Acceleration in First-Order Optimization Methods: Promenading Beyond Convexity or Smoothness, and Applications



David Martínez-Rubio
Keble College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Michaelmas 2021

Starting with what's easy,
makes what's hard: easy.

Miguel de Guzmán

Abstract

Acceleration in optimization is a term that is generally applied to optimization algorithms presenting some common methodology that enjoy convergence rates that improve over other more simple algorithms for the same problem. For example, Nesterov's Accelerated Gradient Descent improves over the gradient descent method for the optimization of smooth and convex functions. In this thesis, we investigate the acceleration phenomenon and its applications for three previously studied research questions in *optimization* and *online learning*: geodesically convex accelerated optimization in Riemannian manifolds, approximation of a proportional fair solution under positive linear constraints, also known as the 1-fair packing problem, and regret minimization in a decentralized cooperative stochastic multi-armed bandit problem with no reward collisions. We improve over previously studied approaches by using acceleration as a key element in all of our algorithms: we obtain an accelerated method for some non-convex problems in the first case, we exploit a local decrease condition of our problem in the second case, avoiding the use of poor and non-global smoothness, and in the third problem, we design a solution for which we can effectively apply an accelerated gossip protocol to spread and use information.

In particular, we provide an extensive overview of acceleration techniques that provide context for ours proofs. In Riemannian optimization, we design the first *global accelerated* algorithm for the optimization of smooth functions that are geodesically convex or geodesically strongly convex and that are defined in manifolds of constant sectional curvature. We reduce these problems to a Euclidean non-convex problem and design a global accelerated method, obtaining a solution with the same rates of Accelerated Gradient Descent in the Euclidean space, up to logarithmic factors and constants depending on the initial distance to an optimum and on the curvature. We also provide some reductions. Regarding our fairness problem, we present a deterministic, accelerated, distributed and width-independent algorithm for the 1-fair packing problem. We obtain a quadratic improvement in the number of iterations and remove polylog(width) factors with respect to the previous best solution. In our bandit problem, we design a decentralized upper confidence bound algorithm to minimize the expected regret. It allows for the use of delayed and approximate estimations of the arms' means which we can obtain fast with an accelerated gossip communication protocol. We provide theoretical and empirical comparisons showing we obtain lower regret than previous state of the art.

Acknowledgements

First of all, I would like to thank my supervisors Varun Kanade and Patrick Rebeschini for their time, guidance, patience, effort and help during all these years. Thank you for all the meetings and for providing me with the flexibility to explore ideas I wanted to pursue. I am also very grateful to Coralia Cartis and Alexandre d'Aspremont for serving as examiners for my thesis. I would like to thank my former professors José F. Fernando Galván, José Manuel Gamboa Mutuberría, Mariemi Alonso García, Marco Antonio Gómez Martín, Narciso Martí Oliet, and David de Frutos Escrig who continued to support me in one way or another well past my undergraduate years. A big thank you to Andrés Sáez Schwedt and Mari Ángeles Salio who helped me in my earliest contacts with mathematics. I want to thank my former Wadham advisor Mark Thompson for his help whenever it was required.

I had a great time collaborating with my co-authors Güneş, Rob, Mario, Tuan Ahn, Varun, Patrick, Chris, Guillermo, Paco, and Sebastian. It was also great to attend and participate in the reading groups happening at Oxford; thanks to everyone involved. I would also like to thank the people at the Engineering Science lab for being such good mates: Adam, Andrew, Bradley, Güneş, Tobias, Marcin, Mario, Max, Michael, Rob Cornish, Rob Zinkov, Stefan, Tom, Tuan Anh, Yuan. A special thanks to Mario Lezcano-Casado for the countless hours we spent with and without mathematics. Also to Guillermo for his creativity, curiosity and his vision. I also want to thank Amartya Sanyal with whom I spent many hours learning and discussing ideas.

I would like to thank my housemates at the 13th of our street, to people who stopped by: Pablo, Álvaro, David, Alice, Pablo again, Álex and to the ones who crashed there. The first year of pandemic I spent a phenomenal lockdown with Mario, Adri and Garima at home and the same goes for Sam and Izar for the second year. A big thanks to the members of the *Comeis?* group: Mario, Paco, Jaime, Alejandro, and Aitor. Also to the extra members of its extensions *Oxfork* and *Caméis?*: Adrià and Guillermo. To Álex, David, Guillermo, Laura, Mateo, and Paula from the *Puntentados* gang and Alejandra, David, Jaime and Marcos from the *Redback* gang. I spent many good times with all of them. Thanks for that invaluable friendship. Last long, and thanks for all the fun.

Last, but not least, I want to thank my family, specially my parents, my brother and my uncle Félix, for their constant support.

Contents

1 Introduction **1**

1.1 Main contributions . . . 4

2 Acceleration in Optimization Methods **9**

2.1 Preliminaries . . . 10

2.2 Online learning: FTRL and Mirror Descent . . . 14

2.2.1 Follow the Regularized Leader . . . 16

2.2.2 (Online) Mirror Descent . . . 21

2.3 Nemirovski's Quasi-accelerated gradient descent . . . 24

2.4 Nesterov's Accelerated Gradient Descent . . . 28

2.5 Linear Coupling . . . 31

2.6 The Approximate Duality Gap Technique . . . 35

2.6.1 Incorporating Mirror Descent . . . 39

2.7 Others . . . 42

3 Acceleration in First-Order Riemannian Optimization **47**

3.1 Introduction . . . 47

3.2 Algorithm . . . 53

3.2.1 Sketch of the proof of Theorem 3.2.4 . . . 58

3.3 Reductions . . . 60

3.4 Discussion . . . 65

3.5 Acceleration: Proofs of Theorem 3.2.4 and Theorem 3.2.5 . . . 66

3.6 Geometric results: Proofs of Lemmas 3.2.1, 3.2.2 and 3.2.3 . . . 76

3.6.1 Preliminaries . . . 77

3.6.2 Distance deformation . . . 78

3.6.3 Angle deformation . . . 83

3.6.4 Gradient deformation and smoothness of f . . . 85

3.6.5 Proof of Lemma 3.2.2 . . . 89

3.7 Constants depending on R and K . . . 92

3.7.1 Comment on hardness results . . . 96

4 Proportional Fairness under Positive Linear Constraints **98**
4.1 Introduction . . . 98
4.2 A distributed accelerated algorithm for 1-Fair Packing . . . 101
4.2.1 Coupling Mirror Descent and Gradient Descent . . . 106
4.3 Other proofs . . . 109

5 Decentralized Cooperative Stochastic Bandits **113**
5.1 Introduction . . . 113
5.2 Related work . . . 115
5.3 Model and problem formulation . . . 117
5.4 Algorithm . . . 118
5.5 Experiments . . . 132

6 Conclusion **135**

Bibliography **136**

Chapter 1

Introduction

In recent years, there has been a surge in the study of first-order methods in optimization, motivated by their applications to large scale problems in fields such as machine learning. An old but effective method, is Gradient Descent (GD), stochastic or not. The deterministic method dates back to at least 1847 (cf. Lemaréchal, 2012), and the stochastic one is from 1951 (Robbins and Monro, 1951). Nowadays the method is still used in many applications. Gradient Descent is an iterative local-search memoryless algorithm that essentially goes from one iterate to the next one by following a local direction of descent with the aim of minimizing a function. In many problem settings, there is a phenomenon which is that one can improve over GD or over other natural simple algorithms for our setting, by using other iterative first-order methods. Further, such solutions usually share some common methodology, like estimating a lower bound on the objective by using the historic information received in previous iterations, in order to obtain some kind of dual progress, or in order to estimate some other structural properties. Such algorithms combine these estimations with other properties of the objective, such as smoothness or finite-sum structure, in order to improve the rate of decrease of the optimality gap. In such a case, one says the algorithm is accelerated. Despite there being no formal definition of what acceleration is, the community agrees on the application of this term to many algorithms sharing these design ideas and properties. We also note that, in fact, acceleration appears in higher order methods as well (Monteiro and Svaiter, 2013; Gasnikov et al., 2019a; Bubeck, Jiang, et al., 2019; Bullins and Peng, 2019; Carmon, Jambulapati, et al., 2020). The most prominent example of an accelerated algorithm is Accelerated Gradient Descent (AGD), applying to smooth convex functions (along with its sibling for smooth, strongly convex functions), which is an algorithm that accelerates over GD and achieves optimality for the black-box optimization of such functions via a first-order oracle (Nesterov, 1983; Nemirovski and Nesterov, 1985). There

Most of the notations in this thesis have a non-highlighted link to their definitions. For example, if you click on any instance of e_i, you will jump to the place where it is defined as the i-th vector of the canonical base. Most definitions are local to their sections and context. Excluding this footnote, this N only appears in Chapter 4 and links to its definition of the number of non-zeros of the matrix defined in the fairness packing problem there. And this N only appears in Chapter 5 and links to its definition as the number of nodes in a decentralized network. This feature is best enjoyed with a reader that can go back to the place where a link was clicked after having clicked on it. See this post for a list of common readers and shortcuts with such function.

have been numerous efforts towards obtaining intuitive and generalizable schemes for the design of accelerated methods, that often achieve optimality. In particular, the smooth convex case and AGD have been subjected to scrutiny in the last fifteen years. Many works provide different points of view and interpretations of this algorithm, some provide generalizations, and a few obtain other different accelerated algorithms for this setting as a result of their more general frameworks. Most existing acceleration techniques, applying to all sort of problems[1], draw ideas from the different interpretations of acceleration for the smooth convex case or from general techniques that recover this case. Because of this, and because a key ingredient of the three main results in this thesis is acceleration, we provide an overview of some acceleration techniques and algorithms in Chapter 2, focusing on the optimization of a smooth convex function f with a first-order oracle. This overview provides context and intuition for subsequent chapters. We refer to these chapters for slow-building introductions and definitions in these topics while we reserve the rest of this introduction for the more experienced reader.

The key to acceleration is often the effective conjugation of a method that builds good lower bounds on f (it can be seen as a dual method in some settings, like in smooth convex optimization), with some other properties the function satisfies, such as smoothness, finite-sum structure, etc. An important technique to obtain good lower bound estimations consists of a reduction to *online learning*. In our overview, we present part of this framework along with the two main algorithmic schemes for solving online learning, namely the Follow the Regularized Leader (FTRL) approach and Online Mirror Descent (OMD). We point out some relationships between them when they are applied to convex optimization, and we provide intuition through some different equivalent formulations. In particular we show how Mirror Descent (MD) creates a looser regularized lower bound than FTRL. It is worth noticing that this does not mean that the estimation on the optimum of f is looser, since the looseness can come from reducing the regularizer, and this case is beneficial. After that, we present some acceleration methods in the smooth convex setting, with its technical details, comments on intuition, historical relationships between works, and we point out some technical relationships between some different acceleration methods. Given the intended scope of this dissertation, the acceleration overview is necessarily incomplete. The literature is vast, but our aim with this presentation is to provide adequate intuition for the proofs and technicalities we present in subsequent chapters, and to summarize and connect several useful related optimization techniques. In Chapter 2, we will overview in detail some of the algorithms mentioned below.

The Conjugate Gradient Descent (CGD) method and the use of Chebyshev polynomials are the first accelerated algorithms in convex optimization. They apply to the optimization of a quadratic. CGD and a particular analysis of it, was the inspiration for the first near-optimal

[1]Some first-order examples are: composite optimization with a smooth convex function plus a non-smooth simple term, (Beck and Teboulle, 2009; Tseng, 2008), stochastic smooth convex optimization where the oracle returns an estimate of the gradient (Xiao, 2010; Lan, 2012; Kavis et al., 2019; Joulani, Raj, et al., 2020), finite-sum smooth convex optimization (Lin, Mairal, and Harchaoui, 2015; Allen-Zhu, 2017a), and finding stationary points in functions with Lipschitz continuous gradient and Hessian (Carmon, Duchi, et al., 2017).

first-order method for smooth convex optimization (Nemirovski and Yudin, 1983a). The method is imbued with geometrical intuition, and is optimal only up to log factors because it required an approximate plane search per iteration. It was later refined to a line search, by using an algebraic trick, that detached in certain degree the geometrical point of view from the analysis. Inspired by this last work, Nesterov (1983) designed the first fully accelerated algorithm that obtained optimal rates up to constants, and provided some generalizations. An algebraic trick was used to show that one can specify, without searches, a good enough point in the line in which the previous method by Nemirovski and Yudin (1983a) was searching. The geometric intuition is not present either in this solution, or at least not in a simple way. But later, the same author developed a more intuitive analysis (Nesterov, 1998; Nesterov, 2005), the technique of *estimate sequences*, which essentially uses an FTRL algorithm to estimate a lower bound on the function and conjugates it with the smoothness property to rapidly decrease the optimality gap. This point of view also implies that one can obtain acceleration by using a particular point in the line where the method by Nemirovski and Yudin (1983a) searches, and such point can be computed a priori and without searches. This method (Nesterov, 2005) was developed before the FTRL algorithmic scheme was really formalized in its full generality in online learning (Shalev-Shwartz and Singer, 2006; Shalev-Shwartz and Singer, 2007; Abernethy, Hazan, and Rakhlin, 2008; Hazan and Kale, 2008), and one can consider this sequence-estimation technique as the first algorithm using FTRL, also called Nesterov's Mirror Descent for evident reasons. The method presented in (Nesterov, 1998) is similar but it uses MD instead of FTRL. Tseng (2008) provided a unified presentation of such techniques and produced some generalizations. Allen-Zhu and Orecchia (2017) presented an intuitive view of the method that used MD as a method that couples a GD iterate with the MD subalgorithm so that the progress of the former balances the regret of the latter. Later, Diakonikolas and Orecchia (2019b); Diakonikolas and Orecchia (2018) model the optimization algorithm as a continuous method that is naturally defined to reduce the optimality gap. It results in a differential equation, that they can discretize to recover AGD if one uses a forward Euler discretization along with a GD step to make the discretization error be non-positive. The same differential equation was first obtained by Su, Boyd, and Candès (2016) and also studied in (Krichene, Bayen, and Bartlett, 2015; Wibisono, Wilson, and Jordan, 2016). But importantly Diakonikolas and Orecchia (2018) also provide a different and novel accelerated method, named Accelerated eXtra Gradient Descent (AXGD), by discretizing the differential equation with an implicit Euler discretization that uses one fixed point iteration to approximate the implicit equation. Monteiro and Svaiter (2013) provide a general accelerated framework that uses the *proximal operator* of f or an inexact implementation of it. Several works used this framework to obtain accelerated algorithms by using different inexactness measures, different solutions of the subproblem, and different learning rates, cf. (Lin, Mairal, and Harchaoui, 2015, Catalyst) and (Carmon, Jambulapati, et al., 2020) as examples. A common choice for solving the proximal subproblem, that is strongly convex, are (possibly accelerated) first-order methods for such setting.

There are other techniques and points of view on acceleration for smooth convex functions, that recover one of the previous algorithmic schemes but with different proofs, intuition and different capabilities of generalizing to other settings (Bubeck, Lee, and Singh, 2015; Joulani, Raj, et al., 2020; Wibisono, Wilson, and Jordan, 2016), to name a few. Notably, Kim and Fessler (2016) found that a change on the learning rate of AGD decreases the worse-case guarantee by a constant, and it was later shown that this is optimal, even considering constants (Drori, 2017). Both the upper and lower bounds use an interesting technique that casts the design of the algorithm with best worse-case guarantees within a family as an optimization problem, whose dual provides a hard instance. The optimization problem is hard a priori but an SDP relaxation allows for finding constant-matching upper and lower bounds on the worse-case performance with respect to several metrics (Drori and Teboulle, 2014; Kim and Fessler, 2016; Taylor, Hendrickx, and Glineur, 2017; Drori, 2017). Joulani, Raj, et al. (2020) show that applying an optimistic anytime online learning algorithm[2] with guess equal to the previously computed gradient recovers AGD and the rates come simply from the online learning guarantee. A different line of work (Scieur, d'Aspremont, and Bach, 2020; Scieur, Bach, and d'Aspremont, 2017; d'Aspremont, Scieur, and Taylor, 2021), applying to smooth and strongly convex unconstrained optimization, extrapolates a point from the sequence of iterates of any optimization algorithm in order to provide improved solutions.

1.1 Main contributions

The accelerated techniques we overview in Chapter 2 are very useful tools in optimization. The unconstrained black-box first-order smooth and convex case is essentially solved, but the main capability of this understanding and of different points of view is on their applications to new problems, by both finding problems for which we can use these fast methods as subalgorithms, and also by applying these techniques to obtain acceleration in other problem settings. The results in this dissertation take these two directions. In Chapter 5, we do the former and model the decentralized solution for our bandit problem so that we can apply an accelerated gossip communication protocol, a particular case of accelerated convex optimization. In Chapter 3 and in Chapter 4 we do the latter and obtain new accelerated algorithms as part of our solutions, with an accelerated algorithm for a class of constrained non-convex functions and another one for a non-globally smooth problem presenting bad local smoothness but with some structure that can be exploited, respectively. We focus on first-order methods. We note that in the problems in Chapter 4 and Chapter 5, the use of first-order information only is essentially a requirement. This is due to the distributed and decentralized models of computation that we use, respectively, and that are motivated from their applications. In Chapter 3 we study the acceleration phenomenon in first-order Riemannian optimization, motivated by the large-scale setting, cf. Section 3.1.

[2]Optimism refers to a technique that consists of taking a guess on the next loss in such a way that we receive lower regret if the guess was close to the actual loss. An anytime algorithm is one whose last iterate enjoys guarantees, as opposed to the mean of the iterates or other constructions.

Applications and generalizations of accelerated methods are very promising directions of study, which have been the focus of the research in this thesis. The goal of Chapter 2 is to overview some preexisting acceleration techniques in order to give intuition about them and to provide context for our results in following chapters. We have curated, for our purposes, the accelerated techniques that are presented. We added intuitive comments along the chapter and included some relationship between methods that help with intuition and that we have not seen in the literature, like the one in Section 2.6.1. In any case, we do not claim novelty on the content of Chapter 2. Everything we added may be known, and most ideas are added in order to provide intuition more than anything else.

Chapter 3 is based on the work (Martínez-Rubio, 2020), that I developed. Chapter 4 is based on (Criado, Martínez-Rubio, and Pokutta, 2021). This work contains an algorithm for optimizing a primal problem and an algorithm for the optimization of its dual problem. The primal algorithm was primarily done by me and the majority of the dual solution was done by Francisco Criado. We have not included the latter in this thesis. The research in this work was supervised by Sebastian Pokutta. Chapter 5 is based on the work (Martínez-Rubio, Kanade, and Rebeschini, 2019) done by me and supervised by Varun Kanade and Patrick Rebeschini. The chapter contains some extensions as well. In the sequel, we summarize the main technical contributions of our results in Chapters 3, 4 and 5. We refer to the corresponding chapters for references of any statement made in this summary.

Riemannian optimization

In Chapter 3, we study Riemannian optimization, which is a field of optimization that considers functions defined on Riemannian manifolds, a setting that has found many applications in machine learning. In particular, we study the design of accelerated first-order optimization algorithms that apply to smooth functions that are defined on Riemannian manifolds and that are convex, (resp. strongly convex) when restricted to any geodesic, that is, they are geodesically convex (resp. strongly geodesically convex) functions. The two previous approaches that provided provable guarantees before our work (Zhang and Sra, 2018; Ahn and Sra, 2020) could essentially obtain the same rates as AGD, up to constants, for strongly geodesically-convex functions defined on Riemannian manifolds of bounded sectional curvature provided that the initial solution starts in a small neighborhood of the minimizer, that shrinks with the condition number. We design an algorithm that optimizes over manifolds of constant sectional curvature, which is a more restricted setting consisting of hyperbolic and spherical spaces, but that obtains *globally* the same rates as AGD, up to constants and log factors. Besides, no previous algorithm achieved acceleration in the geodesically-convex case only. The neighborhood of previous algorithms becomes a single point in such a case, since it is the limit of the condition number tending to infinity. The constant sectional curvature is an important case since many properties and algorithms usually rely on the constant curvature case and use comparison geometry theorems to obtain the bounded curvature results as an interpolation between the properties and algorithms

of the two extremal constant curvature cases. Extending our framework to bounded curvature manifolds is a future direction of research. Studying if the constants appearing in the rates with respect to the initial distance to a minimizer are necessary for full global acceleration is also of interest and a future direction of research. Here, by full acceleration we refer to achieving the same rates as AGD with respect to the desired accuracy, to the smoothness constant, and if it applies, to the strong convexity constant. We also provide reductions from optimization methods for smooth geodesically-convex functions to smooth strongly geodesically-convex functions and vice versa. They are adaptations of Euclidean reductions. In particular reducing to strongly geodesically-convex does not incur in extra log factors. Some of our changes of the latter reduction could be used in the optimal Euclidean reduction by Allen-Zhu and Hazan (2016) to slightly reduce their constants.

Our solution can be seen from two points of view. One can see the algorithm as performing FTRL on surrogate losses whose regularized sum can be easily optimized, and then conjugating the corresponding lower bound with a gradient descent step in order to achieve acceleration. As usual, the losses come from gradients $\nabla f(x_i)$, and in this case they define simple lower bounds on the function (e.g. affine in the geodesically-convex case) but only when viewed from the tangent space of x_i via the inverse exponential map, respectively. The surrogates consist of bounds that are looser by a constant factor and that are simple (e.g. affine) and all of them are defined on the same tangent space, which allows for the global optimization subproblem of FTRL. The second point of view regards the optimization as first performing a reduction to a Euclidean *constrained* non-convex problem that we optimize in an accelerated way. The accelerated optimization presents some problems for the usual approaches: On the one hand we need the constraints to guarantee bounded deformation due to the geometry but on the other hand, since the surrogates cause the regret to be a factor greater than in the convex case, the gradient step used to balance the regret would *want* to use a greater learning rate by the same factor, landing at a point that is not guaranteed to be inside the constraints and not allowing for constrained optimization. We find that by using a backward Euler discretization of the continuous accelerated dynamics, similarly to AXGD, and making use of a binary search per iteration, we can optimize this constrained non-convex problem in an accelerated way.

Proportional fairness

In Chapter 4, we study the proportional fair allocation problem under positive linear constraints, i.e.,

$$\max\left\{\sum_{i=1}^{n}\log(x_i) : Ax \leq \mathbb{1}_m, x \in \mathbb{R}^n_{\geq 0}, A \in \mathcal{M}_{m\times n}(\mathbb{R}_{\geq 0})\right\}.$$

Proportional fairness is a fairness scheme for resource allocation which considers something is unfair if a small transfer of resources between some pair of players results in a proportional increase in the utility of one player larger than the proportional decrease in the utility of the other player. The fair allocation can be computed as the solution of $\max\{\sum_i \log(x_i)\}$. This allocation

is studied and applied in several fields such as rate control of networks, resource allocation in clusters, game theory and economic theory and it is the unique allocation satisfying a natural set of fairness axioms (Bertsimas, Farias, and Trichakis, 2011; Lan et al., 2010). Many of these applications naturally require to obtain the proportional fair solution under positive linear constraints. Our objective is part of a class of well-known resource allocation schemes, known as α-fair solutions for which linear programming (LP) corresponds to $\alpha = 0$ (no fairness at all, it just maximizes utility) and proportional fairness corresponds to $\alpha = 1$. Because of this reason, the problem is also known as the 1-fair packing problem. We design an algorithm that works in a standard distributed model of computation, which is a requirement for some applications. We generalize ideas from one of the currently fastest solutions for large scale packing linear programming (Allen-Zhu and Orecchia, 2019), like taking a non-standard instance of mirror descent by using truncated gradients as losses and linearly coupling it with a gradient descent condition. This would be applied to a regularized objective in order to obtain an accelerated algorithm. This objective is still non-globally smooth and presents bad local smoothness, but presents some smoothness-like structure that can be exploited. This allows us to prove convergence, in $\widetilde{O}(n/\varepsilon)$ iterations, to a solution with additive error ε (note packing LP solutions are designed for guaranteeing multiplicative errors). The previous best solution (Diakonikolas, Fazel, and Orecchia, 2020) did not achieve acceleration and converged with rate $\widetilde{O}(n^2/\varepsilon^2)$, where $\widetilde{O}(\cdot)$ hides logarithmic factors on m, n, ε and, in the latter case, on the width of the matrix as well. The width of the matrix A, is defined as $\frac{\max_{ij} A_{ij}}{\min_{ij}\{A_{ij}:A_{ij}\neq 0\}}$ and can be exponential on the input. Some previous algorithms depended on the width, yielding non-polynomial algorithms. Some other algorithms are polynomial (Marašević, Stein, and Zussman, 2016; Diakonikolas, Fazel, and Orecchia, 2020) and present a logarithmic dependency on the width in the convergence rates. We note our convergence rate does not depend on the width at all. Also, remarkably, our algorithm is deterministic and enjoys deterministic guarantees, as opposed to the solutions in packing LP in (Allen-Zhu and Orecchia, 2019).

Decentralized cooperative bandits

Finally, in Chapter 5 we study a decentralized cooperative stochastic multi-armed bandit problem. That is, we have K actions and N agents in a network and we play a repeated game in which at every iteration, each agent chooses an action and receives a reward that is sampled from a stationary distribution that depends on the action. In our problem, the samples are independent across agents and iterations, as opposed to models that assume that there is some kind of reward collision if two agents choose the same action. Then, agents can synchronously communicate values to their neighbors. After that, the next iteration starts. The aim is to minimize the expected regret after T iterations, i.e., the difference between what we could have obtained if all agents had always chosen an action with optimal mean reward compared to what we obtain with our policy, in expectation. The agents run the same decentralized algorithm we

decide to implement and in particular they have little access to global quantities. They cooperate to minimize the expected regret.

Bandit algorithms are used in many decision tasks such as in recommendation algorithms, and anything going beyond A/B testing that needs to consider a large number of options. Some applications require to be run on decentralized networks and in some other cases we may decide to use a decentralized algorithm in order to obtain robustness to changes in the network or failures in communication links, cf. Chapter 5.

In our solution, we use a classical decentralized communication protocol, namely a gossip algorithm, in order to obtain an approximate average of the rewards' estimations from each agent and to approximate the number of times each action was chosen. Then, we generalize the classical Upper Confidence Bound (UCB) algorithm, that is an algorithm that applies to the stochastic centralized multi-armed bandit problem. Our generalization analyzes the regret incurred by working with approximate and delayed mean rewards and number of actions chosen. Our problem was previously studied by (Landgren, Srivastava, and Leonard, 2016; Landgren, Srivastava, and Leonard, 2019a) and they used another variant of UCB with a natural approach for approximating the same quantities: a *running* unaccelerated gossip protocol, meaning that the algorithms distribute and use the information as soon as it is available. This approach obtains worse estimations due to newly received information that has not had time to average in the network. We introduce a delay in the use of the mean rewards and do not use very recent information with the aim of trading off some delay for better approximations. The delay does not only help in obtaining better estimators but it allows us to average information in batches, providing more flexibility in the gossip protocol. In particular, in this case we can apply an accelerated gossip protocol that reduces the time required to obtain a good enough approximate average, which further decreases the regret.

We show our regret bounds improve over those of previous works, we provide examples of networks that exemplify the difference between the regret bounds, and we also perform experiments demonstrating that our algorithms perform better empirically.

Chapter 2

Acceleration in Optimization Methods

In this section, we provide a non-exhaustive overview of some acceleration methods and techniques. This provides context for the rest of the chapters. We sometimes go over different points of view of the same Accelerated Gradient Descent (AGD) algorithm for smooth convex functions. We also explain some of the history of the acceleration phenomenon. These methods, except for the first one we explain, enjoy a rate of convergence of $O(\sqrt{LR/\varepsilon})$, matching the known lower bound up to constants. We are denoting the smoothness constant by L, R denotes the initial distance to a minimizer and ε represents the target accuracy.

The first lower bounds for first-order smooth convex optimization of a smooth convex function f were given in (Nemirovski and Yudin, 1983a) for the natural class of algorithms in which the next iterate is $x_k \in x_0 + \text{span}\{\nabla f(x_0), \dots, \nabla f(x_{k-1})\}$ and the authors show that any method of the class must query a gradient oracle $\Omega(\sqrt{LR/\varepsilon})$ times in order to achieve a point with ε accuracy. Drori (2017) improves this lower bound by constants, showing that the optimized gradient method of Kim and Fessler (2016) is worse-case optimal in unconstrained smooth convex optimization, even considering constants. These lower bounds assume the dimension is greater than the number of iterations. Arjevani and Shamir (2016) proved a dimension-independent lower bound of $\widetilde{\Omega}(\sqrt{LR/\varepsilon})$ for a rich family of algorithms that satisfy a stationarity condition on the update step (that includes fixed-step AGD, the heavy-ball method, coordinate descent, quasi-Newton methods, ellipsoid method, SVRG, SAGA, among many others). Here $\widetilde{\Omega}(\cdot)$ hides logarithmic factors. These lower bounds assume a black-box access to the computation of the gradient.

Arguments involving potential functions are useful for proving rates of convergence for these kinds of algorithms. We will interpret each of these algorithms' steps as online algorithms mixed with a GD step whose progress balances the instantaneous regret. Further, we will interpret these analyses as coming naturally from a definition of duality gap and the imposition on decreasing it at a certain rate.

2.1 Preliminaries

We start by defining a few basic convex optimization notions and some useful properties. We will build the rest of the chapter on these notions.

Definition 2.1.1 (Convex set). *A set $\mathcal{X} \subseteq \mathbb{R}^n$ is convex if any segment with endpoints in the set is contained in the set. That is, for any $x, y \in \mathcal{X}$ and $\lambda \in [0,1]$ we have $\lambda x + (1-\lambda)y \in \mathcal{X}$.*

Definition 2.1.2 (Convex function). *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a convex set. A function $f : \mathcal{X} \to \mathbb{R}$ is convex if its epigraph $\{(x, y) : x \in \mathcal{X},\ y \geq f(x)\}$ is convex. An equivalent condition is:*

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y) \text{ for } x, y \in \mathcal{X}\ \lambda \in [0,1].$$

Note that we require $\mathcal{X}$ be convex because otherwise f cannot be convex according to the first definition. The second definition does not even make sense if $\mathcal{X}$ is not convex. If f is differentiable, convexity is equivalent to $f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle$ for all $x, y \in \mathcal{X}$ and if f is twice differentiable convexity is also equivalent to having positive semidefinite Hessian everywhere, that is, $\nabla^2 f(x) \succcurlyeq 0$ for all $x \in \mathcal{X}$.

Definition 2.1.3 (Subdifferential). *For a convex function $f : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}$, the set of subdifferentials at a point $x \in \mathcal{X}$, denoted by $\partial f(x)$, is the set of vectors $g \in \mathbb{R}^n$ that satisfy $f(y) \geq f(x) + \langle g, y - x \rangle$ for all $y \in \mathcal{X}$.*

We note that if f is differentiable and $x \in \operatorname{int}(\mathcal{X})$ then $\partial f(x) = \{\nabla f(x)\}$.

Definition 2.1.4 (Smoothness and strong convexity). *Let $f : \mathcal{X} \to \mathbb{R}$ be a convex and differentiable function. We say f is L-smooth and, respectively, μ-strongly convex with respect to $\|\cdot\|$ if for $L \geq \mu > 0$ we have*

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|x - y\|^2,\ \textit{resp.}\ f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2}\|x - y\|^2,$$

where $\|\cdot\|$ is an arbitrary norm.

In this text we assume it is $\|\cdot\| = \|\cdot\|_2$ if the norm is not specified. If we use the 2-norm and f is twice differentiable these two conditions are equivalent to $\mu I \preccurlyeq \nabla^2 f(x) \preccurlyeq LI$ for all $x \in \mathcal{X}$. Under convexity, smoothness is also equivalent to gradient Lipschitzness: $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$ but we note that if f is not convex then gradient Lipschitzness implies smoothness but not the other way around. One can prove that the bounds on eigenvalues of the Hessian imply the inequalities in the definition by taking a second order Taylor approximation and bounding the Lagrange remainder. Even if the function is not twice differentiable this fact provides a good way to recall the definition of the inequalities. Note that if $\mu = 0$ we would have regular convexity.

Notation. From now on in this chapter, we denote by $\mathcal{X} \subseteq \mathbb{R}^n$ a closed and convex set. If we say a function $g : \mathcal{X} \to \mathbb{R}$ is a differentiable function we mean there is a differentiable extension $\widehat{g} : X \to \mathbb{R}$ of g for a convex open set X such that $\mathcal{X} \subseteq X$ and we fix an arbitrary extension so we can define $\nabla g(x) \stackrel{\text{def}}{=} \nabla \widehat{g}(x)$ for $x \notin \operatorname{int}(\mathcal{X})$. For a number $k \in \mathbb{Z}^+$, we use the notation $[k] \stackrel{\text{def}}{=} \{1, 2, \ldots, k\}$. We denote by T the number of steps our algorithms run for. We

use $\|\cdot\|_*$ for the dual norm of $\|\cdot\|$, that is, $\|v\|_* \stackrel{\text{def}}{=} \max_{\|w\|\leq 1}\langle v, w\rangle$. We denote by f an L-smooth convex function with at least one minimizer x^*. Our aim in convex optimization is to find an ε-minimizer of f, which is a point x that satisfies $f(x) - f(x^*) \leq \varepsilon$. We usually denote strongly convex regularizers by ψ.

We define the Fenchel dual, which is a very important dual of a function in convex optimization.

Definition 2.1.5 (Fenchel dual). *For a function $\psi : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}$ we define $\psi^* : \mathbb{R}^n \to \mathbb{R}$ as*

$$\psi^*(z) = \min_{x\in\mathcal{X}}\{-\langle z, x\rangle + \psi(x)\}.$$

We call this function the Fenchel dual of ψ or its convex conjugate.

Fact 2.1.6. *Let $\psi : \mathcal{X} \to \mathbb{R}$ be a differentiable strongly-convex function. Then*

$$\nabla\psi^*(z) = \arg\min_{x\in\mathcal{X}}\{\langle -z, x\rangle + \psi(x)\}.$$

See (Bertsekas, Nedic, and Ozdaglar, 2003) for a proof. We call $\mathcal{D} \stackrel{\text{def}}{=} \nabla\psi(\mathcal{X})$ the dual space of $\mathcal{X}$ via ψ. From the first-order optimality condition of Fact 2.1.6 we can deduce that $\nabla\psi^* \circ \nabla\psi = \mathrm{Id}_{\mathcal{X}}$. On the other hand it holds that $\nabla\psi(\nabla\psi^*(z)) = z$ if $z \in \mathcal{D}$ but this is not the case if $z \notin \mathcal{D}$, since in particular $\nabla\psi(\nabla\psi^*(z)) \in \mathcal{D}$ by definition. Consequently, we have $\nabla\psi \circ \nabla\psi^* = \mathrm{Id}$ if and only if $\nabla\psi(\mathcal{X}) = \mathbb{R}^n$. This happens if $\mathcal{X} = \mathbb{R}^n$ or more in general, if we allow $\mathcal{X}$ not to be closed we can have this property if $\lim_{x\to\partial\mathcal{X}} \|\nabla\psi(x)\|_2 = +\infty$.

Oftentimes in optimization we regularize the objective. This means, we add a function ψ with the aim of biasing an algorithm to satisfy some particular properties. In first-order methods, it is common to regularize the objective with a differentiable and strongly convex function in order to make the minimizer closer to $\arg\min\{\psi(x)\}$, similarly to what we prove in Lemma 3.3.5. This regularization is also useful if we are optimizing related subproblems and want to give some stability to their solutions. We will make this fact precise in Section 2.2. Further, regularization will not hurt the optimization process much if the penalty at a minimizer of the original function, x^*, is of the order of the accuracy ε we want to achieve, i.e., if $\psi(x^*) - \min\psi(x) \leq O(\varepsilon)$. Sometimes we want to dynamically change the point of attraction. That is, we would like to change where we are biasing our iterates toward, maybe because with time we gain better understanding of around where a solution could be. Sometimes we have a regularizer that satisfies good properties, and we would like to have its optimizer at a different point while keeping other properties it may have. For both of these tasks, Bregman divergences are a useful tool, specially if the regularizers are strongly convex.

The idea is very simple. Suppose we want the minimizer of our regularizer ψ to be at a point y. Then, we can simply use $\psi(x) - \langle\nabla\psi(y), x\rangle$ in order to have 0 gradient at y. Further, y will be the only such point if ψ is strongly convex. Also, adding a constant to the regularizer usually does not make a difference to the algorithm because we are interested in where the optimizer of the subproblems is. That is, the penalties the regularizer produces, are relative to the minimum of the

regularizer, like $\psi(x^*) - \min \psi(x)$ above. So we can make the regularizer at y be 0 for simplicity. In this way, we end up with the definition of Bregman divergence $D_\psi(x, y)$ as this regularizer, with y as base point. We present the formal definition and some useful properties of Bregman divergences below. When using Bregman divergences with Mirror Descent (cf. Section 2.2.2) we will have errors depending on them. In order to provide more intuition about this, we will present, in Section 2.6, an equivalent and alternative point of view that shows that these errors can be computed as discretization errors of a continuous method, where one has to integrate property 3 below. This intuitively means that this is the error incurred by changing the dual point since, recall, $\nabla\psi(\cdot)$ maps a point to the dual space.

Definition 2.1.7 (Bregman Divergence). *Let $\psi : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}$ be a differentiable σ-strongly convex function with respect to an arbitrary norm $\|\cdot\|$. We define the Bregman divergence as*

$$D_\psi(x, y) \stackrel{\text{def}}{=} \psi(x) - \psi(y) - \langle \nabla\psi(y), x - y\rangle \text{ for } x, y \in \mathcal{X}.$$

We list and prove some useful properties of the Bregman divergence.

Lemma 2.1.8. *The Bregman divergence satisfies the following properties:*

1. *Suppose ψ is twice differentiable. Then $D_\psi(x, y) = \frac{1}{2}(x - y)^T \nabla^2\psi(z)(x - y)$ for a point z in the segment between x and y.*
2. *$D_\psi(x, y) \geq \frac{\sigma}{2}\|x - y\|^2 \geq 0$ and $D_\psi(x, x) = 0$.*
3. *$\nabla_x D_\psi(x, y) = \nabla\psi(x) - \nabla\psi(y)$.*
4. *$D_\psi(\cdot, y)$ is strongly convex and its Bregman divergence $D_{D_\psi(\cdot,y)}(\cdot, y)$ is itself. More generally: $D_{D_\psi(\cdot,z)}(\cdot, y) \equiv D_\psi(\cdot, y)$ for all y, z.*
5. *$D_\psi(x, y) = D_\psi(z, y) + D_\psi(x, z) + \langle \nabla\psi(z) - \nabla\psi(y), x - z\rangle$, for all $x, y, z \in \text{int}(\mathcal{X})$. This is sometimes called the triangle equality of Bregman divergences.*
6. *$D_{\psi^*}(x, y) \geq D_\psi(\nabla\psi^*(y), \nabla\psi^*(x))$. When $\nabla\psi(\nabla\psi^*(x)) = x$, we have equality.*

Proof

1. It is a consequence of Taylor's theorem. This fact provides the intuition about the Bregman divergence as a local norm, which is a property that can be exploited in online learning.
2. The first part follows straightforwardly from the definition of the strong convexity of ψ and the second part follows from the definition of Bregman divergence.
3. This follows by differentiating the definition of Bregman divergence:

$$\psi(x) - \psi(y) - \langle \nabla\psi(y), x - y\rangle.$$

4. $D_\psi(\cdot, y)$ is strongly convex because it is the sum of the strongly convex function $\psi(\cdot)$ plus a linear term. By using part 3., we have

$$\begin{aligned} D_{D_\psi(\cdot,z)}(x,y) &= D_\psi(x,z) - D_\psi(y,z) - \langle \nabla_w D_\psi(w,z)|_{w=y}, x-y\rangle \\ &= (\psi(x) - \psi(z) - \langle \nabla\psi(z), x-z\rangle) - (\psi(y) - \psi(z) - \langle \nabla\psi(z), y-z\rangle) \\ &\quad - \langle \nabla\psi(y) - \nabla\psi(z), x-y\rangle \\ &= \psi(x) - \psi(y) - \langle \nabla\psi(z), x-z-y+z\rangle - \langle \nabla\psi(y) - \nabla\psi(z), x-y\rangle \\ &= D_\psi(x,y). \end{aligned}$$

This result should be intuitive given the introduction we made of Bregman divergences. Since the zeroth order information is always set to 0 and since for strongly convex ψ there is a unique way of changing the first-order information so the optimizer is at y, it should be intuitive that taking the Bregman divergence with respect to $D_\psi(\cdot, z)$ for any z yields the same result as taking the Bregman divergence with respect to ψ, because it is the same function but with different zeroth and first-order information.

5. Expanding the definitions we obtain:

$$\begin{aligned} &D_\psi(z,y) + D_\psi(x,z) + \langle \nabla\psi(z) - \nabla\psi(y), x-z\rangle \\ &\quad = (\psi(z) - \psi(y) - \langle \nabla\psi(y), z-y\rangle) \\ &\qquad + (\psi(x) - \psi(z) - \langle \nabla\psi(z), x-z\rangle) + \langle \nabla\psi(z) - \nabla\psi(y), x-z\rangle \\ &\quad = \psi(x) - \psi(y) - \langle \nabla\psi(y), z-y\rangle + \langle -\nabla\psi(y), x-z\rangle = D_\psi(x,y). \end{aligned}$$

6.

$$\begin{aligned} D_{\psi^*}(x,y) &= \psi^*(x) - \psi^*(y) - \langle \nabla\psi^*(y), x-y\rangle \\ &\overset{①}{=} \psi(\nabla\psi^*(y)) - \psi(\nabla\psi^*(x)) - \langle x, \nabla\psi^*(y) - \nabla\psi^*(x)\rangle \\ &= D_\psi(\nabla\psi^*(y), \nabla\psi^*(x)) + \langle \nabla\psi(\nabla\psi^*(x)) - x, \nabla\psi^*(y) - \nabla\psi^*(x)\rangle \\ &\overset{②}{\geq} D_\psi(\nabla\psi^*(y), \nabla\psi^*(x)) \end{aligned}$$

In ①, we used $\psi^*(x) = \langle \nabla\psi^*(x), x\rangle - \psi(\nabla\psi^*(x))\ \forall x$ which holds by Fact 2.1.6. In ② we used first-order optimality condition of the Fenchel dual definition at $\nabla\psi^*(y) \in \mathcal{X}$, which is also derived from Fact 2.1.6. That is, the gradient of the minimization problem at the optimum $\nabla\psi^*(x)$ is $\nabla\psi(\nabla\psi^*(x)) - x$ and since the problem is of minimization, this gradient non-negatively correlates with the segment $z - \nabla\psi^*(x)$ for any $z \in \mathcal{X}$. The second part of 6. is trivial.

■

We now proceed to overview some online learning algorithms which are building blocks of accelerated methods.

2.2 Online learning: FTRL and Mirror Descent

An online learning algorithm in the full information setting is an algorithm that plays a repeated game of the following form. Given a set $\mathcal{X} \subseteq \mathbb{R}^n$, and round $t \in [T]$, an adversary chooses a loss function $\ell_t : \mathcal{X} \to \mathbb{R}$, that can depend on the algorithm and previous decisions taken by it. But it cannot predict the outcome of a randomized decision.[1] After the adversary selects the loss ℓ_t, the algorithm chooses a prediction point $x_t \in \mathcal{X}$, observes the loss function ℓ_t and pays the loss $\ell_t(x_t)$. The aim is to minimize the regret, defined as the difference between our losses and the loss we could have obtained with a fixed action, had we known the sequence of loss functions in advance:

$$R_T \stackrel{\text{def}}{=} \sum_{t=1}^{T} \ell_t(x_t) - \min_{u \in \mathcal{X}} \sum_{t=1}^{T} \ell_t(u), \tag{2.2.1}$$

A guarantee would bound the regret regardless of the choices made by the adversary. Even though the sequence of losses can be chosen against our algorithms, the optimizers of $\min_u \sum_{t=1}^{k} \ell_t$ and $\min_u \sum_{t=1}^{k+1} \ell_t$ cannot differ arbitrarily if we make some assumptions on ℓ_t, and algorithms will exploit this fact in order to obtain reasonable regret.

The full information setting is a very general problem, but we note there are many other variants of this game. Another example is the bandit setting, in which one only observes the value $\ell_t(x_t)$ as opposed to the function ℓ_t. For the ideas we convey in this section, the full information setting with convex losses ℓ_t and closed convex sets $\mathcal{X}$ will be enough. But in fact, in Chapter 5, we will study an online learning problem in the bandit setting in which the losses[2] will be sampled from unknown distributions instead of being picked by an adversary, which is another online learning problem.

Online learning is a useful tool for optimization. In particular, for convex optimization we can reduce many problems to online learning. For instance, if we have a convex function $f : \mathcal{X} \to \mathbb{R}$ we want to optimize via a first-order oracle and a deterministic algorithm, we can define an online learning game in which the adversary chooses linear losses $\ell_t(\cdot) = \langle \nabla f(x_t), \cdot \rangle$[3]. Suppose f has at least one minimizer $x^* \in \mathcal{X}$ and define $\bar{x} \stackrel{\text{def}}{=} \frac{1}{T} \sum_{t=1}^{T} x_t \in \mathcal{X}$. Then, we have

$$f(\bar{x}) - f(x^*) \stackrel{①}{\leq} \frac{1}{T} \sum_{t=1}^{T} (f(x_t) - f(x^*)) \stackrel{②}{\leq} \frac{1}{T} \sum_{t=1}^{T} \langle \nabla f(x_t), x_t - x^* \rangle \leq \frac{R_T}{T}, \tag{2.2.2}$$

where ① uses Jensen's inequality and ② uses the definition of convexity. More generally, we could have also defined losses $\ell_t(\cdot) = a_t \langle \nabla f(x_t), \cdot \rangle$ and $\bar{x}_a \stackrel{\text{def}}{=} \frac{\sum_{t=1}^{T} a_t x_t}{\sum_{t=1}^{T} a_t} \in \mathcal{X}$, for $a_t > 0$, to conclude $f(\bar{x}_a) - f(x^*) \leq R_T / (\sum_{t=1}^{T} a_t)$. In even more generality, if we have functions ℓ_t that

[1] In any case, we limit the technical exposition in this section to deterministic algorithms and will only comment on the intuition about why one can generalize some of these ideas to algorithms that use randomness or that access a function through a stochastic oracle.

[2] Note that, following the standard approach, we use rewards instead of losses for the setting in Chapter 5.

[3] Assume for simplicity that f is differentiable, the same argument on this subsection can be done by using subgradients.

lower bound our function f, and such that $f(x_t) = \ell_t(x_t)$, we could use them as losses of an online learning problem, weighted by a_t. In such a case we obtain

$$f(\bar{x}_a) - f(x^*) \leq \frac{1}{\sum_{t=1}^T a_t} \sum_{t=1}^T a_t(\ell_t(x_t) - \ell_t(x^*)) \leq \frac{R_T}{\sum_{t=1}^T a_t}. \tag{2.2.3}$$

The case in Equation (2.2.2) is essentially equivalent to taking losses $\ell_t(x) = f(x_t) + \langle \nabla f(x_t), x - x_t \rangle$, since adding constants to the losses (like $f(x_t) - \langle \nabla f(x_t), x_t \rangle$ in this case) does not change the regret if we play the same points. It is not exactly equivalent because an algorithm could do different things if it observes different loss functions. As another example, if we further assume that f is μ-strongly convex, we could make the losses be $\ell_t(x) = \langle \nabla f(x_t), x \rangle + \frac{\mu}{2}\|x - x_t\|_2^2$, weighted by a_t, to conclude that $f(\bar{x}_a) \leq R_T/(\sum_{t=1}^T a_t)$ with $\bar{x}_a$ is defined as above and R_T is the corresponding regret. Note that in fact we only needed $f(x_t) \leq \ell_t(x_t)$ and $\ell_t(x^*) \leq f(x^*)$ in (2.2.3). If one can set $\ell_t(x_t) = f(x_t)$, the bound is tighter than in the case $\ell_t(x_t) > f(x_t)$, so such an approach is preferred. Also, most of the time one uses $\ell_t(x) \leq f(x)$ for all $x \in \mathcal{X}$ instead of just $\ell_t(x^*) \leq f(x^*)$ because, similarly to the examples with the convexity or the strong convexity assumption, we usually obtain these global lower bounds from our assumptions and we do not know $\ell_t(x^*)$ or x^*. In Chapter 3, we use an online algorithm for one part of our solution and we actually need to use losses that do not satisfy $\ell_t(x) \leq f(x)$ for all $x \in \mathcal{X}$ (the function f is not convex in this case). Our losses there only lower bound the function in a region for which we know x^* is in.

So far, we have presented examples in which an optimization problem is reduced to an online one. But we could even use an online learning algorithm to control one part of the optimization only, while exploiting other properties of our function in other different ways. For simplicity, we will just mention the structure of two examples, leaving the details of their realizations for later. At this point, we know that for convex optimization we can focus on bounding the following, since $f(\bar{x}_a) - f(x^*)$ can be bounded, as in Equation (2.2.3), by a convex combination of

$$f(x_t) - f(x^*) \leq \langle \nabla f(x_t), z_t - x^* \rangle + \langle \nabla f(x_t), x_t - z_t \rangle, \tag{2.2.4}$$

for $t = 1, \ldots, T$. In this decomposition, the online learning algorithm could tell us to play z_t but instead we play x_t, and receive the loss $a_t\langle \nabla f(x_t), \cdot \rangle$. The online learning algorithm controls a weighted sum of $\langle \nabla f(x_t), z_t - x^* \rangle$, for $t = 1, \ldots, T$ and the other part of the right hand side of (2.2.4) could be controlled in other ways. This decomposition in fact will be used for one analysis of Nesterov's AGD. Another example of structure is the following:

$$f(x_t) - f(x^*) \leq \langle \nabla f(x_t), x_t - x^* \rangle = \langle \overline{\nabla f}(x_t), x_t - x^* \rangle + \langle \nabla f(x_t) - \overline{\nabla f}(x_t), x_t - x^* \rangle \tag{2.2.5}$$

where $\overline{\nabla f}(x_t)$ is defined as the vector whose i-th coordinate is $\max\{-1, \min\{1, \nabla_i f(x)\}\}$, that is, $\nabla_i f(x)$ clipped to $[-1, 1]$. In this decomposition, we would have $a_t\langle \overline{\nabla f}(x_t), \cdot \rangle$ as losses. This can be useful because for linear losses, the regret scales with $\|\ell_t\|$, for some norm $\|\cdot\|$ that depends on the algorithm. If we have a function such that $\|\nabla f(x_t)\|$ is very large but it has sufficient

structure so we can control the second summand of (2.2.5), then we can do something useful with the online learning algorithm controlling the first summand without paying the potentially high cost of a bound on $\|\nabla f(x_t)\|$. In fact, this is one of the pieces in the solution of the problem in Chapter 4, which is an idea that already appeared in (Allen-Zhu and Orecchia, 2019).

After this motivation about how online learning can be useful for optimization, we now present the two most common algorithms there are in this area, and their different equivalent formulations. Each formulation provides a different point of view of the methods, which helps with intuition.

While we limit the exposition of this chapter to deterministic optimization, we note that online learning can be applied to optimization with a stochastic gradient oracle as well. For instance, one can consider that our algorithm samples the next prediction point x_t from a distribution and the adversary, that cannot predict the outcome of the sample but has access to our distribution, can select $\mathbb{E}\left[\nabla f(x_t)\right]$ as loss, so we can bound $\mathbb{E}\left[\sum_{t=1}^{T} \nabla f(x_t) - \min_{u\in\mathcal{X}} \nabla f(u)\right]$. We note that when the gradient oracle has more structure, like in the finite sum case, additional ideas are used to achieve acceleration like variance reduction (Defazio, Bach, and Lacoste-Julien, 2014; Johnson and Zhang, 2013; Zhang, Mahdavi, and Jin, 2013) and negative momentum (Allen-Zhu, 2017a).

2.2.1 Follow the Regularized Leader

In an online learning problem, our aim is to predict the best point we would play in hindsight. If the losses are regular enough, intuition would tell us that the optimum of the sum of the currently observed losses could be a good point, since it should not change much from one iteration to another. This intuition is almost true. This strategy works for very regular optimization problems, but it can fail for classes of functions like Lipschitz convex losses whereas a slight but important algorithmic modification proves to be more powerful. Playing the current optimum is the so-called Follow the Leader (FTL) strategy.

Example 2.2.1 (Failure of FTL). *A way to make FTL fail is to generate a sequence of one dimensional linear losses such that the sign of the slope of $\sum_{i=1}^{t} \ell_t$ is changing at every iteration t. In that case the optimizer is very unstable and we predict the point of worse instantaneous loss for $t \geq 2$. Concretely, let $\mathcal{X} = [-1, 1]$ and set the first loss to $\ell_1(x) = x$. Then, let the rest of the sequence $\{\ell_t\}_{t=2}^{T}$ be $-2x, 2x, -2x, 2x$, etc. For any time $t \geq 2$ the loss incurred by FTL is 2, so $\sum_{t=1}^{T} \ell_t(x_t) \geq 2(T-1)$ but $\sum_{t=1}^{T} \ell_t(x) \in \{x, -x\}$ so the loss of the best fixed action is -1 and the regret is at least $2T-1$. In rounds $t = 2, \dots, T$ our algorithm selected the worse possible point. This is by no means a good strategy in this case.*

What is missing in the FTL approach order to have a good algorithm is some mechanism that stabilizes our iterates. This is captured by the Follow the Regularized Leader (FTRL) approach, that for each round $t = 1, \dots, T$, predicts

$$x_t \in \arg\min_{x\in\mathcal{X}} \left\{ \sum_{i=1}^{t-1} \ell_i(x) + \psi_t(x) \right\}, \tag{FTRL}$$

where ψ_t is a regularizer that makes the sum strongly convex with respect to some norm $\|\cdot\|$, and will make the predictions stable against attacks like the one in the example above. But what is the price to pay for adding a regularizer? The interesting point is that it will just add a constant factor in most important cases. Assume for simplicity and without loss of generality that $\min_{x\in\mathcal{X}} \psi_t(x) = 0$, since adding a constant will not change the arg min. Adding a regularizer ψ_t does not contribute to more than a constant factor to the regret bound on R_t we aim to prove if the maximum value it can take is of the order of this bound! So we can regularize without any worries that this will worsen our estimation: it comes (almost) for free. One would only worry in a case in which these small extra constants are important.

Regarding FTRL, we note that the approach still has one potential issue, that was also present in the FTL strategy, which is that the minimization subproblem could be expensive. But as in the case with convex optimization, one can simplify the update by choosing lower bounds $\mathcal{L}_t$ on the losses ℓ_t, such that $\mathcal{L}_t(x_t) = \ell_t(x_t)$ because in that case we have

$$R_T^{\ell} \stackrel{\text{def}}{=} \sum_{t=1}^{T} \ell_t(x_t) - \min_{u\in\mathcal{X}} \sum_{t=1}^{T} \ell_t(u) \leq \sum_{t=1}^{T} \mathcal{L}_t(x_t) - \min_{u\in\mathcal{X}} \sum_{t=1}^{T} \mathcal{L}_t(u) \stackrel{\text{def}}{=} R_T^{\mathcal{L}}, \tag{2.2.6}$$

and $\psi_t(x)$ and $\mathcal{L}_t$ are chosen so that problem $\arg\min_{x\in\mathcal{X}}\{\psi_t(x)+\sum_{i=1}^{t-1}\mathcal{L}_i(x)\}$ can be solved faster while still being strongly convex. We could potentially pay a considerably greater regret by doing this simplification. Solutions trade off efficiency of the subproblem with better estimations of the regret. This depends on the structure of the problem and on which lower bound estimates we can build on the losses that allow for solving the subproblem fast. One should take into account that in light of (2.2.6), for two surrogate losses $\mathcal{L}'_t \leq \mathcal{L}_t \leq \ell_t$ with $\mathcal{L}'_t(x_t) = \mathcal{L}_t(x_t) = \ell_t(x_t)$, we have that $\mathcal{L}'_t$ cannot yield a better regret. It is only natural: looser losses lose more. When using online learning for optimization, this trade-off translates to less subproblem time versus fewer iterations. In many instances, the subproblem is chosen so it can be solved in closed form.

Example 2.2.2. *Let's take a natural surrogate of convex losses: a subgradient. That is $\mathcal{L}_t(x) = \langle g_t, x\rangle$ for some $g_t \in \partial\ell_t(x_t)$. If we take a regularizer $\psi_t(x) = \frac{\sigma}{2}\|x - x_1\|_2^2$, for an arbitrary initial point x_1 (note this is compatible with the definition of x_1 in FTRL), which is σ-strongly convex with respect to $\|\cdot\|_2$ then the minimization problem can be easily solved for $\mathcal{X} = \mathbb{R}^n$:*

$$\arg\min_{x\in\mathcal{X}}\{\sum_{i=1}^{t-1}\langle g_i, x\rangle + \frac{\sigma}{2}\|x - x_1\|_2^2\} = x_1 - \frac{1}{\sigma}\sum_{i=1}^{t-1} g_i. \tag{2.2.7}$$

The following lemma shows a bound on the regret that is incurred when playing according to the FTRL strategy.

Lemma 2.2.3. *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be closed and convex and let $\ell_t : \mathcal{X} \to \mathbb{R}$ be convex. Assume $\psi_t + \sum_{i=1}^{t-1}\ell_i$ is σ_t-strongly convex with respect to $\|\cdot\|$, let x_t be as in (FTRL). Then, for all $u \in \mathcal{X}$ we have*

$$\sum_{t=1}^{T}(\ell_t(x_t) - \ell_t(u)) \leq \psi_{T+1}(u) - \min_{x\in\mathcal{X}}\psi_1(x) + \sum_{t=1}^{T}(\frac{\|g_t\|_*^2}{2\sigma_t} + \psi_t(x_{t+1}) - \psi_{t+1}(x_{t+1})).$$

In particular let $\min_{u\in\mathcal{X}} \psi_1(u) = 0$ *and* $\psi_t(x) = \psi(x)/a_{t-1}$ *for a fixed* ψ *that is* σ*-strongly convex with respect to* $\|\cdot\|$*, and* $0 < a_t \leq a_{t-1}$*, i.e., increasing regularizers. Then for all* $u \in \mathcal{X}$*:*

$$\sum_{t=1}^{T}(\ell_t(x_t) - \ell_t(u)) \leq \frac{\psi(u)}{a_{T-1}} + \sum_{t=1}^{T}\frac{a_{t-1}\|g_t\|_*^2}{2\sigma}, \tag{2.2.8}$$

for all subgradients $g_t \in \partial\ell_t(x_t)$*.*

Proof For each round t we have

$$\begin{aligned}\sum_{t=1}^{T}(\ell_t(x_t) - \ell_t(u)) &\overset{①}{=} \sum_{t=1}^{T}\left[(\ell_t(x_t) + \sum_{i=1}^{t-1}\ell_i(x_t) + \psi_t(x_t)) - (\sum_{i=1}^{t}\ell_i(x_{t+1}) + \psi_{t+1}(x_{t+1}))\right] \\ &\quad + (\sum_{i=1}^{T}\ell_i(x_{T+1}) + \psi_{T+1}(x_{T+1})) - (\sum_{i=1}^{T}\ell_i(u) + \psi_{T+1}(u)) \\ &\quad + \psi_{T+1}(u) - \psi_1(x_1) \\ &\overset{②}{\leq} \psi_{T+1}(u) - \min_{x\in\mathcal{X}}\psi_1(x) + \sum_{t=1}^{T}(\frac{\|g_t\|_*^2}{2\sigma_t} + \psi_t(x_{t+1}) - \psi_{t+1}(x_{t+1})).\end{aligned}$$

In ① , we just add and subtract some terms in order to capture, in the first line, the intuition about the fact that the minimizer of the current sum of losses should be a good solution, if regularized. Indeed, we played x_t without knowing ℓ_t, and we want to compute how well we did in terms of regularized $\sum_{i=1}^{t}\ell_i$ in comparison to the point we would play if we knew the future by one step, that is, the point x_{t+1}, which could (*almost*) be the comparator if we stopped at time t. We say *almost* because this reasoning holds up to regularization. But as we mentioned before, the regularizer in general will be something of the order of the regret we expect to obtain or lower, so having it in the estimation will not hurt more than a constant factor. As a side effect, up to regularization, x_{T+1} is the point we would play if we knew all the losses in hindsight, so we can use it to drop the losses of the arbitrary fixed action, which are worse up to regularization. Thus, we show such comparison in the second line. The third line contains some remaining terms. Hence, in ②, we can drop this second line, use the definition of x_1, and use strong convexity to bound the first line. In order to do the last thing, we would want to use the same function for the comparison, say $\psi_t + \sum_{i=1}^{t}\ell_i$, so we add and subtract $\psi_t(x_{t+1})$ and use the fact that for any σ_t-strongly convex function $f : \mathbb{R}^n \to \mathbb{R}$ with x^* as optimizer and any $g_t \in \partial f(x_t)$ as subgradient at x_t we have

$$f(x_t) - f(x_{t+1}) \overset{①}{\leq} f(x_t) - f(x^*) \overset{②}{\leq} \langle g_t, x_t - x^*\rangle - \sigma_t\|x^* - x_t\|^2 \overset{③}{\leq} \frac{\|g_t\|_*^2}{2\sigma_t},$$

where ① uses optimality of x^*, ② holds by strong convexity and ③ is Cauchy-Schwarz and $2ab \leq a^2 + b^2$ for $a, b \in \mathbb{R}_{\geq 0}$: $\langle v, w\rangle \leq \|v\|_*\|w\| \leq \frac{1}{2}\|v\|_*^2 + \frac{1}{2}\|w\|^2$. Finally, for $f = \psi_t + \sum_{i=1}^{t-1}\ell_i$, g_t can be taken in $\partial\ell_t(x_t)$ since $x_t = \arg\min_{x\in\mathcal{X}}\{f(x) - \ell_t(x)\}$ so $0 \in \partial(f - \ell_t)(x_t)$ and thus any subgradient of $\ell_t(x_t)$ is a subgradient of $f(x_t)$ (add the convexity inequalities for $f - \ell_t$ and ℓ_t at x_t) so the first part of the theorem works for any $g_t \in \partial\ell_t(x_t)$.

For the second part of the theorem we note that adding a constant to the regularizer does not change the update rule so the assumption $\min_{u\in\mathcal{X}} \psi_1(u) = 0$ is done without loss of generality. By the definition of the regularizers we have $\psi_t(x_{t+1}) - \psi_{t+1}(x_{t+1}) \leq 0$ and that ψ_t is $(\frac{\sigma}{a_{t-1}})$-strongly convex. Finally, note that since we did not use x_{T+1} nor ψ_{T+1} for any prediction, we could change them for the purpose of the analysis. The first summand of the bound will be best if ψ_{T+1} is as small as possible, but we require $\psi_T(x_{T+1}) \leq \psi_{T+1}(x_{T+1})$ to remove the regularizers of the second summand. Thus, we can set $\psi_{T+1} = \psi_T$ and obtain $\psi_{T+1}(u) = \psi(u)/a_{T-1}$. ■

Remark 2.2.4 (Arbitrary initialization). *We note that by using $D_\psi(x, x_1)$ as first regularizer in (FTRL) for an arbitrary point $x_1 \in \mathcal{X}$ and some strongly convex ψ we have that the first point defined by (FTRL) is precisely x_1. Also, applying part 2) of the previous Lemma 2.2.3 with regularizers $\psi_t(x) = D_\psi(x, x_1)/a_{t-1}$ for some σ-strongly convex map ψ, we obtain the same result but starting at an arbitrary point x_1. In such a case, the error depends on $D_\psi(u, x_1)$ instead of on $\psi(u)$.*

Remark 2.2.5. *Since we used the strong convexity property of the FTRL subproblem in our theorem, the guarantee depends on the losses via their subgradients only. This means that this bound would be the same if we had just used linearized losses, i.e., using a subgradient in $\partial\ell_t(x_t)$ as the loss. Except for the minor fact that the bound would depend on the exact subgradient that was chosen. However, if the losses satisfy stronger properties, we can obtain improvements. If the losses were μ_t-strongly convex, we could set $\mathcal{L}_t(x) = \langle g_t, x\rangle + \frac{\mu_t}{2}\|x - x_t\|^2$ and we would actually not need any regularizer. The term $\sum_{i=1}^{t-1}\mathcal{L}_i$ would be strongly convex and this fact can be exploited in the same way as in the proof above to directly guarantee stability of the predictions and low regret. There are many other variants and settings in which one could think of applying this technique. Another example would be to have $\psi_t(x) = \frac{\sigma_t}{2}\|x - x_1\|_2^2$ so that solving the subproblem consists of the proximal operator of the loss, which could be applied to proximable lower bound estimations of the actual loss, which is the usual approach in composite optimization. On another note, observe that even if we get the same worse-case regret bound by linearizing the losses or not in the lemma above, in practice one would expect to obtain better empirical performance if the estimations are better. An empirical example in machine learning is (Asi and Duchi, 2019) that uses the fact that many machine learning losses are non-negative so it can use surrogates of losses ℓ_t that are $\max\{0, \langle g_t, x\rangle\}$ and obtains more empirical robustness to hyperparameters as a consequence. In any case, using FTRL with gradients as surrogates for the losses plus strongly convex regularizers will be enough for acceleration of convex and smooth functions, so we will not elaborate more on other variants of FTRL.*

Example 2.2.6 (Non-smooth convex optimization and adaptivity). *We can use the previous Lemma 2.2.3 applied to convex optimization for an L-Lipschitz function to obtain optimal worse-case rates, up to constants. If we use the bound $\|g_t\|_*^2 \leq L^2$, select the regularizer $\psi = \frac{\sigma}{2}\|x - x_1\|_2^2$, and choose constant learning rate we can optimize the bound at*

$a_t = a \stackrel{\text{def}}{=} \sqrt{\frac{\|u-x_1\|_2^2\sigma^2}{TL^2}}$ *and conclude*

$$f\left(\frac{1}{T}\sum_{t=1}^{T} x_t\right) - f(x^*) \overset{①}{\leq} \frac{R_T}{T} \overset{②}{\leq} \frac{1}{T}\left(\frac{\sigma\|u-x_1\|_2^2}{2a} + \frac{aTL^2}{2\sigma}\right) \overset{③}{=} \frac{L\|u-x_1\|_2}{\sqrt{T}}.$$

Inequality ① *holds by* (2.2.2), ② *is the guarantee of Lemma 2.2.3 and* ③ *holds by the choice of* a*, which optimizes the bound by the weighted arithmetic-geometric inequality. The rate obtained is optimal up to constants (Nemirovski and Yudin, 1983b). Note we used the value* $\|u-x_1\|_2$ *in the algorithm, which is generally not known. There are techniques to adapt to this value that allow to obtain the same bound up to a* log *factor without assuming knowledge on* $\|u-x_1\|_2$*, cf. (Orabona and Pál, 2016) for instance. The extra log factor is necessary in the online learning problem with linear and* L*-Lipschitz losses if one does not assume* $\|u-x_1\|_2$ *is known (Streeter and McMahan, 2012). We note that adapting to* L *in case* $\|u-x_1\|_2$ *is known is also possible and it only requires to pay an extra* log *factor. However, adapting to both* L *and* $\|u-x_1\|_2$ *requires paying an exponential penalty on the regret (Cutkosky and Boahen, 2017).*

We can see in this example how the intuition we provided about the regularizer is satisfied: $\psi_{T+1}(u) = \psi(u)/a_{T-1}$ *is of the order of the regret bound. However, note we also had to use the value of* T*, and if we wanted to optimize without setting a final time* T *we would have to set the learning rate* a_t *to something proportional to* $1/\sqrt{t}$ *so regularizer is always of the order of the optimal regret* $\sqrt{t}$ *for any time* t*. In such a case, the second summand becomes proportional to* $\sum_{i=1}^{t} 1/\sqrt{i} \approx \sqrt{t}$ *which is also of the right order, so we get optimal rates up to constants as well. That is, for* $a_t = \sqrt{\frac{\|u-x_1\|_2^2\sigma^2}{tL^2}}$*, we have*

$$f(\bar{x}_t) - f(x^*) \leq \frac{R_t}{t} \leq \frac{L\|u-x_1\|_2\sqrt{t-1}}{2t} + \frac{\sum_{i=1}^{t}\frac{1}{\sqrt{i}}L\|u-x_1\|_2}{2t} = O\left(\frac{L\|u-x_1\|_2}{\sqrt{t}}\right),$$

where $\bar{x}_t \stackrel{\text{def}}{=} \frac{1}{t}\sum_{i=1}^{t} x_i$.

