2019]. Indeed, our framework encapsulates the setting described by Equation (4.5), which is the one analyzed by the authors to show the hardness of their problem. There are two main differences between the two proofs. First, in our setting we need to take into account the initial state τ_{init}, which can be handled by considering a larger block size d. The second difference is that in Basu et al. [2019] any empty slot in the schedule is automatically filled with a pull of the 0 arm, which is the only arm that can be played, while in our proof we have to invoke a finer argument, namely that at any empty slot, playing the

0 arm is the best possible action, since all actions yield a null expected reward, but playing the 0 arm allows every other arm to recharge. □

With this proof, we showed that computing the global optimal policy for LSD bandits is NP-hard, even with full knowledge of the problem instance. Thus, a common approach to tackle this obstacle is to consider a restricted and simpler class of approximating policies and learn the optimal policy among these. For LSD bandits, a natural approximating class is the set of policies with cyclic play sequences. A play sequence $a_1, a_2, \dots$ is cyclic if there exists t_0 and $d > 0$ such that $\forall t \geq t_0$ it holds that $a_{t+d} = a_t$. We state the following Lemma, where we show how cyclic policies are optimal for 2-armed LSD bandits.

Lemma 4.3. *For a LSD bandit with two arms, any deterministic policy induces a sequence of pulls which is cyclic from a certain time step onward. This holds in particular for the optimal deterministic policy.*

Proof. To prove Lemma 4.3, we consider a 2-armed bandit and assume without loss of generality that $a^{(1)}$ is the first action played by the learner. We analyze the possible cases. If there is no switch of actions, then $a^{(1)}$ is played continuously, meaning that the sequence is cyclic. Then, we consider the cases when one or more switches happen. If there is only one switch of actions, then the learner plays $a^{(1)}$ until a certain point where the switch happens and $a^{(2)}$ is played indefinitely after that point, from which the sequence is cyclic. If there are only two switches, it means that after playing $a^{(1)}$ and $a^{(2)}$ the learner will go back to play $a^{(1)}$ indefinitely after some time step, from which the sequence will be again cyclic. If there are three (or more) switches of actions, then there are two switches of actions from $a^{(1)}$ to $a^{(2)}$ in the sequence, meaning that the state $(\tau_1, \tau_2) = (1, -1)$ will be visited twice. Since the policy is deterministic, the same cycle will be repeated. □

Although we proved that any deterministic policy induces a cyclic policy in a 2-armed LSD bandit from a certain step onward, we highlight how this property does not resist to the number of arms. Indeed, we can exhibit optimal policies for 3-armed bandits which do not induce cyclic sequences, showing a policy which never visits the same state twice. We consider a 3-armed LSD bandit with an actions set $\mathcal{A} = \{a^{(1)}, a^{(2)}, a^{(3)}\}$. We analyze the following sequence of actions. For $m = 1, 2, \dots$, the learner plays:

- $a^{(1)}$ consecutively for m times
- $a^{(2)}$ consecutively for m times
- $a^{(3)}$ consecutively for m times.

For simplicity, we assume that the states are all initialized at $\tau_i = 0 \, \forall i = \{1, 2, 3\}$ and not to 1. We allow for this simplification since it only impacts the states in the first block, for the first time they are played. Following this policy, we have that in block m, the states of the actions are given by:

$$(\tau_1, \tau_2, \tau_3) = \begin{cases} (-t, m-1+t, t) & \text{after the } t\text{-th pull of } a^{(1)} \\ (t, -t, m+t) & \text{after the } t\text{-th pull of } a^{(2)} \\ (m+t, t, -t) & \text{after the } t\text{-th pull of } a^{(3)}. \end{cases} \tag{4.6}$$

Using the previous formula, it is easy to see that no state is visited twice. Then, it is easy to choose μ_1, μ_2, μ_3 such that the policy is optimal: for instance, $\mu_i = 1 \ \forall i = 1, 2, 3$. We clarify that this does not imply that there does not exist a cyclic optimal policy for a 3-armed bandit, but rather that not every optimal policy is cyclic. It is rather difficult to answer this question, which is deferred to future work. Nevertheless, we state a weaker result to show that cyclic policies can almost reach the best average reward up to a term which is inversely proportional to the cycle length. We state this in Proposition 4.4.

Proposition 4.4. *Let $(\mu_i)_{i=1}^K$ be an LSD bandit with K arms and constant expected rewards on $\mathbb{Z}^-$. Let $T \geq 0$ be the horizon and $d \geq 0$ that divides T. Then, there exists a cyclic policy π with cycle length d (and $t_0 = 1$) such that*

$$\frac{1}{T}\sum_{t=1}^{T} r_\pi(t) \geq \frac{1}{T}\sum_{t=1}^{T} r^*(t) - \frac{K}{d} \tag{4.7}$$

where $r_\pi(t)$ and $r^*(t)$ are the expected rewards obtained at time step t by π and the optimal policy, respectively. When the μ_i are not constant on $\mathbb{Z}^-$, (4.7) holds with $K+2$ instead of K in the right-hand side. Note also that (4.7) requires $d \geq K$ (respectively $d \geq K+2$) to be non-vacuous, as we have: $0 \leq r^*(t) \leq 1$.

This shows how cyclic policies give good constant factor approximations in general. This would mean that to simplify the problem, the learner could restrict the search space to cyclic policies and look for the best policy inside this comparator class, controlling the approximation error using the cycle length. However, this strategy is not viable either since finding the optimal cyclic policy for LSD bandits remains NP-hard, even when arms are totally ordered. Before proving Proposition 4.4, we state and prove the following proposition.

Proposition 4.5. *Finding the optimal cyclic policy for LSD bandit problems is NP-hard, even with separable reward functions of the form $\mu_i(\tau) = \mu_i^0 \cdot \gamma(\tau)$.*

Proof. Similarly to Proposition 4.2, it is sufficient to consider only the case where the reward functions are constant on $\mathbb{Z}^-$, and actually prove here a stronger result than Proposition 4.5. Namely, we show that for some d, computing

$$B^* = \underset{|B|=d}{\operatorname{argmax}} \ r(B \mid \boldsymbol{\tau}_B) \tag{4.8}$$

where $\boldsymbol{\tau}_B$ is the state reached by the system after a play of block B, is NP-hard, even when the reward functions can be totally ordered. The optimal cyclic policy (with cycle length d) is obtained by repeating indefinitely the solution to (4.8), and the NP-hardness of the latter problem then yields Proposition 4.5. This proof is also based on a reduction of the Pinwheel Scheduling Problem (PSP), see Proposition 4.2. Given a PSP dense instance $(d_i)_{i=1}^K$, we construct an instance of our problem as follows. For every arm $i = 1, \ldots, K$, we set:

$$\mu_i(\tau) = \sqrt{\tau / d_i}\,. \tag{4.9}$$

Note that setting Equation (4.9) is fundamentally different from setting (4.5) as we have a total ordering of the arms, i.e., for all τ we have $\mu_1(\tau) \geq \mu_2(\tau) \geq$

$\ldots \geq \mu_K(\tau)$, with the convention $d_1 \leq d_2 \leq \ldots \leq d_K$. We also highlight that we do not introduce an additional null arm here, as opposed to (4.9). Finally, although the μ_i are unbounded for the moment, thus breaking Definition 4.1, we see at the end of the proof how to consider bounded μ_i without altering the subsequent analysis.

We now show that solving (4.8) with reward functions (4.9) allows us to determine if the PSP instance is a YES or a NO instance. Let n_i be the number of times action $i = 1, \ldots, K$ is played in the block, and τ_{ij}, for $j = 1, \ldots, n_i$ be the different states in which arm i is pulled. Problem (4.8) can be rephrased as:

$$\max_{(n_i)_{i=1}^K} \max_{(\tau_{ij})_{i=1,j=1}^{K,n_i}} \sum_{i=1}^{K} \sum_{j=1}^{n_i} \sqrt{\tau_{ij}/d_i} \qquad \text{subject to } \begin{cases} \sum_{i=1}^K n_i = d \\ \sum_{j=1}^{n_i} \tau_{ij} = d, i = 1, \ldots, K \end{cases} \tag{4.10}$$

We start by maximizing with respect to the $\boldsymbol{\tau} = (\tau_{ij})_{i=1,j=1}^{K,n_i}$. The Lagrangian writes

$$\mathcal{L}(\boldsymbol{\tau}, \boldsymbol{\lambda}) = \sum_{i=1}^{K} \sum_{j=1}^{n_i} \sqrt{\tau_{ij}/d_i} + \sum_{i=1}^{K} \lambda_i \left(\sum_{j=1}^{n_i} \tau_{ij} - d \right) .$$

The KKT conditions (gradient of the Lagrangian and primal feasibility) write

$$\frac{\partial \mathcal{L}(\boldsymbol{\tau}, \lambda)}{\partial \tau_{ij}} = \frac{1}{2\sqrt{d_i \tau_{ij}}} + \lambda_i = 0 \quad i = 1, \ldots, K,\ j = 1, \ldots, d \tag{4.11}$$

$$\sum_{j=1}^{n_i} \tau_{ij} = d \qquad i = 1, \ldots, K. \tag{4.12}$$

Solving (4.11) for τ_{ij} we obtain a quantity independent of j. Then (4.12) implies that $\tau_{ij} = d/n_i$ for each $i = 1, \ldots, K$. Replacing $\tau_{ij} = d/n_i$ into (4.10), we can now maximize with respect to the n_i. The Lagrangian writes

$$\mathcal{L}(\boldsymbol{n}, \lambda) = \sum_{i=1}^{K} \sqrt{d n_i / d_i} + \lambda \left(\sum_{i=1}^{K} n_i - d \right)$$

and the KKT conditions (gradient of the Lagrangian and primal feasibility) are

$$\frac{\partial \mathcal{L}(\boldsymbol{n}, \lambda)}{\partial n_i} = \sqrt{\frac{d}{4 n_i d_i}} + \lambda = 0 \qquad i = 1, \ldots, K \tag{4.13}$$

$$\sum_{i=1}^{K} n_i = d \tag{4.14}$$

such that replacing (4.13) into (4.14), we obtain $n_i = d/d_i$, which implies $\tau_{ij} = d_i$.

Assume now that d can be divided by all the d_i, such that $n_i = d/d_i \in \mathbb{N}$. From the values of n_i and τ_{ij} obtained, we can see that the optimal block for (4.8) corresponds to a Pinwheel schedule. It yields an average reward equal of 1, and is achievable if and only if the Pinwheel instance is a YES instance.

Therefore, if we can solve (4.8), we can tell if the average reward is equal to 1 or smaller, and thus decide whether the instance is a YES or NO instance. We have reduced PSP on dense instances to (4.8), which is therefore shown to be NP-hard. To our knowledge, this is the first hardness result for decomposable reward functions of the form $\mu_i(\tau) = \mu_i^0 \cdot \gamma(\tau)$.
Now, let $d_{\max} = \max_{i=1,\dots,K} d_i$. Note that replacing (4.9) with the bounded functions

$$\mu_i(\tau) = \begin{cases} \sqrt{\tau/d_i} & \text{if } \tau \le d_{\max} \\ \sqrt{d_{\max}/d_i} & \text{otherwise} \end{cases}$$

does not change change the optimal schedule (arm i is played every $d_i \le d_{\max}$ time steps) and the analysis is unchanged. □

After showing that finding the optimal cyclic policy is NP-hard even when the reward functions are separable and ordered, we dedicate the rest of this section to state and prove Lemma 4.6, which will be used in the proof of Proposition 4.4. Then, we conclude this section with an example to show the tightness of our analysis.
We start with the discussion and proof of Proposition 4.4. As mentioned before, we denote with $B = [a^{(1)}, \dots, a_d]$ a general block of d actions and with $r(B|\tau_{init}) = \sum_{t=1}^{d} \mu_{a_t}(\tau_{a_t}(t))$ the sum of expected rewards obtained by playing the actions in block B when $\boldsymbol{\tau}$ at time $t = 1$ inside the block is initialized to $\boldsymbol{\tau}(1) = \tau_{init}$. In order to prove Proposition 4.4, we state and prove the following lemma.

Lemma 4.6. *Let $(\mu_i)_{i=1}^K$ be an LSD bandit with K arms and constant expected rewards on $\mathbb{Z}^-$. Then, for all block B and any initial states $\boldsymbol{\tau}_{\text{init}}, \boldsymbol{\tau}'_{\text{init}} \in \mathbb{Z}^K$, we have*

$$r(B \mid \boldsymbol{\tau}'_{\text{init}}) \ge r(B \mid \boldsymbol{\tau}_{\text{init}}) - K \ . \tag{4.15}$$

If the μ_i are not constant on $\mathbb{Z}^-$, a slight modification of B in the left-hand side is required to maintain a similar bound. Formally, for any block B of length d, there exists a block B' of length d such that for any initial states $\boldsymbol{\tau}_{\text{init}}, \boldsymbol{\tau}'_{\text{init}}$, we have

$$r(B' \mid \boldsymbol{\tau}'_{\text{init}}) \ge r(B \mid \boldsymbol{\tau}_{\text{init}}) - (K+2) \ . \tag{4.16}$$

Proof. Here we present the proof of Lemma 4.6. We start by proving the special case where the reward functions μ_i are constant on $\mathbb{Z}^-$. First, we notice that after the first play of an arm a, the switch of arms $[a, \text{not } a]$ occurs, and its state will become $\tau_a = 1$, independently of what its initial state was. Then, let t_a be the round at which arm a is pulled for the first time. Even in the case where arm a is consecutively pulled many times, the reward collected for τ_{init} and τ'_{init} are $\mu_a(\tau_{init,a} + t_a - 1), \mu_a(-1), \dots, \mu_a(-1)$ and $\mu_a(\tau'_{init,a} + t_a - 1), \mu_a(-1), \dots, \mu_a(-1)$ since the reward functions μ_i are constant on $\mathbb{Z}^-$. This means that the only difference is $\mu_a(\tau'_{init,a} + t_a - 1) - \mu_a(\tau_{init,a} + t_a - 1) \le 1$. So, considering the cardinality K of the set of arms, the total difference cannot exceed K, which leads to the result in Equation (4.15). We specify that in full generality, to be even more specific, the quantity K could be replaced by the number of different arms played in B.

Now, we move to the general proof for reward functions μ_i which are not constant on $\mathbb{Z}^-$. For any block of actions B of size d, we want to find a block B' of length d such that for any initial states $\tau_{init}, \tau'_{init} \in \mathbb{Z}^K$, we have:

$$r(B'|\tau'_{init}) \geq r(B|\tau_{init}) - (K+2). \tag{4.17}$$

We first analyze what happens when we replace B' with the same B in Equation (4.17). As for the constant case, we recall that when playing a block of actions, the arm is impacted by the initial state only in its first pull inside the block (possibly with several consecutive pulls). When this sequence (which might be of length 1 if a switch follows the first pull) is interrupted, the arm's state goes to 1, independently of the state it was before the switch. Furthermore, note that only the first action a_1 in the block can be pulled with a negative state at the beginning of its sequence of pulls. Let $\tau_{new} = \boldsymbol{\delta}(\tau_{init}, a_1)$ where $\boldsymbol{\delta}$ is a componentwise version of δ_a and a_1, as we said, is the action pulled in the first time step of block B. It is easy to check that for all arms $a \neq a_1$, we have that $\tau_{new,a} \geq 1$. Indeed, there are two cases. If the action was in a negative state, then its state becomes 1 when action $a_1 \neq a$ is played as the first action in B. Otherwise, if action a was in a positive state, this state is incremented by 1. This leads to say that the expected rewards obtained during the first play (possibly with consecutive pulls) of an arm a with initial states τ_{init} and τ'_{init} are respectively

$$\mu_a(\tau_a),\, \mu_a(-1),\, \mu_a(-2), \ldots \qquad \text{and} \qquad \mu_a(\tau'_a),\, \mu_a(-1),\, \mu_a(-2), \ldots \tag{4.18}$$

where τ_a and τ'_a are two generic **positive** values (thanks to the remark we made earlier) which depend on $\tau_{\text{init},a}$, $\tau'_{\text{init},a}$, and the place of the first pull of a in the block. Note that the assumption on the boundedness of μ_a ensures that the difference of expected rewards obtained is smaller than 1.
Now, we analyze the first pull of arm a_1. We start by assumin that it is pulled for n_1 consecutive rounds at the beginning of block B (note, we might have $n_1 = 1$). Assuming that $\tau_{init,1}$ is positive and $\tau'_{init,1}$ is negative, the expected rewards collected are respectively

$$\begin{aligned} &\mu_1(\tau_{\text{init},1}),\, \mu_1(-1)\, \ldots\, \mu_1(-n_1+1) \qquad \text{and} \\ &\mu_1(\tau'_{\text{init},1}),\, \mu_1(\tau'_{\text{init},1}-1)\, \ldots\, \mu_1(\tau'_{\text{init},1}-n_1+1)\,. \end{aligned} \tag{4.19}$$

Unlike in the previous case, the difference between the two sequences is not bounded by 1. Indeed, it might be even equal to n_1 if $\tau'_{init,1} \leq -n_1$. This explains why the same B cannot be used in both sides of Equation (4.17). In order to break the sequence of pulls and bound this difference, we introduce B'. Let a_2 be the second action pulled in B (if B consists of a sequence where all the d actions are a_1, we can set a_2 to be any action in $\mathcal{A}$ different from a_1). We assume that the block B' is equal to B except for the second pull of B', where the action pulled in the second time step is necessarily a_2. We can face three different cases:

- If $n_1 = 1$, we have that $B' = B$. Since $n_1 = 1$, the difference between the two sequences of rewards due to the pulls of a_1 is at most 1. For all the other actions in the block, we refer to the analysis presented at the beginning and the total difference is at most K.

- If $n_1 = 2$, and denoted by n_2 the number of times a_2 is played consecutively after a_1, then the expected rewards of the two sequences obtained by playing B and B' with initial state τ_{init} and τ'_{init} respectively are

$$\begin{array}{lllllll} & \mu_1(\tau_{\text{init},1}), & \mu_1(-1) \text{ or } \mu_1(\tau_{\text{init},1}-1), & \mu_2(\tau_2), & \mu_2(-1) & \dots & \mu_2(-n_2+1) \\ \text{and} & \mu_1(\tau'_{\text{init},1}), & \mu_2(\tau'_2), & \mu_2(-1), & \mu_2(-2) & \dots & \mu_2(-n_2) \end{array}$$

where the *or* comes from the fact that we don't know if $\tau_{\text{init},1}$ is positive (then the next reward is $\mu_1(-1)$) or negative (then next reward is $\mu_1(\tau_{\text{init},1}-1)$). Here, τ_2 and τ'_2 are two generic positive numbers, whose values are not important as the difference between the two sequences is contained in the red rewards, and thus bounded by 3 anyway. For all other $K-2$ actions, we can apply the standard analysis, such that in total the difference cannot exceed $K+1$.

- $n_1 \geq 3$. Using the same notation as above, we have now respectively for the first n_1 pulls of B

$$\mu_1(\tau_{\text{init},1}), \quad \mu_1(-1) \text{ or } \mu_1(\tau_{\text{init},1}-1), \quad \mu_1(-2) \text{ or } \mu_1(\tau_{\text{init},1}-2), \quad \dots \quad \mu_1(-n_1+1) \text{ or } \mu_1(\tau_{\text{init},1}-n_1+1)$$

and for B'

$$\mu_1(\tau'_{\text{init},1}), \quad \mu_2(\tau'_2), \quad \mu_1(1), \quad \dots \quad \mu_1(-n_1+3).$$

The red terms incur a loss of at most 3. Then, if $\tau_{\text{init},1} \geq 1$, the remaining rewards (non red) in the play of B are $\sum_{j=3}^{n_1-1} \mu_1(-j) \leq \sum_{j=1}^{n_1-3} \mu_1(-j)$ as μ_1 is nondecreasing on $\mathbb{Z}^-$, so the rewards obtained by B' are greater. If $\tau_{\text{init},1} < 0$, we have $\sum_{j=3}^{n_1-1} \mu_1(\tau_{\text{init},1}-j) \leq \sum_{j=1}^{n_1-3} \mu_1(-j)$ for the same reasons. So overall, the difference is bounded by 3.

As for the n_2 following pulls, recalling that τ_2 and τ'_2 are positive since the preceding pull is $a^{(1)}$, we have

$$\begin{array}{lllll} & \mu_2(\tau_2), & \mu_2(-1), & \dots & \mu_2(-n_2+1) \\ \text{and} & \mu_2(\tau'_2), & \mu_2(-1), & \dots & \mu_2(-n_2+1) \end{array}$$

with a difference bounded by 1. For the $K-2$ other actions, we still have the bound of $K-2$, so that in total the difference does not exceed $K+2$.

□

Lemma 4.6 tells us that playing the same block with different initials states yields a difference in the sequence of rewards which is bounded by K, when the reward functions are constant on $\mathbb{Z}^-$, or $K+2$, when we drop the assumption that the reward functions are constant on $\mathbb{Z}^-$ ad allow the μ_a to be increasing. We use this lemma to prove Proposition 4.4, where we show the approximation error of cyclic policies.

Proof. We start by considering the case where the reward functions μ_i are constant on $\mathbb{Z}^-$. Note that there exists a time step $t_0 \leq T - d + 1$ such that the average reward obtained by the optimal policy in a sequence of d actions, from time step t_0 to $t_0 + d - 1$, is higher than the average reward of the optimal policy over the entire horizon, namely $(1/d)\sum_{t=t_0}^{t_0+d-1} r^*(t) \geq (1/T)\sum_{t=1}^{T} r^*(t)$.
We denote with $B^* = [a^*_{t_0} \dots a^*_{t_0+d-1}]$ this block of d actions, and with $\boldsymbol{\tau}^*(t)$ the sequence of states generated by the optimal policy, such that $\sum_{t=t_0}^{t_0+d-1} r^*(t) = r(B^* \mid \boldsymbol{\tau}^*(t_0))$. Let π be the policy which plays repeatedly the block B^*. However, except for the first d steps, B^* is played by π with an initial state $\boldsymbol{\tau}_{B^*}$, i.e., the state reached by the system after a play of B^*, which might be different from $\boldsymbol{\tau}^*(t_0)$. Applying Lemma 4.6, and assuming for simplicity that states were initialized in π to $\boldsymbol{\tau}_{B^*}$, we obtain

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T} r_\pi(t) = \frac{r(B^* \mid \boldsymbol{\tau}_{B^*})}{d} &\geq \frac{r(B^* \mid \boldsymbol{\tau}^*(t_0)) - K}{d} \\ &\geq \frac{1}{T}\sum_{t=1}^{T} r^*(t) - \frac{K}{d} \,.\end{aligned}$$

If the μ_i are not constant on $\mathbb{Z}^-$, the repeated block is built using the second part of Lemma 4.6. Now, we move to the general proof where we drop the assumption that the reward functions μ_i are constant on $\mathbb{Z}^-$ and analyze the scenario where the μ_i can be increasing on $\mathbb{Z}^-$. Here, we will use the second result of Lemma 4.6 and the way B' is constructed from B.
Similarly to the constant case, we first use the fact that we know that there exists a block B^* of length d with an average reward (at least) greater than the average reward of the complete optimal sequence. We still use the notation where we have $B^* = [a^*_{t_0} \dots a^*_{t_0+d-1}]$ and

$$\frac{r(B^* \mid \boldsymbol{\tau}^*(t_0))}{d} \geq \frac{1}{T}\sum_{t=1}^{T} r^*_t \,.$$

While previously we could simply repeat B^*, now this is not possible anymore due to the possible effects shown in (4.19). Let B', derived from B as in the proof of Lemma 4.6. Namely, if $a^{(1)}$ and $a^{(2)}$ are the first two different actions played in B, we have $B' = [a^*_{t_0}\, a^{(2)}\, a^*_{t_0+1} \dots a^*_{t_0+d-1}]$. Let $\boldsymbol{\tau}_{B'}$ be the state reached after a play of B' from initial state $\mathbf{1}$. Using Lemma 4.6, the expected rewards $r_\pi(t)$ obtained by policy π which plays cyclically B' satisfy

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T} r_\pi(t) &= \frac{r(B' \mid \mathbf{1})}{T} + \frac{T-d}{T}\,\frac{r(B' \mid \boldsymbol{\tau}_{B'})}{d} \\ &= \frac{d}{T}\,\frac{r(B' \mid \mathbf{1})}{d} + \left(1 - \frac{d}{T}\right)\frac{r(B' \mid \boldsymbol{\tau}_{B'})}{d} \\ &\geq \frac{d}{T}\,\frac{r(B^* \mid \boldsymbol{\tau}^*(t_0)) - (K+2)}{d} + \left(1 - \frac{d}{T}\right)\frac{r(B^* \mid \boldsymbol{\tau}^*(t_0)) - (K+2)}{d}\end{aligned}$$

μ_a	— on $\mathbb{Z}^-$	$\nearrow$ on $\mathbb{Z}^-$	$\sim$ on $\mathbb{Z}^-$
$\nearrow$ on $\mathbb{Z}^+$	$-(K-1)$	$-(K+1)$	$-d$
$\sim$ on $\mathbb{Z}^+$	$-K$	$-(K+2)$	$-d$

TABLE 4.1: In this table we show the approximation errors by block of size d when μ_a is constant (—), nondecreasing ($\nearrow$), or non-monotone ($\sim$) on $\mathbb{Z}^-$ and $\mathbb{Z}^+$. The grey cell represents previous works. Our setting covers the blue ones. The red cells indicate vacuous approximations.

$$
\begin{aligned}
&= \frac{r(B^* \mid \boldsymbol{\tau}^*(t_0)) - (K+2)}{d} \\
&\geq \frac{1}{T}\sum_{t=1}^{T} r_t^* - \frac{K+2}{d}\,.
\end{aligned}
$$

□

As for Lemma 4.6, we provide two results for the cases where the reward functions are constant on $\mathbb{Z}^-$ and where the reward functions are increasing on $\mathbb{Z}^-$. The approximation errors in these cases are respectively K/d and $(K+2)/d$. We notice that the approximation errors by block of size d that can be achieved depend on the assumptions on the μ_i for $\mathbb{Z}^-$ and $\mathbb{Z}^+$. We summarize the results achievable depending on different assumptions in Table 4.1. Note that relaxing the assumptions on $\mathbb{Z}^+$ does not affect much the cost of the approximation. On the other hand, the assumptions on $\mathbb{Z}^-$ can lead to critical changes in the approximation error. Indeed, dropping the monotonicity assumption on $\mathbb{Z}^-$ is critical and would not allow for meaningful results.
Before moving to the discussion about the proposed approach to solve LSD bandit, we conclude these approximation results by showing an example where our analysis is essentially tight.

Example 4.2. *Consider the LSD bandit defined by the following reward functions*

$$\mu_i(\tau) = \mathbb{I}\{\tau \geq K-1\} \qquad \forall i \leq K.$$

The optimal policy consists in repeating the block $[a^{(1)} \dots a^{(K)}]$ and obtains an average reward of 1. Let $d = 2K-1$, such that (up to permutations) we have $B^* = [a^{(1)} \dots a^{(K)} a^{(1)} \dots a^{(K-1)}]$. The average reward of B^* among the optimal sequence is 1, which is equal to the global average reward. Now, it is easy to check that playing repeatedly B^* yields an average reward of $K/(2K-1)$. On the other hand, the lower bound given by Proposition 4.4 is $1 - K/d = (K-1)/(2K-1)$, which matches the average reward as $K \to +\infty$.

In this section, we defined the LSD bandit and studied the hardness of the problem. We showed that finding the optimal policy is NP-hard, as well as finding the optimal policy in the restricted comparator class of cyclic policies, even if it is possible to separate the reward functions as $\mu_i(\tau) = \mu_i^0 \cdot \gamma(\tau)$. However, we show the approximation error of cyclic policies, which can approximate the best average reward for a factor K/d or $(K+2)/d$ if the reward functions are respectively constant or increasing on $\mathbb{Z}^-$. Finally, we concluded this section with an example where this approximation error is tight.

4.3 The estimation problem

After defining and studying the problem, in this section, we illustrate the methods we used to tackle it. From the results obtained in Proposition 4.4, we understand that a possible approach to solve LSD bandits is to approximate the optimal sequence of actions considering cyclic policies. In order to employ a cyclic policy, our approach consists of playing small blocks of actions of size $d < T$. Although the optimal block to play would be B^*, we showed in the proof of Proposition 4.2 (Equation (4.4)) that computing this block constitutes an NP-hard problem, since its estimation would require the computation of the optimal sequence of actions. This identifies the first objective, which is to find a block B of d actions to play. But before planning for the actions inside this block, there is an additional challenge to solve. We recall that the initial state of the block has an impact on the average reward collected by the block. When a "good" block, which attains a high average reward, is played with a different initial state, it may become "bad", achieving a much lower average reward due to the impact of the initial state. So, the second challenge is to study how to handle the impact of the initial state. For simplicity, we analyze the case where the μ_i are constant on $\mathbb{Z}^-$. Once the results for this setting are presented in Proposition 4.7, we move to the analysis where μ_i are not constant on $\mathbb{Z}^-$.
We focus on the problems of finding the optimal block and finding a solution to handle the initial states, which are unknown and impacted by previous plays. A natural way to tackle this problem is to play every block twice. In this case, the learner would play B twice and consider only the rewards obtained by the second block, since these are the rewards that are actually representative of the rewards obtained by the block if played cyclically. We denote with τ_B the state reached by the system after playing block B from initial state 1. Therefore, we define

$$B^*_{double} = \underset{|B|=d}{\operatorname{argmax}}\, r(B|\tau_B), \tag{4.20}$$

which indicates the block of size d which maximizes the reward when played with initial state τ_B. In other words, B^* essentially identifies the block which maximizes the reward of its second play when played twice. As the reader can notice, solving Equation (4.20) is not trivial since B influences not only the rewards generated by the actions but also the initial state. Solving this equation would also correspond to computing the optimal cyclic policy, which has been proven to be NP-hard in Proposition 4.5. Furthermore, playing the same block twice to calibrate the initial state seems superfluous since it would misuse the d time steps of the first of the two blocks. This looks even more excessive when one observes that d needs to grow in order to control the approximation error. For this reason, we disfavour this approach.
A cheaper solution for the calibration of the arms is to play all K arms at the beginning of the block of actions. This would consist in playing a calibration block $B_\sigma = [a_{\sigma(1)}, \dots, a_{\sigma(K)}]$ for any permutation $\sigma \in \Sigma_K$, before playing the

actual block of d actions. This allows the control of the initial state of the actions played after the calibration sequence B_σ. As for the previous approach, the rewards of the calibration sequence could not be exploited. However, if we think of natural applications of our model, we may assume that $d > K$. So, in the previous solution the cost of the calibration may increase consistently because d has to grow. However, in the current scenario, the loss of the learner would decrease to K pulls per block. We denote with τ_σ the state of the system after a play of block B_σ and define

$$B^*_\sigma = \operatorname*{argmax}_{|B|=d} r(B|\tau_\sigma) \tag{4.21}$$

as the block which maximizes the reward obtained by the d actions in B from an initial state τ_σ. This approach would consists in playing cyclically the block $[B_\sigma, B^*_\sigma]$ of size $K + d$. Although there is an improvement if compared to the previous method, some inefficiencies can be observed. The first drawback is that the approximation guarantee is actually worse than the one for double plays. Then, it still seems excessive to dissipate K time steps if not all K arms are actually being played in the d-sized block. Especially, when reasoning about possible applications, e.g. song recommendation, it is impractical to play a representative song for each genre, including genres that the user might not enjoy.