Remark 2.2.7 (Dual point of view on FTRL). *It turns out that FTRL with linearized losses has a simple dual formulation. Abusing the notation, let* ℓ_t *be the vector that defines the loss function. If* ψ *is strongly convex then we have, by Fact 2.1.6,*

$$x_t = \arg\min_{x\in\mathcal{X}} \left\{\psi_t(x) + \sum_{i=1}^{t-1}\langle \ell_i, x\rangle\right\} = \nabla\psi_{t+1}^*\left(-\sum_{i=1}^{t-1}\ell_i\right).$$

So if we start with a dual variable $z_1 = 0$*, the algorithm can be written as*

$$x_t = \nabla\psi_t^*(z_t); \qquad z_{t+1} = z_t - \ell_t, \qquad \text{for } t = 1,\dots,T. \tag{2.2.9}$$

That is, FTRL with linearized losses consists of performing gradient descent on the dual space (on the variables z_t*) and then we use the projection onto the primal as the prediction for each iteration. The map* $\nabla\psi_t^*$ *acts as the projection operation. This projection can be computed in two steps in case that we can extend the domain of definition of* $\psi_t : \mathcal{X} \to \mathbb{R}$ *to* $\mathbb{R}^n$*. Indeed, let*

$\Psi_t : \mathbb{R}^n \to \mathbb{R}$ *be a differentiable strongly convex map such that* $\psi_t(x) = \Psi_t(x)$ *for all* $x \in \mathcal{X}$. *Then*

$$x_t = \arg\min_{x\in\mathcal{X}} \{D_{\Psi_t}(x, \hat{x}_t)\}, \text{ where } \hat{x}_t = \arg\min_{x\in\mathbb{R}^n} \left\{ \Psi_t(x) + \sum_{i=1}^{t-1} \langle \ell_i, x\rangle \right\} \overset{①}{=} \nabla\Psi_t^*(-\sum_{i=1}^{t-1} \ell_i), \tag{2.2.10}$$

because

$$\arg\min_{x\in\mathcal{X}}\{D_{\Psi_t}(x, \hat{x}_t)\} \overset{②}{=} \arg\min_{x\in\mathcal{X}}\{\Psi_t(x) - \langle \nabla\Psi(\hat{x}_t), x\rangle\} \overset{③}{=} \arg\min_{x\in\mathcal{X}}\{\Psi_t(x) + \langle \sum_{i=1}^{t-1} \ell_i, x\rangle\} = x_t.$$

where ① *holds by Fact 2.1.6,* ② *uses the definition of the Bregman divergence, and* ③ *is due to* $\nabla\Psi^*(\mathbb{R}^n) = \mathbb{R}^n$, *equation* ①, *and the discussion after Fact 2.1.6, that implies* $\nabla\Psi(\nabla\Psi^*(z)) = z$ *for all* $z \in \mathbb{R}^n$. *This formulation may have computational advantages if the two* arg min*'s in* (2.2.10) *are simple. It also gives intuition about the behavior of the iterates.*

We note that due to this dual formulation and due to historical reasons, the FTRL algorithm with linear losses is also known in the literature by other different names: dual averaging, lazy Mirror Descent, and Nesterov's Mirror Descent. The other online learning algorithm we will present in what follows is (Online) Mirror Descent, which is also called: greedy Mirror Descent, or Nemirovski's Mirror Descent. Because of these different names and due to the fact that the algorithms present some similarities, one can see in the literature that FTRL is sometimes just called Mirror Descent as well. This can be a source of confusion when delving into the literature.

2.2.2 (Online) Mirror Descent

Mirror descent is an algorithm that was originally designed for convex optimization by Nemirovski and Yudin (1983b) and that can be used for online learning with linear losses. When it is used in online learning it is called Online Mirror Descent (OMD). OMD has several different equivalent formulations, each of them providing different intuition. Essentially, we want to keep following our initial intuition about predicting points that give lower values for past losses that at the same time are stable. In order to do that, we explicitly force the next prediction to be close to the current one in this algorithm. On the other hand we saw that FTRL with linear losses can be seen as performing unconstrained GD in the dual space, using a projection operator only to compute a prediction. This algorithm is also equivalent to a dual GD, but this time it is projected GD, where the constraint set is the dual set $\mathcal{D} = \nabla\psi(\mathcal{X})$, and where the projection operator is $\nabla\psi \circ \nabla\psi^*$.[4] The function ψ is a fixed differentiable strongly convex regularizer. So the method can be expressed with these two equivalent formulations. Let $g_t \in \partial\ell_t$ be a subgradient of the convex loss ℓ_t. We have

$$z_{t+1} = \nabla\psi(x_t) - a_t g_t; \qquad x_{t+1} = \nabla\psi^*(z_{t+1}), \qquad \text{for } t = 1, \dots, T. \tag{2.2.11}$$

[4] We note that when $\mathcal{D} = \mathbb{R}^n$, or equivalently, when the projection operator $\nabla\psi \circ \nabla\psi^*$ is the identity, then FTRL and OMD are the same algorithm.

$$x_{t+1} = \arg\min_{x \in \mathcal{X}} \{\langle g_t, x\rangle + \frac{1}{a_t} D_\psi(x, x_t)\}, \qquad \text{for } t = 1, \ldots, T. \tag{2.2.12}$$

Above, $a_t \in \mathbb{R}_{>0}$ is a learning rate and $x_1 \in \mathcal{X}$ is an arbitrary initial point. As with FTRL, if we have an extension $\Psi : \mathbb{R}^n \to \mathbb{R}$ of ψ that is differentiable and strongly convex, we can break the iteration (2.2.12) into two simpler minimization problems:

$$x_{t+1} = \arg\min_{x \in \mathcal{X}} \left\{ D_\Psi(x, \hat{x}_{t+1}) \right\},$$

where $\hat{x}_{t+1} = \arg\min_{x \in \mathbb{R}^n} \left\{ \langle g_t, x\rangle + \frac{1}{a_t} D_\Psi(x, x_t) \right\} = \nabla\Psi^*(\nabla\psi(x_t) - a_t g_t)$.

We have used a fixed regularizer and variable learning rates (which corresponds to the structure of regularizers of part 2 in Lemma 2.2.3) but we could have used generic time varying regularizers as in the FTRL approach. That is, we can generalize the update of (2.2.12) to

$$z_{t+1} = \nabla\psi_{t+1}(x_t) - g_t; \qquad x_{t+1} = \nabla\psi_{t+1}^*(z_{t+1}), \qquad \text{for } t = 1, \ldots, T. \tag{2.2.13}$$

or equivalently

$$x_{t+1} = \arg\min_{x \in \mathcal{X}} \{\langle g_t, x\rangle + D_{\psi_t}(x, x_t)\}, \qquad \text{for } t = 1, \ldots, T. \tag{2.2.14}$$

We have the following guarantee on the OMD algorithm.

Lemma 2.2.8 (Mirror Descent with ψ/a_t as regularizers). *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be closed and convex and let $\ell_t \in \mathbb{R}^n$ be convex losses and $g_t \in \partial\ell_t$. For a σ-strongly convex regularizer $\psi : \mathcal{X} \to \mathbb{R}$ with respect to $\|\cdot\|$, run OMD with regularizers $\psi_t = \psi/a_t$ for some $a_t > 0$, as in (2.2.12). Then, for all $u \in \mathcal{X}$ we have*

1. *Classical Mirror Descent lemma (one step).*

$$\begin{aligned} a_t\langle g_t, x_t - u\rangle &\le a_t\langle g_t, x_{t+1} - x_t\rangle - \frac{\sigma}{2}\|x_{t+1} - x_t\|^2 + D_\psi(u, x_t) - D_\psi(u, x_{t+1}) \\ &\le \frac{a_t^2\|g_t\|_*^2}{2\sigma} + D_\psi(u, x_t) - D_\psi(u, x_{t+1}). \end{aligned}$$

2. *Regret with increasing regularizers, i.e., with decreasing $a_t < a_{t-1}$ (like (2.2.8) in FTRL):*

$$\sum_{t=1}^T \ell_t(x_t) - \ell_t(u) \le \sum_{t=1}^T \langle g_t, x_t - u\rangle \le \frac{1}{2\sigma}\sum_{t=1}^T a_t\|g_t\|_*^2 + \max_{1 \le t \le T} \frac{D_\psi(u, x_t)}{a_T}.$$

3. *Guarantee on weighted losses, with arbitrarily weighted regularizers:*

$$\frac{1}{\sum_{t=1}^T a_t}\sum_{t=1}^T a_t\langle g_t, x_t - u\rangle \le \frac{D_\psi(u, x_1)}{\sum_{t=1}^T a_t} + \frac{1}{2\sigma}\sum_{t=1}^T \frac{a_t^2\|g_t\|_*^2}{\sum_{t=1}^T a_t}.$$

Note we can use part 3. for convex optimization. That is, we can show that with $u = x^*$ and $g_t \in \partial f(x_t)$, the point $\bar{x}_a = \frac{\sum_{t=1}^T a_t x_t}{\sum_{t=1}^T a_t}$ has an optimality gap given by the right hand side of Lemma 2.2.8.3.

Proof of Lemma 2.2.8. Note that $D_{\psi_t}(x,y) = \frac{1}{a_t} D_\psi(x,y)$ for all $x, y \in \mathcal{X}$. The first part of the lemma follows from

$$\begin{aligned}
a_t\langle g_t, x_t - u\rangle &= a_t\langle g_t, x_t - x_{t+1}\rangle + a_t\langle g_t, x_{t+1} - u\rangle \\
&\overset{①}{\leq} a_t\langle g_t, x_t - x_{t+1}\rangle - \langle \nabla_w D_\psi(w, x_t)|_{w=x_{t+1}}, x_{k+1} - u\rangle \\
&\overset{②}{=} a_t\langle g_t, x_t - x_{t+1}\rangle - D_\psi(x_{t+1}, x_t) + D_\psi(u, x_t) - D_\psi(u, x_{t+1}) \\
&\overset{③}{\leq} a_t\langle g_t, x_t - x_{t+1}\rangle - \frac{\sigma}{2}\|x_{t+1} - x_t\|^2 + D_\psi(u, x_t) - D_\psi(u, x_{t+1}) \\
&\overset{④}{\leq} \frac{a_t^2\|g_t\|_*^2}{2\sigma} + D_\psi(u, x_t) - D_\psi(u, x_{t+1})
\end{aligned}$$

Inequality ① comes from the first-order optimality condition of the definition of x_t, that is ⑤ below,

$$\langle \nabla_w D_\psi(w, x_t)|_{w=x_{t+1}} + a_t g_t, u - x_{t+1}\rangle = a_t\langle \nabla_w D_{\psi_t}(w, x_t)|_{w=x_{t+1}} + g_t, u - x_{t+1}\rangle \overset{⑤}{\geq} 0,$$

for all $u \in \mathcal{X}$. ② uses is the triangle equality of Bregman divergences (cf. Lemma 2.1.8.3 and Lemma 2.1.8.5). Inequality ③ uses strong convexity of $D_\psi(\cdot, x_t)$ which holds due to the strong convexity of ψ. Finally ④ is derived from $\langle v, w\rangle - \frac{1}{2}\|w\|^2 \leq \frac{1}{2}\|v\|_*^2$ for $v, w \in \mathbb{R}^n$, that holds by Cauchy-Schwarz and $\|v\|_* \cdot \|w\| \leq \frac{1}{2}\|v\|_*^2 + \frac{1}{2}\|w\|^2$. Essentially, this last step computes the maximum difference between the optimum of a quadratic with negative leading term and such quadratic evaluated at x_t. When we are in an unconstrained problem this bound is the natural thing to use. However, in a constrained problem it could be too loose and it usually makes more sense to use the bound after ③ directly. We note that if $\psi = \frac{\sigma}{2}\|\cdot - x_1\|_2^2$ and $\mathcal{X} = \mathbb{R}^n$ we have that ① and ③ are equalities, so in that case we could obtain an equality for the regret.

For the second part, we add up the first part, divided by a_t, and obtain

$$\begin{aligned}
\sum_{t=1}^T \langle g_t, x_t - u\rangle &\overset{①}{\leq} \frac{1}{2\sigma}\sum_{t=1}^T a_t\|g_t\|_*^2 + \frac{1}{a_1} D_\psi(u, x_1) + \sum_{t=2}^T \left(\frac{1}{a_t} - \frac{1}{a_{t-1}}\right) D_\psi(u, x_t) \\
&\overset{②}{\leq} \frac{1}{2\sigma}\sum_{t=1}^T a_t\|g_t\|_*^2 + \max_{1\leq t\leq T} \frac{D_\psi(u, x_t)}{a_T}.
\end{aligned}$$

In ① we also dropped $-D_\psi(u, x_{T+1})/a_T \leq 0$. Inequality ② uses $a_t \leq a_{t-1}$, bounds the Bregman divergences by the maximum and adds up the telescoping sum. Note that if we were to use constant learning rates the error term coming from Bregman divergences would be $D_{\psi_1}(u, x_1)$. That is, using a constant regularizer we pay for the initial (and constant) regularization at u. The rest of the error terms come from changing the point of attraction of regularizers.

Luckily, since for convex optimization it is enough to bound a regret that uses weighted losses, we can use arbitrary learning rates and prevent paying for the changes in regularizer as in the case above. This is the motivation of the third part of this lemma, which follows by simply adding up the first part, dropping $-D_\psi(u, x_{T+1})/a_T \leq 0$ and normalizing. ■

Remark 2.2.9 (General regularizers). *We can obtain a more general expression using arbitrary regularizers. Indeed, we note that in the proof of the one-step Lemma 2.2.8.1 above we could have used arbitrary regularizers to obtain*

$$\ell_t(x_t) - \ell_t(u) \leq \frac{\|g_t\|_*^2}{2\sigma_t} + D_{\psi_t}(u, x_t) - D_{\psi_t}(u, x_{t+1}),$$

which added up leads to the following, if we drop $-D_{\psi_T}(u, x_{T+1})$ and reorganize terms:

$$\sum_{t=1}^{T}(\ell_t(x_t) - \ell_t(u)) \leq D_{\psi_1}(u, x_1) + \sum_{t=1}^{T}\frac{\|g_t\|_*^2}{2\sigma_t} + \sum_{t=2}^{T}(D_{\psi_t}(u, x_t) - D_{\psi_{t-1}}(u, x_t)).$$

Intuitively, the regret term coming from the regularizers is: the value of the first regularizer at u and then the changes in regularizer at u when we switch from using ψ_{t-1} to using ψ_t.

One could ask what is the motivation for having the Bregman divergence in the optimization problem. The answer is that in fact it is not really needed. In the discussion before Lemma 2.1.8 we motivated the Bregman divergences as regularizers with a specific minimizer (and minimum value 0, just for simplicity and without loss of generality). If we were to use, at iteration t, regularizers ψ_t with minimizer at x_t and define $x_{t+1} = \arg\min_{x\in\mathcal{X}}\{\psi_t(x) + \langle g_t, x\rangle\}$ we would obtain the same thing and we could translate our guarantees if we use these maps. Indeed, this is a triviality since in such a case $D_{\psi_t}(x, x_t) = \psi_t(x) - \psi_t(x_t)$ and the constant $\psi_t(x_t)$ does not change the argmin. We make this point to emphasize that the role of Bregman divergences is just that of regularizers that we build from other regularizers so that the minimizer is at a particular point that we decide.

We note the method is called *Mirror* Descent because of its dual formulation in which one can see that the method is performing (projected) GD on the dual space, similarly to (2.2.9). Note that Mirror Descent (MD) in the offline setting was designed before FTRL. We will discuss in Section 2.6.1 a relationship between FTRL and MD when applied to convex optimization.

2.3 Nemirovski's Quasi-accelerated gradient descent

The Conjugate Gradient Descent method (CGD) is where research on acceleration started. It is a method designed for the minimization of a quadratic function. Its geometrical analysis inspired the first (quasi-)accelerated method for general convex optimization (Nemirovski, 1982a; Nemirovski, 1982b; Nemirovski and Yudin, 1983a). In the sequel, we will prove convergence for the CGD method and its relation to Nemirovski's accelerated gradient descent, that uses a plane search and then an improvement that uses a line search only. This section is adapted from the English translation (Nemirovski, 1982a) of the original work, where we have changed the exposition and the proof to make it more accessible. The connection between CGD and acceleration has been rediscovered from different points of view in (Karimi and Vavasis, 2016; Diakonikolas and Orecchia, 2019a). For the purposes of this section we will only need to know that the CGD method is designed to optimize functions of the form $f(x) = \frac{1}{2}x^\top Ax - b^\top x$, with

$A \succ 0$ and that, starting at an arbitrary point x_0, it will iteratively compute a point x_i satisfying it is the minimizer of f on the affine plane $x_0 + \sum_{k=1}^{i-1} \eta_k \nabla f(x_k)$ for $\eta_k \in \mathbb{R}$. We note that CGD satisfies a stronger guarantee than the one we prove below. In the general guarantee, the error at a given time step T is given by an optimal value over the set of polynomials of degree T, and the analysis of its worse-case convergence relies on showing that with Chebyshev polynomials of the first kind we achieve the rate $O(1/T^2)$. See (Nocedal and Wright, 2006, chap. 5) for a proof. Chebyshev polynomials are classically linked to acceleration in this way and in Chapter 5 we show we can combine this fact with a bandit algorithm in order to solve our decentralized cooperative bandit problem. Note there are other kinds of acceleration by means of interpolation, like Anderson's acceleration (Walker and Ni, 2011), that are beyond the scope of this chapter.

Before we present the proof, we introduce a sequence $\{\eta_i\}_{i=0}^T$ that will appear in most accelerated methods presented in this chapter. Define

$$\eta_0 = 0, \text{ and } \eta_i^2 = \eta_{i-1}^2 + \eta_i = \sum_{j=1}^{i} \eta_j \text{ for } i \geq 1. \tag{2.3.1}$$

We have that $\{\eta_i^2\}_{i=1}^T = \{\sum_{j=1}^i \eta_j\}_{i=1}^T$ dominates the sequence $\{\sum_{j=0}^i \xi_j\}_{i=1}^T$, for $\xi_j = (j+1)/2$ since $\xi_i^2 / \sum_{j=0}^i \xi_j < 1$. Thus,

$$\eta_i^2 = \sum_{j=1}^{i} \eta_i \geq \sum_{j=0}^{i} \xi_j = \frac{(i+1)(i+2)}{4}. \tag{2.3.2}$$

Note that the definition of η_i is implicit and that solving the quadratic equation we can also define it as $\eta_i = (1 + \sqrt{1 + 4\eta_{i-1}^2})/2$. Now we are ready to analyze CGD. In this section we use $\|\cdot\|$ for the norm $\sqrt{\langle \cdot, \cdot \rangle}$ in a Hilbert space.

Proof of convergence of Conjugate Gradient Descent. Let $f(x) = \frac{1}{2} x^\top A x - b^\top x$. Assume $L \succcurlyeq A \succ 0$, i.e., f is strictly convex and L-smooth. Let x^* be the minimizer of f and let $x_0, x_1, \ldots$ be the points the CGD method computes. The convergence rate comes from the three following facts on the iterates:

(CG1) $\langle \nabla f(x_i), x_i - x_0 \rangle = 0$,

(CG2) $\langle \nabla f(x_i), \nabla f(x_j) \rangle = 0$, if $i \neq j$,

(CG3) $f(x_i) \leq f(x_{i-1}) - \frac{1}{2L} \|\nabla f(x_{i-1})\|^2$.

All of them follow easily from the definition of x_i as the minimizer of f on the affine plane $x_0 + \sum_{k=1}^{i-1} \eta_k \nabla f(x_k)$ for parameters $\eta_k \in \mathbb{R}$. Indeed, (CG2) is straightforward, (CG1) is true due to (CG2) and the fact that $x_i - x_0$ is a linear combination of $\{\nabla f(x_j)\}_{j=1}^{i-1}$. (CG3) holds since by definition of x_i and by smoothness we have $f(x_i) \leq f(x_{i-1} - \nabla \frac{1}{L} f(x_{i-1})) \leq f(x_{i-1}) - \frac{1}{2L} \|\nabla f(x_{i-1})\|^2$.

Observe that the following holds

$$f(x_i) - f(x^*) \overset{①}{\leq} \langle \nabla f(x_i), x_i - x^* \rangle \overset{②}{=} \langle \nabla f(x_i), x_0 - x^* \rangle, \tag{2.3.3}$$

by convexity in ① and (CG1) in ②. Adding up (2.3.3) weighted by $\eta_i/\|x_0 - x^*\|$ for $i = 1, \dots, T$ we obtain ① below

$$\begin{aligned}\frac{1}{\|x_0 - x^*\|}\sum_{i=1}^{T}\eta_i(f(x_i) - f(x^*)) &\overset{①}{\leq} \frac{1}{\|x_0 - x^*\|}\langle\sum_{i=1}^{T}\eta_i\nabla f(x_i), x_0 - x^*\rangle \overset{②}{\leq} \|\sum_{i=1}^{T}\eta_i\nabla f(x_i)\| \\ &\overset{③}{=} \sqrt{\sum_{i=1}^{T}\eta_i^2\|\nabla f(x_i)\|^2} \overset{④}{\leq} \sqrt{2L\sum_{i=1}^{T}\eta_i^2[(f(x_i) - f(x^*)) - (f(x_{i+1}) - f(x^*))]} \\ &\overset{⑤}{\leq} \sqrt{2L\sum_{i=1}^{T}(f(x_i) - f(x^*))(\eta_i^2 - \eta_{i-1}^2)} \overset{⑥}{=} \sqrt{2L\sum_{i=1}^{T}\eta_i(f(x_i) - f(x^*))},\end{aligned} \tag{2.3.4}$$

and then ② is Cauchy-Schwarz, ③ holds by (CG2), and ④ follows by (CG3). We used $\eta_0 = 0$ and $-\eta_T^2(f(x_{T+1}) - f(x^*)) \leq 0$ in ⑤, and ⑥ uses $\eta_i^2 = \eta_{i-1}^2 + \eta_i$. Finally:

$$f(x_T) - f(x^*) = \frac{\sum_{i=1}^{T}\eta_i}{\eta_T^2}(f(x_T) - f(x^*)) \overset{①}{\leq} \frac{\sum_{i=1}^{T}\eta_i(f(x_i) - f(x^*))}{\eta_T^2} \overset{②}{\leq} \frac{8L\|x_0 - x^*\|^2}{(T+1)(T+2)},$$

where ① uses that the method is monotonous by definition and ② comes from simplifying (2.3.4) and the bound (2.3.2) on η_T^2. ■

Algorithm 1 Quasi-accelerated gradient descent

Input: An L-smooth wrt $\|\cdot\|$, convex, differentiable $f : \mathbb{R}^n \to \mathbb{R}$. Initial point x_0. Sequence $\{\eta_k\}_{k=0}^{\infty}$ in (2.3.1).

Output: x_T such that $f(x_T) - f(x^*) = O(\frac{L\|x_0 - x^*\|^2}{T^2})$. Searches can be approximated to obtain the same guarantee after $O(T\log(T))$ gradient evaluations.

for $k \leftarrow 0$ **to** $T-1$ **do** ◇ Plane search version
  $x_i \leftarrow \arg\min\{f(x) : x \in P_i\}$, where P_i is the plane such that x_0, $x_{i-1} - \frac{1}{L}\nabla f(x_{i-1}) \in P_i$ and P_i is parallel to $q_{i-1} = \sum_{j=1}^{i-1}\eta_j\nabla f(x_j)$.
end for

for $k \leftarrow 0$ **to** $T-1$ **do** ◇ Line search version
  $x_i \leftarrow \arg\min\{f(x) : x \in \Lambda_i\}$, where Λ_i is the line such that $x_{i-1} - \frac{1}{L}\nabla f(x_{i-1}), x_0 - \sum_{j=1}^{i-1}\frac{\eta_j}{L}\nabla f(x_j) \in \Lambda_i$. ◇ Note $\Lambda_i \subset P_i$.
end for

Nemirovski observed that within the previous proof, (CG2) was not used as such, but a weak version of it. We only needed $\|\sum_{i=1}^{T}\eta_i\nabla f(x_i)\|^2 = \sum_{i=1}^{T}\eta_i^2\|\nabla f(x_i)\|^2$ which is also true if we only impose

$$\langle\nabla f(x_i), \sum_{j=1}^{i-1}\eta_j\nabla f(x_j)\rangle = 0, \text{ for all } i \geq 1. \tag{2.3.5}$$

This condition can be ensured for every general convex L-smooth function. It is enough to find, at the i-th iteration, the point x_i that minimizes f on the plane passing through x_0, $x_{i-1} - \frac{1}{L}\nabla f(x_{i-1})$ and that is and parallel to the vector $q_{i-1} = \sum_{j=1}^{i-1}\eta_j\nabla f(x_j)$. Indeed, by optimality, $\nabla f(x_i)$ will be orthogonal to any vector parallel to the plane. In particular it will

be orthogonal to $x_i - x_0$ and to q_{i-1}, ensuring (CG1) and (2.3.5). Besides, we still have $f(x_i) \leq f(x_{i-1} - \frac{1}{L}\nabla f(x_{i-1})) \leq f(x_{i-1}) - \frac{1}{2L}\|\nabla f(x_{i-1})\|^2$ by smoothness, so (CG3) also holds. Note that the proof of CGD only used convexity despite that we assumed strict convexity (i.e., $A \succ 0$). CGD uses strict convexity in the definition of the algorithm in order to guarantee the three CGD conditions, but it does not use it in the proof. Thus, the same convergence proof as before follows for general convex optimization.

The only new algorithmic operation introduced was a two dimensional minimization per step. Nemirovski later showed that the 2-dimensional minimization can be solved approximately so the convergence still follows and the number of gradient oracle calls to reach a given accuracy is $O(T \ln T)$ if the number of iterations needed with the exact plane search was T. This algorithm was later improved to require a line search only (Nemirovski and Yudin, 1983a). The line is contained within the previous plane. A priori the proof relies on an algebraic trick and the geometrical intuition that CGD provides is lost. The argument is the following: let $y_i \stackrel{\text{def}}{=} x_{i-1} - \frac{1}{L}\nabla f(x_{i-1})$ be a gradient point and define x_i as a point that minimizes f along the line

$$\Lambda_i(\lambda) = y_i + \lambda\left(-y_i + x_0 - \sum_{j=1}^{i-1}\frac{\eta_j}{L}\nabla f(x_j)\right) = (1-\lambda)y_i + \lambda\left(x_0 - \sum_{j=1}^{i-1}\frac{\eta_j}{L}\nabla f(x_j)\right), \quad (2.3.6)$$

for $\lambda \in \mathbb{R}$ and let λ_i be the value used to define x_i.

Proof of Nemirovski's quasi-AGD convergence rates (line search version).

We bound the gap in a similar way to the previous case that used a plane search except that now we cannot remove $\langle \nabla f(x_i), x_i - x_0\rangle$:

$$\begin{aligned}
\sum_{i=1}^{T}\eta_i(f(x_i) - f(x^*)) &\overset{①}{\leq} \sum_{i=1}^{T}\eta_i(\langle \nabla f(x_i), x_i - x_0\rangle + \langle \nabla f(x_i), x_0 - x^*\rangle) \\
&\overset{②}{\leq} \left(\sum_{i=1}^{T}\frac{1}{2L}\|\eta_i\nabla f(x_i)\|^2 - \frac{1}{2L}\|\sum_{j=1}^{T}\eta_j\nabla f(x_j)\|^2\right) + \left(\frac{1}{2L}\|\sum_{j=1}^{T}\eta_j\nabla f(x_j)\|^2 + \frac{L}{2}\|x_0 - x^*\|^2\right) \\
&\overset{③}{\leq} \sum_{i=1}^{T}\eta_i(f(x_i) - f(x^*)) - \eta_T^2(f(x_{T+1}) - f(x^*)) + \frac{L}{2}\|x_0 - x^*\|^2
\end{aligned} \quad (2.3.7)$$

where ① uses convexity and ③ here cancels two terms and then the rest is the same as ④[5], ⑤ and ⑥ in (2.3.4) but without dropping $-\eta_T^2(f(x_{T+1}) - f(x^*))$. Inequality ② holds by Cauchy-Schwarz and Young's inequality $\langle v, w\rangle \leq \frac{1}{2}\|v\|^2 + \frac{1}{2}\|w\|^2$ on the second summand and for the

[5]Note (CG3) is still satisfied because y_i is on the line Λ_i where we search.

first summand we used the inequality

$$\sum_{i=1}^{T} \eta_i \langle \nabla f(x_i), x_i - x_0 \rangle \overset{①}{=} \sum_{i=1}^{T} \eta_i \langle \nabla f(x_i), y_i - x_0 \rangle \overset{②}{=} \frac{1}{L} \sum_{i=1}^{T} \langle \eta_i \nabla f(x_i), -\sum_{j=1}^{i-1} \eta_j \nabla f(x_j) \rangle$$
$$\overset{③}{=} \sum_{i=1}^{T} \left(\frac{1}{2L} \|\eta_i \nabla f(x_i)\|^2 + \frac{1}{2L} \|\sum_{j=1}^{i-1} \eta_j \nabla f(x_j)\|^2 - \frac{1}{2L} \|\sum_{j=1}^{i} \eta_j \nabla f(x_j)\|^2 \right)$$
$$\overset{④}{=} \sum_{i=1}^{T} \frac{1}{2L} \|\eta_i \nabla f(x_i)\|^2 - \frac{1}{2L} \|\sum_{j=1}^{T} \eta_j \nabla f(x_j)\|^2, \tag{2.3.8}$$

where ① uses that by optimality $\nabla f(x_i)$ is perpendicular to the line Λ_i, and both x_i and y_i are on Λ_i. Similarly, ② adds the direction in the definition of Λ_i. We used the identity $-2\langle v, w \rangle = \|v\|^2 + \|w\|^2 - \|v + w\|^2$ in ③, and we telescoped in ④. We obtain from (2.3.7) that $f(x_{T+1}) - f(x^*) \leq \frac{L\|x_0 - x^*\|^2}{2\eta_T^2}$. We already saw that η_i^2 is $\Omega(i^2)$ so this concludes the analysis. ■

Nemirovski and Yudin (1983a) also proved that an implementation with an approximate line search obtains an ε-minimizer in $O(T \ln T)$ oracle calls if the algorithm with the exact line search reaches the same desired accuracy in T iterations.

2.4 Nesterov's Accelerated Gradient Descent

The algorithms in the previous section worked in the unconstrained setting and with smoothness defined by a norm in a Hilbert space. It is an optimal algorithm up to log factors and constants. Later, Nesterov (1983) provided a method that is optimal, up to constants. In this work, Nesterov (1983) provides a more general result: the algorithm allows for constrained minimization in a convex set $\mathcal{X}$, but the function must be defined over all the space since the algorithm may query points outside of $\mathcal{X}$. This paper also contains an accelerated algorithm for the strongly convex case. Later, Nesterov (1998) generalizes this method so the algorithm does not need to query gradients outside of $\mathcal{X}$. Further, Nesterov (2005) generalizes it so it works for a smooth function with respect to an arbitrary norm. This last version also contains an FTRL subalgorithm. The pseudocode of these three methods is in Algorithms 2, 3 and 4, respectively. We are writing equivalent formulations of the algorithms, which are just slightly different from the original ones. Some of the algorithms originally contained simpler learning rates, just for convenience, and yielded slightly worse results by a constant. The algorithms require ψ to be 1-strongly convex just for simplicity. We can use σ-strongly convex maps and change the learning rate accordingly to obtain the same algorithms.

We note that the alternative choice of y_{k+1} in Line 9 of Algorithm 4, despite of being analogous to the one in Algorithm 3, does not appear in its corresponding work. In fact, the author provides a method that requires an MD step per iteration, on top of the FTRL iteration, in order to avoid the optimization subproblem of the gradient descent step. The trick in Line 9 appears explicitly

Algorithm 2 AGD, version in (Nesterov, 1983)

Input: A Hilbert space $\mathcal{E}$ with norm $\|\cdot\| = \sqrt{\langle\cdot,\cdot\rangle}$. An L-smooth wrt $\|\cdot\|$, convex, differentiable $f : \mathcal{E} \to \mathbb{R}$. Optimization is over $\mathcal{X} \subseteq \mathcal{E}$ but x_k can be outside of $\mathcal{X}$. Initial points $x_0 = y_0 = y_{-1} \in \mathcal{X}$. Sequence $\{\eta_k\}_{k=0}^{\infty}$ in (2.3.1).

Output: $y_T \in \mathcal{X}$ such that $f(y_T) - f(x^*) = O(\frac{L\|y_0 - x^*\|^2}{T^2})$.

1: **for** $k \leftarrow 0$ **to** $T-1$ **do**
2: $\quad a_{k+1} = \eta_{k+1}/L$
3: $\quad x_{k+1} \leftarrow y_k + (a_k - 1)(y_k - y_{k-1})/a_{k+1}$
4: $\quad y_{k+1} \leftarrow \arg\min_{y\in\mathcal{X}} \left\{f(x_{k+1}) + \langle\nabla f(x_{k+1}), y - x_{k+1}\rangle + \frac{L}{2}\|y - x_{k+1}\|^2\right\}$
5: **end for**

in (Cohen, Diakonikolas, and Orecchia, 2018) and it could have been independently developed before. Algorithm 3 appears in (Nesterov, 1998) applying to Hilbert spaces, and in its full generality using Bregman divergences in (Auslender and Teboulle, 2006; Tseng, 2008; Allen-Zhu and Orecchia, 2017) although this extension is very similar to the one in Algorithm 4, which appeared before in (Nesterov, 2005). Nonetheless, to the best of our knowledge (Auslender and Teboulle, 2006) is the first reference that analyzes the MD version of AGD with Bregman divergences, (Tseng, 2008) provides a unified analysis for the three methods and their extensions to composite optimization and (Allen-Zhu and Orecchia, 2017) provides an intuitive analysis showing the algorithm consists of an online learning subalgorithm with an instantaneous regret that is balanced by a GD step.

Algorithm 3 AGD where the MD algorithm estimates regularized lower bounds on f.

Input: Convex, differentiable $f : \mathcal{X} \to \mathbb{R}$ that is L-smooth wrt an arbitrary norm $\|\cdot\|$. Initial points $x_0 = y_0 = z'_0 \in \mathcal{X}$, and 1-strongly convex map $\psi : \mathcal{X} \to \mathbb{R}$ wrt $\|\cdot\|$. Sequence $\{\eta_k\}_{k=0}^{\infty}$ in (2.3.1).

Output: $y_T \in \mathcal{X}$ such that $f(y_T) - f(x^*) = O(\frac{L\|y_0 - x^*\|^2}{T^2})$.

1: **for** $k \leftarrow 0$ **to** $T-1$ **do**
2: $\quad a_{k+1} = \eta_{k+1}/L$
3: $\quad \tau_k = 1/a_{k+1}L = 1/\eta_{k+1}$
4: $\quad x_{k+1} \leftarrow \tau_k z'_k + (1-\tau_k)y_k$
5: $\quad$ Any of the following two (they are equivalent MD formulations):
6: $\quad\quad z'_{k+1} \leftarrow \arg\min_{z\in\mathcal{X}} \left\{\langle\nabla f(x_{k+1}), z\rangle + \frac{1}{a_{k+1}} D_\psi(z, z'_k)\right\}$
7: $\quad\quad z_{k+1} = \nabla\psi(z'_k) - a_{k+1}\nabla f(x_{k+1}); \quad z'_{k+1} \leftarrow \nabla\psi^*(z_{k+1})$
8: $\quad$ Any of the following two (GD updates):
9: $\quad\quad y_{k+1} \leftarrow \tau_k z'_{k+1} + (1-\tau_k)y_k$ $\quad\diamond$ Does not require arg min
10: $\quad\quad y_{k+1} \leftarrow \arg\min_{y\in\mathcal{X}} \left\{\langle\nabla f(x_{k+1}), y\rangle + \frac{L}{2}\|x_{k+1} - y\|^2\right\}$ $\quad\diamond$ Better(unquantified) per-step guarantee
11: **end for**

We now see that in the unconstrained case, this method can be viewed as the one in the previous section, but with a closed form expression for selecting a particular point in the line where the previous method was doing a search. It is easy to prove that the three algorithms are the same in the unconstrained case with the 2-norm if we select $\psi(x) = \frac{1}{2}\|x - x_0\|_2^2$ as regularizer. The version for selecting y_{k+1} does not matter since both options also become equivalent in this

Algorithm 4 AGD where the FTRL algorithm estimates regularized lower bounds on f.

Input: Convex, differentiable $f : \mathcal{X} \to \mathbb{R}$ that is L-smooth wrt an arbitrary norm $\|\cdot\|$. Initial points $x_0 = y_0 = z'_0 \in \mathcal{X}$, $z_0 = \nabla\psi(x_0)$, and 1-strongly convex map $\psi : \mathcal{X} \to \mathbb{R}$ wrt $\|\cdot\|$. Sequence $\{\eta_k\}_{k=0}^{\infty}$ in (2.3.1).

Output: $y_T \in \mathcal{X}$ such that $f(y_T) - f(x^*) = O(\frac{L\|y_0 - x^*\|^2}{T^2})$.

for $k \leftarrow 0$ **to** $T-1$ **do**
  $a_{k+1} = \eta_{k+1}/L$
  $\tau_k = 1/a_{k+1}L = 1/\eta_{k+1}$
  $x_{k+1} \leftarrow \tau_k z'_k + (1-\tau_k)y_k$
  Any of the following two (they are equivalent FTRL formulations):
    $z'_{k+1} \leftarrow \arg\min_{z\in\mathcal{X}} \left\{\sum_{i=0}^{k} a_{i+1}\langle\nabla f(x_{i+1}), z\rangle + D_\psi(z, x_0)\right\}$
    $z_{k+1} = z_k - a_{k+1}\nabla f(x_{k+1})$; $z'_{k+1} \leftarrow \nabla\psi^*(z_{k+1})$
  Any of the following two (GD updates):
    $y_{k+1} \leftarrow \tau_k z'_{k+1} + (1-\tau_k)y_k$ ◇ Does not require arg min
    $y_{k+1} \leftarrow \arg\min_{y\in\mathcal{X}} \left\{\langle\nabla f(x_{k+1}), y\rangle + \frac{L}{2}\|x_{k+1} - y\|^2\right\}$ ◇ Better(unquantified) per-step guarantee
end for

setting. We will not go over the proof of this fact, but note that, MD and FTRL are identical in the unconstrained case, so Algorithm 3 and Algorithm 4 are trivially equivalent in this case. Using this fact we can see that indeed Algorithm 3 is selecting the next point as a particular point in the line $\Lambda_i(\lambda)$ in (2.3.6). Indeed, we can use any of the three versions of AGD for the comparison and we will use Algorithm 4. We already saw in (2.2.7) that FTRL with our ψ and weighted losses $\{a_{i+1}\nabla f(x_{i+1})\}_{i=0}^{k-2}\}$ predicts the point

$$z'_{k-1} = x_0 - \sum_{i=0}^{k-2} a_{i+1}\nabla f(x_{i+1}) = x_0 - \sum_{i=1}^{k-1} \frac{\eta_i}{L}\nabla f(x_i).$$

This is one of the two points in the line $\Lambda_i(\lambda)$ and the other is precisely y_{k-1}. One can prove that the precise convex combination $x_k = \tau_{k-2}z'_{k-1} + (1-\tau_{k-2})y_{k-1}$ is enough to achieve accelerated rates.

The original proof in (Nesterov, 1983) uses an argument through a potential function that it is shown it does not increase with iterations. In the other two algorithms, a similar but more intuitive technique is used in which one can appreciate the algorithm is building lower bounds on the regularized function and that z'_k is the minimum of the regularized lower bound. The arguments using potentials use the potential $\Phi_k = A_k(f(x_k) - f(x^*)) + D_\psi(z'_k, x^*)$, where $A_k \overset{\text{def}}{=} \sum_{i=1}^{k} a_k$, and show $\Phi_{k+1} \leq \Phi_k$ in order to conclude

$$f(x_T) - f(x^*) \leq \frac{\Phi_T}{A_T} \leq \frac{\Phi_0}{A_T} = \frac{A_0(f(x_0) - f(x^*)) + D_\psi(z'_0, x^*)}{A_T} = \frac{D_\psi(z'_0, x^*)}{A_T}.$$

The estimate-sequences technique consists of defining lower bound estimates ϕ_k on the regularized objective. More concretely, these estimates satisfy

$$\phi_k(x) \leq (1-\lambda_k)f(x) + \lambda_k\phi_0(x),$$

where $\lambda_k > 0$ is a parameter that tends to 0 when $k \to \infty$ and ϕ_0 is the initial regularization. Then, the algorithm finds a point y_k such that $f(y_k) \leq \min_{x\in\mathcal{X}} \phi_k(x)$ so that

$$f(y_k) \leq \min_{x\in\mathcal{X}} \phi_k(x) \leq \min_{x\in\mathcal{X}}\{(1-\lambda_k)f(x) + \lambda_k\phi_0(x)\} \leq (1-\lambda_k)f(x^*) + \lambda_k\phi_0(x^*),$$

or equivalently

$$f(y_k) - f(x^*) \leq \lambda_k(\phi_0(x^*) - f(x^*)) \to 0.$$

The regularized lower bounds can be built with the online learning algorithms and then the rest of the algorithm ensures finding y_k such that $f(y_k) \leq \min_{x\in\mathcal{X}} \phi_k(x)$. We will not develop these two arguments, as we will go over similar computations in the following sections, but presented from other points of view.

2.5 Linear Coupling

In this section we provide a proof of accelerated convergence for Algorithm 3 based on (Allen-Zhu and Orecchia, 2017) which motivates the analysis using MD as in (2.2.4). The analysis is in fact similar to the estimate sequences technique by Nesterov (1998) but Allen-Zhu and Orecchia (2017) present a different point of view in which one can see that the online learning algorithm is controlling the regret, and the per-iterate regret is precisely proportional to the progress the GD algorithm makes, so one can compensate one with the other (high regret is balanced with high GD progress and if the GD progress is low, then the instantaneous regret is also low). The authors argue that after performing the GD and the MD steps, one can *couple* both iterates (i.e., compute a convex combination) so that the weights in the combination balance this trade-off. This point of view allows for using weaker versions of online learning algorithms (for instance, relaxing the convexity condition) or for the exploitation of different descent conditions a structure our problem may present.

We will first prove that indeed the regret coming from MD can be bounded by the progress of the descent update plus some Bregman divergence terms. Since y_{k+1} is the GD point, the progress is expressed as $f(x_{k+1}) - f(y_{k+1})$.

Lemma 2.5.1 (MD regret and GD progress). *If $\tau_k = \frac{1}{a_{k+1}L}$, then the iterates of Algorithm 3 satisfy that for every $u \in \mathcal{X}$,*

$$a_{k+1}\langle \nabla f(x_{k+1}), z'_k - u\rangle \leq a_{k+1}^2 L\left(f(x_{k+1}) - f(y_{k+1})\right) + D(u, z'_k) - D(u, z'_{k+1}).$$

Besides, the same condition holds regardless of the value of τ_k if $\mathcal{X} = \mathbb{R}^n$ and y_{k+1} is chosen according to the gradient step in Line 10.

Proof We start by taking the first part of the MD guarantee in Lemma 2.2.8.1:

$$a_{k+1}\langle \nabla f(x_{k+1}), z'_k - u\rangle \leq \langle a_{k+1}\nabla f(x_{k+1}), z'_k - z'_{k+1}\rangle - \frac{1}{2}\|z'_k - z'_{k+1}\|^2 + D_\psi(u, z'_k) - D_\psi(u, z'_{k+1})$$

and we bound the first two summands. Assume first that we defined $y_{k+1} \overset{\text{def}}{=} \tau_k z'_{k+1} + (1-\tau_k) y_k \in \mathcal{X}$ so that $x_{k+1} - y_{k+1} = (\tau_k z'_k + (1-\tau_k) y_k) - (\tau_k z'_{k+1} + (1-\tau_k) y_k) = \tau_k (z'_k - z'_{k+1})$. That is, so $x_{k+1} - y_{k+1}$ is proportional to $z'_k - z'_{k+1}$, which allows for using smoothness to complete our task:

$$\begin{aligned}\langle a_{k+1} \nabla f(x_{k+1}), z'_k - z'_{k+1} \rangle - \frac{1}{2}\|z'_k - z'_{k+1}\|^2 &= \langle \frac{a_{k+1}}{\tau_k} \nabla f(x_{k+1}), x_{k+1} - y_{k+1} \rangle - \frac{1}{2\tau_k{}^2}\|x_{k+1} - y_{k+1}\|^2 \\ &\overset{①}{=} a_{k+1}^2 L \left(\langle \nabla f(x_{k+1}), x_{k+1} - y_{k+1} \rangle - \frac{L}{2}\|x_{k+1} - y_{k+1}\|^2 \right) \\ &\overset{②}{\leq} a_{k+1}^2 L (f(x_{k+1}) - f(y_{k+1})),\end{aligned} \tag{2.5.1}$$

where ① follows by the definition of $\tau_k = 1/a_{k+1}L$ which is chosen so the resulting expression is in the form of the smoothness inequality, which we use in ②. Now if we chose y_k as coming from a gradient step then we would only be increasing the right hand side of the previous guarantee and thus we can also use it, or any point with a better guarantee than y_{k+1} given by the smoothness assumption.

In the unconstrained case we can indeed remove the dependence on τ_k if we define $y_{k+1} = x_{k+1} - \frac{1}{L}\nabla f(x_{k+1})$, i.e., as coming the gradient step. This is because the left hand side in (2.5.1) can be bounded, as we did in Lemma 2.2.8.1, by $a_{k+1}^2 \|\nabla f(x_{k+1})\|_*^2$ and this is proportional to the progress GD makes as derived by smoothness:

$$f(x_{k+1}) - f(y_{k+1}) \geq \langle \nabla f(x_k), x_{k+1} - y_{k+1} \rangle - \frac{L}{2}\|x_{k+1} - y_{k+1}\|^2 = \frac{1}{2L}\|\nabla f(x_{k+1})\|_*^2.$$

Indeed, applying Lemma 2.2.8.1 to ① and the above to ② we obtain

$$\begin{aligned}a_{k+1} \langle \nabla f(x_{k+1}), z'_k - u \rangle &\overset{①}{\leq} \frac{a_{k+1}^2}{2}\|\nabla f(x_{k+1})\|_*^2 + D_\psi(u, z'_k) - D_\psi(u, z'_{k+1}) \\ &\overset{②}{\leq} a_{k+1}^2 L \left(f(x_{k+1}) - f(y_{k+1})\right) + D(u, z'_k) - D(u, z'_{k+1}).\end{aligned} \tag{2.5.2}$$

■

The second part of the lemma gives an extra degree of freedom in the unconstrained case, that allows for generalizations, as we will discuss in Remark 2.5.5.

Remark 2.5.2. *Defining y_{k+1} as the gradient descent point essentially provides us with a point with a lower objective value, which is always better in terms of guarantees. In terms of computational time it could be more expensive in the constrained case due to the projection involved. And if we were going to design a similar algorithm but using stochastic gradients it is intuitive that we should not go farther from x_k than needed in order to guarantee our convergence since we could be amplifying the error in our gradient. In fact, (Cohen, Diakonikolas, and Orecchia, 2018) studied theoretically the algorithm that defines y_{k+1} as a convex combination of y_k and z'_{k+1} under noise (but in the case in which FTRL is used as opposed to MD) and also compared*

it empirically with the choice of defining y_{k+1} as the gradient step point and observed the former performed better.

We now show the coupling lemma. The instantaneous regret can be decomposed into the MD regret and another term, which presents a constant that we can tune to obtain a telescoping sum in the analysis. This constant depends on how far or close we place the next iterate x_{k+1} from the MD point z'_k and from the GD point y_k.

Lemma 2.5.3 (Coupling). *The iterates of Algorithm 3 satisfy, for all $u \in \mathcal{X}$:*

$$a_{k+1}\langle \nabla f(x_{k+1}), x_{k+1} - u\rangle \le C(f(y_k) - f(x_{k+1})) + a_{k+1}\langle \nabla f(x_{k+1}), z'_k - u\rangle.$$

where $C = \frac{(1-\tau_k)a_{k+1}}{\tau_k}$ is a constant that ranges in $[0, \infty)$ depending on the value of $\tau_k \in (0, 1]$.

Proof

$$\begin{aligned} a_{k+1}\langle \nabla f(x_{k+1}), x_{k+1} - u\rangle &\overset{①}{=} a_{k+1}\langle \nabla f(x_{k+1}), x_{k+1} - z'_k\rangle + a_{k+1}\langle \nabla f(x_{k+1}), z'_k - u\rangle \\ &\overset{②}{=} \frac{(1-\tau_k)a_{k+1}}{\tau_k}\langle \nabla f(x_{k+1}), y_k - x_{k+1}\rangle + a_{k+1}\langle \nabla f(x_{k+1}), z'_k - u\rangle \\ &\overset{③}{\le} \frac{(1-\tau_k)a_{k+1}}{\tau_k}(f(y_k) - f(x_{k+1})) + a_{k+1}\langle \nabla f(x_{k+1}), z'_k - u\rangle. \end{aligned}$$

In ①, we decomposed the instantaneous regret into the part that MD controls (second summand) and the rest. Because of the definition of x_{k+1} as a convex combination of y_k and z'_k we can apply equality ② and then ③ by convexity. ■

Now we are ready to prove accelerated convergence rates.

Theorem 2.5.4. *If $f(x)$ is convex and L-smooth w.r.t. $\|\cdot\|$ on $\mathcal{X}$ with a minimizer x^*, and $\psi : \mathcal{X} \to \mathbb{R}$ is 1-strongly convex with respect to $\|\cdot\|$, then Algorithm 3 outputs y_T satisfying*

$$\frac{LD_\psi(x^*, z'_0)}{\eta_T^2} = O\left(\frac{LD_\psi(x^*, z'_0)}{T^2}\right),$$

where η_T is defined by the sequence (2.3.1).

Proof We can derive the theorem from the following inequalities, that hold for all $u \in \mathcal{X}$:

$$\begin{aligned} & a_{k+1}(f(x_{k+1}) - f(u)) \\ \overset{①}{\le}\ & a_{k+1}\langle \nabla f(x_{k+1}), x_{k+1} - u\rangle \\ \overset{②}{\le}\ & (a_{k+1}^2 L - a_{k+1})(f(y_k) - f(x_{k+1})) + a_{k+1}\langle \nabla f(x_{k+1}), z'_k - u\rangle \\ \overset{③}{\le}\ & (a_{k+1}^2 L - a_{k+1})(f(y_k) - f(x_{k+1})) + a_{k+1}^2 L(f(x_{k+1}) - f(y_{k+1})) + D(u, z'_k) - D(u, z'_{k+1}) \\ \overset{④}{=}\ & a_k^2 L f(y_k) - a_{k+1}^2 L f(y_{k+1}) + a_{k+1} f(x_{k+1}) + D(u, z'_k) - D(u, z'_{k+1}). \end{aligned} \tag{2.5.3}$$

Above, ① uses convexity. ② uses Lemma 2.5.3 and we used Lemma 2.5.1 in ③. Note that

$\tau_k = \frac{1}{a_{k+1}L}$ is chosen so we can cancel $f(x_{k+1})$ in both sides of the inequality after ④. In the constrained case we have no other choice, since it is needed by Lemma 2.5.1 but in the unconstrained case this is the value one wants anyway, in order to telescope. Indeed, we now have two telescoping sums on the Bregman divergences and on f evaluated at the sequence of y_k, if we choose a_{k+1}^2 satisfying $a_{k+1}^2 L - a_{k+1} = a_k^2 L$ in ④. This precisely leads to $a_k = \eta_i/L$, where $\{\eta_i\}_{k=1}^{\infty}$ is the sequence in (2.3.1). Note that this inequality has the form of the potential function inequality $\Phi_{k+1} \leq \Phi_k$ we mentioned in Section 2.4 but we obtained it by the observation of the MD and GD having proportional regret and progress that we balance. Now, adding up for $k = 0, \dots, T-1$, simplifying, dropping $-D_\psi(u, z_T') \leq 0$ and using $a_0 = 0$, we obtain

$$-\sum_{k=1}^{T} a_k f(u) \leq -a_T^2 L f(y_T) + D(u, z_0').$$

The sum of the weights on the terms with f was the same on both sides of (2.5.3) so now they must be equal as well, i.e., $\sum_{k=1}^{T} a_k = a_T^2 L$.[6] We obtain the final result by rearranging, setting $u = x^*$, and using the value of $a_T = \eta_T/L = \Omega(T^2/L)$ by (2.3.2). ■

Remark 2.5.5. *In the unconstrained case, we can adapt the previous analysis to work with a smooth function f satisfying a much weaker notion than convexity, namely γ-quasar-convexity, defined by*

$$f(x) - f(x^*) \leq \frac{1}{\gamma}\langle \nabla f(x), x - x^* \rangle \text{ for all } x \in \mathbb{R}^n,$$

for a minimizer x^* of f and parameter $\gamma \geq 1$. In order to adapt the previous method, we note that we can use this condition in substitution for ① in (2.5.3) and then we just need to find a point x_{k+1} in the segment $\tau_k z_k' + (1-\tau_k)y_k$, $\tau_k \in [0,1]$ such that we have an analogous result to Lemma 2.5.1 satisfying

$$\frac{1}{\gamma} a_{k+1}\langle \nabla f(x_{k+1}), x_{k+1} - x^* \rangle \leq C(f(y_k) - f(x_{k+1})) + \frac{1}{\gamma} a_{k+1}\langle \nabla f(x_{k+1}), z_k' - u \rangle,$$

where $C = \frac{1}{\gamma}a_{k+1}^2 L - a_{k+1}$ so that we can cancel $f(x_{k+1})$ on both sides of (2.5.3) as before, since in that case we can set $a_{k+1}^2 L/\gamma - a_{k+1}$ to $a_k^2 L/\gamma$ in order to telescope. This is satisfied by $a_k = \eta_i \gamma/L$ which leads to convergence rates of $O(LD_\psi(x^*, z_0')/(\gamma T^2))$, which is optimal for this class of problems (Hinder, Sidford, and Sohoni, 2020). Note Lemma 2.5.1 does not need any modification in the unconstrained case. In the case of unconstrained convex optimization $C = \frac{(1-\tau_k)a_{k+1}}{\tau_k}$ so we can obtain any C for some value of τ_k. But in fact we can obtain the inequality for any C even without convexity. Indeed, Hinder, Sidford, and Sohoni (2020) do exactly that by reasoning that if the condition is not satisfied in any of the ends of the segment, then by a mean value

[6] Of course, we could have computed these values and actually, we knew this fact from the properties of the sequence $\{\eta_i\}_{k=1}^{\infty}$, but this was a nice invariant to compute it too. The fact that the weights are the same on both sides is only natural since we would expect to have the same inequality if we scale the function by a factor or otherwise we would be getting free lunch. Similarly, when getting rates of convergence, we must obtain expressions that are invariant up to scalings on the function or domain. Like LR/ε, or L/μ or $\mu R/\varepsilon$, where R is the initial distance to a minimizer. This makes for a is a good sanity check.

theorem, a point in the segment must satisfy it, and this point can be approximated with a binary search due to the smoothness of f. Further, the error of this approximation only incurs an extra $\log(L/\varepsilon)$ factor in the rates, similarly to the line search in Section 2.3. We are seeing again a recurring trick in high dimensional optimization, which is that solving a constant-dimensional problem can be done by only adding a $\log$ factor to our complexity and it allows for finding points with some extremal property.

On the other hand, it is not clear how to extend this result to the constrained case. This technique does not allow for it since τ_k plays a role in Lemma 2.5.1 in such a case. Modifying τ_k to obtain any particular C changes the regret coming Lemma 2.5.1 and does not allow to cancel the $f(x_{k+1})$ terms. Intuitively, since the regret is a factor of $\frac{1}{\gamma}$ greater, the GD step would need to be a factor $\frac{1}{\gamma}$ longer which in principle could land outside of the constraint set. In Chapter 3, we prove constrained acceleration for a condition which is between convexity and quasar-convexity by using a different technique, namely implicit Euler discretization of the continuous accelerated dynamics. We overview continuous approaches and their discretizations in the next section. Because of the implicit nature of the algorithm, we are able to compensate a regret that is also a $\frac{1}{\gamma}$ factor greater, while keeping feasibility of the iterates.

In general, as in the remark above, this point of view on acceleration suggests that one could tackle different online learning problems with different algorithms and one could exploit different local descent conditions in order to compensate one with the other and still achieve acceleration. Other examples in the literature are a *three point coupling* (Allen-Zhu, 2017a) used in order to be able to incorporate variance reduction to acceleration in the convex finite-sum setting. See (Allen-Zhu, 2018a) as well, in which this technique is used to minimize a convex function which is an average of non-convex functions. Another family of examples concerns finding stationary points of non-convex functions with Lipschitz gradient and Hessian, in the finite-sum setting as well as approximate local minima (points whose gradient norm is $\leq \varepsilon$ and Hessian's eigenvalues are $\geq -\delta$) (Allen-Zhu, 2017b; Allen-Zhu, 2018b; Fang et al., 2018; Xu, Rong, and Yang, 2018; Allen-Zhu and Li, 2018). In these works, convexity is replaced by $\langle \nabla f(x_{k+1}), x_{k+1} - u \rangle \geq f(x_{k+1}) - f(u) - \frac{\hat{\mu}}{2}\|x_{k+1} - u\|^2$ and an online learning algorithm is used to exploit that condition and to couple it with GD. In Chapter 4, we use MD in the form (2.2.5) and exploit a local decrease condition our problem satisfies in order to achieve acceleration, despite of non-global smoothness and bad local smoothness constant.

2.6 The Approximate Duality Gap Technique

The previous section presented a work that explained the acceleration phenomenon as a coupling of a primal algorithm and an online learning algorithm, which in the case of smooth convex optimization is a dual algorithm. It gave geometrical intuition about why acceleration is possible and has allowed other algorithms to be built for different settings, due to its generality. In this section, we present another, later work that also explains the acceleration phenomenon as a

coupling of primal and dual algorithms but from a continuous point of view: The Approximate Duality Gap Technique (ADGT) (Diakonikolas and Orecchia, 2019b). It is also an interesting work because it presents a general technique from which one can derive with relative naturality most classical first-order convex optimization algorithms. It describes the methods first as continuous methods given by differential equations and then the rates of convergence come from their discretizations and discretization errors. The aim of this technique is to directly obtain an algorithm and a potential function to prove its convergence rates from the natural target of reducing the optimality gap on the continuous case and from computing the discretization error with simple integrators. From this new point of view, an essentially different accelerated algorithm was designed for convex optimization (Diakonikolas and Orecchia, 2018). The main idea consists of taking a backward Euler discretization rule of the differential equation. We note that forward Euler discretization rules essentially yield previous existing algorithms. We will focus on the latter in this section. In Chapter 3 we will present a modified version of the backward Euler discretized method in order to achieve acceleration in a non-convex problem which is related to Riemannian optimization. We note that the technique using the backward Euler discretization uses gradient Lipschitzness as opposed to the definition of smoothness. Recall they are not equivalent in general, so extending this technique to other non-convex settings would require gradient Lipschitzness or other tricks. In our non-convex generalization in Chapter 3 we still have the equivalence between smoothness and gradient Lipschitzness. If a smooth function satisfies that any point with zero gradient is a global minimizer, then the function has Lipschitz gradient, and this condition is satisfied for our problem.

Using this technique we will motivate the derivation of Algorithm 4. We will directly prove convergence rates for this method and we will establish a relationship between this algorithm and Algorithm 3, which entails that a convergence rate for the latter implies at least the same convergence rate for the former. Let α_t be an increasing function of time t, with $\alpha_t = 0$, if $t < 1$. We have $t = 1$ is the initial time. We use Lebesgue-Stieltjes integration and its notation, so that $\int_1^t f(x_\tau)\mathrm{d}\alpha_\tau = \int_1^t f(x_\tau)\dot{\alpha}_\tau \mathrm{d}\tau$. We want to work with continuous and discrete approaches in a unified way. Thus, when α_t is a discrete measure, we have that $\dot{\alpha}_t = \sum_{k=1}^{\infty} a_k \delta(t-k))$ is a weighted sum of Dirac delta functions and a_k are step sizes. We define $A_t \stackrel{\text{def}}{=} \int_1^t \mathrm{d}\alpha_\tau = \int_1^t \dot{\alpha}_\tau \mathrm{d}\tau$. In discrete time, it is $A_t = \sum_{k=1}^{\lfloor t \rfloor} a_k = \alpha_t$. In the continuous case note that we have $\alpha_t - A_t = \alpha_1$.

Let y_t be the solution constructed by the algorithm at time t. The continuous method visits the points x_t and has access to $\nabla f(x_t)$. Note y_t does not need to be equal to x_t. We define the duality gap $G_t \stackrel{\text{def}}{=} U_t - L_t$ as the difference between a differentiable upper bound U_t on the function at the current point y_t, and a differentiable lower bound L_t on $f(x^*)$. Since in our case f is differentiable, we use $U_t \stackrel{\text{def}}{=} f(y_t)$. The idea is to enforce the invariant $\frac{d}{dt}(\alpha_t G_t) = 0$, so we have $f(x_t) - f(x^*) \leq G_t = G_1\alpha_1/\alpha_t$ at any time. Combining the convexity inequality for all the points visited by the continuous method we have a lower bound:

$$f(x^*) \geq \frac{\int_1^t f(x_\tau) d\alpha_\tau}{A_t} + \frac{\int_1^t \langle \nabla f(x_\tau), x^* - x_\tau \rangle d\alpha_\tau}{A_t}.$$

However, this lower bound requires the knowledge of x^*. We could compute a looser lower bound by taking the minimum over $u \in \mathcal{X}$ of this expression, substituting x^* by u. However, this would make the lower bound be non-differentiable and we could have problems at $t = 1$. In order to solve the first problem, we first add a regularizer and then take the minimum over $u \in \mathcal{X}$.

$$f(x^*)+\frac{D_\psi(x^*,x_1)}{A_t} \geq \frac{\int_1^t f(x_\tau)d\alpha_\tau}{A_t} + \frac{\min_{u\in\mathcal{X}}\left\{\int_1^t\langle\nabla f(x_\tau), u-x_\tau\rangle d\alpha_\tau + D_\psi(u,x_1)\right\}}{A_t}$$

Let $z_t \stackrel{\text{def}}{=} \nabla\psi(x_1) - \int_1^t \nabla f\left(x_\tau\right) \mathrm{d}\alpha_\tau$. Then, by Fact 2.1.6, and since ψ is strongly convex, we have that $\nabla\psi^*(z_t)$ is the argmin of the expression above. This is analogous to what one obtains with FTRL, and we are essentially bounding the function plus a regularizer that decreases as fast as the convergence rates we aim to prove. Here, the stability added by the regularizer is in the form of differentiability. In order to solve the second problem, we mix this lower bound with the optimal lower bound $f(x^*)$ with weight $\alpha_t - A_t$ (this is only necessary in continuous time, in discrete time this term is 0). Not knowing $f(x^*)$ or $D_\psi(x^*, x_1)$ will not be problematic. Indeed, we only need to guarantee $\frac{d}{dt}(\alpha_t G_t) = 0$. After taking the derivative, these terms will vanish. After rescaling the normalization factor, we finally obtain the differentiable lower bound

$$\begin{aligned} f(x^*) \geq L_t \stackrel{\text{def}}{=} \frac{\int_1^t f(x_\tau)\mathrm{d}\alpha_\tau}{\alpha_t} + \frac{\min_{u\in\mathcal{X}}\left\{\int_1^t\langle\nabla f(x_\tau), u-x_\tau\rangle\mathrm{d}\alpha_\tau + D_\psi(u,x_1)\right\}}{\alpha_t} \\ + \frac{(\alpha_t - A_t)f(x^*) - D_\psi(x^*,x_1)}{\alpha_t}. \end{aligned} \tag{2.6.1}$$

We can now compute

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}(\alpha_t G_t) &= \frac{\mathrm{d}}{\mathrm{d}t}(\alpha_t f(x_t)) - \dot{\alpha}_t(f(x_t) + \langle\nabla f(x_t), \nabla\psi^*(z_t) - x_t\rangle) \\ &= \langle\nabla f(x_t), \alpha_t\dot{x}_t - \dot{\alpha}_t(\nabla\psi^*(z_t) - x_t)\rangle. \end{aligned}$$

We can make the right hand side be 0 by defining the continuous method as starting with $x_1 \in \mathcal{X}, z_1 = \nabla\psi(x_1)$ and having

$$\dot{z}_t = -\dot{\alpha}_t\nabla f(x_t); \quad \dot{x}_t = \dot{\alpha}_t\frac{\nabla\psi(z_t) - x_t}{\alpha_t}. \tag{2.6.2}$$

Now we discretize these dynamics, which will determine the convergence rate of the algorithm. In particular, denote the discretization error of our method by E_k, so that $A_kG_k - A_{k-1}G_{k-1} = E_k$. Then

$$G_T \leq \frac{A_1}{A_T}G_1 + \frac{\sum_{k=1}^{T-1} E_{k+1}}{A_T}.$$

In the discrete case we have to trade off between rapidly increasing α_t and having low discretization error. In this case we will aim to obtain non-positive discretization error. So in a way, on could see this approach as a potential based analysis with potential $A_kG_k (\leq A_{k-1}G_{k-1} \leq \cdots \leq A_1G_1)$. But this point of view has allowed us to obtain the potential function naturally. Suppose now that α_t is a discrete measure, as explained above. The discretization error E_k is incurred because of different integral values in the continuous and discrete cases in the integral under the

minimum of the lower bound, which essentially comes from discontinuities in the updates: we make x_k depend on z_{k-1} in the discrete case. We can compute the discretization error at one step by taking the difference between the integral with our discrete measure and the one with a continuous measure. In the discrete case between time $k-1$ and k we have that $\dot{\alpha}_\tau$ samples the function under the integral at time k so we have

$$\begin{aligned}
E_k &= (-a_k\langle\nabla f(x_k), \nabla\psi^*(z_k) - x_k)\rangle) - \left(-\int_{k-1}^{k}\langle\nabla f(x_\tau), \nabla\psi^*(z_k) - x_\tau\rangle \mathrm{d}\alpha_\tau\right) \\
&\overset{①}{=} -a_k\langle\nabla f(x_k), \nabla\psi^*(z_k) - x_k)\rangle + \int_{k-1}^{k}\langle\nabla f(x_\tau), \alpha_\tau\dot{x}_\tau\rangle \mathrm{d}\tau + \int_{k-1}^{k}\langle -\dot{z}_\tau, \nabla\psi^*(z_k) - \nabla\psi^*(z_\tau)\rangle \mathrm{d}\tau \\
&\overset{②}{=} -a_k\langle\nabla f(x_k), \nabla\psi^*(z_k) - x_k)\rangle + A_{k-1}(f(x_k) - f(x_{k-1})) - D_{\psi^*}(z_{k-1}, z_k) \\
&\overset{③}{\leq} \langle\nabla f(x_k), A_k x_k - A_{k-1}x_{k-1} - a_k\nabla\psi^*(z_k)\rangle - D_{\psi^*}(z_{k-1}, z_k).
\end{aligned} \tag{2.6.3}$$

In ①, we used both equations in (2.6.2). In ②, we integrated the first integral by parts and the computation of the second integral reduces to noting that it equals $\int_{k-1}^{k}\frac{\mathrm{d}}{\mathrm{d}\tau}D_{\psi^*}(z_\tau, z_k)\mathrm{d}\tau$ by Lemma 2.1.8.3. Finally, in ③ we used convexity of f on the second summand. We note we could have computed $A_kG_k - A_{k-1}G_{k-1}$ directly to arrive to the same bound. In the next subsection, in which we analyze Algorithm 3 through this point of view, we will do exactly that for the sake of demonstrating this alternative.

Using the Euler discretization we obtain the update rule $x_k = \frac{A_{k-1}}{A_k}x_{k-1} + \frac{a_k}{A_k}\nabla\psi^*(z_{k-1})$ which makes the first term of the discretization error be $a_k\langle\nabla f(x_k), \nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\rangle$. Now the idea is to take an extra gradient step to reduce the upper bound and compensate for the discretization error. The algorithm then becomes Algorithm 4:

$$x_k = \frac{A_{k-1}}{A_k}y_{k-1} + \frac{a_k}{A_k}\nabla\psi^*(z_{k-1}); \quad z_k = z_{k-1} - a_k\nabla f(x_k); \quad y_k = \text{Line 9 or Line 10}, \tag{2.6.4}$$

for $x_0 = y_0 \in \mathcal{X}, z_0 = \nabla\psi(x_0)$. Note $x_1 = x_0$ and that we have shifted the indices to account for the Euler discretization update. The GD point y_k is taken as in the previous section. The gradient step only changes the upper bound, from $U_k = f(x_k)$ to $U_k = f(y_k)$. Thus, the new discretization error is

$$\begin{aligned}
E_k &\leq A_k(f(y_k) - f(x_k)) + a_k\langle\nabla f(x_k), \nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\rangle - D_{\psi^*}(z_{k-1}, z_k), \\
&\overset{①}{\leq} A_k(f(y_k) - f(x_k)) + a_k\langle\nabla f(x_k), \nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\rangle - \frac{1}{2}\|\nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\|^2,
\end{aligned} \tag{2.6.5}$$

In ①, we bounded the Bregman divergence by using Lemma 2.1.8.6 and Lemma 2.1.8.2. Now, if we use smoothness we will be able to cancel the first summand of E_k above with the other two. Indeed the smoothness inequality is

$$f(y_k) - f(x_k) \leq \langle\nabla f(x_k), y_k - x_k\rangle + \frac{L}{2}\|y_k - x_k\|^2. \tag{2.6.6}$$

If we have $y_k = x_k - \frac{a_k}{A_k}\left(\nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\right) = \frac{A_{k-1}}{A_k}y_{k-1} + \frac{a_k}{A_k}\nabla\psi^*(z_k) \in \mathcal{X}$ (which is the choice y_k =Line 9) we obtain from (2.6.6):

$$A_k(f(y_k) - f(x_k)) \leq -a_k\langle\nabla f(x_k), \nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\rangle + \frac{La_k^2}{2A_k}\|\nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\|^2, \tag{2.6.7}$$

and so the discretization error is bounded as

$$E_k \leq \left(\frac{La_k^2}{2A_k} - \frac{1}{2}\right)\|\nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\|^2.$$

Any point y_k with a lower right hand side of (2.6.6), like the gradient step in Line 10, would yield at least the same guarantee. Finally, recall that $A_k = \sum_{i=1}^k a_i$, so if we set $a_k = \frac{\eta_k}{L}$ we satisfy $\frac{a_k^2}{A_k} = \frac{1}{L}$ and thus $E_k \leq 0$ implying $G_T \leq a_1 G_1/A_T$. Bounding the initial gap and using $A_T = a_T^2 L$, we obtain an optimal convergence bound. Using the definition of U_1 and L_1, we have

$$\begin{aligned} a_1 G_1 &= a_1 U_1 - a_1 L_1 \\ &= a_1 f(y_1) - \Big(a_1 f(x_1) - a_1\langle\nabla f(x_1), \nabla\psi^*(z_1) - x_1\rangle - D_\psi(\nabla\psi^*(z_1), x_1) + D_\psi(x^*, x_1)\Big) \\ &\overset{①}{\leq} D_\psi(x^*, x_1), \end{aligned}$$

where in ① we used the same as what we used in (2.6.5) and (2.6.7) for E_k. Finally using $A_T = a_T^2 L = \frac{\eta_T^2}{L}$, we conclude

$$f(y_T) - f(x^*) \leq G_T \leq \frac{a_1 G_1}{A_T} \leq \frac{LD_\psi(x^*, x_1)}{\eta_T^2} = \frac{LD_\psi(x^*, x_0)}{\eta_T^2}.$$

2.6.1 Incorporating Mirror Descent

Diakonikolas and Orecchia (2019b) analyzed acceleration with the approximate duality gap technique using FTRL. We now we adapt the analysis for the MD version. We will do this by showing a relationship between MD and FTRL that essentially means that MD builds looser regularized lower bounds than FTRL. This analysis provides further intuition about these two online learning algorithms.

Starting at an arbitrary point $\nabla\psi^*(z_0) = x_0$, we have that for $k \geq 0, \ldots, T-1$, Algorithm 3 performs the following steps:

$$\begin{aligned} x_{k+1} &\leftarrow \frac{a_{k+1}}{A_{k+1}}\nabla\psi^*(z_k) + \frac{A_k}{A_{k+1}}y_k \\ z_{k+1} &\leftarrow \nabla\psi(z'_k) - a_{k+1}\nabla f(x_{k+1}) \\ z'_{k+1} &\leftarrow \arg\min_{u\in\mathcal{X}}\left\{a_{k+1}\langle\nabla f(x_{k+1}), u - x_{k+1}\rangle + D_\psi(u, \nabla\psi^*(z_k))\right\} (= \nabla\psi^*(z_{k+1})) \\ y_{k+1} &\leftarrow \text{Line 9 or Line 10 of Algorithm 3} \end{aligned} \tag{2.6.8}$$

We also define $V_{k+1} \overset{\text{def}}{=} \min_{u\in\mathcal{X}}\left\{a_{k+1}\langle\nabla f(x_{k+1}), u - x_{k+1}\rangle + D_\psi(u, \nabla\psi^*(z_k))\right\}$ as the minimum of which $z'_{k+1} = \nabla\psi^*(z_{k+1})$ is arg min. We modify the ADGT argument above by using a

different lower bound. Just for the sake of showing both arguments, this time we will directly work with a discrete measure α_t such that $\dot{\alpha}_t = \sum_{k=1}^{\infty} a_k \delta(t-k)$ and will directly compute the discretization error. But we note that we would obtain exactly the same bound if we used continuous arguments. The lower bound on $f(x^*)$ we use is the following:

$$L_t \stackrel{\text{def}}{=} \frac{1}{A_t}\left(\sum_{k=1}^{t} a_k f(x_k) + \sum_{k=1}^{t} V_k - D_\psi(x^*, x_1)\right). \tag{2.6.9}$$

It is a lower bound on $f(x^*)$ because we can show it is looser than the bound in Equation (2.6.1) we used before. We just need to sequentially apply Lemma 2.6.1. Note $\alpha_t = A_t$ because we are in the discrete case. The bound is the following:

$$\begin{aligned}
(2.6.1) &= \frac{1}{A_t}\left(\sum_{k=1}^{t} a_k f(x_k) + \min_{u\in\mathcal{X}}\left\{\sum_{k=1}^{t}\langle a_k \nabla f(x_k), u - x_k\rangle + D_\psi(u, x_1)\right\} - D_\psi(x^*, x_1)\right) \\
&\stackrel{①}{\geq} \frac{1}{A_t}\left(\sum_{k=1}^{t} a_k f(x_k) + \sum_{k=1}^{t} V_k + \min_{u\in\mathcal{X}}\{D_\psi(u, \nabla\psi^*(z_t))\} - D_\psi(x^*, x_1)\right) \\
&\stackrel{②}{=} \frac{1}{A_t}\left(\sum_{k=1}^{t} a_k f(x_k) + \sum_{k=1}^{t} V_k - D_\psi(x^*, x_1)\right),
\end{aligned}$$

where ① uses $x_1 = z_0' = \nabla\psi^*(z_0)$ and then it uses Lemma 2.6.1 sequentially t times and ② uses the fact that the minimum of the Bregman divergence $D_\psi(u, \nabla\psi^*(z_t))$ is reached at $\nabla\psi^*(z_t)$ and its value is 0.

Lemma 2.6.1. *For all $u \in \mathcal{X}$ we have*

$$a_k\langle \nabla f(x_k), u - x_k\rangle + D_\psi(u, \nabla\psi^*(z_{k-1})) \geq V_k + D_\psi(u, \nabla\psi^*(z_k)).$$

Proof Using that by definition $V_k = a_k\langle \nabla f(x_k), \nabla\psi^*(z_k) - x_k\rangle + D_\psi(\nabla\psi^*(z_k), \nabla\psi^*(z_{k-1}))$ and reorganizing terms we obtain the following equivalent inequality

$$a_k\langle \nabla f(x_k), u - \nabla\psi^*(z_k)\rangle + D_\psi(u, \nabla\psi^*(z_{k-1})) - D_\psi(u, \nabla\psi^*(z_k)) - D_\psi(\nabla\psi^*(z_k), \nabla\psi^*(z_{k-1})) \geq 0$$

And this is equivalent, by the triangle equality of Bregman divergences Lemma 2.1.8.5, to:

$$\langle \nabla\psi(\nabla\psi^*(z_k)) - \nabla\psi(\nabla\psi^*(z_{k-1})) + a_k\nabla f(x_k), u - \nabla\psi^*(z_k)\rangle \geq 0.$$

Since we have $\nabla\psi^*(z_k) = \arg\min_{x\in\mathcal{X}}\{-\langle x, z_k\rangle + \psi(x)\}$ by Fact 2.1.6, then by the first-order optimality condition of the Fenchel dual $\psi^*(z_k) \stackrel{\text{def}}{=} \min_{x\in\mathcal{X}}\{-\langle x, z_k\rangle + \psi(x)\}$ we have ① below:

$$\begin{aligned}
0 &\stackrel{①}{\leq} \langle -z_k + \nabla\psi(z_k'), u - z_k'\rangle \\
&\stackrel{②}{=} \langle -\nabla\psi(z_{k-1}') + a_{k-1}\nabla f(x_k) + \nabla\psi(z_k'), u - z_k'\rangle \geq 0, \\
&\stackrel{③}{=} \langle -\nabla\psi(\nabla\psi^*(z_{k-1})) + a_k\nabla f(x_k) + \nabla\psi(\nabla\psi^*(z_k)), u - \nabla\psi^*(z_k)\rangle,
\end{aligned}$$

where ② holds by the definition of z_k, and in ③ we used $z_k' = \nabla\psi^*(z_k)$ and $z_{k-1}' = \nabla\psi^*(z_{k-1})$.

■

Now we compute the discretization error

$$
\begin{aligned}
E_k &\overset{①}{=} A_k G_k - A_{k-1} G_{k-1} \\
&= A_{k-1}(f(x_k) - f(y_{k-1})) + a_k f(x_k) + A_k(f(y_k) - f(x_k)) \\
&\quad - \sum_{i=1}^{k} a_i f(x_i) - \sum_{i=1}^{k} V_i + \sum_{i=1}^{k-1} a_i f(x_i) + \sum_{i=1}^{k-1} V_i \\
&\overset{②}{\leq} A_{k-1}\langle \nabla f(x_k), x_k - y_{k-1}\rangle - V_k + A_k(f(y_k) - f(x_k)) \\
&\overset{③}{=} A_{k-1}\langle \nabla f(x_k), x_k - y_{k-1}\rangle - a_k\langle \nabla f(x_k), \nabla\psi^*(z_k) - x_k\rangle - D_\psi(\nabla\psi^*(z_k), \nabla\psi^*(z_{k-1})) \\
&\quad + A_k(f(y_k) - f(x_k)). \\
&\overset{④}{\leq} A_k(f(y_k) - f(x_k)) + a_k\langle \nabla f(x_k), \nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\rangle - D_\psi(\nabla\psi^*(z_k), \nabla\psi^*(z_{k-1})).
\end{aligned}
$$

In ① we write out the definition of the gap, taking into account we keep the upper bound $U_k = f(y_k)$, but we add and subtract $A_k f(x_k)$ to make comparisons with respect to the upper bound $f(x_k)$, as in the previous analysis. In ②, we use convexity on the first summand and cancel some terms. In ③ we write the definition of V_k with its argmin $\nabla\psi^*(z_k)$. ④ reorganizes terms and uses equality $A_k x_k = A_{k-1} y_{k-1} + a_k \nabla\psi^*(z_{k-1})$, which is the definition of x_k.

Now, if we take into account the inequalities in Lemma 2.1.8.6 and Lemma 2.1.8.2, namely

$$
-D_{\psi^*}(z_{k-1}, z_k) \leq -D_\psi(\nabla\psi^*(z_k), \nabla\psi^*(z_{k-1})) \leq -\frac{1}{2}\|\nabla\psi^*(z_{k-1}) - \nabla\psi^*(z_k)\|^2,
$$

we arrive at the same bound for the discretization error as in (2.6.5), so the rest of the convergence proof is identical. We note that the looser regularized lower bound used in this case translates to having a negative term using the Bregman divergence $-D_\psi(\nabla\psi^*(z_k), \nabla\psi^*(z_{k-1}))$, as opposed to the term in the previous analysis $-D_{\psi^*}(z_{k-1}, z_k)$. This is again an instance of the fact that looser losses lose more. However, we note that the fact that the regularized lower bound of the algorithm using MD is looser than the one using FTRL does not mean that the actual lower bound that could be guaranteed on $f(x^*)$ is worse. The reduction of the regularized lower bound can indeed be due to a decrease in the regularizer, that we are not accounting for, which would actually improve convergence (the term $D_\psi(x^*, x_0)$ could be substituted by something with a lower value). In any case, this analysis suggests that convergence rates of general accelerated methods based on MD would automatically yield convergence rates for analogous accelerated methods based on FTRL as long as the original algorithm does not use some other specific properties of the MD update. Note that the analysis of the method in this subsection (Algorithm 3) automatically implies the convergence of the previous method based on FTRL (Algorithm 4) with rates that are at least as good.

2.7 Others

We conclude by commenting on some other techniques and algorithms that we do not cover in full detail. As we advanced in the introduction, our list is not comprehensive, and we do not focus on how these and all the previously presented techniques generalize to settings beyond smooth and convex optimization. Our focus instead is on presenting the key ideas of accelerated techniques to the simplest possible setting.

Optimized gradient method

Kim and Fessler (2016) used some relaxations of techniques developed by Drori and Teboulle (2014) to obtain an accelerated method, the optimized gradient method, that improves by a constant over the guarantee of AGD. This convergence was later seen to be optimal in Drori (2017), even considering constants. Both the upper and lower bounds were originally obtained by optimizing the worst case guarantee within a family of first-order methods in which the iterates are linear combinations of the initial point and gradients of past points, but where the coefficients are independent on the function. Finding the algorithm with the best guarantee in the worst case can be described as an optimization problem. The dual of such problem yields a hard instance. Even though the optimization problem does not seem to be efficiently solvable, performing some relaxations allows for efficient optimization and it suffices to find matching upper and lower bounds on performance with respect to several metrics, considering constants (Drori and Teboulle, 2014; Kim and Fessler, 2016; Taylor, Hendrickx, and Glineur, 2017; Drori, 2017). The technique has been applied to other settings. For instance, it has been applied to the strongly convex case in order to obtain the exact complexity, considering constants, of the first-order black-box optimization problem (Drori and Taylor, 2021; Taylor and Drori, 2021).

The optimized gradient method method coincides with unconstrained AGD with smoothness defined by $\|\cdot\|_2$ (note the versions we provided in Section 2.4 are equivalent in this case) except for a slightly different choice of the learning rate. While in AGD we choose $a_{k+1} = \eta_{k+1}/L$ and $\tau_k = 1/\eta_{k+1}$ for $k = 0, \dots, T-1$ (cf. Algorithm 4), in the optimized gradient method one chooses $a_{k+1} = 2\eta_{k+1}/L$ for $k = 0, \dots, T-1$ and $\tau_k = 1/\eta_{k+1}$ for $k = 0, \dots, T-2$ and for $k = T-1$ it is $\tau_{T-1} = 1/\eta'_T$ where $\eta'_T = (1 + \sqrt{1 + 8\eta_{T-1}^2})/2$. As a comparison, we had $\eta_T = (1 + \sqrt{1 + 4\eta_{T-1}^2})/2$ in AGD.

As with Nemirovski's quasi-AGD and with Nesterov's AGD, this method also has an interpretation with an analysis coming from the analysis of CGD we presented in Section 2.3, but using a slightly modified learning rate, cf. (d'Aspremont, Scieur, and Taylor, 2021; Drori and Taylor, 2020).

Optimistic online learning and acceleration

Optimistic online learning is essentially a way of solving online learning that consists of guessing what the next loss is going to be and pretend it is the actual next loss when computing

the prediction. For instance, if we were to use FTRL we would predict the arg min of the regularized sum of losses plus the guess. In order to perform a reduction to convex optimization one feeds weighted gradients $a_k \nabla f(x_k)$ to the algorithm. Typical regret bounds have the form:

$$R_T \leq r(x^*) + \frac{1}{2}\sum_{k=1}^{T} a_k^2 \|\nabla f(x_k) - \tilde{g}_k\|_*^2. \tag{2.7.1}$$

where $r(x^*)$ is a strongly convex regularizer evaluated at x^*, coming from the regularization we used in the algorithm. We also used $\tilde{g}_k$ to denote the guess of the loss that later was revealed to be $\nabla f(x_k)$. If we know a bound on the magnitude of $\nabla f(x_k)$ (for instance, if f is Lipschitz), then making a wrong prediction all of the time only incurs a factor of 2 in the regret with respect to the non-optimistic approach, but being consistently good at predicting the losses can reduce the regret significantly. Guesses can be usually obtained from some structure of our problem. In accelerated smooth convex optimization one uses $\tilde{g}_k = \nabla f(x_{k-1})$. In this context Joulani, Raj, et al. (2020) use an anytime reduction, meaning that they have guarantees for the last iterate (as opposed to a weighted average, for example), and they use FTRL with optimism and show that they achieve acceleration.

The optimistic approach is interesting because it reduces the convergence analysis completely to the regret analysis of the online learning algorithm with linearized losses. Another advantage of this approach is that it readily generalizes to stochastic optimization, composite optimization or both. In some settings, one can also make the algorithm universal, meaning that it achieves smoothness and non-smoothness optimal rates, up to logarithmic factors, without knowledge of the smoothness parameter of the problem.

Proximal point algorithms and acceleration

We use $\|\cdot\| = \|\cdot\|_2$ and $\mathcal{X} = \mathbb{R}^n$ in this section for simplicity. We define the proximal operator of a convex function f as

$$\operatorname{prox}_{\lambda f}(x) \stackrel{\text{def}}{=} \arg\min_y \left\{ f(y) + \frac{1}{2\lambda}\|y - x\|^2 \right\},$$

where $\lambda \in \mathbb{R}_{>0}$ is a parameter. Also, the Moreau envelope of f is defined as

$$M_{\lambda f} \stackrel{\text{def}}{=} \min_y \left\{ f(y) + \frac{1}{2\lambda}\|y - x\|^2 \right\} = (f^* + \frac{\lambda}{2}\|\cdot\|^2)^*$$

This function is a smoothed version of f with smoothness parameter $\frac{1}{\lambda}$ or lower. From the second definition using Fenchel duality and using that an L-smooth function f has $\frac{1}{L}$-strongly convex Fenchel dual and vice versa, we have that $f^* + \frac{\lambda}{2}\|\cdot\|^2$ is $(\frac{1}{L} + \lambda)$-strongly convex and $M_{\lambda f}$ is smooth with parameter $\frac{1}{1/L+\lambda} \leq \frac{1}{\lambda}$. Because of the formulation above as a Fenchel dual, we have that the Moreau envelope is convex. Some other useful properties of the Moreau envelope and the proximal operator are $\nabla M_{\lambda f} = (x - \operatorname{prox}_{\lambda f}(x))/\lambda$, which can be seen to hold by Fact 2.1.6. Also, we can see the proximal operator as an implicit gradient step $\operatorname{prox}_{\lambda f}(x) = x - \lambda \nabla f(\operatorname{prox}_{\lambda f}(x))$,

cf. (Parikh and Boyd, 2014). The Moreau envelope also satisfies that their minimizers are the same minimizers as those of f. Indeed, if x^* is a minimizer of f then trivially $\operatorname{prox}_{\lambda f}(x^*) = x^*$ and so $M_{\lambda f}(x^*) = f(x^*)$. And if x is not a minimizer of f then 0 is not a subgradient of $f(y) + \frac{1}{2\lambda}\|y - x\|^2$ at $y = x$ and thus $\operatorname{prox}_{\lambda f}(x) \neq x$ and then x is not a minimizer of $M_{\lambda f}$ because $M_{\lambda f}(x^*) = f(x^*) < f(\operatorname{prox}_{\lambda f}(x)) + \frac{1}{2\lambda}\|\operatorname{prox}_{\lambda f}(x) - x\|^2 = M_{\lambda f}(x)$. Consequently if we fix λ we could apply AGD to this function and we would obtain an ε-minimizer in $O(\frac{\|x^* - x_0\|^2}{\lambda T^2})$ iterations. The caveat is that we have to compute the proximal operator at each iteration and this subproblem could be expensive, specially if λ is large. Interestingly, one can make use of different λ_k at iteration k and one can solve the proximal subproblem inexactly and still obtain acceleration, like in the approaches in (Monteiro and Svaiter, 2013). Indeed, we can use the lower bound $f(x^*) \geq M_{\lambda_k f}(x) + \langle \nabla M_{\lambda_k f}(x), x^* - x\rangle$ for any λ_k for our online learning problem that estimates $f(x^*)$. Further we can use a gradient step of $M_{\lambda_k f}$ from x in order to balance the regret, as in Linear Coupling. The subproblems are strongly convex, so they can be approximately solved fast. This framework provides great flexibility for the design of accelerated first-order methods.

Let's use the ADGT to derive an algorithm by (Güler, 1992) that uses these lower bounds by defining an upper bound $U_k = f(y_k)$ where $y_k = \operatorname{prox}_{\lambda_k f}(x_k)$ is a gradient point, coming from the aforementioned implicit gradient step, that is, $y_k = x_k - \lambda_k \nabla f(y_k)$. The points x_k will be chosen later. Similarly, define a lower bound combining regularization with what convexity provides for the Moreau envelope at the points x_k:

$$\begin{aligned} f(x^*) \geq L_k &\stackrel{\text{def}}{=} \frac{1}{A_k}\left(\sum_{i=1}^k a_i M_{\lambda_i f}(x_i) + \min_{u\in\mathbb{R}^n}\left\{\sum_{i=1}^k a_i \langle \nabla M_{\lambda_i f}(x_i), u - x_i\rangle + \frac{1}{2}\|u - x_1\|^2\right\}\right. \\ &\qquad \left. - \frac{1}{2}\|x^* - x_1\|^2\right) \\ &= \frac{1}{A_k}\left(\sum_{i=1}^k a_i (f(y_i) + \frac{1}{2\lambda_i}\|x_i - y_i\|^2) + \left(\sum_{i=1}^k a_i \langle \nabla f(y_i), z_i - x_i\rangle + \frac{1}{2}\|z_i - x_1\|^2\right)\right. \\ &\qquad \left. - \frac{1}{2}\|x^* - x_1\|^2\right). \end{aligned}$$

which as in the ADGT section, it comes from a convex combination of the convexity inequalities, adding and subtracting the strongly convex regularizer and taking a minimum in order to remove the relevant dependence of the lower bound on x^*. We defined the notation $z_k = x_1 - \sum_{i=1}^k a_i \nabla f(y_i)$ for the solution of the FTRL subproblem in the lower bound, that is the $\arg\min$ in the minimization problem above. We will choose the values of a_k later, and the algorithm will depend on the choices of λ_k. Recall $A_k = \sum_{i=1}^k a_k$ and that we can update $z_k = z_{k-1} - a_k \nabla f(y_k)$.

It is easy to see, as in the previous continuous analysis of the ADGT, that the derivative of $\alpha_t G_t$ can be made 0 if point x_t satisfies $\alpha_t \dot{x}_t = \dot{\alpha}_t(z_t - x_t)$, which ultimately in the discrete case leads to $A_k x_k = A_{k-1} y_{k-1} + a_k z_{k-1}$, although it will also be clear from the computation of the

discretization error E_k that this choice cancels the right terms. We compute E_k now:

$$
\begin{aligned}
E_k &= A_k G_k - A_{k-1} G_{k-1} \\
&\overset{①}{=} A_{k-1}(f(y_k) - f(y_{k-1})) + a_k f(y_k) \\
&\quad - \left(\sum_{i=1}^{k} a_i \left(f(y_i) + \frac{1}{2\lambda_i}\|y_i - x_i\|^2 + \langle \nabla f(y_i), z_k - x_i \rangle \right) + \frac{1}{2}\|z_k - x_1\|^2 \right) \\
&\quad + \sum_{i=1}^{k-1} a_i \left(f(y_i) + \frac{1}{2\lambda_i}\|y_i - x_i\|^2 + \langle \nabla f(y_i), z_{k-1} - x_i \rangle \right) + \frac{1}{2}\|z_{k-1} - x_1\|^2 \\
&\overset{②}{\leq} A_{k-1}(f(y_k) - f(y_{k-1})) - a_k \left(\frac{1}{2\lambda_k}\|y_k - x_k\|^2 + \langle \nabla f(y_k), z_k - x_k \rangle \right) - \frac{1}{2}\|z_k - z_{k-1}\|^2 \\
&\overset{③}{\leq} \langle \nabla f(y_k), A_{k-1}x_k - A_{k-1}y_{k-1} - a_k z_{k-1} + a_k x_k \rangle + \|\nabla f(y_k)\|^2 \left(-A_{k-1}\lambda_k - \frac{a_k \lambda_k}{2} + a_k^2 - \frac{a_k^2}{2} \right).
\end{aligned}
$$

In the first line after ① we regroup the upper bound terms so it is clear we cancel all of the terms $a_i f(y_i)$ in ②. The second and third lines of ① contain the weighted lower bounds. We canceled the terms $\frac{1}{2}\|x^* - x_1\|^2$. In ② , we extract the summand with index k in the second line after ① and compare what remains in lines 2 and 3: the FTRL objective occurring at step $k-1$ evaluated at the minimizer z_{k-1} and at the point z_k. Because this FTRL problem is 1-strongly convex, we have that the difference is bounded by $-\frac{1}{2}\|z_k - z_{k-1}\|^2$ (note this is precisely one of the reasons why we use FTRL with a strongly convex regularizer in the first place). In ③, the aim is to group all of the terms with $\langle \nabla f(y_k), \cdot \rangle$ and with $\|\nabla f(y_k)\|^2$, respectively. In order to do that, for the first summand we used convexity and the equality $y_k = x_k - \lambda_k \nabla f(y_k)$. The second summand uses the latter equality and $z_k = z_{k-1} - a_k \nabla f(y_k)$. Finally, the third summand uses the latter equality. At this point it is clear that defining x_k so it satisfies $A_k x_k = A_{k-1} y_{k-1} + a_k z_{k-1}$ will cancel the first term of the discretization error after ③. Recall that $A_k = A_{k-1} + a_k = \sum_{i=1}^{k} a_i$. We can now maximize the choice of a_k in order to have a fast convergence rate while keeping the second summand non-positive. Thus, we obtain $a_k = (\lambda_k + \sqrt{\lambda_k^2 + 8A_{k-1}\lambda_k})/2$.

Finally analyzing the initial gap similarly to what we did in Section 2.6 we obtain $A_1 G_1 \leq \frac{1}{2}\|x^* - x_1\|^2$ and so $f(y_T) - f(x^*) \leq G_T \leq \frac{\|x^* - x_1\|^2}{2A_T}$. We could we solve the proximal subproblems inexactly, say that we compute y_k satisfying $y_k = x_k - \lambda_k \nabla f(y_k) + r_k$, for $\|r_k\| \leq \|x_k - y_k\|$, as opposed to $y_k = x_k - \lambda_k \nabla f(y_k)$. Performing a similar analysis we can obtain a greater error term multiplying $\|\nabla f(y_k)\|^2$ and still get non-positive discretization error at the expense of using a slightly smaller learning rate a_k. We will not elaborate the details of this extension, cf. (Monteiro and Svaiter, 2013). We note that the parameters λ_k can be arbitrarily large but in such case the subproblems become harder.

Below, we summarize the algorithm with the exact proximal operator. For $x_0 = z_0$ and for $A_k = \sum_{i=1}^{k} a_k$, $a_k = (\lambda_k + \sqrt{\lambda_k^2 + 8A_k\lambda_k})/2$, we compute:

$$
x_{k+1} = \frac{A_k}{A_{k+1}} y_k + \frac{a_{k+1}}{A_{k+1}} z_k; \quad y_{k+1} = \operatorname{prox}_{\lambda_k f}(x_{k+1}); \quad z_{k+1} = z_k - a_{k+1}\nabla f(y_{k+1}); \text{ for } k = 0 \text{ to } T-1.
$$

We note that one can obtain an accelerated method for smooth convex optimization by solving each (strongly-convex) subproblem in linear time with a first-order method, like GD, cf. (Lin, Mairal, and Harchaoui, 2015).

Some other variants and generalizations exist. There are several other notions of inexact solutions of the proximal problem. The proximal operator can be generalized so it is defined with respect to a general Bregman divergence $D_\psi(\cdot, x)$ as opposed to with $\frac{1}{2}\|\cdot - x\|^2$. For a Euclidean norm, (Carmon, Jambulapati, et al., 2020) use a ball optimization oracle, whose solution coincides with the proximal operator, for some λ. Then they apply a similar approach as in (Monteiro and Svaiter, 2013) and compute bounds on the corresponding λ's. The ball optimization oracle can be implemented fast for problems with a stable Hessian in a ball around each point, among others.

Chapter 3

Acceleration in First-Order Riemannian Optimization

In this chapter, we study the acceleration phenomenon on Riemannian manifolds and introduce a global first-order method that achieves the same rates as accelerated gradient descent in the Euclidean space for the optimization of smooth and geodesically convex (g-convex) or strongly g-convex functions defined on the hyperbolic space or a subset of the sphere, up to constants and log factors. This is the first method that is proved to achieve these rates globally on functions defined on a Riemannian manifold $\mathcal{M}$ other than the Euclidean space. Previous works, on the one hand, did not achieve acceleration under g-convexity. On the other hand, under strong g-convexity previous works only accelerate in a small neighborhood of the minimizer (Zhang and Sra, 2018), or they have global rates which in the worse case equal the iterations that Riemannian Gradient Descent takes to reach the aforementioned neighborhood plus the iterations of the local accelerated algorithm Ahn and Sra, 2020.[1]

As a proxy, we solve a constrained non-convex Euclidean problem in an accelerated way, under a condition between convexity and *quasar-convexity*, of independent interest. Additionally, for any Riemannian manifold of bounded sectional curvature, we provide reductions from optimization methods for smooth and g-convex functions to methods for smooth and strongly g-convex functions and vice versa.

3.1 Introduction

As we have seen in the previous chapter, acceleration in convex optimization is a phenomenon that has drawn plenty of attention and has yielded many important results, since the renowned Accelerated Gradient Descent (AGD) method of Nesterov (1983). In fact, recent efforts towards understanding the acceleration phenomenon have yielded numerous new results going *beyond convexity or the standard oracle model*, in a wide variety of settings (Allen-Zhu, 2017a; Allen-Zhu, 2018b; Allen-Zhu, 2018a; Allen-Zhu and Orecchia, 2019; Allen-Zhu, Qu, et al., 2016; Allen-Zhu,

[1] These two results essentially comes from exploiting the fact that despite the deformations of the geometry, if we restrict the optimization to a small neighborhood of the minimizer then these deformations are small and a strongly geodesically-convex function still satisfies the Polyak-Lojasiewicz inequality (Karimi, Nutini, and Schmidt, 2016) if viewed, via the inverse exponential map, from the tangent space of any point in the neighborhood.

Li, et al., 2017; Carmon, Duchi, et al., 2017; Cohen, Diakonikolas, and Orecchia, 2018; Cutkosky and Sarlós, 2019; Diakonikolas and Jordan, 2021; Diakonikolas and Orecchia, 2018; Gasnikov et al., 2019b; Wang, Rao, and Mahoney, 2016). This surge of research that applies tools of convex optimization to models going beyond convexity has been fruitful. One of these models is the setting of geodesically convex Riemannian optimization. In this setting, the function to optimize is geodesically convex (g-convex), i.e. convex restricted to any geodesic (cf. Definition 3.1.1).

Riemannian optimization, g-convex and non-g-convex alike, is an extensive area of research. In recent years there have been numerous efforts towards obtaining Riemannian optimization algorithms that share analogous properties to the more broadly studied Euclidean first-order methods: deterministic (de Carvalho Bento, Ferreira, and Melo, 2017; Wei et al., 2016; Zhang and Sra, 2016), stochastic (Hosseini and Sra, 2020; Khuzani and Li, 2017; Tripuraneni et al., 2018), variance-reduced (Sato, Kasai, and Mishra, 2019a; Sato, Kasai, and Mishra, 2019b; Zhang, Reddi, and Sra, 2016), adaptive (Kasai, Jawanpuria, and Mishra, 2019), saddle-point-escaping (Criscitiello and Boumal, 2019; Sun, Flammarion, and Fazel, 2019; Zhang, Zhang, and Sra, 2018; Zhou, Yuan, and Feng, 2019; Criscitiello and Boumal, 2020), projection-free (Weber and Sra, 2017; Weber and Sra, 2019), and online learning methods (Wang, Tu, et al., 2021), among others. Unsurprisingly, Riemannian optimization has found many applications in machine learning, including low-rank matrix completion (Cambier and Absil, 2016; Heidel and Schulz, 2018; Mishra and Sepulchre, 2014; Tan et al., 2014; Vandereycken, 2013; Hou, Li, and Zhang, 2020), dictionary learning (Cherian and Sra, 2017; Sun, Qu, and Wright, 2017; Bai, Jiang, and Sun, 2019), optimization under orthogonality constraints (Edelman, Arias, and Smith, 1998), with applications to Recurrent Neural Networks (Lezcano-Casado, 2019; Lezcano-Casado and Martínez-Rubio, 2019), robust covariance estimation in Gaussian distributions (Wiesel, 2012), Gaussian mixture models (Hosseini and Sra, 2015), operator scaling (Allen-Zhu, Garg, et al., 2018) and generalizations coming from symmetries of non-commutative groups (Bürgisser et al., 2019), projection robust optimal transport (Lin, Zheng, et al., 2021) and sparse principal component analysis (Genicot, Huang, and Trendafilov, 2015; Huang and Wei, 2021; Jolliffe, Trendafilov, and Uddin, 2003).

However, the acceleration phenomenon, largely celebrated in the Euclidean space, is still not understood in Riemannian manifolds, although there has been some progress on this topic recently (cf. Related work). This poses the following question, which we study in this chapter:

Can a Riemannian first-order method enjoy the same rates as AGD does in the Euclidean space?

We provide an answer in the affirmative for functions defined on hyperbolic and spherical spaces, up to constants depending on the sectional curvature constant K and the initial distance to a minimizer R, and up to log factors. For $K > 0$ these constants are a small polynomial on $1/\cos(R\sqrt{|K|})$ and for $K < 0$ they are a small polynomial on $\cosh(R\sqrt{|K|})$, the latter being an exponential dependence. We discuss these constants in Section 3.7, which present a difference

with respect to the Euclidean case. We note that if $R\sqrt{|K|} = O(1)$ we obtain the same rates as AGD up to log factors while previous works could only achieve this full acceleration when starting at a small neighborhood satisfying $R\sqrt{|K|} = O((L/\mu)^{-3/4})$. The main results of this chapter are the following.

Main Results

- *Full acceleration.* We design algorithms that provably achieve the same rates of convergence as AGD in the Euclidean space, up to constants and log factors. More precisely, we obtain the rates $\widetilde{O}(L/\sqrt{\varepsilon})$ and $O^*(\sqrt{L/\mu}\log(\mu/\varepsilon))$ when obtaining ε-minimizers of L-smooth functions that are, respectively, g-convex and μ-strongly g-convex, defined on the hyperbolic space or a subset of the sphere. The notation $\widetilde{O}(\cdot)$ and $O^*(\cdot)$ omits $\log(L/\varepsilon)$ and $\log(L/\mu)$ factors, respectively, and constants with respect to K and R. Previous approaches only showed local results (Zhang and Sra, 2018) or obtained results with rates in between the ones obtainable by Riemannian Gradient Descent (RGD) and AGD (Ahn and Sra, 2020). Moreover, these previous works only apply to functions that are smooth and strongly g-convex and not to smooth functions that are only g-convex. As a proxy, we design an accelerated algorithm under a condition between convexity and *quasar-convexity* in the constrained setting, which may be of independent interest.

- *Reductions.* We present two reductions for any Riemannian manifold of bounded sectional curvature. Given an optimization method for smooth and g-convex functions they provide a method for optimizing smooth and strongly g-convex functions, and vice versa.

It is often the case that methods and key geometric inequalities that apply to manifolds with bounded sectional curvatures are obtained from the ones existing for the spaces of constant extremal sectional curvature (Grove, Petersen, and Levy, 1997; Zhang and Sra, 2016; Zhang and Sra, 2018). Consequently, our contribution is relevant not only because we establish an algorithm achieving global acceleration on functions defined on a manifold other than the Euclidean space, but also because understanding the constant sectional curvature case is an important step towards understanding the more general case of obtaining algorithms that optimize g-convex functions, strongly or not, defined on manifolds of bounded sectional curvature.

Structure of this chapter

We provide some definitions, notations, and related work in the rest of this section. We introduce our algorithms and their ideas in Section 3.2, along with a sketch of the main optimization proof. Section 3.3 contains the reductions from g-convex smooth problems to strongly g-convex smooth problems. Section 3.5 contains the proofs of convergence of the accelerated algorithms. Section 3.3 contains the proofs of the reductions and the corollaries showing how to apply them to our algorithms. In Section 3.6, we prove our geometric lemmas that show how to reduce our

Riemannian problem to the Euclidean non-convex problem that we solve in an accelerated way. In Section 3.7 we comment on the constants of our algorithm and on hardness results.

Basic Geometric Definitions

We recall basic definitions of Riemannian geometry that we use in this chapter. For a thorough introduction we refer to (Petersen, Axler, and Ribet, 2006). A Riemannian manifold $(\mathcal{M}, \mathfrak{g})$ is a real smooth manifold $\mathcal{M}$ equipped with a metric $\mathfrak{g}$, which is a smoothly varying inner product. For $x \in \mathcal{M}$ and any two vectors $v, w \in T_x\mathcal{M}$ in the tangent space of $\mathcal{M}$, the inner product $\langle v, w\rangle_x$ is $\mathfrak{g}(v, w)$. For $v \in T_x\mathcal{M}$, the norm is defined as usual $\|v\|_x \stackrel{\text{def}}{=} \sqrt{\langle v, v\rangle_x}$. Typically, x is known given v or w, so we will just write $\langle v, w\rangle$ or $\|v\|$ if x is clear from context. A geodesic is a curve $\gamma : [0, 1] \to \mathcal{M}$ of unit speed that is locally distance minimizing. A uniquely geodesic space is a space such that for every two points there is one and only one geodesic that joins them. In such a case the exponential map $\operatorname{Exp}_x : T_x\mathcal{M} \to \mathcal{M}$ and inverse exponential map $\operatorname{Exp}_x^{-1} : \mathcal{M} \to T_x\mathcal{M}$ are well defined for every pair of points, and are as follows. Given $x, y \in \mathcal{M}$, $v \in T_x\mathcal{M}$, and a geodesic γ of length $\|v\|$ such that $\gamma(0) = x$, $\gamma(1) = y$, $\gamma'(0) = v/\|v\|$, we have that $\operatorname{Exp}_x(v) = y$ and $\operatorname{Exp}_x^{-1}(y) = v$. Note, however, that $\operatorname{Exp}_x(\cdot)$ might not be defined for each $v \in T_x\mathcal{M}$. We denote by $d(x, y)$ the distance between x and y. Its value is the same as $\|\operatorname{Exp}_x^{-1}(y)\|$. Given a 2-dimensional subspace $V \subseteq T_x\mathcal{M}$, the sectional curvature at x with respect to V is defined as the Gauss curvature of the manifold $\operatorname{Exp}_x(V)$ at x.

Notation

Let $\mathcal{M}$ be a n-dimensional Riemannian manifold. Given two points $x, y \in \mathcal{M}$ and a vector $v \in T_x\mathcal{M}$ in the tangent space of x, we use the formal notation $\langle v, y - x\rangle \stackrel{\text{def}}{=} -\langle v, x - y\rangle \stackrel{\text{def}}{=} \langle v, \operatorname{Exp}_x^{-1}(y)\rangle$. We call $F : \mathcal{M} \to \mathbb{R}$ a function we want to optimize and that has at least one global minimum at x^*. We denote by x_0 an initial point, in $\mathcal{M}$, of an optimization algorithm. We denote $R \geq d(x_0, x^*)$ a bound on the initial distance to x^*. We use the notation $\operatorname{Exp}_{x_0}(\bar{B}(0, R)) \subset \mathcal{M}$ to mean that $\mathcal{M}$ is such that Exp_{x_0} is defined on the closed ball $\bar{B}(0, R) \subset T_{x_0}\mathcal{M}$. We use $\mathcal{M}_K$ to denote any manifold that is a subset of an n-dimensional complete and simply connected manifold of constant sectional curvature K, namely a subset of the hyperbolic space or sphere (Petersen, Axler, and Ribet, 2006), with the inherited metric, and such that $\operatorname{Exp}_{x_0}(\bar{B}(0, R)) \subset \mathcal{M}_K$. In such a case, we use $\mathcal{B}_R$ for $\operatorname{Exp}_{x_0}(\bar{B}(0, R))$. Note that we are not making explicit the dependence on n, x_0, $\mathcal{M}_K$, and K. We want to work with the standard choice of uniquely geodesic manifolds (Ahn and Sra, 2020; Liu, Shang, et al., 2017; Zhang and Sra, 2016; Zhang and Sra, 2018). Therefore, if $K > 0$ we restrict ourselves to $R < \pi/2\sqrt{K}$, so $\mathcal{B}_R$ is uniquely geodesic (it is contained in an open hemisphere). Note that by definition $x^* \in \mathcal{B}_R$. For $M \subseteq \mathbb{R}^n$, we denote by $h : \mathcal{M} \to M$ a geodesic map (Kreyszig, 1991), which is a diffeomorphism such that the image and the inverse image of a geodesic is a geodesic. Unless specified otherwise, we will have $h(x_0) = \mathbb{0}_n$. Given a point $x \in \mathcal{M}$ we use the notation $\tilde{x} \stackrel{\text{def}}{=} h(x)$ and vice versa; any point in M will use a tilde. Given a vector $v \in T_x\mathcal{M}$, we call $\tilde{v} \in \mathbb{R}^n$ the vector of the same norm such

that $\{\tilde{x}+\tilde{\lambda}\tilde{v}|\tilde{\lambda}\in\mathbb{R}^+, \tilde{x}+\tilde{\lambda}\tilde{v}\in M\} = \{h(\mathrm{Exp}_x(\lambda v))|\lambda\in I\subseteq\mathbb{R}^+\}$, for some interval I. Likewise, given x and a vector $\tilde{v}\in\mathbb{R}^n$, we define $v\in T_x\mathcal{M}$. In the case of $\mathcal{M}_K$, we call $\mathcal{X} = h(\mathcal{B}_R)$. The big O notations $\widetilde{O}(\cdot)$ and $O^*(\cdot)$ omit $\log(L/\varepsilon)$ and $\log(L/\mu)$ factors, respectively, and constant factors depending on R and K.

We define now the main properties that will be assumed on the function F to be minimized.

Definition 3.1.1 (Geodesic Convexity and Smoothness). *Let $F:\mathcal{M}\to\mathbb{R}$ be a differentiable function defined on a Riemannian manifold $(\mathcal{M},\mathfrak{g})$. Given $L\geq\mu>0$, we say that F is L-smooth, and respectively μ-strongly g-convex, if for any two points $x,y\in\mathcal{M}$, F satisfies*

$$F(y)\leq F(x)+\langle\nabla F(x), y-x\rangle+\frac{L}{2}d(x,y)^2,\ \textit{resp.}\ F(y)\geq F(x)+\langle\nabla F(x), y-x\rangle+\frac{\mu}{2}d(x,y)^2.$$

We say F is g-convex if the second inequality above, i.e. μ-strong g-convexity, is satisfied with $\mu=0$. We have used the formal notation above for the subtraction of points in the inner product.

Our main technique consists of mapping the function domain to a subset M of the Euclidean space via a geodesic map h. Given the gradient of a point $x\in\mathcal{M}$, convexity defines a lower bound on the function that is affine over the tangent space of x, namely $\ell(y) = F(x)+\langle\nabla F(x), y-x\rangle\leq F(y)$ and it implies a minimizer must be in the halfspace $H=\{y|\langle\nabla F(x), y-x\rangle\leq 0\}$, since $\ell(\cdot)$ is greater than $F(x)$ outside of H. This lower bound induces, via the geodesic map, a function on M. And H is mapped to a halfspace H' in the Euclidean space, because $\{h(y)|\langle\nabla F(x), y-x\rangle=0\}$ is mapped to a hyperplane by the definition of geodesic map. We find a lower bound of $\ell\circ h^{-1}$ that is affine over H' and such that it is equal to $F(x)$ at $h(x)$, despite the geodesic map being non-conformal, deforming distances, and breaking convexity, cf. Lemma 3.2.2. This allows to aggregate the lower bounds easily in the Euclidean space by taking an average, in the same spirit as mirror dual averaging algorithms do. We believe that effective lower bound aggregation is key to achieving Riemannian acceleration and optimality and it has been the main hurdle of previous algorithms. Using this strategy, we are able to define a continuous method that we discretize using an approximate implementation of the implicit Euler method, achieving the same rates as the Euclidean AGD, up to constants and log factors, for the optimization of g-convex smooth functions. Our reductions take into account the deformations produced by the geometry to generalize existing optimal Euclidean reductions (Allen-Zhu and Hazan, 2016; Allen-Zhu and Orecchia, 2017). Applying them, we obtain an analogous algorithm for strongly g-convex and smooth functions. Applying them again to the latter they yield an algorithm for g-convex smooth functions with the rates of the same order as the first one.

Comparison with Related Work.

There are a number of works that study the problem of first-order acceleration in Riemannian manifolds of bounded sectional curvature. The first study is (Liu, Shang, et al., 2017). In this work, the authors develop an accelerated method with the same rates as AGD for both g-convex and strongly g-convex functions, provided that at each step a given non-linear equation can

be solved. No algorithm for solving this equation has been found and, in principle, it could be intractable or infeasible. In (Alimisis et al., 2020) a continuous method analogous to the continuous approach to accelerated methods is presented, but it is not known if there exists an accelerated discretization of it. In (Alimisis et al., 2021), an algorithm presented is claimed to enjoy an accelerated rate of convergence, but fails to provide convergence when the function value gets below a potentially large constant that depends on the manifold and smoothness constant. The work (Lin, Saparbayeva, et al., 2020) is inspired by accelerated algorithms and focuses on adapting to the strong g-convex parameter but does not obtain accelerated algorithms. In (Huang and Wei, 2019) an accelerated algorithm is presented but relying on strong geometric inequalities that are not proved to be satisfied. Zhang and Sra (2018) obtain a *local* algorithm that optimizes L-smooth and μ-strongly g-convex functions achieving the same rates as AGD in the Euclidean space, up to constants. That is, the initial point needs to start close to the optimum, $O((\mu/L)^{3/4})$ close, to be precise[2]. Their approach consists of adapting Nesterov's estimate sequence technique by keeping a quadratic on $T_{x_i}\mathcal{M}$ that induces on $\mathcal{M}$ a regularized lower bound on $F(x^*)$ via $\text{Exp}_{x_i}(\cdot)$. They build another lower bound by aggregating the information yielded by the gradient $\nabla F(x_i)$ to it, and use a geometric lemma to find a quadratic in $T_{x_{i+1}}\mathcal{M}$ whose induced function lower bounds the previous one. Ahn and Sra (2020) generalize the previous algorithm and, by using similar ideas for the lower bound, they adapt it to work globally, obtaining strictly better rates than RGD, recovering the local acceleration of the previous paper, but not achieving global rates comparable to the ones of AGD. In fact, they prove that their algorithm eventually decreases the function value at a rate close to AGD but this can take as many iterations as the ones needed by RGD to reach the neighborhood of the previous local algorithm. We take a step back and focus on the constant sectional curvature case to provide a global algorithm that achieves the same rates as AGD, up to constants on R, K, and log factors. It is common to characterize the properties of spaces of bounded sectional curvature by using the ones of the spaces of constant extremal sectional curvature (Grove, Petersen, and Levy, 1997; Zhang and Sra, 2016; Zhang and Sra, 2018), which makes the study of the constant sectional curvature case critical to the development of full accelerated algorithms in the general bounded sectional curvature case. We also study g-convexity besides strong g-convexity. No previous accelerated algorithms applied to this case. Because of the hardness of the geometry, our convergence rates incur greater constants depending on R and K with respect to the Euclidean case. They are a small polynomial on $1/\cos(R\sqrt{|K|})$ in spherical spaces and $\cosh(R\sqrt{|K|})$ in hyperbolic spaces, the latter being an exponential dependence. It is not clear if these constants can be avoided in fully accelerated algorithms and we provide a discussion about this in Section 3.7. We note

[2]This result essentially comes from exploiting the fact that despite the deformations of the geometry, if we restrict the optimization to a small neighborhood of the minimizer then these deformations are small and a strongly geodesically-convex function still satisfies the Polyak-Lojasiewicz inequality (Karimi, Nutini, and Schmidt, 2016) with constant $O(\frac{L}{\mu})$ if viewed, via the inverse exponential map, from the tangent space of any point in the neighborhood. This does not hold true outside of the neighborhood

Hamilton and Moitra (2021) showed a hardness result for the hyperbolic case with a strongly g-convex function, if the function and gradient oracle is noisy, cf. Remark 3.7.3.

There are some related works that study *Euclidean optimization*. One of which is the *approximate duality gap technique* (Diakonikolas and Orecchia, 2019b), which presents a unified view of the analysis of first-order methods. It defines a continuous duality gap and by enforcing a natural invariant, it obtains accelerated continuous dynamics and their discretizations for most classical first-order methods. A derived work (Diakonikolas and Orecchia, 2018) obtains acceleration in a fundamentally different way from previous acceleration approaches, namely using an approximate implicit Euler method for the discretization of the acceleration dynamics. Our convergence analysis of Theorem 3.2.4 draws ideas from these two works. We will see in the sequel that, for our manifolds of interest, g-convexity is related to a model known as quasar-convexity or weak-quasi-convexity (Guminov and Gasnikov, 2017; Guminov, Nesterov, et al., 2019; Hinder, Sidford, and Sohoni, 2020).

Remark 3.1.2 (Rates of related work). *The local algorithm in (Zhang and Sra, 2018) requires starting $O((L/\mu)^{-3/4})$ close to the optimum and it finds an ε-minimizer in $O(\sqrt{L/\mu}\log(\mu/\varepsilon))$. On the other hand RGD has a convergence rate of $O(L/\mu\log(\mu/\varepsilon))$. Hence, we could run both algorithms in parallel and restart them every few iterations from the best of the two points that both algorithms yielded. In that case we would obtain the convergence rate $O^*(L/\mu+\sqrt{L/\mu}\log(\mu/\varepsilon))$. Indeed, note that we would just compute twice as many gradients as if we run RGD but we perform as well as if we first run RGD until it gets into the desired neighborhood and then we run the local accelerated algorithm. And by μ-strong g-convexity we can guarantee we are $\mu\bar{\varepsilon}^2/2$-close to a minimizer if we are at an $\bar{\varepsilon}$-minimizer so if we set $\bar{\varepsilon}$ so that $\mu\bar{\varepsilon}^2/2 = O((L/\mu)^{-3/4})$ and run RGD we reach the neighborhood after $O((L/\mu)\log(\mu(L/\mu)^{3/4}))$ iterations.*

We note that this mix of RGD and the local algorithm in (Zhang and Sra, 2018) enjoys the same worse case guarantee of (Ahn and Sra, 2020). This latter work is a generalization of (Zhang and Sra, 2018) that eventually accelerates, but a careful look at the paper and analysis reveals that in order to reach accelerated rates it needs as much time as RGD takes to reach the accelerating neighborhood of (Zhang and Sra, 2018). Indeed, they can guarantee that for their iterates y_t, their algorithm converges at an accelerated rate $f(y_t) - f(x^*) \leq O(f(y_{t-1}) - f(x^*))(1-\sqrt{\mu/L}))$ when $t = \Omega^*(\frac{1}{\log(1/\lambda)}) = \Omega^*(L/\mu)$, where $\lambda = \Omega(1-\mu/L)$. A summary of rates is presented in Table 3.1.

3.2 Algorithm

We study the minimization problem $\min_{x\in\mathcal{M}_K} F(x)$ with a gradient oracle, for a twice differentiable smooth function $F : \mathcal{M}_K \to \mathbb{R}$ that is g-convex or strongly g-convex, for an initial point x_0, a minimizer x^* of F that is assumed to exist, and a constant $R > d(x_0, x^*)$. We recall $\mathcal{M}_K$ refers to any manifold of constant non-zero sectional curvature such that $\mathcal{B}_R = \mathrm{Exp}_{x_0}(\bar{B}(0,R)) \subset \mathcal{M}_K$. We work in this setting in this entire section. We perform optimization restricted to $\mathcal{B}_R$

Table 3.1: Rates of related works for different problems. AGD is a Euclidean algorithm. Our Algorithm 5 works in manifolds of constant sectional curvature $K \neq 0$. The rest of the algorithms work in manifolds of sectional curvature that is bounded above and below.

Method	**μ-st. g-convex**	**g-convex**
AGD (Nesterov, 1983)	$O(\sqrt{L/\mu}\log(\mu/\varepsilon))$	$O(\sqrt{L/\varepsilon})$
(Zhang and Sra, 2018) (only works locally)	$O(\sqrt{L/\mu}\log(\mu/\varepsilon))$	-
(Ahn and Sra, 2020)	$O^*(L/\mu + \sqrt{L/\mu}\log(\mu/\varepsilon))$	-
(Ours) Remark 3.1.2 (RGD+(Zhang and Sra, 2018))	$O^*(L/\mu + \sqrt{L/\mu}\log(\mu/\varepsilon))$	-
(Ours) Corollary 3.3.2 and Algorithm 5 resp.	$O^*(\sqrt{L/\mu}\log(\mu/\varepsilon))$	$\widetilde{O}(\sqrt{L/\varepsilon})$

in order to control the deformations caused by the geometry. We defer the proofs of the lemmas and theorems in this and the following section to later sections. We assume without loss of generality that the sectional curvature of $\mathcal{M}_K$ is $K \in \{1, -1\}$, since for any other value of K and any function $F : \mathcal{M}_K \to \mathbb{R}$ defined on such a manifold, we can reparametrize F by a rescaling, so it is defined over a manifold of constant sectional curvature $K \in \{1, -1\}$. The parameters L, μ and R are rescaled accordingly as a function of K, cf. Remark 3.6.1. We denote the special cosine by $\mathrm{C}_K(\cdot)$, which is $\cos(\cdot)$ if $K = 1$ and $\cosh(\cdot)$ if $K = -1$. For a geodesic map $h : \mathcal{M} \to M$, we define $\mathcal{X} \stackrel{\text{def}}{=} h(\mathcal{B}_R) \subseteq M \subseteq \mathbb{R}^n$. We use classical geodesic maps for the manifolds that we consider: the Gnomonic projection for $K = 1$ and the Beltrami-Klein projection for $K = -1$ (Greenberg, 1993). They map an open hemisphere and the hyperbolic space of curvature $K \in \{1, -1\}$ to $\mathbb{R}^n$ and $B(0, 1) \subseteq \mathbb{R}^n$, respectively. We will derive our results from the following characterization of h (Greenberg, 1993). Let $\tilde{x}, \tilde{y} \in \mathcal{X}$ be two points. Recall that we denote $x = h^{-1}(\tilde{x}), y = h^{-1}(\tilde{y}) \in \mathcal{B}_R$. Then we have that $d(x, y)$, the distance between x and y with the metric of $\mathcal{M}_K$, satisfies

$$\mathrm{C}_K(d(x, y)) = \frac{1 + K\langle \tilde{x}, \tilde{y}\rangle}{\sqrt{1 + K\|\tilde{x}\|^2} \cdot \sqrt{1 + K\|\tilde{y}\|^2}}. \tag{3.2.1}$$

Observe that the expression is symmetric with respect to rotations. In particular, $\mathcal{X}$ is a closed ball of some radius $\tilde{R}$. Using $\tilde{x} = 0$ and $\tilde{y}$ such that $d(x_0, y) = R$, we have $\mathrm{C}_K(R) = (1 + K\tilde{R}^2)^{-1/2}$.

Consider a point $x \in \mathcal{B}_R$ and the lower bound provided by the g-convexity assumption when computing $\nabla F(x)$. Dropping the μ term in case of strong g-convexity, this bound is affine over $T_x\mathcal{B}_R$. In order to define a duality gap, as we show in Section 3.2.1, we would like our algorithm to aggregate effectively the lower bounds it computes during the course of the optimization. The deformations of the geometry make the aggregation a difficult task, despite the fact that we have a simple description of each individual lower bound: each of them is affine over $T_{x_i}\mathcal{B}_R$ but these simple functions are defined on different tangent spaces. We deal with this problem by obtaining a lower bound that is looser by a constant depending on R, and that is affine over $\mathcal{X} \subset \mathbb{R}^n$. In

this way the aggregation becomes easier: all of them are simple and are in the same space. Then, we are able to combine this lower bound with decreasing upper bounds in the fashion some other accelerated methods work in the Euclidean space (Allen-Zhu and Orecchia, 2017; Diakonikolas and Orecchia, 2018; Diakonikolas and Orecchia, 2019b; Nesterov, 1983). Alternatively, we can see the approach in this chapter as the constrained optimization problem of minimizing the non-convex function $f : \mathcal{X} \to \mathbb{R}$, $\tilde{x} \mapsto F(h^{-1}(\tilde{x}))$

$$\text{minimize} \quad f(\tilde{x}), \qquad \text{for } \tilde{x} \in \mathcal{X}.$$

In the rest of the section, we will focus on the g-convex case. For simplicity, instead of solving the strongly g-convex case directly in an analogous way by finding a lower bound that is quadratic over $\mathcal{X}$, we rely on the reductions of Section 3.3 to obtain the accelerated algorithm in this case.

The following two lemmas show that finding the affine lower bound is possible, and is defined as a function of $\nabla f(\tilde{x})$. We first gauge the deformations caused by the geodesic map h. Distances are deformed, the map h is not conformal, and the image of the geodesic $\operatorname{Exp}_x(\lambda \nabla F(x))$ is not mapped into the image of the geodesic $\tilde{x} + \tilde{\lambda} \nabla f(\tilde{x})$, i.e. the direction of the gradient changes. We are able to find the affine lower bound after bounding these deformations.

Lemma 3.2.1. [↓] *Let $K \in \{1, -1\}$. Let $x, y \in \mathcal{B}_R$ be two different points, and in part b) different from x_0. Let $\tilde{\alpha}$ be the angle $\angle \tilde{x}_0 \tilde{x} \tilde{y}$, formed by the vectors $\tilde{x}_0 - \tilde{x}$ and $\tilde{y} - \tilde{x}$. Let α be the corresponding angle, the one between the vectors $\operatorname{Exp}_x^{-1}(x_0)$ and $\operatorname{Exp}_x^{-1}(y)$. Assume without loss of generality that $\tilde{x} \in \operatorname{span}\{\tilde{e}_1\}$ and $\nabla f(\tilde{x}) \in \operatorname{span}\{\tilde{e}_1, \tilde{e}_2\}$ for the canonical orthonormal basis $\{\tilde{e}_i\}_{i=1}^n$. Let $e_i \in T_x \mathcal{M}_K$ be the unit vector such that h maps the image of the geodesic $\operatorname{Exp}_x(\lambda e_i)$ to the image of the geodesic $\tilde{x} + \tilde{\lambda} e_i$, for $i = 1, \dots, n$, and $\lambda, \tilde{\lambda} \geq 0$. Then, the following holds.*

a) Distance deformation:

$$K\, \mathrm{C}_K^2(R) \leq K \frac{d(x, y)}{\|\tilde{x} - \tilde{y}\|} \leq K.$$

b) Angle deformation:

$$\sin(\alpha) = \sin(\tilde{\alpha}) \sqrt{\frac{1 + K\|\tilde{x}\|^2}{1 + K\|\tilde{x}\|^2 \sin^2(\tilde{\alpha})}}, \qquad \cos(\alpha) = \cos(\tilde{\alpha}) \sqrt{\frac{1}{1 + K\|\tilde{x}\|^2 \sin^2(\tilde{\alpha})}}.$$

c) Gradient deformation:

$$\nabla F(x) = (1 + K\|\tilde{x}\|^2) \nabla f(\tilde{x})_1 e_1 + \sqrt{1 + K\|\tilde{x}\|^2} \nabla f(\tilde{x})_2 e_2 \quad \text{and} \quad e_i \perp e_j \ \text{ for } i \neq j.$$

And if $v \in T_x \mathcal{M}_K$ is a vector that is normal to $\nabla F(x)$, then $\tilde{v}$ is normal to $\nabla f(x)$.

The following uses the deformations described in the previous lemma to obtain the affine lower bound on the function, given a gradient at a point $\tilde{x}$. Note that Lemma 3.2.1.c) implies that we have $\langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x} \rangle = 0$ if and only if $\langle \nabla F(x), y - x \rangle = 0$. In the proof we lower bound, generally, affine functions defined on $T_x \mathcal{M}_K$ by affine functions in the Euclidean space $\mathcal{X}$. This

generality allows to obtain a result with constants that only depend on R. See Remark 3.7.2 for a discussion on these constants.