The final solution we propose, which is the one adopted in our solution to LSD bandits, is to reduce the cost of the calibration by ensuring that there is no calibration for those actions that are not played in the block. We simplify this idea by saying that the system discards the rewards obtained each time an action is pulled for the first time in the block, considering only the rewards from its second pull forward. Essentially, the first pull of each action inside the block B is not considered by the learner. Formally, we can define

$$\widetilde{r}(B) = \sum_{t=1}^{d} \mu_{a_t}(\tau_{a_t}(t)) \mathbb{I}\{\exists t_0 < t : a_{t_0} = a_t\}, \tag{4.22}$$

where t indexes the steps inside block B of size d and $\widetilde{r}(B)$ quantifies the total reward obtained by a block B of size d counting the reward of an action only if it is its second pull or higher. The reward obtained by the first pull of an action in B is ignored. An important note is that with this definition the reward $\widetilde{r}(B)$ is independent from the initial state of the block, so the following equation is well defined:

$$\widetilde{B}^* = \operatorname*{argmax}_{|B|=d} \widetilde{r}(B). \tag{4.23}$$

Observe that, because past plays influence the state and therefore future rewards, the first pull of an arm in a block is not reliable due to the missing knowledge of its initial state. Instead of adding a calibration sequence prior to the play of block B, the solution we propose exploits the first pulls to calibrate the arms, without enforcing an excessive and superfluous amount of actions which would not be present in B. Indeed, with this method only the arms actually played in block B are calibrated. Furthermore, there is not a clear separation between the calibration and the exploitation phase. In fact, these are intertwined since there is no obligation to have all the first pulls in the first steps of B. As one can see, this allows a higher degree of freedom when planning for the actions to play in B. Above all, the primary advantage of this solution is that the loss paid in the maximization of $\widetilde{r}$ is controllable, differently from the maximization of r since we have

$$r(B|\tau_{init} - K \leq \widetilde{r}(B) \leq r(B|\tau_{init}), \tag{4.24}$$

for any block B initial state τ_{init}.
What follows this discussion is the statement of the approximation results for all our different approaches.

Proposition 4.7. *Let $(\mu_i)_{i=1}^K$ be a LSD bandit with K arms and constant expected rewards on $\mathbb{Z}^-$. Let $T \geq 0$ be the horizon and $d \geq 0$ that divides T. Let $r_{\text{double}}(t)$ be the expected rewards obtained at time step t by the policy playing cyclically B^*_{double}. We have*

$$\frac{1}{T}\sum_{t=1}^{T} r_{\text{double}}(t) \geq \left(1 - \frac{d}{T}\right)\left(\frac{1}{T}\sum_{t=1}^{T} r^*(t) - \frac{K}{d}\right). \tag{4.25}$$

Let $\sigma \in \mathfrak{S}_K$, and assume that $d+K$ divides T. Let $r_\sigma(t)$ be the expected rewards obtained at time step t by the policy playing cyclically $[B_\sigma, B^*_\sigma]$. We have

$$\frac{1}{T}\sum_{t=1}^{T} r_\sigma(t) \geq \frac{d}{d+K}\left(\frac{1}{T}\sum_{t=1}^{T} r^*(t)\right) - \frac{K}{d+K}. \tag{4.26}$$

Let $\widetilde{r}^*(t)$ be the expected rewards obtained at time step t by the policy playing cyclically $\widetilde{B}^*$. We have

$$\frac{1}{T}\sum_{t=1}^{T} \widetilde{r}^*(t) \geq \frac{1}{T}\sum_{t=1}^{T} r^*(t) - \frac{K}{d}. \tag{4.27}$$

Proof. Recall the notation B^*_{double}, and the additional notation B^* and $\boldsymbol{\tau}^*(t_0)$ introduced in the proof of Proposition 4.4 in the main body of the paper. We have

$$\begin{aligned}
\frac{1}{T}\sum_{t=1}^{T} r_{\text{double}}(t) &= \frac{r(B^*_{\text{double}} \mid \mathbf{1})}{T} + \frac{T-d}{T}\,\frac{r(B^*_{\text{double}} \mid \boldsymbol{\tau}_{B^*_{\text{double}}})}{d} \\
&\geq \left(1 - \frac{d}{T}\right)\frac{r(B^*_{\text{double}} \mid \boldsymbol{\tau}_{B^*_{\text{double}}})}{d} \\
&\geq \left(1 - \frac{d}{T}\right)\frac{r(B^* \mid \boldsymbol{\tau}_{B^*})}{d} \qquad (4.28) \\
&\geq \left(1 - \frac{d}{T}\right)\frac{r\big(B^* \mid \boldsymbol{\tau}^*(t_0)\big) - K}{d} \qquad (4.29)
\end{aligned}$$

$$\geq \left(1 - \frac{d}{T}\right) \left(\frac{1}{T} \sum_{t=1}^{T} r^*(t) - \frac{K}{d}\right) , \tag{4.30}$$

where (4.28) holds because B^*_{double} is a maximizer of $r(B \mid \boldsymbol{\tau}_B)$, (4.29) holds due to Lemma 4.6, and (4.30) is a direct consequence of the definition of B^*. Using similar arguments, we also have

$$\frac{1}{T}\sum_{t=1}^{T} r_\sigma(t) \geq \frac{r(B^*_\sigma \mid \boldsymbol{\tau}_{B_\sigma})}{d+K} \geq \frac{d}{d+K}\,\frac{r(B^* \mid \boldsymbol{\tau}_{B_\sigma})}{d} \geq \frac{d}{d+K}\,\frac{r\big(B^* \mid \boldsymbol{\tau}^*(t_0)\big) - K}{d}$$
$$\geq \frac{d}{d+K}\left(\frac{1}{T}\sum_{t=1}^{T} r^*(t)\right) - \frac{K}{d+K} ,$$

and

$$\frac{1}{T}\sum_{t=1}^{T} \text{Rew}^*(t) \geq \frac{\text{Rew}(\widetilde{B}^*)}{d} \geq \frac{\text{Rew}(B^*)}{d} \geq \frac{r\big(B^* \mid \boldsymbol{\tau}^*(t_0)\big) - K}{d} \geq \frac{1}{T}\sum_{t=1}^{T} r^*(t) - \frac{K}{d} .$$

□

From Proposition 4.7, one can notice how the approximation result for the solution which ignores the first pulls, Equation (4.27), is able to avoid interferences between blocks and is tighter than both the results for the double play approach Equation (4.25) and the calibration sequence approach Equation (4.26) as soon as

$$\frac{1}{T}\sum_{t=1}^{T} r^*(t) \geq \frac{K}{d}, \tag{4.31}$$

which is implicitly assumed for the bounds to be non vacuous. Moreover, the result obtained in Equation (4.25) shows that computing a sequence of blocks with small regret against B^*_{double} requires playing each block twice, with no guarantee that the first play will provide any reward, dividing Equation (4.25) by 2. On the contrary, in the next section we show how to estimate $\widetilde{B}^*$ with tight regret bounds by adapting the `CombUCB1` algorithm from Gai et al. [2012].

Before discussing the estimation problem and the algorithm proposed for LSD bandits, we switch to the setting where the μ_i are not constant on $\mathbb{Z}^-$ and show how the results obtained in Proposition 4.7 adapt to this setting.
While the definition of B^*_{double} and B^*_σ remain unchanged (respectively 4.20 and 4.21), the definition of $\widetilde{B}^*$ needs to be adapted. The essential idea behind $\widetilde{r}$ is to use the first pulls of any arm in the block as a calibration step. The previous definition we gave works for the setting where the μ_i are constant on $\mathbb{Z}^-$, but not when the μ_i are not constant. Indeed, suppose that arm a has been played at the end of the previous block and is in state $\tau_a \leq 0$ at the beginning of block B. Imagine that there are several pulls of a in the first steps of B. In this case, the state of arm a will keep decreasing, starting from the state left from the previous block. The state will decrease as $\tau_a, \tau_a - 1, \tau_a - 2, \ldots$, but since τ_a is unknown then all these subsequent states will be unknown as well. Even if we

ignore the first pull of a in B, this behaviour would not allow us to calibrate the state of the arm, because $\tau_a - 1, \tau_a - 2$ will remain unknown. To overcome this limit, we introduce the following definition:

$$\widetilde{B}^* = \operatorname*{argmax}_{|B|=d, a_1 \neq a_2} \widetilde{r}(B) \tag{4.32}$$

where we maintain the same concept of ignoring the first pulls for calibration, but we enforce the first two actions in the block, a_1 and a_2, to be different. By doing so, we ensure that with a_2 the state of a_1 is reset to $\tau_1 = 1$ even in case it was $\tau_1 \leq 0$ at the beginning of the block. Without this constraint, it would not be possible to obtain a valid approximation error. Note that the second claim of Lemma 4.6 also possesses two different first actions, meaning that this extra constraint in Equation (4.32) does not harm the approximation.
Using this definition, we are able to rewrite the Proposition 4.7 for the setting where the reward functions μ_i are not constant over $\mathbb{Z}^-$.

Proposition 4.8. *Let $(\mu_i)_{i=1}^K$ be a LSD bandit with K arms. Let $T \geq 0$ be the horizon, and $d \geq 0$ that divides T. Let $r_{\text{double}}(t)$ be the expected rewards obtained at time step t by the policy playing cyclically B^*_{double}. We have*

$$\frac{1}{T}\sum_{t=1}^{T} r_{\text{double}}(t) \geq \left(1 - \frac{d}{T}\right)\left(\frac{1}{T}\sum_{t=1}^{T} r^*(t) - \frac{K+2}{d}\right).$$

Let $\sigma \in \mathfrak{S}_K$, and assume that $d+K$ divides T. Let $r_\sigma(t)$ be the expected rewards obtained at time step t by the policy playing cyclically $[B_\sigma, B^*_\sigma]$. We have

$$\frac{1}{T}\sum_{t=1}^{T} r_\sigma(t) \geq \frac{d}{d+K}\left(\frac{1}{T}\sum_{t=1}^{T} r^*(t)\right) - \frac{K+2}{d+K}.$$

Let $\widetilde{r}^*(t)$ be the expected rewards obtained at time step t by the policy playing cyclically $\widetilde{B}^*$. We have

$$\frac{1}{T}\sum_{t=1}^{T} \widetilde{r}^*(t) \geq \frac{1}{T}\sum_{t=1}^{T} r^*(t) - \frac{K+2}{d}.$$

Proof. The proof is similar to that of Proposition 4.7, see Section 4.3. The only difference is that we cannot use $r(B^* \mid \boldsymbol{\tau}_{B^*}) \geq r\big(B^* \mid \boldsymbol{\tau}^*(t_0)\big) - K$, as we did during the proof of the first claim of Proposition 4.7. Instead, we have to involve $(B^*)'$, as defined in Lemma 4.6. Then, we have

$$r(B^*_{\text{double}} \mid \boldsymbol{\tau}_{B^*_{\text{double}}}) \geq r\big((B^*)' \mid \boldsymbol{\tau}_{(B^*)'}\big) \geq r\big(B^* \mid \boldsymbol{\tau}^*(t_0)\big) - (K+2),$$

$$r(B^*_\sigma \mid \boldsymbol{\tau}_{B_\sigma}) \geq r\big((B^*)' \mid \boldsymbol{\tau}_{B_\sigma}\big) \geq r\big(B^* \mid \boldsymbol{\tau}^*(t_0)\big) - (K+2),$$

$$\operatorname{Rew}\big(\widetilde{B}^*\big) \geq \operatorname{Rew}\big((B^*)'\big) \geq r\big(B^* \mid \boldsymbol{\tau}^*(t_0)\big) - (K+2),$$

which allows to complete the missing parts in the proofs. Note that in the last equation, the first inequality holds true thanks to the fact that $(B^*)'$ has also two first actions that are different, while the second inequality can be recovered from similar arguments as those used to prove Lemma 4.6. □

4.4 The proposed solution: the `ISI-CombUCB1` algorithm

After identifying an approach which solves the calibration problem and attains a valuable approximation error, the next challenge is to estimate the best block of size d to play. The approach we propose is built upon the `CombUCB1` algorithm for combinatorial semi-bandits introduced in Gai et al. [2012].
As explained in Chapter 2, combinatorial semi-bandit (CSB) is a class of multiarmed bandits where the action consists of a super-arm of N actions chosen from a universe of $L > N$ base actions, under some combinatorial constraints. Once the super-arm is played, the learner observes the individual rewards of the N base actions and receives their sum as a reward for the super-arm. In this setting, the regret measures the performance between the super-arm selected by the player and the best subset of N base actions satisfying the constraints. We recall that a standard K-armed bandit is a particular instance of a combinatorial semi-bandits, with $N = 1$, $L = K$, and no constraints. We use the knowledge of CBS to solve our estimation problem, showing that computing the best block with small regret against $\widetilde{B}^*$ with respect to $\widetilde{r}$ can be reduced to a CSB problem.
We reduce our estimation problem to a CSB problem where $N = d$, since the super-arm of size N corresponds to our block of d actions. The universe of base actions of cardinality L in CSB is more intricate to determine in our setting. In CSB the super-arm of size N allows one action to be played at most once in the super-arm. On the contrary, in the LSD setting a block B may contain multiple plays of the same action. Moreover, while in CSB the rewards of the actions are i.i.d., in block B the rewards also depend on the state. To overcome these limitations, we consider a broad universe of Kd^2 base actions in the LSD setting. In this set of base actions, we consider as individual entities the same action played with different states and at different positions within the block. Namely, a base action is index by an arm $a^{(i)} \in \mathcal{A} = \{a^{(1)}, \dots, a^{(K)}\}$, a state $\tau \in \{1, \dots, d\}$, and a position inside the block $t \in \{1, \dots, d\}$. Note that we limit the state τ in $\{1, \dots, d\}$ due to the fact that with the approach we adopt the state of every action is calibrated inside the block. The state coordinate ensures the i.i.d. nature of the rewards, while the time coordinate allows to remove the arm multiplicities, therefore making the map from a block to its representation one-to-one. This representation is necessary to be able to reduce our problem to a CSB problem and adapt the algorithm for CSB to our LSD setting. Before moving to the algorithm, we highlight that the structure of the problem plays a critical role, as the action space is of size $L = Kd^2$, instead of K^d in the absence of any structure. However, we also note that from a pure estimation point of view, a space of size Kd is enough, since we know that the position of the action in the sequence does not influence its reward - the state does.

Algorithm 1 `ISI-CombUCB1`

```
input : number of arms $K$, block size $d$, horizon $T$
init  : $\forall i \le K, j \le d, \quad T_1(i,j) = 0, \quad \overline{X}_1(i,j) = 0,$
        $U_1(i,j) = +\infty.$
for $t$ from 1 to $T/d$ do
    // Play the best block for $\widetilde{r}$ based on the UCBs
    Play $\widehat{B}_t = [a_{t,1} \dots a_{t,d}]$ that maximizes in $(a_s)_{s \le d}$
        $$\sum_{s=1}^{d} U_t\big(a_s, \tau_{a_s}(s)\big)\mathbb{I}\{\exists s_0 < s \colon a_{s_0} = a_s\}$$
    Get rewards $X_t\big(a_{t,1}, \tau_{a_{t,1}}(1)\big) \dots X_t\big(a_{t,d}, \tau_{a_{t,d}}(d)\big)$
    // Update the statistics
    for $i \le K$ and $j \le d$ do
        if arm $i$ is played with delay $j$ in block $\widehat{B}_t$ (counting the multiplicities) then
            $T_{t+1}(i,j) = T_t(i,j) + 1,$
            $\overline{X}_{t+1}(i,j) = \frac{T_t(i,j)\overline{X}_t(i,j) + X_t(i,j)}{T_{t+1}(i,j)}$
        else
            $T_{t+1}(i,j) = T_t(i,j), \quad \overline{X}_{t+1}(i,j) = \overline{X}_t(i,j)$
        $U_{t+1}(i,j) = \overline{X}_{T_{t+1}(i,j)}(i,j) + \sqrt{\frac{\alpha \log(t+1)}{T_{t+1}(i,j)}}$
```

To solve a LSD bandit, we adapt the `CombUCB1` algorithm introduced by Gai et al. [2012] for combinatorial semi-bandits. Its analysis was later refined by Chen et al. [2013] and Kveton et al. [2015]. In the latter, the authors provide an improvement in the performance of the algorithm, proving tight regret bounds. With the adjustments we presented on the structure of the problem, it is possible to apply `CombUCB1` to our setting in a black box fashion. In this case, the regret bound of $\mathcal{O}(\sqrt{NLn \log n})$ in Kveton et al. [2015] is adapted by placing $N = d$, $L = Kd^2$, and $n = T/d$, since we do not play a super-arm per time step but a block which takes up d time steps over the horizon T. We adapt `CombUCB1` to the LSD setting and present our algorithm called `ISI-CombUCB1`, which stands for `Initial States Independent CombUCB1`. Before showing the regret bound obtained by combining the regret bound of `CombUCB1` and the approximation result, we discuss its steps.
The pseudocode for ISI-CombUCB1 is presented in Algorithm 1. The algorithm recovers the main steps of `CombUCB1` Gai et al. [2012], but adapts it to the maximization of $\widetilde{r}$. It receives as input the set of actions, the block size d, and the horizon T. Before starting the game, when there is no information about the arms and their rewards, each arm has a UCB index equal to $+\infty$, to make

sure that at the beginning of the game each action is played once. Note that the initialization is such that pulling an uninitialized arm is always better than pulling any combination of initialized arms. As a consequence, the latter lasts at most Kd rounds, like in standard `CombUCB1`.
For every time step t, from 1 to T/d, the algorithm plays a block of actions $\widehat{B}_t = [a_{t,1}, \dots, a_{t,d}]$ selecting the arms $(a_s)_{s \le d}$ which maximize the following quantity

$$\sum_{s=1}^{d} U_t\big(a_s, \tau_{a_s}(s)\big)\mathbb{I}\{\exists s_0 < s \colon a_{s_0} = a_s\},$$

which is the sum of the UCB indexes of the arms in the d steps of the block, ignoring their first pulls. The block $\widehat{B}_t$ will contain the d-sized admissible subset of arms that maximize this quantity. After identifying these arms, the learner plays the block of actions and obtains the rewards $X_t\big(a_{t,1}, \tau_{a_{t,1}}(1)\big) \dots X_t\big(a_{t,d}, \tau_{a_{t,d}}(d)\big)$, which are used to update the arms' statistics stored by the algorithm. Specifically, the rewards are used to update the $T_t(i,j)$, where $i \le K$ index the arm, $j \le d$ index the state, and $T_t(i,j)$ stores the number of plays of arm i with state j. Also, the rewards are used to update the empirical mean $\bar{X}_t(i,j)$, which again is indexed by the arm and the state and stores the empirical average of the reward obtained so far from playing arm i with state j. Using $T_t(i,j)$ and $\bar{X}_t(i,j)$, the algorithm finally updates the UCB indexes using the following formula:

$$U_{t+1}(i,j) = \bar{X}_{T_{t+1}(i,j)}(i,j) + \sqrt{\frac{\alpha \log(t+1)}{T_{t+1}(i,j)}} \tag{4.33}$$

and repeats the process until T is reached. In ISI-CombUCB1, the optimization problem to select the best subset amounts to solve Equation (4.32), but using the UCBs instead of the true expected rewards. As pointed out during the construction of the universe, in our case some actions share the same rewards: indeed, in our setting the position of the arm inside the block does not influence its reward, only the state does. Thus, it is sufficient to maintain only Kd UCBs, one for each combination of arm and delay. The key step in the algorithm is to compute the block $\widehat{B}_t$ which maximizes the current rewards estimates. We formally define this optimization problem as follows. Let $F \in \{0,1\}^{K\times d}$, such that $F[i,t] = 1$ if and only if arm $a^{(i)}$ is played for the first time in the block at time step t. Let $Y \in \{0,1\}^{K\times d\times d}$, such that $Y[i,j,t] = 1$ if and only if arm i is played with delay j at time step t. Let $Z = (F, Y)$ be the Kd^2-sized representation we introduced earlier. Note that the column associated with $t = 1$ in Y is filled with zeros, as a pull here is by definition a first pull, and thus encoded in F. This notation of size $Kd(d+1)$ is however more convenient for coherence. The constraints for Z needed to describe a valid sequence of arm pulls are: Action Consistency (AC), i.e., at each time step, one and only one action is played, Unique First Pull (UFP), i.e., there is only one first pull per arm, First Pulls First (FPF), i.e., first pulls must precede any other pull of the same arm, and Time Consistency (TC), i.e., an arm can be pulled with delay j at time step t only if it was pulled at time step $t - j$, and not pulled since.

These constraints write (for index limits that make sense)

$$\forall t \qquad \sum_i F[i,t] + \sum_{i,j} Y[i,j,t] = 1 \tag{AC}$$

$$\forall i \qquad \sum_t F[i,t] \leq 1 \tag{UFP}$$

$$\forall i,t \qquad \sum_j Y[i,j,t] \leq \sum_{s=1}^{t-1} F[i,s] \tag{FPF}$$

$$\begin{aligned}\forall i,j,t \qquad Y[i,j,t] \leq F[i,t-j] &+ \sum_l Y[i,l,t-j] \\ &- \sum_s Y[i,j-s,t-s]\end{aligned} \tag{TC}$$

The objective function is the sum of the current UCBs for the second actions present in the block and writes $\sum_{i,j,s} Y[i,j,s]\, U_t(i,j)$. Noticing that all the relations are linear, we can derive $c_t \in \mathbb{R}^{Kd^2}$, $G \in \mathbb{R}^{Kd^2+K \times Kd^2}$, $h \in \mathbb{R}^{Kd^2+K}$, $A \in \mathbb{R}^{d \times Kd^2}$, and $b \in \mathbb{R}^d$, such that the optimization problem writes as

$$\max_{z \in \{0,1\}^{Kd^2}} \quad c_t^\top z \qquad \text{s.t.} \quad \begin{cases} Gz \preceq h \\ Az = b \end{cases} \tag{4.34}$$

where z is a vector version of Z. We highlight that (4.34) is an integer linear program, which is NP-hard to solve in general. This is expected, as our approach enjoys a sublinear linear regret against OPT (which is NP-hard to compute), and is therefore bound to be intractable. In Simchi-Levi et al. [2021] (Lemma 6), a Fully Polynomial-Time Approximation Scheme (FPTAS) is used to address a similar problem. However, we highlight that, although similar at first sight, our two ILPs are fundamentally different. While the authors try to select the best K arms at a fixed time step, we aim at selecting an optimal block, that takes into account the evolution of the rewards over time. This time constraint is specific to our problem and prevents us from using standard FPTASs. Instead, we propose a heuristics based on a branch-and-bound-like approach [Land and Doig, 2010; Clausen, 1999], where we use a LP relaxation to estimate the value of the objective. This amounts to testing every admissible (discrete) first action, and then keeping the one maximizing the relaxed objective (in which all subsequent actions are relaxed in $[0,1]$). The same is repeated to choose the following $d-1$ discrete actions. Overall, we solve d Linear Programs, of sizes $Kd, 2Kd, \ldots, Kd^2$. Given a horizon T, and choosing d as in Theorem 4.9, the total running time is $\mathcal{O}(K^{5/2}T^{9/4})$.
To ease readability, here we considered positive delays only. We now extend the discussion of the optimization problem to negative states and expand the discussion to the heuristic. We start by adapting the Integer Linear Program which constitutes the optimization problem of `ISI-CombUCB1`. As previously said, the difference is that we need to enforce the first two actions of the block to be different for calibration purposes. The second important difference is the size of the representation: we introduce Y^+ and Y^-, both of size Kd^2, to encode pulls in positive and negative states respectively. The problem can be described

as follows.
Let $F \in \{0,1\}^{K\times d}$, such that $F[i,t]=1$ if and only if arm i is played for the first time in the block at time step t. Let $Y^+ \in \{0,1\}^{K\times d\times d}$, such that $Y^+[i,j,t]=1$ if and only if arm i is played at time step t in state j. Let $Y^- \in \{0,1\}^{K\times d\times d}$, such that $Y^-[i,j,t]=1$ if and only if arm i is played at time step t in state $-j$. In comparison to the previous Integer Linear Program, the objective function, i.e., the sum of the current UCBs for the second actions present in the block, and the conditions (AC), (UFP), and (FPF) remain unchanged. Their formula are recalled for completeness. As for the novelties, we introduce the Change Action (CA) constraint, i.e., the second pull in the block must differ from the first. Time Consistency (TC) is now divided into TC for positive states (TC$^+$), i.e., an arm can be pulled in state $j \geq 1$ at time step t only if it was pulled at time step $t-j$, and not pulled since, and TC for negative states, (TC$^-$), i.e., an arm can be pulled in state $-j \leq -1$ at time step t only if it has been pulled consecutively for the last j time steps. Note that it is easier to express TC$^-$ using two conditions, depending on whether $-j=-1$ or $-j \leq -2$. Overall, we have (for index limits that make sense)

$$\sum_{i,j,s} \Big(Y^+[i,j,s] + Y^-[i,j,s]\Big)\, U_t(i,j) \qquad \text{(objective)}$$

$$\forall t \qquad \sum_i F[i,t] + \sum_{i,j} Y^+[i,j,t] + \sum_{i,j} Y^-[i,j,t] = 1 \qquad \text{(AC)}$$

$$\forall i \qquad \sum_t F[i,t] \leq 1 \qquad \text{(UFP)}$$

$$\forall i,t \qquad \sum_j Y^+[i,j,t] + \sum_j Y^-[i,j,t] \leq \sum_{s=1}^{t-1} F[i,s] \qquad \text{(FPF)}$$

$$\sum_i F[i,2] = 1 \qquad \text{(CA)}$$

$$\begin{aligned}\forall i,j,t \qquad Y^+[i,j,t] \leq{}& F[i,t-(j+1)] + \sum_l Y^+[i,l,t-(j+1)] \qquad (4.35)\\ &- \sum_s Y^+[i,j-s,t-s] + \sum_l Y^-[i,l,t-(j+1)]\\ &- \sum_l Y^-[i,l,t-j] \qquad \text{(TC}^+\text{)}\end{aligned}$$

$$\forall i,t \qquad Y^-[i,1,t] \leq \sum_l Y^+[i,l,t-1] + F[i,t-1] \qquad (\text{TC}_1^-)$$

$$\forall i,t,j\geq 2 \qquad Y^-[i,j,t] \leq Y^-[i,j-1,t-1] \qquad (\text{TC}_2^-)$$

Similarly to the previous ILP, we can approximately solve the above Integer Linear Program by a Branch-and-Bound-like approach, presented in Algorithm 2. This heuristic works as follows. For every $i \leq K$, we set the first action of the block (of total size d) to $a^{(i)}$. We then solve a relaxed version of the ILP, optimizing only for actions $a_2, \dots a_d$ (recall that a_1 is fixed to $a^{(i)}$), and allowing for continuous values in $[0, 1]$, instead of $\{0, 1\}$. This can be done efficiently as the relaxed problem is a standard Linear Program. We finally set a_1 to the $a^{(i)}$ which has given the highest reward according to the relaxed ILP. We reiterate, by testing values for the second action, and solving the relaxed version with respect to $a^{(3)} \dots a_d \in [0, 1]^{d-2}$, and so on. Let `LP` be the function that takes as input the current UCBs and a fixed block of size $s < d$, and outputs the best continuous solution in $[0, 1]^{d-s}$ for actions $a_{s+1} \dots a_d$. Let `reward` be the function returning the objective value of any sequence (possibly partially continuous). The heuristic is summarized in Algorithm 2.

Algorithm 2 Approximate ILP Solver

```
input : Current UCBs $U_t = [U_t(i,j)] \in \mathbb{R}^{K \times 2d}$ for all $i \leq K$ and $-d \leq j \leq d$
init  : block = []
for $s = 1 \dots d$ do
    for $i = 1 \dots K$ do
        block_tmp = block + [$a^{(i)}$]        // test $a^{(i)}$ as next discrete
            action (current block size: $s$)
        block_cont = LP($U_t$, block_tmp)      // find the best continuous
            continuation (of size $d - s$)
        $r^i$ = reward(block_tmp, block_cont)   // compute the total relaxed
            reward of the half-discrete
                                               // half-continuous block (of overall
            size $d$)
    $i^* = \text{argmax}_{i \leq K}\ r^i$
    block = block + [$a_{i^*}$]                // keep the discrete action with
        highest relaxed reward
return block
```

We point out a few aspects of our solution. Although it is generally hard to derive approximation guarantees, this approach works well in practice, delivering the optimal solution in all the cases we tested. Moreover, as discussed in Kveton et al. [2015], if `ISI-CombUCB1` is run with the approximate solver, Theorem 4.9 can be adapted to bound the regret against the best block according to the solver's approximation. Finally, note that our calibration approach does not affect the complexity of finding the reward-maximizing block. Indeed, `CombUCB1` is also required to solve an integer linear program similar to ours to find the subset of base actions maximizing the block reward.
After discussing the algorithm, the optimization problem, and the heuristic used to solve it, we analyze our solution to prove guarantees on the performance of Algorithm 1. As we did for the previous discussions, we provide two results for the two settings where the expected rewards are constant on $\mathbb{Z}^-$ or where they are not.

Theorem 4.9. *Let $(\mu_i)_{i=1}^K$ be an LSD bandit with K arms and constant expected rewards on $\mathbb{Z}^-$. Let $T \geq 0$ be the horizon, and choose $d \geq 0$ that divides T.*

Then `ISI-CombUCB1`*, run with block size d and exploration parameter $\alpha = 1.5$, has regret bounded by*

$$R_T \leq \frac{KT}{d} + 47d\sqrt{KT \log \frac{T}{d}} + \left(\frac{\pi^2}{3} + 1\right) Kd^3 \ .$$

Choosing $d \propto T^{1/4}$, we obtain $R_T = \widetilde{\mathcal{O}}(KT^{3/4})$, where $\widetilde{\mathcal{O}}$ is neglecting logarithmic factors.

Theorem 4.10. *Let $(\mu_i)_{i=1}^K$ be an LSD bandit with K arms. Let $T \geq 0$ be the horizon, and choose $d \geq 0$ that divides T. Then* `ISI-CombUCB1`*, run with block size d and exploration parameter $\alpha = 1.5$, has regret bounded by*

$$R_T \leq \frac{(K+2)T}{d} + 47d\sqrt{2KT \log \frac{T}{d}} + \left(\frac{\pi^2}{3} + 1\right) 2Kd^3 \, .$$

Choosing $d \propto T^{1/4}$, we obtain $R_T = \widetilde{\mathcal{O}}(KT^{3/4})$, where $\widetilde{\mathcal{O}}$ is neglecting logarithmic factors.

This bound is the result of adapting the bound of $\mathcal{O}(\sqrt{NLn \log n})$ from Kveton et al. [2015] to our setting in a black box fashion.

Our bounds in Theorem 4.9 and Theorem 4.10 are easily interpretable. The impact of the approximation results from Proposition 4.7 and Proposition 4.8 in Theorem 4.9 and Theorem 4.10 respectively is represented by the first terms of the sum, $\frac{KT}{d}$ and $\frac{(K+2)T}{d}$, where the learner pays the price for the calibration of the arms by ignoring their first pull in the block. The terms $\left(\frac{\pi^2}{3} + 1\right) Kd^3$ and $\left(\frac{\pi^2}{3} + 1\right) 2Kd^3$ respectively in Theorem 4.9 and Theorem 4.10 account for the loss payed for the initialization of the algorithm. In Kveton et al. [2015], the term on the right-hand side of the parenthesis was KL due to their size of the universe, which considered a ground set of L elements and selected a super-arm of maximum size K. Adapting the bound to our universe for μ_i constant on $\mathbb{Z}^-$, we consider a super-arm of size d and a set of actions of size Kd^2, since we interpret as different arms the different K actions played with different d states at different d time steps in the block. This adaptation ($K = d, L = Kd^2$) leads to the term Kd^3 in Theorem 4.9. In Theorem 4.10 the assumption that the μ_i are non-decreasing on $\mathbb{Z}^-$ and the subsequent consideration of negative states lead to a universe where the possible states are doubled in size since we can have $2d$ possible states, $\tau \in \{-d, \dots, -1, 1, \dots, d\}$, therefore $K = d, L = 2Kd^2$, resulting in the term $2Kd^3$ in the bound. Finally, the middle term accounts for the loss of the estimation process computed by `ISI-CombUCB1` and the branch-and-bound solution.

Note that the second claims of Theorem 4.9 and Theorem 4.10 require a horizon-dependent tuning. The doubling trick can be used to make the algorithm anytime without harming the bound. Regarding the optimality, we recall that our approximation result in $\mathcal{O}(KT/d)$ is tight, see e.g., Example 4.2. As for the estimation part, Kveton et al. [2015] proved a lower bound for `CombUCB1` that matches the upper bound up to a factor $\sqrt{\log n}$, where n is their horizon. Instantiating this lower bound to our case, we obtain an overall lower of bound of order $\Omega(KT/d + d\sqrt{KT})$. This allows us to state that our result is tight up to a polylogarithmic factor of $\sqrt{\log(T/d)}$.

4.5 Experiments on the LSD bandit setting

To conclude the presentation of this work, we present the experiments conducted on the performance of `ISI-CombUCB1`.
As in [Warlop et al., 2018], we work with synthetic data to conduct these experiments. This choice is motivated by the counterfactual nature of the learning problem in bandits. Indeed, the datasets typically used in the field of recommender systems are not suitable for these bandit problems. This is due to the fact that a suitable dataset would necessarily store the user's level of satisfaction with respect to all possible ordered sequences of all the items in a dataset, where the reward may highly depend on the specific order of the sequence. Since this type of information is not provided by datasets available online, we rely on synthetic data.
We benchmark our algorithm against vanilla `CombUCB1` and `Oraclegreedy`, which is the algorithm that plays in each time step $a_t = \operatorname{argmax}_i \mu_i(\tau_i(t))$, and breaks ties randomly. Because computing OPT is NP-hard, and that consequently computing the regret is not possible, we measure the performance in terms of total cumulative reward, averaged over ten repetitions. For both `ISI-CombUCB1` and `CombUCB1` the UCB index of an arm $a^{(i)}$ is computed using the formula

$$U_t(a^{(i)}, \tau_i(t)) = \sqrt{\frac{\alpha \log t}{T_i(t)}}, \tag{4.36}$$

where for both the algorithms $\alpha = 1.5$, as in Kveton et al. [2015]. For both algorithms, the same heuristic based on branch-and-bound and LP relaxations are used to compute the reward-maximizing block at each time step. Let d be the block size of `CombUCB1`, which consequently maintains Kd UCBs. , `CombUCB1` is not calibrated, so an arm $a^{(i)}$ may be pulled with state $\tau_i \geq d$. Whenever this happens, we assume the algorithm updates the estimate of $\mu_i(d)$, which actually becomes an estimate of $\mu_i(\tau_i \geq d)$. In order to maintain Kd UCBs for `ISI-CombUCB1`, we consider blocks of size $d + 1$. This is necessary in order to reach the state $\tau_i = d$. Indeed, since the rewards are ignored the first time an arm is pulled, reaching a state $\tau_i = d$ requires at least $d + 1$ steps. We recall that the "extra" first pull does not provide any information and it is only used for calibration purposes, nevertheless it is considered in the measurement of the performance.

To analyze the behaviour of these algorithms, we consider an instance of LSD bandits with $K = 5$ arms, Bernoulli rewards, and the following reward functions (see Figure 4.2):

$$\mu_1(\tau) = 0.95\,\mathbb{I}\{\tau = 3\}, \quad \mu_2(\tau) = \begin{cases} 0.16 & \text{if } \tau = 6 \\ 0.96 & \text{if } \tau \geq 9 \\ 0.14 & \text{otherwise} \end{cases}, \quad \mu_{3,4,5}(\tau) = 0.15 \quad \forall \tau\,. \tag{4.37}$$

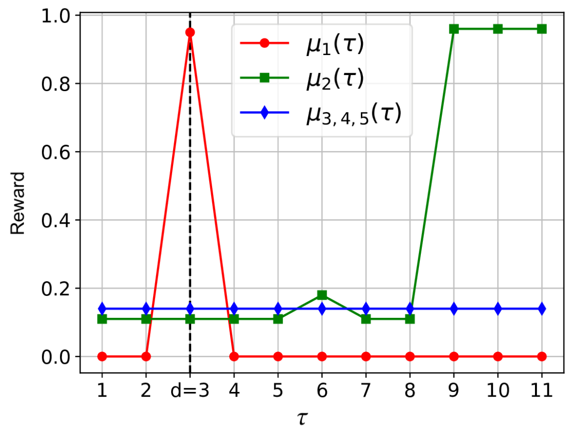

FIGURE 4.2: This figure shows the reward (y-axis) functions μ_i for $i = \{1, 2, 3, 4, 5\}$ with respect to τ (x-axis), as defined in Equation (4.37).

In Figure 4.3, we show the empirical results obtained for $d = 3$. As one can notice, `ISI-CombUCB1` reaches a higher cumulative reward compared to `CombUCB1` and `OracleGreedy`. Indeed, `ISI-CombUCB1` converges towards the block $[a^{(1)}, a^{(i)}, a^{(i)}, a^{(1)}]$, where $a^{(i)}$ is any arm in $\{a^{(3)}, a^{(4)}, a^{(5)}\}$. Note that the first pull of a_1 in the block is necessary for the calibration of the arm, which returns its highest possible reward in the last step of the block when pulled with state $\tau_1 = 3$. This allows `ISI-CombUCB1` to exploit arm a_1 in its best possible state. On the contrary, `OracleGreedy` is not able to exploit the reward of arm a_1. In fact, it pulls any constant arm for the first 8 time steps except at $t = 3$ and $t = 6$, where it plays arm a_1. When `OracleGreedy` gets to $t = 9$ it faces a choice between $\mu_1(3)$ and $\mu_2(9)$ and thus chooses the latter since it has a slightly higher reward than action a_1. This behaviour prevents it from ever pulling μ_1 again, and will only get $\mu_2(9)$ every nine steps with any combination of constant actions in between. As for `CombUCB1`, it is never able to pull $\mu_1(3)$. This is due to fact that whenever the algorithm pulls $\mu_1(\tau_i)$ with $\tau_i > 3$, the expected reward is small and, contributing to the estimate of $\mu_1(3)$ and its UCB index, the algorithm will believe that the action is suboptimal pulling it less and less frequently.
After showing the performance of `ISI-CombUCB1` exceeds the performances of `CombUCB1` and `OracleGreedy`, we work on the same instance of LSD bandits and benchmark two new algorithms based on calibration sequences (CS): `CS-worst` and `CS-best`. The goal here is to elaborate on the performance of our algorithm and compare it more carefully to other potential solutions, which were discussed in Proposition 4.7. Here, we intend to support our theoretical discussion with empirical validation. Specifically, we consider the approach of maximizing Equation (4.21) proposed in Equation (4.26) of Proposition 4.7, where the idea was to play a calibration sequence of size K at the beginning of the block so that all arms are calibrated before being pulled in the d actions of the block. These approaches first calibrate the system by playing a permutation σ of all the arms and then play the best block according to the state reached after σ. In the previous sections, we discarded this solution because we presented a more efficient

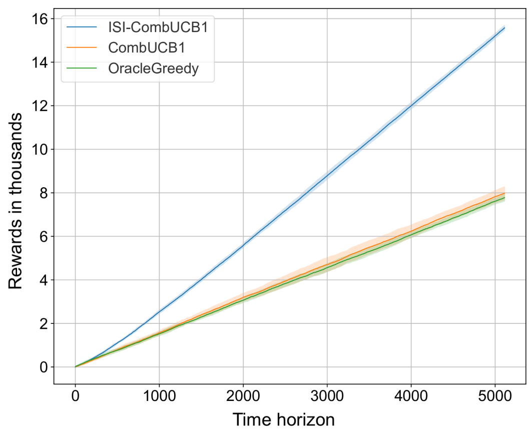

FIGURE 4.3: This figure depicts the comparison of the performance of `ISI-CombUCB1`, `CombUCB1`, and `OracleGreedy` measured in terms of cumulative rewards in thousands (y-axis), averaged over ten repetitions, with the time horizon on the x-axis.

way of solving our problem using $\widetilde{r}$ (Equation (4.22)), which ignores the first pull of any action played in the block. Indeed, CS-based approaches are known to be suboptimal, as they calibrate more arms than necessary, here $K = 5$ arms instead of 2 (number of different arms in the optimal block).
Since we did not define a specific calibration sequence, we tested two different permutations for this approach. Note that since the calibration phase is of size $K = 5 > 3$, it might prevent these algorithms from seeing the spike of arm $a^{(1)}$ at $\tau_1 = 3$. `CS-worst` plays a calibration sequence $\sigma_{worst} = [a^{(1)}, a^{(2)}, a^{(3)}, a^{(4)}, a^{(5)}]$, while `CS-best` calibrates the state pulling the sequence $\sigma_{best} = [a^{(5)}, a^{(4)}, a^{(3)}, a^{(2)}, a^{(1)}]$. Their name indicates whether they can observe the spike or not. As for `ISI-CombUCB1` and `CombUCB1`, these two CS approaches use the same branch-and-bound heuristic and the LP relaxations presented in the previous sections. We add the performances of `CS-worst` and `CS-best` to the results, presented in Figure 4.4.

Considering the previous experiments, one may argue that the instance of a 5-armed bandit presented in Equation (4.37) is unfair to `OracleGreedy`, because the algorithm is tricked by phenomena that occur beyond the block size d. Therefore, we conclude this section with the presentation of a new example where this is not the case.
We consider an instance of a 2-armed LSD bandits with Bernoulli rewards and the following reward functions

$$\mu_1(\tau) = \begin{cases} 0.06 & \text{if } \tau = 1 \\ 0.95 & \text{if } \tau \geq 2 \end{cases} \qquad \text{and} \qquad \mu_2(\tau) = 0.05 \quad \forall \tau\,. \tag{4.38}$$

We represent the reward functions in Figure 4.5 and the empirical evaluation for $d = 10$ in Figure 4.6. In this scenario, `OracleGreedy` constantly pulls $a^{(1)}$, getting an average reward of 0.06 and never pulling any other action. On the other hand, `ISI-CombUCB1` and `CombUCB1` behave similarly alternating between $a^{(1)}$ ad $a^{(2)}$, for an optimal average of 0.5. As opposed to `CombUCB1`, our algorithm

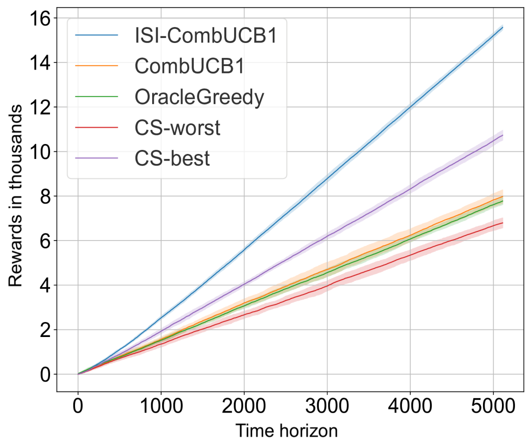

FIGURE 4.4: Here we plot the cumulative rewards in thousands (y-axis) with respect to the time horizon (x-axis) for the following algorithms: our approach `ISI-CombUCB1`, `CombUCB1`, `OracleGreedy`, `CS-worst`, and `CS-best`.

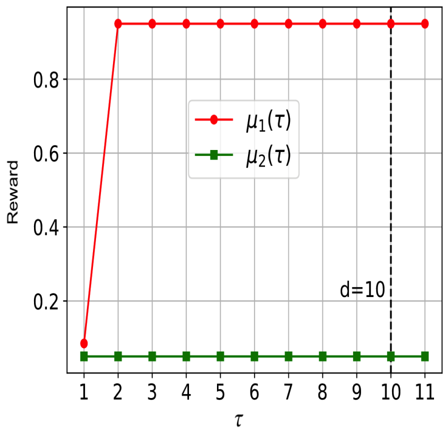

FIGURE 4.5: This figure shows the reward (y-axis) functions with respect to τ (x-axis).

requires a calibration pull before playing the optimal block. For this reason, in this steps `ISI-CombUCB1` receives a small reward and its average reward downgrades slightly, explaining the small gap between the two algorithms observed in Figure 4.6. Besides this gap due to the calibration, both algorithms converge towards the same optimal block. Overall, note that although the calibration is an operation that might occasionally degrade the performance (in a controlled way), on the other hand it guarantees to avoid risky decoys, such as the one discussed in the previous example (see Figure 4.3). This behaviour is in line with the fact that the regret of `ISI-CombUCB1` is well understood theoretically, while `CombUCB1` is hard to analyze due to the interferences.

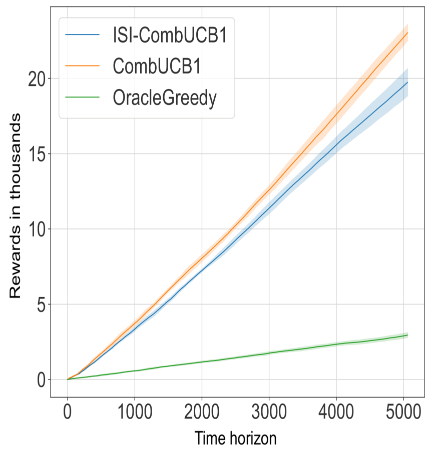

FIGURE 4.6: Here we present the cumulative rewards in thousands (y-axis) with respect to the time horizon (x-axis), comparing our solution `ISI-CombUCB1` to `CombUCB1` and `OracleGreedy`.

4.6 Conclusions

In this chapter, we presented a new model of non-stationary bandits able to solve one of the challenges discussed at the beginning: to formalize a multi-armed bandit model able to generalise to different forms of non-stationarity. Indeed, LSD bandits show it is possible to account for both satiation and seasonality with a single framework, able to generalize to these behaviours. This is done by using the concept of state associated with the arms and allowing for arm-dependent functions, relaxing two typical assumptions in non-stationary bandits, that μ_a is (1) non-decreasing on $\mathbb{Z}$ and constant on $\mathbb{Z}^-$, and using the minimal assumptions that will admit meaningful theoretical guarantees.
While this model is able to generalise different non-stationary settings facing both seasonality and satiation, focusing on a finite set of actions is not always representative of the reality of industrial applications. Indeed, applications are often characterised by an exponentially increasing data flood, making it difficult to gain exhaustive information on all products in a catalogue of millions of items. For this reason, our second objective is to focus on a linear non-stationary MAB model able to handle these types of non-stationary behaviours as well as to deal with a continuous and infinite set of actions.

Chapter 5

A novel linear non-stationary bandit model with an infinite set of actions

Contents

Since many non-stationary phenomena have been mainly studied in settings with finitely many arms, our goal here is to study non-stationarity in a linear embedding. In this chapter, we focus on the linear setting where the environment is characterized by a continuous infinite set of actions $a \in \mathcal{A} \subset \mathbb{R}^d$, namely the action space is represented by the Euclidean unit ball. Due to the structured set of actions, the peculiarity of this environment is that playing an action will influence not only the future rewards of this action but also the future rewards of other actions in the set. Therefore, the primary concern is to model non-stationary behaviours considering the interdependencies between actions. Since maintaining the concept of state in a linear setting is not trivial, we decided to model non-stationarity using a fixed-size window. The model we proposed is called *Linear Bandits with Memory* (LBM) and is presented in Clerici et al. [2023]. In particular, this model captures both rotting and rising behaviour and recovers stationary linear bandits as a special case. The core idea behind this model is that current rewards are influenced by the learner's past actions in a fixed-size window. Specifically, we modify the usual reward function of linear bandits $X_t = \langle a_t, \theta^* \rangle$ (see Chapter 3) by introducing a window of size m controlled by an exponent γ, so that the reward function becomes $X_t = \langle a_t, A_{t-1}\theta^* \rangle$ where $A_{t-1} = A(a_{t-1}, \ldots, a_{t-m}) = \left(A_0 + \sum_{s=1}^{m} a_{t-s}a_{t-s}^\top\right)^\gamma$. Note that while the size m of the window defines how many of the past actions influence the reward, the exponent γ controls the nature of the non-stationarity, capturing rotting behaviours when $\gamma < 0$ and rising behaviours when $\gamma > 0$, as well as stationary linear bandits when $\gamma = 0$. In the rest of this chapter, we will define the model and prove regret guarantees for a solution based on OFUL for the setting where these two parameters, m and γ, are known. However, aware of the fact that knowing m and γ is not always realistic, we propose a bandit selection approach to extend our solution in the case where these two parameters are unknown to the learner. Before introducing a formal definition of our model and discussing it in the rest of this chapter, we spend some words explaining the necessity of addressing non-stationary behaviours in a linear environment.

5.1 Motivations and applications

As mentioned at the beginning of this dissertation, we aim to study theoretical models of multiarmed bandits which could be applied in recommender systems, particularly in music streaming platforms. The goal is to propose models that not only address non-stationary phenomena, which are essential to model the trend of a user's preferences in such systems, but also the presence of an infinite and structured set of items, which characterizes many practical applications. Indeed, if we think of recommender systems, we can see how selecting an item may influence the level of satisfaction with other items which share some features. Let's consider a music recommender system where the actions are songs and the different dimensions of the problem represent different music genres. Let $d = 3$, where the first dimension represents *pop*, the second *rock*, and the third *jazz*. Let $a = [0.6, 0.4, 0]$ be a song which fits prevalently into the pop genre but has some rock sonorities. It is reasonable to think that having played a song with 0.4 amount of *rock* may influence the satiation of a user with other rock songs. With this simple example, it is easy to understand how the level of satiation of a user for an item is not only influenced by the times the user selected an exact item, but in a more complex scenario it can potentially be influenced by any other action with whom it shares some features. Furthermore, if we refer to this type of environment, it becomes natural to think that the choices made by the user far back in the past will stop influencing its level of satisfaction after a certain point. Indeed, it is reasonable to think that your current level of satisfaction with certain genres is not impacted by the songs you listened to months ago. To limit a potentially infinite impact, which would seem unrealistic, we decided to consider a fixed-size window. The window indicates that over a certain threshold, dictated by its size m, past actions do not influence future rewards anymore.

On top of addressing the structured set of actions, we also define a model which recovers two different behaviours: rotting and rising. In the example of music recommendation, it is easier to think of a decreasing reward function dependent on the number of plays of a song. Indeed, one may grow tired of listening to the same genre. However, it is plausible to propose a model where the reward function is increasing. Specifically, among the possible applications targeted by our model, we can think of algorithmic selection. In this setting, one owns some resources and must choose among a selection of algorithms and assign them a

chunk of resources. In this case, the increasing reward function models the quality of an algorithm, which improves the more it has been selected. Indeed, our modelling choices allow us to recover increasing reward functions.
Thus, we aim to propose a new bandit model motivated by applications, but also complex enough to capture nontrivial cross-arm effects while remaining tractable from a learning viewpoint.

5.2 Linear Bandits with Memory (LBM): a definition of the model

We define a Linear Bandit with Memory (LBM) as a linear multiarmed bandit where in each time step $t = 1, \dots, T$, the learner selects an action a_t from an infinite set of actions $\mathcal{A} \subset \mathcal{B}_d$, where $\mathcal{B}_d$ is the Euclidean unit ball. Since we are considering a linear embedding, the expected reward of an arm a_t is given by a linear function between the action played and an unknown parameter $\theta^* \in \mathbb{R}^d$. However, as we anticipated, the peculiarity of this model is to address non-stationary behaviours by considering past actions in a finite-size window, also called memory. Let m be the size of the memory. The influence of these past actions is addressed using a correlation matrix $\sum_{s=1}^{m} a_{t-s}a_{t-s}^\top$, where m indicates how many past actions still have an impact on the current time step. To model the rotting and rising behaviour, we introduce a parameter γ, which is a positive or negative exponent controlling the type of behaviour and its strength. Using these two parameters m and γ, the reward of an action a_t is given by the following formula

$$X_t = \langle a_t, A(a_{t-m}, \dots, a_{t-1})\theta^* \rangle + \eta_t \tag{5.1}$$

where η_t stands for noise and is a 1-sub-Gaussian random variable independent from the actions played by the learner and

$$A(a_1, \dots, a_m) = \Big(A_0 + \sum_{s=1}^{m} a_s a_s^\top\Big)^\gamma. \tag{5.2}$$

For simplicity, in the rest of the paper, we use the abbreviation

$$A_{t-1} = A(a_{t-m}, \dots, a_{t-1})$$

and refer to it as the *memory matrix*. Conventionally, we set $a_{1-m} = a_{2-m} = \dots = a_0 = 0_d$ and choose $A_0 = I_d$, unless otherwise stated. Note that at any time step t the expected reward $r_t = \mathbb{E}[X_t]$ satisfies $|r_t| \leq \|A_{t-1}\|_*$.
Using this definition of the memory matrix, the model can retrieve standard linear bandits when $\gamma = 0$, since the reward function reduces to $X_t = \langle a_t, \theta^* \rangle + \eta_t$. When $\gamma < 0$, the impact of the memory causes a rotting behaviour, meaning that the more certain actions are played the more their rewards decrease. On the other hand, when $\gamma > 0$, the model shows a rising behaviour, where the rewards of future actions increase the more these have been played in the past. The choice of the covariance matrix is intuitive, as it stores the previously

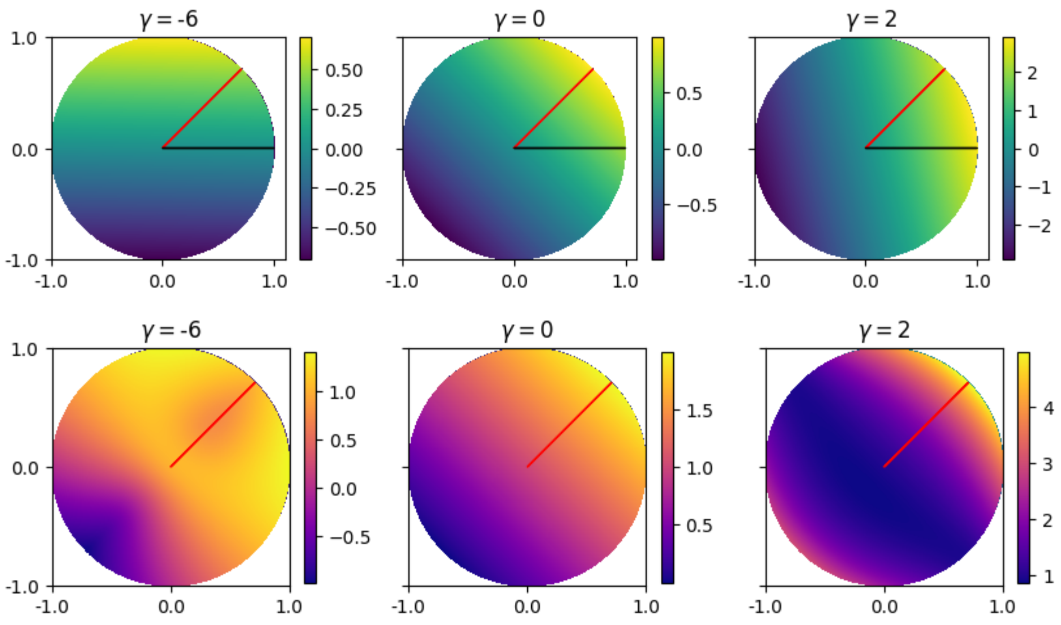

FIGURE 5.1: In the top pane, we plot the effect of the memory matrix (5.2) on the action space for $d = 2$, $m = 1$, and $\gamma \in \{-6, 0, 2\}$. The red arrow is θ^* and the black arrow is action a_{t-1}. The colour level indicates the value of the instantaneous expected reward of any action a_t (point on the disk). When $\gamma = -6$, the rotting effect is so powerful that the optimal action a_t is orthogonal to a_{t-1}. When $\gamma = 0$, the optimal action remains θ^*, independently of a_{t-1}. For $\gamma = 2$, the optimal action is shifted between θ^* and a_{t-1}. However, the top plot does not show that playing constantly θ^* is not the optimal policy. In the bottom pane, we consider horizon $T = 2$, with the same choices of parameters. For a given action a_1, since $T = 2$, it is possible to determine the best possible next action a_2. The colour now indicates the sum of expected rewards as a function of the initial action a_1 (point on the disk). For $\gamma = -6$, we see that playing θ^* is not optimal anymore. On the other side, it shows that not playing θ^* is more harmful when $\gamma = 2$ than when $\gamma = 0$.

played actions and naturally encodes the directions where satiation or excitation occurs. To facilitate the understanding of such phenomena, we provide a graphical explanation in Figure 5.1.

As in every bandit model discussed so far, the aim is to maximize the expected sum of rewards obtained by the learner over the horizon T. The performance is measured by the expected regret, defined as:

$$R_T = \sum_{t=1}^{T} r_t^* - \mathbb{E}\left[\sum_{t=1}^{T} X_t\right], \tag{5.3}$$

where $r_t^* = \langle a_t^*, A(a_{t-m}^*, \ldots, a_{t-1}^*)\theta^*\rangle$ and $(a_t^*)_{t=1}^T$ is the optimal sequence of actions which maximizes the expected sum of rewards over the horizon and is defined as:

$$a_1^*, \ldots, a_T^* = \operatorname*{argmax}_{a_1,\ldots,a_T \in \mathcal{A}} \sum_{t=1}^{T} \langle a_t, A(a_{t-m}, \ldots, a_{t-1})\theta^* \rangle. \tag{5.4}$$

In the rest of this chapter, we use OPT to refer to $\sum_{t=1}^{T} r_t^*$. Therefore, a LBM is fully characterized by the action set $\mathcal{A}$, the unknown parameter θ^*, the memory size m, and the exponent γ.
As anticipated, this model can recover different state-of-the-art bandits, such as stationary linear, rotting, and rising bandits. We provide details on these reductions in the following examples.

Example 5.1 (Stationary linear bandits)**.** *Consider a linear bandit model, defined by an action set $\mathcal{A} \subset \mathcal{B}_d$ and $\theta^* \in \mathcal{B}_d$. This is equivalent to a LBM with the same $\mathcal{A}$ and θ^*, and memory matrix A such that $A(a_1, \ldots, a_m) = I_d$ for any $a_1, \ldots, a_m \in \mathcal{A}^m$, i.e., when $m = 0$ or $\gamma = 0$.*

Example 5.2 (Rotting and rising rested bandits)**.** *In rotting [Levine et al., 2017; Seznec et al., 2019] or rising [Metelli et al., 2022] bandits, the expected reward of an arm k at time step t is fully determined by the number $n_k(t)$ of times arm k has been played before time t. Formally, each arm is equipped with a function μ_k such that the expected reward at time t is given by $\mu_k(n_k(t))$. In particular, requiring all the μ_k to be nonincreasing corresponds to the rotting bandits model, and requiring all the μ_k to be nondecreasing corresponds to the rested rising bandits model. Now, let $d = K$, $\mathcal{A} = (e_k)_{1 \le k \le K}$, $\theta^* = (1/\sqrt{K}, \ldots, 1/\sqrt{K})$, and $m = +\infty$. By the definition of A, see (5.2), and the orthogonality of the actions, it is easy to check that the expected reward of playing action e_k at time step t is given by $(1+n_k(t))^\gamma/\sqrt{K}$. When $\gamma \le 0$, this is a nonincreasing function of $n_k(t)$, and we recover rotting bandits. Conversely, when $\gamma \ge 0$, we recover rising bandits. We note however that the class of decreasing (respectively increasing) functions we can consider is restricted to the set of monomials of the form $n \mapsto (1+n)^\gamma/\sqrt{K}$, for $\gamma \le 0$ (respectively $\gamma \ge 0$). Extending it to generic polynomials is clearly possible, although it requires more computations in the model selection phase, see Remark 5.5.*
Although rotting and rising bandits require infinite memory, we argue on both practical and theoretical grounds that in our setting a finite value of m is preferable. First, in many applications, it is reasonable to assume that the effect of past actions will vanish at some point. If one has listened to a song long enough ago, this should not affect anymore how much they enjoy the song now. Second, permanent effects may trivialize the problem on the theoretical side: Consider $m \to \infty$ and $\gamma \le -1/2$, then for any sequence of actions $(a_t)_{t \ge 1}$ we have

$$\sum_{t=1}^{T} \langle a_t, A_{t-1}\theta^* \rangle \le \sum_{t=1}^{T} \left\| A_{t-1} a_t \right\|_2 \le \sqrt{T \sum_{t=1}^{T} \left\| A_{t-1} a_t \right\|_2^2}$$
$$\le \sqrt{2dT \log(1 + T/d)} := B_T\,,$$

where we have used the elliptical potential lemma [Lattimore and Szepesvári, 2020, Lemma 19.4]. Hence, as soon as $\gamma \le -1/2$, we have OPT $\le B_T$*, and* *the trivial strategy playing constantly* 0 *enjoys a small regret* B_T. *Focusing on finite memory* m *thus yields more interesting problems, although it prevents a full generalization of rotting bandits with finitely many arms. We note however that when* $m < \infty$, *the spirit of rotting (resp., rising) bandits is still preserved, as playing an action does decrease (resp., increase) its efficiency for the next pulls (within the time window), see also Figure 5.1.*

We highlight the relationship between our model and the result discussed in 3 about [Seznec, 2020]. In [Seznec, 2020] (Proposition 4.7.2, Corollary 4.7.3), the authors highlight the impossibility of learning rotting linear bandits under their modelling choices (see Rotting rested bandits in 3 for a detailed discussion). Indeed, they prove an incompatibility between learning linear bandits and rotting bandits in their model, showing that it is impossible to learn a model that generalizes both. Because our main goal is to focus on the linear setting, our model cannot capture the K-armed rotting bandit setting in its full generality. Therefore, we decided to focus on the linear setting, which would allow us to recover the rested variants of these models.
After these examples, one can think that a naive approach to solve the LBM problem could be to neglect non-stationarity. Assuming that θ^* is known, one may play at time t the action $a_t^{\text{greedy}} = \text{argmax}_{a\in\mathcal{A}}\langle a, A_{t-1}\theta^*\rangle$. This strategy, which we refer to as *oracle greedy*, may be optimal in some cases, e.g., in rising isotropic settings, see Heidari et al. [2016, Section 3.1] and Metelli et al. [2022, Theorem 4.1] for discussions in the K-armed case. However, we highlight that it may also be arbitrarily bad, as stated in the next proposition.