Lemma 3.2.2. [↓] *Let $F : \mathcal{M}_K \to \mathbb{R}$ be a differentiable function and let $f = F \circ h^{-1}$. Then, there are constants $\gamma_n, \gamma_p \in (0, 1]$ depending on R such that for all $x, y \in \mathcal{B}_R$ satisfying $\langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x} \rangle \neq 0$ we have:*

$$\gamma_p \leq \frac{\langle \nabla F(x), y - x \rangle}{\langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x} \rangle} \leq \frac{1}{\gamma_n}. \tag{3.2.2}$$

In particular, if F is g-convex we have the following condition, that we call tilted-convexity:

$$\begin{aligned} f(\tilde{x}) + \frac{1}{\gamma_n} \langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x} \rangle \leq f(\tilde{y}) \qquad & \text{if } \langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x} \rangle \leq 0, \\ f(\tilde{x}) + \gamma_p \langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x} \rangle \leq f(\tilde{y}) \qquad & \text{if } \langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x} \rangle \geq 0. \end{aligned} \tag{3.2.3}$$

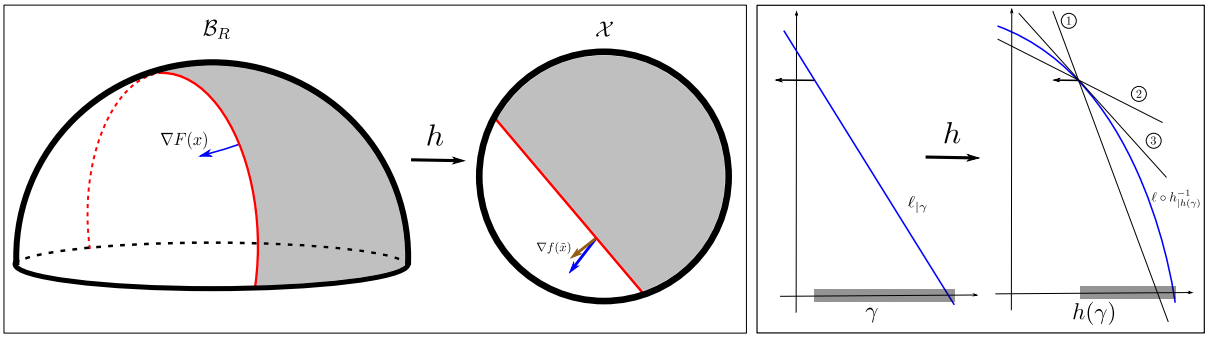


Figure 3.1: Deformations of the map h.

We provide intuition for the previous lemma through Figure 3.1. The geodesic that $\nabla F(x)$ induces on $\mathcal{B}_R$ corresponds to the geodesic that the vector on the right of $\nabla f(\tilde{x})$ would induce on $\mathcal{X}$, which is in a different direction than the one induced by $\nabla f(\tilde{x})$. The angle and gradient deformations of Lemma 3.2.1 allow to show that, for any direction, inducing a geodesic γ, the slope of the affine function induced by $\nabla f(\tilde{x})$ on $\mathcal{X}$ is within a constant factor c_1 of the one of the lower bound ℓ defined by $\nabla F(x)$ in $\mathcal{B}_R$. Our main aim is to bound $F(x^*)$ and the shaded area of each image represents where x^* can be. On the right, we exemplify the deformation on a geodesic γ passing through x. Initially, we have the affine lower bound ℓ, but the map h deforms the domain. To lower bound the function on the shaded region, we can use an affine lower bound ①. Its slope is within a constant factor of the one of the tangent line ③ by the distance deformation of Lemma 3.2.1 and the factor c_1—the latter bounds the change of the directional derivatives, represented by the horizontal vector pointing to the left. This gives the first line of (3.2.3), and the second one is analogous by using another affine function ②.

The first inequality in tilted-convexity shows the affine lower bound, which can be used to bound $f(\tilde{x}^*) = F(x^*)$. This first inequality, only applied to $\tilde{y} = \tilde{x}^*$ for a function $f : \mathbb{R}^n \to \mathbb{R}$, defines a model known in the literature as quasar-convexity or weak-quasi-convexity (Guminov and Gasnikov, 2017; Guminov, Nesterov, et al., 2019; Hinder, Sidford, and Sohoni,

2020), for which accelerated algorithms exist in the *unconstrained case*, provided smoothness is also satisfied. However, to the best of our knowledge, there is no known algorithm for solving the constrained case in an accelerated way. The condition in (3.2.3) is a relaxation of convexity that is stronger than quasar-convexity. We will make use of (3.2.3) in order to obtain acceleration in the constrained setting. This is of independent interest. Recall that we need the constraint to guarantee bounded deformation due to the geometry. We also require smoothness of f. We prove in the following lemma that f is as smooth as F up to a constant depending on R.

Lemma 3.2.3. [↓] *The function f is $O(L)$-smooth in $\mathcal{X} = h(\mathcal{B}_R)$ if $F : \mathcal{M}_K \to \mathbb{R}$ is L-smooth in $\mathcal{B}_R$.*

Inspired by the *approximate duality gap technique* (Diakonikolas and Orecchia, 2019b) we obtain accelerated continuous dynamics, for the optimization of the function f. Then we achieve acceleration by obtaining an implicit Euler discretization of the dynamics. Diakonikolas and Orecchia (2018) obtained an accelerated method for convex functions by also making use of an implicit Euler discretization. Their algorithm is fundamentally different from AGD and techniques as Linear Coupling (Allen-Zhu and Orecchia, 2017) or Nesterov's estimate sequence (Nesterov, 1983). The latter techniques use a balancing gradient step at each iteration to compensate the regret of an implicit or explicit dual algorithm, like mirror descent or Follow the Regularized Leader. Our use of a looser lower bound makes this regret greater by a constant factor and it complicates guaranteeing finding a gradient step within the constraints to compensate this greater regret. We state here the accelerated theorem and provide a sketch of the proof in Section 3.2.1.

Theorem 3.2.4. [↓] *Let $Q \subseteq \mathbb{R}^n$ be a closed convex set. Let $f : Q \to \mathbb{R}$ be an $\tilde{L}$-smooth, tilted-convex function with constants $\gamma_{\mathrm{n}}, \gamma_{\mathrm{p}} \in (0, 1]$. Assume there is a point $\tilde{x}^* \in Q$ such that $\nabla f(\tilde{x}^*) = 0$. We can obtain an ε-minimizer of f using $\widetilde{O}([\tilde{L}/(\gamma_{\mathrm{n}}^2 \gamma_{\mathrm{p}} \varepsilon)]^{1/2})$ queries to the gradient oracle of f.*

Finally, we have Riemannian acceleration as a consequence of Lemma 3.2.2, Lemma 3.2.3 and Theorem 3.2.4.

Theorem 3.2.5 (g-Convex Acceleration). [↓] *Let $F : \mathcal{M}_K \to \mathbb{R}$ be an L-smooth and g-convex function with a point $x^* \in \mathcal{B}_R$ satisfying $\nabla F(x^*) = 0$. Algorithm 5 computes a point $x_T \in \mathcal{B}_R$ satisfying $F(x_T) - F(x^*) \leq \varepsilon$ using $\widetilde{O}(\sqrt{L/\varepsilon})$ queries to the gradient oracle.*

We provide a sketch of the main optimization theorem in the section below. The full proof can be found in Section 3.5. Our use of geodesic maps was a choice we used to be able to aggregate lower bounds. Our method showcases that an effective lower bound aggregation makes possible to achieve global full acceleration. It suggests that acceleration could also be achieved for functions defined on other manifolds by using our accelerated techniques if we can effectively aggregate the lower bounds yielded by the gradient at each iteration to build a lower bound on $F(x^*)$, similarly as in (3.2.4) below. We observe that if there is a geodesic map mapping a manifold into a convex subset of the Euclidean space then the manifold must necessarily have

constant sectional curvature, cf. Beltrami's Theorem (Busemann and Phadke, 1984; Kreyszig, 1991). This means that lower bound aggregation in other manifolds would need to use a different kind of transformations. The field of comparison geometry allows to obtain properties of spaces of bounded sectional curvature by using the properties of the spaces that have constant curvature equal to the bounds of the former (Grove, Petersen, and Levy, 1997). Other Riemannian optimization algorithms have used comparison theorems that allow to obtain convergence bounds after computing the maximum possible deformations in spaces of extremal constant sectional curvature and relating them to the spaces of bounded sectional curvature (Zhang and Sra, 2016; Zhang and Sra, 2018). The generalization to algorithms to optimize functions defined on manifolds of bounded sectional curvature is a future direction of research.

3.2.1 Sketch of the proof of Theorem 3.2.4

Algorithm 5 Global Fully Accelerated g-Convex Minimization

Input: Initial point $x_0 \in \mathcal{M}_K$. Constants $\tilde{L}$, γ_{p}, γ_{n}. Geodesic map h satisfying (3.2.1) and $h(x_0) = 0$.
Smooth and g-convex function $F : \mathcal{M}_K \to \mathbb{R}$ with a minimizer $x^* \in \mathcal{B}_R$.
Bound on the distance to a minimum $R \geq d(x_0, x^*)$. Accuracy ε and number of iterations T.

$\mathcal{X} \overset{\text{def}}{=} h(\mathcal{B}_R) \subseteq M$; $f \overset{\text{def}}{=} F \circ h^{-1}$ and $\psi(\tilde{x}) \overset{\text{def}}{=} \frac{1}{2}\|\tilde{x}\|^2$ if $\tilde{x} \in \mathcal{X}$, o/w $\psi(\tilde{x}) = +\infty$.
$\tilde{z}_0 \leftarrow \nabla\psi(\tilde{x}_0)$; $A_0 \leftarrow 0$
for $i = 0$ **to** $T - 1$ **do**
  $a_{i+1} \leftarrow (i+1)\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}/2\tilde{L}$
  $A_{i+1} \leftarrow A_i + a_{i+1}$
  $\lambda \leftarrow \mathrm{BinaryLineSearch}(\tilde{x}_i, \tilde{z}_i, f, \mathcal{X}, a_{i+1}, A_i, \varepsilon, \tilde{L}, \gamma_{\mathrm{n}}, \gamma_{\mathrm{p}})$ ◇ (cf. Algorithm 6)
  $\tilde{\chi}_i \leftarrow (1-\lambda)\tilde{x}_i + \lambda\nabla\psi^*(\tilde{z}_i)$
  $\tilde{\zeta}_i \leftarrow \tilde{z}_i - (a_{i+1}/\gamma_{\mathrm{n}})\nabla f(\tilde{\chi}_i)$
  $\tilde{x}_{i+1} \leftarrow (1-\lambda)\tilde{x}_i + \lambda\nabla\psi^*(\tilde{\zeta}_i)$ ◇ $\left[\nabla\psi^*(\tilde{p}) = \arg\min_{\tilde{z}\in\mathcal{X}}\{\|\tilde{z} - \tilde{p}\|\} = \Pi_{\mathcal{X}}(\tilde{p})\right]$
  $\tilde{z}_{i+1} \leftarrow \tilde{z}_i - (a_{i+1}/\gamma_{\mathrm{n}})\nabla f(\tilde{x}_{i+1})$
end for
return x_T.

Inspired by the *approximate duality gap technique* (Diakonikolas and Orecchia, 2019b), we let α_t be an increasing function of time t, and denote $A_t = \int_{t_0}^t d\alpha_\tau = \int_{t_0}^t \dot{\alpha}_\tau d\tau$. We define a continuous method that keeps a solution $\tilde{x}_t$, along with a differentiable upper bound U_t on $f(\tilde{x}_t)$ and a lower bound L_t on $f(\tilde{x}^*)$. In our case f is differentiable so we can just take $U_t = f(\tilde{x}_t)$. The lower bound comes from

$$f(\tilde{x}^*) \geq \frac{\int_{t_0}^t f(\tilde{x}_\tau)d\alpha_\tau}{A_t} + \frac{\int_{t_0}^t \frac{1}{\gamma_{\mathrm{n}}}\langle\nabla f(\tilde{x}_\tau), \tilde{x}^* - \tilde{x}_\tau\rangle d\alpha_\tau}{A_t}, \tag{3.2.4}$$

after adding and subtracting a regularizer ψ, which is a 1-strongly convex function, and after removing the unknown $\tilde{x}^*$ by taking a minimum over $\mathcal{X}$. Note (3.2.4) comes from averaging (3.2.3) for $\tilde{y} = \tilde{x}^*$. Then, if we define the gap $G_t = U_t - L_t$ and design a method that forces $\alpha_t G_t$ to be non-increasing, we can deduce $f(x_t) - f(\tilde{x}^*) \leq G_t \leq \alpha_{t_0} G_{t_0}/\alpha_t$. By forcing $\frac{d}{dt}(\alpha_t G_t) = 0$,

we naturally obtain the following continuous dynamics, where z_t is a mirror point and ψ^* is the Fenchel dual of ψ.

$$\dot{\tilde{z}}_t = -\frac{1}{\gamma_{\mathrm{n}}}\dot{\alpha}_t \nabla f(\tilde{x}_t); \quad \dot{\tilde{x}}_t \quad = \frac{1}{\gamma_{\mathrm{n}}}\dot{\alpha}_t \frac{\nabla\psi^*(\tilde{z}_t) - \tilde{x}_t}{\alpha_t}; \quad \tilde{z}_{t_0} = \nabla\psi^*(\tilde{x}_{t_0}), \tilde{x}_{t_0} \in \mathcal{X}. \tag{3.2.5}$$

We note that except for the constant γ_{n}, these dynamics match the accelerated dynamics used in the optimization of convex functions (Krichene, Bayen, and Bartlett, 2015; Diakonikolas and Orecchia, 2018; Diakonikolas and Orecchia, 2019b). The AXGD algorithm (Diakonikolas and Orecchia, 2018), designed for the accelerated optimization of convex functions, discretizes the dynamics coming from the optimization of convex functions by using an approximate implementation of implicit Euler discretization. This has the advantage of not needing a gradient step per iteration to compensate for some positive discretization error. In our case, the extra error we incur by using a looser lower bound does not seem to be able to be compensated by a gradient step in the constrained case, as other acceleration techniques like Linear Coupling (Allen-Zhu and Orecchia, 2017) or Nesterov's estimate sequence (Nesterov, 1983) do, so obtaining an approximate implicit Euler discretization proves to be a better approach. However, our dynamics are different and in our case we must use tilted-convexity (3.2.3) instead of convexity. We are able to obtain the following discretization coming from an approximate implicit Euler discretization:

$$\begin{cases} \tilde{\chi}_i = \frac{\hat{\gamma}_i A_i}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_{\mathrm{n}}}\tilde{x}_i + \frac{a_{i+1}/\gamma_{\mathrm{n}}}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_{\mathrm{n}}}\nabla\psi^*(\tilde{z}_i); & \tilde{\zeta}_i = \tilde{z}_i - \frac{a_{i+1}}{\gamma_{\mathrm{n}}}\nabla f(\tilde{\chi}_i) \\ \tilde{x}_{i+1} = \frac{\hat{\gamma}_i A_i}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_{\mathrm{n}}}\tilde{x}_i + \frac{a_{i+1}/\gamma_{\mathrm{n}}}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_{\mathrm{n}}}\nabla\psi^*(\tilde{\zeta}_i); & \tilde{z}_{i+1} = \tilde{z}_i - \frac{a_{i+1}}{\gamma_{\mathrm{n}}}\nabla f(\tilde{x}_{i+1}) \end{cases} \tag{3.2.6}$$

where $\hat{\gamma}_i \in [\gamma_{\mathrm{p}}, 1/\gamma_{\mathrm{n}}]$ is a parameter, $\tilde{x}_0 \in \mathcal{X}$ is an arbitrary point, $\tilde{z}_0 = \nabla\psi(\tilde{x}_0)$ and now α_t is a discrete measure and $\dot{\alpha}_t$ is a weighted sum of Dirac delta functions $\dot{\alpha}_t = \sum_{i=1}^{\infty} a_i \delta(t-(t_0+i-1))$, for $a_i > 0$. However, not having convexity, in order to have per-iteration discretization error less than $\hat{\varepsilon}/A_T$, we require $\hat{\gamma}_i$ to be such that $\tilde{x}_{i+1}$ satisfies

$$f(\tilde{x}_{i+1}) - f(\tilde{x}_i) \leq \hat{\gamma}_i \langle \nabla f(\tilde{x}_{i+1}), \tilde{x}_{i+1} - \tilde{x}_i \rangle + \hat{\varepsilon}, \tag{3.2.7}$$

where $\hat{\varepsilon}$ is chosen so that the accumulated discretization error is $< \varepsilon/2$, after having performed the steps necessary to obtain an $\varepsilon/2$ minimizer. We would like to use (3.2.3) to find such a $\hat{\gamma}_i$ but we need to take into account that we only know $\tilde{x}_{i+1}$ a posteriori. Indeed, using (3.2.3) we conclude that setting $\hat{\gamma}_i$ to $1/\gamma_{\mathrm{n}}$ or γ_{p} then we either satisfy (3.2.7) or there is a point $\hat{\gamma}_i \in (\gamma_{\mathrm{p}}, 1/\gamma_{\mathrm{n}})$ for which $\langle \nabla f(\tilde{x}_{i+1}), \tilde{x}_{i+1} - \tilde{x}_i \rangle = 0$, which satisfies the inequality for $\hat{\varepsilon} = 0$. Then, using smoothness of f, the fact that $\nabla f(\tilde{x}^*) = 0$, and boundedness of $\mathcal{X}$ we can guarantee that a binary search finds a point satisfying (3.2.7) in $O(\log(\tilde{L}i/\gamma_n\hat{\varepsilon}))$ iterations. Each iteration of the binary search requires to run (3.2.6), that is, one step of the discretization. Computing the final discretization error, we obtain acceleration after choosing appropriate learning rates a_i. Algorithm 5 contains the pseudocode of this algorithm along with the reduction of the problem from minimizing F to minimizing f. We chose $\psi(\tilde{x}) \stackrel{\text{def}}{=} \frac{1}{2}\|\tilde{x}\|^2 + i_{\mathcal{X}}(\tilde{x})$ as our regularizer, where $i_{\mathcal{X}}$ is the indicator function that takes value 0 in $\mathcal{X}$ and $+\infty$ otherwise.

3.3 Reductions

The construction of reductions proves to be very useful in order to facilitate the design of algorithms in different settings. Moreover, reductions are a helpful tool to infer new lower bounds without extra ad hoc analyses. We present two reductions. Namely, we show how we can minimize a g-convex smooth problem with an algorithm the minimizes strongly g-convex functions, and vice versa. We will see in Corollary 3.3.2 and Example 3.3.6 that these reductions preserve full acceleration. These reductions are generalizations of some reductions designed to work in the Euclidean space (Allen-Zhu and Hazan, 2016; Allen-Zhu and Orecchia, 2017). The reduction to strongly g-convex functions takes into account the effect of the deformation of the space on the strong convexity of the function $F_y(x) = d(x,y)^2/2$, for $x, y \in \mathcal{M}$. It does not entail an extra $\log(1/\varepsilon)$ factor. The reduction to g-convexity requires the rate of the algorithm that applies to g-convex functions to be proportional to the squared distance between the initial point and the optimum $d(x_0, x^*)^2$ or to a bound R^2. The proofs of the statements in this section can be found in Section 3.3. We will use $\text{Time}_{\text{ns}}(\cdot)$ and $\text{Time}(\cdot)$ to denote the time algorithms $\mathcal{A}_{\text{ns}}$ and $\mathcal{A}$ below require, respectively, to perform the tasks we define below.

Theorem 3.3.1. *Let $\mathcal{M}$ be a Riemannian manifold, let $F : \mathcal{M} \to \mathbb{R}$ be an L-smooth and μ-strongly g-convex function, with a minimizer x^*. Suppose we have an algorithm $\mathcal{A}_{\text{ns}}$ to minimize F, such that for any starting point x_0 such that $d(x_0, x^*) \leq R$, it produces a point $\hat{x}_T$ in time $T = \text{Time}_{\text{ns}}(L, \mu, R)$ satisfying $F(\hat{x}_T) - F(x^*) \leq \mu R^2/4$. Then we can compute an ε-minimizer of F in time $O(\text{Time}_{\text{ns}}(L, \mu, R) \log(\mu/\varepsilon))$.*

Proof Let $\mathcal{A}_{\text{ns}}$ be the algorithm in the statement of the theorem. By strong g-convexity of F and the assumptions on $\mathcal{A}_{\text{ns}}$ we have that $\hat{x}_T$, the point computed by $\mathcal{A}_{\text{ns}}$, satisfies

$$\frac{\mu}{2} d(\hat{x}_T, x^*)^2 \leq F(\hat{x}_T) - F(x^*) \leq \frac{\mu}{2} \frac{R^2}{2},$$

after $T = \text{Time}_{\text{ns}}(L, \mu, R)$ queries to the gradient oracle. This implies $d(\hat{x}_T, x^*)^2 \leq R^2/2$. We perform this process $r \stackrel{\text{def}}{=} \lceil \log(\mu R^2/\varepsilon) - 1 \rceil$ times. We use the previous output as input for the next round. The distance bound to x^* can be updated to a lower value. We denote R_i the distance bound from the input to x^* at stage i and we set its value to $R_i = R_{i-1}/\sqrt{2}$, for $R_1 = R$. Thus, after r stages we obtain a point $\hat{x}_T^r$ that satisfies

$$F(\hat{x}_T^r) - F(x^*) \leq \frac{\mu \cdot R_r^2}{4} = \frac{\mu \cdot R_1^2}{4 \cdot 2^{r-1}} \leq \varepsilon.$$

And the total running time is $\text{Time}_{\text{ns}}(L, \mu, R) \cdot r = O(\text{Time}_{\text{ns}}(L, \mu, R) \log(\mu/\varepsilon))$. ■

Theorem 3.3.1 implies that if we forget about the strong g-convexity of a function and we treat it as if it is just g-convex then we can run in stages an algorithm designed for optimizing g-convex functions and still achieve acceleration. The fact that the function is strongly g-convex is only used between stages. We exemplify the power of the reduction by applying it to Algorithm 5.

Corollary 3.3.2. *We can compute an ε-minimizer of an L-smooth and μ-strongly g-convex function $F : \mathcal{M}_K \to \mathbb{R}$ in $O^*(\sqrt{L/\mu}\log(\mu/\varepsilon))$ queries to the gradient oracle.*

We note that in the strongly convex case, by decreasing the function value by a factor we can guarantee we decrease the distance to x^* by another factor, so we can periodically recenter the geodesic map to reduce the constants produced by the deformations of the geometry, as we show in the following proof of the corollary.

Proof We can assume without loss of generality $K \in \{-1, 1\}$ as we did in Section 3.2. Let R be an upper bound on the distance between the initial point x_0 and an optimizer x^*, i.e. $d(x_0, x^*) \leq R$. Note that $\|\tilde{x}_0 - \tilde{x}^*\|/R$ is bounded by a constant depending on R by Lemma 3.2.1.a). Note that γ_{n} and γ_{p} are constants depending on R by Lemma 3.2.2. As any g-strongly convex function is g-convex, by using Theorem 3.5.3 and Lemma 3.5.4 with $\varepsilon = \mu\frac{R^2}{4}$ we obtain that Algorithm 5 obtains a $\mu\frac{R^2}{4}$-minimizer in at most

$$T = O\left(\frac{\|\tilde{x}_0 - \tilde{x}^*\|}{R}\sqrt{\frac{4L}{\mu\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}}}\log\left(\frac{\|\tilde{x}_0 - \tilde{x}^*\|}{R}\sqrt{\frac{4L}{\mu\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}}}\right)\right) = O\left(\sqrt{L/\mu}\log(L/\mu)\right)$$

queries to the gradient oracle. Subsequent stages, i.e. calls to Algorithm 5, use the point computed at the previous stage as its input. The distance bound to x^* is updated, following the proof of Theorem 3.3.1. Because the constant depending on R in the running time of the subroutine decreases when R has a lower value, subsequent stages need a time which is $O(\sqrt{L/\mu}\log(L/\mu))$ as well. So we satisfy the assumption of Theorem 3.3.1 for $\mathrm{Time}_{\mathrm{ns}}(L, \mu, R) = O(\sqrt{L/\mu}\log(L/\mu))$. We conclude that given $\varepsilon > 0$ and running Algorithm 5 in stages, we obtain an ε-minimizer of F in

$$O(\sqrt{L/\mu}\log(L/\mu)\log(\mu/\varepsilon)) = O^*(\sqrt{L/\mu}\log(\mu/\varepsilon))$$

queries to the gradient oracle.

Note that each time we call Algorithm 5 we recenter the geodesic map. In order to perform the method with these recentering steps, we need the function F to be defined over at least $\mathrm{Exp}_{x_0}(\bar{B}(0, R\cdot(1+2^{-1/2})))$, since subsequent centers are only guaranteed to be $\leq R/\sqrt{2}$ close to x^*, and they could get slightly farther than R from x_0. But they are no farther than $R + R/\sqrt{2}$ since $d(x_0, \hat{x}_T^i) \leq d(x_0, x^*) + d(x^*, \hat{x}_T^i) \leq R + R/\sqrt{2}$, where $\hat{x}_T^i$ is the center at stage i, and where $i > 1$. ■

Finally, we show the reverse reduction.

Theorem 3.3.3. *Let $\mathcal{M}$ be a Riemannian manifold of bounded sectional curvature, let $F : \mathcal{M} \to \mathbb{R}$ be an L-smooth and g-convex function, and assume there is a point $x^* \in \mathcal{M}$ such that $\nabla F(x^*) = 0$. Let x_0 be a starting point such that $d(x_0, x^*) \leq R$ and let Δ satisfy $F(x_0) - F(x^*) \leq \Delta$. Assume Exp_{x_0} is a diffeomorphism when restricted to $\bar{B}(0, R)$ and that we have an algorithm $\mathcal{A}$ that given an L-smooth and μ-strongly g-convex function $\hat{F} : \mathcal{M} \to \mathbb{R}$, with minimizer in $\mathrm{Exp}_{x_0}(\bar{B}(0, R))$, and any initial point $\hat{x}_0 \in \mathcal{M}$, produces a point $\hat{x} \in \mathrm{Exp}_{x_0}(\bar{B}(0, R))$*

in time $\hat{T} = \text{Time}(L, \mu, \mathcal{M}, R)$ *satisfying* $\hat{F}(\hat{x}) - \min_{x\in\mathcal{M}} \hat{F}(x) \leq (\hat{F}(\hat{x}_0) - \min_{x\in\mathcal{M}} \hat{F}(x))/4$. *Let* $T = \lceil \log_2(\Delta/\varepsilon) \rceil + 1$. *Then, we can compute an* ε*-minimizer in time* $\sum_{t=0}^{T-1} \text{Time}(L + 2^{-t}\Delta\mathcal{K}_R^-/R^2, 2^{-t}\Delta\mathcal{K}_R^+/R^2, \mathcal{M}, R)$, *where* $\mathcal{K}_R^+$ *and* $\mathcal{K}_R^-$ *are constants that depend on* R *and the bounds on the sectional curvature of* $\mathcal{M}$.

We need two results before proving the theorem. We first state the following fact. Recall that $\text{Exp}_{x_0}(\bar{B}(0, R)) \subset \mathcal{M}$. We define the following quantities

$$\mathcal{K}_R^+ \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } K_{\max} \leq 0 \\ \sqrt{K_{\max}} R \cot(\sqrt{K_{\max}} R) & \text{if } K_{\max} > 0 \end{cases}$$

$$\mathcal{K}_R^- \stackrel{\text{def}}{=} \begin{cases} \sqrt{-K_{\min}} R \coth(\sqrt{-K_{\min}} R) & \text{if } K_{\min} < 0 \\ 1 & \text{if } K_{\min} \geq 0 \end{cases}$$

Here $K_{\max}$ and $K_{\min}$ are the upper and lower bounds on the sectional curvature of the manifold $\mathcal{M}$.

Fact 3.3.4. *Let* $\mathcal{M}$ *be a manifold with sectional curvature bounded below and above by* $K_{\min}$ *and* $K_{\max}$, *respectively. For a point* $x_0 \in \mathcal{M}$, *assume* Exp_{x_0} *is a diffeomorphism when restricted to* $\bar{B}(0, R)$. *The function* $f : \mathcal{M} \to \mathbb{R}$ *defined as* $f(x) = \frac{1}{2}d(x, x_0)^2$ *is* $\mathcal{K}_R^+$*-g-strongly convex and* $\mathcal{K}_R^-$*-smooth in* $\text{Exp}_{x_0}(\bar{B}(0, R))$.

The result regarding strong convexity can be found, for instance, in (Alimisis et al., 2020) and it is a direct consequence of the following inequality, which can also be found in (Alimisis et al., 2020):

$$d(y, x_0)^2 \geq d(x, x_0)^2 - 2\langle \text{Exp}_x^{-1}(x_0), y - x\rangle + \mathcal{K}_R^+ d(x, y)^2,$$

along with the fact that $\text{grad}\, f(x) = -\text{Exp}_x^{-1}(x_0)$. The result regarding smoothness is, similarly, obtained from the following inequality:

$$d(y, x_0)^2 \leq d(x, x_0)^2 - 2\langle \text{Exp}_x^{-1}(x_0), y - x\rangle + \mathcal{K}_R^- d(x, y)^2,$$

which can be found in (Zhang and Sra, 2016) (Lemma 6). These inequalities are tight in spaces of constant sectional curvature. Alternatively, one can derive these inequalities from upper and lower bounds on the Hessian of $f(x) = \frac{1}{2}d(x, x_0)$, as it was done in (Lezcano-Casado, 2020) (Theorem 3.15).

We prove now that the regularization makes the minimum to be closer to x_0, so the assumption of the theorem on $\hat{F}$ holds for the functions we use. Define x_{i+1} as the minimizer of $F^{(\mu_i)}$.

Lemma 3.3.5. *We have* $d(x_{i+1}, x_0) \leq d(x^*, x_0)$.

Proof By the fact that x_{i+1} is the minimizer of $F^{(\mu_i)}$ we have $F^{(\mu_i)}(x_{i+1}) - F^{(\mu_i)}(x^*) \leq 0$. Note that by g-strong convexity, equality only holds if $x_{i+1} = x^*$ which only happens if $x_0 = x_{i+1} = x^*$.

By using the definition of $F^{(\mu_i)}(x) = F(x) + \frac{\mu_i}{2}d(x, x_0)^2$ we have:

$$\begin{aligned} & F(x_{i+1}) + \frac{\mu_i}{2}d(x_{i+1}, x_0)^2 - F(x^*) - \frac{\mu_i}{2}d(x^*, x_0)^2 \leq 0 \\ \Rightarrow \quad & d(x_{i+1}, x_0) \leq d(x^*, x_0), \end{aligned}$$

where in the last step we used the fact $F(x_{i+1}) - F(x^*) \geq 0$ that holds because x^* is the minimizer of F. ■

We note that previous techniques proved and used the fact that $d(x_{i+1}, x^*) \leq d(x_0, x^*)$ instead (Allen-Zhu and Hazan, 2016). But crucially, we need our former lemma in order to prove the bound for our non-Euclidean case. Our variant can be applied to (Allen-Zhu and Hazan, 2016) to decrease the constants of their Euclidean reduction. Now we are ready to prove the theorem.

Proof of Theorem 3.3.3. The algorithm is the following. We successively regularize the function with strongly g-convex regularizers in this way $F^{(\mu_i)}(x) \stackrel{\text{def}}{=} F(x) + \frac{\mu_i}{2}d(x, x_0)^2$ for $i \geq 0$. For each $i \geq 0$, we use the algorithm $\mathcal{A}$ on the function $F^{(\mu_i)}$ for the time in the statement of the theorem and obtain a point $\hat{x}_{i+1}$, starting from point $\hat{x}_i$, where $\hat{x}_0 = x_0$. The regularizers are decreased exponentially $\mu_{i+1} = \mu_i/2$ from $\mu_0 = \Delta/R^2$, until we reach roughly $\mu_T = \varepsilon/R^2$, see below for the precise value. Fact 3.3.4 says that indeed $\frac{\mu_i}{2}d(x, x_0)^2$ is a strongly g-convex regularizer.

Let's see how this algorithm works. We start with $\hat{x}_0 = x_0$ and compute $\hat{x}_{i+1}$ using algorithm $\mathcal{A}$ with starting point $\hat{x}_i$ and function $F^{(\mu_i)}$ for time $\text{Time}(L^{(i)}, \mu^{(i)}, \mathcal{M}, R)$, where $L^{(i)}$ and $\mu^{(i)}$ are the smoothness and strong g-convexity parameters of $F^{(\mu_i)}$. We denote by x_{i+1} the minimizer of $F^{(\mu_i)}$. We pick $\mu_i = \mu_{i-1}/2$ and we will choose later the value of μ_0 and the total number of stages. By the assumption of the theorem on $\mathcal{A}$, we have that

$$F^{(\mu_i)}(\hat{x}_{i+1}) - \min_{x \in \mathcal{M}} F^{(\mu_i)}(x) = F^{(\mu_i)}(\hat{x}_{i+1}) - F^{(\mu_i)}(x_{i+1}) \leq \frac{F^{(\mu_i)}(\hat{x}_i) - F^{(\mu_i)}(x_{i+1})}{4}. \tag{3.3.1}$$

Define $D_i \stackrel{\text{def}}{=} F^{(\mu_i)}(\hat{x}_i) - F^{(\mu_i)}(x_{i+1})$ to be the initial objective distance to the minimum on function $F^{(\mu_i)}$ before we call $\mathcal{A}$ for the $(i+1)$-th time. At the beginning, we have the upper bound $D_0 = F^{(\mu_0)}(\hat{x}_0) - \min_x F^{(\mu_0)}(x) \leq F(x_0) - F(x^*)$. For each stage $i \geq 1$, we compute that

$$\begin{aligned} D_i &= F^{(\mu_i)}(\hat{x}_i) - F^{(\mu_i)}(x_{i+1}) \\ &\overset{①}{=} F^{(\mu_{i-1})}(\hat{x}_i) - \frac{\mu_{i-1} - \mu_i}{2}d(x_0, \hat{x}_i)^2 - F^{(\mu_{i-1})}(x_{i+1}) + \frac{\mu_{i-1} - \mu_i}{2}d(x_0, x_{i+1})^2 \\ &\overset{②}{\leq} F^{(\mu_{i-1})}(\hat{x}_i) - F^{(\mu_{i-1})}(x_i) + \frac{\mu_{i-1} - \mu_i}{2}d(x_0, x_{i+1})^2 \\ &\overset{③}{\leq} \frac{D_{i-1}}{4} + \frac{\mu_i}{2}d(x_0, x_{i+1})^2 \overset{④}{\leq} \frac{D_{i-1}}{4} + \frac{\mu_i}{2}d(x_0, x^*)^2. \end{aligned}$$

Above, ① follows from the definition of $F^{(\mu_i)}(\cdot)$ and $F^{(\mu_{i-1})}(\cdot)$; ② follows from the fact that x_i is the minimizer of $F^{(\mu_{i-1})}(\cdot)$. And we dropped the negative term $-(\mu_{i-1} - \mu_i)d(x_0, \hat{x}_i)/2$. ③

follows from the definition of D_{i-1}, the assumption on $\mathcal{A}$, and the choice $\mu_i = \mu_{i-1}/2$ for $i \geq 1$; and ④ follows from Lemma 3.3.5. Now applying the above inequality recursively, we have

$$D_T \leq \frac{D_0}{4^T} + d(x_0, x^*)^2 \cdot (\frac{\mu_T}{2} + \frac{\mu_{T-1}}{8} + \cdots) \leq \frac{F(x_0) - F(x^*)}{4^T} + \mu_T \cdot d(x_0, x^*)^2. \tag{3.3.2}$$

We have used the choice $\mu_i = \mu_{i-1}/2$ for the second inequality. Lastly, we can prove that $\hat{x}_T$, the last point computed, satisfies

$$\begin{aligned} F(\hat{x}_T) - F(x^*) &\overset{①}{\leq} F^{(\mu_T)}(\hat{x}_T) - F^{(\mu_T)}(x^*) + \frac{\mu_T}{2} d(x_0, x^*)^2 \\ &\overset{②}{\leq} F^{(\mu_T)}(\hat{x}_T) - F^{(\mu_T)}(x_{T+1}) + \frac{\mu_T}{2} d(x_0, x^*)^2 \\ &\overset{③}{=} D_T + \frac{\mu_T}{2} d(x_0, x^*)^2 \overset{④}{\leq} \frac{F(x_0) - F(x^*)}{4^T} + \frac{3\mu_T}{2} d(x_0, x^*)^2. \end{aligned}$$

We use the definition of $F^{(\mu_T)}$ in ① and drop $-\frac{\mu_T}{2} d(x_0, \hat{x}_T)^2$. In ② we use the fact that x_{T+1} is the minimizer of $F^{(\mu_T)}$. The definition of D_T is used in ③. We use inequality (3.3.2) for step ④. Recall the assumption of the theorem $F(x_0) - F(x^*) \leq \Delta$. Finally, by choosing $T = \lceil \log_2(\Delta/\varepsilon) \rceil + 1$ and $\mu_0 = \Delta/R^2$ we obtain that the point $\hat{x}_T$ satisfies

$$F(\hat{x}_T) - F(x^*) \leq \frac{F(x_0) - F(x^*)}{4\Delta/\varepsilon} + \frac{3\mu_0}{8\Delta/\varepsilon} d(x_0, x^*)^2 \leq \frac{\varepsilon}{4} + \frac{3\varepsilon}{8} < \varepsilon,$$

and can be computed in time $\sum_{t=0}^{T-1} \mathrm{Time}(L + 2^{-t}\mu_0 \mathcal{K}_R^-, 2^{-t}\mu_0 \mathcal{K}_R^+, \mathcal{M}, R)$, since by Fact 3.3.4 the function $F^{(\mu_t)}$ is $L + 2^{-t}\mu_0 \mathcal{K}_R^-$ smooth and $2^{-t}\mu_0 \mathcal{K}_R^+$ g-strongly convex. ■

Example 3.3.6. Applying Theorem 3.3.3 to the algorithm in Corollary 3.3.2 we can optimize L-smooth and g-convex functions defined on $\mathcal{M}_K$ with a gradient oracle complexity of $\widetilde{O}(L/\sqrt{\varepsilon})$.

We use the algorithm in Corollary 3.3.2 as the algorithm $\mathcal{A}$ of the reduction of Theorem 3.3.3. Given a manifold under consideration $\mathcal{M}_K$, the assumption on $\mathcal{A}$ is satisfied for $\mathrm{Time}(L, \mu, \mathcal{M}_K, R) = O(\sqrt{L/\mu} \log(L/\mu))$. Indeed, if Δ is a bound on the gap $\hat{F}(x_0) - \hat{F}(x^*) = \hat{F}(x_0) - \min_{x \in \mathcal{M}_K} \hat{F}(x) = \hat{F}(x_0) - \min_{x \in \mathrm{Exp}_{x_0}(\bar{B}(0,R))} \hat{F}(x)$ for some μ-strongly g-convex $\hat{F}$, then we know that $d(x_0, x^*)^2 \leq \frac{2\Delta}{\mu}$ by μ-strong g-convexity. By calling the algorithm in Corollary 3.3.2 with $\varepsilon = \frac{\Delta}{4}$ we require a time that is

$$\begin{aligned} O(\sqrt{L/\mu} \log(L/\mu) \log(\mu \cdot d(x_0, x^*)^2/(\Delta/4))) &= O(\sqrt{L/\mu} \log(L/\mu) \log(\mu \cdot (2\Delta/\mu)/(\Delta/4))) \\ &= O(\sqrt{L/\mu} \log(L/\mu)). \end{aligned}$$

Let $T = \lceil \log_2(\Delta/\varepsilon) \rceil + 1$. The reduction of Theorem 3.3.3 gives an algorithm with rates

$$\begin{aligned}
&\sum_{t=0}^{T-1} \mathrm{Time}(L + 2^{-t}\mu_0\mathcal{K}_R^-, 2^{-t}\mu_0\mathcal{K}_R^+, \mathcal{M}_K, R) \\
&\quad \overset{①}{=} O\left(\sum_{t=0}^{T-1} \sqrt{\frac{\mathcal{K}_R^-}{\mathcal{K}_R^+} + \frac{L}{2^{-t}\mathcal{K}_R^+\Delta/R^2}} \cdot \log\left(\frac{\mathcal{K}_R^-}{\mathcal{K}_R^+} + \frac{L}{2^{-t}\mathcal{K}_R^+\Delta/R^2}\right)\right) \\
&\quad \overset{②}{=} O\left(\left(\sqrt{\frac{\mathcal{K}_R^-}{\mathcal{K}_R^+}}\log(\Delta/\varepsilon) + \sum_{t=0}^{T-1}\sqrt{\frac{L}{2^{-t}\mathcal{K}_R^+\Delta/R^2}}\right)\log\left(\frac{\mathcal{K}_R^-}{\mathcal{K}_R^+} + \frac{L}{\mathcal{K}_R^+\varepsilon}\right)\right) \\
&\quad \overset{③}{=} O\left(\left(\sqrt{\frac{\mathcal{K}_R^-}{\mathcal{K}_R^+}}\log(\Delta/\varepsilon) + \sqrt{\frac{L}{\mathcal{K}_R^+\varepsilon}}\right)\log\left(\frac{\mathcal{K}_R^-}{\mathcal{K}_R^+} + \frac{L}{\mathcal{K}_R^+\varepsilon}\right)\right) \\
&\quad \overset{④}{=} \widetilde{O}(\sqrt{L/\varepsilon})
\end{aligned}$$

In ① we write down the definition and use the value $\mu_0 = \Delta/R^2$. In ② we have used Minkowski's inequality $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$. We added up the first group of summands. For the log factor, we upper bounded $L/(2^{-t}\mathcal{K}_R^+\Delta/R^2) = O(L/\mathcal{K}_R^+\varepsilon)$, for $t < T$. In ③ we used the fact that $\sqrt{1/\varepsilon} + \sqrt{1/2\varepsilon} + \cdots = O(\sqrt{1/\varepsilon})$, along with the fact $\varepsilon/2R^2 \leq 2^{-(T-1)}\mu_0 \leq \varepsilon/R^2$. Note that by L-smoothness and the diameter being $2R$, we have $\Delta \leq 2LR^2$ so $\sqrt{\mathcal{K}_R^-/\mathcal{K}_R^+}\log(\Delta/\varepsilon) = \widetilde{O}(1)$. We applied this in ④.

Note that this reduction cannot be applied to the locally accelerated algorithm in (Zhang and Sra, 2018), that we discussed in the related work section. The reduction runs in stages by regularizing each time with a strongly g-convex regularizer whose parameter decreases exponentially until we use a regularizer with $O(\varepsilon)$ maximum function value. The local assumption required by the algorithm in (Zhang and Sra, 2018) on the closeness to the minimum cannot be guaranteed. In (Ahn and Sra, 2020), the authors give a global algorithm whose rates are strictly better than RGD. The reduction could be applied to a version of this algorithm, that works with a ball constraint, and one would obtain a method for smooth and g-convex functions defined on manifolds of bounded sectional curvature and whose rates are strictly better than RGD.

3.4 Discussion

We proposed an algorithm with the same rates as AGD, for the optimization of smooth and strongly g-convex functions, up to constants and log factors, while previous approaches essentially only reached this for a ball around the minimizer of radius $O((\mu/L)^{3/4})$. Our algorithm also applies to g-convex functions while previous accelerated algorithms did not apply. We focused on hyperbolic and spherical spaces, that have constant sectional curvature. The study of geometric properties for this is often employed to conclude that a space of bounded sectional curvature satisfies a property that is in between the ones for the cases of constant extremal sectional curvature. Several previous algorithms have been developed for the general case by utilizing this philosophy, for instance (Ahn and Sra, 2020; Ferreira, Louzeiro, and da Fonseca Prudente, 2019; Wang, Li, and Yao, 2015; Zhang and Sra, 2016; Zhang and Sra, 2018). Using the techniques and

insights developed in this work is a promising direction of research to design an algorithm with the same rates as AGD for manifolds of bounded sectional curvature.

The key technique of our algorithm is the effective lower bound aggregation. Indeed, lower bound aggregation is the main hurdle to obtain accelerated first-order methods defined on Riemannian manifolds. Whereas the process of obtaining decreasing upper bounds on the function works similarly as in the Euclidean space—the same approach of locally minimizing the upper bound given by the smoothness assumption is used—obtaining adequate lower bounds proves to be a difficult task. We usually want a simple lower bound such that it, or a regularized version of it, can be easily optimized globally. We also want that the lower bound combines the knowledge that the g-convexity or strong g-convexity provides for all the queried points, commonly an average. These Riemannian convexity assumptions provide simple lower bounds, namely linear or quadratic, but each with respect to each of the tangent spaces of the queried points only. The deformations of the space complicate the aggregation of the lower bounds. We deal with this problem by finding appropriate lower bounds via the use of a geodesic map and takes into account the deformations incurred to derive a fully accelerated algorithm. We also used other tools for designing the accelerated algorithm. We worked with a relaxation of convexity that allowed to perform a binary search to reduce the discretization error. We had to use an implicit discretization of some accelerated continuous dynamics, since at least the vanilla application of usual approaches like Linear Coupling (Allen-Zhu and Orecchia, 2017) or Nesterov's estimate sequence (Nesterov, 1983), that can be seen as a forward Euler discretization of the accelerated dynamics combined with a balancing gradient step (Diakonikolas and Orecchia, 2019b), did not work in our constrained case. We interpret that the difficulty arises from trying to keep the gradient step inside the constraints while being able to compensate for a lower bound that is looser by a constant factor.

3.5 Acceleration: Proofs of Theorem 3.2.4 and Theorem 3.2.5

In this section we will make use of the *approximate duality gap technique* (Diakonikolas and Orecchia, 2019b), which is we presented in Section 2.6 applied to smooth convex optimization and that is a technique that provides a structure to design and prove first-order methods and their guarantees for the optimization of convex problems. We take inspiration from these ideas to apply them to the non-convex problem we have at hand Theorem 3.2.4, as it was sketched in Section 3.2.1.

For simplicity, we will use $\psi(\tilde{x}) = \frac{1}{2}\|\tilde{x}\|^2 + i_Q(\tilde{x})$ in Algorithm 5 as regularizer. Here $i_Q(x) = 0$ if $x \in Q$ and $i_Q(x) = +\infty$ otherwise. The gradient of the Fenchel dual of $\psi(\cdot)$ is $\nabla\psi^*(\tilde{z}) = \arg\min_{\tilde{z}' \in Q}\{\|\tilde{z}' - \tilde{z}\|\}$, that is, the Euclidean projection $\Pi_Q(\tilde{z})$ of the point $\tilde{z}$ onto Q. Note that when we apply Theorem 3.2.4 to Theorem 3.2.5 our constraint Q will be $\mathcal{X}$, that is, a ball centered at $\mathbb{0}_n$ of radius $\tilde{R}$, so the projection of a point $\tilde{z}$ outside of $\mathcal{X}$ will be the vector normalization $\tilde{R}\tilde{z}/\|\tilde{z}\|$. Any continuously differentiable strongly convex ψ would work, provided

that $\nabla\psi^*(z)$ is easily computable, preferably in closed form. Note that by the Fenchel-Moreau theorem we have for any such map that $\psi^{**} = \psi$.

We recall we assume that f satisfies tilted-convexity (3.2.3):

$$f(\tilde{x}) + \frac{1}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x}\rangle \leq f(\tilde{y}) \qquad \text{if } \langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x}\rangle \leq 0,$$
$$f(\tilde{x}) + \gamma_{\mathrm{p}}\langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x}\rangle \leq f(\tilde{y}) \qquad \text{if } \langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x}\rangle \geq 0.$$

We define a continuous method, similarly to how we proceeded in Section 2.6, and we discretize it with an approximate implementation of the implicit Euler method. Let $\tilde{x}_t$ be the solution obtained by the algorithm at time t. The duality gap is also defined $G_t \stackrel{\text{def}}{=} U_t - L_t$ as the difference between a differentiable upper bound U_t on the function at the current point and a lower bound on $f(\tilde{x}^*)$. Since in our case f is differentiable we use $U_t \stackrel{\text{def}}{=} f(\tilde{x}_t)$.

Note that for a global minimum $\tilde{x}^*$ of f and any other point $\tilde{x} \in Q$, we have $\langle \nabla f(\tilde{x}), \tilde{x}^* - \tilde{x}\rangle \leq 0$. Otherwise, we would obtain a contradiction since by tilted-convexity (3.2.3) we would have

$$f(\tilde{x}) < f(\tilde{x}) + \gamma_{\mathrm{p}}\langle \nabla f(\tilde{x}), \tilde{x}^* - \tilde{x}\rangle \leq f(\tilde{x}^*).$$

Therefore, in order to define an appropriate lower bound, we will make use of the inequality $f(\tilde{x}^*) \geq f(\tilde{x}) + \frac{1}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{x}), \tilde{x}^* - \tilde{x}\rangle$, for any $\tilde{x} \in Q$, which holds true by tilted-convexity (3.2.3), for $\tilde{y} = \tilde{x}^*$. Combining this inequality for all the points visited by the continuous method we have

$$f(\tilde{x}^*) \geq \frac{\int_{t_0}^t f(\tilde{x}_\tau) d\alpha_\tau}{A_t} + \frac{\int_{t_0}^t \frac{1}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{x}_\tau), \tilde{x}^* - \tilde{x}_\tau\rangle d\alpha_\tau}{A_t}.$$

Adding and subtracting the strongly convex regularizer $D_\psi(\tilde{x}^*, \tilde{x}_{t_0})$, taking a minimum over $\mathcal{X}$ to remove the dependence on $\mathcal{X}$ and rescaling to avoid problems at t_0, similarly to the approach in Section 2.6 we obtain the following lower bound:

$$\begin{aligned} f(\tilde{x}^*) \geq L_t \stackrel{\text{def}}{=} & \frac{\int_{t_0}^t f(\tilde{x}_\tau) d\alpha_\tau}{\alpha_t} + \frac{\min_{\tilde{u}\in Q}\left\{\int_{t_0}^t \langle \frac{1}{\gamma_{\mathrm{n}}}\nabla f(\tilde{x}_\tau), \tilde{u} - \tilde{x}_\tau\rangle d\alpha_\tau + D_\psi(\tilde{u}, \tilde{x}_{t_0})\right\}}{\alpha_t} \\ & + \frac{(\alpha_t - A_t) f(\tilde{x}^*) - D_\psi(\tilde{x}^*, \tilde{x}_{t_0})}{\alpha_t}. \end{aligned} \tag{3.5.1}$$

Let $\tilde{z}_t = \nabla\psi(\tilde{x}_{t_0}) - \int_{t_0}^t \frac{1}{\gamma_{\mathrm{n}}}\nabla f(\tilde{x}_\tau) d\alpha_\tau$. Then, by Fact 2.1.6, we can compute the optimum $\tilde{u} \in Q$ above as

$$\nabla\psi^*(\tilde{z}_t) = \arg\min_{\tilde{u}\in Q}\left\{\int_{t_0}^t \langle \frac{1}{\gamma_{\mathrm{n}}}\nabla f(\tilde{x}_\tau), \tilde{u} - \tilde{x}_\tau\rangle d\alpha_\tau + D_\psi(\tilde{u}, \tilde{x}_{t_0})\right\}. \tag{3.5.2}$$

Recalling $U_t = f(\tilde{x}_t)$ and using (3.5.1) and (3.5.2) we obtain:

$$\begin{aligned} \frac{d}{dt}(\alpha_t G_t) &= \frac{d}{dt}(\alpha_t f(\tilde{x}_t)) - \dot{\alpha}_t f(\tilde{x}_t) - \dot{\alpha}_t \frac{1}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{x}_t), \nabla\psi^*(\tilde{z}_t) - \tilde{x}_t\rangle \\ &= \frac{1}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{x}_t), \gamma_{\mathrm{n}}\alpha_t\dot{\tilde{x}}_t - \dot{\alpha}_t(\nabla\psi^*(\tilde{z}_t) - \tilde{x}_t)\rangle. \end{aligned}$$

Thus, to satisfy the invariant $\frac{d}{dt}(\alpha_t G_t) = 0$, it is enough to set $\gamma_{\mathrm{n}}\alpha_t\dot{\tilde{x}}_t = \dot{\alpha}_t(\nabla\psi^*(\tilde{z}_t) - \tilde{x}_t)$,

yielding the following continuous accelerated dynamics

$$\dot{\tilde{z}}_t = -\frac{1}{\gamma_{\mathrm{n}}}\dot{\alpha}_t \nabla f(\tilde{x}_t); \quad \dot{\tilde{x}}_t \ = \frac{1}{\gamma_{\mathrm{n}}}\dot{\alpha}_t \frac{\nabla\psi^*(\tilde{z}_t) - \tilde{x}_t}{\alpha_t}; \tag{3.5.3}$$

for arbitrary $\tilde{x}_{t_0} \in Q$ and for $\tilde{z}_{t_0} = \nabla\psi(\tilde{x}_{t_0})$. Now we proceed to discretize the dynamics, so from now on we will use a discrete measure α_t, as we described above. We set t_0 to 1. Let $E_{i+1} \stackrel{\text{def}}{=} A_{i+1}G_{i+1} - A_iG_i$ be the discretization error. Then we have

$$G_t = \frac{A_1}{A_t}G_1 + \frac{\sum_{i=1}^{t-1} E_{i+1}}{A_t}.$$

Lemma 3.5.1. *If we have*

$$f(\tilde{x}_{i+1}) - f(\tilde{x}_i) \leq \hat{\gamma}_i\langle\nabla f(\tilde{x}_{i+1}), \tilde{x}_{i+1} - \tilde{x}_i\rangle + \hat{\varepsilon}_i, \tag{3.5.4}$$

for some $\hat{\gamma}_i, \hat{\varepsilon}_i \geq 0$, then the discretization error satisfies

$$E_{i+1} \leq \langle\nabla f(\tilde{x}_{i+1}), (A_i\hat{\gamma}_i + \frac{a_{i+1}}{\gamma_{\mathrm{n}}})\tilde{x}_{i+1} - \hat{\gamma}_iA_i\tilde{x}_i - \frac{a_{i+1}}{\gamma_{\mathrm{n}}}\nabla\psi^*(\tilde{z}_{i+1}))\rangle - D_{\psi^*}(\tilde{z}_i, \tilde{z}_{i+1}) + A_i\hat{\varepsilon}_i.$$

Proof In a similar way to (Diakonikolas and Orecchia, 2018), we could compute the discretization error as the difference between the gap and the gap computed allowing continuous integration rules in the integrals that it contains. However, we will directly bound E_{i+1} as $A_{i+1}G_{i+1} - A_iG_i$ instead. Recall that in discrete time we have $\alpha_i = A_i$ so the definition of the lower bound in discrete time becomes the following, by combining (3.5.1) and (3.5.2):

$$L_i = \sum_{j=1}^{i} a_j f(\tilde{x}_j) + \sum_{j=1}^{i}\langle\frac{a_j}{\gamma_{\mathrm{n}}}\nabla f(\tilde{x}_j), \nabla\psi^*(\tilde{z}_i) - \tilde{x}_j\rangle + D_\psi(\nabla\psi^*(\tilde{z}_i), \tilde{x}_{t_0}) - D_\psi(\tilde{x}^*, \tilde{x}_{t_0}).$$

Hence, using the definition of G_i, U_i, L_i we have

$$\begin{aligned}
&A_{i+1}G_{i+1} - A_iG_i \\
&\quad = (A_{i+1}f(\tilde{x}_{i+1}) - A_if(\tilde{x}_i)) - A_{i+1}L_{i+1} + A_iL_i \\
&\quad \stackrel{①}{=} (A_if(\tilde{x}_{i+1}) - A_if(\tilde{x}_i) + a_{i+1}f(\tilde{x}_{i+1})) \\
&\qquad - \sum_{j=1}^{i+1} a_jf(\tilde{x}_j) - \sum_{j=1}^{i+1}\frac{a_j}{\gamma_{\mathrm{n}}}\langle\nabla f(\tilde{x}_j), \nabla\psi^*(\tilde{z}_{i+1}) - \tilde{x}_j\rangle - D_\psi(\nabla\psi^*(\tilde{z}_{i+1}), \tilde{x}_{t_0}) \\
&\qquad + \sum_{j=1}^{i} a_jf(\tilde{x}_j) + \sum_{j=1}^{i}\frac{a_j}{\gamma_{\mathrm{n}}}\langle\nabla f(\tilde{x}_j), \nabla\psi^*(\tilde{z}_i) - \tilde{x}_j\rangle + D_\psi(\nabla\psi^*(\tilde{z}_i), \tilde{x}_{t_0}) \\
&\quad \stackrel{②}{=} A_i(f(\tilde{x}_{i+1}) - f(\tilde{x}_i)) - \langle\frac{a_{i+1}}{\gamma_{\mathrm{n}}}\nabla f(\tilde{x}_{i+1}), \nabla\psi^*(\tilde{z}_{i+1}) - \tilde{x}_{i+1}\rangle \\
&\qquad + \sum_{j=1}^{i}\langle\frac{a_j}{\gamma_{\mathrm{n}}}\nabla f(\tilde{x}_j), \nabla\psi^*(\tilde{z}_i) - \nabla\psi^*(\tilde{z}_{i+1})\rangle \\
&\qquad [-\langle\nabla\psi(\tilde{x}_{t_0}), \nabla\psi^*(\tilde{z}_i) - \nabla\psi^*(\tilde{z}_{i+1})\rangle + \psi(\nabla\psi^*(\tilde{z}_i)) - \psi(\nabla\psi^*(\tilde{z}_{i+1}))]
\end{aligned}$$

$$\overset{③}{=} A_i(f(\tilde{x}_{i+1}) - f(\tilde{x}_i)) - \langle \frac{a_{i+1}}{\gamma_n}\nabla f(\tilde{x}_{i+1}), \nabla\psi^*(\tilde{z}_{i+1}) - \tilde{x}_{i+1}\rangle - D_{\psi^*}(\tilde{z}_i, \tilde{z}_{i+1})$$

$$\overset{④}{\leq} \langle \nabla f(\tilde{x}_{i+1}), (A_i\hat{\gamma}_i + \frac{a_{i+1}}{\gamma_n})\tilde{x}_{i+1} - \hat{\gamma}_i A_i \tilde{x}_i - \frac{a_{i+1}}{\gamma_n}\nabla\psi^*(\tilde{z}_{i+1})\rangle - D_{\psi^*}(\tilde{z}_i, \tilde{z}_{i+1}) + A_i\hat{\varepsilon}_i.$$

In ① we write down the definitions of L_{i+1} and L_i and split the first summand so it is clear that in ② we cancel all the $a_j f(\tilde{x}_j)$. In ② we also cancel some terms involved in the inner products, we write the definitions of the Bregman divergences and cancel some of their terms. For equality ③, we recall $\tilde{z}_i = \nabla\psi(\tilde{x}_{t_0}) - \sum_{j=1}^{i}\frac{a_j}{\gamma_n}\nabla f(\tilde{x}_j)$ so we use this fact and $\psi^*(\tilde{z}) = \langle\nabla\psi^*(\tilde{z}), \tilde{z}\rangle - \psi(\nabla\psi^*(\tilde{z}))$ (which holds by Fact 2.1.6) for $\tilde{z} = \tilde{z}_i$ and $\tilde{z} = \tilde{z}_{i+1}$ to conclude that the last two lines equal $-D_{\psi^*}(\tilde{z}_i, \tilde{z}_{i+1})$. Inequality ④ uses (3.5.4). ■

We show now how to cancel out the discretization error by an approximate implementation of implicit Euler discretization of (3.5.3). Note that we need to take into account the tilted-convexity assumption (3.2.3) instead of the usual convexity assumption. According to the previous lemma, we can set $\tilde{x}_{i+1}$ so that the right hand side of the inner product in the bound of E_{i+1} is 0. Assume for the moment, that the point $\tilde{x}_{i+1}$ we are going to compute satisfies the assumption of the previous lemma for some $\hat{\gamma}_i \in [\gamma_p, 1/\gamma_n]$. Thus, the implicit equation that defines the ideal method we would like to have is

$$\tilde{x}_{i+1} = \frac{\hat{\gamma}_i A_i}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_n}\tilde{x}_i + \frac{a_{i+1}/\gamma_n}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_n}\nabla\psi^*(\tilde{z}_i - \frac{a_{i+1}}{\gamma_n}\nabla f(\tilde{x}_{i+1})).$$

Note that $\tilde{x}_{i+1}$ is a convex combination of the other two points so it stays in Q. Indeed, the initial point is in Q and by (3.5.2) we have that $\nabla\psi^*(\tilde{z}_j) \in Q$ for all $j \geq 0$. However this method is implicit and possibly computationally expensive to implement. Nonetheless, two steps of a fixed point iteration procedure of this equation will be enough to have discretization error that is bounded by the term $A_i\hat{\varepsilon}_i$: the last term of our bound. The error in the bound of E_{i+1} that the inner product incurs is compensated by the Bregman divergence term. In such a case, the equations of this method become, for $i \geq 0$:

$$\begin{cases} \tilde{\chi}_i = \frac{\hat{\gamma}_i A_i}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_n}\tilde{x}_i + \frac{a_{i+1}/\gamma_n}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_n}\nabla\psi^*(\tilde{z}_i); & \tilde{\zeta}_i = \tilde{z}_i - \frac{a_{i+1}}{\gamma_n}\nabla f(\tilde{\chi}_i) \\ \tilde{x}_{i+1} = \frac{\hat{\gamma}_i A_i}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_n}\tilde{x}_i + \frac{a_{i+1}/\gamma_n}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_n}\nabla\psi^*(\tilde{\zeta}_i); & \tilde{z}_{i+1} = \tilde{z}_i - \frac{a_{i+1}}{\gamma_n}\nabla f(\tilde{x}_{i+1}) \end{cases} \tag{3.5.5}$$

We prove now that this indeed leads to an accelerated algorithm. After this, we will show that we can perform a binary search at each iteration, to ensure that even if we do not know $\tilde{x}_{i+1}$ a priori, we can compute a $\hat{\gamma}_i \in [\gamma_p, 1/\gamma_n]$ satisfying assumption (3.5.4). This will only add a log factor to the overall complexity.

Lemma 3.5.2. *Consider the method given in (3.5.5), starting from an arbitrary point $\tilde{x}_0 \in Q$ with $\tilde{z}_0 = \nabla\psi(\tilde{x}_0)$ and $A_0 = 0$. Assume we can compute $\hat{\gamma}_i$ such that $\tilde{x}_{i+1}$ satisfies (3.5.4). Then, the error from Lemma 3.5.1 is bounded by*

$$E_{i+1} \leq \frac{a_{i+1}}{\gamma_n}\langle\nabla f(\tilde{x}_{i+1}) - \nabla f(\tilde{\chi}_i), \nabla\psi^*(\tilde{\zeta}_i) - \nabla\psi^*(\tilde{z}_{i+1})\rangle - D_{\psi^*}(\tilde{\zeta}_i, \tilde{z}_{i+1}) - D_{\psi^*}(\tilde{z}_i, \tilde{\zeta}_i) + A_i\hat{\varepsilon}_i.$$

Proof Using Lemma 3.5.1 and the third line of (3.5.5) we have

$$\begin{aligned} E_{i+1} - A_i\hat{\varepsilon}_i &\le \frac{a_{i+1}}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{x}_{i+1}), \nabla\psi^*(\tilde{\zeta}_i) - \nabla\psi^*(\tilde{z}_{i+1})\rangle - D_{\psi^*}(\tilde{z}_i, \tilde{z}_{i+1}) \\ &\le \frac{a_{i+1}}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{x}_{i+1}) - \nabla f(\tilde{\chi}_i) + \nabla f(\tilde{\chi}_i), \nabla\psi^*(\tilde{\zeta}_i) - \nabla\psi^*(\tilde{z}_{i+1})\rangle - D_{\psi^*}(\tilde{z}_i, \tilde{z}_{i+1}) \end{aligned}$$

By the definition of $\tilde{\zeta}_i$ we have $(a_{i+1}/\gamma_{\mathrm{n}})\nabla f(\tilde{\chi}_i) = \tilde{z}_i - \tilde{\zeta}_i$. Using this fact and the triangle equality of Bregman divergences Lemma 2.1.8.5, we obtain

$$\begin{aligned} \frac{a_{i+1}}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{\chi}_i), \nabla\psi^*(\tilde{\zeta}_i) - \nabla\psi^*(\tilde{z}_{i+1})\rangle &= \langle \tilde{z}_i - \tilde{\zeta}_i, \nabla\psi^*(\tilde{\zeta}_i) - \nabla\psi^*(\tilde{z}_{i+1})\rangle \\ &= D_{\psi^*}(\tilde{z}_i, \tilde{z}_{i+1}) - D_{\psi^*}(\tilde{\zeta}_i, \tilde{z}_{i+1}) - D_{\psi^*}(\tilde{z}_i, \tilde{\zeta}_i). \end{aligned}$$

The lemma follows after combining these two equations. ■

Theorem 3.5.3. *Let Q be a closed convex set of diameter D. Let $f : Q \to \mathbb{R}$ be an $\tilde{L}$-smooth tilted-convex function with constants $\gamma_{\mathrm{n}}, \gamma_{\mathrm{p}}$. Assume there is a point $\tilde{x}^* \in Q$ such that $\nabla f(\tilde{x}^*) = 0$. Let $\psi : Q \to \mathbb{R}$ be a σ-strongly convex map. Let $\tilde{x}_i, \tilde{z}_i, \tilde{\chi}_i, \tilde{\zeta}_i$ be updated according to (3.5.5), for $i \ge 0$ starting from an arbitrary initial point $\tilde{x}_0 \in Q$ with $\tilde{z}_0 = \nabla\psi(\tilde{x}_0)$ and $A_0 = 0$, assuming we can find $\hat{\gamma}_i$ at each iteration satisfying (3.5.4). If $\tilde{L}a_{i+1}^2/\gamma_{\mathrm{n}}\sigma \le a_{i+1} + A_i\gamma_{\mathrm{n}}\gamma_{\mathrm{p}}$, then for all $T \ge 1$ we have*

$$f(\tilde{x}_T) - f(\tilde{x}^*) \le \frac{D_\psi(\tilde{x}^*, \tilde{\chi}_0)}{A_T} + \sum_{i=1}^{T-1}\frac{A_i\hat{\varepsilon}_i}{A_T}.$$

In particular, if $a_i = \frac{i}{2}\frac{\sigma}{\tilde{L}}\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}$, $\psi(\tilde{x}) = \frac{\sigma}{2}\|\tilde{x}\|^2$, $\hat{\varepsilon}_i = \frac{A_T\varepsilon}{2(T-1)A_i}$ and $T = \left\lceil \sqrt{4\tilde{L}\|\tilde{x}_0 - \tilde{x}^*\|^2/(\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}\varepsilon)} \right\rceil = O(\sqrt{\tilde{L}/(\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}\varepsilon)})$ then $f(\tilde{x}_T) - f(\tilde{x}^*) < \varepsilon$.

Proof We bound the right hand side of the discretization error given by Lemma 3.5.2. Define $a = \|\nabla\psi^*(\tilde{\zeta}_i) - \nabla\psi^*(\tilde{z}_{i+1})\|$ and $b = \|\nabla\psi^*(\tilde{\zeta}_i) - \nabla\psi^*(\tilde{z}_i)\|$. We have

$$\begin{aligned} E_{i+1} - A_i\hat{\varepsilon}_i &\overset{①}{\le} \frac{a_{i+1}}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{x}_{i+1}) - \nabla f(\tilde{\chi}_i), \nabla\psi^*(\tilde{\zeta}_i) - \nabla\psi^*(\tilde{z}_{i+1})\rangle - D_{\psi^*}(\tilde{\zeta}_i, \tilde{z}_{i+1}) - D_{\psi^*}(\tilde{z}_i, \tilde{\zeta}_i) \\ &\overset{②}{\le} \frac{a_{i+1}}{\gamma_{\mathrm{n}}}\tilde{L}\|\tilde{x}_{i+1} - \tilde{\chi}_i\| \cdot a - D_{\psi^*}(\tilde{\zeta}_i, \tilde{z}_{i+1}) - D_{\psi^*}(\tilde{z}_i, \tilde{\zeta}_i) \\ &\overset{③}{\le} \frac{a_{i+1}}{\gamma_{\mathrm{n}}}\tilde{L}\|\tilde{x}_{i+1} - \tilde{\chi}_i\| \cdot a - \frac{\sigma}{2}(a^2 + b^2) \\ &\overset{④}{\le} \frac{a_{i+1}^2/\gamma_{\mathrm{n}}^2}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_{\mathrm{n}}}\tilde{L} \cdot ab - \frac{\sigma}{2}(a^2 + b^2) \overset{⑤}{\le} ab\left(\frac{a_{i+1}^2/\gamma_{\mathrm{n}}^2}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_{\mathrm{n}}}\tilde{L} - \sigma\right). \end{aligned}$$

Here ① follows from Lemma 3.5.2, ② uses the Cauchy-Schwartz inequality and gradient Lipschitzness, which is equivalent to smoothness for differentiable tilted convex functions, as we point out in the proof of Lemma 3.2.3. In ③, we used Lemma 2.1.8.2, and ④ uses the fact that by the definition of the method (3.5.5) we have $\tilde{x}_{i+1} - \tilde{\chi}_i = \frac{a_{i+1}/\gamma_{\mathrm{n}}}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_{\mathrm{n}}}(\nabla\psi^*(\tilde{\zeta}_i) - \nabla\psi^*(\tilde{z}_i))$. Finally ⑤ uses $-(a^2 + b^2) \le -2ab$, which comes from $(a - b)^2 \ge 0$. By the previous inequality, if we want $E_{i+1} \le A_i\hat{\varepsilon}_i$, it is enough to guarantee the right hand side of the last expression is ≤ 0

which is implied by

$$\frac{\tilde{L}}{\sigma\gamma_{\mathrm{n}}}a_{i+1}^2 \leq a_{i+1} + A_i\gamma_{\mathrm{n}}\gamma_{\mathrm{p}}, \tag{3.5.6}$$

since $\gamma_{\mathrm{p}} \leq \hat{\gamma}_i$. And this is the assumption we made in the theorem. By inspection, if we use the value in the second part of the statement of the theorem $a_i = \frac{i}{2}\cdot\frac{\sigma}{\tilde{L}}\cdot\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}$ into the previous inequality and noting that $A_i = \frac{i(i+1)}{4}\cdot\frac{\sigma}{\tilde{L}}\cdot\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}$ we prove that the previous inequality is satisfied:

$$\begin{aligned}\frac{\tilde{L}}{\sigma\gamma_{\mathrm{n}}}a_{i+1}^2 &= \frac{(i+1)^2}{4}\cdot\frac{\sigma}{\tilde{L}}\cdot\gamma_{\mathrm{n}}^3\gamma_{\mathrm{p}}^2 \leq \left(\frac{i+1}{2}+\frac{i(i+1)}{4}\right)\frac{\sigma}{\tilde{L}}\cdot\gamma_{\mathrm{n}}^3\gamma_{\mathrm{p}}^2\\ &\leq \frac{i+1}{2}\frac{\sigma}{\tilde{L}}\cdot\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}} + \frac{i(i+1)}{4}\frac{\sigma}{\tilde{L}}\cdot\gamma_{\mathrm{n}}^3\gamma_{\mathrm{p}}^2 = a_{i+1} + A_i\gamma_{\mathrm{n}}\gamma_{\mathrm{p}}.\end{aligned}$$

So this choice, and in particular any choice that satisfies (3.5.6), guarantees discretization error $E_{i+1} \leq A_i\hat{\varepsilon}_i$. By the definition of G_i and E_i we have

$$f(\tilde{x}_T) - f(\tilde{x}^*) \leq \frac{A_1G_1}{A_T} + \sum_{i=1}^{T-1}\frac{A_i\hat{\varepsilon}_i}{A_T}$$

So it only remains to bound the initial gap G_1. In order to do this, we note that the initial conditions and the method imply the following computation of the first points, from $\tilde{x}_0 \in Q$, which is an arbitrary initial point:

$$\begin{cases}\tilde{z}_0 = \nabla\psi(\tilde{x}_0)\\ \tilde{\chi}_0 = \frac{\hat{\gamma}_0A_0}{A_0\hat{\gamma}_0+a_1/\gamma_{\mathrm{n}}}\tilde{x}_0 + \frac{a_1/\gamma_{\mathrm{n}}}{A_0\hat{\gamma}_0+a_1/\gamma_{\mathrm{n}}}\nabla\psi^*(\tilde{z}_0) = \nabla\psi^*(\nabla\psi(\tilde{x}_0)) = \tilde{x}_0\\ \tilde{\zeta}_0 = \tilde{z}_0 - \frac{a_1}{\gamma_{\mathrm{n}}}\nabla f(\tilde{\chi}_0) = \tilde{z}_0 - \frac{a_1}{\gamma_{\mathrm{n}}}\nabla f(\tilde{x}_0)\\ \tilde{x}_1 = \frac{\hat{\gamma}_0A_0}{A_0\hat{\gamma}_0+a_1/\gamma_{\mathrm{n}}}\tilde{x}_0 + \frac{a_1/\gamma_{\mathrm{n}}}{A_0\hat{\gamma}_0+a_1/\gamma_{\mathrm{n}}}\nabla\psi^*(\tilde{\zeta}_0) = \nabla\psi^*(\tilde{\zeta}_0)\end{cases} \tag{3.5.7}$$

We have used $A_0 = 0$. Note this first iteration does not depend on $\hat{\gamma}_0$. Also, by using this discretization we start at $\tilde{x}_0$ so we modify the definition of the lower bound (3.5.1) so the regularizer added measures the distance from $\tilde{x}_0$. This change of $\tilde{x}_{t_0}$ to $\tilde{x}_0 = \tilde{\chi}_0$ only changes the initial gap. Thus, the first lower bound computed is

$$L_1 = f(\tilde{x}_1) + \frac{1}{\gamma_{\mathrm{n}}}\langle\nabla f(\tilde{x}_1), \nabla\psi^*(\tilde{z}_1) - \tilde{x}_1\rangle + \frac{1}{A_1}D_\psi(\nabla\psi^*(\tilde{z}_1), \tilde{\chi}_0) - \frac{1}{A_1}D_\psi(\tilde{x}^*, \tilde{\chi}_0).$$

Using $a_1 = A_1$, $\tilde{x}_1 = \nabla\psi^*(\tilde{\zeta}_0)$, $(a_1/\gamma_{\mathrm{n}})\nabla f(\tilde{\chi}_0) = \tilde{z}_0 - \tilde{\zeta}_0$, and the triangle equality for Bregman divergences Lemma 2.1.8.5 we obtain

$$\begin{aligned}\frac{1}{\gamma_{\mathrm{n}}}\langle\nabla f(\tilde{\chi}_0), \nabla\psi^*(\tilde{z}_1) - \tilde{x}_1\rangle &= \frac{1}{A_1}\langle\tilde{z}_0 - \tilde{\zeta}_0, \nabla\psi^*(\tilde{z}_1) - \nabla\psi^*(\tilde{\zeta}_0)\rangle\\ &= \frac{1}{A_1}\left(D_{\psi^*}(\tilde{z}_0, \tilde{\zeta}_0) - D_{\psi^*}(\tilde{z}_0, \tilde{z}_1) + D_{\psi^*}(\tilde{\zeta}_0, \tilde{z}_1)\right).\end{aligned} \tag{3.5.8}$$

On the other hand, by smoothness of f and the initial condition we have

$$\frac{1}{\gamma_{\mathrm{n}}}\langle\nabla f(\tilde{x}_1) - \nabla f(\tilde{\chi}_0), \nabla\psi^*(\tilde{z}_1) - \tilde{x}_1\rangle \geq -\frac{\tilde{L}}{\gamma_{\mathrm{n}}}\|\nabla\psi^*(\tilde{\zeta}_0) - \tilde{\chi}_0\|\|\nabla\psi^*(\tilde{z}_1) - \tilde{x}_1\|. \tag{3.5.9}$$

We can now finally bound G_1:

$$\begin{aligned}
G_1 &\overset{①}{\leq} \frac{\tilde{L}}{\gamma_{\mathrm{n}}}\|\nabla\psi^*(\tilde{\zeta}_0)-\tilde{\chi}_0\|\cdot\|\nabla\psi^*(\tilde{z}_1)-\tilde{x}_1\| + \frac{-D_{\psi^*}(\tilde{z}_0,\tilde{\zeta}_0)-D_{\psi^*}(\tilde{\zeta}_0,\tilde{z}_1)+D_{\psi}(\tilde{x}^*,\tilde{\chi}_0)}{A_1}\\
&\overset{②}{\leq} \frac{\tilde{L}}{\gamma_{\mathrm{n}}}\|\nabla\psi^*(\tilde{\zeta}_0)-\tilde{\chi}_0\|\cdot\|\nabla\psi^*(\tilde{z}_1)-\tilde{x}_1\|\\
&\quad - \frac{\sigma}{2A_1}\left(\|\nabla\psi^*(\tilde{\zeta}_0)-\tilde{\chi}_0\|^2+\|\nabla\psi^*(\tilde{z}_1)-\tilde{x}_1\|^2\right)+\frac{1}{A_1}D_{\psi}(\tilde{x}^*,\tilde{\chi}_0)\\
&\overset{③}{\leq} \|\nabla\psi^*(\tilde{\zeta}_0)-\tilde{\chi}_0\|\cdot\|\nabla\psi^*(\tilde{z}_1)-\tilde{x}_1\|\left(\frac{\tilde{L}}{\gamma_{\mathrm{n}}}-\frac{\sigma}{A_1}\right)+\frac{1}{A_1}D_{\psi}(\tilde{x}^*,\tilde{\chi}_0)\overset{④}{\leq}\frac{1}{A_1}D_{\psi}(\tilde{x}^*,\tilde{\chi}_0).
\end{aligned}$$

We used in ① the definition of $G_1 = U_1 - L_1 = f(\tilde{x}_1) - L_1$ and we bound the inner product in L_1 using $-((3.5.8)+(3.5.9))$. Also, since $\tilde{z}_0 = \nabla\psi(\tilde{\chi}_0)$ we have $D_{\psi^*}(\tilde{z}_0,\tilde{z}_1) = D_{\psi^*}(\nabla\psi(\tilde{\chi}_0),\tilde{z}_1) = D_{\psi}(\nabla\psi^*(\tilde{z}_1),\tilde{\chi}_0)$, so we can cancel two of the Bregman divergences. In ②, we used Lemma 2.1.8.6 and Lemma 2.1.8.2, $\nabla\psi^*(\tilde{z}_0) = \tilde{x}_0 = \tilde{\chi}_0$, and $\nabla\psi^*(\tilde{\zeta}_0) = \tilde{x}_1$. In ③ we used again the inequality $-(a^2+b^2) \leq -2ab$. Finally ④ is deduced from $A_1 = a_1 \leq \sigma\gamma_{\mathrm{n}}/\tilde{L}$ which comes from the assumption $\tilde{L}a_{i+1}^2/\gamma_{\mathrm{n}}\sigma \leq a_{i+1} + A_i\gamma_{\mathrm{n}}\gamma_{\mathrm{p}}$ for $i = 0$.

The first part of the theorem follows. The second one is a straightforward application of the first one as we see below. Indeed, taking into account $A_T = \frac{T(T+1)\sigma\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}}{4\tilde{L}}$, and the choice of $T = \left\lceil\sqrt{4\tilde{L}\|\tilde{x}_0-\tilde{x}^*\|^2/(\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}\varepsilon)}\right\rceil$, $\psi(\tilde{x}) = \frac{\sigma}{2}\|\tilde{x}\|^2$, and $\hat{\varepsilon}_i = \frac{A_T\varepsilon}{2(T-1)A_i}$ we derive the second statement.

$$f(\tilde{x}_T) - f(\tilde{x}^*) \leq \frac{A_1G_1}{A_T} + \sum_{i=1}^{T-1}\frac{A_i\hat{\varepsilon}_i}{A_T} \leq \frac{\frac{\sigma}{2}\|\tilde{x}_0-\tilde{x}^*\|^2}{A_T} + \frac{\varepsilon}{2} < \frac{2\tilde{L}\|\tilde{x}_0-\tilde{x}^*\|^2}{\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}T^2} + \frac{\varepsilon}{2} \leq \varepsilon.$$

■

We present now the final lemma, that proves that $\hat{\gamma}_i$ can be found efficiently. As we advanced in the sketch of the proof, we use a binary search. The idea behind it is that due to tilted-convexity (3.2.3) we satisfy the equation for $\hat{\gamma}_i = \frac{1}{\gamma_{\mathrm{n}}}$ or $\hat{\gamma}_i = \gamma_{\mathrm{p}}$, or there is $\hat{\gamma}_i \in (\gamma_{\mathrm{p}}, 1/\gamma_{\mathrm{n}})$ such that $\langle\nabla f(\tilde{x}_{i+1}), \tilde{x}_{i+1} - \tilde{x}_i\rangle = 0$. The existence of $\tilde{x}^*$ that satisfies $\nabla f(\tilde{x}^*) = 0$ along with the boundedness of Q and smoothness, imply the Lipschitzness of f. Both Lipschitzness and smoothness allow to prove that a binary search finds efficiently a suitable point.

Lemma 3.5.4. *Let $Q \subseteq \mathbb{R}^n$ be a convex set of diameter $2\tilde{R}$. Let $f : Q \to \mathbb{R}$ be a function that satisfies tilted-convexity (3.2.3), is $\tilde{L}$ smooth and such that there is $\tilde{x}^* \in Q$ such that $\nabla f(\tilde{x}^*) = 0$. Let the strongly convex parameter of $\psi(\cdot)$ be $\sigma = O(1)$. Let $i \geq 1$ be an index. Given two points $\tilde{x}_i, \tilde{z}_i \in Q$ and the method in (3.2.6) using the learning rates $a_i = \frac{i}{2}\cdot\frac{\sigma}{\tilde{L}}\cdot\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}$ prescribed in Theorem 3.5.3, we can compute $\hat{\gamma}_i$ satisfying (3.5.4), i.e.,*

$$f(\tilde{x}_{i+1}) - f(\tilde{x}_i) \leq \hat{\gamma}_i\langle\nabla f(\tilde{x}_{i+1}), \tilde{x}_{i+1} - \tilde{x}_i\rangle + \hat{\varepsilon}_i. \tag{3.5.10}$$

And the computation of $\hat{\gamma}_i$ requires no more than $O\left(\log\left(\frac{\tilde{L}\tilde{R}}{\gamma_{\mathrm{n}}\hat{\varepsilon}_i}\cdot i\right)\right)$ queries to the gradient oracle.

Proof Let $\hat{\Gamma}_i(\lambda) : [\frac{a_{i+1}}{A_{i+1}}, \frac{a_{i+1}/\gamma_n}{A_i\gamma_p + a_{i+1}/\gamma_n}] \to \mathbb{R}$ be defined as

$$\hat{\Gamma}_i\left(\frac{a_{i+1}/\gamma_n}{A_i\tilde{\mathbf{x}} + a_{i+1}/\gamma_n}\right) = \tilde{\mathbf{x}}, \text{ for } \tilde{\mathbf{x}} \in [\gamma_p, \frac{1}{\gamma_n}]. \tag{3.5.11}$$

By monotonicity, it is well defined. Let $\tilde{x}_{i+1}^{\lambda}$ be the point computed by one iteration of (3.2.6) using the parameter $\hat{\gamma}_i = \hat{\Gamma}_i(\lambda)$. Likewise, we define the rest of the points in iteration (3.2.6) depending on λ. We first try $\hat{\gamma}_i = 1/\gamma_n$ and $\hat{\gamma}_i = \gamma_p$ and use any of them if they satisfy the conditions. If neither of them do, it means that for the first choice we had $\langle \nabla f(\tilde{x}_{i+1}^{\lambda_1}), \tilde{x}_{i+1}^{\lambda_1} - \tilde{x}_i\rangle < 0$ and for the second one, it is $\langle \nabla f(\tilde{x}_{i+1}^{\lambda_2}), \tilde{x}_{i+1}^{\lambda_2} - \tilde{x}_i\rangle > 0$, for $\lambda_1 = \hat{\Gamma}_i^{-1}(1/\gamma_n)$ and $\lambda_2 = \hat{\Gamma}_i^{-1}(\gamma_p)$. Therefore, by continuity, there is $\lambda^* \in [\lambda_1, \lambda_2]$ such that $\langle \nabla f(\tilde{x}_{i+1}^{\lambda^*}), \tilde{x}_{i+1}^{\lambda^*} - \tilde{x}_i\rangle = 0$. The continuity condition is easy to prove. We omit it because it is derived from the Lipschitzness condition that we will prove below. Such a point satisfies (3.5.4) for $\hat{\varepsilon}_i = 0$. We will prove that the function $\mathcal{G}_i : [\frac{a_{i+1}}{A_{i+1}}, \frac{a_{i+1}/\gamma_n}{A_i\gamma_p + a_{i+1}/\gamma_n}] \to \mathbb{R}$, defined as

$$\mathcal{G}_i(\lambda) \stackrel{\text{def}}{=} -\hat{\Gamma}_i(\lambda)\langle \nabla f(\tilde{x}_{i+1}^{\lambda}), \tilde{x}_{i+1}^{\lambda} - \tilde{x}_i\rangle + (f(\tilde{x}_{i+1}^{\lambda}) - f(\tilde{x}_i)), \tag{3.5.12}$$

is Lipschitz so we can guarantee that (3.5.4) holds for a large enough interval around λ^*. Finally, we will be able to perform a binary search to efficiently find a point in such interval or another interval around another point that satisfies that the inner product is 0.

So

$$\begin{aligned}
|\mathcal{G}_i(\lambda) - \mathcal{G}_i(\lambda')| \leq\ & |f(\tilde{x}_{i+1}^{\lambda}) - f(\tilde{x}_{i+1}^{\lambda'})| \\
& + |\hat{\Gamma}_i(\lambda')| \cdot |\langle \nabla f(\tilde{x}_{i+1}^{\lambda'}), \tilde{x}_{i+1}^{\lambda'} - \tilde{x}_i\rangle - \langle \nabla f(\tilde{x}_{i+1}^{\lambda}), \tilde{x}_{i+1}^{\lambda} - \tilde{x}_i\rangle| \\
& + |\langle \nabla f(\tilde{x}_{i+1}^{\lambda}), \tilde{x}_{i+1}^{\lambda} - \tilde{x}_i\rangle| \cdot |\hat{\Gamma}_i(\lambda') - \hat{\Gamma}_i(\lambda)|
\end{aligned} \tag{3.5.13}$$

We have used the triangular inequality and the inequality

$$|\alpha_1\beta_1 - \alpha_2\beta_2| \leq |\alpha_1||\beta_1 - \beta_2| + |\beta_2||\alpha_1 - \alpha_2|, \tag{3.5.14}$$

which is a direct consequence of the triangular inequality, after adding and subtracting $\alpha_1\beta_2$ in the $|\cdot|$ on the left hand side. We bound each of the three summands of the previous inequality separately, but first we bound the following which will be useful for our other bounds,

$$\begin{aligned}
\|\tilde{x}_{i+1}^{\lambda'} - \tilde{x}_{i+1}^{\lambda}\| &\overset{①}{=} \|(\lambda'\nabla\psi^*(\tilde{\zeta}_i^{\lambda'}) + (1-\lambda')\tilde{x}_i) - (\lambda\nabla\psi^*(\tilde{\zeta}_i^{\lambda}) + (1-\lambda)\tilde{x}_i)\| \\
&\overset{②}{\leq} \|\nabla\psi^*(\tilde{\zeta}_i^{\lambda}) - \tilde{x}_i\||\lambda' - \lambda| + \|\lambda'\nabla\psi^*(\tilde{\zeta}_i^{\lambda'}) - \lambda'\nabla\psi^*(\tilde{\zeta}_i^{\lambda})\| \\
&\overset{③}{\leq} 2\tilde{R}|\lambda - \lambda'| + \|\nabla\psi^*(\tilde{\zeta}_i^{\lambda'}) - \nabla\psi^*(\tilde{\zeta}_i^{\lambda})\| \overset{④}{\leq} 2\tilde{R}|\lambda - \lambda'| + \frac{a_{i+1}}{\gamma_n\sigma}\|\nabla f(\tilde{\chi}_i^{\lambda}) - \nabla f(\tilde{\chi}_i^{\lambda'})\| \\
&\overset{⑤}{\leq} 2\tilde{R}|\lambda - \lambda'| + \frac{a_{i+1}\tilde{L}}{\gamma_n\sigma}\|\tilde{\chi}_i^{\lambda} - \tilde{\chi}_i^{\lambda'}\| \overset{⑥}{\leq} \left(2\tilde{R} + \frac{2a_{i+1}\tilde{L}\tilde{R}}{\gamma_n\sigma}\right)|\lambda - \lambda'|
\end{aligned} \tag{3.5.15}$$

Here, ① uses the definition of $\tilde{x}_{i+1}^{\lambda}$ as a convex combination of $\tilde{x}_i$ and $\nabla\psi^*(\tilde{\zeta}_i^{\lambda})$. ② adds and substracts $\lambda'\nabla\psi^*(\tilde{\zeta}_i^{\lambda})$, groups terms and uses the triangular inequality. In ③ we use the fact

that the diameter of Q is $2\tilde{R}$ and bound $\lambda' \leq 1$, and $|\lambda| \leq 1$. ④ uses the $\frac{1}{\sigma}$ Lipschitzness of $\nabla\psi^*(\cdot)$, which is a consequence of the σ-strong convexity of $\psi(\cdot)$. ⑤ uses the smoothness of f. In ⑥, from the definition of $\tilde{\chi}_i^\lambda$ we have that $\|\tilde{\chi}_i^\lambda - \tilde{\chi}_i^{\lambda'}\| \leq \|\tilde{x}_i - \tilde{z}_i\||\lambda - \lambda'|$. We bounded this further using the diameter of Q.

Note that f is Lipschitz over Q. By the existence of $\tilde{x}^*$, $\tilde{L}$-smoothness, and the diameter of Q we have $\|\nabla f(\tilde{x})\| = \|\nabla f(\tilde{x}) - \nabla f(\tilde{x}^*)\| \leq \tilde{L}\|\tilde{x} - \tilde{x}^*\| \leq 2R\tilde{L}$. So the Lipschitz constant $L_{\mathtt{p}}$ of f is $L_{\mathtt{p}} \leq 2R\tilde{L}$. Now we can proceed and bound the three summands of (3.5.13). The first one reduces to the inequality above after using Lipschitzness of $f(\cdot)$:

$$|f(\tilde{x}_{i+1}^\lambda) - f(\tilde{x}_{i+1}^{\lambda'})| \leq L_{\mathtt{p}}\|\tilde{x}_{i+1}^{\lambda'} - \tilde{x}_{i+1}^\lambda\|. \tag{3.5.16}$$

We prove Lipschitzness of $\hat{\Gamma}_i$. Note that

$$|(\hat{\Gamma}_i^{-1})'(\tilde{\mathtt{x}})| = \left|\frac{A_i a_{i+1}/\gamma_{\mathtt{n}}}{(A_i\tilde{\mathtt{x}} + a_{i+1}/\gamma_{\mathtt{n}})^2}\right| \geq \frac{\gamma_{\mathtt{n}} A_i a_{i+1}}{A_{i+1}^2}, \tag{3.5.17}$$

so $\hat{\Gamma}_i'(\lambda)$ is bounded by $A_{i+1}^2/(\gamma_{\mathtt{n}} A_i a_{i+1})$ for any λ. In order to bound the second summand, we use $\tilde{\mathtt{x}} \in [\gamma_{\mathtt{p}}, 1/\gamma_{\mathtt{n}}]$ and obtain $|\hat{\Gamma}_i(\lambda)| \leq \frac{1}{\gamma_{\mathtt{n}}}$. For the second factor, we add and subtract $\langle \nabla f(\tilde{x}_{i+1}^\lambda), \tilde{x}_{i+1}^{\lambda'} - \tilde{x}_i\rangle$ and use the triangular inequality and then Cauchy-Schwartz. Thus, we obtain

$$\begin{aligned}
&|\langle \nabla f(\tilde{x}_{i+1}^{\lambda'}), \tilde{x}_{i+1}^{\lambda'} - \tilde{x}_i\rangle - \langle \nabla f(\tilde{x}_{i+1}^\lambda), \tilde{x}_{i+1}^\lambda - \tilde{x}_i\rangle| \\
&\leq \|\nabla f(\tilde{x}_{i+1}^\lambda)\| \cdot \|\tilde{x}_{i+1}^{\lambda'} - \tilde{x}_{i+1}^\lambda\| + \|\nabla f(\tilde{x}_{i+1}^{\lambda'}) - \nabla f(\tilde{x}_{i+1}^\lambda)\| \cdot \|\tilde{x}_{i+1}^{\lambda'} - \tilde{x}_i\| \\
&\overset{①}{\leq} (L_{\mathtt{p}} + 2\tilde{L}\tilde{R})\|\tilde{x}_{i+1}^{\lambda'} - \tilde{x}_{i+1}^\lambda\|.
\end{aligned} \tag{3.5.18}$$

In ①, we used Lipschitzness to bound the first factor. We also used the diameter of Q to bound the last factor and the smoothness of $f(\cdot)$ to bound the first factor of the second summand.

For the third summand, we will bound the first factor using Cauchy-Schwartz, Lipschitzness of $f(\cdot)$ and the diameter of Q. We just proved in (3.5.17) that $\hat{\Gamma}_i$ is Lipschitz, so use this property for the second factor. The result is the following

$$|\langle \nabla f(\tilde{x}_{i+1}^\lambda), \tilde{x}_{i+1}^\lambda - \tilde{x}_i\rangle| \cdot |\hat{\Gamma}_i(\lambda') - \hat{\Gamma}_i(\lambda)| \leq 2L_{\mathtt{p}}\tilde{R}\frac{A_{i+1}^2}{\gamma_{\mathtt{n}} A_i a_{i+1}}|\lambda' - \lambda|. \tag{3.5.19}$$

Applying the bounds of the three summands (3.5.16), (3.5.17), (3.5.18), (3.5.19) into (3.5.13) we obtain the inequality $|\mathcal{G}_i(\lambda') - \mathcal{G}_i(\lambda)| \leq \hat{L}|\lambda' - \lambda|$ for

$$\hat{L} = \left(2\tilde{R} + \frac{2a_{i+1}\tilde{L}\tilde{R}}{\gamma_{\mathtt{n}}\sigma}\right)\left(L_{\mathtt{p}} + (L_{\mathtt{p}} + 2\tilde{L}\tilde{R})\frac{1}{\gamma_{\mathtt{n}}}\right) + 2L_{\mathtt{p}}\tilde{R}\frac{A_{i+1}^2}{\gamma_{\mathtt{n}} A_i a_{i+1}}.$$

We will use the following to bound $\hat{L}$. If we use the learning rates prescribed in Theorem 3.5.3, namely $a_i = \frac{i\sigma\gamma_{\mathtt{n}}^2\gamma_{\mathtt{p}}}{2L}$ and thus $A_i = \frac{i(i+1)\sigma\gamma_{\mathtt{n}}^2\gamma_{\mathtt{p}}}{4L}$ we can bound $A_{i+1}^2/(A_i a_{i+1}) \leq 4(i+2)$, using that $i \geq 1$. We recall we computed $L_{\mathtt{p}} \leq 2\tilde{R}\tilde{L}$ and that we assumed $\sigma = O(1)$. In Algorithm 5 we use $\sigma = 1$.

On the other hand the initial length of the search interval, which is the domain of definition of $\mathcal{G}_i$ is at most 1 since the interval is in $(0,1)$. Recall we are denoting by λ^* a value such that $\langle \nabla f(\tilde{x}_{i+1}^{\lambda^*}), \tilde{x}_{i+1}^{\lambda^*} - \tilde{x}_i \rangle = 0$ so $\mathcal{G}_i(\lambda^*) \leq 0$. Lipschitzness of G implies that if $\mathcal{G}_i(\lambda^*) \leq 0$ then $\mathcal{G}_i(\lambda) \leq \hat{\varepsilon}_i$ for

$$\lambda \in [\lambda^* - \frac{\hat{\varepsilon}_i}{\hat{L}}, \lambda^* + \frac{\hat{\varepsilon}_i}{\hat{L}}] \cap [\hat{\Gamma}_i^{-1}(1/\gamma_{\mathrm{n}}), \hat{\Gamma}_i^{-1}(\gamma_{\mathrm{p}})].$$

If the extremal points, $\hat{\Gamma}_i^{-1}(1/\gamma_{\mathrm{n}}), \hat{\Gamma}_i^{-1}(\gamma_{\mathrm{p}})$ did not satisfy (3.5.10), then this interval is of length $\frac{2\hat{\varepsilon}_i}{\hat{L}}$ and a point in such interval or another interval that is around another point $\bar{\lambda}^*$ that satisfies $\langle \nabla f(\tilde{x}_{i+1}^{\bar{\lambda}^*}), \tilde{x}_{i+1}^{\bar{\lambda}^*} - \tilde{x}_i \rangle = 0$ can be found with a binary search in at most

$$O\left(\log\left(\frac{\hat{L}}{\hat{\varepsilon}_i}\right)\right) \overset{①}{=} O\left(\log\left(\frac{\tilde{L}\tilde{R}}{\gamma_{\mathrm{n}}\hat{\varepsilon}_i} \cdot i\right)\right)$$

iterations, provided that at each step we can ensure we halve the size of the search interval. The bounds of the previous paragraph are applied in ①.The binary search can be done easily: we start with $[\hat{\Gamma}_i^{-1}(1/\gamma_{\mathrm{n}}), \hat{\Gamma}_i^{-1}(\gamma_{\mathrm{p}})]$ and assume the extremes do not satisfy (3.5.10), so the sign of $\langle \nabla f(\tilde{x}_{i+1}^{\lambda}), \tilde{x}_{i+1}^{\lambda} - \tilde{x}_i \rangle$ is different for each extreme. Each iteration of the binary search queries the midpoint of the current working interval and if (3.5.10) is not satisfied, we keep the half of the interval such that the extremes keep having the sign of $\langle \nabla f(\tilde{x}_{i+1}^{\lambda}), \tilde{x}_{i+1}^{\lambda} - \tilde{x}_i \rangle$ different from each other, ensuring that there is a point in which this expression evaluates to 0 and thus keeping the invariant. We include the pseudocode of this binary search in Algorithm 6. ■

We proceed to prove Theorem 3.2.4, which is an immediate consequence of the previous results.

Proof of Theorem 3.2.4. The proof follows from Theorem 3.5.3, provided that we can find $\hat{\gamma}_i$ satisfying (3.5.4). Lemma 3.5.4 shows that this is possible after performing a logarithmic number of queries to the gradient oracle. Note that given our choice of $\hat{\varepsilon}_i$, T and a_i, the number of queries to the gradient oracle Lemma 3.5.4 requires is no more than $O(\log(\tilde{L}R/\gamma_{\mathrm{n}}\varepsilon))$ for any $i \leq T$. So we find an ε-minimizer of f after $\widetilde{O}(\sqrt{\tilde{L}/(\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}\varepsilon)})$ queries to the gradient oracle. ■

Proof of Theorem 3.2.5. Given the function to optimize $F : \mathcal{M}_K \to \mathbb{R}$ and the geodesic map h, we define $f = F \circ h^{-1}$. Using Lemma 3.2.3 we know that f is $\tilde{L}$-smooth, with $\tilde{L} = O(L)$. Lemma 3.2.2 proves that f satisfies tilted-convexity (3.2.3) for constants γ_{n} and γ_{p} depending on R. So Theorem 3.2.4 applies and the total number of queries to the oracle needed to obtain an ε-minimizer of f is $\widetilde{O}(\sqrt{\tilde{L}/\gamma_{\mathrm{n}}^2\gamma_{\mathrm{p}}\varepsilon}) = \widetilde{O}(\sqrt{L/\varepsilon})$. The result follows, since $f(\tilde{x}_T) - f(\tilde{x}^*) = F(x_T) - F(x^*)$. ■

We recall a few concepts that were assumed during Section 3.2 to better interpret Theorem 3.2.5. We work in the hyperbolic space, or in an open hemisphere. The aim is to minimize a smooth and g-convex function defined on any of these manifolds, or a submanifold of them. The existence of a point x^* that satisfies $\nabla F(x^*) = 0$ is assumed. Starting from an arbitrary point x_0, we let R be a bound of the distance between x_0 and x^*, that is, $R \geq d(x_0, x^*)$. We

Algorithm 6 BinaryLineSearch($\tilde{x}_i, \tilde{z}_i, f, \mathcal{X}, a_{i+1}, A_i, \varepsilon, \tilde{L}, \gamma_n, \gamma_p$)

Input: Points $\tilde{x}_i$, $\tilde{z}_i$, function f, domain $\mathcal{X}$, learning rate a_{i+1}, accumulated learning rate A_i, final target accuracy ε, final number of iterations T, smoothness constant $\tilde{L}$, constants γ_n, γ_p. Define $\hat{\varepsilon}_i \leftarrow (A_T\varepsilon)/(2(T-1)A_i)$ as in Theorem 3.5.3, i.e. with $A_T = T(T+1)\gamma_n^2\gamma_p/4\tilde{L}$. $\hat{\Gamma}_i$ defined as in (3.5.11) and $\mathcal{G}_i$ defined as in (3.5.12) i.e.

$$\mathcal{G}_i(\lambda) \stackrel{\text{def}}{=} -\hat{\Gamma}_i(\lambda)\langle \nabla f(\tilde{x}_{i+1}^\lambda), \tilde{x}_{i+1}^\lambda - \tilde{x}_i\rangle + (f(\tilde{x}_{i+1}^\lambda) - f(\tilde{x}_i)),$$

for x_{i+1}^λ being the result of method (3.5.5) when $\hat{\gamma}_i = \hat{\Gamma}_i(\lambda)$.

Output: $\lambda = \frac{a_{i+1}/\gamma_n}{A_i\hat{\gamma}_i + a_{i+1}/\gamma_n}$ for $\hat{\gamma}_i$ such that $\mathcal{G}_i(\hat{\Gamma}_i^{-1}(\hat{\gamma}_i)) \le \hat{\varepsilon}_i$.

if $\mathcal{G}_i(\hat{\Gamma}_i^{-1}(1/\gamma_n)) \le \hat{\varepsilon}_i$ **then** $\lambda = \hat{\Gamma}_i^{-1}(1/\gamma_n)$
else if $\mathcal{G}_i(\hat{\Gamma}_i^{-1}(\gamma_p)) \le \hat{\varepsilon}_i$ **then** $\lambda = \hat{\Gamma}_i^{-1}(\gamma_p)$
else
  left $\leftarrow \hat{\Gamma}_i^{-1}(1/\gamma_n)$
  right $\leftarrow \hat{\Gamma}_i^{-1}(\gamma_p)$
  $\lambda \leftarrow (\text{left} + \text{right})/2$
  while $\mathcal{G}_i(\lambda) > \hat{\varepsilon}_i$ **do**
    if $\langle \nabla f(\tilde{x}_{i+1}^\lambda), \tilde{x}_{i+1}^\lambda - \tilde{x}_i\rangle < 0$ **then** right $\leftarrow \lambda$
    else left $\leftarrow \lambda$
    end if
    $\lambda \leftarrow (\text{left} + \text{right})/2$
  end while
end if
return λ

perform optimization over $\mathcal{B}_R = \text{Exp}_{x_0}(\bar{B}(0,R))$. Note $x^* \in \mathcal{B}_R$. We assume $F : \mathcal{M}_K \to \mathbb{R}$ is a differentiable function, $\mathcal{B}_R \subset \mathcal{M}_K$, and $\mathcal{M}_K$ has constant sectional curvature K. If K is positive, we restrict $R < \pi/2\sqrt{K}$ so $\mathcal{B}_R$ is contained in an open hemisphere and it is uniquely geodesic. We define a geodesic map $h : \mathcal{M}_K \to M$, where $M \subset \mathbb{R}^n$ and define the function $f : h(\mathcal{M}_K) \to \mathbb{R}$ as $f = F \circ h^{-1}$. We perform constrained optimization over this function f in $\mathcal{X} = h(\mathcal{B}_R)$ in an accelerated way, up to constants and log factors, where the constants appear as an effect of the deformation of the geometry and depend on R and K only.

We note that the assumption of the existence of x^* such that $\nabla F(x^*) = 0$ was assumed for simplicity only and it is not necessary, since $\hat{x}^* = \arg\min_{x\in\mathcal{B}_R}\{F(x)\}$ always satisfies the first inequality in tilted-convexity (3.2.3)—the same proof used for x^* works— so the lower bounds L_i can be defined in the same way as we did. In that case, if we wanted to optimize with a ball constraint, then we would use the Lipschitz constant of F when restricted to B_R, which in the case $\nabla F(\hat{x}^*) = 0$ we have that it is $O(L)$. The Lipschitz constant would be used for the analysis of the binary search and for the computation of $\tilde{L}$, that is the smoothness constant of f.

3.6 Geometric results: Proofs of Lemmas 3.2.1, 3.2.2 and 3.2.3

In this section we prove the lemmas that take into account the deformations of the geometry and the geodesic map h to obtain relationships between F and f. Namely Lemma 3.2.1,

Lemma 3.2.2 and Lemma 3.2.3. First, we recall the characterizations of the geodesic map and some consequences. Then in Section 3.6.2, Section 3.6.3 and Section 3.6.4, we prove the results related to distances angles and gradient deformations, respectively. That is, each of the three parts of Lemma 3.2.1. In Section 3.6.4 we also prove Lemma 3.2.3, which comes naturally after the proof of Lemma 3.2.1.c). In Section 3.6.5 we prove Lemma 3.2.2 and finish with a proof on lower bounds for the condition number of strongly g-convex functions and an intuitive comment on its implications.

Before this, we note that we can assume without loss of generality that the curvature of our manifolds of interest can be taken to be $K \in \{1, -1\}$. One can see that the final rates depend on K through R, L and μ.

Remark 3.6.1. *For a function $F : \mathcal{M}_K \to \mathbb{R}$ where $\mathcal{M}_K$ is a manifold of constant sectional curvature $K \notin \{1, -1, 0\}$, we can apply a rescaling to the Gnomonic or Beltrami-Klein projection to define a function on a manifold of constant sectional curvature $K \in \{1, -1\}$. Namely, we can map $\mathcal{M}_K$ to M via the geodesic map $h : \mathcal{M}_K \to M$, then we can rescale M by multiplying each vector in M by the factor $\sqrt{|K|}$ and then we can apply the inverse geodesic map for the manifold of curvature $K \in \{1, -1\}$. If R is the original bound of the initial distance to an optimum, and F is L-smooth and μ-strongly g-convex (possibly with $\mu = 0$) then the initial distance bound becomes $\sqrt{|K|}R$ and the induced function becomes $L/|K|$-smooth and $\mu/|K|$-strongly g-convex. This is a consequence of the transformation rescaling distances by a factor of $\sqrt{|K|}$, i.e. if $r : \mathcal{M}_K \to \mathcal{M}_{K/|K|}$ is the rescaling function, then $d_K(x, y)\sqrt{|K|} = d_{K/|K|}(r(x), r(y))$, where $d_c(\cdot, \cdot)$ denotes the distance on the manifold of constant sectional curvature c.*

3.6.1 Preliminaries

We recall our characterization of the geodesic map. Given two points $\tilde{x}, \tilde{y} \in \mathcal{X}$, we have that $d(x, y)$, the distance between x and y with the metric of $\mathcal{M}_K$, satisfies

$$\mathrm{C}_K(d(x, y)) = \frac{1 + K\langle \tilde{x}, \tilde{y} \rangle}{\sqrt{1 + K\|\tilde{x}\|^2} \cdot \sqrt{1 + K\|\tilde{y}\|^2}}. \tag{3.6.1}$$

And since the expression is symmetric with respect to rotations, $\mathcal{X} = h(\mathcal{B}_R)$ is a closed ball of radius $\tilde{R}$, with $\mathrm{C}_K(R) = (1 + K\tilde{R}^2)^{-1/2}$. Equivalently,

$$\tilde{R} = \tan(R) \quad \text{if } K = 1; \quad \text{and} \quad \tilde{R} = \tanh(R) \quad \text{if } K = -1. \tag{3.6.2}$$

Similarly, we can write the distances as

$$\begin{aligned} d(x, y) &= \arccos\left(\frac{1 + \langle \tilde{x}, \tilde{y} \rangle}{\sqrt{1 + \|\tilde{x}^2\|}\sqrt{1 + \|\tilde{y}^2\|}}\right) \quad \text{if } K = 1, \\ d(x, y) &= \operatorname{arccosh}\left(\frac{1 - \langle \tilde{x}, \tilde{y} \rangle}{\sqrt{1 - \|\tilde{x}^2\|}\sqrt{1 - \|\tilde{y}^2\|}}\right) \quad \text{if } K = -1, \end{aligned} \tag{3.6.3}$$

Alternatively, we have the following expression for the distance $d(x, y)$ when $K = -1$. Let $\tilde{a}, \tilde{b}$ be the two points of intersection of the ball $B(0, 1) \supseteq M$ with the line joining $\tilde{x}, \tilde{y}$, so the order of the points in the line is $\tilde{a}, \tilde{x}, \tilde{y}, \tilde{b}$. Then

$$d(x, y) = \frac{1}{2} \log \left(\frac{\|\tilde{a} - \tilde{y}\| \|\tilde{x} - \tilde{b}\|}{\|\tilde{a} - \tilde{x}\| \|\tilde{b} - \tilde{y}\|} \right) \text{ if } K = -1. \tag{3.6.4}$$

We will use this expression when working with the hyperbolic space. A simple elementary proof of the equivalence of the expressions in (3.6.3) and (3.6.4) when $K = -1$ is the following. We can assume without loss of generality that we work with the hyperbolic plane, i.e. $n = 2$. By rotational symmetry, we can also assume that $\tilde{x} = (x_1, x_2)$ and $\tilde{y} = (y_1, y_2)$, for $x_1 = y_1$. In fact, it is enough to prove it in the case $x_2 = 0$ because we can split a general segment into two, each with one endpoint at $(x_1, 0)$, and then add their lengths up. So according to (3.6.3) and (3.6.4), respectively, we have

$$\frac{1}{\cosh^2(d(x, y))} = \frac{(1 - x_1^2)(1 - y_1^2 - y_2^2)}{(1 - x_1^2)^2} = \frac{(1 - x_1^2 - y_2^2)}{1 - x_1^2},$$

$$\begin{aligned} d(x, y) &= \frac{1}{2} \log \left(\frac{(\sqrt{1 - y_1^2} + y_2)(\sqrt{1 - x_1^2})}{(\sqrt{1 - x_1^2})(\sqrt{1 - y_1^2} - y_2)} \right) = \frac{1}{2} \log \left(\frac{1 + y_2/\sqrt{1 - x_1^2}}{1 - y_2/\sqrt{1 - x_1^2}} \right) \\ &= \operatorname{arctanh} \left(\frac{y_2}{\sqrt{1 - x_1^2}} \right). \end{aligned}$$

where we have used the equality $\operatorname{arctanh}(t) = \frac{1}{2} \log(\frac{1+t}{1-t})$. Now, using the trigonometric identity $\frac{1}{\cosh^2(t)} = 1 - \tanh^2(t)$, for $t = d(x, y)$, we obtain that the two expressions above are equal. See Theorem 7.4 in (Greenberg, 1993) (p. 268) for more details about the distance formula under this geodesic map.

The spherical case is of a remarkable simplicity. If we have an (n)-sphere of radius 1 centered at $\mathbb{0}_{n+1}$, we can see the transformation of the geodesic map as the projection onto the plane $x_{n+1} = 1$. Given two points $\mathbf{x} = (\tilde{x}, 1)$, $\mathbf{y} = (\tilde{y}, 1)$ then the angle between these two vectors is the distance of the projected points on the sphere so we have $\cos(d(x, y)) = \langle \mathbf{x}, \mathbf{y} \rangle / \|\mathbf{x}\| \|\mathbf{y}\|$ which is equivalent to the corresponding formula in (3.6.3).

3.6.2 Distance deformation

Lemma 3.6.2. *Let $x, y \in \mathcal{B}_R = \operatorname{Exp}_{x_0}(\bar{B}(0, R)) \subseteq \mathcal{M}_K$ be two different points, where $\mathcal{M}_K$ is the hyperbolic space with constant sectional curvature $K = -1$. Then, we have*

$$1 \leq \frac{d(x, y)}{\|\tilde{x} - \tilde{y}\|} \leq \cosh^2(R).$$

Proof We can assume without loss of generality that the dimension is $n = 2$. As in (3.6.2), let $\tilde{R} = \tanh(R)$, so any point $\tilde{x} \in \mathcal{X}$ satisfies $\|\tilde{x}\| \leq \tilde{R}$, or equivalently $d(x, x_0) \leq R$. Recall $\tilde{x}_0 = h(x_0) = 0$. Without loss of generality, we parametrize an arbitrary segment of length ℓ in

$\mathcal{X}$ by two endpoints $\tilde{x}, \tilde{y}$ with coordinates $\tilde{x} = (x_1, x_2)$ and $\tilde{y} = (x_1 - \ell, x_2)$, for $0 \leq x_2 \leq \tilde{R}$, $0 \leq x_1 \leq \sqrt{\tilde{R}^2 - x_2^2}$ and $0 < \ell \leq x_1 + \sqrt{\tilde{R}^2 - x_2^2}$. Let $\mathfrak{d}(x_1, x_2, \ell) \stackrel{\text{def}}{=} \frac{d(x,y)}{\ell}$, the quantity we aim to bound. We will prove the upper bound on $\mathfrak{d}(x_1, x_2, \ell)$ in three steps.

1. If $x_1 = \ell$ then $\mathfrak{d}(\cdot)$ is larger the larger x_1 is. This allows to prove that it is enough to consider points with the extra constraint $\ell \leq x_1$.

2. The partial derivative of $\mathfrak{d}(\cdot)$ with respect to x_1, whenever $\ell \leq x_1$, is non-negative. So we can just look at the points for which $x_1 = \sqrt{\tilde{R}^2 - x_2^2}$.

3. With the constraints above, $\mathfrak{d}(\cdot)$ is larger the smaller ℓ is. So we have

$$\mathfrak{d}(x_1, x_2, \ell) \leq \lim_{\ell \to 0} \mathfrak{d}(\sqrt{\tilde{R}^2 - x_2^2}, x_2, \ell) = \sqrt{1 - x_2^2}/(1 - \tilde{R}^2).$$

   This expression is maximized at $x_2 = 0$ and evaluates to $1/(1 - \tanh^2(R)) = \cosh^2(R)$.

We proceed now to prove the steps above. For the first step, we note

$$\mathfrak{d}(x_1, x_2, x_1) = \frac{1}{2x_1} \log \left(\frac{\sqrt{1-x_2^2}(\sqrt{1-x_2^2} + x_1)}{\sqrt{1-x_2^2}(\sqrt{1-x_2^2} - x_1)} \right) = \frac{1}{2x_1} \log \left(1 + \frac{2x_1}{\sqrt{1-x_2^2} - x_1} \right).$$

We prove that the inverse of this expression is not increasing with respect to x_1. By taking a partial derivative:

$$\begin{aligned} \frac{\partial(1/\mathfrak{d}(x_1, x_2, x_1))}{\partial x_1} &= 2 \frac{\frac{-2x_1\sqrt{1-x_2^2}}{1-x_2^2-x_1^2} + \log(1 + 2x_1/(\sqrt{1-x_2^2} - x_1))}{\log(1 + 2x_1/(\sqrt{1-x_2^2} - x_1))^2} \stackrel{?}{\leq} 0 \\ \iff \frac{2x_1\sqrt{1-x_2^2}}{1-x_2^2-x_1^2} &- \log(1 + (2x_1\sqrt{1-x_2^2} + 2x_1^2)/(1-x_2^2-x_1^2)) \stackrel{?}{\geq} 0. \end{aligned}$$

In order to see that the last inequality is true, note that the expression on the left hand side is 0 when $x_1 = x_2 = 0$. And the partial derivatives of this with respect to x_1 and x_2, respectively, are:

$$\frac{4\sqrt{1-x_2^2}x_1^2}{(1-x_2^2-x_1^2)^2} \text{ and } \frac{4x_2x_1^3}{\sqrt{1-x_2^2}(1-x_2^2-x_1^2)^2}.$$

Both are greater than 0 in the interior of the domain $0 \leq x_2 \leq \tilde{R}$, $0 \leq x_1 \leq \sqrt{\tilde{R}^2 - x_2^2}$ and at least 0 in the border. Now we use this monotonicity to prove that we can consider $\ell \leq x_1$ only. Suppose $\ell > x_1$. The segment ℓ is divided into two parts by the e_2 axis and we can assume without loss of generality that the negative part is no greater than the other, i.e. $x_1 \geq \ell - x_1$. Otherwise, we can perform the computations after a symmetry over the e_2 axis. Let $\tilde{r}$ be the point $(0, x_2)$. We want to see that the segment from $\tilde{x}$ to $\tilde{r}$ gives a greater value of $\mathfrak{d}(\cdot)$:

$$\begin{aligned} \frac{d(x,r)}{x_1} \geq \frac{d(x,y)}{\ell} &\iff d(x,r)(x_1 + (\ell - x_1)) \geq x_1(d(x,r) + d(r,y)) \\ &\iff d(x,r)/x_1 \geq d(r,y)/(\ell - x_1), \end{aligned}$$

and the last inequality holds true by the monotonicity we just proved.

In order to prove the second step, we take the partial derivative of $\mathfrak{d}(x_1, x_2, \ell)$ with respect to x_1. We have

$$\mathfrak{d}(x_1, x_2, \ell) = \frac{1}{2\ell} \log \left(\frac{1 + \ell/(\sqrt{1-x_2^2} - x_1)}{1 - \ell/\sqrt{1-x_2^2} + x_1} \right),$$

$$\frac{\partial \mathfrak{d}(x_1, x_2, \ell)}{\partial x_1} = \frac{\sqrt{1-d^2}(2x_1 - \ell)}{2(1-x_2^2-x_1^2)(1-x_2^2-(x_1-\ell)^2)}.$$

And the derivative is positive in the domain we are considering.

We now prove step 3. We want to show that $\mathfrak{d}(\sqrt{\tilde{R}^2 - x_2^2}, x_2, \ell\cdot)$ decreases with ℓ, within our constraints $\ell \leq x_1 = \sqrt{\tilde{R}^2 - x_2^2}$, $0 \leq x_2 \leq \tilde{R}$. If we split the segment joining $\tilde{x}$ and $\tilde{y}$ in half, with respect to the metric in $\mathcal{X}$, we see that due to the monotonicity proved in step 1, the segment that is farther to the origin is longer in $\mathcal{M}$ than the other one and so $\mathfrak{d}(\cdot)$ is greater for this half of the segment than for the original one. In symbols, let $\tilde{r}$ be the middle point of the segment joining $\tilde{x}$ and $\tilde{y}$. We have by monotonicity that $\mathfrak{d}(x_1, x_2, \ell/2) \geq \mathfrak{d}(x_1, x_2 - \ell/2, \ell/2)$. So $\mathfrak{d}(x_1, x_2, \ell/2) = \frac{d(\tilde{x},\tilde{r})}{\ell/2} \geq \frac{d(\tilde{x},\tilde{r})+d(\tilde{r},\tilde{y})}{\ell} = \mathfrak{d}(x_1, x_2, \ell)$. Thus,

$$\mathfrak{d}(x_1, x_2, \ell) \leq \lim_{\ell \to 0} \mathfrak{d}(\sqrt{\tilde{R}^2 - x_2^2}, x_2, \ell) = \lim_{\ell \to 0} \frac{1}{2\ell} \log \left(\frac{1 + \ell/\left(\sqrt{1-x_2^2} - \sqrt{\tilde{R}^2 - x_2^2}\right)}{1 - \ell/\left(\sqrt{1-x_2^2} + \sqrt{\tilde{R}^2 - x_2^2}\right)} \right)$$

$$\overset{①}{=} \lim_{\ell \to 0} \frac{\sqrt{1-x_2^2}}{1 - \tilde{R}^2 - 2\ell\sqrt{\tilde{R}^2 - x_2^2} + \ell^2} = \frac{\sqrt{1-x_2^2}}{1-\tilde{R}^2}.$$

We used L'Hôpital's rule for ①. We can maximize the last the result of the limit by setting $x_2 = 0$ and obtain that for any two different $\tilde{x}, \tilde{y} \in \mathcal{X}$

$$\frac{d(x,y)}{\|\tilde{x} - \tilde{y}\|} \leq \frac{1}{1-\tilde{R}^2} = \frac{1}{1 - \tanh^2(R)} = \cosh^2(R).$$

The lower bound is similar, assume that $\ell > x_1$ and define $\tilde{r}$ as above. We assume again without loss of generality that $x_1 \geq \ell - x_1$. Then

$$\frac{d(x,r) + d(r,y)}{\ell} \geq \frac{d(x,r)}{\ell - x_1} \iff \frac{d(r,y)}{x_1} \geq \frac{d(x,r)}{\ell - x_1}$$

and the latter is true by the monotonicity proved in step 1. This means that we can also consider $\ell \leq x_1$. But this time, according to step 2, we want x_1 to be the lowest possible, so it is enough to consider $x_1 = \ell$. Using step 1 again, we obtain that the lowest value of $\mathfrak{d}(\cdot)$ can be bounded by the limit $\lim_{\ell \to 0} \mathfrak{d}(\ell, x_2, \ell)$ which using L'Hôpital's rule in ① is

$$\mathfrak{d}(x_1, x_2, \ell) \geq \lim_{\ell \to 0} \mathfrak{d}(\ell, x_2, \ell) = \lim_{\ell \to 0} \frac{1}{2\ell} \log \left(1 + \frac{2\ell}{\sqrt{1-x_2^2} - \ell} \right)$$

$$\overset{①}{=} \lim_{\ell \to 0} \frac{\frac{2(\sqrt{1-x_2^2}-\ell)+2\ell}{(\sqrt{1-x_2^2}-\ell)^2}}{2(1 + 2\ell/(\sqrt{1-x_2^2} - \ell))} = \frac{1}{\sqrt{1-x_2^2}}.$$

The expression is minimized at $x_2 = 0$ and evaluates to 1. ■

The proof of the corresponding lemma for the sphere is analogous, we add it for completeness.

Lemma 3.6.3. *Let $x, y \in B_R = \mathrm{Exp}_{x_0}(\bar{B}(0,R)) \subseteq \mathcal{M}_K$ be two different points, where $\mathcal{M}_K$ is the spherical space with constant sectional curvature $K = 1$, and $R < \pi/2$. Then, we have*

$$\cos^2(R) \leq \frac{d(x,y)}{\|\tilde{x} - \tilde{y}\|} \leq 1.$$

Proof We proceed in a similar way than with the hyperbolic case. We can also work with $d = 2$ only, since $\tilde{x}, \tilde{y}$ and $\tilde{x}_0$ lie on a plane. We parametrize a general pair of points as $\tilde{x} = (x_1, x_2) \in \mathcal{X}$ and $y = (x_1 - \ell, x_2) \in \mathcal{X}$, so $x_1^2 + x_2^2 \leq \tilde{R}^2$, for $\tilde{R} = \tan(R)$ and by definition $\ell = \|\tilde{x} - \tilde{y}\|$.

Let $\mathfrak{d}(x_1, x_2, \ell) \stackrel{\text{def}}{=} d(x,y)/\|\tilde{x} - \tilde{y}\|$. We proceed to prove the result in three steps, similarly to the hyperbolic case.

1. If $x_1 = \ell$ then $\mathfrak{d}(x_1, x_2, \ell)$ decreases whenever x_1 increases. This allows to prove that it is enough to consider points in which $\ell \leq x_1$.
2. $\frac{\partial \mathfrak{d}(\cdot)}{\partial x_1} \leq 0$, whenever $\ell \leq x_1$. So we can consider $x_1 = \sqrt{\tilde{R}^2 - x_2^2}$ only.
3. With the constraints above, $\mathfrak{d}(\cdot)$ increases with ℓ, so in order to lower bound $\mathfrak{d}(\cdot)$ we can consider $\lim_{\ell \to 0} \mathfrak{d}(\sqrt{\tilde{R} - x_2}, x_2, \ell) = \sqrt{1 + x_2^2}/(1 + \tilde{R}^2)$. This is minimized at $x_2 = 0$ and evaluates to $1/(1 + \tilde{R}^2)$.

For the first step, we compute the partial derivative:

$$\frac{\partial \mathfrak{d}(x_1, x_2, x_1)}{\partial x_1} = \frac{x_1\sqrt{1 + x_2^2}/(1 + x_1^2 + x_2^2) - \arccos\left(\sqrt{(1 + x_2^2)/(1 + x_1^2 + x_2^2)}\right)}{x_1^2}. \tag{3.6.5}$$

In order to see that it is non-positive, we compute the partial derivative of the denominator with respect to x_2 and obtain $\frac{2x_1^3 x_2}{\sqrt{1+x_2^2}(1+x_1^2+x_2^2)} \geq 0$, so in order to maximize (3.6.5) we set $x_2 = \sqrt{\tilde{R} - x_1^2}$. In that case, the numerator is

$$\frac{x_1\sqrt{1 + R^2 - x_1^2}}{1 + R^2} - \arccos\left(\sqrt{\frac{1 + R^2 - x_1^2}{1 + R^2}}\right), \tag{3.6.6}$$

and its derivative with respect to x_1 is $-\frac{2x_1^2}{(1+R^2)\sqrt{1+R^2-x_1^2}} \leq 0$. and given that (3.6.6) with $x_1 = 0$ evaluates to 0 we conclude that (3.6.5) is non-positive. Similarly to Lemma 3.6.2, suppose the horizontal segment that joins $\tilde{x}$ and $\tilde{y}$ passes through $\tilde{r} \stackrel{\text{def}}{=} (0, x_2)$. And suppose without loss of generality that $d(x,r) \geq d(r,y)$, i.e. $x_1 \geq \ell - x_1$. Then by the monotonicity we just proved, we have

$$\frac{d(x,r)}{\|\tilde{x} - \tilde{r}\|} = \mathfrak{d}(x_1, x_2, x_1) \leq \mathfrak{d}(\ell - x_1, x_2, \ell - x_1) = \frac{d(r,y)}{\|\tilde{r} - \tilde{y}\|}. \tag{3.6.7}$$

And this implies $\mathfrak{d}(x_1, x_2, x_1) \leq \mathfrak{d}(x_1, x_2, \ell)$. Indeed, that is equivalent to show

$$\frac{d(x,r)}{\|\tilde{x} - \tilde{r}\|} \leq \frac{d(x,y)}{\|\tilde{x} - \tilde{y}\|} = \frac{d(x,r) + d(r+y)}{\|\tilde{x} - \tilde{r}\| + \|\tilde{r} - \tilde{y}\|}.$$

Which is true, since after simplifying we arrive to (3.6.7). So in order to lower bound $\mathfrak{d}(\cdot)$, it is enough to consider $\ell \leq x_1$.

We focus on step 2 now. We have

$$\frac{\partial \mathfrak{d}(x_1, x_2, \ell)}{\partial x_1} = \frac{\sqrt{1+x_2^2}(\ell - 2x_1)}{(1+x_2^2+(\ell-x_1)^2)(1+x_2^2+x_1^2)}.$$

which is non-positive given the restrictions we imposed after step 1. So in order to lower bound $\mathfrak{d}(\cdot)$ we can consider $x_1 = \sqrt{\tilde{R}-x_2^2}$ only.

Finally, in order to complete step 3 we compute

$$\begin{aligned}\frac{\partial \mathfrak{d}(\sqrt{\tilde{R}-x_2^2}, x_2, \ell)}{\partial \ell} &= \frac{\sqrt{1+x_2^2}}{\ell(1+\tilde{R}^2)+\ell^3-2\ell^2\sqrt{\tilde{R}^2-x_2^2}} \\ &\quad - \frac{1}{\ell^2}\arccos\left(\frac{1+\tilde{R}^2-\ell\sqrt{\tilde{R}^2-x_2^2}}{\sqrt{(1+\tilde{R}^2)(1+\tilde{R}^2+\ell^2-2\ell\sqrt{\tilde{R}^2-x_2^2})}}\right)\end{aligned}$$

And in order to prove that this is non-negative, we will prove that the same expression is non-negative, when multiplied by ℓ^2. We compute the partial derivative of the aforementioned expression with respect to ℓ:

$$\frac{\partial}{\partial \ell}\left(\frac{\partial \mathfrak{d}(\sqrt{\tilde{R}-x_2^2}, x_2, \ell)}{\partial \ell}\ell^2\right) = \frac{2\ell\sqrt{1+x_2^2}(\sqrt{\tilde{R}^2-x_2^2}-\ell)}{(1+\tilde{R}^2+\ell^2-2\ell\sqrt{\tilde{R}^2-x_2^2})^2} \geq 0.$$

And $\ell^2(\partial \mathfrak{d}(\sqrt{\tilde{R}-x_2^2}, x_2, \ell)/\partial \ell)$ evaluated at 0 is 0 for all choices of parameters R and x_2 in the domain. So we conclude that $\partial \mathfrak{d}(\sqrt{\tilde{R}-x_2^2}, x_2, \ell)/\partial \ell \geq 0$.

Thus, we can consider the limit when $\ell \to 0$ in order to lower bound $\mathfrak{d}(\cdot)$. In the defined domain, we have

$$\begin{aligned}\lim_{\ell\to 0}\mathfrak{d}(\sqrt{\tilde{R}-x_2}, x_2, \ell) &= \lim_{\ell\to 0}\frac{1}{\ell}\arccos\left(\frac{1+\tilde{R}^2-x\sqrt{\tilde{R}^2-x_2^2}}{\sqrt{1+\tilde{R}^2}\sqrt{1+x_2^2+(\ell-\sqrt{\tilde{R}^2-x_2^2})^2}}\right) \\ &\overset{①}{=} \lim_{\ell\to 0}\frac{\sqrt{1+x_2^2}}{1+\tilde{R}^2+\ell^2-2\ell\sqrt{\tilde{R}^2-x_2^2}} = \frac{\sqrt{1+x_2^2}}{1+\tilde{R}^2}.\end{aligned}$$

We used L'Hôpital's rule for ①. Now, the right hand side of the previous expression is minimized at $x_2 = 0$ so we conclude that we have

$$\cos^2(R) = \frac{1}{1+\tan^2(R)} = \frac{1}{1+\tilde{R}^2} \leq \mathfrak{d}(x_1, x_2, \ell) = \frac{d(p,q)}{\|\tilde{p}-\tilde{q}\|}.$$

The upper bound uses again a similar argument. Assume that $\ell > x_1$ and define $\tilde{r}$ as above.

We assume again without loss of generality that $x_1 \geq \ell - x_1$. Then

$$\frac{d(x,r)+d(r,y)}{\ell} \leq \frac{d(x,r)}{\ell - x_1} \iff \frac{d(r,y)}{x_1} \leq \frac{d(x,r)}{\ell - x_1}$$

and the latter is true by the monotonicity proved in step 1. Consequently we can just consider the points that satisfy $\ell \leq x_1$. By step 2, $\mathfrak{d}(\cdot)$ is maximal whenever x_1 is the lowest possible, so it is enough to consider $x_1 = \ell$. Using step 1 again, we obtain that the greatest value of $\mathfrak{d}(\cdot)$ can be bounded by the limit $\lim_{\ell\to 0} \mathfrak{d}(\ell, x_2, \ell)$ which using L'Hôpital's rule in ① and simplifying is

$$\mathfrak{d}(x_1,x_2,\ell) \leq \lim_{\ell\to 0} \mathfrak{d}(\ell,x_2,\ell) = \lim_{\ell\to 0} \frac{1}{\ell} \arccos\left(\sqrt{\frac{1+x_2^2}{1+\ell^2+x_2^2}}\right) \overset{①}{=} \frac{1}{\sqrt{1+x_2^2}}.$$

The expression is maximized at $x_2 = 0$ and evaluates to 1. ■

3.6.3 Angle deformation

Lemma 3.6.4. *Let $x, y \in B_R = \operatorname{Exp}_{x_0}(\bar{B}(0,R)) \subseteq \mathcal{M}_K$ be two different points and different from x_0, where $\mathcal{M}_K$ is a manifold constant sectional curvature $K \in \{1,-1\}$, and if $K = 1$, then $R < \pi/2$. Let $\tilde{\alpha}$ be the angle $\angle x_0 x y$, formed by the vectors $x_0 - x$ and $y - x$. Let α be the corresponding angle between the vectors $\operatorname{Exp}_x^{-1}(x_0)$ and $\operatorname{Exp}_x^{-1}(y)$. The following holds:*

$$\sin(\alpha) = \sin(\tilde{\alpha})\sqrt{\frac{1+K\|\tilde{x}\|^2}{1+K\|\tilde{x}\|^2\sin^2(\tilde{\alpha})}}, \qquad \cos(\alpha) = \cos(\tilde{\alpha})\sqrt{\frac{1}{1+K\|\tilde{x}\|^2\sin^2(\tilde{\alpha})}}.$$

Proof Note that we can restrict ourselves to $\alpha \in [0,\pi]$ because we have $\widetilde{(-w)} = -\tilde{w}$ (recall our notation about vectors with tilde). This means that the result for the range $\alpha \in [-\pi, 0]$ can be deduced from the result for $-\alpha$.