Proposition 5.1. *The oracle greedy strategy, which plays at each time step* $a_t^{\text{greedy}} = \text{argmax}_{a\in\mathcal{A}}\langle a, A_{t-1}\theta^*\rangle$, *can suffer linear regret, both in rotting or rising scenarios.*

Proof. To prove this proposition, we build two instances of LBM, one rotting and one rising, in which the oracle greedy strategy suffers linear regret. We highlight that the other strategy exhibited, which performs better than oracle greedy, may not be optimal.

Rotting instance. Let $\mathcal{A} = \mathcal{B}_d$, $\theta^* = e_1$, $m = d-1$, and A such that

$$A(a_1, \dots, a_m) = \left(I_d + \sum_{s=1}^{m} a_s a_s^\top\right)^{-\gamma},$$

for some $\gamma > 0$ to be specified later. Oracle greedy, which plays at each time step $a_t^{\text{greedy}} = \text{argmax}_{a\in\mathcal{A}}\langle a, A_{t-1}\theta^*\rangle$, constantly plays e_1. After the first m pulls, it collects a reward of $1/d^\gamma$ at every time step. On the other side, the strategy that plays cyclically the block $e_1 \dots e_d$ collects a reward of 1 every $d = m+1$ time steps, i.e., an average reward of $1/d$ per step. Hence, up to the transitive first m pulls, the cumulative reward of oracle greedy after T rounds is T/d^γ, and that of the cyclic policy is T/d. The regret of oracle greedy is thus at least

$$T\left(\frac{1}{d} - \frac{1}{d^\gamma}\right),$$

which is linear for $\gamma > 1$.

Rising instance. Let $m \geq 1$, $d = 2$, $\mathcal{A} = \mathcal{B}_2$, $\theta^* = (\varepsilon, 1)$ where $\varepsilon > 0$ is to be specified later, and A such that

$$A(a_1, \ldots, a_m) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \sum_{s=1}^{m} a_s a_s^\top .$$

Oracle greedy constantly plays e_1 collecting a reward of $(m+1)\theta_1^*$ from round $m+1$ onward. On the other side, the strategy that plays constantly e_2 collects a reward of $m\theta_2^*$ from round $m+1$ onward. Hence, the regret of oracle greedy from round $m+1$ onward is at least $(T-m)[m-(m+1)\varepsilon]$, which is linear for $\varepsilon < m/(m+1)$. $\square$

Hence, one cannot neglect non-stationarity as it is deleterious for the learner's performance. Instead, it is necessary to find new sophisticated strategies, which may include long-term planning. Before describing our analysis and solution of this problem, we highlight that LBMs may also be generalized to contextual bandits [Lattimore and Szepesvári, 2020], but cannot be reduced to adversarial linear bandits [Lattimore and Szepesvári, 2020].

Remark 5.1 (Contextual bandits)**.** *In contextual bandits, at each time step t the learner is provided a context c_t (e.g., data about a user). The learner then picks an action $a_t \in \mathcal{A}$ (based on c_t), and receives a reward whose expectation depends linearly on the vector $\psi(c_t, a_t) \in \mathbb{R}^d$, where ψ is a known feature map. Note that it is equivalent to have the learner playing actions $a_t \in \mathbb{R}^d$ that belong to a subset $\mathcal{A}_t = \{\psi(c_t, a) \in \mathbb{R}^d : a \in \mathcal{A}\}$. The analysis developed for LBM still holds when $\mathcal{A}_t$ depends on t, and can thus be generalized to contextual bandits with memory.*

Remark 5.2 (Adversarial linear bandits)**.** *The reduction of LBM to adversarial linear bandits is impossible. Indeed, we consider an instance of policy regret, which allows us to write the regret as:*

$$R_T = \max_{b_1,\ldots,b_T \in \mathcal{A}^T} \sum_{t=1}^{T} \langle b_t, A(b_{t-m}, \ldots, b_{t-1})\theta^* \rangle - \mathbb{E}\left[\sum_{t=1}^{T} \langle a_t, A(a_{t-m}, \ldots, a_{t-1})\theta^* \rangle\right] . \tag{5.5}$$

Instead, the regret in adversarial linear bandits is

$$R_T = \max_{a \in \mathcal{A}} \sum_{t=1}^{T} \langle a, X_t \rangle - \mathbb{E}\left[\sum_{t=1}^{T} \langle a_t, X_t \rangle\right] . \tag{5.6}$$

Note that, while in Equation (5.5) the learner competes against the optimal trajectory, in Equation (5.6) the competitor is the best single action in hindsight. Moreover, it is not possible to use the substitution $X_t = A(a_{t-m}, \ldots, a_{t-1})\theta^*$ since the adversarial loss vectors are the same for the learner and the comparator but different in our definition of the regret.

5.3 The approximation errors

After defining the LBM model, we start by focusing on cyclic policies. Although the optimal policy may not be cyclic, it is possible to show that cyclic policies provide a reasonable approximation to the optimal policy while being easier to learn. The difficulty lies in the fact that finding a block of actions maximizing the sum of expected rewards is not a well-defined problem for LBM. Indeed, the initial conditions, determined by the m actions preceding the current block, influence the rewards. To overcome this issue, we introduce the following proxy reward function which considers a block of size $m+L$ and counts only for the rewards obtained from action a_{m+1} onward. For any $m, L \geq 1$ and any block $\boldsymbol{a} = a_1, \ldots, a_{m+L}$ of $m+L$ actions, let

$$\widetilde{r}(\boldsymbol{a}) = \sum_{t=m+1}^{m+L} \left\langle a_t, A_{t-1}\theta^* \right\rangle = \sum_{t=m+1}^{m+L} \left\langle A_{t-1}a_t, \theta^* \right\rangle . \tag{5.7}$$

Although $\widetilde{r}(\boldsymbol{a})$ counts only for the last L rewards collected in the block, the first $a_1, \ldots, a_m$ actions still play a vital role as they influence the following rewards in the block, since they are used in the computation of $A_m, \ldots, A_{2m-1}$. The relevant aspect is that $\widetilde{r}(\boldsymbol{a})$ is independent from the initial state of the block, so that

$$\widetilde{\boldsymbol{a}} = \underset{\boldsymbol{a} \in \mathcal{B}_d^{m+L}}{\operatorname{argmax}} \ \widetilde{r}(\boldsymbol{a}) \tag{5.8}$$

is well-defined. Considering a block $\widetilde{\boldsymbol{a}}$ of size $m+L$, it is relevant to quantify the approximation error incurred when playing cyclically $\widetilde{\boldsymbol{a}}$ compared to the optimal sequence of actions $(a_t^*)_{t=1}^T$ defined in Equation (5.4). To analyze this approximation error, it is necessary to define the maximal and minimal instantaneous reward one can obtain. To compute it, we introduce the notation $R = \sup_{a_1, \ldots, a_{m+1} \in \mathcal{A}} \left|\langle a_{m+1}, A(a_1, \ldots, a_m)\theta^* \rangle\right|$. In (5.12) we will provide a bound on R in terms of m and γ. In the meantime we use R to define the approximation error in the next proposition, showing that it is tight up to constant.

Proposition 5.2. *For any $m, L \geq 1$, let $\widetilde{\boldsymbol{a}}$ be the block of $m+L$ actions defined in (5.8) and $(\widetilde{r}_t)_{t=1}^T$ be the expected rewards collected when playing cyclically $\widetilde{\boldsymbol{a}}$. We have*

$$\mathrm{OPT} - \sum_{t=1}^{T} \widetilde{r}_t \leq \frac{2mR}{m+L} T \ . \tag{5.9}$$

Proof. Recall that the optimal sequence is denoted $(a_t^*)_{t=1}^T$ and collects rewards $(r_t^*)_{t=1}^T$. Let $L > 0$; by definition, there exists a block of actions of length L in $(a_t^*)_{t=1}^T$ with an average expected reward higher than OPT/T. Let t^* be the first index of this block, we thus have $(1/L)\sum_{t=t^*}^{t^*+L-1} r_t^* \geq \mathrm{OPT}/T$. However, this average expected reward is realized only using the initial matrix A_{t^*-1}, generated from $a_{t^*-1}^*, \ldots, a_{t^*-m}^*$. Let $\boldsymbol{a}^* = a_{t^*-m}^*, \ldots, a_{t^*+L-1}^*$ of length $m+L$. Note that, by definition, we have that $\widetilde{r}(\widetilde{\boldsymbol{a}}) \geq \widetilde{r}(\boldsymbol{a}^*) = \sum_{t=t^*}^{t^*+L-1} r_t^* \geq L\ \mathrm{OPT}/T$. Furthermore, by (5.12), when playing cyclically $\widetilde{\boldsymbol{a}}$ one obtains at least a reward of $-R$ in each one of the first m pulls of the block. Collecting all the pieces, we obtain

$$\begin{aligned}
\sum_{t=1}^{T} \widetilde{r}_t &\geq \frac{T}{m+L}\Big(-mR + \widetilde{r}(\widetilde{\boldsymbol{a}})\Big) \\
&\geq \frac{T}{m+L}\Big(-mR + \widetilde{r}(\boldsymbol{a}^*)\Big) \\
&\geq \frac{T}{m+L}\left(-mR + L\,\frac{\text{OPT}}{T}\right) \\
&= \frac{L}{m+L}\text{OPT} - \frac{mR}{m+L}T \\
&\geq \frac{L}{m+L}\text{OPT} + \frac{m}{m+L}\text{OPT} - \frac{mR}{m+L}T - \frac{mR}{m+L}T \qquad (5.10) \\
&= \text{OPT} - \frac{2mR}{m+L}T\,,
\end{aligned}$$

where (5.10) derives from $\text{OPT} \leq RT$. □

Looking at Proposition 5.2, one can notice how the cycle length L on the right-hand side of Equation (5.9) is as expected. Indeed, by increasing L, the cyclic policy achieves an expected reward closer to OPT. We also highlight that for $m = 0$ we recover the stationary behaviour, where there are no long-term effects and the performance is oblivious to the block length. In this case, we recover $\sum_{t=1}^{T} \widetilde{r}_t = OPT$ independently of L. The following step is to show that Proposition 5.2 is tight up to constants.

Proposition 5.3 (Tight approximation)**.** *For any $m, L \geq 1$ and $\gamma \leq 0$, let $\widetilde{\boldsymbol{a}}$ be the block of $m+L$ actions defined in (5.8) and $(\widetilde{r}_t)_{t=1}^{T}$ be the expected rewards collected when playing cyclically $\widetilde{\boldsymbol{a}}$. Then, there exists a choice of $\mathcal{A}$ and θ^* such that*

$$\text{OPT} - \sum_{t=1}^{T} \widetilde{r}_t \geq \frac{mR}{m+L}T\,. \qquad (5.11)$$

Proof. Let $d = m+1$, $\mathcal{A} = \{0_d\} \cup (e_k)_{k\leq d}$, $\theta^* = (1/\sqrt{d}, \dots, 1/\sqrt{d})$, and $\gamma \leq 0$. For simplicity, we note the basis modulo d, i.e., $e_{k+d} = e_k$ for any $k \in \mathbb{N}$. Note that for any $a_1, \dots, a_{m+1} \in \mathcal{A}$ we have $\big|\langle a_{m+1}, A_m\theta^*\rangle\big| \leq \|a_{m+1}\|_1\ \|A_m\theta^*\|_\infty \leq 1/\sqrt{d}$, such that one can take $R = 1/\sqrt{d}$. Observe now that the strategy which plays cyclically $e_1, \dots, e_d$ collects a reward of $1/\sqrt{d}$ at each time step, which is optimal, such that $\text{OPT} = T/\sqrt{d}$. Further, it is easy to check that block $\widetilde{\boldsymbol{a}}$, composed of m pulls of 0_d followed by $e_1, \dots, e_L$ satisfies $\widetilde{r}(\widetilde{\boldsymbol{a}}) = L/\sqrt{d}$, which is optimal for similar reasons. Playing cyclically $\widetilde{\boldsymbol{a}}$, one gets a reward of $L/\sqrt{d}$ every $m+L$ pulls. In other terms, we have

$$\text{OPT} - \sum_{t=1}^{T} \widetilde{r}_t = \frac{T}{\sqrt{d}} - \frac{L}{m+L}\frac{T}{\sqrt{d}} = \frac{m}{m+L}\frac{T}{\sqrt{d}} = \frac{mR}{m+L}T\,.$$

□

To clarify the statements of the previous propositions, we show that it is easy to compute upper bounds on R. Let $a_1, \ldots, a_{m+1} \in \mathcal{A}$, and $A_m = A(a_1, \ldots, a_m)$, we have that

$$|r_m| = \left|\langle a_{m+1}, A_m \theta^* \rangle\right| \leq \|a_{m+1}\|_2 \, \|A_m \theta^*\|_2 \leq \|A_m\|_* \, \|\theta^*\|_2 \leq (m+1)^{\gamma^+} , \quad (5.12)$$

such that one can take $R = (m+1)^{\gamma^+}$. Note that any other choice of dual norms could have been used to upper bound $\left|\langle a_{m+1}, A_m \theta^* \rangle\right|$, as done in Proposition 5.3. For simplicity, we restrict ourselves to the Euclidean norm from now on, and use $R = (m+1)^{\gamma^+}$.
Before moving to the estimation problem, we highlight the necessity of optimizing over the first m actions since there is not a "pre-sequence" of actions universally optimal for every block.

Remark 5.3 (On the necessity of optimizing over the first actions.)**.** *We highlight that optimizing over the first m actions in Equation (5.8) is necessary, as there exists no such "pre-sequence" which is universally optimal. Indeed, let A_t and A'_t be the memory matrices generated by actions $a_1 \ldots a_{m+L}$ and $a'_1 \ldots a'_m \, a_{m+1} \ldots a_{m+L}$ respectively. It is of immediate observation that if the pre-sequence $a_1 \ldots a_m$ is better than $a'_1 \ldots a'_m$ with respect to some model $\theta \in \mathbb{R}^d$, i.e., if we have that $\sum_{t=m+1}^{m+L} \langle a_t, A_{t-1}\theta \rangle \geq \sum_{t=m+1}^{m+L} \langle a_t, A'_{t-1}\theta \rangle$, then the opposite holds true for $-\theta$. Hence, one cannot determine a priori a good pre-sequence and has to optimize for it.*

5.4 The estimation problem

In the previous section, we showed that for every block length there exists a cyclic policy providing a reasonable approximation compared to OPT, and that this approximation cannot be improved in general. Since we adopt a cyclic policy, our next objective is to build a sequence of blocks with small regret against $\widetilde{\boldsymbol{a}}$. Here, we show that learning the optimal block in the cyclic policy can be seen as a stationary linear bandit problem equipped with a specific set of actions. We show how this problem can be solved using the OFUL algorithm. We start presenting a naive solution and build on it to provide a refined approach which exploits the structure of the latent parameter θ^*.
We introduce some notation used in the rest of this chapter. Since we consider blocks of length $m+L$, let $\boldsymbol{\theta}^* = (0_d, \ldots, 0_d, \theta^*, \ldots, \theta^*) \in \mathbb{R}^{d(m+L)}$ be the vector concatenating m times 0_d and L times θ^*. Looking at the right-hand side of Equation (5.7), we introduce a subset of $\mathbb{R}^{d(m+L)}$ composed of blocks $\boldsymbol{b} = b_1 \ldots b_{m+L}$ whose actions are of the form $b_i = A_{i-1}a^{(i)}$ for some block $\boldsymbol{a} \in \mathcal{A}^{m+L}$.
Let

$$\boldsymbol{\mathcal{B}} = \left\{ \boldsymbol{b} \in \mathbb{R}^{d(m+L)} \colon \exists \, \boldsymbol{a} \in \mathcal{A}^{m+L} \text{ such that } \begin{cases} b_i = a^{(i)} & 1 \leq i \leq m \\ b_i = A_{i-1}a^{(i)} & m+1 \leq i \leq m+L \end{cases} \right\},$$

where the $(A_i)_{i=m+1}^{m+L-1}$ are the memory matrices generated from $\boldsymbol{a}$. Using this notation, it is easy to see that for any $\boldsymbol{a} \in \mathcal{A}^{m+L}$ and the corresponding $\boldsymbol{b} \in \boldsymbol{\mathcal{B}}$ we have $\widetilde{r}(\boldsymbol{a}) = \langle \boldsymbol{b}, \boldsymbol{\theta}^* \rangle$. Therefore, estimating $\widetilde{\boldsymbol{b}}$, which is the block in $\boldsymbol{\mathcal{B}}$ associated

to $\widetilde{\boldsymbol{a}}$, reduces to a standard stationary linear bandit problem in $\mathbb{R}^{d(m+L)}$, with parameter $\boldsymbol{\theta}^*$ and feasible set $\boldsymbol{\mathcal{B}}$. By doing this, we have transformed the non-stationarity of the rewards into a constraint on the action set.
Since the problem can be now seen as a stationary linear bandit, it seems reasonable to adopt OFUL [Abbasi-Yadkori et al., 2011]. We adapt it by playing at time step $t = \tau(m+L)$, the block $\boldsymbol{a}_\tau \in \mathcal{A}^{m+L}$, whose associated block $\boldsymbol{b}_\tau$ in $\boldsymbol{\mathcal{B}}$ satisfies

$$\boldsymbol{b}_\tau = \operatorname*{argmax}_{\boldsymbol{b}\in\boldsymbol{\mathcal{B}}} \sup_{\boldsymbol{\theta}\in\boldsymbol{\mathcal{C}}_{\tau-1}} \langle \boldsymbol{b}, \boldsymbol{\theta} \rangle \,, \tag{5.13}$$

where $\boldsymbol{\mathcal{C}}_\tau = \left\{\boldsymbol{\theta} \in \mathbb{R}^{d(m+L)} \colon \left\|\widehat{\boldsymbol{\theta}}_\tau - \boldsymbol{\theta}\right\|_{\boldsymbol{V}_\tau} \le \boldsymbol{\beta}_\tau(\delta)\right\}$, with $\boldsymbol{\beta}_\tau(\delta)$ defined in Equation (A.4),
$\boldsymbol{V}_\tau = \sum_{\tau'=1}^{\tau} \boldsymbol{b}_{\tau'}\boldsymbol{b}_{\tau'}^\top + \lambda I_{d(m+L)}$, $\boldsymbol{y}_\tau = \sum_{i=m+1}^{m+L} y_{\tau,i}$, using $y_{\tau,i}$ to denote the reward obtained by the i^{th} action of block τ, and $\widehat{\boldsymbol{\theta}}_\tau = \boldsymbol{V}_\tau^{-1}\big(\sum_{\tau'=1}^{\tau} \boldsymbol{y}_{\tau'}\boldsymbol{b}_{\tau'}\big)$. Knowing that $\|\boldsymbol{\theta}^*\|_2^2 \le L$, that for any block $\boldsymbol{b} \in \boldsymbol{\mathcal{B}}$ we have $\|\boldsymbol{b}\|_2^2 \le m + L(m+1)^{2\gamma^+}$ and $\langle \boldsymbol{\theta}^*, \boldsymbol{b}\rangle \le L(m+1)^{\gamma^+}$, it is possible to adapt the OFUL's analysis and get the following regret bound.

Proposition 5.4. *Let $\lambda \in [1, d]$, $L \ge m$, and $\boldsymbol{a}_\tau$ be the blocks of actions in $\mathbb{R}^{d(m+L)}$ associated to the $\boldsymbol{b}_\tau$ defined in (5.13). Then we have*

$$\mathbb{E}\left[\sum_{\tau=1}^{T/(m+L)} \widetilde{r}(\widetilde{\boldsymbol{a}}) - \widetilde{r}(\boldsymbol{a}_\tau)\right] = \widetilde{\mathcal{O}}\Big(dL^{3/2}(m+1)^{\gamma^+}\sqrt{T}\Big) \,.$$

We defer the proof of Proposition 5.4 in Appendix A.3. We notice that when $m = 0$ and $L = 1$, we recover the stationary case and the block approach coincides with OFUL. In this case, we recover (up to log factors) the $\mathcal{O}(d\sqrt{T})$ bound for standard linear bandits. In the following proposition, we prove a more general and stronger high-probability bound, which also specializes to known results for linear bandits in the stationary case.

Proposition 5.5. *Let $\lambda \ge 1$, $\delta \in (0,1)$, and $\boldsymbol{a}_\tau$ be the blocks of actions in $\mathbb{R}^{d(m+L)}$ associated to the $\boldsymbol{b}_\tau$ defined in (5.13). Then, with probability at least $1-\delta$ we have*

$$\begin{aligned}\sum_{\tau=1}^{T/(m+L)} \widetilde{r}(\widetilde{\boldsymbol{a}}) - \widetilde{r}(\boldsymbol{a}_\tau) \le 4L(m+1)^{\gamma^+}\sqrt{Td\,\ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d(m+L)\lambda}\right)} \\ \cdot\left(\sqrt{\lambda L} + \sqrt{\ln\left(\frac{1}{\delta}\right) + d(m+L)\,\ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d(m+L)\lambda}\right)}\right) .\end{aligned}$$

For this result as well we defer the proof in Appendix A.3.
One can notice how the approach presented above is wasteful. Indeed, in the naive approach, we estimate a latent parameter $\widehat{\boldsymbol{\theta}}_\tau$ which is a concatenated vector of parameters $\boldsymbol{\theta}^* \in \mathbb{R}^{d(m+L)}$, with degraded accuracy due to the increased dimension. However, the only parameter relevant to our model is $\theta^* \in \mathbb{R}^d$. Furthermore, the proposed method uses only the sum of the rewards obtained by a block, while the learner has access to finer-grained information, namely the

rewards obtained by each individual action $a_1, \dots, a_{m+L}$ in the block. To exploit this information, we propose a new refined approach. Let $\boldsymbol{a}_\tau = a_{\tau,1} \dots a_{\tau,m+L}$ be the block of actions played at block time step τ, $A_{\tau,i-1} = A(a_{\tau,i-m}, \dots, a_{\tau,i-1})$, and $b_{\tau,i} = A_{\tau,i-1} a_{\tau,i}$ for $i \geq m$. Instead of computing $\widehat{\boldsymbol{\theta}}_\tau$, we estimate the following parameter:

$$\widehat{\theta}_\tau = V_\tau^{-1} \left(\sum_{\tau'=1}^{\tau} \sum_{i=m+1}^{m+L} y_{\tau',i}\, b_{\tau',i} \right) , \tag{5.14}$$

where $V_\tau = \sum_{\tau'=1}^{\tau} \sum_{i=m+1}^{m+L} b_{\tau',i} b_{\tau',i}^\top + \lambda I_d$. Here, $\widehat{\theta}_\tau$ is the standard regularized least square estimator of θ^* when only the last L rewards of each block of size $m+L$ are considered. However, the $\widehat{\theta}_\tau$ are only computed every $m+L$ rounds. The regret is computed here at the block level, such that at each block time step τ the learner chooses upfront an entire block to play, preventing from updating the estimates between the individual actions of the block. Following the principle of optimism in the face of uncertainty, a natural strategy then consists of playing

$$\boldsymbol{a}_\tau = \underset{a_{\tau,i} \in \mathcal{A}}{\operatorname{argmax}} \sup_{\theta \in \mathcal{C}_{\tau-1}} \sum_{i=1}^{L} \langle a_{\tau,i}, A_{\tau,i-1}\theta \rangle , \tag{5.15}$$

where $\mathcal{C}_\tau = \left\{\theta \in \mathbb{R}^d \colon \left\|\widehat{\theta}_\tau - \theta\right\|_{V_\tau} \leq \beta_\tau(\delta)\right\}$, for some $\beta_\tau(\delta)$ defined in (A.5). Expressed in terms of $\boldsymbol{b}_\tau$, the estimate (5.15) corresponds to

$$\boldsymbol{b}_\tau = \underset{\boldsymbol{b} \in \boldsymbol{\mathcal{B}}}{\operatorname{argmax}} \sup_{\boldsymbol{\theta} \in \boldsymbol{\mathcal{D}}_{\tau-1}} \langle \boldsymbol{b}, \boldsymbol{\theta} \rangle , \tag{5.16}$$

where $\boldsymbol{\mathcal{D}}_\tau = \left\{\boldsymbol{\theta} \in \mathbb{R}^{d(m+L)} \colon \exists \theta \in \mathcal{C}_\tau \text{ such that } \boldsymbol{\theta} = (0_d, \dots, 0_d, \theta, \dots, \theta)\right\}$. This estimate is similar to (5.13), except for the improved confidence set $\boldsymbol{\mathcal{D}}_\tau$ that leverages the structure of $\boldsymbol{\theta}^*$. A dedicated analysis to deal with the fact that the estimates $\widehat{\theta}_\tau$ are not "up to date" for actions inside the block then allows bounding the regret of the sequence $\boldsymbol{a}_\tau$ against the optimal $\widetilde{\boldsymbol{a}}$. Setting the block size L to balance this bound with the approximation error of Proposition 5.2 yields the final regret bound.

Theorem 5.6. *Let $\lambda \in [1, d]$, and $\boldsymbol{a}_\tau$ be the blocks of actions in $\mathbb{R}^{d(m+L)}$ defined in (5.15). Then we have*

$$\mathbb{E}\left[\sum_{\tau=1}^{T/(m+L)} \widetilde{r}(\widetilde{\boldsymbol{a}}) - \widetilde{r}(\boldsymbol{a}_\tau)\right] = \widetilde{\mathcal{O}}\Big(dL(m+1)^{\gamma^+}\sqrt{T}\Big) .$$

Suppose that $m \geq 1$, $T \geq d^2m^2 + 1$, and set $L = \left\lceil \sqrt{m/d}\; T^{1/4} \right\rceil - m$. Let y_t be the rewards collected when playing $\boldsymbol{a}_\tau$ as defined in (5.15). Then we have

$$\mathrm{OPT} - \mathbb{E}\left[\sum_{t=1}^{T} y_t\right] = \widetilde{\mathcal{O}}\left(\sqrt{d}\; (m+1)^{\frac{1}{2}+\gamma^+}\, T^{3/4}\right) .$$

When $m = 0$ (i.e., in the stationary case), setting $L = 1$ recovers the `OFUL` *bound.*

See Appendix A.4 for the proof of Theorem 5.6. From these bounds, one can notice how the refined approach and the improved confidence bounds lead to an improvement in the dependence of L, which is reduced from $L^{3/2}$ to L. If one would have solved the approximation-estimation tradeoff using Proposition 5.4, this would lead to an overall regret bound of order $d^{2/5}(m+1)^{\frac{3}{5}+\gamma^+} T^{4/5}$, worse than the bound provided by the second claim of Theorem 5.6.
Finding a lower bound matching Theorem 5.6 for arbitrary values of m and γ remains an open problem. We highlight that lower bounds for nonstationary bandits are particularly hard to obtain, and that most papers on this topic do not prove any, see e.g., Levine et al. [2017]; Kleinberg and Immorlica [2018]; Pike-Burke and Grunewalder [2019]; Cella and Cesa-Bianchi [2020]; Metelli et al. [2022]. Yet, Proposition 5.3 shows that the control of the approximation error provided by Proposition 5.2 is optimal up to constants. Moreover, our estimation error is tight in general, as in the stationary case (i.e., $m = 0$) Theorem 5.6 matches the lower bound for stationary linear bandits, see e.g., [Lattimore and Szepesvári, 2020, Theorems 24.1 and 24.2]. We obtain that our regret bound cannot be improved (up to log factors) by strategies based on our approximation/estimation decomposition when the action set is the hypercube or the unit ball. As we can see from the optimal choice of L in Theorem 5.6, `OFUL-memory` requires the knowledge of the horizon T, the memory size m, and the exponent γ, which might all be unknown in practice. While adaptation to T can be achieved by using the doubling trick, an adaptation to m and γ is more involved. Before discussing this problem, showing that `OFUL-memory` can be wrapped by a model selection algorithm to learn m and γ, we state a few remarks and show the implementation of the algorithms.

Remark 5.4 (An over-optimistic variant)**.** *Note that* $\boldsymbol{\mathcal{D}}_\tau = \left\{\boldsymbol{\theta} \in \mathbb{R}^{d(m+L)} : \exists \theta \in \mathcal{C}_\tau \text{ such that } \boldsymbol{\theta} = (0_d, \dots, 0_d, \theta, \dots, \theta)\right\}$ *is not the only improved confidence set that one can build from* $\mathcal{C}_\tau$*. Indeed, it is immediate to check that our proof remains unchanged if one uses instead* $\boldsymbol{\mathcal{D}}_\tau^{opt} = \left\{\boldsymbol{\theta} \in \mathbb{R}^{d(m+L)} : \exists\, \theta_1, \dots, \theta_L \in \mathcal{C}_\tau \text{ such that } \boldsymbol{\theta} = (0_d, \dots, 0_d, \theta_1, \dots, \theta_L)\right\}$*. Optimizing* (5.16) *over* $\boldsymbol{\mathcal{D}}_{\tau-1}^{opt}$ *and not* $\boldsymbol{\mathcal{D}}_{\tau-1}$ *creates an over-optimistic block version of the UCB, composed of the sum of the UCBs of the single-actions in the block, although the latter might be attained at different models* θ_i*, while we know that* $\boldsymbol{\theta}^*$ *is the same model* θ^* *repeated* L *times. Still, since each* θ_i *is estimated in the confidence set* $\mathcal{C}_{\tau-1}$ *of reduced dimension, the guarantees are unchanged. In the rest of the paper, we refer to this variant as the* over-optimistic *version of* `OFUL-memory`*, denoted by* `O3M`.

Remark 5.5 (Generic matrix mapping A)**.** *Note that our analysis naturally extends to any matrix mapping* A*, as long as the latter is known. The term* $(m+1)^{\gamma^+}$ *in Theorem* 5.6 *is then replaced with* $\sup_{a_1 \dots a_m} \|A(a_1, \dots, a_m)\|_*$*. We stress that changing the monomial in Equation* (5.2) *into a polynomial does not affect the algorithm. In the adaptive case, looking for polynomials just consists of tracking more parameters (namely, the different coefficients of the polynomial). We highlight however that having access to such knowledge is unlikely in practice. This is why focus on the simpler parametric family* (5.2)*, which encompasses rotting and rising scenarios while allowing us to learn simultaneously* m *and* γ*, as shown in the next section.*

Algorithm 3 `OFUL-memory (OM, O3M)`

```
input : action space $\mathcal{A} \subset \mathbb{R}^d$, memory size $m$, exponent $\gamma$, regularization
        parameter $\lambda$, horizon $T$.
init  : set $L = \sqrt{m/4\,d}\; T^{1/4} - m$,  $\widehat{\theta}_0 = 0_d$,  $V_0 = \lambda I_d$,  $\beta_0 = 0$.
for $\tau = 1, \dots, T/(m+L)$ do
    // OM                                        // O3M
    $\boldsymbol{a}_\tau = \operatorname{argmax}_{a_{\tau,i} \in \mathcal{A}} \sup_{\theta \in \mathcal{C}_{\tau-1}} \sum_{i=1}^{L} \langle a_{\tau,i}, A_{\tau,i-1}\theta \rangle$   or   $\boldsymbol{a}_\tau = \operatorname{argmax}_{a_{\tau,i} \in \mathcal{A}} \sup_{\theta_i \in \mathcal{C}_{\tau-1}} \sum_{i=1}^{L} \langle a_{\tau,i}, A_{\tau,i-1}\theta_i \rangle$
    // Play and update confidence set
    Play $\boldsymbol{a}_\tau$, collect $y_{\tau,1}, \dots, y_{\tau,m+L}$, and compute $\mathcal{C}_\tau$, i.e., $\widehat{\theta}_\tau$, $V_\tau$, and $\beta_\tau$ via
    (5.14) and (A.5).
```

When we allow for a generic matrix A, we can extend the set of recovered bandits with the following example.

Example 5.3 (Bandits with delay-dependent rewards)**.** *In bandits with delay-dependent rewards [Kleinberg and Immorlica, 2018; Cella and Cesa-Bianchi, 2020; Laforgue et al., 2022], the expected reward of an arm is given by $\mu_k(\tau_k(t))$, where $\tau_k(t)$ is the delay of arm k, i.e., $\tau_k(t) = t - t_k$ and $t_k \leq t$ is the last time step when arm k was played. Following Example 5.2, it is easy to build a diagonal matrix that corresponds to this model. Note that this construction may be extended to a bandit whose arms' expected rewards at time step t are fully determined by the history $a_1, \dots, a_{t-1}$.*

A reader may see our LBM problem as a Reinforcement Learning problem. To clarify this point, we present the following remark.

Remark 5.6 (Solving LBM with a general Reinforcement Learning (RL) approach)**.** *Our setting may be seen as an MDP with a d-dimensional continuous space of actions, a (md)-dimensional continuous state space (for the past m actions), a deterministic transition function parameterized by an unknown scalar γ, and a stochastic reward function with a linear dependence on an additional d-dimensional latent parameter θ^*. The optimal policy in this MDP is generally nonstationary, and we are not aware of RL algorithms whose regret can be bounded without relying on more specific assumptions on the MDP. By exploiting the structure of the MDP, and restricting to cyclic policies, we show instead that the original problem can be solved using stationary bandit techniques.*

After clarifying the functioning of our algorithm, we discuss the practical implementations of our approaches, `OFUL-memory` (`OM`) and `over-optimistic OFUL-memory` (`O3M`, see Remark 5.4), both summarized in Algorithm 3.

Maximizing the UCBs. We start by making explicit the UCBs used in `OM` and `O3M`, see (5.16), optimized over $\boldsymbol{\mathcal{D}}_\tau$ or $\boldsymbol{\mathcal{D}}_\tau^{\text{opt}}$. Using the formula for $\mathcal{C}_\tau$, one can check that they are given by $\mathrm{UCB}_\tau(\boldsymbol{a}) = \sum_{j=m+1}^{m+L} \left\langle a^{(j)}, A_{j-1}\widehat{\theta}_{\tau-1} \right\rangle + B(\boldsymbol{a})$, where

$$B(\boldsymbol{a}) = \beta_{\tau-1} \Big\| \sum_{j=m+1}^{m+L} A_{j-1}^\top a^{(j)} \Big\|_{V_{\tau-1}^{-1}} \quad \text{(OM)} \tag{5.17}$$

for `OM` and

$$B(\boldsymbol{a}) = \beta_{\tau-1} \Big\| A_{j-1}^\top a^{(j)} \Big\|_{V_{\tau-1}^{-1}} \quad \text{(O3M)} \tag{5.18}$$

for `O3M`. The two UCBs only differ in their exploration bonuses. Note that by the triangle inequality, we have $\mathrm{UCB}_\tau^{\mathrm{OM}}(\boldsymbol{a}) \leq \mathrm{UCB}_\tau^{\mathrm{O3M}}(\boldsymbol{a})$ for any $\boldsymbol{a}$. Thanks to this closed form in terms of $\boldsymbol{a}$, it is possible to solve $\mathrm{argmax}_{\boldsymbol{a}} \mathrm{UCB}_\tau(\boldsymbol{a})$, using gradient ascent.

Computational complexity. As described in Algorithm 3, our approach consists of two steps: updating the confidence region C_τ, i.e., $\widehat{\theta}_\tau$ and β_τ according to (5.14) and (A.5), and computing the block $\boldsymbol{a}_\tau$ that maximizes the UCB index. The first step is performed by online Ridge Regression, and has a computational cost of $\mathcal{O}(Ld^2)$. We note here the advantage of our refined algorithm over the naive concatenated approach, whose Ridge regression update has cost $\mathcal{O}(L^2d^2)$. The maximization of the UCB indices, performed through Gradient Ascent (GA) has time complexity per iteration of $\mathcal{O}\big((m+L)d^2\big)$. Hence, the overall complexity of an epoch of Algorithm 3 is $\mathcal{O}\big((m+L)d^2 \cdot n_{\mathrm{it}}\big)$, where n_{it} is the number of iterations performed by GA. Recall that the epochs of Algorithm 3 correspond to blocks of $m+L$ actions, such that the actual per-round complexity is $\mathcal{O}(d^2 \cdot n_{\mathrm{it}})$.

5.5 An approach for model selection: Bandit Combiner

As mentioned before, we provide a solution to the case where the parameters m and γ of a LBM are unknown. Indeed, in the absence of prior knowledge on the nature of the non-stationary mechanism at work, a natural idea consists of instantiating several LBMs with different values of γ and running a model selection algorithm for bandits [Foster et al., 2019; Cutkosky et al., 2020; Pacchiano et al., 2020]. In bandit model selection, where a master algorithm runs the different LBMs, the adaptation to the memory size m becomes more complex. Indeed, the different putative values for m induce different block sizes (see Theorem 5.6) which perturb the time and reward scales of the master algorithm. For instance, bandits with larger block length will collect more rewards per block, although they might not be more efficient on average. Our solution consists in feeding the master algorithm with averaged rewards. One may then control the true regret (i.e., not averaged) of the output sequence, against a scaled version of the optimal sequence through Lemma 5.7, which links the normalized regret of a block meta-algorithm to the true regret of the corresponding sequence of blocks.

Lemma 5.7. *Suppose that a block-based bandit algorithm (in our case the bandit combiner) produces a sequence of T_{bc} blocks $\boldsymbol{a}_\tau$, with possibly different cardinalities $|\boldsymbol{a}_\tau|$, such that*

$$\sum_{\tau=1}^{T_{\mathrm{bc}}} \frac{\widetilde{r}(\widetilde{\boldsymbol{a}})}{|\widetilde{\boldsymbol{a}}|} - \sum_{\tau=1}^{T_{\mathrm{bc}}} \frac{\widetilde{r}(\boldsymbol{a}_\tau)}{|\boldsymbol{a}_\tau|} \leq F(T_{\mathrm{bc}}) \,,$$

for some sublinear function F. Then, we have

$$\frac{\min_\tau |\boldsymbol{a}_\tau|}{\max_\tau |\boldsymbol{a}_\tau|} \left(\widetilde{r}(\widetilde{\boldsymbol{a}}) \frac{\sum_\tau |\boldsymbol{a}_\tau|}{|\widetilde{\boldsymbol{a}}|} \right) - \sum_{\tau=1}^{T_{\mathrm{bc}}} \widetilde{r}(\boldsymbol{a}_\tau) \;\leq\; \min_\tau |\boldsymbol{a}_\tau| \, F(T_{\mathrm{bc}}) \;.$$

In particular, if all blocks have the same cardinality the last bound is just the block regret bound scaled by $|\boldsymbol{a}_\tau|$.

Combining this result with Theorem 5.6 and [Cutkosky et al., 2020, Corollary 2] yields the following corollary.

Corollary 5.8. *Consider an instance of LBM with unknown parameters $(m_\star, \gamma_\star)$. Assume a bandit combiner is run on $N \leq d\sqrt{m_\star}$ instances of `OFUL-memory` (Algorithm 4), each using a different pair of parameters (m_i, γ_i) from a set $\mathcal{S} = \big\{(m_1, \gamma_1), \ldots, (m_N, \gamma_N)\big\}$ such that $(m_\star, \gamma_\star) \in \mathcal{S}$. Let $M = (\max_j m_j)/(\min_j m_j)$. Then, for all $T \geq (m_\star + 1)^{2\gamma_\star^+}/m_\star d^4$, the expected rewards $\big(r_t^{\mathrm{bc}}\big)_{t=1}^T$ of the bandit combiner satisfy*

$$\frac{\mathrm{OPT}}{\sqrt{M}} - \mathbb{E}\left[\sum_{t=1}^T r_t^{\mathrm{bc}}\right] \;=\; \widetilde{\mathcal{O}}\Big(M\, d\, (m_\star + 1)^{1 + \frac{3}{2}\gamma_\star^+}\, T^{3/4}\Big) \,.$$

We defer the proof of Corollary 5.8 to Appendix A.5. The practical implementation of the bandit combiner algorithm builds on the approach of Cutkosky et al. [2020]. Here, we show our adaptation of the Bandit Combiner algorithm Cutkosky et al. [2020] to instances of `O3M`. The meta-algorithm is fed with different bandit algorithms, namely different instances of `O3M` with different choices of parameters m_j and γ_j, and at each round plays a block according to one of the algorithms. Each `O3M` instance comes with a *putative* regret bound $C_j T^{\alpha_j}$, which is the regret bound satisfied by the algorithm *should it be well-specified*, i.e., if the rewards are indeed generated through a memory matrix with memory m_j and exponent γ_j. Note that in order to be comparable across the different instances, the putative regrets apply to the average rewards. The values of C_j and α_j can be computed using Theorem 5.6, see the proof of Corollary 5.8 for details. The putative regrets are then used to successively discard the instances that are not well specified, and eventually identify the instance using parameters $(m_\star, \gamma_\star)$. Knowing C_j and α_j, we can compute for any j the target regret

$$R_j = C_j\, T_{\mathrm{bc}}^{2/3} + \frac{5\sqrt{30}}{18} C_j^{3/2}\, T_{\mathrm{bc}}^{2/3} + 1152 (m_j + 1)^{2\gamma_j^+}\, T^{1/3} \log(T_{\mathrm{bc}}^3 N/\delta) + (N-1) T^{2/3} \,, \tag{5.19}$$

where T_{bc} is the number of blocks the Bandit Combiner is called on, see Appendix A.5 for details. Here, we note how the presence of $(m_j+1)^{2\gamma_j^+}$ is impacting differently the rising and rotting scenarios. Using [Cutkosky et al., 2020, Corollary 2], the regret of Algorithm 4 is finally given by $3R_{j_\star}$, where $j_\star$ is the index such that $(m_{j_\star}, \gamma_{j_\star}) = (m_\star, \gamma_\star)$. The pseudo-code of the algorithm, which is an adaptation of Bandit Combiner in Cutkosky et al. [2020], is summarized in Algorithm 4.

Algorithm 4 `Bandit Combiner on Over-Optimistic OFUL-Memory (O3M)`

```
input : Instances O3M($m_1, \gamma_1$), ..., O3M($m_N, \gamma_N$), horizon $T_{\text{bc}}$
        numbers $C_1, \dots, C_N > 0$, target regrets $R_1, \dots, R_N$.
        Set $T(i) = 0, \mathcal{S}_i = 0, \Delta_i = 0$ for $i = 1, \dots, N$, and set $I_0 = \{1, \dots, N\}$
for $t = 1, \dots, T_{bc}$ do
    if there is some $i \in I_t$ with $T(i) = 0$ then
        $i_t = i$
    else
        For each $i \in I_t$, compute the UCB index:
          $\text{UCB}(i) = \min\left\{(m_i+1)^{2\gamma_i^+}, \frac{C_i}{\sqrt{T(i)}} + 4(m_i+1)^{2\gamma_i^+}\sqrt{\frac{2\log(T^3N/\delta)}{T(i)}}\right\} - \frac{R_i}{T_{\text{bc}}}$
        Set $i_t = \text{argmax}_{i \in I_t} \frac{\mathcal{S}_i}{T(i)} + \text{UCB}(i)$
    Obtain from instance O3M($m_{i_t}, \gamma_{i_t}$) a block of size $m_{i_t} + L_{i_t}$ and play it
    Return the total reward $r_{i_t}$ collected in the last $L_{i_t}$ time steps of the block
    to O3M($m_{i_t}, \gamma_{i_t}$)
    Compute the average reward $\widehat{r}_{i_t} = \frac{r_{i_t}}{L_{i_t}}$
    Update $\Delta_{i_t} \leftarrow \Delta_{i_t} + \mathcal{S}_{i_t}/T(i_t) - \widehat{r}_{i_t}$ (where we set $0/0 = 0$) and $\mathcal{S}_{i_t} \leftarrow \mathcal{S}_{i_t} + \widehat{r}_{i_t}$
    Update the number of plays $T(i_t) \leftarrow T(i_t) + 1$
    if $\Delta_{i_t} \geq C_{i_t} T(i_t)^{\gamma_{i_t}} + 12\,(m_{i_t}+1)^{2\gamma_{i_t}^+}\sqrt{2\log(T^3N/\delta)T(i_t)}$ then
        $I_t = I_{t-1} \setminus \{i_t\}$
    else
        $I_t = I_{t-1}$
```

5.6 Experiments on the LBM setting

To conclude the discussion of the LBM model, we present an experimental evaluation of our approach. We perform experiments to validate the theoretical performance of `OM` and `O3M` (Algorithm 3). Similarly to [Warlop et al., 2018], we work with synthetic data because of the counterfactual nature of the learning problem in bandits. Indeed, for a dataset to be suitable for our setting, we

would need information not only on the items actually chosen by the users and their rewards, but we would also need information and rewards of all possible ordered sequences of all the items in a dataset. Since this type of information is not provided by datasets typically available online, we examine the realization of O3M on synthetic data.
Unless stated otherwise, we set $d = 3$ while $\theta^* \in \mathbb{R}^d$ is generated uniformly at random with unit norm. The rewards are generated according to (5.1) and (5.2), and perturbed by Gaussian noise with standard deviation $\sigma = 1/10$.
Concerning the algorithm Algorithm 3, we test the `O3M` approach, since it is the best performing algorithm out of the three approaches we present, `O3M`, `OM`, and `OM-Block`. The reason why `O3M` is the best approach out of the three is that this approach exploits the knowledge that the unknown parameter θ^* is the same for every action in the block since it remains unchanged throughout the horizon. So, while other approaches consider $\boldsymbol{\theta}^* \in \mathbb{R}^{d\times(m+L)}$, `O3M` is actually exploiting the knowledge that $\boldsymbol{\theta}^* = \{0_d, \dots, 0_d, \theta^*, \dots, \theta^*\}$, where the 0_d corresponds to the first m steps while θ^* is repeated for each one of the last L steps of the block. For all these reasons, we decided to plot the best performing algorithm out of our three approaches.

Rotting with Bandit Combiner. We start by analyzing the rotting scenario with $m = 2$ and $\gamma = -3$. We measure the performance in terms of the cumulative reward averaged over 5 runs (this is enough because the variance is small). In Figure 4.3 (left pane) we compare the performance of `O3M` against oracle greedy, vanilla `OFUL`, and two instances of Bandit Combiner (Algorithm 4). The first instance, Combiner γ, works in the setting where the misspecified parameter is γ and the algorithm is run over the set $\{-4, -3, -2, -1, 0\}$ of possible values for γ with the true value being -3. The second instance, Combiner m, tests the setting where the misspecified parameter is m. In this case the algorithm is run over the set $\{0, 2, 3\}$ of possible values for m with the true value being 2. The results—see Figure 4.3 (left pane)—show that `O3M` can plan the actions in the block ensuring that a good arm is not played right away if a higher reward can be obtained later on in the block. This means that `O3M` is waiting to play certain actions until the corresponding entries of A have been offloaded, preventing A to negatively impact the reward of these actions. Although learning m proves to be more difficult, which is consistent with the impact of $M = (\max_j m_j)/(\min_j m_j)$ in Corollary 5.8, Combiner m run on instances of `O3M` is competitive with `O3M` run with the true parameters. Note that with isotropic initialization there is no point in running Combiner γ with values of γ larger than zero. Indeed, in the isotropic case oracle greedy is optimal, stationary, and with the same optimal action for any $\gamma \geq 0$. The empirical performance of our algorithms in a non-isotropic rising setting is investigated in the next example.

Rising with non-isotropic initialization. When $\gamma > 0$ (rising setting) andn$A_0 \neq I_d$ (non-isotropic initialization), there are instances for which oracle greedy is suboptimal, as we show next. Let $d = 2$, $m = 2$, $\gamma = 1$, $A_0 = e_1 e_1^\top$, and $\theta^* = (\sqrt{\epsilon}, \sqrt{1-\epsilon})$. Since γ is positive, the rising scenario is such that the more we play a direction the more its reward will increase. Due to the nature of θ^* and A_0, we see that oracle greedy starts to pull action $e_1 = (1, 0)$ and

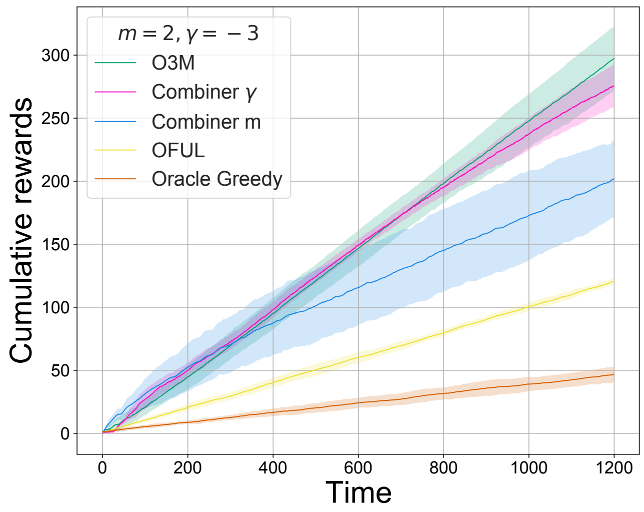

FIGURE 5.2: This figure shows the cumulative rewards obtained by the tested algorithms in rotting experiment.

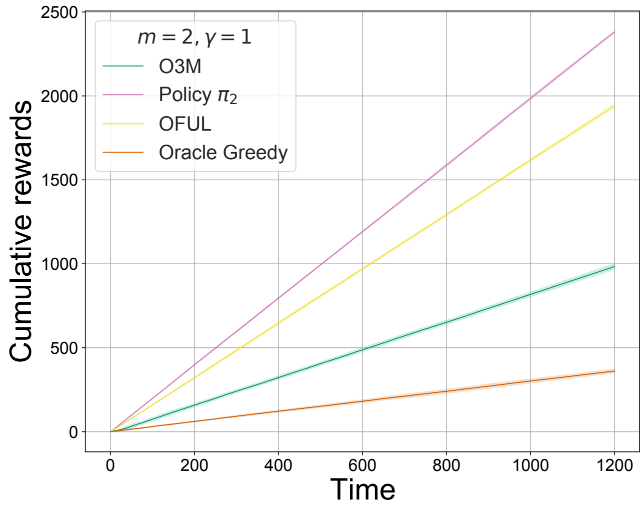

FIGURE 5.3: Here, the plot represents the cumulative rewards obtained by the tested algorithms in the rising experiment with non-isotropic initialization.

will always play it, obtaining a cumulative reward of $T(1+m)\sqrt{\epsilon}$. Instead, a better strategy would be to play $e_2 = (0, 1)$ all the time, collecting a cumulative reward of $Tm\sqrt{1-\epsilon}$. We call this strategy π_2 and in Figure 4.3 (right pane) we compare the performance of `O3M` with oracle greedy, π_2, and `OFUL`. Here `OFUL` performs well because the optimal action is stationary and, unlike oracle greedy, `OFUL` can use exploration to discover that e_2 is better than e_1. Here we show how Greedy is indeed suboptimal, as well as `O3M` and `OFUL`. We show that among the strategies we test, the best policy for this specific instance of the problem is Policy π_2, the stationary policy which plays action e_2 all the time. If we compare `OFUL` to `O3M`, we see that there is a gap between the two due to the time that `O3M` spends in trying to plan for the best actions to play in the block of size $m + L$, while `OFUL` has a learning bias since it looks for a stationary policy. After some exploration, the two algorithms tend to play the same actions. However, since we have no guarantees on what the optimal policy is and furthermore we do not have any formal guarantee that `OFUL` is better than us in the rising setting.

Moreover, we justify the loss we incur in `O3M` by highlighting that the strength of our algorithm is to be able to approach different settings, such as rotting and rising, with a unique algorithm. There are settings where it is not clear from the beginning whether the learner will meet a rotting or rising scenario. If we consider the example of music recommender systems that we present in the introduction, we can think of two possible behaviours. In one case, there may be a user who grows tired of listening to the same music genres and its level of satisfaction will decrease with time (rotting scenario). On the other side, a user may be passionate about a newly released album and would want to listen to the same songs over and over, with their level of satisfaction growing with time (rising scenario). In these situations, one may choose to adopt our algorithm since a simple adjustment of the parameter γ would allow one to move from one setting to the other. In addition to the fact that in the case where m and γ are unknown one can adopt the Bandit Combiner solution we proposed.
At the beginning of this section, we stated that we would use the `O3M` approach during the experiments since it is the best approach out of the ones proposed. We also explained how `OM-Block` it is the worst out of the three, `O3M`, `OM`, and `OM-Block`, since it does not exploit the knowledge on the concatenated vector $\boldsymbol{\theta}^*$. To support this decision, we end this section with an additional experiment comparing the regrets of `O3M` and `OM-Block`. To be able to plot the regret, one must know OPT, which in our setting is hard to compute in general. Since in the rising scenario with an isotropic initialization OPT is oracle greedy, which is easy to compute, we present this experiment in a rising setting with $m = 1$ and $\gamma = 2$. We plot the regret of `O3M` and `OM-Block` against the number of time steps, measuring the performance at different time horizons and for different sizes of L (where L depends on T). Specifically, we instantiated `O3M` and `OM-Block` for increasing values of L, setting the horizon of each instance based on the equations in Theorem 5.6 and Proposition 5.4. Figure 5.4 shows how the dimension of $\widehat{\theta}$, which is d for `O3M` and $d \times L$ for `OM-Block`, has an actual impact on the performance since `O3M` outperforms `OM-Block`.

5.7 Conclusions

In this chapter, we introduced a novel linear non-stationary bandit, called Linear Bandit with Memory, to overcome the limitations left by the previous work on LSD bandits in Chapter 4. We aimed to propose a bandit problem able to deal with non-stationarity in an infinite set of actions. Here, we showed how the LBM model, which uses a fixed-size window, can deal with cross-arm dependencies generated from the structured set of actions considered. We introduced the model and showed that it can recover stationary linear bandits as well as rested rotting and rising bandits. After computing the approximation error, we proposed three different approaches, `O3M`, `OM`, and `OM-Block`, for the

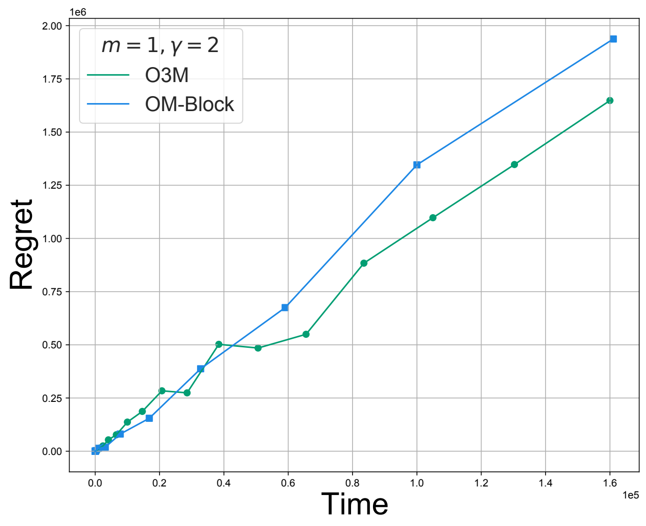

FIGURE 5.4: The plot compares the regret (y-axis) of `O3M` and `OM-Block` in the rising setting, where the computation of the regret is possible. Each dot is a separate run where the value of L is tuned to the corresponding horizon (x-axis).

estimation problem. We considered cyclic policies and showed an adaptation of the `OFUL` algorithm to find the best block of actions. In the analysis of the regret bound, we showed that when $m = 0$ the bound recovers `OFUL`'s bound for stationary linear bandits $\mathcal{O}(d\sqrt{T})$ up to log factors. When we optimize the block length L to balance the tradeoff between the approximation and estimation errors, we get a regret upper bound of order $\sqrt{d}\,(m+1)^{\frac{1}{2}+\max\{\gamma,0\}}T^{3/4}$ (ignoring log factors) against the optimal sequence of actions for $T \geq (md)^2$. We showed how this analysis holds even for a general matrix mapping A, where the only impact on the regret is the substitution of the term $(m+1)^{\gamma^+}$ in Theorem 5.6 with $\sup_{a_1 \dots a_m} \|A(a_1, \dots, a_m)\|_*$. We also provided a solution for the setting where m and γ are unknown. For this case, we proposed `Bandit Combiner` on `O3M`, and computed its regret upper bound in Corollary 5.8. Finally, we offer an experimental validation of our approach against natural baselines.

Chapter 6

The Spotify™ network

Contents

We started this dissertation by stating that we aim to study non-stationary bandit models inspired by problems faced in music recommender systems. In the previous chapters, we proposed two models which address some of the obstacles in song recommendation. Particularly, we focused on examples where the arms of the bandits were music genres and on those settings where the user's preferences are non-stationary as they depend on the user's past actions. In this chapter, we continue to focus on music recommender systems but switch to a different perspective. We specifically concentrate on one of the most popular music streaming platforms, Spotify™. The objective is to analyse the Spotify data and look for specific patterns which could provide useful insights to improve recommender systems. Starting from the data made available by the service, we analysed the directed graph of music credits, where the nodes are artists and the arc between two nodes represents a collaboration between the two. We compute different centrality measures on the graph conditioned on the musical genre, supporting and extending the results previously obtained in the analysis of the undirected graph made by [South et al., 2020]. Both [South et al., 2020] and our work show that the centrality measures for classical and hip-hop have the highest median values compared to other genres. Although we noticed the same phenomena when examining the popularity of the nodes and the number of their connections, we propose an alternative explanation to the one proposed in [South et al., 2020]. We argue that the directed graph is much more informative and allows us to distinguish between two different crediting patterns: citation and collaboration. Our contribution to this line of research is the definition of a new reciprocity index, which quantifies how much an artist is inclined to collaborate. Although our contribution concerns the analysis and this new definition of reciprocity, we point out how one of the applications of this index of reciprocity is to potentially integrate it into music recommendation systems as a feature to provide more informed recommendations to users. The contribution related to this chapter is presented in [Clerici* and Tiraboschi*, 2023.].

6.1 Background and literature review

A music streaming platform is oftentimes a resource for a user who wishes to discover new songs. As discussed in this dissertation, music recommender systems are useful for fulfilling this function. These systems usually exploit the information about items and users to quantify how much a certain user could enjoy an item. Among the abundant collection of features that could be exploited, one interesting piece of information that could be considered in this application is the collaboration among artists. This chapter focuses on extracting and analysing this information using data from Spotify™ to build a music credits network. The choice of using this data is due to Spotify™ being currently the leading music streaming platform by the number of paying users [Wikipedia, 2023] and to the fact that the data is easily accessible to third-party applications via a REST Web API [Spotify, 2021].
The analysis presented in this chapter will focus on graph-related metrics conditioned on music genre. The literature presents several works and third-party applications focusing on musicological analyses of music genres. We mention the web page "*Every Noise AT Once*" [McDonald, 2013], which is a visualization tool for similarities between music genres, and the web platform by [Baratè and Ludovico, 2022], which investigates music genres labels. Other works investigated the correlation between the popularity index (with values in the range $[0, 100]$) provided by Spotify™ and other features such as audio [Sciandra and Spera, 2022], precomputed features [Araujo, 2020], and metadata, including graph-related metrics [Matsumoto et al., 2020].
Focusing on the correlation between commercial success and graph-related metrics of music genres, an important work is [Oliveira et al., 2020]. Here, the authors study an undirected weighted graph with music genres as nodes and arcs whose weight indicates the number of hit songs resulting from the collaboration between artists belonging to the two music genres connected by this arc. Their analysis consists of using Exploratory Factor Analysis to find latent variables correlated with graph-related metrics and DBSCAN clustering to identify different "collaboration profiles".
The most relevant work we discuss is [South et al., 2020], which investigates precisely the same raw data we are analysing in the next sections of this chapter and discussed in Section 6.2. The authors use this data to build the undirected

graph of Spotify™ music credits, with artists as nodes and collaborations as arcs. They analyzed the eigenvector centrality and observed a critical transition when removing the least popular artists from the graph. In fact, they showed that *classical* music attains the highest value of eigenvector centrality when the entire graph is considered. However, when removing the artists whose popularity is below a certain threshold, *hip-hop* artists outperform *classical* artists and become the most central in the graph. To formalize this phenomenon, they propose a *Social Group Centrality* (SGC) model, which aims at defining the dichotomy between two types of social influence. They define two types of social influence related to three types of nodes in the graph. One node can belong to the class of *community leaders*, *celebrities*, and *masses*. The latter is the class grouping all nodes with low popularity and a low number of connections. On one hand, *community leaders* are those nodes with high popularity which have a large number of connections to the *masses* nodes. On the other hand, *celebrities* are those nodes with high popularity which have a large number of connections to nodes with high popularity. In these two classes, *community leaders* and *celebrities*, the authors identified a dichotomy in the type of influence, therefore collaboration, between artists. They justify their hypothesis by presenting an analysis of *thresholded graphs*: considering the entire undirected graph, where each node has a popularity index, a *thresholded graph* is the subgraph resulting from the elimination of those nodes in the graph whose popularity is below a certain threshold. After analysing the eigenvector centrality of each music genre for different threshold values, the authors observed the presence of a critical transition when the threshold is set to 47. When considering threshold values below 47, *classical* music is the genre with the highest centrality value. However, when the threshold is equal to 47 or greater, *hip-hop* artists become the most central in the graph.

In the next sections, we present an analysis of several centrality measures and show that the SGC model does not adapt to other types of centralities. Therefore, we propose a different explanation for this phenomenon based on the analysis of the directed graph and the hypothesis that the type of connection between nodes, distinguishing between collaboration and citation, may provide a better interpretation. Before discussing the difference between the two explanations, ours and the one provided by [South et al., 2020], in Section 6.5, we introduce the Spotify™ data employed in this study along with the theoretical metrics investigated in our analysis.

6.2 The Spotify™ data

The analyses and contributions in the rest of this chapter work on the directed graph of music credits network built using data from Spotify™. As mentioned above, the raw data we use was collected between December 2017 and December 2018 by exploring the network via breadth-first search with Kanye West as a starting point for the collection due to being identified as one of the most popular artists in the network and very likely to be part of the largest connected component of the graph.