We start with the case $K = -1$. We can assume without loss of generality that the dimension is $n = 2$, and that the coordinates of $\tilde{x}$ are $(0, x_2)$, for $x_2 \leq \tanh(R)$ that $\tilde{y} = (y_1, y_2)$, for some $y_1 \leq 0$ and $\tilde{\delta} \overset{\text{def}}{=} \angle \tilde{y}\tilde{x}_0\tilde{x} \in [0,\pi/2]$, since we can make the distance $\|\tilde{x} - \tilde{y}\|$ as small as we want. Recall $\tilde{x}_0 = \mathbb{0}_n$. We recall that $d(x,x_0) = \operatorname{arctanh}(\|\tilde{x}\|)$ and we note that $\sinh(\operatorname{arctanh}(t)) = \frac{t}{1-t^2}$, so that $\sinh(d(x,x_0)) = \|\tilde{x}\|/\sqrt{1-\|\tilde{x}\|^2}$, for any $\tilde{x} \in \mathcal{X}$. We will apply the hyperbolic and Euclidean law of sines Fact 3.6.5 in order to compute the value of $\sin(\alpha)$ with respect to $\tilde{\alpha}$. Let $\tilde{a}$ and $\tilde{b}$ be points in the border of $B(0,1)$ such that the segment joining $\tilde{a}$ and $\tilde{b}$ is a chord that contains $\tilde{x}$ and $\tilde{y}$ and $\|\tilde{a}-\tilde{x}\| \leq \|\tilde{b}-\tilde{x}\|$. So $\|\tilde{a}-\tilde{x}\|$ and $\|\tilde{b}-\tilde{x}\|$ are $\sqrt{1-\|\tilde{x}\|^2\sin^2(\tilde{\alpha})} - \|\tilde{x}\|\cos(\tilde{\alpha})$ and $\sqrt{1-\|\tilde{x}\|^2\sin^2(\tilde{\alpha})} + \|\tilde{x}\|\cos(\tilde{\alpha})$, respectively. We have

$$\begin{aligned}
\sin(\alpha) &\overset{①}{=} \frac{\sinh(d(x_0,y))\sin(\tilde{\delta})}{\sinh(d(x,y))} \overset{②}{=} \frac{\|\tilde{x}_0-\tilde{y}\|}{\sqrt{1-\|\tilde{x}_0-\tilde{y}\|^2}} \cdot \frac{\|\tilde{x}-\tilde{y}\|\sin(\tilde{\alpha})}{\|\tilde{x}_0-\tilde{y}\|} \cdot \frac{1}{\sinh(d(x,y))} \\
&\overset{③}{=} \frac{\sin(\tilde{\alpha})}{\sqrt{1-\|\tilde{x}\|^2+\|\tilde{x}-\tilde{y}\|(-2\|\tilde{x}\|\cos(\tilde{\alpha})+\|\tilde{x}-\tilde{y}\|)}} \cdot \frac{\|\tilde{x}-\tilde{y}\|}{\sinh(d(x,y))} \\
&\overset{④}{=} \frac{\sin(\tilde{\alpha})}{\sqrt{1-\|\tilde{x}\|^2}} \lim_{d(x,y)\to 0} \|\tilde{x}-\tilde{y}\| \frac{1}{\sinh(d(x,y))}
\end{aligned}$$

$$\overset{⑤}{=} \frac{\sin(\tilde{\alpha})}{\sqrt{1-\|\tilde{x}\|^2}} \lim_{d(x,y)\to 0} \frac{(e^{2d(x,y)}-1)(\|\tilde{a}-\tilde{x}\|\cdot\|\tilde{b}-\tilde{x}\|)}{e^{2d(x,y)}(\|\tilde{a}-\tilde{x}\|+\|\tilde{b}-\tilde{x}\|)} \cdot \frac{2e^{d(x,y)}}{e^{2d(x,y)}-1}$$
$$= \frac{\sin(\tilde{\alpha})}{\sqrt{1-\|\tilde{x}\|^2}} \cdot \frac{2\|\tilde{a}-\tilde{x}\|\cdot\|\tilde{b}-\tilde{x}\|}{\|\tilde{a}-\tilde{x}\|+\|\tilde{b}-\tilde{x}\|}$$
$$\overset{⑥}{=} \frac{\sin(\tilde{\alpha})}{\sqrt{1-\|\tilde{x}\|^2}} \cdot \frac{2(1-\|\tilde{x}\|^2\sin^2(\tilde{\alpha})-\|\tilde{x}\|^2\cos^2(\tilde{\alpha}))}{2\sqrt{1-\|\tilde{x}\|^2\sin^2(\tilde{\alpha})}} = \sin(\tilde{\alpha})\sqrt{\frac{1-\|\tilde{x}\|^2}{1-\|\tilde{x}\|^2\sin^2(\tilde{\alpha})}}.$$

In ① we used the hyperbolic sine theorem. In ② we used the expression above regarding segments that pass through the origin, and the Euclidean sine theorem. In ③, we simplify and use that the coordinates of $\tilde{y}$ are $(-\sin(\tilde{\alpha})\|\tilde{x}-\tilde{y}\|, \|\tilde{x}\|-\cos(\tilde{\alpha})\|\tilde{x}-\tilde{y}\|)$. Then, in ④, since $\sin(\alpha)$ does not depend on $\|\tilde{x}-\tilde{y}\|$, we can take the limit when $d(x,y)\to 0$, by which we mean we take the limit $\tilde{y}\to\tilde{x}$ by keeping the angle $\tilde{\alpha}$ constant. Since a posteriori the limit of each fraction exists, we compute them one at a time. ⑤ uses (3.6.4) and the definition of $\sinh(d(x,y))$ for the last factor. The equality for the other factor can be checked with (3.6.4). In ⑥ we substitute $\|\tilde{a}-\tilde{x}\|$ and $\|\tilde{b}-\tilde{x}\|$ by their values.

The spherical case is similar to the hyperbolic case. We also assume without loss of generality that the dimension is $n=2$. Define $\tilde{y}$ as a point such that $\angle\tilde{x}_0\tilde{x}\tilde{y}=\tilde{\alpha}$. We can assume without loss of generality that the coordinates of $\tilde{x}$ are $(0,x_2)$, that $\tilde{y}=(y_1,y_2)$, for $y_1\leq 0$, and $\tilde{\delta}\overset{\text{def}}{=}\angle\tilde{y}\tilde{x}_0\tilde{x}\in[0,\pi/2]$, since we can make the distance $\|\tilde{x}-\tilde{y}\|$ as small as we want. We recall that by (3.6.2) we have $d(x_0,x)=\arctan(\|\tilde{x}_0-\tilde{x}\|)$ and we note that $\sin(\arctan(t))=\frac{t}{1+t^2}$, so that $\sin(d(x_0,x))=\|\tilde{x}_0-\tilde{x}\|/\sqrt{1+\|\tilde{x}_0-\tilde{x}\|^2}$, for any $\tilde{x}\in\mathcal{X}$. We will apply the spherical and Euclidean law of sines Fact 3.6.5 in order to compute the value of $\sin(\alpha)$ with respect to $\tilde{\alpha}$. We have

$$\sin(\alpha) \overset{①}{=} \frac{\sin(d(x_0,y))\sin(\tilde{\delta})}{\sin(d(x,y))} \overset{②}{=} \frac{\|\tilde{x}_0-\tilde{y}\|}{\sqrt{1+\|\tilde{x}_0-\tilde{y}\|^2}} \cdot \frac{\|\tilde{x}-\tilde{y}\|\sin(\tilde{\alpha})}{\|\tilde{x}_0-\tilde{y}\|} \frac{1}{\sin(d(x,y))}$$
$$\overset{③}{=} \frac{\sin(\tilde{\alpha})\|\tilde{x}-\tilde{y}\|}{\sqrt{1+\|\tilde{x}_0-\tilde{y}\|^2}\sqrt{1-\frac{(1-\|x\|\cos(\tilde{\alpha})\|\tilde{x}-\tilde{y}\|+\|\tilde{x}\|^2)^2}{(1+\|\tilde{x}\|^2)(1+\|\tilde{x}_0-\tilde{y}\|^2)}}}$$
$$\overset{④}{=} \frac{\sin(\tilde{\alpha})\|\tilde{x}-\tilde{y}\|}{\sqrt{\|\tilde{x}-\tilde{y}\|^2(1+\|\tilde{x}\|^2-\|\tilde{x}\|^2\cos^2(\tilde{\alpha}))/(1+\|\tilde{x}\|^2)}} \overset{⑤}{=} \sin(\tilde{\alpha})\sqrt{\frac{1+\|\tilde{x}\|^2}{1+\|\tilde{x}\|^2\sin^2(\tilde{\alpha})}}.$$

In ① we used the spherical sine theorem. In ② we used the expression above regarding segments that pass through the origin, and the Euclidean sine theorem. In ③, we use the fact that the coordinates of $\tilde{y}$ are $(-\sin(\tilde{\alpha})\|\tilde{x}-\tilde{y}\|, \|\tilde{x}\|^2-\cos(\tilde{\alpha})\|\tilde{x}-\tilde{y}\|)$, use the distance formula (3.6.3) and the trigonometric equality $\sin(\arccos(x))=\sqrt{1-x^2}$. Then, in ④ and ⑤, we multiply and simplify.

Finally, in both cases, the cosine formula is derived from the identity $\sin^2(\alpha)+\cos^2(\alpha)=1$ after noticing that the sign of $\cos(\alpha)$ and the sign of $\cos(\tilde{\alpha})$ are the same. The latter fact can be seen to hold true by noticing that α is monotonous with respect to $\tilde{\alpha}$ and the fact that $\tilde{\alpha}=\pi/2$ implies $\sin(\alpha)=0$. ■

Fact 3.6.5 (Constant Curvature non-Euclidean Laws of Sines and of Cosines). *Let $K \neq 0$ and let $\mathrm{S}_K(\cdot)$ and $\mathrm{C}_K(\cdot)$ denote the special sine and cosine, respectively, defined as $\mathrm{S}_K(t) = \sin(\sqrt{K}t)$ and $\mathrm{C}_K(t) = \cos(\sqrt{K}t)$ if $K > 0$, and as $\mathrm{S}_K(t) = \sinh(\sqrt{-K}t)$ and $\mathrm{C}_K(t) = \cosh(\sqrt{-K}t)$ if $K < 0$. Let a, b, c be the lengths of the sides of a geodesic triangle defined on a manifold of constant sectional curvature K. Let α, β, γ be the angles of the geodesic triangle, that are opposite to the sides a, b, c. The following holds:*

- *Law of sines:*
$$\frac{\sin(\alpha)}{\mathrm{S}_K(a)} = \frac{\sin(\beta)}{\mathrm{S}_K(b)} = \frac{\sin(\gamma)}{\mathrm{S}_K(c)}.$$
- *Law of cosines:*
$$\mathrm{C}_K(a) = \mathrm{C}_K(b)\,\mathrm{C}_K(c) + \cos(\alpha)\,\mathrm{S}_K(b)\,\mathrm{S}_K(c).$$

We refer to (Greenberg, 1993) for a proof of these classical theorems.

3.6.4 Gradient deformation and smoothness of f

Lemma 3.6.4, with $\tilde{\alpha} = \pi/2$, shows that $e_1 \perp e_j$, for $j \neq 1$. The rotational symmetry implies $e_i \perp e_j$ for $i \neq j$ and $i, j > 1$. As in Lemma 3.2.1, let $x \in \mathcal{B}_R$ be a point and assume without loss of generality that $\tilde{x} \in \mathrm{span}\{\tilde{e}_1\}$ and $\nabla f(\tilde{x}) \in \mathrm{span}\{\tilde{e}_1, \tilde{e}_2\}$. It can be assumed without loss of generality because of the symmetries. So we can assume the dimension is $n = 2$. Using Lemma 3.2.1 we obtain that $\tilde{\alpha} = 0$ implies $\alpha = 0$. Also $\tilde{\alpha} = \pi/2$ implies $\alpha = \pi/2$, so the adjoint of the differential of h^{-1} at x, $(\mathrm{d}h^{-1})^*_x$ diagonalizes and has e_1 and e_2 as eigenvectors. We only need to compute the eigenvalues. The computation of the first one uses that the geodesic passing from x_0 and x can be parametrized as $h^{-1}(\tilde{x}_0 + \arctan(\tilde{\lambda}\tilde{e}_1))$ if $K = 1$ and $h^{-1}(\tilde{x}_0 + \mathrm{arctanh}(\tilde{\lambda}\tilde{e}_1))$ if $K = -1$, by (3.6.1). The derivative of $\arctan(\cdot)$ or $\mathrm{arctanh}(\cdot)$ reveals that the first eigenvector, the one corresponding to e_1, is $1/(1 + K\|\tilde{x}^2\|)$, i.e. $\nabla f(\tilde{x})_1 = \nabla F(x)_1/(1 + K\|\tilde{x}^2\|)$. For the second one, let $x = (x_1, 0)$ and $y = (y_1, y_2)$, with $y_1 = x_1$ the second eigenvector results from the computation, for $K = -1$:

$$\lim_{y_2 \to 0} \frac{d(x,y)}{y_2} = \lim_{y_2 \to 0} \frac{\log(1 + \frac{2y_2}{\sqrt{1-x_1^2}-y_2})}{2y_2} \overset{①}{=} \lim_{y_2 \to 0} \frac{\frac{2}{\sqrt{1-x_1^2}-y_2} + \frac{2y_2}{(\sqrt{1-x_1^2}-y_2)^2}}{2 + \frac{4y_2}{\sqrt{1-x_1^2}-y_2}} = \frac{1}{\sqrt{1-x_1^2}},$$

and for $K = 1$:

$$\lim_{y_2 \to 0} \frac{d(x,y)}{y_2} = \lim_{y_2 \to 0} \frac{\arccos(\frac{\sqrt{1+x_1^2}}{\sqrt{1+x_1^2+y_2^2}})}{y_2} \overset{②}{=} \lim_{y_2 \to 0} \frac{\sqrt{1+x_1^2}}{1 + x_1^2 + y_2^2} = \frac{1}{\sqrt{1+x_1^2}}.$$

So, since $x_1 = \|\tilde{x}\|$, we have $\nabla f(\tilde{x})_2 = \nabla F(x)_2/\sqrt{1 + K\|\tilde{x}\|^2}$ for $K \in \{1, -1\}$. We used L'Hôpital's rule in ① and ②.

Also note that if $v \in T_x\mathcal{M}_K$ is a vector normal to $\nabla F(x)$, then $\tilde{v}$ is normal to $\nabla f(x)$. It is easy to see this geometrically: Indeed, no matter how h changes the geometry, since it is a geodesic map, a geodesic in the direction of first-order constant increase of F is mapped via h

to a geodesic in the direction of first-order constant increase of f. And the respective gradients must be perpendicular to all these directions. Alternatively, this can be seen algebraically. Suppose first $n = 2$, then v is proportional to $(\nabla F(x)_2, -\nabla F(x)_1) = (\sqrt{1 + K\|\tilde{x}\|^2}\nabla f(\tilde{x})_2, -(1+ K\|\tilde{x}\|^2)\nabla f(\tilde{x}_1))$. And a vector $\tilde{v}'$ normal to $\nabla f(x)$ must be proportional to $(-\nabla f(x)_2, \nabla f(x)_1)$. Let α be the angle formed by v and $-e_1$, $\tilde{\alpha}$ the corresponding angle formed between $\tilde{v}$ and $-\tilde{e}_1$, and $\tilde{\alpha}'$ the angle formed by $\tilde{v}'$ and $-\tilde{e}_1$. Then we have, using the expression for the vectors proportional to v and $\tilde{v}'$:

$$\sin(\alpha) = \frac{-f(x)_2}{\sqrt{\nabla f(x)_2^2 + (1 + \|x\|^2)\nabla f(x)_1^2}} \text{ and } \sin(\tilde{\alpha}') = \frac{-f(x)_2}{\sqrt{\nabla f(x)_2^2 + \nabla f(x)_1^2}}$$

and using one equation on the other yields $\sin(\alpha) = \sin(\tilde{\alpha}')\sqrt{(1 + K\|\tilde{x}^2\|)/(1 + K\|\tilde{x}^2\| \sin^2(\tilde{\alpha}'))}$, which after applying Lemma 3.6.4 we obtain $\sin(\tilde{\alpha}') = \sin(\tilde{\alpha})$ from which we conclude that $\tilde{\alpha}' = \tilde{\alpha}$ given that the angles are in the same quadrant. So $\tilde{v} \perp \nabla f(x)$. In order to prove this for $n \geq 3$ one can apply the reduction (3.6.14) to the case $n = 2$ that we obtain in the next section.

Combining the results obtained so far in Section 3.6, we can prove Lemma 3.2.1. We continue by proving Lemma 3.2.3, which will generalize the computations we just performed, in order to analyze the Hessian of f and provide smoothness. Then, in the next section, we combine the results in Lemma 3.2.1 to prove Lemma 3.2.2.

Proof of Lemma 3.2.1. The lemma follows from Lemmas 3.6.2, 3.6.3, 3.6.4 and the previous reasoning in this Section 3.6.4. ■

Proof of Lemma 3.2.3. Recall $F : \mathcal{M}_K \to \mathbb{R}$ is a function defined on a manifold of constant sectional curvature with a point $x^* \in \mathcal{B}_R$ such that $\nabla F(x^*) = 0$ and we call R an upper bound on $d(x_0, x^*)$, for an initial point x_0.

We will compute the Hessian of $f = F \circ h^{-1}$ and we will bound its spectral norm for any point $\tilde{x} \in \mathcal{X}$. We can assume without loss of generality that $n = 2$ and $\tilde{x} = (\tilde{\ell}, 0)$, for $\tilde{\ell} > 0$ (the case $\tilde{\ell} = 0$ is trivial), since there is a rotational symmetry with e_1 as axis. This means that by rotating we could align the top eigenvector of the Hessian at a point so that it is in $\text{span}\{e_1, e_2\}$. Let $\tilde{y} = (y_1, y_2) \in \mathcal{X}$ be another point, with $y_1 = \tilde{\ell}$. We can also assume that $y_2 > 0$ without loss of generality, because of our symmetry. Our approach will be the following. We know by Lemma 3.2.1.b) and by the beginning of this Section 3.6.4 that the adjoint of the differential of h^{-1} at y, $(\mathrm{d}h^{-1})_y^*$ has $\text{Exp}_y^{-1}(x_0)$ and a normal vector to it as eigenvectors. Their corresponding eigenvalues are $1/(1 + K\|\tilde{y}\|^2)$ and $1/\sqrt{1 + K\|\tilde{y}\|^2}$, respectively. Consider the basis $\{e_1, e_2\}$ of $T_x\mathcal{M}_K$ as defined at the beginning of this section, i.e. where e_1 is a unit vector proportional to $-\text{Exp}_x^{-1}(x_0)$ and e_2 is the normal vector to e_1 that makes the basis orthonormal. Consider this basis being transported to y using parallel transport and denote the result $\{v_y, v_y^\perp\}$. Assume we have the gradient $\nabla F(y)$ written in this basis. Then we can compute the gradient of f at y by applying $(\mathrm{d}h^{-1})_y^*$ to $\nabla F(y)$. In order to do that, we compose the change of basis from $\{v_y, v_y^\perp\}$ to the basis of eigenvectors of $(\mathrm{d}h^{-1})_y^*$, then we apply a diagonal transformation given by the eigenvalues and finally we change the basis to $\{\tilde{e}_1, \tilde{e}_2\}$. Once this is done, we can differentiate

with respect to y_2 in order to compute a column of the Hessian. Let $\tilde{\alpha}$ be the angle formed by the vectors $\tilde{y}$ and $\tilde{x}$. Note that $\tilde{\alpha} = \arctan(y_2/y_1)$. Let $\tilde{\gamma}$ be the angle formed by the vectors $(\tilde{y} - \tilde{x})$ and $-\tilde{y}$. That is, the angle $\tilde{\gamma} = \pi - \angle\tilde{x}\tilde{y}\tilde{x}_0$. Since $v_y^{\perp}$ is the parallel transport of $e_2^{\perp}$, the angle between $v_y^{\perp}$ and the vector $\operatorname{Exp}_y^{-1}(x_0)$ is γ. Note we use the same convention as before for the angles, i.e. γ is the corresponding angle to $\tilde{\gamma}$, meaning that if γ is the angle between two intersecting geodesics in $\mathcal{B}_R$, then $\tilde{\gamma}$ is the angle between the respective corresponding geodesics in $\mathcal{X}$. Note the first change of basis is a rotation and that the angle of rotation is $\gamma - \pi/2$. The last change of basis is a rotation with angle equal to the angle formed by a vector $\tilde{v}$ normal to $-\tilde{y}$ ($\tilde{v}$ is the one such that $-\tilde{y} \times \tilde{v} > 0$) and the vector $\tilde{e}_2$. This vector is equal to $\tilde{\alpha}$. So we have

$$\nabla f(y) = \begin{pmatrix} \cos(\tilde{\alpha}) & -\sin(\tilde{\alpha}) \\ \sin(\tilde{\alpha}) & \cos(\tilde{\alpha}) \end{pmatrix} \begin{pmatrix} \frac{1}{1+K(y_1^2+y_2^2)} & 0 \\ 0 & \frac{1}{\sqrt{1+K(y_1^2+y_2^2)}} \end{pmatrix} \begin{pmatrix} \sin(\gamma) & -\cos(\gamma) \\ \cos(\gamma) & \sin(\gamma) \end{pmatrix} \nabla F(y) \tag{3.6.8}$$

We want to take the derivative of this expression with respect to y_2 and we want to evaluate it at $y_2 = 0$. Let the matrices above be A, B and C so that $\nabla f(y) = ABC\nabla F(y)$. Using Lemma 3.2.1.c) we have

$$\begin{aligned} \sin(\gamma) &= \sin(\tilde{\gamma})\sqrt{\frac{1+K(y_1^2+y_2^2)}{1+K(y_1^2+y_2^2)\sin^2(\tilde{\gamma})}} \overset{①}{=} \cos(\tilde{\alpha})\sqrt{\frac{1+K(y_1^2+y_2^2)}{1+K(y_1^2+y_2^2)\cos^2(\tilde{\alpha})}}, \\ \cos(\gamma) &= -\sin(\tilde{\alpha})\sqrt{\frac{1}{1+K(y_1^2+y_2^2)\cos^2(\tilde{\alpha})}}, \end{aligned} \tag{3.6.9}$$

where ① follows from $\sin(\tilde{\gamma}) = \sin(\tilde{\alpha} + \pi/2) = \cos(\tilde{\alpha})$. Now we compute some quantities

$$A|_{y_2=0} = I,\ B|_{y_2=0} = \begin{pmatrix} \frac{1}{1+Ky_1^2} & 0 \\ 0 & \frac{1}{\sqrt{1+Ky_1^2}} \end{pmatrix},\ C|_{y_2=0} = I,$$

$$\left.\frac{\partial A}{\partial y_2}\right|_{y_2=0} = \left.\frac{\partial \tilde{\alpha}}{\partial y_2}\right|_{y_2=0} \cdot \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \overset{①}{=} \begin{pmatrix} 0 & \frac{-1}{y_1} \\ \frac{1}{y_1} & 0 \end{pmatrix},$$

$$\left.\frac{\partial B}{\partial y_2}\right|_{y_2=0} = \left.\begin{pmatrix} \frac{2Ky_2}{(1+K(y_1^2+y_2^2))^2} & 0 \\ 0 & \frac{2Ky_2}{2(1+K(y_1^2+y_2^2))^{3/2}} \end{pmatrix}\right|_{y_2=0} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$

$$\left.\frac{\partial C}{\partial y_2}\right|_{y_2=0} \overset{②}{=} \begin{pmatrix} 0 & \frac{1}{y_1\sqrt{1+Ky_1^2}} \\ \frac{-1}{y_1\sqrt{1+Ky_1^2}} & 0 \end{pmatrix}.$$

Equalities ① and ② follow after using (3.6.9), $\tilde{\alpha} = \arctan(\frac{y_2}{y_1})$ and taking derivatives. Now we

differentiate (3.6.8) with respect to y_2 and evaluate to $y_2 = 0$ using the chain rule. The result is

$$\begin{aligned}\begin{pmatrix}\nabla^2 f(\tilde{x})_{12}\\ \nabla^2 f(\tilde{x})_{22}\end{pmatrix} &= \left(\frac{\partial A}{\partial y_2}BC\nabla F(x) + A\frac{\partial B}{\partial y_2}C\nabla F(x) + AB\frac{\partial C}{\partial y_2}\nabla F(x) + ABC\frac{\partial \nabla F(x)}{\partial y_2}\right)\Big|_{y_2=0}\\ &= \begin{pmatrix}\frac{-\nabla f(\tilde{x})_2}{y_1\sqrt{1+Ky_1^2}}\\ \frac{\nabla f(\tilde{x})_1}{y_1(1+Ky_1^2)}\end{pmatrix} + \begin{pmatrix}0\\0\end{pmatrix} + \begin{pmatrix}\frac{\nabla f(\tilde{x})_2}{y_1(1+Ky_1^2)^{3/2}}\\ \frac{-\nabla f(\tilde{x})_1}{y_1(1+Ky_1^2)}\end{pmatrix} + \begin{pmatrix}\frac{\nabla^2 F(x)_{12}}{(1+Ky_1^2)^{3/2}}\\ \frac{\nabla^2 F(x)_{22}}{1+Ky_1^2}\end{pmatrix}\end{aligned}$$

Computing the other column of the Hessian is easier. We can just consider (3.6.8) with $\tilde{\alpha} = 0$, $\gamma = \pi/2$ and vary y_1. Taking derivatives with respect to y_1 gives

$$\begin{pmatrix}\nabla^2 f(\tilde{x})_{11}\\ \nabla^2 f(\tilde{x})_{21}\end{pmatrix} = \begin{pmatrix}\frac{-2Ky_1\nabla f(\tilde{x})_1}{(1+Ky_1^2)^2}\\ \frac{-Ky_1\nabla f(\tilde{x})_2}{(1+Ky_1^2)^{3/2}}\end{pmatrix} + \begin{pmatrix}\frac{\nabla^2 F(x)_{11}}{(1+Ky_1^2)^2}\\ \frac{\nabla^2 F(x)_{21}}{(1+Ky_1^2)^{3/2}}\end{pmatrix}.$$

Note in the computations of both of the columns of the Hessian we have used

$$\frac{\partial \nabla F(y)_i}{\partial y_1} = \nabla F(x)_{i1}\cdot\frac{1}{1+Ky_1^2} \quad \text{and} \quad \frac{\partial \nabla F(y)_i}{\partial y_2}\Big|_{y_2=0} = \nabla F(x)_{i2}\cdot\frac{1}{\sqrt{1+Ky_1^2}},$$

for $i = 1, 2$. The eigenvalues of the adjoint of the differential of h^{-1} appear as a factor because we are differentiating with respect to the geodesic in $\mathcal{X}$ which moves at a different speed than the corresponding geodesic in $\mathcal{B}_R$. Note as well, as a sanity check, that the cross derivatives are equal, since

$$-\frac{1}{y_1\sqrt{1+Ky_1^2}} + \frac{1}{y_1(1+Ky_1^2)^{3/2}} = \frac{1}{y_1\sqrt{1+Ky_1^2}}\left(-1+\frac{1}{1+Ky_1^2}\right) = \frac{-Ky_1}{(1+Ky_1^2)^{3/2}}.$$

Finally, we bound the new smoothness constant $\tilde{L}$ by bounding the spectral norm of this Hessian. First note that using $y_1 = \tilde{\ell}$ we have that $\frac{1}{\sqrt{1+K\tilde{\ell}^2}} = \mathrm{C}_K(\ell)$ and then for $K = -1$ it is $\tilde{\ell} = \tanh(\ell)$ and for $K = 1$ it is $\tilde{\ell} = \tan(\ell)$, where $\ell = d(x, x_0) < R$. We have that since there is a point $x^* \in \mathcal{B}_R$ such that $\nabla F(x^*) = 0$ and F is L-smooth, then it is $\|\nabla F(x)\| \leq 2LR$. Similarly, by L-smoothness $|\nabla^2 F(x)_{ij}| \leq L$. We are now ready to prove $\tilde{L}$-smoothness of f.

$$\begin{aligned}\tilde{L}^2 &\leq \max_{\tilde{x}\in\mathcal{X}}\|\nabla^2 f(\tilde{x})\|_2^2\\ &\leq \max_{\tilde{x}\in\mathcal{X}}\|\nabla^2 f(\tilde{x})\|_F^2 = \max_{\tilde{x}\in\mathcal{X}}\{\nabla^2 f(\tilde{x})_{11} + 2\nabla^2 f(\tilde{x})_{12} + \nabla^2 f(\tilde{x})_{22}\}\\ &\leq L^2([\mathrm{C}_K^4(R) + 4R\,\mathrm{S}_K(R)\,\mathrm{C}_K^3(R)]^2 + 2[\mathrm{C}_K^3(R) + 2R\,\mathrm{S}_K(R)\,\mathrm{C}_K^2(R)]^2 + \mathrm{C}_K^4(R))\end{aligned}$$

and this can be bounded by $44L^2\max\{1, R^2\}$ if $K = 1$ and $44L^2\max\{1, R^2\}\,\mathrm{C}_K^8(R)$ if $K = -1$. In any case, it is $O(L^2)$. We note that for tilted convex functions we have that gradient Lipschitzness, smoothness and bounded Hessian are equivalent properties. Indeed, this is a classical result for convex functions and the proof of the implication that does not hold for general differentiable functions (showing that smoothness implies gradient Lipschitzness) only requires that a point with zero gradient is a global minimizer, cf. (Zhou, 2018) for instance. This property is true for tilted convex functions. ■

3.6.5 Proof of Lemma 3.2.2

Proof Assume for the moment that the dimension is $n = 2$. We can assume without loss of generality that $\tilde{x} = (\tilde{\ell}, 0)$. We are given two vectors, that are the gradients $\nabla F(x)$, $\nabla f(\tilde{x})$ and a vector $w \in T_x\mathcal{M}_K$. Let $\tilde{\delta}$ be the angle between $\tilde{w}$ and $-\tilde{x}$. Let δ be the corresponding angle, i.e. the angle between w and $u \stackrel{\text{def}}{=} \text{Exp}_x^{-1}(x_0)$. Let α be the angle in between $\nabla F(x)$ and u. Let $\tilde{\beta}$ be the angle in between $\nabla f(\tilde{x})$ and $-x$. $\tilde{\alpha}$ and β are defined similarly. We claim

$$\frac{\langle \frac{\nabla F(x)}{\|\nabla F(x)\|}, \frac{w}{\|w\|} \rangle}{\langle \frac{\nabla f(\tilde{x})}{\|\nabla f(\tilde{x})\|}, \frac{\tilde{w}}{\|\tilde{w}\|} \rangle} = \sqrt{\frac{1 + K\tilde{\ell}^2}{(1 + K\tilde{\ell}^2 \sin^2(\tilde{\delta}))(1 + K\tilde{\ell}^2 \cos^2(\tilde{\beta}))}}. \tag{3.6.10}$$

Let's see how to arrive to this expression. By Lemma 3.2.1.c) we have

$$\tan(\alpha) = \frac{\tan(\tilde{\beta})}{\sqrt{1 + K\tilde{\ell}^2}}. \tag{3.6.11}$$

From this relationship we deduce

$$\cos(\alpha) = \cos(\tilde{\beta}) \sqrt{\frac{1 + K\tilde{\ell}^2}{1 + K\tilde{\ell}^2 \cos^2(\tilde{\beta})}}, \tag{3.6.12}$$

that comes from squaring (3.6.11), reorganizing terms and noting that $\text{sign}(\cos(\alpha)) = \text{sign}(\cos(\tilde{\beta}))$ which is implied by Lemma 3.2.1.c). We are now ready to prove the claim (3.6.10) (for $n = 2$). We have

$$\begin{aligned}
\frac{\langle \frac{\nabla F(x)}{\|\nabla F(x)\|}, \frac{w}{\|w\|} \rangle}{\langle \frac{\nabla f(\tilde{x})}{\|\nabla f(\tilde{x})\|}, \frac{\tilde{w}}{\|\tilde{w}\|} \rangle} &= \frac{\cos(\alpha - \delta)}{\cos(\tilde{\beta} - \tilde{\delta})} \\
&\overset{②}{=} \frac{\cos(\delta) + \tan(\alpha)\sin(\delta)}{\cos(\tilde{\beta})\cos(\tilde{\delta}) + \sin(\tilde{\beta})\sin(\tilde{\delta})} \cos(\alpha) \\
&\overset{③}{=} \frac{\frac{\cos(\tilde{\delta})}{\sqrt{1+K\tilde{\ell}^2 \sin^2(\tilde{\delta})}} + \frac{\tan(\tilde{\beta})}{\sqrt{1+K\tilde{\ell}^2}} \frac{\sin(\tilde{\delta})\sqrt{1+K\tilde{\ell}^2}}{\sqrt{1+K\tilde{\ell}^2 \sin^2(\tilde{\delta})}}}{\cos(\tilde{\beta})\cos(\tilde{\delta}) + \sin(\tilde{\beta})\sin(\tilde{\delta})} \cos(\tilde{\beta}) \sqrt{\frac{1 + K\tilde{\ell}^2}{1 + K\tilde{\ell}^2 \cos^2(\tilde{\beta})}} \\
&\overset{④}{=} \sqrt{\frac{1 + K\tilde{\ell}^2}{(1 + K\tilde{\ell}^2 \sin^2(\tilde{\delta}))(1 + K\tilde{\ell}^2 \cos^2(\tilde{\beta}))}}.
\end{aligned}$$

Equality ① follows by the definition of $\alpha, \delta, \tilde{\delta}$, and $\tilde{\beta}$. In ②, we used trigonometric identities. In ③ we used Lemma 3.2.1.c), (3.6.11) and (3.6.12). By reordering the expression, the denominator cancels out with a factor of the numerator in ④.

In order to work with arbitrary dimension, we note it is enough to prove it for $n = 3$, since in order to bound $\langle \frac{\nabla F(x)}{\|\nabla F(x)\|}, \frac{v}{\|v\|} \rangle / \langle \frac{\nabla f(\tilde{x})}{\|\nabla f(\tilde{x})\|}, \frac{\tilde{v}}{\|\tilde{v}\|} \rangle$, it is enough to consider the following submanifold

$$\mathcal{M}'_K \stackrel{\text{def}}{=} \text{Exp}_x(\text{span}\{v, \text{Exp}_x^{-1}(x_0), \nabla F(x)\}).$$

for an arbitrary vector $v \in T_x\mathcal{M}_K$ and a point x defined as above. The case $n = 3$ can be further reduced to the case $n = 2$ in the following way. Suppose $\mathcal{M}'_K$ is a three dimensional manifold

(if it is one or two dimensional there is nothing to do). Define the following orthonormal basis of $T_x\mathcal{M}_K$, $\{e_1, e_2, e_3\}$ where $e_1 = -\operatorname{Exp}_x^{-1}(x_0)/\|\operatorname{Exp}_x^{-1}(x_0)\|$, e_2 is a unit vector, normal to e_1 such that $e_2 \in \operatorname{span}\{e_1, \nabla F(x)\}$. And e_3 is a vector that completes the orthonormal basis. In this basis, let v be parametrized by $\|v\|(\sin(\delta), \cos(\nu)\cos(\delta), \sin(\nu)\cos(\delta))$, where δ can be thought as the angle between the vector v and its projection onto the plane $\operatorname{span}\{e_2, e_3\}$ and ν can be thought as the angle between this projection and its projection onto e_2. Similarly we parametrize $\tilde{v}$ by $\|\tilde{v}\|(\sin(\tilde{\delta}), \cos(\tilde{\nu})\cos(\tilde{\delta}), \sin(\tilde{\nu})\cos(\tilde{\delta}))$, where the base used is the analogous base to the previous one, i.e. the vectors $\{\tilde{e}_1, \tilde{e}_2, \tilde{e}_3\}$. Taking into account that $e_2 \perp e_1$, $e_3 \perp e_1$, $\tilde{e}_2 \perp \tilde{e}_1$, $\tilde{e}_3 \perp \tilde{e}_1$, and the fact that e_1 is parallel to $-\operatorname{Exp}_x(x_0)$, by the radial symmetry of the geodesic map we have that $\nu = \tilde{\nu}$. Also, by looking at the submanifold $\operatorname{Exp}_x(\operatorname{span}\{e_1, v\})$ and using Lemma 3.2.1.c) we have $\sin(\delta) = \sin(\tilde{\delta})\sqrt{\frac{1+K\tilde{\ell}^2}{1+K\tilde{\ell}^2\sin(\tilde{\delta})}}$. If we want to compare $\langle \nabla F(x), v\rangle$ with $\langle \nabla f(\tilde{x}), \tilde{v}\rangle$ we should be able to just zero out the third components of v and $\tilde{v}$ and work in $n = 2$. But in order to completely obtain a reduction to the two-dimensional case we studied a few paragraphs above, we would need to prove that if we call $w \overset{\text{def}}{=} (\sin(\delta), \cos(\nu)\cos(\delta), 0)$ the vector v with the third component made 0, then $\tilde{w}$ is in the same direction of the vector $\tilde{v}$, when the third component is made 0. The norm of these two vectors will not be the same, however. Let $\tilde{w}' = (\sin(\tilde{\delta}), \cos(\nu)\cos(\tilde{\delta}), 0)$ be the vector $\tilde{v}$ when the third component is made 0. Then

$$\|w\| = \|v\|\sqrt{\sin^2(\delta) + \cos^2(\delta)\cos^2(\nu)} \text{ and } \|\tilde{w}'\| = \|\tilde{v}\|\sqrt{\sin^2(\tilde{\delta}) + \cos^2(\tilde{\delta})\cos^2(\nu)}. \tag{3.6.13}$$

But indeed, we claim

$$\tilde{w} \text{ and } \tilde{w}' \text{ have the same direction.} \tag{3.6.14}$$

This is easy to see geometrically: since we are working with a geodesic map, the submanifolds $\operatorname{Exp}_x(\operatorname{span}\{v, e_3\})$ and $\operatorname{Exp}_x(\operatorname{span}\{e_1, e_2\})$ contain w. Similarly the submanifolds $x+\operatorname{span}\{\tilde{v}, \tilde{e}_3\}$ and $x+\operatorname{span}\{\tilde{e}_1, \tilde{e}_2\}$ contain $\tilde{w}'$. If the intersections of each of these pair of manifolds is a geodesic then the geodesic map must map one intersection to the other one, implying $\tilde{w}$ is proportional to $\tilde{w}'$. If the intersections are degenerate the case is trivial. Alternatively, one can prove this fact algebraically after some computations. It will be convenient for the rest of the proof so we will also include it here. If we call δ^* and $\tilde{\delta}'$ the angles formed by, respectively, the vectors e_2 and w, and the vectors $\tilde{e}_2$ and $\tilde{w}'$, then we have $\tilde{w}'$ is proportional to $\tilde{w}$ if $\tilde{\delta}' = \tilde{\delta}^*$, or equivalently $\delta' = \delta^*$. Using the definitions of w and $\tilde{w}'$ we have

$$\begin{aligned}\sin(\delta^*) = \sin\left(\arctan\left(\frac{\sin(\delta)}{\cos(\nu)\cos(\delta)}\right)\right) &= \frac{\tan(\delta)/\cos(\nu)}{(\tan(\delta)/\cos(\nu))^2 + 1} \\ &= \frac{\sin(\delta)}{\sqrt{\sin^2(\delta) + \cos^2(\nu)\cos^2(\delta)}},\end{aligned}$$

and analogously

$$\begin{aligned}\sin(\tilde{\delta}') = \sin\left(\arctan\left(\frac{\sin(\tilde{\delta})}{\cos(\nu)\cos(\tilde{\delta})}\right)\right) &= \frac{\tan(\tilde{\delta})/\cos(\nu)}{(\tan(\tilde{\delta})/\cos(\nu))^2+1} \\ &= \frac{\sin(\tilde{\delta})}{\sqrt{\sin^2(\tilde{\delta})+\cos^2(\nu)\cos^2(\tilde{\delta})}}.\end{aligned} \tag{3.6.15}$$

Using Lemma 3.2.1.c) for the pairs δ', $\tilde{\delta}'$ and δ^*, $\tilde{\delta}^*$, and the equations above we obtain

$$\sin(\delta^*) = \frac{\sin(\tilde{\delta})\sqrt{\frac{1+K\tilde{\ell}^2}{1+K\tilde{\ell}^2\sin^2(\tilde{\delta})}}}{\sqrt{\sin^2(\tilde{\delta})\frac{1+K\tilde{\ell}^2}{1+K\tilde{\ell}^2\sin^2(\tilde{\delta})}+\cos^2(\nu)\frac{\cos^2(\tilde{\delta})}{1+K\tilde{\ell}^2\sin^2(\tilde{\delta})}}} = \frac{\sin(\tilde{\delta})\sqrt{1+K\tilde{\ell}^2}}{\sqrt{\sin^2(\tilde{\delta})(1+K\tilde{\ell}^2)+\cos^2(\nu)\cos^2(\tilde{\delta})}},$$

and

$$\sin(\delta') = \frac{\sin(\tilde{\delta})}{\sqrt{\sin^2(\tilde{\delta})+\cos^2(\nu)\cos^2(\tilde{\delta})}}\sqrt{\frac{1+K\tilde{\ell}^2}{1+K\tilde{\ell}^2\left(\frac{\sin^2(\tilde{\delta})}{\sin^2(\tilde{\delta})+\cos^2(\nu)\cos^2(\tilde{\delta})}\right)}},$$

The two expressions on the right hand side are equal. This implies $\sin(\delta') = \sin(\delta^*)$. Since the angles were in the same quadrant we have $\delta' = \delta^*$.

We can now come back to the study of $\frac{\langle \nabla F(x), v\rangle}{\langle \nabla f(\tilde{x}), \tilde{v}\rangle}$. By (3.6.13) we have

$$\frac{\langle \nabla F(x), v\rangle}{\langle \nabla f(\tilde{x}), \tilde{v}\rangle} = \frac{\|\nabla F(x)\|}{\|\nabla f(\tilde{x})\|}\frac{\|v\|}{\|\tilde{v}\|}\frac{\langle \frac{\nabla F(x)}{\|\nabla F(x)\|}, \frac{w}{\|w\|}\rangle}{\langle \frac{\nabla f(\tilde{x})}{\|\nabla f(\tilde{x})\|}, \frac{\tilde{w}}{\|\tilde{w}\|}\rangle}\frac{\sqrt{\sin^2(\delta)+\cos^2(\delta)\cos^2(\nu)}}{\sqrt{\sin^2(\tilde{\delta})+\cos^2(\tilde{\delta})\cos^2(\nu)}}$$

We now operate the last two fractions. Using (3.6.10) and (3.6.13) we get that the product of the last two fractions above is equal to

$$\sqrt{\frac{1+K\tilde{\ell}^2}{(1+K\tilde{\ell}^2\sin^2(\tilde{\delta}^*))(1+K\tilde{\ell}^2\cos^2(\tilde{\beta}))}}\frac{\sqrt{\sin^2(\tilde{\delta})\frac{1+K\tilde{\ell}^2}{(1+K\tilde{\ell}^2\sin^2(\tilde{\delta}))}+\cos^2(\nu)\frac{\cos^2(\tilde{\delta})}{1+K\tilde{\ell}^2\sin(\tilde{\delta})}}}{\sin^2(\tilde{\delta})+\cos^2(\tilde{\delta})\cos^2(\nu)}$$

which after using (3.6.15), and simplifying it yields $\sqrt{\frac{1+K\tilde{\ell}^2}{(1+K\tilde{\ell}^2\sin^2(\tilde{\delta}))(1+K\tilde{\ell}^2\cos^2(\tilde{\beta}))}}$ (recall $\tilde{\delta}^* = \tilde{\delta}'$). So finally we have

$$\frac{\langle \nabla F(x), v\rangle}{\langle \nabla f(\tilde{x}), \tilde{v}\rangle} = \frac{\|\nabla F(x)\|}{\|\nabla f(\tilde{x})\|}\frac{\|v\|}{\|\tilde{v}\|}\sqrt{\frac{1+K\tilde{\ell}^2}{(1+K\tilde{\ell}^2\sin^2(\tilde{\delta}))(1+K\tilde{\ell}^2\cos^2(\tilde{\beta}))}}.$$

In order to bound the previous expression, we now use Lemma 3.2.1.c) and Lemma 3.2.1.a), and bound $\sin^2(\tilde{\delta})$ and $\cos^2(\tilde{\beta})$ by 0 or 1 depending on the inequality. Recall that, by (3.6.2) we have $1/\sqrt{1+K\tilde{\ell}^2} = \mathrm{C}_K(\ell)$, for $\ell = d(x, x_0) \leq R$. And $\tilde{\ell} = \|\tilde{x}\|$. Let's proceed. We obtain, for $K = -1$

$$\cosh^{-3}(R) \leq \frac{1}{\cosh^2(\ell)}\cdot 1 \cdot \frac{1}{\cosh(\ell)} \leq \frac{\langle \nabla F(x), v\rangle}{\langle \nabla f(\tilde{x}), \tilde{v}\rangle} \leq \frac{1}{\cosh(\ell)}\cdot\cosh^2(\ell)\cdot\cosh(\ell) \leq \cosh^2(R).$$

and for $K = 1$ it is

$$\cos^2(R) \leq \frac{1}{\cos(\ell)} \cdot \cos^2(\ell) \cdot \cos(\ell) \leq \frac{\langle \nabla F(x), v\rangle}{\langle \nabla f(\tilde{x}), \tilde{v}\rangle} \leq \frac{1}{\cos^2(\ell)} \cdot 1 \cdot \frac{1}{\cos(\ell)} \leq \cos^{-3}(R).$$

The first part of Lemma 3.2.2 follows, for $\gamma_{\mathrm{p}} = \cosh^{-3}(R)$ and $\gamma_{\mathrm{n}} = \cosh^{-2}(R)$ when $K = -1$, and $\gamma_{\mathrm{p}} = \cos^2(R)$ and $\gamma_{\mathrm{n}} = \cos^3(R)$ when $K = 1$.

The second part of Lemma 3.2.2 follows readily from the first one and g-convexity of F, as in the following. It holds

$$f(\tilde{x}) + \frac{1}{\gamma_{\mathrm{n}}}\langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x}\rangle \overset{①}{\leq} F(x) + \langle \nabla F(x), y - x\rangle \overset{②}{\leq} F(y) = f(\tilde{y}),$$

and

$$f(\tilde{x}) + \gamma_{\mathrm{p}}\langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x}\rangle \overset{③}{\leq} F(x) + \langle \nabla F(x), y - x\rangle \overset{④}{\leq} F(y) = f(\tilde{y}),$$

where ① and ③ hold if $\langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x}\rangle \leq 0$ and $\langle \nabla f(\tilde{x}), \tilde{y} - \tilde{x}\rangle \geq 0$, respectively, by the first part of this theorem. Inequalities ② and ④ hold by g-convexity of F.

■

3.7 Constants depending on R and K

We discuss the value of the constants of our algorithms in Remark 3.7.2 and discuss recent hardness results in Remark 3.7.3. But we start by proving a relevant result that says that the condition number of an L-smooth and μ-strongly g-convex function $F : \mathcal{B}_R \to \mathbb{R}$ is lower bounded by a term depending on R and K, where the condition number is defined by L/μ. This is unlike in the Euclidean case, for which there are functions with condition number 1.

In particular, we show that the function $x \mapsto \frac{1}{2}d(x, x_0)^2$ has minimum condition number on $\mathcal{B}_R$, and is $(R\sqrt{|K|}\cot_K(R))^{-\operatorname{sign}(K)}$, where $\cot_K(R)$ is the special cotangent that is $\cot(\sqrt{|K|}R)$ if $K > 0$ and $\coth(\sqrt{|K|}R)$ if $K < 0$. And $\operatorname{sign}(K)$ is $K/|K|$ for $K \neq 0$. The fact about the condition number of $\frac{1}{2}d(x, x_0)^2$ can be obtained from the proof of Fact 3.3.4, and actually the fact per se as a comparison geometry theorem that uses that the inequality there is satisfied with equality in the constant curvature case. However, we recover the computation of this condition number while proving the proposition.

Proposition 3.7.1. *Let $F : \mathcal{M}_K \to \mathbb{R}$ be an L-smooth and μ-strongly convex function on $\mathcal{B}_R \subset \mathcal{M}_K$. Assume F is twice differentiable with continuous Hessian. Then, the condition number L/μ of F on $\mathcal{B}_R$ is at least the condition number of the function $\frac{1}{2}d(x, x_0)^2$ on $\mathcal{B}_R$.*

Proof As we have done before, we can assume $K \in \{1, -1\}$ because the other cases can be reduced to this one by a rescaling, cf. Remark 3.6.1. Recall that by definition of $\mathcal{M}_K$ and $\mathcal{B}_R$, for $K > 0$, we have that $R < \pi/2\sqrt{K}$.

We start by noting that given F, we can obtain another function G whose condition number is at most the one of F and such that it is symmetric with respect to every rotation whose axis

goes through x_0. Formally, $G = G \circ \operatorname{Exp}_{x_0} \circ \sigma \circ \operatorname{Exp}_{x_0}^{-1}$ for a rotation $\sigma \in \mathrm{SO}(n)$. Equivalently, the function $G(x)$ depends on $\| \operatorname{Exp}_{x_0}^{-1}(x)\|$ only. Indeed, an average of F and itself after performing an arbitrary rotation σ, that is $(F + F \circ \operatorname{Exp}_{x_0} \circ \sigma \circ \operatorname{Exp}_{x_0}^{-1})/2$, has a condition number that is at most the condition number of F. This is due to the Hessian being linear and its maximum and minimum eigenvalues over the domain determining the condition number. That is, the smoothness constant can only decrease or stay the same after performing the average. It would only be the same if, at some point, the Hessian matrices of each of the two added functions both have the same eigenvector with maximum eigenvalue and it equals the smoothness constant. The argument for the minimum is analogous. This argument extends to the case in which we integrate the function, pointwise, over $\mathrm{SO}(n)$ after applying a rotation. That is, defining $g(x) = \int_{SO(n)} F \circ \operatorname{Exp}_{x_0} \circ \sigma \circ \operatorname{Exp}_{x_0}^{-1}(x) d\sigma$ we obtain a symmetric function with condition number that is at most the condition number of F. So without loss of generality we can solely study symmetric functions G and in fact, due to the symmetries we do not lose generality if we work in dimension $n = 2$.

Denote $y_x = \| \operatorname{Exp}_{x_0}^{-1}(x)\| \in \mathbb{R}$. We will express the condition number of G by using the function $g : \mathbb{R} \to \mathbb{R}$, defined as $g(y_x) = G(x)$ for any point $x \in \mathcal{M}_K$. Note the function is well defined by the symmetry property on G. A basis formed by (two) eigenvectors of $\nabla^2 G(x)$ can be chosen to have vectors in the direction of $\operatorname{Exp}_x^{-1}(x_0)$ and its normal. Indeed, either every vector is an eigenvector associated to the same eigenvalue, which satisfies the above, or by the symmetry of $\nabla^2 G(x)$, there exists a base $\{v_1, v_2\}$ of orthonormal eigenvectors, associated with different eigenvalues $\lambda_1 > \lambda_2$. By the symmetry of G we have that $\lambda_1 = v_1^\top \nabla^2 G(x) v_1 = v_1'^\top \nabla^2 G(x) v_1'$, where v_1' is the symmetric vector to v_1 with respect to $\operatorname{Exp}_x^{-1}(x_0)$. However, since $\lambda_1 \neq \lambda_2$ then the only unit vectors v that can satisfy $\lambda_1 = v^\top \nabla^2 G(x) v$ are $\pm v_1$, so $v_1 = v_1'$ and therefore v_1 and v_2 can be taken to be in the direction of $\operatorname{Exp}_x^{-1}(x_0)$ and its normal. Consequently, one eigenvalue of $\nabla^2 G(x)$ is $g''(y_x)$. We can compute the other eigenvalue by using the non-Euclidean cosine theorem, cf. Fact 3.6.5. In order to do this, first note that $\nabla G(x)$ must be in the direction of $\operatorname{Exp}_x^{-1}(x_0)$ by the symmetry of G and it must be $\|\nabla G(x)\| = g'(y_x)$. Now given $x \in \mathcal{M}$ and small enough $\eta \in \mathbb{R}$, we consider a right geodesic triangle with vertices x_0, x and z_η, where $z_\eta = \operatorname{Exp}_x(\eta v_2)$ for v_2 defined above. Recall it is a unit vector that is normal to $\operatorname{Exp}_x^{-1}(x_0)$ and it is an eigenvector of $\nabla^2 G(x)$. The definition of z_η implies that the angle between $\operatorname{Exp}_x^{-1}(x_0)$ and $\operatorname{Exp}_x^{-1}(x_2)$ is $\pi/2$ and $d(x, z_\eta) = \eta$. Let $\alpha(\eta)$ be the angle between $\operatorname{Exp}_{z_\eta}^{-1}(x_0)$ and $\operatorname{Exp}_{z_\eta}^{-1}(x)$. Since we are only interested about the eigenvalue of $\nabla^2 G(x)$ associated to the eigenvector v_2 we can project $\nabla G(z_\eta)$ onto $\operatorname{Exp}_{z_\eta}^{-1}(x)$, which has norm $\|\nabla G(z_\eta) \cos(\alpha(\eta))\|$. We compute the

eigenvalue as

$$\begin{aligned}\lim_{\eta\to 0}\frac{\|\nabla G(z_\eta)\|\cos(\alpha(\eta))}{\eta} &= \|\nabla G(x)\|\lim_{\eta\to 0}\frac{\cos(\alpha(\eta))}{\eta}\\ &\overset{①}{=} g'(y_x)\lim_{\eta\to 0}\frac{\mathrm{C}_K(d(x_0,x))-\mathrm{C}_K(\eta)\,\mathrm{C}_K(d(x_0,z_\eta))}{K\,\mathrm{S}_K(d(x_0,z_\eta))\eta\,\mathrm{S}_K(\eta)}\\ &\overset{②}{=} g'(y_x)\lim_{\eta\to 0}\frac{\mathrm{C}_K(d(x_0,x))(1-\mathrm{C}_K^2(\eta))/K}{\mathrm{S}_K(d(x_0,z_\eta))\eta\,\mathrm{S}_K(\eta)}\\ &\overset{③}{=} g'(y_x)\cot_K(d(x_0,x)) = g'(y_x)\cot_K(y_x).\end{aligned}$$

Above, ① uses the cosine theorem, cf. Fact 3.6.5, applied as

$$\mathrm{C}_K(d(x_0,x)) = \mathrm{C}_K(d(x,z_\eta))\,\mathrm{C}_K(d(x_0,z_\eta)) + K\cos(\alpha(\eta))\,\mathrm{S}_K(d(x_0,z_\eta))\,\mathrm{S}_K(d(x,z_\eta)).$$

Recall that we have $\eta = d(x,z_\eta)$ by definition. Equality ② uses the cosine theorem again, with a different ordering of the sides so we obtain

$$\mathrm{C}_K(d(x_0,z_\eta)) = \mathrm{C}_K(d(x_0,x))\,\mathrm{C}_K(d(x,z_\eta)),$$

by using the right angle of the geodesic triangle. Finally ③ simplifies some terms, since $(1-\mathrm{C}_K^2(\eta))/K = \mathrm{S}_K^2(\eta)$ and uses that $d(x_0,z_\eta)$ and $\mathrm{S}_K(\eta)/\eta$ tend to $d(x_0,x)$ and 1, respectively, when $\eta\to 0$. We conclude that the condition number of G is

$$\kappa_G = \frac{\max_{y\in[0,R]}\{g''(y), g'(y)\cot_K(y)\}}{\min_{y\in[0,R]}\{g''(y), g'(y)\cot_K(y)\}}. \tag{3.7.1}$$

We only need to prove that for any twice differentiable function $g:[0,R]\to\mathbb{R}$ with continuous second derivative, the quotient above is at least the value of the quotient that we obtain for $g(y) = y^2/2$, which is $(R\cot_K(R))^{-K}$. This is computed by noticing that, for that choice of g, we have that $g''(y) = 1$, that if $K = 1$ then $g'(y)\cot_K(y)\le 1$ and it reaches its minimum at $y = R$. If $K = -1$ then $g'(y)\cot_K(y)\ge 1$ and it is maximum at $y = R$. Note this implies that the condition number of $\frac{1}{2}d(x_0,x)^2$ on $\mathcal{B}_R$ is $(R\cot_K(R))^{-K}$, as it was advanced before.

Given g, let a,b be tight constants such that $g''(y)\in[a,b]$ for $y\in[0,R]$. Such constants must exist since g'' is a continuous function defined on a compact. We have $g'(y)\le by$, since by the symmetry and differentiability of G it must be $g'(0) = 0$. We obtain a lower bound on κ_G if we lower bound the numerator of (3.7.1) by $\max_{y\in[0,R]}\{g''(y)\} = b$ and if we upper bound the denominator by $g'(R)\cot_K(R)$. We obtain

$$\kappa_G \ge \frac{b}{g'(R)\cot_K(R)} \ge \frac{b}{Rb\cot_K(R)} = \frac{1}{R\cot_K(R)}.$$

Similarly, if we lower bound the denominator of (3.7.1) by $aR\cot_K(R)\le\max_{y\in[0,R]}\{g'(y)\cot_K(y)\}$ and upper bound the denominator by $a = \min_{y\in[0,R]}\{g''(y)\}$ we obtain

$$\kappa_G \ge \frac{aR\cot_K(R)}{a} = R\cot_K(R).$$

For each case $K \in \{1, -1\}$, there is only one of the lower bounds above such that its right hand side is greater than 1 and it precisely matches the value of the condition number of $\frac{1}{2}d(x_0, x)^2$ we computed above. ■

Remark 3.7.2. *The previous proposition intuitively suggests that it could be unavoidable to have some particular constants depending on R in the rates of any optimization algorithm. For starters, optimizing a g-convex function by adding a strongly g-convex regularizer and optimizing the resulting strongly g-convex problem would entail rates containing a factor depending on the condition number of the regularizer, which the proposition proves it is at least the value $(R\sqrt{|K|}\cot_K(R))^{-\operatorname{sign}(K)}$. This implies that in the case of positive curvature $K = 1$, a μ-strongly g-convex and L-smooth function defined on the ball $\mathcal{B}_R$ must have condition number that is at least $\tan(R)/R \in [\frac{2}{\pi\cos(R)}, \frac{1}{\cos(R)}]$. This grows fast with R, but it is only natural if one takes into account that no strongly g-convex function exists if $R \geq \frac{\pi}{2}$, due to the space containing a full geodesic circle (so the constant function is the only g-convex function in this domain). Optimization in manifolds of positive curvature only makes sense in spaces of low diameter.*

The classical domain of application of accelerated methods for strongly convex functions consists of functions with large condition number κ, due to the $\sqrt{\kappa}$-dependence of the rates. For $K = 1$, the constants of our algorithm $1/\gamma_{\mathrm{p}} = \cos^{-2}(R)$ and $1/\gamma_{\mathrm{n}} = \cos^{-3}(R)$ (we also have the constant $\sqrt{44\max\{1, R^2\}}$ coming from $\tilde{L}$) might seem large but they are a small polynomial of the minimum attainable condition number. If the condition number is large or, in its limit to infinity, whenever the function is g-convex, then acceleration is beneficial. For the case $K = -1$ the previous proposition shows that the minimum condition number is $R\tanh(R)$. In this case, our constants are $1/\gamma_{\mathrm{p}} = \cosh^3(R)$ and $1/\gamma_{\mathrm{n}} = \cosh^2(R)$, and a constant of a similar nature coming from $\tilde{L}$ (cf, Proof of Lemma 3.2.3), which do not present an analogous dependency with respect to the minimum attainable condition number as in the previous case. This exponential dependence could be due to the exponential volume that a ball contains in the hyperbolic space. Studying if these constants are necessary for a global full accelerated method is interesting open problem and future direction of research. Regardless, the essence of our results for μ-strongly g-convex functions is that we can optimize at a full accelerated rate globally as opposed to essentially fully accelerating in a small neighborhood of radius $O((\mu/L)^{3/4})$ around the minimizer (this is explicit in (Zhang and Sra, 2018) and implicit in (Ahn and Sra, 2020) since the rates of AGD are nearly achieved only after a number of steps that is what RGD needs to reach the neighborhood). Note that, additionally, we can achieve acceleration in the g-convex case, which was not possible before. In any case, we note that in machine learning applications, it has been observed that the iterates do not get far from initialization (Nagarajan and Kolter, 2019), especially in overparametrized models. Consequently, in such regime, R being a small constant is not a strong assumption and the constants of our algorithms do not become significant.

In the sequel, we discuss the work of Hamilton and Moitra (2021), that shows a hardness result in this direction. Our intuition is that, due to the geometry, it is necessary to have an additive

and/or multiplicative constant depending on R on the optimal rates of convergence, similarly to the multiplicative constant R that one has in the lower bound for the class of smooth and convex functions in the Euclidean space. And for *easy* strongly g-convex functions (low condition number), this hardness could dominate convergence. However, when the condition number is large, which is the traditional regime of application of accelerated methods, or in its limit to infinity, that is in the case of g-convexity, acceleration becomes again a very useful tool.

3.7.1 Comment on hardness results

Remark 3.7.3. *After this work was publicly available, the work (Hamilton and Moitra, 2021) was released. In the noisy setting, in the hyperbolic plane, the authors claim "*[to have] dashed these hopes [of having Nesterov-like accelerated algorithms] by showing that acceleration is impossible even in the simplest of settings where we want to minimize a smooth and strongly geodesically convex function over the hyperbolic plane*". We argue that is not the case.*

They essentially argue that, in their setting with a noisy oracle in the hyperbolic plane, one needs $\gtrsim R/\log(R)$ *noisy queries to the gradient or function value for optimizing functions of the form* $d(x,x^*)^2$*, while their condition numbers are* $L/\mu \approx R$ *so obtaining rates* $\lesssim \sqrt{L/\mu}$ *is impossible in general. But it does not preclude to have an algorithm with rates that are, for instance,* $\lesssim R + \sqrt{L/\mu}\log(1/\varepsilon)$*. Or a similar expression that involves some other additive or multiplicative constants depending on* R*. In fact, we believe that they are able to show that "*acceleration is impossible even in the simplest of settings*" precisely because they study the simplest of settings! That is, they show there is some hardness depending on the geometry. In particular, when the condition number is low this hardness can dominate the convergence. For instance, for rates* $R + \sqrt{L/\mu}\log(1/\varepsilon)$ *the* R *can dominate convergence unless* $L/\mu \geq R^2$ *or* ε *is small enough. The result does not mean that acceleration is doomed to fail. In fact, the problems for which acceleration gets the most improvement are ill-conditioned problems and for those one would expect to still have acceleration in their noisy setting. In particular, acceleration is of importance when* L/μ *is large or in the limit to infinity, that is, when the function is g-convex.*

We note that Hamilton and Moitra (2021) independently proved a similar result as our Proposition 3.7.1, limited to the hyperbolic plane. In particular they show that the condition number for an L*-smooth* μ*-strongly g-convex function* F *defined on the hyperbolic disk of curvature* $K=-1$ *must be* $\kappa_F = L/\mu \geq \Omega(R)$*, which is similar to the precise result that we found that had optimal constant* $R\cot(R)$*.*

In conclusion, we have:

- A lower bound $\widetilde{\Omega}(R)$ on the number of oracle queries needed for minimization, under certain assumptions.
- If the condition number is small, say it is about the smallest value it can take, $\frac{L}{\mu} = \widetilde{\Theta}(R)$, then obtaining complexity $\sqrt{\frac{L}{\mu}}$ is not possible.

- The lower bound does not preclude acceleration when $\frac{L}{\mu}$ is larger, say for instance $\frac{L}{\mu} \geq R^2$. In fact, one of the results of this chapter shows that indeed we can get $\sqrt{\frac{L}{\mu}}\log(\frac{1}{\varepsilon})$ up to constants depending on R and K and up to log factors.

- Despite of our constants, we obtain the same rates as AGD up to log factors when $R = O(1)$. This is not an unrealistic assumption, since it has been observed that in several machine learning applications, iterates do not get far from initialization (Nagarajan and Kolter, 2019), especially in overparametrized models.

Chapter 4

Proportional Fairness under Positive Linear Constraints

The proportional fair resource allocation problem is a major problem studied in flow control of networks, operations research, and economic theory, where it has found numerous applications. This problem, defined as the constrained maximization of $\sum_i \log x_i$, is known as the packing proportional fairness problem when the feasible set is defined by positive linear constraints and $x \in \mathbb{R}^n_{\geq 0}$. In this chapter, we present a distributed accelerated first-order method for this problem which improves upon previous approaches. The algorithm is *width-independent*.

4.1 Introduction

The assignment of bounded resources to several agents under some notions of fairness is a topic studied in networking, operations research, game theory, and economic theory. The allocation obtained by the maximization of the function $\sum_{i=1}^n \log(x_i)$ over a convex set $C \subseteq \mathbb{R}^n_{\geq 0}$, known as a *proportional fair allocation*, is an important solution that arises under a natural set of fairness axioms. These axioms are: Pareto optimality, affine invariance, independence of irrelevant alternatives (if the fair allocation of a feasible set is in a particular subset, the fair allocation of that subset is the same), and symmetry (labels or order of agents should not matter) (Bertsimas, Farias, and Trichakis, 2011; Lan et al., 2010). It corresponds to Nash bargaining solutions (Nash, 1950) and it also has applications to multi-resource allocation in compute clusters (Bonald and Roberts, 2015; Jin and Hayashi, 2018; Joe-Wong et al., 2012), rate control in networks (Kelly, 1997) and game theory (Jain and Vazirani, 2010; Jain and Vazirani, 2007). For example, in the field of rate control in networks, one can be interested about allocating

Other important allocations are linear objectives (no fairness), the max-min allocations (Mo and Walrand, 2000), or α-fair allocations (Atkinson, 1970; Mo and Walrand, 2000; McCormick et al., 2014), which generalize all of the others. Proportional fairness corresponds to $\alpha = 1$. A natural restriction, that many of these applications require, are positive linear constraints. This results in the *packing proportional fairness problem*, also known as the 1-*fair packing problem*.