The dataset contains 1 250 114 artists. We build the directed graph, also called digraph, where nodes represent artists. In our digraph, there is an arc from node x to node y if there is a song in artist x's discography for which artist y is credited. An arc from node x to node y can be read as "artist x credits artist y for one of their songs". The total number of arcs in our digraph is 7 435 330. The Spotify™ data also comes with some useful metadata, although there are nodes in the data collection for which there is none or partial metadata. Metadata was collected for 625 061 artists, around 50%. We consider two metadata: music genre and popularity. Popularity is a value within the range $[0, 100]$ and it is computed based on the number, duration, and recency of streams of an artist's discography [Spotify, 2021]. In the dataset, 64 273 artists (around 10% of the artists with metadata) had a non-empty value for the music genre. Intuitively, an artist can be associated with one or more genres. The total number of different labels of music genre is 1 533. Although it may appear as an oversimplification, we decided to reduce this high number of genres since it would require more than a million pairs of genres to compare and because some genres have very few elements, which would affect negatively the statistical significance of the tests. Therefore, we decided to follow the definition of music genres as sets [Fabbri, 2004] and considered 16 super-genres (i.e. supersets). Because there is not a general consensus on the classification of genres into super-genres, we curated our own taxonomy, which was largely informed by AllMusic's genre classification [AllMusic, 2021] and supported by MusicMap [Crauwels, 2016], a "genealogy of popular music genres", and Every Noise At Once [McDonald, 2013], a data-driven map of music genres. We identified 16 super-genres: *classical*, *pop*, *hip-hop*, *rock*, *jazz*, *blues*, *soul-R'n'B*, *country*, *folk*, *easy-listening*, *avant-garde*, *electronic*, *latin*, *African*, *Asian*, *Caribbean*. As in the raw data, an artist can belong to multiple super-genres.

6.3 Theoretical definitions

Before delving into the contribution of this chapter, it is necessary to spend some time on some theoretical definitions relevant to the content of the next sections. We will use the following notation. We define G as a directed graph, or digraph, characterized by a set of nodes V and a set of arcs $E \subseteq V^2$. Let $N = |V|$ be the number of nodes in the directed graph. Without loss of generality, we consider the set V to be the set of integers in the range $[0, N]$. We denote with $A \in 2^{N \times N}$ the adjacency matrix of the graph, such that $A_{i,j} = \mathbb{I}\{(i, j) \in E\}$.

6.3.1 Reciprocity

We start by introducing the definition of reciprocity of a digraph, which is a metric that quantifies how frequently, if there is an arc from node i to node j, there is also the arc from j to i. In [Garlaschelli and Loffredo, 2004], the authors define reciprocity as the correlation coefficient between the entries in the adjacency matrix and the entries in its transpose, ignoring entries on the diagonal

$$\rho := \frac{\mathrm{Cov}\left[A_{i,j}, A_{j,i}\right]}{\mathrm{Var}\left[A_{i,j}\right]} \tag{6.1}$$

If the adjacency matrix is symmetrical, then $\rho = 1$. The digraph is perfectly reciprocal, and it could be represented as an undirected graph. If $A_{i,j} = 1 - A_{j,i}$ for $i \neq j$, then $\rho = -1$ and the digraph is unilaterally connected. If the covariance is 0, then $\rho = 0$ and arcs are reciprocated as often as they would if the same number of arcs was distributed at random in the graph.

6.3.2 Reachable Sets

In a digraph, the reachable set of a node is defined as the set of nodes that are reachable from that node, i.e. nodes at a finite distance from it. The co-reachable set of a node is the set of nodes from which that node is reachable. The co-reachable set of a node in a digraph is the reachable set of that node in the transposed graph. The co-reachable set of node i is:

$$\mathcal{K}_i := \{j \in V \mid i \neq j \wedge d(j,i) < +\infty\} \tag{6.2}$$

where $d(j,i)$ is the distance from node j to node i.

6.3.3 Centrality Metrics

A centrality metric indicates the importance of a node in a network. The Spotify™ digraph is large enough that computing some centrality metrics is intractable. Therefore, we are focusing mainly on geometric centralities, which can be approximated efficiently using HyperBall [Boldi and Vigna, 2013], and PageRank.

In-degree

One of the simplest and most intuitive measures for centrality is the *in-degree*, which is the number of incoming arcs of a node. The in-degree of node i is

$$c_i^{in} := \sum_{j=1}^{N} A_{j,i} \tag{6.3}$$

Closeness

Closeness centrality is based on the intuition that a node is more central the closer it is to all other nodes in the graph. The closeness of a node is defined as the reciprocal of the sum of the incoming distances from any other node.

$$c_i^{closeness} := \frac{1}{\sum_{j \in \mathcal{K}_i} d(j,i)} \tag{6.4}$$

The distances from non co-reachable nodes are ignored: their distance is infinite and the centrality would result to be zero. However, nodes with a small co-reachable set tend to have a high centrality value [Boldi and Vigna, 2014].

Lin Centrality

[Lin, 1976] introduced a modified version of closeness centrality, that is weighted by the square of the cardinality of the co-reachable set.

$$c_i^{lin} := \frac{|\mathcal{K}_i|^2}{\sum_{j\in\mathcal{K}_i} d(j,i)} \tag{6.5}$$

Harmonic Centrality

Harmonic centrality [Rochat, 2009] addresses the weaknesses of closeness, by taking the harmonic sum of the distances instead of the reciprocal of the sum.

$$c_i^{harmonic} := \sum_{j\in\mathcal{K}_i} \frac{1}{d(j,i)} \tag{6.6}$$

Harmonic centrality naturally ignores nodes outside the co-reachable set, because $\lim_{d\to\infty} 1/d = 0$.

PageRank

PageRank is a spectral measure of centrality. The vector of PageRank values for all nodes can be defined as the solution p to the following equation [Boldi and Vigna, 2014]

$$\begin{gathered} p = \alpha p\bar{A} + (1-\alpha)v \\ p \in [0,1]^N \mid \|p\|_1 = 1 \end{gathered} \tag{6.7}$$

The PageRank of a node can be interpreted as the probability distribution of ending a random walk on that node.

6.4 Analysis of several centrality measures on the Spotify™ music credits network

The definitions presented in the previous section are necessary to comprehend the analysis we conducted on the Spotify™ music credits network, which is presented in this section. We considered the directed graph where the nodes in the graph represent artists and where an arc from node x to node y denotes that artist y has been credited for a song in artist x's discography. We used WebGraph [Boldi and Vigna, 2004a,b], a Java library for compression and analysis of very large graph to perform the analyses and JPype [Nelson et al., 2020] to interface WebGraph with Python, used for data visualization purposes.
We analysed the distribution of several centrality measures for artists belonging to different music genres. We considered the 16 super-genres and computed the distribution of centrality values of a genre examining the artists belonging to

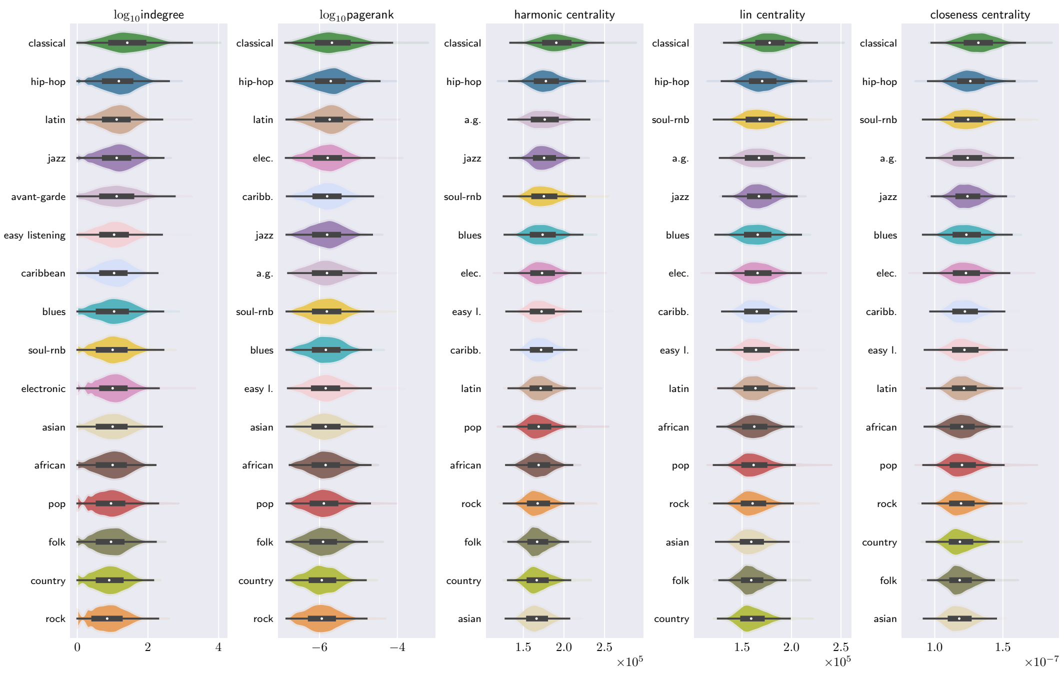

FIGURE 6.1: This figure shows the violin plots of the distribution of different centrality measures conditioned on the musical genre. Genres are sorted in decreasing order of median value for each centrality. In-degree and PageRank are shown on a logarithmic scale.

that genre. We performed this analysis for all the five centrality measures defined in the previous section: in-degree, closeness, Lin centrality, harmonic centrality, and PageRank. The results are summarized in Figure 6.1. It is evident how for every centrality the top two genres with the highest median values are always *classical* and *hip-hop*, as previously found in the undirected graph [South et al., 2020]. Specifically, we noticed that the mean value for *classical* artists is significantly higher than all other genres. For what concerns *hip-hop* artists, their mean value is second for Lin centrality and closeness, while for in-degree, harmonic centrality and PageRank the gap between their mean value and the one in the third place is inconclusive, although it is still greater than all other 13 genres. For all the other genres, their position varies depending on the centrality considered. We assessed the significance of the differences between mean values using a Bayesian Student-T test [Kruschke, 2013], implemented in PyMC3 [Salvatier et al., 2016]. We defined the ROPE as an effect size between -0.1 and $+0.1$ (a "very small" effect size [Cohen, 1988]) and set the significance threshold at $\alpha = 0.05$.

6.5 The importance of a directed graph

After analysing the distribution of different centralities conditioned on the music genre, we considered the concept of *thresholded graph* mentioned in the previous sections and proposed by [South et al., 2020]. Indeed, one of the relevant

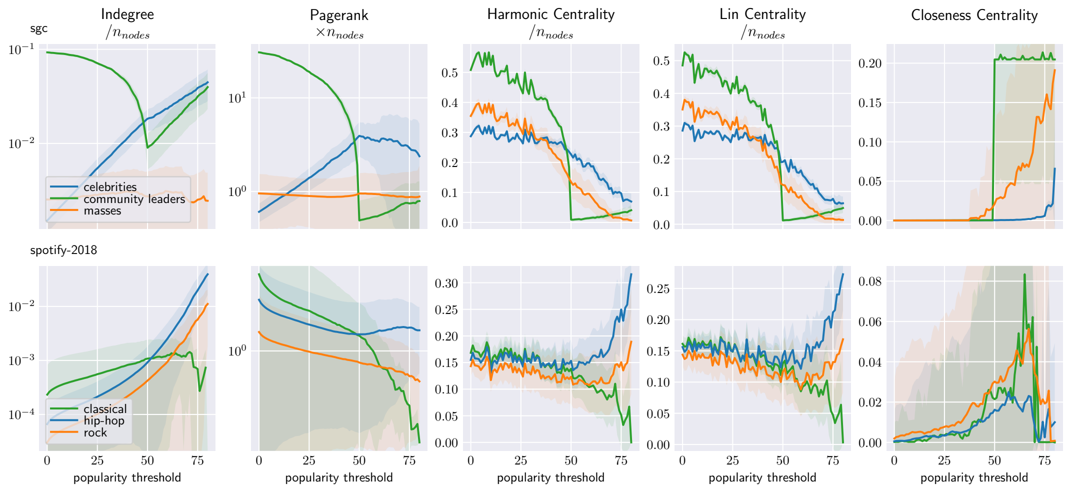

Figure 6.2: Transitions in node centralities under thresholding in the Spotify™ graph and in an SGC graph. On the x-axis are the popularity threshold values. Centrality values are normalized to remove trends naturally arising from changing the number of nodes in the graph (in-degree, harmonic centrality, and Lin centrality are divided by the number of nodes, PageRank is multiplied by the number of nodes). Solid lines are the average centrality values for nodes of one musical genre (or SGC class). Filled areas are between the average plus and minus 0.67 times the standard deviation (50% HDI for a normal distribution). Indegree and PageRank are shown on a logarithmic scale.

contributions of [South et al., 2020] was to show that when removing nodes from the undirected graph based on a popularity threshold greater or equal to 47 there is a transition from *classical* to *hip-hop* artists as the most central in the graph. Therefore, as a natural consequence, our following objective is to study the thresholded graphs for the centralities considered in our analysis. We implemented the SGC model in NetworkX [Hagberg et al., 2008] and compared the centrality transitions in a graph sampled from the SGC model with our Spotify™ digraph. We considered different thresholds and computed the average centrality values of the three classes of the SGC model and of three super-genres (*classical*, *hip-hop*, *rock*) in the Spotify™ digraph. Results are shown in Figure 6.2. One can observe that the SGC model graph and the Spotify™ digraph have different behaviours. Indeed, while the SGC graph shows a transition around a threshold of $47-50$, this does not happen in the Spotify™ digraph.
As a further investigation, we studied the distribution of in-degrees and out-degrees of the nodes in the Spotify™ digraph. The in-degree of a node is the number of incoming arcs and the out-degree is the number of outgoing arcs. In the Spotify™ digraph, this means that the in-degree of a node x indicates how many artists have credited artist x for one of their songs, while the out-degree of a node x denotes the number of times artist x credits another artist for one of their songs. We can observe how the in-degree and out-degree have very different scales, where the outdegree scales from 0 to the order of hundreds,

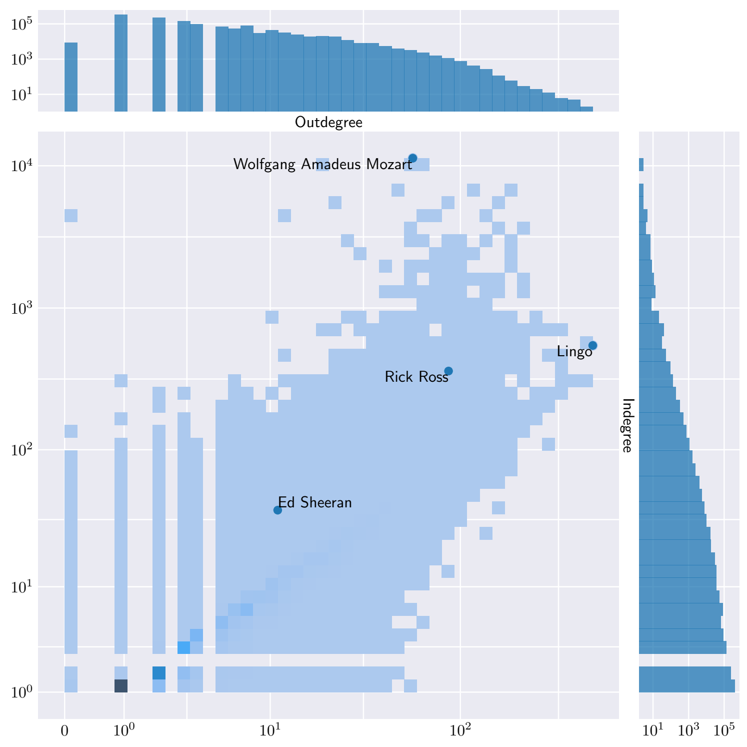

Figure 6.3: Distribution of in-degrees and out-degrees in the graph on a logarithmic scale. Joint distribution is shown as a heatmap (darker colours signify higher frequencies). The degrees of some reference artists are shown: Wolfgang Amadeus Mozart (highest in-degree), Lingo (highest out-degree), Ed Sheeran (highest popularity), and Rick Ross (one of the most central artists in the thresholded graphs).

while the in-degree scales from 0 to the order of ten thousand, which shows that there are significant asymmetries in the collaborations. The artist with the highest in-degree appears to be Mozart, with an in-degree of 11 301 and an out-degree of 57. This means that Mozart has been credited by 11 301 other artists but, in his discography, only 57 other artists are credited. On the other hand, the artist with the highest out-degree is Lingo, a DJ who produces remixes sampling from various other artists' discographies. From these results, it seems evident how crucial it is to consider the directed graph, as these asymmetries may be relevant to the phenomena observed in the analysis of centralities.

6.6 A new node-wise index of reciprocity

As a result of the observations presented above, we argue that what distinguishes *community leaders* and *celebrities* in the Spotify™ graph is that the former are highly cited while the latter are truly collaborative. To identify this dichotomy, we propose a node-wise index of reciprocity to distinguish between *citations* and *collaborations*.

In Section 6.3.1 we presented the reciprocity metric of the entire graph. However, we are interested in investigating reciprocity between nodes and therefore how much each node reciprocates arcs. [Cheng et al., 2011] define the reciprocity over each pair of arcs. They propose this metric which defines an arc as reciprocated if the arc between the same nodes, but in the opposite direction, exists. However, in our data, the direction of an arc is not entirely reliable. For example, the out-degree of Mozart is 57, but that does not mean that Mozart features 57 other artists in his own work. If one looks at these arcs, it is possible to see that the outgoing arcs are mainly orchestras that performed Mozart's music. [Wardil and Hauert, 2014] propose two indices to quantify a node's reciprocity: altruism and activity, where the first accounts for the balance between incoming and outgoing arcs and the second stands for the normalized number of arcs in either direction. However, having two indices is not practical. Therefore, we propose a novel node-wise index of reciprocity defined as Pearson's correlation coefficient between the entries in the adjacency matrix corresponding to incoming and outgoing arcs, which are the entries on the node's column and row, respectively. It takes after the graph reciprocity index introduced by [Garlaschelli and Loffredo, 2004]

$$\rho_i := \frac{\mathrm{Cov}\left[A_{i,j}, A_{j,i}\right]}{\sqrt{\mathrm{Var}\left[A_{i,j}\right]\mathrm{Var}\left[A_{j,i}\right]}}. \tag{6.8}$$

Empirically, it can be computed as

$$\widehat{\rho}_i = \frac{\overleftrightarrow{a}_i - \overrightarrow{a}_i \overleftarrow{a}_i}{\sqrt{(1-\overrightarrow{a}_i)\overrightarrow{a}_i(1-\overleftarrow{a}_i)\overleftarrow{a}_i}} \tag{6.9}$$

where $\overrightarrow{a}$ is the normalized out-degree, $\overleftarrow{a}$ is the normalized in-degree, and $\overleftrightarrow{a}_i$ is the normalized number of reciprocated arcs.

$$\overrightarrow{a}_i := \frac{1}{N}\sum_{j=1}^{N} A_{i,j} \tag{6.10}$$

$$\overleftarrow{a}_i := \frac{1}{N}\sum_{j=1}^{N} A_{j,i} \tag{6.11}$$

$$\overleftrightarrow{a}_i := \frac{1}{N}\sum_{j=1}^{N} A_{i,j} \cdot A_{j,i}. \tag{6.12}$$

In our graph, reciprocity is undefined for nodes with no outgoing arcs: in this case, we define the reciprocity to be zero as no edge is reciprocated. It would be undefined for nodes with no incoming arcs, too, but there is no such node in our digraph, because of the data collection policy.
We present the violin plots of the distribution of reciprocity conditioned on the music genre in Figure 6.4 and summarize the reciprocity values in Table 6.1. As an interesting result, it is possible to observe that the music genre with the highest median value of reciprocity is African followed by Asian, two music genres defined by their geographical origin. This shows that the artists belonging to these two genres are highly collaborative, favoured by their geographical proximity. It would be interesting to further investigate the possible causes of

Genre	Q1	MED	Q3	IQR
african	0.739	0.889	1.000	0.261
asian	0.655	0.866	1.000	0.345
rock	0.655	0.866	1.000	0.345
avant-garde	0.615	0.866	0.970	0.355
folk	0.577	0.845	1.000	0.423
hip-hop	0.674	0.840	0.941	0.267
pop	0.623	0.833	1.000	0.377
latin	0.598	0.816	0.943	0.345
caribbean	0.577	0.816	0.935	0.358
jazz	0.539	0.804	0.939	0.400
soul-rnb	0.552	0.791	0.935	0.384
country	0.488	0.783	0.957	0.469
electronic	0.577	0.775	0.926	0.348
easy listening	0.479	0.747	0.926	0.447
classical	0.500	0.693	0.866	0.366
blues	0.365	0.655	0.913	0.548
Overall	0.583	0.816	0.949	0.365

TABLE 6.1: Summary of reciprocity values by super-genre: first quartile, median, third quartile and interquartile range.

these high reciprocity values on an ethnomusicological level.
Our main observation is that the placement of *classical* and *hip-hop* artists. The median reciprocity of *classical* music is shown to be the lowest among all the 16 super-genres, preceded by *blues*. The difference between the median values of *classical* and *hip-hop* artists is significant and practically relevant, as the *hip-hop* has a higher reciprocity which ranks them as the sixth highest median reciprocity. This shows how artists in *classical* music do not reciprocate much, while *hip-hop* artists tend to collaborate more. This confirms our hypothesis that there is a preference for *citations* in *classical* music, while *hip-hop* artists prefer *collaborations*.
To give a qualitative insight, we sorted all the nodes by reciprocity and considered the ones with the highest popularity to find some examples that might be familiar to many people. Amongst the least reciprocating nodes, we can find many well-known artists. The top 10 artists in this sorting are: *Lil Pump*, *Green Day*, *Jorge & Mateus*, *Wham!*, *Oasis*, *Muse*, *Pearl Jam*, *Bruce Springsteen*, *Journey*, and *The Beach Boys*. We believe that this may be due to the high number of cover songs that other artists published. On the other side, the top 10 most reciprocating nodes are less popular, both quantitatively and qualitatively. We report some exceptions: *Julian Casablancas* in tenth position, and *Guns N' Roses* in first position. The neighbours of *Guns N' Roses* are eight: five members of the band, two orchestras that recorded some of their songs, and one musician affiliated with one of the orchestras. What is unusual, in their case, is the fact that no unreciprocated covers had been uploaded at the time of the data collection by artists reached by the breadth-first-search

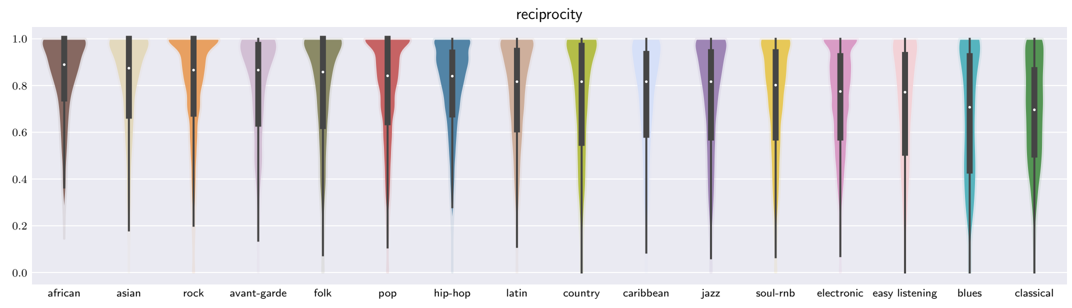

FIGURE 6.4: Violin-plot of the distribution of reciprocity values conditioned on music genre. Genres are sorted in decreasing order of median value. Note that the top two genres (African and Asian) have a strong geographical characterization. Classical music artists have the lowest median reciprocity; this shows the tendency, in that genre, to perform compositions by other authors.

crawler. This could be due to copyrights on the platform at the moment of the data collection and it would probably be different if we repeated the experiment with updated data, or if we had the entire Spotify™ database.

6.7 Conclusions

In this chapter, we presented our contribution to a different line of research, which focused on the analysis of the Spotify™ music credits network. We studied the directed graph and several centrality measures, which led to the observation that *classical* and *hip-hop* are the most central genres in the graph. This result was previously supported in [South et al., 2020], however our explanation deviates from theirs. In fact, we believe that the analysis of the directed graph instead of the undirected one can better explain the phenomena observed in the thresholded graphs, where the most central genre switches from *classical* to *hip-hop*. We argue that this dichotomy is due to two different types of connections between artists: *citations* and *collaborations*. We propose a node-wise index of reciprocity which indicates how much a node is being reciprocated by others. Using this index and observing the in-degree and out-degree of the nodes in the graph, it is possible to notice how *classical* artists are characterized by a high number of citations and therefore a low reciprocity, while *hip-hop* artists by actual collaborations and consequentially by a high reciprocity. This is also explained by many artists and orchestras citing and sampling from works of *classical* artists such as Bach, Mozart, and Beethoven. On the other hand, *hip-hop* artists tend to collaborate more, with many featuring in their discography. To quantify how much an artist is inclined to collaborate, one can employ the proposed reciprocity index. This index could also find applications in recommender systems. For example, it can be used as a feature to improve the automatic compilation of playlists. If an artist belongs to a community of collaborating musicians, it is reasonable to think that collaborating artists might share some commonalities in their music style. So, if a user listens to an artist in this community, it is reasonable to believe that the user might also be interested in the works of the collaborating musicians. Therefore, reciprocity could be used to filter out false collaborators and identify those for which the user could sympathize. By using this index, a recommender system may provide more informed recommendations and support human decision-making. In the future directions presented in Chapter 7, we elaborate more on the potential employment of this index in recommender systems.

Chapter 7

Conclusions

This thesis studies two different lines of research related by their application to recommender systems and music streaming platforms. In Chapter 1, we discussed the importance of recommender systems caused by the development of the World Wide Web and the growing offer of online services, particularly streaming services and their catalogues. With an increasing number of products being offered to the user, streaming services had to perform some sort of personalization to improve the user's experience and to efficiently profit from the wide range of offered products. We decided to focus on recommender systems, focusing specifically on song recommendation in music streaming services. This choice is due to the interest in studying complex dynamics typical of these settings, where the user's preferences constantly change with time or with the user's past actions and their history of interactions with items. Here, the focal problem usually faced in recommender systems is not to find the best product to recommend, but to find a specific set of items and a specific order for this set which could satisfy the user's experience on the platform.
After presenting some basic concepts about multiarmed bandits in Chapter 2 to better comprehend the topics of this dissertation, we identified in the class of non-stationary multiarmed bandits an efficient tool to answer the problems which typically arise in song recommendation. Indeed, non-stationary bandits are proven to be an efficient method to deal with recommendation problems in settings where the user's behaviour is not stationary. We discussed the current state of the art in Chapter 3. In this chapter, we made a distinction between exogenous and endogenous non-stationarity. The first class groups all those MAB models where the non-stationarity is intrinsic to the environment, it depends on time and it is not affected by the user's actions. In this family of exogenous non-stationarity, it is possible to identify different categories depending on the assumptions: piece-wise stationary MAB, MAB with variation budget, and restless MAB. On the other hand, we can also identify a family of bandits where the non-stationarity directly depends on the user's past actions. This adds a layer of complexity since it is assumed that the user's choice will influence future rewards. Inside this class of bandits with endogenous non-stationarity, we can point to rested bandits and state-dependent bandits, where each model offers different assumptions.
Studying the family of endogenous bandits, we identified a weakness that could have been improved. Indeed, there was a lack of a model able to generalize to

different types of endogenous non-stationarity, specifically satiation and seasonality, encapsulating these phenomena in a single framework where the reward function is arm-dependent and the bandit problem is addressed by a single algorithm. To overcome this, we proposed a new non-stationary bandit model for finite sets of actions where the reward of an arm depends on its state, which is designated by the last time the arm took part in a switch of actions. We designed this model which can represent different types of non-stationary trends, without the need to restrict to monotonic reward functions. Using this concept of switch of actions, the bandit can model all types of trends in the non-stationarity as long as the reward function is bounded in $[0, 1]$ and non-decreasing in $\mathbb{Z}^-$. We called this model Last Switch Dependent bandit and studied its hardness as well as the approximation and estimation errors. We provided an algorithm to solve this problem by adapting the CombUCB1 algorithm for combinatorial semi-bandits by [Kveton et al., 2015] and presented an experimental evaluation of our model against natural baselines. With this model, we can address both satiation and seasonality in a single framework, with arm-dependent reward functions. However, a limitation of this work is the fact that the actions set considered is finite, which is not always representative of practical applications. After presenting our contribution to the class of finitely many arms MABs, we extended our contribution to the linear setting, where the reward of an action is the result of a linear function between the action and a hidden parameter. The aim was to provide a non-stationary bandit model able to generalize to different types of non-stationarity and simultaneously deal with an infinite set of actions. Addressing this setting properly is much more complex due to the presence of interferences between arms. Indeed, while in a finite set of actions playing one arm only influences the reward of that specific arm, the linear setting and its infinite set of actions cause cross-arm dependencies so that playing any action may influence any other action in the set. To tackle these problems, we proposed a non-stationary linear bandit called Linear Bandit with Memory, where the reward of an action at time t depends on the actions played in the past in a window, called *memory matrix*, of size m and elevated to the exponent γ. We analysed this setting, studying the approximation and estimation errors as well as proposing an algorithm to solve the bandit problem, along with experimental validation. We notice that the modelling choices of LBM allow the representation of monotonic, either increasing or decreasing, functions able to partially recover rested rotting and rising bandits in a single framework. However, this model relies on the knowledge of the parameters m and γ, which might be unrealistic in practice. Therefore, we provide a solution to the case where m and γ are unknown. In this setting, we propose an adaptation of Bandit Combiner on LBM, a meta-bandit algorithm which simultaneously solves the LBM problem while learning the two parameters. Moreover, we prove that the results obtained in this section hold also for the case where the form of the memory matrix is not defined by m and γ but can take a general form. We state Remark 5.5 where we explain how the results adapt in this case.
After proposing online learning models for tackling the problems arising in song recommendation, we also focused on the analysis of the Spotify™ music credits network. In Chapter 6, we studied the Spotify™ data as a directed graph of the

music credits network between artists. We analysed the centrality distributions of music genres and observed two particular behaviours in *classical* and *hip-hop* artists, who appear to be the most central in the graph. We analysed the *thresholded graphs* for several centrality measures and the in-degree and out-degree distributions of the nodes. We noticed an asymmetry in the collaboration patterns and proposed a new index of reciprocity to identify two different types of connections, distinguishing between *citations* and *collaborations*. Using this index, we were able to explain the centrality values of *classical* and *hip-hop* artists, where the centrality of the first genre is motivated by a high number of citations while the second by a high number of collaborations. Therefore, we show how reciprocity can discriminate between these two different types of crediting patterns between artists.

7.1 Future developments

Reflecting on the results discussed in this thesis, we end this dissertation by discussing some future directions. Concerning the broader theoretical research on non-stationary bandits, a possible new research could focus on different types of non-stationarity, focusing on not only relative but also absolute seasonality, to expand on the type of users' behaviours which could be considered.
A different research line would be to rephrase the modalities of interactions between the learner and the bandit. In the models discussed in this research, we focused on bandits whose goal was to suggest a song to the user, who would listen and rate this song. A possible idea is to formalize a new setting where the bandit identifies not a single arm but a subset of $n \in \mathbb{N}^{>1}$ arms to offer the user the possibility of choosing the action to play among these. This could be done by identifying the best n actions relying on a UCB index to assess their performance. However, a major problem would be to efficiently exploit the user's feedback with respect to the n items in the subset. Indeed, one could think of a way of exploiting the user's interactions with the items in this subset to infer some information about his preferences. One could also study this problem by formalizing a hierarchy of arms. In the examples illustrated in this thesis, we considered bandits for music recommender systems with music genres as arms. However, in song recommendation the system needs to pick a specific song. Therefore, one could think of formalizing the problem as a bandit with a hierarchy of arms built on two layers: the arms at a higher level representing music genres and the arms at a lower level representing actual songs. The bandit would need to identify first the music genre and then a specific song belonging to that genre. After the item has been suggested, the user's feedback could be used to update the information not only on the specific song but also about the music genre it belongs to. In this setting, the optimal trajectory is very complex, as it consists of the best song belonging to the best music genre at time t. Therefore, the learner must first identify the best genre and then the best song in this tree of arms. This would make the problem difficult as it would result in a higher probability of choosing a suboptimal arm and therefore narrowing the upper bound on the regret would be more complicated.

Concerning the second line of research, a future direction would be to use the reciprocity index as an additional feature in music recommender systems. In these settings, where finding music similar to the user's interests is essential for providing tailored suggestions, identifying true collaborations and therefore a community of artists who cooperate may be a helpful tool in making informed recommendations. This could be useful in automatic playlist continuation. Another possibility would be to study which genres collaborate more within the genre and outside of it. An index measuring this behaviour could be used to foster different levels of exploration depending on the user's preferences.
One concrete example would be to focus on engineering a solution which exploits the bandit models proposed in this manuscript and combines them with the reciprocity index. This could be done using non-stationary bandit models and integrating them into a multitask bandit problem. By incorporating this index in the process of recommending a new song, the system would be able to exploit additional information to identify the potential song candidate to suggest to the user. One practical example would be the formalization of a bandit where reciprocity is used to improve the exploration. Another idea is to also exploit the Spotify credits network to better identify the collaborating artists that could potentially be enjoyed by the user. All these suggestions could be integrated into a single framework to improve once again the user's personalized experience within a music streaming platform.

Appendix A

Appendix

A.1 Proof of the regret bound of OFUL

Here we are presenting the proof of the regret of the `OFUL` algorithm in [Abbasi-Yadkori et al., 2011, Appendix C, Proof of Theorem 3]. It is assumed that $\|\theta^*\|_2 \leq S$, $\|a_t\|_2 \leq L$, and $V = I\lambda$ with $\lambda > 0$. As for many regret analysis in the bandit literature, the first step is to decompose the instantaneous regret.

$$\begin{aligned} X_t &= \langle a^*, \theta^* \rangle - \langle a_t, \theta^* \rangle \\ &\leq \langle a_t, \widetilde{\theta}_t \rangle - \langle a_t, \theta^* \rangle \end{aligned}$$

where the last step is because $\langle a_t, \widetilde{\theta}_t \rangle$ is optimistic

$$\begin{aligned} &= \langle a_t, \widetilde{\theta}_t - \theta^* \rangle \\ &= \langle a_t, \widehat{\theta}_{t-1} - \theta^* \rangle + \langle a_t, \widetilde{\theta}_t - \widehat{\theta}_{t-1} \rangle \\ &= \|\widehat{\theta}_{t-1} - \theta^*\|_{\bar{V}_{t-1}^{-1}} \|a_t\|_{\bar{V}_{t-1}^{-1}} + \|\widetilde{\theta}_t - \widehat{\theta}_{t-1}\|_{\bar{V}_{t-1}^{-1}} \|a_t\|_{\bar{V}_{t-1}^{-1}} \\ &\leq 2\sqrt{\beta_{t-1}(\delta)} \|a_t\|_{V_t^{-1}} \end{aligned}$$

where the last step is obtained using Cauchy-Schwarz and knowing that $\sqrt{\beta_{t-1}(\delta)}$ is bounding the difference between the estimation of theta in the confidence set and its true value. Since the rewards are bounded in $[0, 1]$, we know $X_t \leq 2$. So we can write

$$\begin{aligned} X_t &\leq 2 \min(\sqrt{\beta_{t-1}(\delta)} \|a_t\|^2_{\bar{V}_{t-1}^{-1}}, 1) \\ &\leq 2\sqrt{\beta_{t-1}(\delta)} \min(\|a_t\|^2_{\bar{V}_{t-1}^{-1}}, 1). \end{aligned}$$

Therefore, we can write that with probability $1-\delta$ for all $T \geq 0$, OFUL achieves a regret upper bounded by

$$\begin{aligned} R_T \leq \sqrt{T \sum_{t=1}^{T} X_t^2} \leq \sqrt{8\beta_T(\delta) T \sum_{t=1}^{T} \min(\|a_t\|_{V_t^{-1}}, 1)} \leq 4\sqrt{\beta_T(\delta) T \log(det(V_T))} \\ \leq 4\sqrt{Td \log(\lambda + TL/d)}(\lambda^{1/2} S + R\sqrt{2\log(1/\delta) + d\log(1 + TL/(\lambda d))}) \end{aligned}$$

where the last step comes [Abbasi-Yadkori et al., 2011, Lemma 11], where the authors bound $\sum_{t=1}^{T} \|a_t\|^2_{\bar{V}_{t-1}^{-1}} \leq 2 \log \frac{\det(\bar{V}_T)}{det(V)}$, and [Abbasi-Yadkori et al., 2011, Theorem 2], where they define the confidence set as $C_t = \{\theta \in \mathbb{R}^d : \|\widehat{\theta}_t - \theta\|_{\bar{V}_t} \leq R\sqrt{d \log\left(\frac{1+tL^2/\lambda}{\delta}\right)} + \lambda^{1/2} S\}$.

A.2 Proof of the regret bound of CombUCB1

In this section of the Appendix, we present the proof of the `CombUCB1` algorithm from [Kveton et al., 2015]. The regret analysis relies on the decomposition between two events: when the suboptimality gaps of the arms are larger than ε and at most ε.
In [Kveton et al., 2015, Lemma 1], the authors state that

$$\mathcal{F}_t = \{\Delta_{\boldsymbol{a}_t} \leq 2 \sum_{a_i \in \widetilde{\boldsymbol{a}}_t} c_{T, T_{t-1}(a_i)}, \Delta_{\boldsymbol{a}_t} > 0\}$$

, where $\widetilde{\boldsymbol{a}}_t = \boldsymbol{a}_t\ \boldsymbol{a}^*$, and prove that $R_T \leq \mathbb{E}[\widehat{R}_T] + \left(\frac{\pi^2}{3}+1\right) KL$, where $\left(\frac{\pi^2}{3}+1\right) KL$ bounds the initialization of the algorithm from time step 0 to t_0. Thus, $\widehat{R}_T = \sum_{t=t_0}^{T} \Delta_{\boldsymbol{a}_t} \mathbb{I}\{\mathcal{F}_t\}$. The authors partition $\widehat{R}_T$ as

$$\begin{aligned} \widehat{R}_T &= \sum_{t=t_0}^{T} \Delta_{\boldsymbol{a}_t} \mathbb{I}\{\mathcal{F}_t, \Delta_{\boldsymbol{a}_t} < \varepsilon\} + \sum_{t=t_0}^{T} \Delta_{\boldsymbol{a}_t} \mathbb{I}\{\mathcal{F}_t, \Delta_{\boldsymbol{a}_t} \geq \varepsilon\} \\ &\leq \varepsilon T + \sum_{t=t_0}^{T} \Delta_{\boldsymbol{a}_t} \mathbb{I}\{\mathcal{F}_t, \Delta_{\boldsymbol{a}_t} \geq \varepsilon\}. \end{aligned}$$

The second term is then bounded as:

$$\sum_{t=t_0}^{T} \Delta_{\boldsymbol{a}_t} \mathbb{I}\{\mathcal{F}_t, \Delta_{\boldsymbol{a}_t} \geq \varepsilon\} \leq \sum_{a_i \in \mathcal{A}} K \frac{534}{\varepsilon} \log(n) \leq KL \frac{534}{\varepsilon} \log(T).$$

Therefore, it is possible to conclude the analysis by writing

$$R_T \leq \frac{534KL}{\varepsilon} \log(T) + \varepsilon T + \left(\frac{\pi^2}{3} + 1\right) KL, \tag{A.1}$$

and by choosing $\varepsilon = \sqrt{\frac{534KL\log(T)}{T}}$:

$$\begin{aligned} R_T &\leq 2\sqrt{534KLT\log(T)} + \left(\frac{\pi^2}{3} + 1\right) KL \\ &< 47\sqrt{KLT\log(T) + \left(\frac{\pi^2}{3} + 1\right) KL}. \end{aligned}$$

Adapting the result using the terms in Chapter 3, where the cardinality of the ground set is $L = K$ and the maximum number of chosen items is $K = m$, the result becomes $R_T < 47\sqrt{KmT\log(T) + \left(\frac{\pi^2}{3}+1\right)mK}$.

A.3 Proof of Proposition 5.4

We prove the (stronger) high probability version of Proposition 5.4.

Proposition 5.5. *Let $\lambda \geq 1$, $\delta \in (0,1)$, and $\boldsymbol{a}_\tau$ be the blocks of actions in $\mathbb{R}^{d(m+L)}$ associated to the $\boldsymbol{b}_\tau$ defined in (5.13). Then, with probability at least $1-\delta$ we have*

$$\sum_{\tau=1}^{T/(m+L)} \widetilde{r}(\widetilde{\boldsymbol{a}}) - \widetilde{r}(\boldsymbol{a}_\tau) \leq 4L(m+1)^{\gamma^+}\sqrt{Td\,\ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d(m+L)\lambda}\right)}$$
$$\cdot\left(\sqrt{\lambda}L + \sqrt{\ln\left(\frac{1}{\delta}\right) + d(m+L)\,\ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d(m+L)\lambda}\right)}\right).$$

Proof. The proof essentially follows that of [Abbasi-Yadkori et al., 2011, Theorem 3]. The main difference is that our version of OFUL operates at the block level. This implies a smaller time horizon, but also and increased dimension and an instantaneous regret $\langle\widetilde{\boldsymbol{b}},\boldsymbol{\theta}^*\rangle - \langle\boldsymbol{b}_\tau,\boldsymbol{\theta}^*\rangle$ upper bounded by $2L(m+1)^{\gamma^+}$ instead of 1. We detail the main steps of the proof for completeness. Recall that running OFUL in our case amounts to compute at every block time step τ

$$\widehat{\boldsymbol{\theta}}_\tau = \boldsymbol{V}_\tau^{-1}\left(\sum_{\tau'=1}^{\tau}\boldsymbol{y}_{\tau'}\,\boldsymbol{b}_{\tau'}\right),$$

where

$$\boldsymbol{V}_\tau = \sum_{\tau'=1}^{\tau}\boldsymbol{b}_{\tau'}\boldsymbol{b}_{\tau'}^\top + \lambda I_{d(m+L)}\,, \qquad \text{and} \qquad \boldsymbol{y}_\tau = \sum_{i=m+1}^{m+L} y_{\tau,i}\,,$$

since we associate with a block of actions the sum of rewards obtained after time step m. Note that by the determinant-trace inequality, see e.g., [Abbasi-Yadkori et al., 2011, Lemma 10], with actions $\boldsymbol{b}_\tau$ that satisfy $\|\boldsymbol{b}_\tau\|_2^2 \leq m + L(m+1)^{2\gamma^+}$ we have

$$\frac{|\boldsymbol{V}_\tau|}{|\lambda I_{d(m+L)}|} \leq \left(1+\frac{\tau(m+L(m+1)^{2\gamma^+})}{d(m+L)\lambda}\right)^{d(m+L)} \leq \left(1+\frac{\tau(m+1)^{2\gamma^+}}{d\lambda}\right)^{d(m+L)}. \tag{A.2}$$

The action played at block time step τ is the block $\boldsymbol{a}_\tau \in \mathcal{B}_d^{m+L}$ associated with

$$\boldsymbol{b}_\tau = \underset{\boldsymbol{b}\in\boldsymbol{\mathcal{B}}}{\operatorname{argmax}}\ \sup_{\boldsymbol{\theta}\in\boldsymbol{\mathcal{C}}_{\tau-1}}\langle\boldsymbol{b},\boldsymbol{\theta}\rangle\,, \tag{A.3}$$

where

$$\boldsymbol{\mathcal{C}}_\tau = \left\{\boldsymbol{\theta}\in\mathbb{R}^{d(m+L)} \colon \|\widehat{\boldsymbol{\theta}}_\tau - \boldsymbol{\theta}\|_{\boldsymbol{V}_\tau} \leq \boldsymbol{\beta}_\tau(\delta)\right\},$$

with

$$\boldsymbol{\beta}_\tau(\delta) = \sqrt{2\ln\left(\frac{1}{\delta}\right) + d(m+L)\,\ln\left(1 + \frac{\tau(m+1)^{2\gamma^+}}{d\lambda}\right)} + \sqrt{\lambda L}\,. \tag{A.4}$$

Applying [Abbasi-Yadkori et al., 2011, Theorem 2] to $\boldsymbol{\theta}^* \in \mathbb{R}^{d(m+L)}$ which satisfies $\|\boldsymbol{\theta}^*\|_2 \leq \sqrt{L}$ we have that $\boldsymbol{\theta}^* \in \boldsymbol{\mathcal{C}}_\tau$ for every τ with probability at least $1-\delta$. Denoting by $\widetilde{\boldsymbol{\theta}}_\tau$ the model that maximizes (A.3), we thus have that with probability at least $1-\delta$, the inequality $\langle \widetilde{\boldsymbol{b}}, \boldsymbol{\theta}^* \rangle \leq \langle \boldsymbol{b}_\tau, \widetilde{\boldsymbol{\theta}}_\tau \rangle$ holds for every τ, and consequently

$$\begin{aligned}
&\sum_{\tau=1}^{T/(m+L)} \langle \widetilde{\boldsymbol{b}}, \boldsymbol{\theta}^* \rangle - \langle \boldsymbol{b}_\tau, \boldsymbol{\theta}^* \rangle \\
&\qquad \leq \sum_{\tau=1}^{T/(m+L)} \min\left\{2L(m+1)^{\gamma^+}, \langle \boldsymbol{b}_\tau, \widetilde{\boldsymbol{\theta}}_\tau - \boldsymbol{\theta}^* \rangle\right\} \\
&\qquad \leq \sum_{\tau=1}^{T/(m+L)} \min\left\{2L(m+1)^{\gamma^+}, \left\|\widetilde{\boldsymbol{\theta}}_\tau - \boldsymbol{\theta}^*\right\|_{\boldsymbol{V}_{\tau-1}} \|\boldsymbol{b}_\tau\|_{\boldsymbol{V}_{\tau-1}^{-1}}\right\} \\
&\qquad \leq \sum_{\tau=1}^{T/(m+L)} \min\left\{2L(m+1)^{\gamma^+}, 2\boldsymbol{\beta}_\tau(\delta)\, \|\boldsymbol{b}_\tau\|_{\boldsymbol{V}_{\tau-1}^{-1}}\right\} \\
&\qquad \leq 2L(m+1)^{\gamma^+}\, \boldsymbol{\beta}_{T/(m+L)}(\delta) \sum_{\tau=1}^{T/(m+L)} \min\left\{1, \|\boldsymbol{b}_\tau\|_{\boldsymbol{V}_{\tau-1}^{-1}}\right\} \\
&\qquad \leq 2L(m+1)^{\gamma^+}\, \boldsymbol{\beta}_{T/(m+L)}(\delta) \sqrt{\frac{T}{m+L} \sum_{\tau=1}^{T/(m+L)} \min\left\{1, \|\boldsymbol{b}_\tau\|^2_{\boldsymbol{V}_{\tau-1}^{-1}}\right\}} \\
&\qquad \leq 2\sqrt{2}L(m+1)^{\gamma^+}\, \boldsymbol{\beta}_{T/(m+L)}(\delta) \sqrt{\frac{T}{m+L}\, \ln\frac{|\boldsymbol{V}_{T/(m+L)}|}{|\lambda I_{d(m+L)}|}} \\
&\qquad \leq 4L(m+1)^{\gamma^+} \sqrt{Td\,\ln\left(1 + \frac{T(m+1)^{2\gamma^+}}{d(m+L)\lambda}\right)} \\
&\qquad\qquad \cdot \left(\sqrt{\lambda L} + \sqrt{\ln\left(\frac{1}{\delta}\right) + d(m+L)\,\ln\left(1 + \frac{T(m+1)^{2\gamma^+}}{d(m+L)\lambda}\right)}\right),
\end{aligned}$$

where we have used [Abbasi-Yadkori et al., 2011, Lemma 11], as well as (A.2) and (A.4). Note that in the stationary case, i.e., when $m = 0$ and $L = 1$, we exactly recover [Abbasi-Yadkori et al., 2011, Theorem 3]. Proposition 5.4 is obtained by setting $\lambda \in [1, d]$, $L \geq m$, and $\delta = 1/T$. □

A.4 Proof of Theorem 5.6

We prove the high probability version of Theorem 5.6, obtained by setting $\lambda \in [1, d]$, and $\delta = 1/T$.

Theorem A.1. *Let $\lambda \geq 1$, $\delta \in (0,1)$, and $\boldsymbol{a}_\tau$ be the blocks of actions in $\mathbb{R}^{d(m+L)}$ defined in (5.15). Then, with probability at least $1-\delta$ we have*

$$\sum_{\tau=1}^{T/(m+L)} \widetilde{r}(\widetilde{\boldsymbol{a}}) - \widetilde{r}(\boldsymbol{a}_\tau) \leq 4L(m+1)^{\gamma^+} \sqrt{Td\ \ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d\lambda}\right)} \cdot \left(\sqrt{\lambda} + \sqrt{\ln\left(\frac{1}{\delta}\right) + d\ \ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d(m+L)\lambda}\right)}\right).$$

Let $m \geq 1$, $T \geq m^2d^2+1$, and set $L = \lceil \sqrt{m/d}\, T^{1/4} \rceil - m$. Let r_t be the rewards collected when playing $\boldsymbol{a}_\tau$ as defined in (5.15). Then, with probability at least $1-\delta$ we have

$$\mathrm{OPT} - \sum_{t=1}^{T} r_t \leq 4\sqrt{d}\,(m+1)^{\frac{1}{2}+\gamma^+}\ T^{3/4}\left[1 + 2\sqrt{\ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d\lambda}\right)} \cdot \left(\sqrt{\frac{\lambda}{d}} + \sqrt{\frac{\ln(1/\delta)}{d} + \ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d\lambda}\right)}\right)\right].$$

Proof. The proof is along the lines of OFUL's analysis. The main difficulty is that we cannot use the elliptical potential lemma, see e.g., [Lattimore and Szepesvári, 2020, Lemma 19.4] due to the delay accumulated by V_τ, which is computed every $m+L$ round only. Let

$$\beta_\tau(\delta) = \sqrt{2\ln\left(\frac{1}{\delta}\right) + d\ \ln\left(1+\frac{\tau(m+1)^{2\gamma^+}}{d\lambda}\right)} + \sqrt{\lambda}\,. \tag{A.5}$$

By [Abbasi-Yadkori et al., 2011, Theorem 2], we have with probability at least $1-\delta$ that $\theta^* \in \mathcal{C}_\tau$ for every τ. It follows directly that $\boldsymbol{\theta}^* \in \boldsymbol{\mathcal{D}}_\tau$ for any τ, such that $\langle \widetilde{\boldsymbol{b}}, \boldsymbol{\theta}^* \rangle \leq \langle \boldsymbol{b}_\tau, \widetilde{\boldsymbol{\theta}}_\tau \rangle$, where $\widetilde{\boldsymbol{\theta}}_\tau = (0_d, \dots, 0_d, \widetilde{\theta}_\tau, \dots, \widetilde{\theta}_\tau)$ with $\widetilde{\theta}_\tau \in \mathbb{R}^d$ that maximizes (5.15) over $\mathcal{C}_{\tau-1}$. It can be shown that the regret is upper bounded by $\sum_\tau \sum_{i=m+1}^{m+L} \langle b_{\tau,i}, \widetilde{\theta}_\tau - \theta^* \rangle$. Following the standard analysis, one could then use

$$\left\langle b_{\tau,i}, \widetilde{\theta}_\tau - \theta^* \right\rangle \leq \|b_{\tau,i}\|_{V_{\tau-1}^{-1}} \left\|\widetilde{\theta}_t - \theta^*\right\|_{V_{\tau-1}}.$$

While the confidence set gives $\left\|\widetilde{\theta}_t - \theta^*\right\|_{V_{\tau-1}} \leq 2\beta_{\tau-1}(\delta)$, it is much more complex to bound the quantity $\sum_{i=m+1}^{m+L} \|b_{\tau,i}\|_{V_{\tau-1}^{-1}}$. Indeed, the elliptical potential lemma allows to bound $\sum_t \|a_t\|^2_{V_{t-1}^{-1}}$ when $V_t = \sum_{s\leq t} a_s a_s^\top + \lambda I_d$. However, recall that in our case we have $V_\tau = \sum_{\tau'=1}^{\tau} \sum_{i=m+1}^{m+L} b_{\tau',i} b_{\tau',i}^\top + \lambda I_d$, which is only computed every $m+L$ rounds. As a consequence, there exists a "delay" between $V_{\tau-1}$ and the action $b_{\tau,i}$ for $i \geq m+2$, preventing from using the lemma. Therefore, we propose to use instead

$$\left\langle b_{\tau,i}, \widetilde{\theta}_\tau - \theta^* \right\rangle \leq \|b_{\tau,i}\|_{V_{\tau,i-1}^{-1}} \left\|\widetilde{\theta}_t - \theta^*\right\|_{V_{\tau,i-1}}, \quad \text{where} \quad V_{\tau,i} = V_{\tau-1} + \sum_{j=m+1}^{i} b_{\tau,j} b_{\tau,j}^\top\,. \tag{A.6}$$

By doing so, the elliptical potential lemma applies. On the other hand, one has to control $\left\|\widetilde{\theta}_t - \theta^*\right\|_{V_{\tau,i-1}}$, which is not anymore bounded by $2\beta_{\tau-1}(\delta)$ since the subscript matrix is $V_{\tau,i-1}$ instead of $V_{\tau-1}$. Still, one can show that for any $i \leq m+L$ we have

$$
\begin{aligned}
&\left\|\widetilde{\theta}_t - \theta^*\right\|^2_{V_{\tau,i-1}} \\
&\quad = \operatorname{Tr}\left(V_{\tau,i-1}\left(\widetilde{\theta}_t - \theta^*\right)\left(\widetilde{\theta}_t - \theta^*\right)^\top\right) \\
&\quad = \operatorname{Tr}\left(\left(V_{\tau-1} + \sum_{j=m+1}^{i-1} b_{\tau,j} b_{\tau,j}^\top\right)\left(\widetilde{\theta}_t - \theta^*\right)\left(\widetilde{\theta}_t - \theta^*\right)^\top\right) \\
&\quad = \operatorname{Tr}\left(\left(I_d + \sum_{j=m+1}^{i-1}\left(V_{\tau-1}^{-1/2} b_{\tau,j}\right)\left(V_{\tau-1}^{-1/2} b_{\tau,j}\right)^\top\right) V_{\tau-1}^{1/2}\left(\widetilde{\theta}_t - \theta^*\right)\left(\widetilde{\theta}_t - \theta^*\right)^\top V_{\tau-1}^{1/2}\right) \\
&\quad \leq \left\|I_d + \sum_{j=m+1}^{i-1}\left(V_{\tau-1}^{-1/2} b_{\tau,j}\right)\left(V_{\tau-1}^{-1/2} b_{\tau,j}\right)^\top\right\|_* \operatorname{Tr}\left(V_{\tau-1}^{1/2}\left(\widetilde{\theta}_t - \theta^*\right)\left(\widetilde{\theta}_t - \theta^*\right)^\top V_{\tau-1}^{1/2}\right) \\
&\quad \leq \left(1 + \sum_{j=m+1}^{i-1}\left\|V_{\tau-1}^{-1/2} b_{\tau,j}\right\|_2^2\right)\left\|\widetilde{\theta}_t - \theta^*\right\|^2_{V_{\tau-1}} \\
&\quad \leq \left(1 + (L-1)(m+1)^{2\gamma^+}\right)\left\|\widetilde{\theta}_t - \theta^*\right\|^2_{V_{\tau-1}} \\
&\quad \leq L(m+1)^{2\gamma^+}\left\|\widetilde{\theta}_t - \theta^*\right\|^2_{V_{\tau-1}}.
\end{aligned} \tag{A.7}
$$

Recalling also that $\langle \widetilde{\boldsymbol{b}}, \boldsymbol{\theta}^*\rangle - \langle \boldsymbol{b}_\tau, \boldsymbol{\theta}^*\rangle \leq 2L(m+1)^{\gamma^+}$, we have with probability at least $1-\delta$

$$
\begin{aligned}
&\sum_{\tau=1}^{T/(m+L)} \langle \widetilde{\boldsymbol{b}}, \boldsymbol{\theta}^*\rangle - \langle \boldsymbol{b}_\tau, \boldsymbol{\theta}^*\rangle \\
&\quad \leq \sum_{\tau=1}^{T/(m+L)} \min\left\{2L(m+1)^{\gamma^+},\, \langle \boldsymbol{b}_\tau, \widetilde{\boldsymbol{\theta}}_\tau - \boldsymbol{\theta}^*\rangle\right\} \\
&\quad = \sum_{\tau=1}^{T/(m+L)} \min\left\{2L(m+1)^{\gamma^+},\, \sum_{i=m+1}^{m+L} \langle b_{\tau,i}, \widetilde{\theta}_\tau - \theta^*\rangle\right\} \\
&\quad \leq \sum_{\tau=1}^{T/(m+L)} \min\left\{2L(m+1)^{\gamma^+},\, \sum_{i=m+1}^{m+L} \|b_{\tau,i}\|_{V_{\tau,i-1}^{-1}} \left\|\widetilde{\theta}_t - \theta^*\right\|_{V_{\tau,i-1}}\right\} \\
&\quad \leq \sum_{\tau=1}^{T/(m+L)} \min\left\{2L(m+1)^{\gamma^+},\, 2\sqrt{L}(m+1)^{\gamma^+}\beta_{\tau-1}(\delta) \sum_{i=m+1}^{m+L} \|b_{\tau,i}\|_{V_{\tau,i-1}^{-1}}\right\} \\
&\quad \leq 2L(m+1)^{\gamma^+}\beta_{T/(m+L)}(\delta) \sum_{\tau=1}^{T/(m+L)} \sum_{i=m+1}^{m+L} \min\left\{1,\, \|b_{\tau,i}\|_{V_{\tau,i-1}^{-1}}\right\} \\
&\quad \leq 2L(m+1)^{\gamma^+}\beta_{T/(m+L)}(\delta) \sqrt{\frac{T\,L}{m+L} \sum_{\tau=1}^{T/(m+L)} \sum_{i=m+1}^{m+L} \min\left\{1,\, \|b_{\tau,i}\|^2_{V_{\tau,i-1}^{-1}}\right\}} \\
&\quad \leq 2\sqrt{2}L(m+1)^{\gamma^+}\beta_{T/(m+L)}(\delta) \sqrt{T\, \ln \frac{|V_{T/(m+L)}|}{|\lambda I_d|}}
\end{aligned}
$$

$$\leq 4L(m+1)^{\gamma^+} \sqrt{Td \ \ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d\lambda}\right)} \cdot \left(\sqrt{\lambda}+\sqrt{\ln\left(\frac{1}{\delta}\right)+d\ \ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d(m+L)\lambda}\right)}\right), \tag{A.8}$$

where we have used (A.5), (A.6), and (A.7). Similarly to Proposition 5.5, note that in the stationary case, i.e., when $m=0$ and $L=1$, we exactly recover [Abbasi-Yadkori et al., 2011, Theorem 3]. The first claim of Theorem 5.6 is obtained by setting $\lambda \in [1,d]$, and $\delta = 1/T$.
Let R_T denote the right-hand side of (A.8). Combining this bound with the arguments of Proposition 4.4, we have with probability $1-\delta$

$$\begin{aligned}
\sum_{t=1}^{T} r_t &\geq \sum_{\tau=1}^{T/(m+L)} \widetilde{r}(\boldsymbol{a}_\tau) - \frac{m(m+1)^{\gamma^+}}{m+L}\,T && \text{(A.9)}\\
&= \sum_{\tau=1}^{T/(m+L)} \langle \boldsymbol{b}_\tau, \boldsymbol{\theta}^* \rangle - \frac{m(m+1)^{\gamma^+}}{m+L}\,T \\
&\geq \sum_{\tau=1}^{T/(m+L)} \langle \widetilde{\boldsymbol{b}}, \boldsymbol{\theta}^* \rangle - R_T - \frac{m(m+1)^{\gamma^+}}{m+L}\,T && \text{(A.10)}\\
&= \sum_{\tau=1}^{T/(m+L)} \widetilde{r}(\widetilde{\boldsymbol{a}}) - R_T - \frac{m(m+1)^{\gamma^+}}{m+L}\,T \\
&\geq \sum_{t=1}^{T} \widetilde{r}_t - R_T - \frac{2m(m+1)^{\gamma^+}}{m+L}\,T && \text{(A.11)}\\
&\geq \mathrm{OPT} - R_T - \frac{4m(m+1)^{\gamma^+}}{m+L}\,T && \text{(A.12)}\\
&\geq \mathrm{OPT} - 4(m+1)^{\gamma^+}\left[\frac{mT}{m+L} + (m+L)\sqrt{Td\ \ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d\lambda}\right)}\right. \\
&\quad \left.\cdot\left(\sqrt{\lambda}+\sqrt{\ln\left(\frac{1}{\delta}\right)+d\ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d(m+L)\lambda}\right)}\right)\right],
\end{aligned}$$

where (A.9) and (A.11) come from the fact that any instantaneous reward is bounded by $(m+1)^{\gamma^+}$, see (5.12), (A.10) from (A.8), and (A.12) from Proposition 4.4.
Now, assume that $m \geq 1$, $T \geq d^2m^2+1$, and let $L = \left\lceil \sqrt{m/d}\ T^{1/4} \right\rceil - m$. By the condition on T, we have $\sqrt{m/d}\ T^{1/4} > m \geq 1$, such that $L \geq 1$ and

$$\sqrt{\frac{m}{d}}T^{1/4} \leq \left\lceil \sqrt{\frac{m}{d}}T^{1/4} \right\rceil = L+m \leq \sqrt{\frac{m}{d}}T^{1/4}+1 \leq 2\sqrt{\frac{m}{d}}T^{1/4}.$$

Substituting in the above bound, we have with probability $1-\delta$

$$\mathrm{OPT} - \sum_{t=1}^{T} r_t \le 4\sqrt{d}\,(m+1)^{\frac{1}{2}+\gamma^+}\; T^{3/4}\left[1 + 2\sqrt{\ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d\lambda}\right)}\right.$$
$$\left.\cdot\left(\sqrt{\frac{\lambda}{d}} + \sqrt{\frac{\ln(1/\delta)}{d} + \ln\left(1+\frac{T(m+1)^{2\gamma^+}}{d\lambda}\right)}\right)\right].$$

The second claim of Theorem 5.6 is obtained by setting $\lambda \in [1,d]$, and $\delta = 1/T$. □

A.5 Proof of Corollary 5.8

Lemma 5.7. *Suppose that a block-based bandit algorithm (in our case the bandit combiner) produces a sequence of T_{bc} blocks $\boldsymbol{a}_\tau$, with possibly different cardinalities $|\boldsymbol{a}_\tau|$, such that*

$$\sum_{\tau=1}^{T_{\mathrm{bc}}} \frac{\widetilde{r}(\widetilde{\boldsymbol{a}})}{|\widetilde{\boldsymbol{a}}|} - \sum_{\tau=1}^{T_{\mathrm{bc}}} \frac{\widetilde{r}(\boldsymbol{a}_\tau)}{|\boldsymbol{a}_\tau|} \le F(T_{\mathrm{bc}}),$$

for some sublinear function F. Then, we have

$$\frac{\min_\tau |\boldsymbol{a}_\tau|}{\max_\tau |\boldsymbol{a}_\tau|}\left(\widetilde{r}(\widetilde{\boldsymbol{a}})\,\frac{\sum_\tau |\boldsymbol{a}_\tau|}{|\widetilde{\boldsymbol{a}}|}\right) - \sum_{\tau=1}^{T_{\mathrm{bc}}} \widetilde{r}(\boldsymbol{a}_\tau) \;\le\; \min_\tau |\boldsymbol{a}_\tau|\,F(T_{\mathrm{bc}}).$$

In particular, if all blocks have the same cardinality the last bound is just the block regret bound scaled by $|\boldsymbol{a}_\tau|$.