The main focus of this chapter is on solving this problem. Given $A \in \mathcal{M}_{m\times n}(\mathbb{R}_{\geq 0})$ in the set of $m \times n$ matrices with non-negative entries, the 1-fair packing problem is defined[1] as

$$\max_{x\in\mathbb{R}^n_{\geq 0}} \left\{ f(x) \stackrel{\text{def}}{=} \sum_{i=1}^{n} \log x_i : Ax \leq \mathbb{1}_m \right\}. \tag{1FP}$$

We use the notation $\mathbb{1}_d$ for the d-dimensional vector of ones. We focus on *width-independent* algorithms that additively ε-approximate the optimum of those problems. That means, respectively, that we can find $\widehat{x}$ in time that depends at most polylogarithmically on the width ρ of the matrix A, and that it satisfies $f^* - f(\widehat{x}) \leq \varepsilon$, where f^* is the optimal value. Note that (1FP) has a unique optimizer, by strong concavity. The width ρ of A is defined as $\max\{A_{ij}\}/\min_{A_{ij}\neq 0}\{A_{ij}\}$, the maximum ratio of the non-zero entries of A. Note that in general width-dependent algorithms are not polynomial. Smoothness and Lipschitz constants of the objectives do not scale polylogarithmically with ρ and thus, direct application of classical first-order methods leads to non-polynomial algorithms. We note that all of the parameters on our algorithm use the known quantities ε, m and n, an in particular convergence rates depend on these values only.

Example 4.1.1. *Suppose we have a network represented by a weighted graph with m edges and suppose we have n source-target pairs. The weights of the graph correspond to capacities, like information that can be transmitted between two nodes in Internet protocols. Assume for simplicity that there is a single path connecting each source to its respective target. The resource allocation problem consists of assigning a flow to each source-target path such that the sum of the flows at each edge does not exceed the capacity of the edge. In such a case, the feasible set can be written, after a rescaling, as $\{x \in \mathbb{R}^n_{\geq 0} : Ax \leq \mathbb{1}_m\}$, where $A_{ij} \geq 0$. Allocation maximizing the total flow can lead to unfair solutions in which some pairs could even receive no flow. The only allocation satisfying the aforementioned fairness axioms is the solution to the problem we study in this chapter.*

Most works dealing with α-fair functions assume, without loss of generality, that A is given so that the minimum non-zero entry of A is 1 and the maximum entry is ρ. However, we assume without loss of generality that

$$\max_{j\in[m]}\{A_{ij}\} = 1, \text{ for all } i \in [n]. \tag{4.1.1}$$

We can do so because, for our problem, we can rescale each primal coordinate multiplicatively, rescaling the columns of A accordingly, which only changes the objectives by an additive constant. Thus, the additive guarantees we will obtain are also satisfied in the non-scaled problem.

[1] Consider $f(x) = -\infty$ if x is not feasible and the fact that f tends fast to $-\infty$ when any coordinate tends to 0. In such a case, the geometry of this problem is such that the graph of the function could be considered to be *an isolated hill or mountain with steep or precipitous sides*, that is, a butte. For this reason, we consider our algorithm is ***Eye**ing the **Bu**tte of **Pro**portional **Fairn**ess*, or, as a short name in a non-rhotic accent, it is **Ibuprofen**. We hope that, in the context of proportional fairness with positive linear constraints, our algorithm receives a widespread adoption as successful as that of the original Ibuprofen.

Our algorithm solves the problem in a distributed model of computation with n agents. Each agent $j \in [n]$ controls variable x_j and only has access to global parameters like m, n, or the target accuracy ε, to the j-th column of A, and in each round it receives the slack $(Ax)_i - 1$ of all the constraints i in which j participates. This is a standard distributed model of computation. We refer to (Kelly and Yudovina, 2014; Awerbuch and Khandekar, 2008) for its motivation and applications.

Notations

We let e_i be the vector with 1 in coordinate i and 0 elsewhere. We denote by A_i a row of A. Recall that for $k \in \mathbb{N}$, we use the notation $[k] \stackrel{\text{def}}{=} \{1, 2, \ldots, k\}$. Throughout this chapter, $\log(\cdot)$ represents the natural logarithm. For $v \in \mathbb{R}^n$, the notation $\exp(v)$ means entrywise exponential. We use $\odot$ for the entrywise product. Given a 1-strongly convex map ψ, we denote its Bregman divergence by $D_\psi(x, y) \stackrel{\text{def}}{=} \nabla\psi(x) - \nabla\psi(y) - \langle \nabla\psi(y), x - y\rangle$. We denote by N the number of non-zero entries of the matrix A. The notation $\widetilde{O}(\cdot)$ hides logarithmic factors with respect to m, n, $1/\varepsilon$ and ρ. But note that the rates of our algorithms do not depend on ρ.

Related Work

Despite the importance and widespread applicability of fairness objectives, width-independent (and thus polynomial) algorithms for many α-fair packing problems were not developed until recently. Width-independent algorithms were first designed for 0-fair packing, i.e., for packing linear programs (packing LPs), that have a longer history (Luby and Nisan, 1993). For this problem there are currently nearly linear-time width-independent iterative algorithms (Allen-Zhu and Orecchia, 2019) and distributed algorithms (Allen-Zhu and Orecchia, 2015; Diakonikolas and Orecchia, 2017). These works focus obtaining solutions whose rates have a small dependence on the input, motivated by the high dimensional case, at the expense of having polynomial dependence on $\frac{1}{\varepsilon}$. This is as opposed to interior point methods, that can achieve high accuracy but with worse dependence on the input. Likewise, we focus on obtaining solutions whose rates have with low dependence on the input. Marašević, Stein, and Zussman (2016) studied the width-independent optimization of α-fair packing problems for any $\alpha \in [0, \infty]$ with a stateless algorithm and Diakonikolas, Fazel, and Orecchia (2020) gave better rates with a non-stateless algorithm. Both works use the same distributed framework as ours. For the particular case of 1-fair packing, the latter work obtains an unaccelerated algorithm that runs in $\widetilde{O}(n^2/\varepsilon^2)$ distributed iterations. Beck, Nedic, et al. (2014) study the optimization of the dual problem by using Nesterov's accelerated method, and then they reconstruct a primal solution. However, both primal and dual solutions depend on the smoothness constant of the dual problem, which in the worst case is proportional to ρ^2, and therefore it is not a polynomial algorithm. In contrast, our algorithms do not depend on ρ at all. Obtaining a priori lower bounds on each of the coordinates of the optimizer is of theoretical and practical interest, since it provides certain amount of resource that can be assigned to each agent before solving the problem. These were studied in (Marašević,

Table 4.1: Comparison of algorithms for 1-fair packing. The work of one iteration is linear in N, the number of non-zero entries in A.

Work	Problem	Iterations	Width-dependence?
Beck, Nedic, et al. (2014)	Primal	$O(\rho^2 mn/\varepsilon)$	Yes
Marašević, Stein, and Zussman (2016)	Primal	$\widetilde{O}(n^5/\varepsilon^5)$	nearly No (polylog)
Diakonikolas, Fazel, and Orecchia (2020)	Primal	$\widetilde{O}(n^2/\varepsilon^2)$	nearly No (polylog)
Our Theorem 4.2.5	Primal	$\widetilde{O}(n/\varepsilon)$	No
Beck, Nedic, et al. (2014)	Dual	$O(\rho\sqrt{mn/\varepsilon})$	Yes
Criado, Martínez-Rubio, and Pokutta (2021)	Dual	$\widetilde{O}(n^2/\varepsilon)$	No

Stein, and Zussman, 2016) and were improved by Allybokus et al. (2018). In Lemma 4.3.1, we show a lower bound of this kind for our problem when it is normalized as in (4.1.1).

Contribution and Main Results

Our contribution can be summarized as follows; See Table 4.1 for a comparison with previous works. We design a distributed accelerated algorithm for 1-fair packing by using an accelerated technique that uses truncated gradients of a regularized objective, similarly to (Allen-Zhu and Orecchia, 2019) for packing LP. However, in contrast, our algorithm and its guarantees are deterministic. Also, our algorithm makes use of a different regularization and an analysis that yields additive error guarantees as opposed to multiplicative ones. Our algorithm is width-independent.

4.2 A distributed accelerated algorithm for 1-Fair Packing

Allen-Zhu and Orecchia (2019) designed a packing LP algorithm (0-fair packing) that is an accelerated coordinate descent method applied to a smoothed objective and that runs in $\widetilde{O}(N/\varepsilon)$ total expected work in order to approximate a solution with a *multiplicative* error. The main techniques are the application of linear coupling (Allen-Zhu and Orecchia, 2017) to non-standard instances of mirror descent and gradient descent. In particular, the mirror descent method is run with a truncated *stochastic* gradient and the descent method uses a particular smoothness property that their problem satisfies. In this section, we apply similar techniques in order to obtain a *deterministic* accelerated descent method applied to a smoothed objective coming from the 1-fair packing problem, that approximates the objective *additively*. The analysis follows the philosophy of mixing an online learning algorithm with a descent method, that was explained in Section 2.5. We note that Diakonikolas, Fazel, and Orecchia (2020) also made use of this smoothing and truncated gradients for the 1-fair packing problem, but their algorithm does not achieve acceleration and their analysis is more similar to the one of gradient descent.

We reparametrize Problem (1FP) so that the objective function is linear at the expense of making the constraints more complex. That is, we define the function $\hat{f} : \mathbb{R}^n \to \mathbb{R}, x \mapsto$

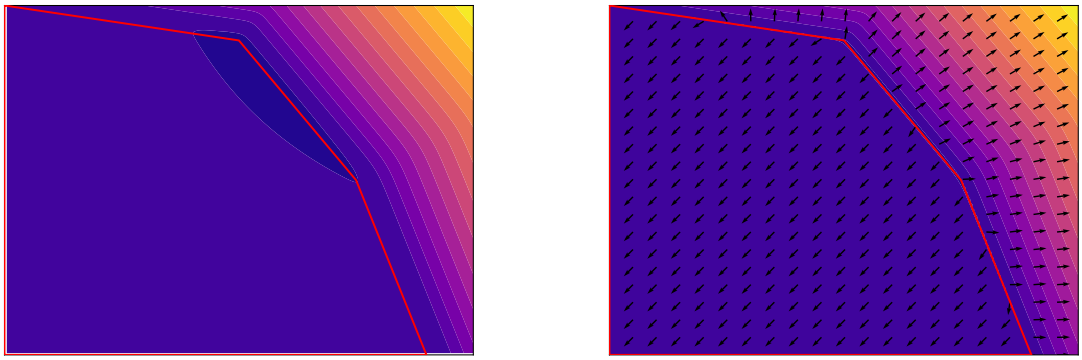


Figure 4.1: Regularized objective f_r (left) and its gradient (right), for a sample matrix $A \in \mathcal{M}_{3\times 2}$. For visualization purposes we show $\log(f_r(x))$ and $\log(\|\nabla f_r(x)\|)$, represented by color, and we indicate the direction of the gradient with normalized arrows. Also, note that we show the results in the original space (i.e., before reparametrizing, so the constraints appear to be linear) but the gradient was computed as originally defined in (4.2.2) (i.e., after reparametrizing).

$f(\exp(x)) = \langle \mathbb{1}_n, x\rangle$. The optimization problem becomes

$$\max_{x\in\mathbb{R}^n} \left\{ \hat{f}(x) \stackrel{\text{def}}{=} \langle \mathbb{1}_n, x\rangle : A\exp(x) \le \mathbb{1}_m \right\}. \tag{4.2.1}$$

Then, we regularize the negative of the reparametrized objective by adding a fast-growing barrier:

$$f_r(x) \stackrel{\text{def}}{=} -\langle \mathbb{1}_n, x\rangle + \frac{\beta}{1+\beta}\sum_{i=1}^{m} (A\exp(x))_i^{\frac{1+\beta}{\beta}},$$

with gradient at coordinate $j \in [m]$ given by

$$\nabla_j f_r(x) = -1 + \sum_{i=1}^{n} (A\exp(x))_i^{\frac{1}{\beta}} a_{ij} \exp(x_j), \tag{4.2.2}$$

where $\beta \stackrel{\text{def}}{=} \frac{\varepsilon}{6n\log(2mn^2/\varepsilon)}$. In this way, we can work with an unconstrained minimization problem. The resulting function is not globally smooth but when the absolute value of a coordinate of the gradient is large, it is positive, and in that case we are able to take a small gradient descent step and decrease the function considerably. The intuition is that if the gradient is large, then the function value along the segment of the gradient step, as a function of the step, can decrease fast. But it cannot increase fast since there are no large negative gradient coordinates. We depict f_r in Figure 4.1. The barrier also allows to maintain almost feasibility, as we show in Proposition 4.2.1 below. It is chosen to grow fast enough so that a point satisfying $(A\exp(x))_i > 1+\varepsilon/n$, for some $i \in [n]$, will have an optimality gap that is greater than the required accuracy. On the other hand, the regularizer is very small in the feasible region that is not too close to the boundary.

Let x^* be the maximizer of $\hat{f}$, let $\bar{x}^* \stackrel{\text{def}}{=} \exp(x^*)$ be the solution to Problem (1FP), and let x_r^* be the minimizer of f_r. We have $x^* \in [-\log(n), 0]^n$ by Lemma 4.3.1. Let $\omega \stackrel{\text{def}}{=} \log(mn/(1-\varepsilon/n))$ and define the box $B \stackrel{\text{def}}{=} [-\omega, 0]^n$. We restrict ourselves to this domain and formulate our final

problem, that we will minimize with an accelerated method:

$$\min_{x \in B} f_r(x). \tag{1FP-primalReg}$$

Note $f_r(x) \geq 0$ if $x \in B$. We add the redundant and simple box constraints B in order to later guarantee a bound on the regret of the mirror descent method that runs within the algorithm. We show that it suffices to obtain an ε-minimizer of Problem (1FP-primalReg) in order to obtain an $O(\varepsilon)$-minimizer for the original Problem (1FP).

Proposition 4.2.1. *Let $\varepsilon \in (0, n/2]$. Let x_r^* be the minimizer of (1FP-primalReg), let x^* be the maximizer of $\hat{f}$ and let $x_r^\varepsilon \in B$ be a point such that $f_r(x_r^\varepsilon) - f_r(x_r^*) \leq \varepsilon$, i.e., an ε-minimizer of (1FP-primalReg). Then we have:*

1) $\hat{f}(x^*) - \hat{f}(x_r^*) \leq \hat{f}(x^*) + f_r(x_r^*) \leq 3\varepsilon$.

2) *The point x_r^ε satisfies $A \exp(x_r^\varepsilon) \leq 1 + \varepsilon/n$.*

3) *The point $u = x_r^\varepsilon - \log(1+\varepsilon/n)\mathbb{1}_n$ satisfies $f(\exp(x^*)) - f(\exp(u)) = \hat{f}(x^*) - \hat{f}(u) \leq 5\varepsilon$ and $A \exp(u) \leq \mathbb{1}_m$.*

Proof Recall $\beta = \frac{\varepsilon}{6n \log(2mn^2/\varepsilon)}$ and that $\log(\cdot)$ is the natural logarithm. For the first part, take the point $x = \log(1-\varepsilon/n)\mathbb{1}_n + x^* \in B$. It satisfies $A \exp(x) \leq (1-\varepsilon/n)\mathbb{1}_m$, because x^* is feasible. Thus

$$\frac{\beta}{1+\beta} \sum_{i=1}^{m} (A \exp(x))_i^{\frac{1+\beta}{\beta}} \leq m(1-\varepsilon/n)^{1/\beta} \leq m \left(\frac{\varepsilon}{2mn^2}\right)^6 \leq \varepsilon. \tag{4.2.3}$$

We used $\frac{\beta}{1+\beta} \leq 1$, $\frac{1+\beta}{\beta} \geq \frac{1}{\beta}$, and $(1-\varepsilon/n)^{\frac{n}{\varepsilon}} \leq e^{-1}$. Consequently, we have

$$\begin{aligned} \hat{f}(x^*) - \hat{f}(x_r^*) &\overset{①}{\leq} \hat{f}(x^*) + f_r(x_r^*) \overset{②}{\leq} \hat{f}(x^*) + f_r(x) \\ &= \langle \mathbb{1}_n, x^* \rangle + \left(-\langle \mathbb{1}_n, \log(1-\varepsilon/n)\mathbb{1}_n + x^* \rangle + \frac{\beta}{1+\beta} \sum_{i=1}^{m} (A \exp(x))_i^{\frac{1+\beta}{\beta}} \right) \\ &\overset{③}{\leq} n \log\left(\frac{1}{1-\varepsilon/n}\right) + \varepsilon \overset{④}{\leq} 3\varepsilon. \end{aligned} \tag{4.2.4}$$

Above, ① is true by definition of f_r being $-\hat{f}$ plus a non-negative regularizer. The point x is in B and $x_r^* = \arg\min_{x \in B}\{f_r(x)\}$ so we have ②. Inequality ③ uses (4.2.3) and ④ uses $\log(x) \leq x - 1$ and $\varepsilon/n \leq 1/2$.

For the second part, suppose for the moment that there is some i such that $(A \exp(x_r^\varepsilon))_i > 1 + \varepsilon/n$. In that case

$$(A \exp(x_r^\varepsilon))_i^{\frac{1+\beta}{\beta}} \geq (1+\varepsilon/n)^{(2n/\varepsilon)\cdot 3\log(2mn^2/\varepsilon)} \geq \left(\frac{2mn^2}{\varepsilon}\right)^3,$$

since $(1+\varepsilon/n)^{2n/\varepsilon} \geq e$ when $\varepsilon/n \leq 1/2$. We have $x_r^\varepsilon \in B$ so $f_r(x_r^\varepsilon) \geq -\langle \mathbb{1}_n, x_r^\varepsilon\rangle + \frac{\beta}{1+\beta}\left(\frac{2mn^2}{\varepsilon}\right)^3 \geq \frac{\beta}{2}\left(\frac{2mn^2}{\varepsilon}\right)^3$. On the other hand, it holds for the point $y = -\log(mn)\mathbb{1}_n$ that

$$\begin{aligned} f_r(y) &= n\log(mn) + \frac{\beta}{1+\beta}\sum_{i=1}^{m}(A\exp(y))_i^{\frac{1+\beta}{\beta}} \\ &\overset{①}{\leq} n\log(mn) + m\,(1/m)^{\frac{1+\beta}{\beta}} \leq n\log(mn)+1 \\ &\overset{②}{<} \frac{\beta}{2}\left(\frac{2mn^2}{\varepsilon}\right)^3 - \varepsilon < f_r(x_r^\varepsilon) - \varepsilon, \end{aligned} \tag{4.2.5}$$

contradicting the assumption $f_r(x_r^\varepsilon) - f_r(x_r^*) \leq \varepsilon$, as we would obtain $\varepsilon < f_r(x_r^\varepsilon) - f_r(y) \leq f_r(x_r^\varepsilon) - f_r(x_r^*)$, since $y \in B$. So it must be $(A\exp(x_r^\varepsilon))_i \leq 1+\varepsilon/n$. Inequality ① uses that the maximum entry of A is 1, and $\frac{\beta}{1+\beta} \leq 1$. One can show ② by proving the stronger inequality that results from substituting β by $\varepsilon/(6n\cdot 2mn^2/\varepsilon)$, which is a lower value. Computing derivatives in both sides shows that this inequality holds if it does for $m=1$ and $\varepsilon = \frac{n}{2}$, and the latter is easy to check.

For the third part, we have $A\exp(u) = A\frac{\exp(x_r^\varepsilon)}{1+\varepsilon/n} \leq \mathbb{1}_m$. And finally, putting all together we obtain

$$\begin{aligned} \hat{f}(x^*) - \hat{f}(u) &= \hat{f}(x^*) - \hat{f}(x_r^\varepsilon) + n\log(1+\varepsilon/n) \leq \hat{f}(x^*) + f_r(x_r^\varepsilon) + n\log(1+\varepsilon/n) \\ &\leq \hat{f}(x^*) + f_r(x_r^*) + \varepsilon + n\log(1+\varepsilon/n) \leq 4\varepsilon + n\log(1+\varepsilon/n) \leq 5\varepsilon. \end{aligned}$$

■

Algorithm 7 Accelerated descent method for 1-Fair Packing

Input: Matrix $A \in \mathcal{M}_{m\times n}(\mathbb{R}_{\geq 0})$ normalized as in (4.1.1). Accuracy $\varepsilon \in (0, n/2]$.

$\beta \leftarrow \frac{\varepsilon}{6n\log(2mn^2/\varepsilon)}$; $\omega \leftarrow \log(\frac{mn}{1-\varepsilon/n})$; $L = \max\left\{\frac{4\omega(1+\beta)}{\beta}, \frac{16n\log(2mn)}{3\varepsilon} + \frac{1}{3}\right\} = \widetilde{O}(n/\varepsilon)$

$\eta_0 \leftarrow \frac{1}{3L}$; $C_k = 3\eta_k L$; $\tau \leftarrow \tau_k = \eta_k/C_k = 1/3L$.

$T \leftarrow \lceil \log(\frac{4n\log(2mn)}{\varepsilon})/\log(\frac{1}{1-\tau})\rceil \leq \lceil 3L\log(\frac{4n\log(2mn)}{\varepsilon})\rceil = \widetilde{O}(n/\varepsilon)$

$x^{(0)} \leftarrow y^{(0)} \leftarrow z^{(0)} \leftarrow -\log(mn/(1-\varepsilon/n))\mathbb{1}_n$

for $k = 1$ **to** T **do**

  $\eta_k \leftarrow C_k - C_{k-1} = \frac{1}{1-\tau}\eta_{k-1}$

  $x^{(k)} \leftarrow \tau z^{(k-1)} + (1-\tau)y^{(k-1)}$

  $z^{(k)} \leftarrow \arg\min_{z\in B}\left\{\frac{1}{2\omega}\|z - z^{(k-1)}\|_2^2 + \langle \eta_k \overline{\nabla f_r}(x^{(k)}), z\rangle\right\}$ ◇ Mirror descent step

  $y^{(k)} \leftarrow x^{(k)} + \frac{1}{\eta_k L}(z^{(k)} - z^{(k-1)})$ ◇ Gradient descent step

end for

return $\widehat{x} \overset{\text{def}}{=} \exp\left(y^{(T)}\right)/(1+\varepsilon/n)$

Output: $f(\widehat{x}) - f(\bar{x}^*) \leq \varepsilon$ and $\widehat{x}$ is feasible, i.e., $A\widehat{x} \leq 1$. The total number of iterations is $\widetilde{O}(n/\varepsilon)$ to obtain an $O(\varepsilon)$-approximate solution.

In the sequel, we will present the different parts of Algorithm 7 and their analyses. In particular, the notation and definitions used are compatible with the choices in the algorithm and most of the parameter choices naturally occur throughout the arguments. Our optimization algorithm starts at the points $x^{(0)} = y^{(0)} = z^{(0)} = -\log(mn/(1-\varepsilon/n))\mathbb{1}_n$ and updates each of

these variables $x^{(k)}, y^{(k)}$ and $z^{(k)}$ once in each iteration. They remain in B, cf. Lemma 4.3.2. The role of the three variables is the following: $z^{(k)}$ will be a mirror point and $y^{(k)}$ will be a gradient descent point, in the sense that in order to compute them we apply mirror descent and gradient descent. Then, the point $x^{(k)}$ will be a convex combination of both, that will balance the regret of $z^{(k)}$ with the primal progress of $y^{(k)}$, effectively coupling these two algorithms.

It is important to note that we do not use the gradient $\nabla f_r(x)$ for our mirror descent loss. Instead, we use a truncation of the gradient, similarly to (Diakonikolas, Fazel, and Orecchia, 2020). More precisely, the loss we perform the mirror descent step on is the truncated gradient $\overline{\nabla f_r}(x^{(k)}) \in \mathbb{R}^n$ defined as

$$\overline{\nabla_i f_r}(x^{(k)}) \stackrel{\text{def}}{=} \min\{1, \nabla_i f_r(x^{(k)})\} \text{ for all } i \in [n]. \tag{4.2.6}$$

Note that $\overline{\nabla f_r}(x^{(k)}) \in [-1,1]^n$ because $\nabla f_r(x) \in [-1,\infty]^n$ for any $x \in \mathbb{R}^n$, as the regularizer has positive gradient; see also definition of $f_r(x)$ and its gradient. The truncation allows mirror descent to control one part of the regret, which will not depend on the global Lipschitz constant. Gradient descent will compensate for both such regret and the part that is not controlled by mirror descent.

Let $\Pi_{\mathcal{X}}(\cdot)$ be the $\|\cdot\|_2$-projection map of a point onto a convex set $\mathcal{X}$. The mirror descent update can be written in closed form as any of the two following equivalent ways

$$\begin{aligned} z^{(k)} &\leftarrow \Pi_B(z^{(k-1)} - \omega\eta_k \overline{\nabla f_r}(x^{(k)})), \\ z_i^{(k)} &\leftarrow \Pi_{[-\omega,0]}(z_i^{(k-1)} - \omega\eta_k \overline{\nabla_i f_r}(x^{(k)})), \text{ for all } i \in [n]. \end{aligned} \tag{4.2.7}$$

That is, projecting back to the box, in case of the $\|\cdot\|_2$, consists of simply clipping each coordinate. We bound the regret coming from this mirror descent step by making use of a slight variation of the classical mirror descent lemma.

Lemma 4.2.2 (Mirror Descent Guarantee). *Let $u \in B$ and choose L as in Algorithm 7. It holds that:*

$$\langle \eta_k \overline{\nabla f_r}(x^{(k)}), z^{(k-1)} - u\rangle \le \eta_k^2 L \langle \overline{\nabla f_r}(x^{(k)}), x^{(k)} - y^{(k)}\rangle + \frac{1}{2\omega}\|z^{(k-1)} - u\|_2^2 - \frac{1}{2\omega}\|z^{(k)} - u\|_2^2.$$

Proof Use Lemma 4.3.4.b) with loss $\ell^{(k)} = \overline{\nabla f_r}(x^{(k)})$, learning rate $\eta = \eta_k$, and regularizer $\psi(x) = \frac{1}{2\omega}\|x\|_2^2$, that yields Bregman divergence $D_\psi(x,y) = \frac{1}{2\omega}\|x-y\|_2^2$. Use that $z^{(k-1)} - z^{(k)} = \eta_k L(x^{(k)} - y^{(k)})$.

■

Next, we will analyze the role of the gradient descent step. For this, we will use the following lemma, that can be derived from a lemma in (Diakonikolas, Fazel, and Orecchia, 2020). The details of the proof of the latter lemma were not provided, so for the sake of completeness, we include it as Lemma 4.3.3 and provide a full proof in Section 4.3.

Lemma 4.2.3 (Descent Lemma). *Given $x^{(k)}$ and $y^{(k)}$ as defined in Algorithm 7, the following holds:*

$$f_r(x^{(k)}) - f_r(y^{(k)}) \geq \frac{1}{2}\langle \nabla f_r(x^{(k)}), x^{(k)} - y^{(k)} \rangle \geq 0.$$

Proof We have $x^{(k)} - y^{(k)} = (z^{(k-1)} - z^{(k)})/\eta_k L$ by definition of the gradient descent step. With this, we first conclude that $\frac{1}{2}\langle \nabla f_r(x^{(k)}), x^{(k)} - y^{(k)} \rangle \geq 0$, as $\nabla_i f_r(x^{(k)})$ and $x_i^{(k)} - y_i^{(k)}$ have the same sign for all $i \in [n]$, cf. (4.2.7).

We apply Lemma 4.3.3 with $y^{(k)}$ corresponding to $x + \Delta$ and $x^{(k)}$ corresponding to x. To this end, we choose each c_i satisfying ① below

$$\frac{c_i \beta}{4(1+\beta)} |\overline{\nabla_i f_r}(x^{(k)})| \overset{①}{=} |x_i^{(k)} - y_i^{(k)}| \overset{②}{=} \frac{1}{\eta_k L} |z_i^{(k-1)} - z_i^{(k)}| \overset{③}{\leq} \frac{\omega}{L} |\overline{\nabla_i f_r}(x^{(k)})|,$$

where ② holds by definition of $y^{(k)}$ and ③ holds by the mirror descent update (4.2.7). Thus, it suffices to pick c_i such that $c_i \leq \frac{4\omega(1+\beta)}{\beta L} \leq 1$, where the last inequality holds true by the definition of L. In fact, the value of L was chosen to satisfy the previous inequality. Hence, Lemma 4.3.3 can be applied. We obtain:

$$f_r(x^{(k)}) - f_r(y^{(k)}) \geq \sum_{i=1}^{n} \left(1 - \frac{c_i}{2}\right) \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \geq \frac{1}{2}\langle \nabla f_r(x^{(k)}), x^{(k)} - y^{(k)} \rangle.$$

as desired. ■

4.2.1 Coupling Mirror Descent and Gradient Descent

We first prove a lemma that shows we can compensate for the regret coming from mirror descent as well as the rest of the regret.

Lemma 4.2.4. *Let $C_k \overset{\text{def}}{=} 3\eta_k L$, and let $\nu^{(k)} \overset{\text{def}}{=} \nabla f_r(x^{(k)}) - \overline{\nabla f_r}(x^{(k)}) \in [0, \infty)^n$. For all $u \in B$, we have*

$$\langle \eta_k \nu^{(k)}, z^{(k-1)} - u \rangle + {\eta_k}^2 L \langle \overline{\nabla f_r}(x^{(k)}), x^{(k)} - y^{(k)} \rangle \leq C_k (f_r(x^{(k)}) - f_r(y^{(k)})).$$

Proof It is enough to show that for all $i \in [n]$ we have

$$\eta_k \nu_i^{(k)} (z_i^{(k-1)} - u_i) + {\eta_k}^2 L \overline{\nabla_i f_r}(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \leq \frac{3}{2} \eta_k L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \tag{4.2.8}$$

because then we can conclude with

$$\begin{aligned} \langle \eta_k \nu^{(k)}, z^{(k-1)} - u \rangle + {\eta_k}^2 L \langle \overline{\nabla f_r}(x^{(k)}), x^{(k)} - y^{(k)} \rangle &\overset{①}{\leq} \frac{3}{2} \eta_k L \langle \nabla f_r(x^{(k)}), x^{(k)} - y^{(k)} \rangle \\ &\overset{②}{\leq} 3\eta_k L (f_r(x^{(k)}) - f_r(y^{(k)})), \end{aligned} \tag{4.2.9}$$

by adding up (4.2.8) in ① and Lemma 4.2.3 in ②. In the analysis of (4.2.8) we exploit the simple but crucial fact that is that the gradient step for each coordinate is independent of the gradient step of other coordinates, due to the constraint set being a box. The rest of the analysis

is analogous to Lemma 3.10 in (Allen-Zhu and Orecchia, 2019). We present the proof in three cases. In the cases below, we will use $\nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \geq 0$, cf. Lemma 4.2.3. And also the fact that $\eta_k \leq 1/4$, as we observe in (4.2.13).

- If $\nu_i^{(k)} = 0$ then $\overline{\nabla_i f_r}(x^{(k)}) = \nabla_i f_r(x^{(k)}) \in [-1, 1]$. In such a case, we have

$$\begin{aligned}\eta_k \nu_i^{(k)}(z_i^{(k-1)} - u_i) + {\eta_k}^2 L \overline{\nabla_i f_r}(x^{(k)})(x_i^{(k)} - y_i^{(k)}) &= {\eta_k}^2 L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \\ &\leq \frac{3}{2}\eta_k L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}).\end{aligned}$$

- If $\nu_i^{(k)} > 0$ and $z_i^{(k)} > -\omega$ then the mirror descent step did not need to project along coordinate i, and we have $z_i^{(k)} = z_i^{(k-1)} - \omega\eta_k$, and thus $y_i^{(k)} = x_i^{(k)} - \omega/L$. In this case

$$\begin{aligned}&\eta_k \nu_i^{(k)}(z_i^{(k-1)} - u_i) + {\eta_k}^2 L \overline{\nabla_i f_r}(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \\ &\overset{①}{\leq} \eta_k \nabla_i f_r(x^{(k)})\omega + {\eta_k}^2 L \nabla_i f_r(x^{(k)})(x^{(i)} - y^{(i)}) \\ &= \eta_k L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}) + {\eta_k}^2 L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \\ &\leq \frac{3}{2}\eta_k L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}).\end{aligned}$$

  Above, we obtain ① from $z_i^{(k)} - u_i \leq \omega$ because $z^{(k)}, u \in B$, the fact that $\nu_i^{(k)}$ and $x_i^{(k)} - y_i^{(k)}$ are positive, and $1 = \overline{\nabla_i f_r}(x^{(k)}) \leq \nabla_i f_r(x^{(k)})$, $0 < \nu_i^{(k)} \leq \nabla_i f_r(x^{(k)})$.

- If $\nu_i^{(k)} > 0$ and $z_i^{(k)} = -\omega$ then

$$\begin{aligned}&\eta_k \nu_i^{(k)}(z_i^{(k-1)} - u_i) + {\eta_k}^2 L \overline{\nabla_i f_r}(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \\ &\overset{①}{\leq} \eta_k \nabla_i f_r(x^{(k)})(z_i^{(k-1)} - z_i^{(k)}) + {\eta_k}^2 L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \\ &\overset{②}{\leq} {\eta_k}^2 L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}) + {\eta_k}^2 L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}) \\ &\leq \frac{3}{2}\eta_k L \nabla_i f_r(x^{(k)})(x_i^{(k)} - y_i^{(k)}).\end{aligned}$$

We have ① because in this case, $u_i - z_i^{(k)}$, $x_i^{(k)} - y_i^{(k)}$, $\nu_i^{(k)}$, $\overline{\nabla_i f_r}(x^{(k)})$ are all ≥ 0. We also used $0 < \nu_i^{(k)} < \nabla_i f_r(x^{(k)})$, $0 < \overline{\nabla_i f_r}(x^{(k)}) < \nabla_i f_r(x^{(k)})$. In ②, we used $z_i^{(k-1)} - z_i^{(k)} = \eta_k L(x_i^{(k)} - y_i^{(k)})$. ■

With these tools at hand, we can now use a linear coupling argument to establish an accelerated convergence rate. Note that the algorithm is as simple as performing the mirror descent, gradient descent and coupling, after a careful choice of parameters. All of which depend on known quantities.

Theorem 4.2.5. *Let $\varepsilon \leq n/2$ and let $\bar{x}^*$ be the solution to (1FP) and let x_r^* be the minimizer of (1FP-primalReg). Algorithm 7 computes a point $y^{(T)} \in B$ such that $f_r(y^{(T)}) - f_r(x_r^*) \leq \varepsilon$ in a number of iterations $T = \widetilde{O}(n/\varepsilon)$. Besides, $\widehat{x} \overset{\text{def}}{=} \exp\left(y^{(T)}\right)/(1 + \varepsilon/n)$ is a feasible point of (1FP), i.e., $A\widehat{x} \leq \mathbb{1}_m$, such that $f(\bar{x}^*) - f(\widehat{x}) \leq 5\varepsilon = O(\varepsilon)$.*

Proof We start by bounding the gap with respect to $x^{(k)}$:

$$
\begin{aligned}
&\eta_k(f_r(x^{(k)}) - f_r(u)) \\
&\overset{①}{\leq} \langle \eta_k \nabla f_r(x^{(k)}), x^{(k)} - u \rangle \\
&= \langle \eta_k \nabla f_r(x^{(k)}), x^{(k)} - z^{(k-1)} \rangle + \langle \eta_k \nu^{(k)}, z^{(k-1)} - u \rangle + \langle \eta_k \overline{\nabla f_r}(x^{(k)}), z^{(k-1)} - u \rangle \\
&\overset{②}{=} \frac{(1-\tau)\eta_k}{\tau} \langle \nabla f_r(x^{(k)}), y^{(k-1)} - x^{(k)} \rangle \rangle + \langle \eta_k \nu^{(k)}, z^{(k-1)} - u \rangle + \langle \eta_k \overline{\nabla f_r}(x^{(k)}), z^{(k-1)} - u \rangle \\
&\overset{③}{\leq} \frac{(1-\tau)\eta_k}{\tau} (f_r(y^{(k-1)}) - f_r(x^{(k)})) + \langle \eta_k \nu^{(k)}, z^{(k-1)} - u \rangle + \langle {\eta_k}^2 L \overline{\nabla f_r}(x^{(k)}), x^{(k)} - y^{(k)} \rangle \\
&\qquad + \frac{1}{2\omega} \|z^{(k-1)} - u\|_2^2 - \frac{1}{2\omega} \|z^{(k)} - u\|_2^2] \\
&\overset{④}{\leq} \frac{(1-\tau)\eta_k}{\tau} (f_r(y^{(k-1)}) - f_r(x^{(k)})) + C_k (f_r(x^{(k)}) - f_r(y^{(k)}) + \frac{1}{2\omega} \|z^{(k-1)} - u\|_2^2 \\
&\qquad - \frac{1}{2\omega} \|z^{(k)} - u\|_2^2 \\
&\overset{⑤}{\leq} \eta_k f_r(x^{(k)}) + (C_k - \eta_k) f_r(y^{(k-1)}) - C_k f_r(y^{(k)}) + \frac{1}{2\omega} \|z^{(k-1)} - u\|_2^2 - \frac{1}{2\omega} \|z^{(k)} - u\|_2^2 \quad (4.2.10)
\end{aligned}
$$

We used convexity in ①. The definition of $x^{(k)}$ is used in ②. Inequality ③ uses convexity and Lemma 4.2.2. We applied Lemma 4.2.4 in ④. In ⑤, we substituted the value of τ, which is picked to be $\tau \overset{\text{def}}{=} \eta_k / C_k = \frac{1}{3L}$ so we can cancel $f_r(x^{(k)})$ in both sides of (4.2.10).

The choice of η_k is made so that $C_k - \eta_k = C_{k-1}$ (or equiv. $(3L-1)\eta_k = 3L\eta_{k-1}$), which allows to telescope the previous expression. Adding up (4.2.10) for $k = 1, \ldots, T$ with $u = x_r^*$, we have

$$
\left(-C_0 - \sum_{k=1}^{T} \eta_k \right) f_r(x_r^*) \leq C_0 (f_r(y^{(0)}) - f_r(x_r^*)) - C_T f_r(y^{(T)}) + \frac{1}{2\omega} \|z^{(0)} - x_r^*\|_2^2.
$$

We dropped $-\frac{1}{2\omega} \|z^{(T)} - x_r^*\|_2^2 \leq 0$. Now, since $\eta_k = C_k - C_{k-1}$ we have $-C_0 - \sum_{k=1}^{T} \eta_k = -C_T$. So reorganizing terms we obtain

$$
\begin{aligned}
f_r(y^{(T)}) &\leq f_r(x_r^*) + \frac{1}{C_T} \left(C_0 (f_r(y^{(0)}) - f_r(x_r^*)) + \frac{1}{2\omega} \|z^{(0)} - x_r^*\|_2^2 \right) \\
&\overset{①}{\leq} f_r(x_r^*) + \frac{1}{C_T} \left(C_0 (n(\log(2mn) + 1) + \frac{n \log(mn)}{2} \right) \\
&\overset{②}{\leq} f_r(x_r^*) + \varepsilon
\end{aligned} \quad (4.2.11)
$$

Above, ① uses $f_r(y^{(0)}) \leq n \log(mn/(1 - \varepsilon/n)) + \varepsilon \leq n(\log(2mn) + 1)$ and $-f_r(x_r^*) \leq 0$. For the former, take into account that $-\log(mn)\mathbb{1}_n$ is feasible and so the regularizer at $y^{(0)} = -\log(mn/(1-\varepsilon/n))\mathbb{1}_n$ is at most ε, cf. (4.2.3). Recall $\varepsilon < n/2$. We also bounded the last summand by using that $z^{(0)}, x_r^* \in B$ so $\|z^{(0)} - x_r^*\|_2^2 \leq n\omega^2$.

At this point, the only free parameters left are C_0 (via η_0) and T. We set $\eta_0 = \frac{1}{3L}$ so that

$C_0 = 1$. And we have that $C_T = 3L\eta_T = 3L\eta_0(1-\tau)^{-T} = (1-\tau)^{-T}$. So if we pick T such that

$$\frac{1}{C_T} = (1-\tau)^T \leq \frac{\varepsilon}{4n\log(2mn)}, \tag{4.2.12}$$

we will obtain ②. We will pick the smallest T that satisfies (4.2.12). That is,

$$T = \left\lceil \frac{\log(4n\log(2mn)/\varepsilon)}{\log(1/(1-\tau))} \right\rceil \leq \left\lceil 3L\log\left(\frac{4n\log(2mn)}{\varepsilon}\right) \right\rceil = \widetilde{O}(n/\varepsilon).$$

On the other hand, by definition of T as the minimum natural number satisfying (4.2.12), we have,

$$(1-\tau)^T = \frac{\eta_0}{\eta_T} \geq \frac{\varepsilon}{4n\log(2mn)}(1-\tau).$$

We can use this inequality to show $\eta_k \leq \frac{1}{4}$, for all $k \in [T]$, which is used in the proof of Lemma 4.2.4. It is enough that ① below is satisfied:

$$\eta_k \leq \eta_T \leq \frac{4n\log(2mn)\eta_0}{\varepsilon}\frac{1}{1-\tau} = \frac{4n\log(2mn)}{3L\varepsilon}\frac{3L}{3L-1} \overset{①}{\leq} \frac{1}{4}. \tag{4.2.13}$$

If $L \geq \frac{16n\log(2mn)}{3\varepsilon} + \frac{1}{3}$ then ① holds, and we chose L to satisfy this inequality.

We note we could increase T by a factor $C > 1$ inside of its log in the numerator, so that the error obtained in ② is ε/C. However, the requirement on L above would increase by a factor of C, so we would end up with an extra factor of C in the value of T. Also, the reduction caused by the smoothing already incurs in an ε additive error, cf. Proposition 4.2.1. Part 1 of Proposition 4.2.1 could use $x = \log(1-\frac{\varepsilon}{nC})\mathbb{1}_n + x^*$ so that inequality (4.2.4) ends up being bounded by $O(\varepsilon/C)$, but that would require to have β be C times smaller for (4.2.3) to work. This would also require to make L larger by a factor of C so we would also end up having the C in the total number of iterations to obtain an $O(\varepsilon)$-optimizer.

In conclusion, we obtain an ε minimizer of f_r in $\widetilde{O}(n/\varepsilon)$ iterations and by Proposition 4.2.1, we get that $\widehat{x}$ is a feasible point that is a 5ε optimizer of Problem (1FP). Finally, we note that each iteration of the algorithm can be implemented in $O(N)$ operations, that are distributed, where the bottleneck is the computation of the gradient. From the definition of the gradient, it is clear that it can be computed in our distributed model of computation and that each agent only needs their local variables for the rest of steps of the algorithm. ■

4.3 Other proofs

Lemma 4.3.1. *Let A satisfy the normalization in (4.1.1), and let $\bar{x}^*$ be the optimizer of Problem (1FP). Then $\bar{x}^*_i \geq 1/n$, for all $i \in [n]$.*

Proof The normalization ensures that e_i are feasible points, for $i \in [n]$. That is, $Ae_i \leq 1$ because each $A_{ij} \leq 1$. Since $\bar{x}^*$ is the maximizer of Problem (1FP), by the first order optimality condition we have $\langle \nabla f(\bar{x}^*), x - \bar{x}^* \rangle \leq 0$, for any feasible point x. Suppose there is a coordinate $i \in [n]$ such that $\bar{x}^*_i < \frac{1}{n}$. Then, $\langle \nabla f(\bar{x}^*), e_i - \bar{x}^* \rangle = \frac{1}{\bar{x}^*_i} - \sum_{j=1}^n \bar{x}^*_j/\bar{x}^*_j > 0$, which is a contradiction. ■

Lemma 4.3.2. *The iterates of Algorithm 7 remain in the box B.*

Proof For all $k \geq 0$, we have $z^{(k)} \in B$ by definition. If we have that $y^{(k-1)} \in B$, then $x^{(k)} \in B$ since $x^{(k)}$ is a convex combination of $y^{(k-1)}$ and $z^{(k-1)}$. So we only have to prove that for all $k \geq 0$, we have $y^{(k)} \in B$. We prove by induction that, for $k \geq 1$, it holds that $y^{(k)}$ is a convex combination of $\{z^{(i)}\}_{i=0}^{k}$ and that the weight of $z^{(k)}$ in this convex combination is $\frac{1}{\eta_k L}$. Firstly, we have $y^{(1)} = (1 - \frac{1}{\eta_1 L})z^{(0)} + \frac{1}{\eta_1 L} z^{(1)}$ (recall $x^{(0)} = z^{(0)}$). Now assuming our property holds up to $k-1$, use the definition of $y^{(k)}$ and $x^{(k)}$, to compute $y^{(k)} = \tau z^{(k-1)} + (1-\tau)y^{(k-1)} + \frac{1}{\eta_k L}(z^{(k)} - z^{(k-1)})$. This is an affine combination of the $z^{(i)}$'s, by induction hypothesis. Moreover, the weights add up to $1 = \tau + (1-\tau) + \frac{1}{\eta_k L} - \frac{1}{\eta_k L}$, and the weight on $z^{(k)}$ is $\frac{1}{\eta_k L}$. So we only have to prove the weight on $z^{(k-1)}$ is ≥ 0 in order to show that we indeed have a convex combination and not just an affine one. By induction hypothesis, we know the weight on $z^{(k-1)}$ coming from $y^{(k-1)}$ is $\frac{1}{\eta_{k-1} L}$. Hence, the weight on $z^{(k-1)}$ is $\tau + (1-\tau)\frac{1}{\eta_{k-1} L} - \frac{1}{\eta_k L} = \tau > 0$, where the equality uses the definition of η_k. ■

Lemma 4.3.3 (Lemma 3.1 in (Diakonikolas, Fazel, and Orecchia, 2020)). *Let $c \in [0,1]^n$ and let $\Delta \in \mathbb{R}^n$ be defined as $\Delta_j = -\frac{c_j \beta}{4(1+\beta)} \overline{\nabla_j f_r}(x)$, for $j \in [n]$. Then*

$$f_r(x + \Delta) - f_r(x) \leq \sum_{j=1}^{n} (1 - \frac{c_j}{2}) \Delta_j \nabla_j f_r(x).$$

Proof By using a Taylor expansion, there is a $t \in [0,1]$ such that

$$f_r(x + \Delta) - f_r(x) = \langle \nabla f_r(x), \Delta \rangle + \frac{1}{2} \Delta^\top \nabla^2 f_r(x + t\Delta) \Delta. \tag{4.3.1}$$

The gradient and Hessian of f_r are given by

$$\begin{aligned} \nabla_j f_r(x) &= -1 + \sum_{i=1}^{m} (A \exp(x))_i^{\frac{1}{\beta}} a_{ij} \exp(x_j), \\ \nabla^2_{jk} f_r(x) &= \mathbb{1}_{\{j=k\}} \sum_{i=1}^{m} (A \exp(x))_i^{\frac{1}{\beta}} a_{ij} \exp(x_j) \\ &\quad + \frac{1}{\beta} \sum_{i=1}^{m} (A \exp(x))_i^{\frac{1}{\beta}-1} a_{ij} \exp(x_j) \, a_{ik} \exp(x_k) \end{aligned} \tag{4.3.2}$$

In order to control how much the function changes, we will require $\nabla^2_{jk} f_r(x + t\Delta) \leq 2 \nabla^2_{jk} f_r(x)$. We can guarantee this condition if we guarantee that each summand in the expression above does not grow by more than a factor of 2. Let $\Delta_\ell = \max_{i \in [n]} \{\Delta_i\}$. If $\Delta_\ell < 0$, then the condition holds trivially, so we can assume $\Delta_\ell > 0$. It suffices to have $\exp(\Delta_l)^{\frac{1}{\beta}+1} \leq 2$. In other words, it suffices to have

$$\Delta_\ell \leq \frac{\ln 2}{1 + \frac{1}{\beta}}. \tag{4.3.3}$$

In fact, we will use $\Delta_j = -\frac{c_j}{4} \cdot \frac{\beta}{1+\beta} \overline{\nabla_j f_r}(x)$ for all $j \in [n]$, which satisfy the condition since $c_j \leq 1$

and $-\overline{\nabla_j f_r}(x) \leq 1$. In such a case, we have

$$\begin{aligned}
\frac{1}{2}\Delta^\top \nabla^2 f_r(x+t\Delta)\Delta &\overset{①}{\leq} \sum_{j=1}^{n}\sum_{i=1}^{m}(A\exp(x))_i^{\frac{1}{\beta}}\Delta_j^2 a_{ij}\exp(x_j) \\
&\quad + \frac{1}{\beta}\sum_{i=1}^{m}(A\exp(x))_i^{\frac{1}{\beta}-1}\left(\sum_{j=1}^{n}\Delta_j a_{ij}\exp(x_j)\right)^2 \\
&\overset{②}{\leq} \frac{\beta+1}{\beta}\sum_{j=1}^{n}\sum_{i=1}^{m}(A\exp(x))_i^{\frac{1}{\beta}}\Delta_j^2 a_{ij}\exp(x_j) \\
&\overset{③}{=} \frac{\beta+1}{\beta}\sum_{j=1}^{n}\Delta_j^2(\nabla_j f_r(x)+1) \\
&\overset{④}{=} \sum_{j=1}^{n} -\frac{c_j}{4}\Delta_j\overline{\nabla_j f_r}(x)(\nabla_j f_r(x)+1) \\
&\overset{⑤}{\leq} -\sum_{j=1}^{n}\frac{c_j}{2}\Delta_j\nabla_j f_r(x)
\end{aligned} \tag{4.3.4}$$

We used the inequality $\nabla^2_{jk} f_r(x+t\Delta) \leq 2\nabla^2_{jk} f_r(x)$ in ①. We used Cauchy-Schwarz in ② with vectors $(\sqrt{a_{i1}\exp(x_1)},\ldots,\sqrt{a_{in}\exp(x_n)})$ and $(\Delta_j\sqrt{a_{i1}\exp(x_1)},\ldots,\Delta_j\sqrt{a_{in}\exp(x_n)})$ in order to bound the last factor, so that the two first lines of the right hand side become proportional. In ③, we used the definition of the gradient. In ④, we used the value of Δ. Finally, ⑤ is a direct consequence of the truncated gradient definition (one can check the inequality for the three cases in $\nabla_j f_r(x) \in \{[-1,0),[0,1],(1,\infty)\}$, while taking into account the sign of Δ_j). Now, substituting into (4.3.1) we obtain:

$$f_r(x+\Delta) - f_r(x) \leq \sum_{j=1}^{n}\left(1-\frac{c_j}{2}\right)\Delta_j\nabla_j f_r(x).$$

■

Lemma 4.3.4 (Mirror Descent Lemma). *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a closed convex set and let $\psi : \mathcal{X} \to \mathbb{R}$ be a 1-strongly convex map with respect to $\|\cdot\|$. Let $\|\cdot\|_*$ be the dual norm to $\|\cdot\|$ and let $\ell^{(k)} \in \mathbb{R}^n$ be an arbitrary loss vector. Given $z^{(k-1)} \in \mathcal{X}$, let $z^{(k)} \overset{\text{def}}{=} \arg\min_{z\in\mathcal{X}}\{D_\psi(z, z^{(k-1)}) + \eta\langle\ell^{(k)}, z\rangle\}$. Then, for all $u \in \mathcal{X}$ we have*

a) $\eta\langle\ell^{(k)}, z^{(k-1)} - u\rangle \leq \frac{\eta^2}{2}\|\ell^{(k)}\|_*^2 + D_\psi(u, z^{(k-1)}) - D_\psi(u, z^{(k)})$.

b) $\eta\langle\ell^{(k)}, z^{(k-1)} - u\rangle \leq \frac{\eta^2}{2}\langle\ell^{(k)}, z^{(k-1)} - z^{(k)}\rangle + D_\psi(u, z^{(k-1)}) - D_\psi(u, z^{(k)})$.

Proof This proof follows the classical proof of Mirror Descent that we showed in Section 2.2.2. In fact part a) is the same and part b) is obtained by a simple modification of the bound.

We note that, by definition, we have $\frac{\partial}{\partial x} D_\psi(x,y) = \nabla\psi(x) - \nabla\psi(y)$. The lemma is due to

$$\begin{aligned}
\langle \eta\ell^{(k)}, z^{(k-1)} - u\rangle &= \langle \eta\ell^{(k)}, z^{(k-1)} - z^{(k)}\rangle + \langle \eta\ell^{(k)}, z^{(k)} - u\rangle \\
&\overset{①}{\leq} \langle \eta\ell^{(k)}, z^{(k-1)} - z^{(k)}\rangle - \langle \nabla\psi(z^{(k)}) - \nabla\psi(z^{(k-1)}), z^{(k)} - u\rangle \\
&\overset{②}{=} \langle \eta\ell^{(k)}, z^{(k-1)} - z^{(k)}\rangle - D_\psi(z^{(k)}, z^{(k-1)}) + D_\psi(u, z^{(k-1)}) - D_\psi(u, z^{(k)}) \\
&\overset{③}{\leq} \frac{\eta^2}{2}\|\ell^{(k)}\|_*^2 + D_\psi(u, z^{(k-1)}) - D_\psi(u, z^{(k)}).
\end{aligned}$$

Inequality ① comes from the first order condition of the definiton of $z^{(k)}$, that is, $\langle \nabla\psi(z^{(k)}) - \nabla\psi(z^{(k-1)}) + \eta\ell^{(k)}, u - z^{(k)}\rangle \geq 0$ for all $u \in \mathcal{X}$. ② is the triangle equality of Bregman divergences, and can be easily checked by using the definition. If we drop the term $-D_\psi(z^{(k)}, z^{(k-1)})$ after ②, we obtain part *b*) of this lemma. ③ leads to part *a*), which is the classical mirror descent lemma. It uses the bound $D_\psi(z^{(k)}, z^{(k-1)}) \geq \frac{1}{2}\|z^{(k)} - z^{(k-1)}\|^2$, which holds due to the strong convexity of ψ. And then we applied the inequality $\langle v, w\rangle - \frac{1}{2}\|w\|^2 \leq \frac{1}{2}\|v\|_*^2$ for $v, w \in \mathbb{R}^n$, that holds by Cauchy-Schwarz and $\|v\|_* \cdot \|w\| \leq \frac{1}{2}\|v\|_*^2 + \frac{1}{2}\|w\|^2$.

■

Chapter 5

Decentralized Cooperative Stochastic Bandits

In this chapter, we study a decentralized cooperative stochastic multi-armed bandit problem with K arms on a network of N agents. We study this problem under a model in which the reward distribution of each arm is the same for each agent. Moreover, rewards are drawn independently across agents and time steps, that is, there are no collisions or penalties if two agents pull from the same arm at the same iteration. In each round, each agent chooses an arm to play and subsequently sends a message to her neighbors. And the goal is to minimize the overall regret of the entire network. We design a fully decentralized algorithm that uses an accelerated consensus procedure to compute delayed estimates of the average of rewards obtained by all the agents for each arm, and then it uses an upper confidence bound (UCB) algorithm that accounts for the delay and error of the estimates. A regret analysis of our algorithm is provided. For Gaussian rewards, the regret is bounded by the optimal regret of a centralized network plus a natural and simple term depending on the spectral gap of the communication matrix. Our algorithm is simpler to analyze than those proposed in prior work, achieves better regret bounds, and performs better empirically.

5.1 Introduction

The multi-armed bandit (MAB) problem is one of the most widely studied problems in online learning. In the most basic setting of this problem, an agent has to pull one among a finite set of arms *or actions*, and she receives a reward that depends on the chosen action. This process is repeated over a finite time-horizon and the goal is to get a cumulative reward as close as possible to the reward she could have obtained by committing to the best fixed action, in hindsight. The agent only observes the rewards corresponding to the actions she chooses, i.e., the *bandit* setting as opposed to the full-information setting.

There are two main variants of the MAB problem—the stochastic and adversarial versions. In this work, our focus is on the former, where each action yields a reward that is drawn from a fixed unknown, but stationary, distribution. In the latter version, rewards may be chosen by an adversary who may be aware of the strategy employed by the agent, but cannot predict the

outcome of the random choices made by the agent. Optimal algorithms have been developed for both the stochastic and the adversarial versions, cf. (Bubeck and Cesa-Bianchi, 2012). The MAB problem epitomizes the exploration-exploitation tradeoff that appears in most online learning settings: in order to maximize the cumulative reward, it is necessary to trade off between the exploration of the hitherto under-explored arms and the exploitation of the seemingly best arm. Variants of the MAB problem are used in a wide variety of applications ranging from online advertising systems to clinical trials, queuing and scheduling.

In several applications, the "agent" solving the MAB problem may itself be a decentralized system, as in (Gai, Krishnamachari, and Jain, 2010; Tekin and Liu, 2012; Tran-Thanh, Rogers, and Jennings, 2012; Stranders et al., 2012; Buccapatnam, Eryilmaz, and Shroff, 2013; Anandkumar et al., 2011). The reason for using decentralized computation may be an inherent restriction in some cases, e.g. if we want to solve MAB problems on systems that are already decentralized, or it could be a choice made to improve the total running time—using N units allows N arms to be pulled at each time step. When the agent is a distributed system, restrictions on communication in the system introduce additional tradeoffs between communication cost and regret. Apart from the one considered in this work, there are several formulations of decentralized or distributed MAB problems, some of which are discussed in the related work section below.

Problem Formulation: This work focuses on a *decentralized* stochastic MAB problem. We consider a network consisting of N agents that play the *same* MAB problem *synchronously* for T rounds, and the goal is to obtain regret close to that incurred by an optimal centralized algorithm running for NT rounds (NT is the total number of arm pulls made by the decentralized algorithm). At each time step, all agents simultaneously pull some arm and obtain a reward drawn from the distribution corresponding to the pulled arm. The rewards are drawn independently across agents and time steps. After the rewards have been received, the agents can send messages to their neighbors.

Main Contributions: We solve the decentralized MAB problem using a *gossip* algorithm (cf. Section 5.2). Our algorithm incurs regret equal to the optimal regret in the centralized problem plus a term that depends on the spectral gap of the underlying communication graph and the number of agents (see Theorem 5.4.2 for a precise statement). At the end of each round, each agent sends $O(K)$ values to her neighbors. The amount of communication permitted can be reduced at the expense of incurring greater regret by having some delay, captured by our analysis. We assume the algorithm knows the total number of agents in the network N and a lower bound on the spectral gap of the communication matrix. We make these assumptions, which are standard in the decentralized literature (Boyd et al., 2006; Scaman et al., 2017; Duchi, Agarwal, and Wainwright, 2012; Dimakis et al., 2010), for clarity of exposition only, since N can be estimated and, for an important family of communication matrices that can be easily built in a decentralized way and that depends on the Laplacian, the spectral gap can be estimated as well (Franceschelli et al., 2013), which is enough for our purposes.

The key contribution of this work is an algorithm for the decentralized setting that exhibits a natural and simple dependence on the spectral gap of the communication matrix. In particular, for this algorithm we have

- A regret bound, that is lower compared to other algorithms previously designed for the same setting. This is achieved with the use delayed estimators of the relevant information that is communicated in order to reduce their variance.
- Standard assumptions about the global information available can be implemented on an arbitrary network.
- The use of accelerated communication, which reduces the regret dependence on the spectral gap, which is important for scalability purposes. The algorithm can also deal with stochastic communication.

5.2 Related work

Distributed Algorithms. The development of distributed algorithms for optimization and decision-making problems is active area of research, motivated in part by the recent development of large scale distributed systems that enable speeding up computations. In some cases, distributed or decentralized computation is a necessary restriction that is part of the problem, as is the case in packet routing or sensor networks. *Gossip algorithms* are a commonly used framework in this area (Boyd et al., 2006; Nedic and Ozdaglar, 2009; Shah, 2009; Dimakis et al., 2010; Duchi, Agarwal, and Wainwright, 2012; Scaman et al., 2017). In gossip algorithms, we have an iterative procedure with processing units at the nodes of a graph and the communication pattern dictated by the edges of the graph. A common sub-problem in these applications is to have a value at each node that we want to average across the network. In fact, most solutions reduce to approximate averaging. This can be achieved by using the following simple and effective method: make each node compute iteratively a weighted average of its own value and the ones communicated by its neighbors, ensuring that the final value at each node converges to the average of the initial values across the network. Formally, this can be represented as a multiplication of the vector of current estimates by a communication matrix P that respects the network structure and satisfies some conditions that guarantee fast averaging. The averaging can be accelerated by the use of Chebyshev polynomials (cf. Lemma 5.4.1). A number of works in the literature consider Chebyshev acceleration applied to gossip algorithms, e.g., (Arioli and Scott, 2014; Scaman et al., 2017).

Distributed Bandits: There are several works that study stochastic and non-stochastic distributed or decentralized multi-armed bandit problems. The precise models vary considerably. In the stochastic case, the work of Landgren, Srivastava, and Leonard (2019b); Landgren, Srivastava, and Leonard (2019a) proposes three algorithms to solve the same problem that we consider in this work coop-UCB, coop-UCB2 and coop-UCL. The algorithm coop-UCB follows a variant

of the natural approach to solve this problem that is discussed in Section 5.4. It needs to know more global information about the graph than just the number of nodes and the spectral gap: the algorithm uses a value per node that depends on the whole spectrum and the set of eigenvectors of the communication matrix. The algorithm coop-UCB2 is a modification of coop-UCB, in which the only information used about the graph is the number of nodes, and there is an exploration function $f(T)$ that allows a tradeoff from the factor multiplying $\ln T$ in the regret and the T independent summand of the regret. Finally, coop-UCL is a Bayesian algorithm that also incurs regret similar to coop-UCB2. Our algorithm obtains lower asymptotic regret, with respect to T and N, than all these algorithms while retaining the same computational complexity (cf. Remark 5.4.6).

Our work also draws on techniques on stochastic bandits with delayed feedback. There are various works that study learning with delayed feedback. The most relevant work to our problem is (Joulani, György, and Szepesvári, 2013) which studies general online learning problems under delayed feedback. Our setting differs in that we not only deal with delayed rewards but with approximations of them.

Several other variants of distributed stochastic MAB problems have been proposed. Chakraborty et al. (2017) considers the setting where at each time step, the agents can either broadcast the last obtained reward to the whole network or pull an arm. Korda, Szörényi, and Shuai (2016) studies the setting where each agent can only send information to one other agent per round, but this can be any agent in the network, not necessarily a neighbor. Szörényi et al. (2013) studies the MAB problem in P2P random networks and analyze the regret based on delayed reward estimates. Wang, Hu, et al. (2020) considers a non-decentralized version of our problem. Assuming a given time horizon, they get near optimal regret using near optimal communication. Some other works do not assume independence of the rewards drawn across the network. Liu and Zhao (2010) and Kalathil, Nayyar, and Jain (2014) consider a distributed MAB problem with collisions: if two players pull the same arm, the reward is split or no reward is obtained at all. Moreover in the latter work and a follow-up (Nayyar, Kalathil, and Jain, 2018), the act of communicating increases the regret. Anandkumar et al. (2011) also considers a model with collisions and agents have to learn from *action collisions* rather than by exchanging information. Shahrampour, Rakhlin, and Jadbabaie (2017) considers the setting where each agent plays a different MAB problem and the total regret is minimized in order to identify the best action when averaged across nodes. Nodes only send values to their neighbors but it is not a completely decentralized algorithm, since at each time step the arm played by all the nodes is given by the majority vote of the agents. Xu, Tekin, et al. (2015) studies a distributed MAB problem with global feedback, i.e., with no communication involved. Kar, Poor, and Cui (2011) also considers a different distributed bandit model in which only one agent observes the rewards for the actions she plays, while the others observe nothing and have to rely on the information broadcast by the first agent.

The problem of identifying an ε-optimal arm using a distributed network has also been studied. Hillel et al. (2013) provides matching upper and lower bounds in the case that the communication happens only once and when the graph topology is the complete graph. The authors provide an algorithm that achieves a speed up of N (the number agents) with $\log 1/\varepsilon$ communication steps.

In the adversarial version, the best possible regret bound in the centralized setting is $\sqrt{KT}$ (Audibert and Bubeck, 2009). In the decentralized case, a trivial algorithm that has no communication incurs regret $N\sqrt{KT}$; and a lower bound of $N\sqrt{T}$ is known (Cesa-Bianchi et al., 2016); thus, only the dependence on K can be improved. Awerbuch and Kleinberg (2008) study a distributed adversarial MAB problem with some Byzantine users, i.e., users that do not follow the protocol or report fake observations as they wish. In the case in which there are no Byzantine users they obtain a regret of $O(T^{2/3}(N+K)\log N \log T)$. To the best of our knowledge, this is the first work that considers a decentralized adversarial MAB problem. They allow $\log(N)$ communication rounds between decision steps so it differs with our model in terms of communication. Also in the adversarial case, Cesa-Bianchi et al. (2016) studies an algorithm that achieves regret $N(\sqrt{K^{1/2}T\log K}+\sqrt{K}\log T)$ and proves some results that are graph-dependent. The model is the same as ours, but in addition to the rewards she obtains, each agent communicates to her neighbors all the values she receives from her neighbors in the last d rounds, that is potentially $O(Nd)$, which exceeds what our model allows. They get the aforementioned regret bound by setting $d=\sqrt{K}$. Bar-On and Mansour (2019) studies the same problem and obtains a per agent regret of $\widetilde{O}(\sqrt{T(1+K/|\mathcal{N}(v)|)})$, where $|\mathcal{N}(v)|$ is the number of neighbors of agent v.