Proof. We have

$$\begin{aligned}
\sum_{\tau=1}^{T_{\mathrm{bc}}} \widetilde{r}(\boldsymbol{a}_\tau) &\ge \min_\tau |\boldsymbol{a}_\tau| \sum_{\tau=1}^{T_{\mathrm{bc}}} \frac{\widetilde{r}(\boldsymbol{a}_\tau)}{|\boldsymbol{a}_\tau|} \\
&\ge \min_\tau |\boldsymbol{a}_\tau| \left(\sum_{\tau=1}^{T_{\mathrm{bc}}} \frac{\widetilde{r}(\widetilde{\boldsymbol{a}})}{|\widetilde{\boldsymbol{a}}|} - F(T_{\mathrm{bc}})\right) \\
&= \frac{\min_\tau |\boldsymbol{a}_\tau|}{\max_\tau |\boldsymbol{a}_\tau|}\,\frac{\widetilde{r}(\widetilde{\boldsymbol{a}})}{|\widetilde{\boldsymbol{a}}|}\,\max_\tau |\boldsymbol{a}_\tau|\,T_{\mathrm{bc}} - \min_\tau |\boldsymbol{a}_\tau|\,F(T_{\mathrm{bc}}) \\
&\ge \frac{\min_\tau |\boldsymbol{a}_\tau|}{\max_\tau |\boldsymbol{a}_\tau|}\left(\widetilde{r}(\widetilde{\boldsymbol{a}})\,\frac{\sum_\tau |\boldsymbol{a}_\tau|}{|\widetilde{\boldsymbol{a}}|}\right) - \min_\tau |\boldsymbol{a}_\tau|\,F(T_{\mathrm{bc}}).
\end{aligned}$$

□

Corollary 5.8. *Consider an instance of LBM with unknown parameters $(m_\star, \gamma_\star)$. Assume a bandit combiner is run on $N \le d\sqrt{m_\star}$ instances of `OFUL-memory` (Algorithm 4), each using a different pair of parameters (m_i, γ_i) from a set $\mathcal{S} = \big\{(m_1,\gamma_1),\dots,(m_N,\gamma_N)\big\}$ such that $(m_\star,\gamma_\star) \in \mathcal{S}$. Let $M = (\max_j m_j)/(\min_j m_j)$. Then, for all $T \ge (m_\star+1)^{2\gamma_\star^+}/m_\star d^4$, the expected rewards $\big(r_t^{\mathrm{bc}}\big)_{t=1}^{T}$ of the bandit* *combiner satisfy*

$$\frac{\mathrm{OPT}}{\sqrt{M}} - \mathbb{E}\left[\sum_{t=1}^{T} r_t^{\mathrm{bc}}\right] = \widetilde{\mathcal{O}}\Big(M\, d\, (m_\star + 1)^{1+\frac{3}{2}\gamma_\star^+}\, T^{3/4}\Big)\,.$$

Proof. Let $m_\star$ be the true memory size, and $L_\star = L(m_\star)$ the corresponding (partial) block length. Throughout the proof, $\widetilde{\boldsymbol{a}}$ denotes the block defined in (5.8) with length $m_\star + L_\star$. First observe that only one of the OFUL-memory instances we test is well-specified, i.e., has the true parameters $(m_\star, \gamma_\star)$. We can thus rewrite the regret bound for the Bandit Combiner [Cutkosky et al., 2020, Corollary 2], generalized to rewards bounded in $[-R, R]$ as follows

$$\mathrm{Regret}_{\mathrm{bc}} = \widetilde{\mathcal{O}}\left(C_\star T_{\mathrm{bc}}^{\alpha_\star} + C_\star^{\frac{1}{\alpha_\star}} T_{\mathrm{bc}} \eta_\star^{\frac{1-\alpha_\star}{\alpha_\star}} + R^2 T_{\mathrm{bc}} \eta_\star + \sum_{j \neq \star} \frac{1}{\eta_j}\right)\,, \tag{A.13}$$

where $T_{\mathrm{bc}} = T/(m_\star + L_\star)$ is the bandit combiner horizon, $C_\star$ and $\alpha_\star$ are the constants in the regret bound of the well-specified instance (see below how we determine them), and the η_j are free parameters to be tuned. We now derive $C_\star$ and $\alpha_\star$. To that end, we must establish the regret bound of the well-specified instance, and identify $C_\star$ and $\alpha_\star$ such that this bound is equal to $C_\star T_{\mathrm{bc}}^{\alpha_\star}$, where $C_\star$ may contain logarithmic factors. For the well-specified instance, the first claim of Theorem A.1 gives that, with probability at least $1 - \delta$, we have

$$\sum_{\tau=1}^{T/(m_\star+L_\star)} \widetilde{r}(\widetilde{\boldsymbol{a}}) - \widetilde{r}(\boldsymbol{a}_\tau) \leq 4(m_\star + L_\star)(m_\star + 1)^{\gamma_\star^+} \sqrt{Td\, \ln\left(1 + \frac{T(m_\star+1)^{2\gamma_\star^+}}{d\lambda}\right)} \left(\sqrt{\lambda} + \sqrt{\ln\left(\frac{1}{\delta}\right) + d\, \ln\left(1 + \frac{T(m_\star+1)^{2\gamma_\star^+}}{d(m_\star+L_\star)\lambda}\right)}\right)$$

$$\sum_{\tau=1}^{T/(m_\star+L_\star)} \frac{\widetilde{r}(\widetilde{\boldsymbol{a}})}{|\widetilde{\boldsymbol{a}}|} - \frac{\widetilde{r}(\boldsymbol{a}_\tau)}{|\boldsymbol{a}_\tau|} \leq T^{1/2}\, 4(m_\star + 1)^{\gamma_\star^+} \sqrt{d \ln\left(1 + \frac{T(m_\star+1)^{2\gamma_\star^+}}{d\lambda}\right)} \left(\sqrt{\lambda} + \sqrt{\ln\left(\frac{1}{\delta}\right) + d\, \ln\left(1 + \frac{T(m_\star+1)^{2\gamma_\star^+}}{d(m_\star+L_\star)\lambda}\right)}\right)\,, \tag{A.14}$$

where we have used that $|\boldsymbol{a}_\tau| = |\widetilde{\boldsymbol{a}}| = m_\star + L_\star$ for every τ. Note that the right-hand side of (A.14) is expressed in terms of T, which is not the correct horizon, $T/(m_\star + L_\star)$. However, recall that we have

$$m_\star + L_\star \leq 2\sqrt{\frac{m_\star}{d}}\, T^{1/4}$$

$$(m_\star + L_\star)^4 \leq \left(\frac{4m_\star}{d}\right)^2 T$$

$$\begin{aligned} T^3 &\le \left(\frac{4m_\star}{d}\right)^2 \left(\frac{T}{m_\star + L_\star}\right)^4 \\ T^{1/2} &\le \left(\frac{4m_\star}{d}\right)^{1/3} \left(\frac{T}{m_\star + L_\star}\right)^{2/3} , \end{aligned}$$

such that by substituting in (A.14) and identifying we have $\alpha_\star = 2/3$, and

$$C_\star = 4\left(\frac{4m_\star}{d}\right)^{1/3} (m_\star+1)^{\gamma_\star^+} \sqrt{d \ln\left(1 + \frac{T_{\text{bc}}(m_\star + L_\star)(m_\star+1)^{2\gamma_\star^+}}{d\lambda}\right)} \\ \left(\sqrt{\lambda} + \sqrt{\ln\left(\frac{1}{\delta}\right) + d\ \ln\left(1 + \frac{T_{\text{bc}}(m_\star+1)^{2\gamma_\star^+}}{d\lambda}\right)}\ \right) .$$

Setting $\eta_j = T_{\text{bc}}^{-2/3}$, and substituting in (A.13) with $R = (m_\star + 1)^{\gamma_\star^+}$, we have that with high probability

$$\sum_{\tau=1}^{T_{\text{bc}}} \frac{\widetilde{r}(\widetilde{\boldsymbol{a}})}{|\widetilde{\boldsymbol{a}}|} - \frac{\widetilde{r}(\boldsymbol{a}_\tau^{\text{bc}})}{|\boldsymbol{a}_\tau^{\text{bc}}|} = \widetilde{\mathcal{O}}\Big(\big(C_\star^{3/2} + N\big)\, T_{\text{bc}}^{2/3} +\ (m_\star+1)^{2\gamma_\star^+}\, T_{\text{bc}}^{1/3}\Big) .$$

Now, recall that $T_{\text{bc}} = \mathcal{O}\big(\sqrt{d/m_\star}\, T^{3/4}\big)$, and that $C_\star = \widetilde{\mathcal{O}}\big((m_\star+1)^{\frac{1}{3}+\gamma_\star^+}\, d^{2/3}\big)$. Hence, $N \le d\sqrt{m_\star}$ implies $N = \mathcal{O}\big(C_j^{3/2}\big)$, and $(m_\star+1)^{\gamma_\star^+} \le d^2\sqrt{m_\star T}$ implies $(m_\star+1)^{\gamma_\star^+}\, T_{\text{bc}}^{1/3} = \mathcal{O}\big(C_\star^{3/2}\, T_{\text{bc}}^{2/3}\big)$. Setting $\lambda \in [1, d]$, $\delta = 1/T$, we obtain

$$\mathbb{E}\left[\sum_{\tau=1}^{T_{\text{bc}}} \frac{\widetilde{r}(\widetilde{\boldsymbol{a}})}{|\widetilde{\boldsymbol{a}}|} - \frac{\widetilde{r}(\boldsymbol{a}_\tau^{\text{bc}})}{|\boldsymbol{a}_\tau^{\text{bc}}|}\right] = \widetilde{\mathcal{O}}\Big(d\,\sqrt{m_\star}\,(m_\star+1)^{\frac{3}{2}\gamma_\star^+}\, T_{\text{bc}}^{2/3}\Big) . \tag{A.15}$$

Let m_τ be the memory size associated to the bandit played at block time step τ by Algorithm 4. Let $m_{\min} = \min_j m_j$ and $m_{\max} = \max_j m_j$. Finally, let $L_{\min}$ and $L_{\max}$ the (partial) block length associated with $m_{\min}$ and $m_{\max}$. We have

$$\sum_{t=1}^{T} r_t^{\text{bc}} \ge \sum_{\tau=1}^{T_{\text{bc}}} \Big(\widetilde{r}(\boldsymbol{a}_\tau^{\text{bc}}) - m_\tau\,(m_\star+1)^{\gamma_\star^+}\Big) \ge \sum_{\tau=1}^{T_{\text{bc}}} \widetilde{r}(\boldsymbol{a}_\tau^{\text{bc}}) - m_{\max}\,(m_\star+1)^{\gamma_\star^+}\, T_{\text{bc}} ,$$

such that by Lemma 5.7 and (A.15) we obtain

$$\mathbb{E}\left[\frac{\min_\tau |\boldsymbol{a}_\tau|}{\max_\tau |\boldsymbol{a}_\tau|}\left(\widetilde{r}(\widetilde{\boldsymbol{a}})\,\frac{\sum_\tau |\boldsymbol{a}_\tau|}{|\widetilde{\boldsymbol{a}}|}\right) - \sum_{t=1}^{T} r_t^{\text{bc}}\right] \le m_{\max}\,(m_\star+1)^{\gamma_\star^+}\, T_{\text{bc}} \\ + \min_\tau |\boldsymbol{a}_\tau|\, \widetilde{\mathcal{O}}\Big(d\,\sqrt{m_\star}\,(m_\star+1)^{\frac{3}{2}\gamma_\star^+}\, T_{\text{bc}}^{2/3}\Big) ,$$

$$\mathbb{E}\left[\frac{m_{\min} + L_{\min}}{m_{\max} + L_{\max}}\left(\frac{L_\star\,\text{OPT}}{T}\,\frac{T}{m_\star + L_\star}\right) - \sum_{t=1}^{T} r_t^{\text{bc}}\right] \le \frac{m_{\max}\,(m_\star+1)^{\gamma_\star^+}\, T}{m_{\min} + L_{\min}} \\ + (m_{\min} + L_{\min})^{1/3}\, \widetilde{\mathcal{O}}\Big(d\,\sqrt{m_\star}\,(m_\star+1)^{\frac{3}{2}\gamma_\star^+}\, T^{2/3}\Big) ,$$

$$\begin{aligned}\mathbb{E}\left[\sqrt{\frac{m_{\min}}{m_{\max}}}\ \mathrm{OPT} - \sum_{t=1}^{T} r_t^{\mathrm{bc}}\right] &\leq \frac{m_{\max}}{m_{\min}}\sqrt{d\,m_\star}\,(m_\star+1)^{\gamma_\star^+}\,T^{3/4} \\ &\quad + \widetilde{\mathcal{O}}\Big(d\,m_\star\,(m_\star+1)^{\frac{3}{2}\gamma_\star^+}\,T^{3/4}\Big) \\ &= \frac{m_{\max}}{m_{\min}}\ \widetilde{\mathcal{O}}\Big(d\,m_\star\,(m_\star+1)^{\frac{3}{2}\gamma_\star^+}\,T^{3/4}\Big)\,,\end{aligned}$$

where we have used the fact that $m_{\min} + L_{\min} = \sqrt{m_{\min}/d}\ T^{1/4}$, and $m_{\max} + L_{\max} = \sqrt{m_{\max}/d}\ T^{1/4}$. Corollary 5.8 is obtained by setting $M = m_{\max}/m_{\min}$.

□

Bibliography

Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems*, 24, 2011.

S. Agrawal and N. Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on learning theory*, pages 39–1. JMLR Workshop and Conference Proceedings, 2012.

R. Alami, O. Maillard, and R. Féraud. Memory bandits: a bayesian approach for the switching bandit problem. In *NIPS 2017-31st conference on neural information processing systems*, 2017.

AllMusic. Music genres. https://www.allmusic.com/genres, 2021. Accessed: 2021-05-06.

C. V. S. Araujo. A model for predicting music popularity on spotify. Extended Abstracts for the Late-Breaking Demo Session of the 21st International Society for Music Information Retrieval Conference, 2020.

A. Atsidakou, O. Papadigenopoulos, S. Basu, C. Caramanis, and S. Shakkottai. Combinatorial blocking bandits with stochastic delays. In *International Conference on Machine Learning*, pages 404–413. PMLR, 2021.

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. IEEE, 1995.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47:235–256, 2002.

P. Auer, P. Gajane, and R. Ortner. Adaptively tracking the best bandit arm with an unknown number of distribution changes. In A. Beygelzimer and D. Hsu, editors, *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 138–158. PMLR, 25–28 Jun 2019. URL https://proceedings.mlr.press/v99/auer19a.html.

A. Bar-Noy, R. Bhatia, J. Naor, and B. Schieber. Minimizing service and operation costs of periodic scheduling. *Mathematics of Operations Research*, 27(3):518–544, 2002.

A. Baratè and L. A. Ludovico. A web platform to extract and investigate music genre labels in spotify. In *Proceedings of the 19th Sound and Music Computing Conference*, 2022. ISBN 978-2-9584126-0-9. doi: 10.5281/zenodo.6573370.

S. Basu, R. Sen, S. Sanghavi, and S. Shakkottai. Blocking bandits. *Advances in Neural Information Processing Systems*, 32, 2019.

S. Basu, O. Papadigenopoulos, C. Caramanis, and S. Shakkottai. Contextual blocking bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 271–279. PMLR, 2021.

O. Besbes, Y. Gur, and A. Zeevi. Stochastic multi-armed-bandit problem with non-stationary rewards. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/903ce9225fca3e988c2af215d4e544d3-Paper.pdf.

O. Besbes, Y. Gur, and A. Zeevi. Optimal exploration-exploitation in a multi-armed-bandit problem with non-stationary rewards, 2019.

L. Besson and E. Kaufmann. The generalized likelihood ratio test meets klucb: an improved algorithm for piece-wise non-stationary bandits. *Proceedings of Machine Learning Research vol XX*, 1:35, 2019.

P. Boldi and S. Vigna. The webgraph framework i: compression techniques. In *Proceedings of the 13th international conference on World Wide Web*, pages 595–602, New York, NY, 2004a. ACM.

P. Boldi and S. Vigna. The webgraph framework ii: Codes for the world-wide web. In *Data Compression Conference, 2004. Proceedings. DCC 2004*, page 528, New York, NY, 2004b. IEEE, ACM.

P. Boldi and S. Vigna. In-core computation of geometric centralities with hyperball: A hundred billion nodes and beyond. In *2013 IEEE 13th International Conference on Data Mining Workshops*, pages 621–628, New York, NY, 2013. IEEE. doi: 10.1109/ICDMW.2013.10.

P. Boldi and S. Vigna. Axioms for centrality. *Internet Mathematics*, 10(3-4): 222–262, 2014. doi: 10.1080/15427951.2013.865686.

D. Bouneffouf and R. Féraud. Multi-armed bandit problem with known trend. *Neurocomputing*, 205:16–21, 2016.

R. R. Bush and F. Mosteller. A stochastic model with applications to learning. *The Annals of Mathematical Statistics*, pages 559–585, 1953.

L. Cella and N. Cesa-Bianchi. Stochastic bandits with delay-dependent payoffs. In *International Conference on Artificial Intelligence and Statistics*, pages 1168–1177. PMLR, 2020.

N. Cesa-Bianchi and G. Lugosi. Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422, 2012.

W. Chen, Y. Wang, and Y. Yuan. Combinatorial multi-armed bandit: General framework and applications. In *International conference on machine learning*, pages 151–159. PMLR, 2013.

J. Cheng, D. M. Romero, B. Meeder, and J. Kleinberg. Predicting reciprocity in social networks. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, 2011. doi: 10.1109/PASSAT/SocialCom.2011.110.

W. C. Cheung, D. Simchi-Levi, and R. Zhu. Learning to optimize under non-stationarity. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1079–1087. PMLR, 2019.

J. Clausen. Branch and bound algorithms-principles and examples. *Department of Computer Science, University of Copenhagen*, pages 1–30, 1999.

G. Clerici* and M. Tiraboschi*. Citation is not collaboration: Music-genre dependence of graph-related metrics in a music credits network. In *Proceedings of the 20th Sound and Music Computing*, pages 317–322. Royal College of Music and KTH Royal Institute of Technology, 2023.

G. Clerici, P. Laforgue, and N. Cesa-Bianchi. Linear bandits with memory, 2023.

J. Cohen. *Statistical power analysis for the behavioral sciences.* Routledge, Abingdon-on-Thames, UK, 1988. doi: 10.4324/9780203771587.

K. Crauwels. Musicmap - the genealogy and history of popular music genres from origin till present (1870-2016). https://musicmap.info, 2016. Accessed: 2021-10-05.

A. Cutkosky, A. Das, and M. Purohit. Upper confidence bounds for combining stochastic bandits. *arXiv preprint arXiv:2012.13115*, 2020.

M. R. Dimitrijević, J. Faganel, M. Gregorić, P. Nathan, and J. Trontelj. Habituation: effects of regular and stochastic stimulation. *Journal of Neurology, Neurosurgery & Psychiatry*, 35(2):234–242, 1972.

F. Fabbri. A theory of musical genres: two applications. *Popular music: critical concepts in media and cultural studies*, 3:7–35, 2004.

D. J. Foster, A. Krishnamurthy, and H. Luo. Model selection for contextual bandits. *Advances in Neural Information Processing Systems*, 32, 2019.

M. Fuller-Tyszkiewicz, B. Richardson, V. Lewis, J. Linardon, J. Mills, K. Juknaitis, C. Lewis, K. Coulson, R. O'Donnell, L. Arulkadacham, et al. A randomized trial exploring mindfulness and gratitude exercises as ehealth-based micro-interventions for improving body satisfaction. *Computers in Human Behavior*, 95:58–65, 2019.

Y. Gai, B. Krishnamachari, and R. Jain. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Transactions on Networking*, 20(5):1466–1478, 2012.

A. Garivier and E. Moulines. On upper-confidence bound policies for switching bandit problems. In *International Conference on Algorithmic Learning Theory*, pages 174–188. Springer, 2011.

D. Garlaschelli and M. I. Loffredo. Patterns of link reciprocity in directed networks. *Physical Review Letters*, 93:268701, Dec 2004. doi: 10.1103/PhysRevLett.93.268701. URL https://link.aps.org/doi/10.1103/PhysRevLett.93.268701.

J. Gittins, K. Glazebrook, and R. Weber. *Multi-armed bandit allocation indices.* John Wiley & Sons, 2011.

J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 41(2):148–164, 1979.

A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught, and J. Millman, editors, *Proceedings of the 7th Python in Science Conference*, Pasadena, CA USA, 2008. SciPy2008.

C. Hartland, N. Baskiotis, S. Gelly, M. Sebag, and O. Teytaud. Change point detection and meta-bandits for online learning in dynamic environments. In *CAp 2007: 9è Conférence francophone sur l'apprentissage automatique*, pages 237–250, 2007.

H. Heidari, M. J. Kearns, and A. Roth. Tight policy regret bounds for improving and decaying bandits. In *IJCAI*, pages 1562–1570, 2016.

R. Holte, A. Mok, L. Rosier, I. Tulchinsky, and D. Varvel. The pinwheel: A real-time scheduling problem. In *Proceedings of the 22nd Hawaii International Conference of System Science*, pages 693–702, 1989.

J.-h. Kim, M. Vojnovic, and S.-Y. Yun. Rotting infinitely many-armed bandits. In *International Conference on Machine Learning*, pages 11229–11254. PMLR, 2022.

R. Kleinberg and N. Immorlica. Recharging bandits. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 309–319. IEEE, 2018.

T. Kocák, G. Neu, M. Valko, and R. Munos. Efficient learning by implicit exploration in bandit problems with side observations. *Advances in Neural Information Processing Systems*, 27, 2014.

L. Kocsis and C. Szepesvari. Bandit based monte-carlo planning, ecml'06 in: Ecml-06, 2006.

G. Kovacs, Z. Wu, and M. S. Bernstein. Rotating online behavior change interventions increases effectiveness but also increases attrition. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):1–25, 2018.

J. K. Kruschke. Bayesian estimation supersedes the t test. *Journal of Experimental Psychology: General*, 142(2):573, 2013. doi: 10.1037/a0029146.

M. Kunaver and T. Požrl. Diversity in recommender systems–a survey. *Knowledge-based systems*, 123:154–162, 2017.

B. Kveton, Z. Wen, A. Ashkan, and C. Szepesvari. Tight regret bounds for stochastic combinatorial semi-bandits. In *Artificial Intelligence and Statistics*, pages 535–543. PMLR, 2015.

P. Laforgue, G. Clerici, N. Cesa-Bianchi, and R. Gilad-Bachrach. A last switch dependent analysis of satiation and seasonality in bandits. In G. Camps-Valls, F. J. R. Ruiz, and I. Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 971–990. PMLR, 28–30 Mar 2022. URL https://proceedings.mlr.press/v151/laforgue22a.html.

T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

A. H. Land and A. G. Doig. *An automatic method for solving discrete programming problems*. Springer, 2010.

T. Lattimore and C. Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

L. Leqi, F. Kilinc Karzan, Z. Lipton, and A. Montgomery. Rebounding bandits for modeling satiation effects. *Advances in Neural Information Processing Systems*, 34:4003–4014, 2021.

N. Levine, K. Crammer, and S. Mannor. Rotting bandits. *Advances in neural information processing systems*, 30, 2017.

L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670, 2010.

N. Lin. *Foundations of Social Research*. McGraw-Hill, New York, NY, 1976.

F. Liu, J. Lee, and N. Shroff. A change-detection based framework for piecewise-stationary multi-armed bandit problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

G. Lugosi, C. Pike-Burke, and P.-A. Savalle. Bandit problems with fidelity rewards. *arXiv preprint arXiv:2111.13026*, 2021.

Y. Matsumoto, R. Harakawa, T. Ogawa, and M. Haseyama. Context-aware network analysis of music streaming services for popularity estimation of artists. *IEEE Access*, 8:48673–48685, 2020. doi: 10.1109/ACCESS.2020.2978281.

G. McDonald. Every noise at once. https://everynoise.com, 2013. Accessed: 2021-10-05.

G. Meinlschmidt, J.-H. Lee, E. Stalujanis, A. Belardi, M. Oh, E. K. Jung, H.-C. Kim, J. Alfano, S.-S. Yoo, and M. Tegethoff. Smartphone-based psychotherapeutic micro-interventions to improve mood in a real-world setting. *Frontiers in psychology*, 7:1112, 2016.

A. M. Metelli, F. Trovo, M. Pirola, and M. Restelli. Stochastic rising bandits. In *International Conference on Machine Learning*, pages 15421–15457. PMLR, 2022.

K. E. Nelson, M. K. Scherer, and U. N. N. S. Administration. Jpype, 6 2020.

G. P. Oliveira, M. O. Silva, D. B. Seufitelli, A. Lacerda, and M. M. Moro. Detecting collaboration profiles in success-based music genre networks. In *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pages 726–732, Montreal, Canada, 2020. ISMIR.

J. E. Owen, E. Kuhn, B. K. Jaworski, P. McGee-Vincent, K. Juhasz, J. E. Hoffman, and C. Rosen. Va mobile apps for ptsd and related problems: public health resources for veterans and those who care for them. *Mhealth*, 4, 2018.

A. Pacchiano, M. Phan, Y. Abbasi Yadkori, A. Rao, J. Zimmert, T. Lattimore, and C. Szepesvari. Model selection in contextual stochastic bandit problems. *Advances in Neural Information Processing Systems*, 33:10328–10337, 2020.

P. Paredes, R. Gilad-Bachrach, M. Czerwinski, A. Roseway, K. Rowan, and J. Hernandez. Poptherapy: Coping with stress through pop-culture. In *Proceedings of the 8th international conference on pervasive computing technologies for healthcare*, pages 109–117, 2014.

C. Pike-Burke and S. Grunewalder. Recovering bandits. *Advances in Neural Information Processing Systems*, 32, 2019.

A. Rajaraman and J. D. Ullman. *Mining of massive datasets*. Cambridge University Press, 2011.

Y. Rochat. Closeness centrality extended to unconnected graphs: The harmonic centrality index. Technical report, Institute of Applied Mathematics University of Lausanne, 2009.

Y. Russac, C. Vernade, and O. Cappé. Weighted linear bandits for non-stationary environments. *Advances in Neural Information Processing Systems*, 32, 2019.

D. J. Russo, B. Van Roy, A. Kazerouni, I. Osband, Z. Wen, et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1): 1–96, 2018.

D. R.-J. G.-J. Rydning, J. Reinsel, and J. Gantz. The digitization of the world from edge to core. *Framingham: International Data Corporation*, 16:1–28, 2018.

J. Salvatier, T. V. Wiecki, and C. Fonnesbeck. Probabilistic programming in python using pymc3. *PeerJ Computer Science*, 2:e55, 2016. doi: 10.7717/peerj-cs.55.

M. Schedl, H. Zamani, C.-W. Chen, Y. Deldjoo, and M. Elahi. Current challenges and visions in music recommender systems research. *International Journal of Multimedia Information Retrieval*, 7:95–116, 2018.

J. Schroeder, C. Wilkes, K. Rowan, A. Toledo, A. Paradiso, M. Czerwinski, G. Mark, and M. M. Linehan. Pocket skills: A conversational mobile web app to support dialectical behavioral therapy. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2018.

M. Sciandra and I. C. Spera. A model-based approach to spotify data analysis: a beta glmm. *Journal of Applied Statistics*, 49(1):214–229, 2022. doi: 10.1080/02664763.2020.1803810.

J. Seznec. *Apprentissage automatique séquentiel pour les systèmes éducatifs intelligents*. PhD thesis, École Doctorale Sciences Pour l'Ingénieur, 2020.

J. Seznec, A. Locatelli, A. Carpentier, A. Lazaric, and M. Valko. Rotting bandits are no harder than stochastic ones. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2564–2572. PMLR, 2019.

J. Seznec, P. Menard, A. Lazaric, and M. Valko. A single algorithm for both restless and rested rotting bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 3784–3794. PMLR, 2020.

D. Simchi-Levi, Z. Zheng, and F. Zhu. Dynamic planning and learning under recovering rewards. In *International Conference on Machine Learning*, pages 9702–9711. PMLR, 2021.

T. South, M. Roughan, and L. Mitchell. Popularity and centrality in spotify networks: critical transitions in eigenvector centrality. *Journal of Complex Networks*, 8(6):1–18, Dec 2020. ISSN 2051-1329. doi: 10.1093/comnet/cnaa050.

Spotify. Spotify web api. https://developer.spotify.com/documentation/web-api, 2021. Accessed: 2022-05-06.

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. Wardil and C. Hauert. Origin and structure of dynamic cooperative networks. *Scientific Reports*, 4(1):5725, Jul 2014. ISSN 2045-2322. doi: 10.1038/srep05725.

R. Warlop, A. Lazaric, and J. Mary. Fighting boredom in recommender systems with linear reinforcement learning. *Advances in Neural Information Processing Systems*, 31, 2018.

L. Wei and V. Srivatsva. On abruptly-changing and slowly-varying multiarmed bandit problems. In *2018 Annual American Control Conference (ACC)*, pages 6291–6296. IEEE, 2018.

P. Whittle. Restless bandits: Activity allocation in a changing world. *Journal of applied probability*, 25(A):287–298, 1988.

Wikipedia. Comparison of music streaming services. `https://en.wikipedia.org/wiki/Comparison_of_music_streaming_services`, 2023. Accessed: 2023-02-06.

M. Woodroofe. A one-armed bandit problem with a concomitant variable. *Journal of the American Statistical Association*, 74(368):799–806, 1979.

J. Y. Yu, S. Mannor, and N. Shimkin. Markov decision processes with arbitrary reward processes. *Mathematics of Operations Research*, 34(3):737–757, 2009.