5.3 Model and problem formulation

We consider a multi-agent network with N agents. The agents are represented by the nodes of an undirected and connected graph G and each agent can only communicate to her neighbors. Agents play the same K-armed bandit problem for T time steps, send some values to their neighbors after each play and receive the information sent by their respective neighbors to use it in the next time step if they so wish. If an agent plays arm k, she receives a reward drawn from a fixed distribution with mean μ_k that is independent of the agent. The draw is independent of actions taken at previous time steps and of actions played by other agents. We assume that rewards come from subgaussian distributions with variance proxy σ^2, that is, they are random variables X satisfying $\mathbb{E}\left[\exp\left(s(X-\mathbb{E}\left[X\right])\right)\right] \leq \exp\left(\sigma^2 s^2/2\right)$. We use the notations $\mathbb{1}_d$ and $\mathbb{0}_d$ for the d-dimensional vectors of ones, and zeros, respectively. On the other hand, we use $\mathbb{1}\left(\cdot\right)$ for the indicator function, that is 1 if the argument is true and 0 otherwise.

Assume without loss of generality that $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_K$, and let the suboptimality gap be defined as $\Delta_k \stackrel{\text{def}}{=} \mu_1 - \mu_k$ for any action k. Let $I^{(t,i)}$ be the random variable that represents the action played by agent i at time t. Let $n_i^{(t,k)}$ be the number of times arm k is pulled by node

i up to time t and let $n^{(t,k)} \stackrel{\text{def}}{=} \sum_{i=1}^{N} n_i^{(t,k)}$ be the number of times arm k is pulled by all the nodes in the network up to time t. We define the regret of the whole network as

$$R(T) \stackrel{\text{def}}{=} TN\mu_1 - \mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{N} \mu_{I^{(t,i)}}\right] = \sum_{k=1}^{K} \Delta_k \mathbb{E}\left[n^{(T,k)}\right].$$

We will use this notion of regret in the entire chapter. This is sometimes known as the expected regret or pseudoregret.

The problem is to minimize the regret while allowing each agent to send $\mathsf{poly}(K)$ values to her neighbors per iteration. We allow to know only little information about the graph: the total number of nodes and a lower bound on the spectral gap of the communication matrix P, i.e., $1 - |\lambda_2|$. Here λ_2 is the second greatest eigenvalue of P in absolute value. The communication matrix can be built with little extra information about the graph, like the maximum degree of nodes of the graph (Xiao and Boyd, 2004). However, we want to avoid building global structures, like a spanning tree to propagate the information with a message passing algorithm, for instance. This is because we focus on the decentralized case, not the distributed case only. Decentralized algorithms help designing solutions for the same problem in time varying graphs or in networks prone to communication errors. We allow for stochastic communication in this work (see the discussion after Theorem 5.4.2) and we hope our results serve as a starting point for designing other algorithms for other changing graphs. Decentralized solutions are also of interest for their own sake. Bandit algorithms have a huge reach in recommendation algorithms and anything going beyond A/B testing that requires to consider a large number of options (Glowacka, 2019; Brodén et al., 2017; Mary, Gaudel, and Preux, 2015). Our algorithm applies not only to networks we can control and for which we can decide to implement a decentralized protocol in order to obtain robustness. But instead, importantly, it also applies to any network that is already decentralized by nature. There has been a surge in the implementation of decentralized systems in recent years, one can think for instance of blockchain structures (Raval, 2016) but any other system running its service in a decentralized fashion for privacy, control on data or any other reason can be considered. If a system of this kind wants to choose, for instance, what choices of their application lead to better usability, while at the same time incurring lower regret, they may decide to implement a decentralized MAB problem to obtain an answer as a community. Similarly for many other decision problems.

5.4 Algorithm

We propose an algorithm that is an adaptation of the Upper Confidence Bound algorithm (UCB) to the problem at hand and that uses a gossip protocol. We call the algorithm Decentralized Delayed Upper Confidence Bound (DDUCB). UCB is a popular algorithm for the stochastic MAB problem. At each time step, UCB computes an upper bound of a confidence interval for the mean of each arm k, using two values: the empirical mean observed, $\widehat{\mu}_k^{(t)}$, and the number of times arm k was pulled, $n^{(t,k)}$. UCB plays at time $t+1$ the arm that maximizes the upper

confidence bound $\widehat{\mu}_k^{(t)} + \sqrt{4\eta\sigma^2 \ln(t)/n^{(t,k)}}$, where $\eta > 1$ is an exploration parameter. In our setting, as the pulls are distributed across the network, agents do not have access to these two values, namely the number of times each arm was pulled across the network and the empirical mean reward observed for each arm computed using the total number of pulls. Our algorithm maintains good approximations of these values and it incurs a regret that is no more than the one for a centralized UCB plus a term depending on the spectral gap and the number of nodes, but independent of time. The latter summand is a consequence of the approximation of the mean rewards. Let $\nu^{(t,k)}$ be the sum of rewards coming from all the pulls done to arm k by the entire network up to time t. We could use a gossip protocol, for every $k \in \{1, \dots, K\}$, to obtain at each node a good approximation of $\nu^{(t,k)}$ and the number of times arm k was pulled, i.e., $n^{(t,k)}$. Let $\widehat{\nu}_i^{(t,k)}$, $\widehat{n}_i^{(t,k)}$ be the approximations of $\nu^{(t,k)}$ and $n^{(t,k)}$ made by node i with a gossip protocol at time t, respectively. Having this information at hand, agents could compute the ratio $\widehat{\nu}_i^{(t,k)}/\widehat{n}_i^{(t,k)}$ to get an estimation of the average reward of each arm. But care needs to be taken when computing the foregoing approximations.

A classical and effective way to keep a running approximation of the average of values that are iteratively added at each node is what we will refer to as the *running consensus* (Braca, Maranò, and Matta, 2008). Let $\mathcal{N}(i)$ be the set of neighbors of agent i in graph G. In this protocol, every agent stores her current approximation and performs communication and computation steps alternately: at each time step each agent computes a weighted average of her neighbors' values and adds the new value she has computed. We can represent this operation in the following way. We will work with communication matrices P as described in the following. Let $P \in \mathbb{R}^{N\times N}$ be a symmetric matrix that respects the structure of the network, which is represented by a graph G. So $P_{ij} = 0$ if there is no edge in G that connects i to j. We consider P for which the sum of each row and the sum of each column is 1, which implies that $\lambda_1 \stackrel{\text{def}}{=} 1$ is an eigenvalue of P with $\mathbb{1}_N/\sqrt{N}$ as unit eigenvector. We further assume all other eigenvalues of P, namely $\lambda_2, \dots, \lambda_N$, are less than one in absolute value, i.e., $1 = \lambda_1 > |\lambda_2| \geq \cdots \geq |\lambda_N| \geq 0$. These three conditions imply that the limit values in the network are averaged, i.e., P^s converges to $\mathbb{1}_N\mathbb{1}_N^\top/N$ for large s. See (Xiao and Boyd, 2004) for a proof and (Duchi, Agarwal, and Wainwright, 2012; Xiao and Boyd, 2004) for a discussion on how to choose P. We will refer to a matrix satisfying these three conditions as a *gossip matrix*. If we let $x^{(t)} \in \mathbb{R}^N$ denote the vector containing the current approximations for all the agents and by $y^{(t)} \in \mathbb{R}^N$ the vector containing the new values added by each node, then the running consensus protocol can be written as

$$x^{(t+1)} = Px^{(t)} + y^{(t)}. \tag{5.4.1}$$

whereas iterates of classical consensus of one vector are

$$x^{(t+1)} = Px^{(t)}. \tag{5.4.2}$$

The conditions imposed on P not only ensure that values are averaged in the limit but also

that the averaging process is fast. In particular, for any $s \in \mathbb{N}$ and any $v \in \mathbb{R}^N$, we have

$$\|P^s v - \mathbb{1}_N \frac{\sum_i v_i}{N}\|_2 = \|P^s v - \mathbb{1}_N \mathbb{1}_N^\top v/N\|_2 \leq |\lambda_2|^s \|v\|_2 \tag{5.4.3}$$

because $P^s - \mathbb{1}_N \mathbb{1}_N^\top/N$ has eigenvalues $\{0, \lambda_2^s, \ldots, \lambda_N^s\}$. A natural approach to the problem is to use $2K$ running consensus algorithms to compute approximations of $\nu^{(t,k)}/N$ and $n^{(t,k)}/N$, $k = 1, \ldots, K$. Landgren, Srivastava, and Leonard (2019b) follow this approach and use extra global information of the graph, as described in the section on related work, to account for the inaccuracy of the mean estimate. We can estimate average rewards by their ratio and the number of times each arm was pulled can be estimated by multiplying the quantity $n^{(t,k)}/N$ by N. The running consensus protocols would be the following. For $k = 1, \ldots, K$, start with $\widehat{\nu}^{(1,k)} = \mathbb{0}_N$ and update $\widehat{\nu}^{(t+1,k)} = P\widehat{\nu}^{(t,k)} + \pi^{(t,k)}$, where the *i-th* entry of $\pi^{(t,k)} \in \mathbb{R}^N$ contains the reward observed by node i at time t if arm k is pulled. Else, it is 0. Note that the *i-th* entry is only computed by the *i-th* node. Similarly, for $k = 1, \ldots, K$, start with $\widehat{n}^{(1,k)} = \mathbb{0}_N$ and update $\widehat{n}^{(t+1,k)} = P\widehat{n}^{(t,k)} + p^{(t,k)} \in \mathbb{R}^N$, where $p_i^{(t,k)}$ is 1 if at time t node i pulled arm k and 0 otherwise.

The problem with this approach is that even if the values computed are being mixed at a fast pace it takes some time for the last added values to be mixed, resulting in poor approximations, especially if N is large. This phenomenon is more intense when the spectral gap is smaller. Indeed, we can rewrite (5.4.1) as $x^{(t)} = \sum_{s=1}^{t-1} P^{t-1-s} y^{(s)}$, assuming that $x^{(1)} = 0$. For the values of s that are not too close to $t-1$ we have by (5.4.3) that $P^{t-1-s} y^{(s)}$ is very close to the vector that has as entries the average of the values in $y^{(s)}$, that is, $\left(\frac{1}{N}\sum_{j=1}^N y_j^{(s)}\right)\mathbb{1}_N$. However, for values of s close to $t-1$ this is not true and the value of $y^{(s)}$ heavily influences the resulting estimate, being specially inaccurate as an estimation of the true mean if N is large. The key observations that lead to the algorithm we propose are that the number of the values of s *close to* $t-1$ is small, as we quantify in the sequel, that we can make it even smaller using accelerated gossip techniques and that the regret of UCB does not increase much when working with delayed values of rewards so we can temporarily ignore the recently computed rewards in order to work with much more accurate approximations of $\nu^{(t,k)}/N$ and $n^{(t,k)}/N$. In particular, with C communication steps agents can compute a polynomial q_C of degree C of the communication matrix P applied to a vector, that is, $q_C(P)v$. The acceleration comes from computing a rescaled Chebyshev polynomial and it is encapsulated in the following lemma. This approach was used in previous works (Scaman et al., 2017).

Lemma 5.4.1. *Let P be a gossip matrix. Let $v \in \mathbb{R}^N$ and let $C = \lceil \frac{\ln(2N/\varepsilon)}{\sqrt{2\ln(1/|\lambda_2|)}} \rceil$. Agents can compute, after C communication steps, the value $q_C(P)v$ for a polynomial q_C of degree C which satisfies $\|q_C(P)v - \mathbb{1}_N \mathbb{1}_N^\top v/N\|_2 \leq \varepsilon/N \|v\|_2$.*

Proof This Chebyshev approach comes from the first accelerated algorithms that minimize a convex quadratic, cf. Section 2.3. Define the Chebyshev polynomials of the first kind as $T_0(t) = 1$, $T_1(t) = t$ and $T_r(t) = 2tT_{r-1}(t) - T_{r-2}(t)$ for $r > 1$. Then, define the rescaled

Chebyshev polynomial

$$q_r(t) = \frac{T_r(t/|\lambda_2|)}{T_r(1/|\lambda_2|)}.$$

Let $\kappa = \frac{1+|\lambda_2|}{1-|\lambda_2|}$, and $C = \lceil \frac{\ln(2N/\varepsilon)}{\sqrt{2\ln(1/|\lambda_2|)}} \rceil$. Then for any $t \in [-|\lambda_2|, |\lambda_2|]$ the polynomial q_C satisfies:

$$q_C(t) \overset{①}{\leq} 2\frac{\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^C}{1+\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{2C}} < 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^C \overset{②}{\leq} 2\exp\left(-\log(2N/\varepsilon)\right) \leq \frac{\varepsilon}{N}.$$

See (Auzinger and Melenk, 2011) for ①. Inequality ② is true for $x \in [0,1)$ since

$$\exp\left(1/\sqrt{2\ln(1/x)}\ln\left(1+\frac{-2}{\sqrt{(1+x)/(1-x)}+1}\right)\right) \leq e^{-1},$$

because the expression is monotone its $\lim_{x\to 1^-}$ of it is e^{-1}. We also have $q_C(1) = 1$. This implies that the absolute value of all the eigenvalues of the matrix $q_C(P)$ is less than $\frac{\varepsilon}{N}$ but for the greatest one, which is 1. The previous property implies $\|q_C(P) - \frac{1}{N}\mathbb{1}_N\mathbb{1}_N^\top\|_2 \leq \frac{\varepsilon}{N}$, because the matrix $q_C(P)$ is diagonalizable by orthogonality and so it splits as a sum of weighted outer products of its weighted eigenvectors. Finally, we have

$$\|q_C(P)v - \frac{1}{N}\mathbb{1}_N\mathbb{1}_N^\top v\|_2 \leq \|q_C(P) - \frac{1}{N}\mathbb{1}_N\mathbb{1}_N^\top\|_2\|v\|_2 \leq \frac{\varepsilon}{N}\|v\|_2.$$

Note that by definition $q_r(P)$ can be computed iteratively as

$$\omega^{(r+1)}q_{r+1}(P) = \frac{2}{|\lambda_2|}\omega^{(r)}Pq_r(P) - \omega^{(r-1)}q_{r-1}(P), \tag{5.4.4}$$

for $r \geq 1$ where $\omega^{(r)} = T_r(1/|\lambda_2|)$. Again by the definition of the Chebyshev polynomial, $\omega^{(r)}$ can be computed iteratively as $\omega^{(0)} = 1$, $\omega^{(1)} = 1/|\lambda_2|$ and $\omega^{(r+1)} = 2\omega^{(r)}/|\lambda_2| - \omega^{(r-1)}$ for $r > 1$. Also note that if we have a vector $u \in \mathbb{R}^N$ we can slightly modify the recursion in (5.4.4) to compute $q_C(P)u$ using the gossip protocol C times:

$$y^{(r+1)} = \frac{\omega^{(r)}}{\omega^{(r+1)}}\frac{2}{|\lambda_2|}Py^{(r)} - \frac{\omega^{(r-1)}}{\omega^{(r+1)}}y^{(r-1)}, \tag{5.4.5}$$

for $r > 1$, where we denote $y^{(r)} = q_r(P)u \in \mathbb{R}^N$. ■

For a given a vector u we want to mix, it holds that coordinate j of $q_C(P)u$, which is the value held by node j, is a weighted sum of u_i, $1 \leq i \leq N$ where weights $q_C(P)_{i,j}$ are close to $1/N$, since applying the lemma to $v \in \{e_1, \ldots, e_N\}$ we obtain that $|q_C(P)_{i,j} - \frac{1}{N}| \leq \varepsilon/N$. Analogously, it can be done with $P^s u$, coming from (5.4.2), for s such that $|\lambda_2|^s \leq \varepsilon/N$. At iteration r of either protocol (5.4.2) or (5.4.5), agent i only computes the i-th entry of $P^r u$ or $q_r(P)u$ and only uses her own information and the weighted information sent by her neighbors, i.e., at most one multiplication by P is allowed. We can use either method to approximate $\nu^{(t,k)}/N$ and $n^{(t,k)}/N$.

We now describe DDUCB at node i. The pseudocode is given in Algorithm 9. We use Greek letters to denote variables that contain rewards estimators, and corresponding Latin letters to

Algorithm 8 Accelerated communication and mixing step. $\text{mix}(y_i^{(r)}, r, i)$

if r is 0 **then**
  $\omega^{(0)} \leftarrow 1/2$; $\omega^{(-1)} \leftarrow 0$
  $y_i^{(0)} \leftarrow y_i^{(0)}/2$; $y_i^{(-1)} \leftarrow (0, \ldots, 0)$
end if
Send $y_i^{(r)}$ to neighbors
Receive corresponding values $y_j^{(r)}, \forall j \in \mathcal{N}(i)$
$\bar{y}_i^{(r)} \leftarrow \sum_{j \in \mathcal{N}(i)} 2P_{ij} y_j^{(r)}/|\lambda_2|$
$\omega^{(r+1)} \leftarrow 2\omega^{(r)}/|\lambda_2| - \omega^{(r-1)}$
$y_i^{(r+1)} = \frac{\omega^{(r)}}{\omega^{(r+1)}} \bar{y}_i^{(r)} - \frac{\omega^{(r-1)}}{\omega^{(r+1)}} y_i^{(r-1)}$
if r is 0 **then**
  $y_i^{(0)} \leftarrow 2y_i^{(0)}$; $\omega^{(0)} \leftarrow 2\omega^{(0)}$
end if
return $y_i^{(r+1)}$

denote variables that contain counter estimators. Agents run an accelerated running consensus in stages of C iterations. Each node maintains three pairs of K-dimensional vectors. The variable $\alpha^{(i)}$ contains rewards that are already mixed, $\beta^{(i)}$ contains rewards that are being mixed and $\gamma^{(i)}$ contains rewards obtained in the current stage. The vectors $a^{(i)}$, $b^{(i)}$ and $c^{(i)}$ store the number of arm pulls associated to the quantities $\alpha^{(i)}$, $\beta^{(i)}$ and $\gamma^{(i)}$, respectively. At the beginning, agent i pulls each arm once and initialize $\alpha^{(i)}$ and $a^{(i)}$ with the observed values divided by N, since the these variables are designed to contain average estimations of their respective quantities. During each stage, for C iterations, agent i uses $\alpha^{(i)}$ and $a^{(i)}$, as updated at the end of the previous stage, to decide which arm to pull using an upper confidence bound. Variables $\beta^{(i)}$ and $b^{(i)}$ are being mixed in an accelerated way and $\gamma^{(i)}$ and $c^{(i)}$ are added new values obtained by the new pulls done in the current stage. After C iterations, values in $\beta^{(i)}$ and $b^{(i)}$ are mixed enough so we add them to $\alpha^{(i)}$ and $a^{(i)}$. The only exception being the end of the first stage in which the values of the latter variables are overwritten by the former ones. Variables $\delta^{(i)}$ and $d^{(i)}$ just serve to make this distinction. The unmixed information about the pulls obtained in the last stage, i.e., $\gamma^{(i)}$ and $c^{(i)}$, is assigned to $\beta^{(i)}$ and $b^{(i)}$ so the process can start again. Variables $\gamma^{(i)}$ and $c^{(i)}$ are reset with zeroes. There are T iterations in total.

Now we describe some mathematical properties about the variables during the execution of the algorithm. Let t_S be the time at which a stage begins, so it ends at $t_S + C - 1$. At $t = t_S$, using the notation above, it is $\alpha_k^{(i)} = \sum_{s=1}^{t_S - C} \left(q_C(P)\pi^{(s,k)}\right)_i$ and $a_k^{(i)} = \sum_{s=1}^{t_S - C} \left(q_C(P)p^{(s,k)}\right)_i$ but in the first stage, in which their values are initialized from a local pull. In particular, let $X_1^{(i)}, \ldots, X_K^{(i)}$ denote the rewards obtained when pulling all the arms before starting the first stage. Then the initialization is $\alpha^{(i)} \leftarrow (X_1^{(i)}/N, \ldots, X_K^{(i)}/N)$ and $a^{(i)} \leftarrow (1/N, \ldots, 1/N)$. The division by N is due to $\alpha_k^{(i)}$ and $a_k^{(i)}$ being the approximations of $\nu_i^{(t,k)}/N$ and $n_i^{(t,k)}/N$. The algorithm does not update $\alpha^{(i)}$ and $a^{(i)}$ again until $t = t_S + C$, so they contain information that at the end of the stage is delayed by $2C - 1$ iterations. The time step s used to compute the upper confidence bound is $(t_S - C)N$, since $\alpha^{(i)}$ and $a^{(i)}$ contain information about that number

Algorithm 9 Decentralized Delayed UCB at node i (and some variants, cf. Remark 5.4.7). "mix" is given by (5.4.2) (no acceleration) or Algorithm 8 (acceleration)

$\zeta^{(i)} \leftarrow (X_1^{(i)}, \dots, X_K^{(i)})$; $z^{(i)} \leftarrow \mathbb{1}_K$
$C \leftarrow$ (5.4.6)
$\alpha^{(i)} \leftarrow \zeta^{(i)}/N$; $a^{(i)} \leftarrow z^{(i)}/N$; $\beta^{(i)} \leftarrow \zeta^{(i)}$; $b^{(i)} \leftarrow z^{(i)}$
$\gamma^{(i)} \leftarrow \mathbb{0}_K$; $c^{(i)} \leftarrow \mathbb{0}_K$; $\delta^{(i)} \leftarrow \mathbb{0}_K$; $d^{(i)} \leftarrow \mathbb{0}_K$
$t \leftarrow K$; $s \leftarrow K$
while $t \le T$ **do**
  for $r = 0$ **to** $C - 1$ **do**
    $k^* \leftarrow \arg\max_{k \in \{1,\dots,K\}} \left\{ \frac{\alpha_k^{(i)}}{a_k^{(i)}} + \sqrt{\frac{2\eta\sigma^2 \ln s}{N a_k^{(i)}}} \right\}$
    $u \leftarrow$ Play arm k^*, return reward
    $\gamma_{k^*}^{(i)} \leftarrow \gamma_{k^*}^{(i)} + u$; $c_{k^*}^{(i)} \leftarrow c_{k^*}^{(i)} + 1$
    $\beta^{(i)} \leftarrow \mathrm{mix}(\beta^{(i)}, r, i)$; $b^{(i)} \leftarrow \mathrm{mix}(b^{(i)}, r, i)$
    $t \leftarrow t + 1$ ◇ It also works if we use 13-14 and/or 15
    // $\alpha_{k^*}^{(i)} \leftarrow \alpha_{k^*}^{(i)} + u/N$; $a_{k^*}^{(i)} \leftarrow a_{k^*}^{(i)} + 1/N$
    // $s \leftarrow s + 1$
    // $\delta^{(i)} \leftarrow \mathrm{unaccel_mix}(\delta^{(i)}, i)$; $d^{(i)} \leftarrow \mathrm{unaccel_mix}(d^{(i)}, i)$ ◇ cf. (5.4.2)
    if $t > T$ **then return end if**
  end for
  $s \leftarrow (t - C)N$
  $\delta^{(i)} \leftarrow \delta^{(i)} + \beta^{(i)}$; $d^{(i)} \leftarrow d^{(i)} + b^{(i)}$; $\alpha^{(i)} \leftarrow \delta^{(i)}$; $a^{(i)} \leftarrow d^{(i)}$
  $\beta^{(i)} \leftarrow \gamma^{(i)}$; $b^{(i)} \leftarrow c^{(i)}$; $\gamma^{(i)} \leftarrow \mathbb{0}_K$; $c^{(i)} \leftarrow \mathbb{0}_K$
end while

of rewards and pulls. The variable $\gamma^{(i)}$ is needed because we need to mix $\beta^{(i)}$ for C steps so the Chebyshev polynomial of degree C is computed and meanwhile we store the new rewards by adding them to $\gamma^{(i)}$. In this way agents compute upper confidence bounds with accurate approximations, with a delay of at most $2C - 1$. As we will see, the regret of UCB does not increase much when working with delayed estimates. In particular, having a delay of $\mathfrak{d}$ steps increases the regret by at most $\mathfrak{d} \sum_{k=1}^K \Delta_k$.

We now present the regret which the DDUCB algorithm incurs. We use $A \lesssim B$ to denote there is a constant $c > 0$ such that $A \le cB$.

Theorem 5.4.2 (Regret of DDUCB). *Let P be a gossip matrix. Consider the distributed multi-armed bandit problem with N nodes, K actions and subgaussian rewards with variance proxy σ^2. The algorithm DDUCB satisfies:*

$$R(T) \lesssim \sum_{k:\Delta_k > 0} \frac{\eta(1+\varepsilon)\sigma^2 \ln(TN)}{\Delta_k} + \left(NC + \frac{\eta}{\eta - 1} \right) \sum_{k=1}^K \Delta_k.$$

Here, $\eta > 1$ is an exploration parameter, $\varepsilon \in (0, \frac{\eta-1}{7(\eta+1)})$ is an accuracy parameter. For instance, we can set $\eta = 2$, $\varepsilon = 1/22$ so the regret is

$$R(T) \lesssim \sum_{k:\Delta_k > 0} \frac{\sigma^2 \ln(TN)}{\Delta_k} + NC \sum_{k=1}^K \Delta_k.$$

And this choice of parameters incurs small constants only. The parameter

$$C = \left\lceil \frac{\ln(N/\varepsilon)}{\ln(1/|\lambda_2|)} \right\rceil \text{ or } C = \left\lceil \frac{\ln(2N/\varepsilon)}{\sqrt{2\ln(1/|\lambda_2|)}} \right\rceil \tag{5.4.6}$$

is a delay parameter for the unaccelerated (5.4.2) *and accelerated* (5.4.4) *mixing protocols, respectively.*

Note the constant C, which is a parameter that indicates when values are close enough to be mixed, requires the value $|\lambda_2|$. However, if we use DDUCB with delay parameter set to any upper bound E of the suggested value C, then the inequality above still holds true, if we substitute C by E. So having an upper bound on $|\lambda_{22}|$ is enough. The accelerated protocol improves in the harder regime, when the spectral gap is close to 1, as long as $|\lambda_2| > 1/e$, i.e., whenever $1/\ln^{1/2}|\lambda_2^{-1}| < 1/\ln|\lambda_2^{-1}|$. The knowledge of the spectral gap is a standard assumption in decentralized literature (Scaman et al., 2017; Duchi, Agarwal, and Wainwright, 2012; Dimakis et al., 2010) and there are works that show how to estimate it (Franceschelli et al., 2013). Other gossip protocols different from (5.4.2) and (5.4.4) can be used, as long as they guarantee deviation from the average $\mathbb{1}_N\mathbb{1}_N^\top v/N$ lower than $\|v\|_2\varepsilon/N$ after C steps, for a vector $v \in \mathbb{R}^N$. For instance, for some graphs it can be convenient to use the accelerated protocol in (Berthier, Bach, and Gaillard, 2020). Decentralized algorithms like ours are also useful to deal with problems on time-varying graphs or on networks prone to communication errors. In particular, stochastic gossip algorithms also come guarantees with high probability. For instance if we have stochastic i.i.d. communication matrices $P(t)$ that are symmetric, then with $C = \lceil \log(N^2/(\varepsilon^2\delta))/\log(\lambda_2(\mathbb{E}\left[P(t)^2\right])^{-1})\rceil$ and $v \in \mathbb{R}^N$ it is $\|\prod_{i=r}^{r+C} P(i)v - \mathbb{1}_N\mathbb{1}_N^\top v/N\| \leq \varepsilon\|v\|_2/N$ with probability at least $1-\delta$ (Duchi, Agarwal, and Wainwright, 2012; Loizou and Richtárik, 2019). So (5.4.2) can be used in this case to have an algorithm that satisfies the inequality with high probability for every stage. See (Loizou and Richtárik, 2019) and references therein for more on stochastic gossip algorithms. Note that communication rules independent of the time step, as in the unaccelerated running consensus, allow for the possibility that the mixed values are mixed throughout the whole execution of the algorithm, i.e., we can update $\beta^{(i)}$ and $b^{(i)}$ as $\beta^{(i)} \leftarrow \beta^{(i)} + \gamma^{(i)}$ and $b^{(i)} \leftarrow b^{(i)} + c^{(i)}$ (in such a case we need to use $\alpha^{(i)} \leftarrow \beta^{(i)}$ and $a^{(i)} \leftarrow b^{(i)}$ too). This provides more accurate estimations of the means. Our accelerated protocol is not stationary, but one can use an *asymptotically* optimal and stationary gossip protocol (cf. (Liu, Anderson, et al., 2013; Berthier, Bach, and Gaillard, 2020)) for regimes of spectral gap close to 1.

The proof of Theorem 5.4.2 is along the lines of the one for the standard UCB algorithm cf. (Auer, Cesa-Bianchi, and Fischer, 2002) but requires a couple of key modifications. Firstly, we need to control the error due to the fact that each agent decides with some delay which arm to pull, because only mixed information is is used. Secondly, we need to control the error due to agents only having approximations of $\nu^{(t,k)}$ and $n^{(t,k)}$, that is, to the true sum of rewards and number of times each arm was pulled respectively.

We present two lemmas before the proof. As explained after Lemma 5.4.1, one can think about each reward as being added at each node weighted by a number. The weights are entries of $q_C(P)$ or P^C and they approach $1/N$ quickly.

Lemma 5.4.3 (Concentration). *Let $Y_1, \dots, Y_D$ be independent random variables coming from the distribution associated to an arm k, which we assume to be subgaussian with variance proxy σ^2 and mean μ_k. Let $\varepsilon \in (0, \frac{\eta-1}{7(\eta+1)})$, for $\eta > 1$, and let $s > 1$. Let w_j be a number such that $|w_j - \frac{1}{N}| < \varepsilon/N$, for all $j = 1, \dots, D$. Then*

$$\mathbb{P}\left[\xi \cdot \left(\frac{\sum_{j=1}^D w_j Y_j}{\sum_{j=1}^D w_j} - \mu_k\right) \geq \sqrt{\frac{4\eta\sigma^2 \ln s}{N \sum_{j=1}^D w_j}}\right] \leq \frac{1}{s^{\eta+1}}$$

where $\xi \in \{-1, 1\}$.

Proof Since Y_j is subgaussian with variance proxy σ^2 we have that $w_j Y_j / (\sum w_j)$ is subgaussian with variance proxy $w_j^2 \sigma^2 / (\sum_j w_j)^2$. Therefore, using subgaussianity and the fact that the random variables Y_j, for $j = 1, \dots, D$, are independent we can bound the left hand side by Hoeffding's inequality for subgaussian random variables:

$$\exp\left(-\frac{(4\eta\sigma^2 \ln s)/(N \sum w_j)}{2\sigma^2 \sum w_j^2 / (\sum w_j)^2}\right) = \frac{1}{s^{2\eta/(NW)}},$$

where $W \stackrel{\text{def}}{=} \sum w_j^2 / \sum w_j$. Using $\left|w_j - \frac{1}{N}\right| < \varepsilon/N$ we obtain

$$\eta/(NW) = \eta\left(N \frac{\sum w_j^2}{\sum w_j}\right)^{-1} \geq \eta\left(N \frac{D((1+\varepsilon)/N)^2}{D((1-\varepsilon)/N)}\right)^{-1} = \frac{\eta(1-\varepsilon)}{(1+\varepsilon)^2} > \frac{\eta+1}{2}.$$

The last step is a consequence of $\varepsilon < \frac{\eta-1}{7(\eta+1)}$. The result follows. ■

At time t and at node i, we want to use the variables $\alpha^{(t,i)}$ and $a^{(t,i)}$, which we use to denote the value of $\alpha^{(i)}$ and $a^{(i)}$ from Algorithm 9 at time t, to decide the next arm to pull at that node. Consider the rewards computed by all the nodes until C steps before the last time $\alpha^{(i)}$ and $a^{(i)}$ were updated. That is, the rewards whose sum is approximated by $N\alpha^{(i)}$. Let $J_k^{(t)}$ be the number of these rewards that come from arm k and let $X_k^{(j)}, 1 \leq j \leq J_k^{(t)}$ be such rewards sorted by the round they were obtained and, in case of a tie, by node index. Note this agrees with our notation for the pulls in the initialization of Algorithm 9. With this notation, we have in node i that $\alpha_k^{(t,i)}$ is the sum of each of the $X_k^{(j)}$ multiplied by a weight $w_k^{(t,i,j)}$, where the sum of all the weights of all the nodes is $\sum_i w_k^{(t,i,j)} = 1$. Indeed, by construction it is $\alpha_k^{(t,i)} = \sum_{j \in J_k^{(t)}} P_{i,j}^C X_k^{(j)}$ or $\alpha_k^{(t,i)} = \sum_{j \in J_k^{(t)}} q_C(P)_{i,j} X_k^{(j)}$, depending on the communication protocol, then $w_k^{(t,i,j)}$ corresponds to an entry of P^C or $q_C(P)$, which are doubly stochastic matrices. All rewards considered have been mixing for C steps. This ensures $\left|w_k^{(t,i,j)} - \frac{1}{N}\right| < \frac{\varepsilon}{N}$ by (5.4.3) and Lemma 5.4.1, so the previous lemma can be applied to these weights. Define the empirical mean of arm k at node i

and time t as

$$\widehat{\mu}_k^{(t,i)} \stackrel{\text{def}}{=} \frac{\alpha_k^{(t,i)}}{a_k^{(t,i)}} = \frac{\sum_{j=1}^{J_k^{(t)}} w_k^{(t,i,j)} X_k^{(j)}}{\sum_{j=1}^{J_k^{(t)}} w_k^{(t,i,j)}}.$$

Let $\text{UCB}(t,s,k,i) \stackrel{\text{def}}{=} \widehat{\mu}_k^{(t,i)} + \sqrt{\frac{4\eta\sigma^2 \ln s}{N\sum_j w_k^{(t,i,j)}}}$ and let $I^{(t,i)}$ be the random variable that represents the arm pulled at time t by node i, which is the one that maximizes $\text{UCB}(t,s,k,i)$, for a certain value s.

Lemma 5.4.4. *Let k be a suboptimal arm. We have*

$$\mathbb{P}\left(I^{(t,i)} = k, N\sum_{j=1}^{J_k^{(t)}} w_k^{(t,i,j)} > \frac{16\eta\sigma^2 \ln s}{\Delta_k^2}\right) \leq \frac{2Ns^{(t)}}{s^{\eta+1}},$$

where $s^{(t)} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{k=1}^K J_k^{(t)}$, and where $Ns^{(t)}$ corresponds to the number of rewards obtained by all the nodes until C steps before the last time $\alpha^{(i)}$ and $a^{(i)}$ were updated.

Proof Recall that $\mu_1 \geq \mu_k$ for any $k \in [K]$. It is enough to bound $\mathbb{P}\left(\text{UCB}(t,s,1,i) \leq \mu_1\right)$ and $\mathbb{P}\left(\widehat{\mu}_k^{(t,i)} > \mu_k + \sqrt{\frac{4\eta\sigma^2 \ln s}{N\sum w_j}}\right)$ and use the union bound, since if these two events are false we can apply ① and ③ in the following and obtain $I^{(t,i)} \neq k$:

$$\begin{aligned}\text{UCB}(t,s,k,i) = \widehat{\mu}_k^{(t,i)} + \sqrt{\frac{4\eta\sigma^2 \ln s}{N\sum_j w_k^{(t,i,j)}}} &\overset{①}{\leq} \mu_k + 2\sqrt{\frac{4\eta\sigma^2 \ln s}{N\sum_j w_k^{(t,i,j)}}} \overset{②}{<} \mu_k + \Delta_k \\ &= \mu_1 \overset{③}{<} \text{UCB}(t,s,1,i).\end{aligned}$$

Inequality ② is equivalent to $N\sum_j w_k^{(t,i,j)} > \frac{16\eta\sigma^2 \ln s}{\Delta_k^2}$.

Now, since $1 \leq J_k^{(t)} \leq Ns^{(t)}$ we have by the union bound and Lemma 5.4.3

$$\begin{aligned}\mathbb{P}(\text{UCB}(t,s,1,i) \leq \mu_1) &\leq \mathbb{P}\left(\exists \ell \in \{1,\dots,Ns^{(t)}\} : J_k^{(t)} = \ell, \text{UCB}(t,s,1,i) \leq \mu_1\right) \\ &\leq \sum_{\ell=1}^{Ns^{(t)}} \mathbb{P}\left(\text{UCB}(t,s,1,i) \leq \mu_1 | J_k^{(t)} = \ell\right) \leq \sum_{\ell=1}^{Ns^{(t)}} \frac{1}{s^{\eta+1}} = \frac{Ns^{(t)}}{s^{\eta+1}}.\end{aligned}$$

The bound of $\mathbb{P}\left(\widehat{\mu}_k^{(t,i)} > \mu_k + \sqrt{\frac{4\eta\sigma^2 \ln s}{N\sum w_j}}\right)$ is analogous. ■

Now we proceed to prove the theorem.

Proof of Theorem 5.4.2. For every $t \geq K$ we can write t uniquely as $K + C\rho_t + r_t$, where $\rho_t \geq 0$ and $0 \leq r_t < C$. In such a case it is

$$s^{(t)} = K\max\left(\mathbb{1}\left(\rho_t > 0\right), 1/N\right) + C(\rho_t - 1)\mathbb{1}\left(\rho_t > 1\right),$$

where $s^{(t)}$ is defined in Lemma 5.4.4. The time step s that we use to compute the upper confidence bounds at time t is $s = Ns^{(t)}$. It is fixed every C iterations. For $t \geq K + C$, the value $s^{(t)} + C$ is equal to the last time step in which the variables $\alpha^{(i)}$ and $a^{(i)}$ were updated. Thus, by definition $J_k^{(t)} = n^{(s^{(t)},k)}$. Remember $n_i^{(t,k)}$ is the number of times arm k is pulled by node i up to time

t, and $n^{(t,k)} = \sum_{i=1}^{N} n_i^{(t,k)}$. Since $R(T) = \sum_{k=1}^{K} \Delta_k \mathbb{E}[n^{(T,k)}]$, it is enough to bound $\mathbb{E}[n^{(T,k)}]$ for every $k = 1, \dots, K$.

Let k be fixed and denote the event $\{I^{(t,i)} = k\}$ by $A^{(t,i)}$. We have

$$
\begin{aligned}
\mathbb{E}[n^{(T,k)}] &= N + \mathbb{E}\left[\sum_{i=1}^{N}\sum_{t=K+1}^{T} \mathbb{1}\left(A^{(t,i)}\right)\right] \\
&= N + \mathbb{E}\left[\sum_{i=1}^{N}\sum_{t=K+1}^{T} \mathbb{1}\left(A^{(t,i)}, 1 \le \frac{16\eta\sigma^2 \ln(s^{(t)}N)}{N\sum_j w_k^{(t,i,j)}\Delta_k^2}\right)\right] \\
&+ \mathbb{E}\left[\mathbb{1}\left(A^{(t,i)}, 1 > \frac{16\eta\sigma^2 \ln(s^{(t)}N)}{N\sum_j w_k^{(t,i,j)}\Delta_k^2}\right)\right] \\
&\overset{①}{\le} N + \mathbb{E}\left[\sum_{i=1}^{N}\sum_{t=K+1}^{T} \mathbb{1}\left(A^{(t,i)}, n^{(s^{(t)},k)} \le \frac{16\eta\sigma^2 \ln(TN)}{\Delta_k^2/(1+2\varepsilon)}\right)\right] + N\sum_{t=K+1}^{T}\frac{2}{(s^{(t)}N)^\eta} \\
&\overset{②}{<} N + \frac{16\eta\sigma^2\ln(TN)}{\Delta_k^2/(1+2\varepsilon)} + 2NC + \frac{2NC}{K^\eta}\left(1+\frac{1}{N^\eta}\right) + \frac{2}{(NC)^{\eta-1}}\sum_{r=\lfloor K/C\rfloor+1}^{\infty}\frac{1}{r^\eta} \\
&\overset{③}{\lesssim} \frac{\eta\sigma^2\ln(TN)}{\Delta_k^2/(1+\varepsilon)} + NC + \frac{\eta}{\eta-1}.
\end{aligned}
$$

For the bound of the first summand in ① note that $\left|w_k^{(t,i,j)} - \frac{1}{N}\right| < \varepsilon/N$ and that $J_k^{(t)} = n^{(s^{(t)},k)}$. Thus,

$$
\left(N\sum_j w_k^{(t,i,j)}\right)^{-1} \le \left(n^{(s^{(t)},k)}(1-\varepsilon)\right)^{-1} \le (1+2\varepsilon)/n^{(s^{(t)},k)}.
$$

We have used $\varepsilon < 1/2$ for the last step, which is a consequence of $\varepsilon < \frac{\eta-1}{7(\eta+1)}$ for $\eta > 1$. The bound for the second summand uses Lemma 5.4.4. For the bound of the expectation in ②, note that, by definition, $A^{(t,i)}$ for $1 \le t \le T$ can only happen $n^{(t,k)}$ times but $n^{(t,k)} \le n^{(s^{(t)},k)} + N(t - s_t) \le n^{(s^{(t)},k)} + 2NC$. So $\mathbb{1}\left(I_{t,i+1} = k, n^{(s^{(t)},k)} < \frac{16\eta\sigma^2\ln(TN)}{\Delta_k^2/(1+2\varepsilon)}\right)$ can be 1 at most $\frac{6\eta\sigma^2\ln(TN)}{\Delta_k^2/(1+2\varepsilon)} + 2NC$ times. The term $2NC$ accounts for the delay of the algorithm. In the second part of inequality ② we substitute $s^{(t)}$ by its value and for $t > K + 2C$ we bound it by the greatest multiple of C that is less than $s^{(t)}$. For ③, note that the sum over r is bounded by $\zeta(\eta)$, where $\zeta(\cdot)$ is the Riemann zeta function. Then we use $\zeta(x) < \frac{x}{x-1}$ for all $x > 1$, cf. (Ireland and Rosen, 1982), Proposition 16.1.2. This yields the bound. We bounded $1/N^\eta$, $1/K^\eta$ and $1/(NC)^{\eta-1}$ by 1. ■

Remark 5.4.5 (Lower bounds). *In order to interpret the regret obtained in the previous theorem, assuming Gaussian rewards in this remark, it is useful to note that running the centralized UCB algorithm for TN steps incurs a regret bounded above by $\sum_{k:\Delta_k>0}\frac{\sigma^2\ln(TN)}{\Delta_k} + \sum_{k=1}^{K}\Delta_k$, up to a constant. Moreover, running N separate instances of UCB at each node without allowing communication incurs a regret of $R(T) \lesssim \sum_{k:\Delta_k>0}\frac{N\sigma^2\ln(T)}{\Delta_k} + N\sum_{k=1}^{K}\Delta_k$. On the other hand, the following is an asymptotic lower bound for any consistent centralized policy (Lai and Robbins, 1985): $\liminf_{T\to\infty}\frac{R(T)}{\ln T} \ge \sum_{k:\Delta_k>0}\frac{2\sigma^2}{\Delta_k}$. Thus, we see that the regret obtained in Theorem 5.4.2*

improves significantly the dependence on N of the regret with respect to the trivial algorithm that does not involve communication, and that it is, up to constants, asymptotically optimal in terms of T, with constant N and K. Since in the first iteration of this problem N arms have to be pulled and there is no prior information on the arms' distribution, any asymptotically optimal algorithm in terms of N and K must pull $\Theta(\frac{N}{K}+1)$ times each arm, yielding expected regret of at least the one of the uniform policy: $\left(\frac{N}{K}+1\right)\sum_{k=1}^{K}\Delta_k$, up to a constant.

To sum up, by the lower bound above and the latter argument, we have the following lower bounds for the problem we consider, where $o(\cdot)$ in the first inequality is taken with respect to T only:

$$R(T) \gtrsim \Big(\sum_{k:\Delta_k>0}\frac{\sigma^2}{\Delta_k}+o(1)\Big)\ln(T), \qquad R(T) \gtrsim \Big(\Big(\frac{N}{K}+1\Big)\sum_{k=1}^{K}\Delta_k\Big).$$

The regret obtained in Theorem 5.4.2 is asymptotically optimal with respect to T, as it matches the first inequality, for constant N and K. With respect to the second one, our theorem is optimal in terms of N and K up to at most a factor of $\min(K,N)\ln(N)/\sqrt{\ln(1/|\lambda_2|)}$. Closing this gap is an interesting open problem. We conjecture it would require improving the lower bound.

Remark 5.4.6 (Comparison with previous work). *We will compare DDUCB using the unaccelerated protocol versus coopUCB and coopUCB2, (Landgren, Srivastava, and Leonard, 2019a; Landgren, Srivastava, and Leonard, 2019b) in terms of regret bounds. Note that if $|\lambda_2|>1/e$, our regret bound for the accelerated protocol is even better than the one for the unaccelerated protocol. We start with coopUCB. One can obtain from (Landgren, Srivastava, and Leonard, 2019b) that coopUCB, when using $\ln(tN)$ in the upper confidence bound satisfies the regret bound $R(T)\leq A+B\sum_{k=1}^{K}\Delta_k$, where*

$$A \overset{\text{def}}{=} \sum_{k:\Delta_k>0}\sum_{j=1}^{N}\frac{8\gamma\sigma^2(1+\varepsilon_c^j)}{N\Delta_k}\ln(TN), \qquad B \overset{\text{def}}{=} N\Big(\frac{\gamma}{\gamma-1}+\sqrt{N}\sum_{j=2}^{N}\frac{|\lambda_j|}{1-|\lambda_j|}\Big).$$

Here, $\gamma>1$ is an exploration parameter that the algorithm receives as input and ε_c^j is a non-negative graph-dependent value, which is only 0 when the graph is a complete graph and is potentially large in general. Thus, A is at least $\sum_{k:\Delta_k>0}\frac{8\sigma^2\ln(TN)}{\Delta_k}$. Hence, up to a graph-independent constant, A is always greater than the first summand in the regret of our algorithm in Theorem 5.4.2. Note that $\frac{\gamma}{\gamma-1}\geq 1$ and $\frac{1}{1-|\lambda_2|}\geq\frac{1}{\ln(|\lambda_2|^{-1})}$ so

$$B \geq N\left(1+\frac{\lambda_2'}{\ln(\sqrt{N}/\lambda_2')}\right),$$

where $\lambda_2' \overset{\text{def}}{=} \sqrt{N}|\lambda_2| \in [0,\sqrt{N})$. The factor multiplying $\sum_{k=1}^{K}\Delta_k$ in the second summand in Theorem 5.4.2 is $N\ln N/\ln(1/|\lambda_2|)\leq 2B$, since the inequality below holds.

$$2B \geq 2N\Big(1+\frac{\lambda_2'}{\ln(\sqrt{N}/\lambda_2')}\Big) \geq \frac{N\ln N}{\ln(\sqrt{N}/\lambda_2')} \Leftrightarrow \ln N - 2\ln(\lambda_2') + 2\lambda_2' \geq \ln N.$$

In the case of a complete graph, the problem reduces to a centralized batched bandit problem, in which N actions are taken at each time step (Perchet et al., 2015). The communication in this case is trivial: just send the obtained rewards to your neighbors. So not surprisingly our work and (Landgren, Srivastava, and Leonard, 2019b) incur the same regret in such a case. However, the previous reasoning proves that for every graph our asymptotic regret is never worse and for many graphs we get substantial improvement. Depending on the graph, A and B can be much greater than the lower bound we have used for both of them for comparison purposes. We show in the following that in the case of a cycle graph with a natural communication matrix, these two parts are substantially worse in (Landgren, Srivastava, and Leonard, 2019b), namely $\Theta(N^2)$ versus $\Theta(1)$ and $\Theta(N^{7/2})$ versus $\Theta(N^2 \log N)$ for the term multiplying $\sum_{k:\Delta_k>0} \sigma^2 \ln(TN)/\Delta_k$ in A and for B, respectively. In general, the algorithm we propose presents several improvements with respect to coopUCB. We get a graph-independent value multiplying $\ln(TN)$ in the first summand of the regret whereas A contains the $1+\varepsilon_c^j$ graph-dependent values. In B, just the sum $N(\frac{\gamma}{\gamma-1} + \sqrt{N}\frac{|\lambda_2|}{1-|\lambda_2|})$ is of greater order than our second summand. Moreover, B contains other terms depending on the eigenvalues λ_j for $j \geq 3$. We get this while using less global information about the graph. This is of interest for decentralization purposes. It has computational implications as well, since in principle the computation of ε_c^j needs the entire set of eigenvalues and eigenvectors of P. Now we focus on coopUCB2, which only needs N and does not have to estimate λ_2. Its regret bound $A_2 + B_2 \sum_{k=1}^K \Delta_k$ depends on the exploration function $f(T)$, that has to be sublogarithmic. A_2 can be made graph-independent if f is chosen to be $g(t)/N$ for sublogarithmic g. The original paper does not mention this fact, but checking the proof in (Landgren, Srivastava, and Leonard, 2019a) we note that $f(T)$ can depend on N so we can obtain such graph-independent bound. In any case, it is always $A_2 \gtrsim$ our factor in Theorem 5.4.2. And on the other hand B_2 is at least B plus the superexponential term $f^{-1}(\varepsilon_c^k)$. Or $g^{-1}(N\varepsilon_c^k)$, if the other approach is used.

We present the example with the cycle graph now. If we take P to be symmetric, it is $\sum_{j=1}^N \frac{\varepsilon_c^j}{N} = \sum_{j=2}^N \frac{{\lambda_j}^2}{1-{\lambda_j}^2}$. Consider the graph G to be a cycle with an odd number of nodes and greater than 1, and take as P the matrix such that $P_{ij} = 1/2$ if $i = j \pm 1 \mod N$ and $P_{ij} = 0$ otherwise. Then P is a circulant matrix and their eigenvalues are $\cos(2\pi j/N)$, $j = 0, 1, \ldots, N-1$. Then $\frac{{\lambda_2}^2}{1-{\lambda_2}^2} = \cot^2\left(\frac{2\pi}{N}\right) \geq \frac{N^2}{4\pi^2} - \frac{2}{3}$ and $\frac{{\lambda_3}^2}{1-{\lambda_3}^2} = \cot^2\left(\frac{4\pi}{N}\right) \geq \frac{N^2}{16\pi^2} - \frac{2}{3}$.

As a consequence, B is greater than the corresponding summand in Theorem 5.4.2 in our bound by at least a summand which is $\Theta(N^{7/2})$. On the other hand our summand is $\Theta(N^2 \log N)$, when using (5.4.4) and $\Theta(N^3 \log N)$ when using (5.4.2). In addition, A is greater than the corresponding summand in Theorem 5.4.2 by a factor of $\Theta(N^2)$. The corresponding factor in coopUCB2 is of the same order as with DDUCB, but the T independent summand in coopUCB2 is superexponential in N in contrast to our $\Theta(N^2 \log N)$ or $\Theta(N^3 \log N)$. The bounds above can be proven by a Taylor expansion: $x^2 \cot^2\left(\frac{1}{x}\right) = 1 - \frac{2x^2}{3} + \frac{\xi^4}{15}$, for $x > 0$ and $\xi \in [0, x]$. So $\cot^2\left(\frac{1}{x}\right) \geq \frac{1}{x^2} - \frac{2}{3}$. These are the latter for $x = \frac{N}{2\pi}$ and $x = \frac{N}{4\pi}$.

We note that in (Landgren, Srivastava, and Leonard, 2019a; Landgren, Srivastava, and Leonard, 2019b), the authors used the time step $\ln(t)$ instead of $\ln(tN)$ to define the upper con-

fidence bound. We compared both algorithms in the case in which $\ln(tN)$ *is used. A similar comparison holds for the other case.*

Remark 5.4.7 (Variants of DDUCB). *The algorithm can be modified slightly to obtain better estimations of* $\nu^{(t,k)}/N$ *and* $n^{(t,k)}/N$*, which implies the regret is improved. The easiest (and recommended) modification is the following. While waiting for the vectors* $\beta^{(i)}$ *and* $b^{(i)}$*,* $i = 1, \dots N$ *to be mixed, each agent* i *adds to the variables* $\alpha^{(i)}$ *and* $a^{(i)}$ *the information of the pulls that are done times* $1/N$*. The variable* s *accounting for the time step has to be modified accordingly. It contains the number of pulls made to obtain the approximations of* $\alpha^{(i)}$ *and* $a^{(i)}$*, so it needs to be increased by one when adding one extra reward. This corresponds to uncommenting Lines 13-14 in Algorithm 9. Since the values of* $\alpha^{(i)}$ *and* $a^{(i)}$ *are overwritten after the for loop, the assignment of* s *which is after the loop remains unchanged. Note that if the lines are not uncommented then each time the for loop is executed the* C *pulls that are done by an agent are taken with respect to the same arm. Another variant that would provide better estimations, while keeping the communication cost* $O(K)$ *would consist of also sending the information of the new pull,* $\pi_i^{(t,k)}$ *and* $p_i^{(t,k)}$*, to the neighbors of* i*, receiving their respective values of their new pulls and adding these values to* $\alpha^{(i)}$ *and* $a^{(i)}$ *multiplied by* $1/N$*, respectively. Our analysis of the algorithm presents no modifications for the sake of clarity of exposition. The same asymptotic upper bound on the regret in Theorem 5.4.2 can be computed for these two variations.*

We can vary the communication rate with some trade-offs. On the one hand, we can mix values of $\delta^{(i)}$ *and* $d^{(i)}$ *at each iteration of the for loop, in an unaccelerated way and with (5.4.2) (see Algorithm 9, Line 15) to get even more precise estimations. In such a case, we could use* $\delta^{(i)}$ *and* $d^{(i)}$ *to compute the upper confidence bounds instead of* $\alpha^{(i)}$ *and* $a^{(i)}$*. However, that approach cannot benefit from using the information from local pulls obtained during the stage. On the other hand, if each agent could not communicate* $2K$ *values per iteration, corresponding to the mixing step in Line 11, the algorithm can be slightly modified to account for it at the expense of incurring greater regret. Suppose each agent can only communicate* L *values to her neighbors per iteration. Let* E *be* $\lceil 2KC/L \rceil$*. If each agent runs the algorithm in stages of* E *iterations, ensuring to send each element of* $\beta^{(i)}$ *and* $b^{(i)}$ *exactly* C *times and using the mixing step* C *times, then the bounds in Theorem 5.4.2, substituting* C *by* E*, still hold. Again, in the asymptotic bound,* $N \ln N / \sqrt{\ln(1/|\lambda_2|)}$ *would be substituted by* NE*. In each iteration, agents have to send values corresponding to the same entries of* $\beta^{(i)}$ *or* $b^{(i)}$*. The factor of* C *in the second summand of the regret accounts for the number of rounds of delay since a reward is obtained until it is used to compute upper confidence bounds. If we decrease the communication rate and compensate it with a greater delay, the approximations in* $\alpha^{(i)}$ *and* $a^{(i)}$ *satisfy the same properties as in the original algorithm. Only the second summand in the regret increases because of an increment of the delay. Also note that in practice we send a finite number of bits and the error incurred by the approximation can be encapsulated in our* ε *parameter. And we know how it will affect the regret bound.*

Remark 5.4.8 (Estimation of the number of nodes). *The total number of nodes can be estimated at the beginning of the algorithm, with high probability. Given a value per node, the gossip protocol allows for the computation at each node of the average of those values. If a node starts with a number $u \neq 0$ and the rest of the nodes start with the value 0, then using the gossip protocol after some iterations makes the nodes hold an approximation of the value u/N. The approximation improves exponentially with the number of steps and it does not depend on N, but on the spectral gap. See (5.4.3), for instance. To recover N, the value u is broadcast as well at the same time the values are being mixed, so at each time step a node receives from her neighbors the mixing value and u. In order to compute this protocol in a decentralized way, we make every node compute a number u_i at random and they start broadcasting and mixing it separately. However, during the mixing process we make each node only keep and mix the value corresponding to the minimum u_i so at the end of this process each node only contains $u = \min u_i$ and the approximate value of u/N, if no two nodes started with $\min u_i$, which only occurs with low probability. The procedure can be repeated to increase the probability of success.*

The approximations of N can be broadcast and nodes could use the minimum and maximum as lower and upper bounds on N. The algorithm really only needs upper and lower bounds on N. The delay constant C would be computed with the upper bound on N and the upper confidence bound would be computed using the lower bound, which translates to using a greater exploration parameter η. Since our analysis was done in general for the delay and the exploration parameters, the bounds in Theorem 5.4.2 hold, substituting the delay and exploration parameters by the new values.

We can also obtain an instance independent regret bound, which we present in the following theorem.

Theorem 5.4.9 (Instance Independent Regret Analysis). *The regret achieved by the DDUCB algorithm is*

$$R(T) \lesssim \sqrt{KTN\sigma^2 \ln(TN)} + K\frac{N\Lambda \ln N}{\sqrt{\ln(1/|\lambda_2|)}},$$

where Λ is an upper bound on the gaps Δ_k, $k = 1, \ldots, K$. Here, $\lesssim$ does not only omit constants but also η and ε.

Proof Define D_1 as the set of arms such that their respective gaps are all less than $\sqrt{\frac{K}{TN}\sigma^2 \ln(TN)}$ and D_2 as the set of arms that are not in D_1. Then we can bound the regret incurred by pulling arms in D_1, in the following way

$$\sum_{k \in D_1} \mathbb{E}[n^{(T,k)}]\Delta_k \leq \sqrt{\frac{K}{TN}\sigma^2 \ln(TN)} \sum_{k \in D_1} \mathbb{E}[n^{(T,k)}] \leq \sqrt{KTN\sigma^2 \ln(TN)}.$$

Using Theorem 5.4.2 we can bound the regret obtained by the pulls done to arms in D_2:

$$\sum_{k\in D_2}\mathbb{E}[n^{(T,k)}]\Delta_k \lesssim \sum_{k\in D_2}\frac{\sigma^2\ln(TN)}{\Delta_k}+\frac{N\ln(N)}{\sqrt{\ln(1/|\lambda_2|)}}\Delta_k$$
$$\leq \sum_{k\in D_2}\sqrt{\frac{TN\sigma^2\ln(TN)}{K}}+\frac{N\Lambda\ln(N)}{\sqrt{\ln(1/|\lambda_2|)}} \leq \sqrt{KTN\sigma^2\ln(TN)}+K\frac{N\Lambda\ln(N)}{\sqrt{\ln(1/|\lambda_2|)}}.$$

Adding the two bounds above yields the result.

■

5.5 Experiments

We show that the algorithm proposed in this work, DDUCB, does not only enjoy a better theoretical regret guarantee but it also performs better in practice. The code for the experiments in this work can be found at https://github.com/damaru2/decentralized-bandits. We have observed that the accelerated method performs well with some fixed values for the exploration parameter η and the parameter ε that measures the precision of the mixing after a stage. We use these values, that are $\eta = 2$, $\varepsilon = 1/22$. On the other hand, the constant C that appears in the unaccelerated method is usually excessively large, so it is convenient to heuristically decrease it, which corresponds to using a different value of ε. We set ε so the value of C for the unaccelerated method is the same as the accelerated one. We have used a modification of DDUCB, that we recommend, consisting of adding to the variables $\alpha^{(i)}$ and $a^{(i)}$ the values $1/N$ times the information of the pulls that are done by agent i, while waiting for the vectors $\beta^{(i)}$ and $b^{(i)}$ to be mixed. This modification adds extra information that is available so it is always convenient to use it.

We tuned γ, the exploration parameter of coopUCB and coopUCB2 (Landgren, Srivastava, and Leonard, 2019a), to get best results for that algorithm and plot the execution for the best choice of γ, which was $\gamma = 0.0001$. We also use $\gamma = 2$ for comparison purposes. In Fig. 5.1 one can observe that after a few stages DDUCB algorithms learn which the best arm with high precision is and the regret curve that is observed afterwards shows an almost horizontal behavior. After 10000 iterations, coopUCB not only accumulates a greater regret but the slope indicates that it still has not learned effectively which arm is the best. In some instances, for coopUCB2 and when its optimal exploration parameter is set, it can start having better regret than DDUCB, but even in that case it becomes worse than DDUCB after some iterations.

The distributions of the arms in the bandit problem used in the experiments are Gaussian with variance 1. There is one arm with mean 1 and 16 other arms with mean 0.8. We have executed the algorithms for cycle graphs of size 100 and 200 and for square grids of size 100 and 225. Each algorithm for each setting was executed 10 times. Average regret is shown in Fig 5.1. The experiments we present are representative of the regret behavior we have observed in a greater variety of scenarios upon different choices of means, number of arms and variance. Using

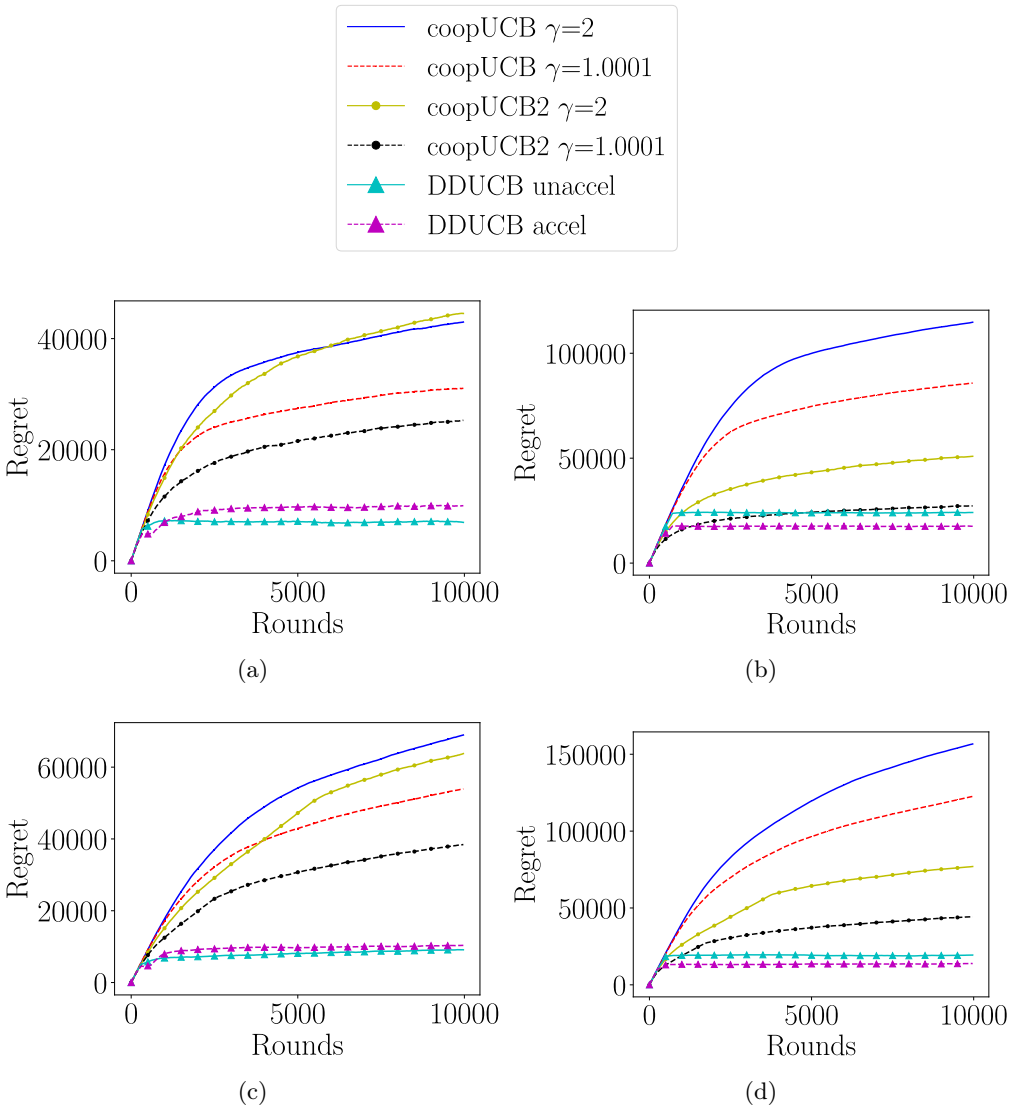


Figure 5.1: Simulation of DDUCB, coopUCB, and coopUCB2 for cycle graphs of (a) 100 nodes and (b) 200 nodes. Simulation of DDUCB, coopUCB, and coopUCB2 for square grids of (c) 100 nodes and (d) 225 nodes. Recall that, in every case, the number of actions that are selected equals the number of rounds times the number of nodes.

different exploration parameters for coopUCB or coopUCB2 did not make it show a behavior as effective as the one observed for DDUCB.

The matrix P was chosen according to (Duchi, Agarwal, and Wainwright, 2012). That is, we define the graph Laplacian as $\mathcal{L} = I - D^{-1/2}AD^{-1/2}$, where A is the adjacency matrix of the communication graph G and D is a diagonal matrix such that D_{ii} contains the degree of node i. Then for regular graphs, if we call δ the common degree of every node, we pick $P = I - \frac{\delta}{\delta+1}\mathcal{L}$. For non regular graphs, like the square grid we used, letting $\delta_{\max}$ be the maximum degree of the nodes we pick $P = I - \frac{1}{\delta_{\max}+1}D^{1/2}\mathcal{L}D^{1/2}$. These matrices always satisfy the assumptions needed for a communication matrix. Building the matrix only requires the mild assumption of knowing an upper bound on $\delta_{\max}$. And the spectral gap can be estimated (Franceschelli et al., 2013). For reference, for our choice of P, the inverse of the spectral graph of the cycle is $O(N^2)$ and it

is $O(N)$ for the grid (Duchi, Agarwal, and Wainwright, 2012). Note it is $O(1)$ for an expander graph.

Chapter 6

Conclusion

In this research, we explored several problems in optimization and online learning. A key element in all of our solutions was the application and extensions of accelerated optimization techniques. We overviewed some ideas that show that one very fruitful point of view for acceleration is the use of online learning algorithms, like Mirror Descent or Follow the Regularized Leader, for the estimation of lower bounds on the function at the same time that one exploits other properties of the function to jointly minimize the duality gap as fast as possible. Indeed, this was a crucial tool for our algorithms.

Accelerated frameworks and other optimization techniques have allowed to generalize classical optimization solutions with a black-box gradient oracle to obtain algorithms that work in a variety of settings, like stochastic, composite, finite sum, non-convex, universal algorithms, to name a few. These general algorithms have numerous applications and sometimes the right formulation and treatment of our problems allows to solve a part of them by making use of these algorithms, like what we did in the decentralized bandit problem in Chapter 5. The development of new more general optimization algorithms with a black-box gradient oracle is an active and fruitful area of research. We contributed to this direction in Chapter 3, by extending accelerated techniques to a constrained non-convex problem that, as we show, can be used for the optimization of geodesically convex functions defined in Riemannian manifolds. However, we would argue that as we gain more understanding about how to obtain general solutions, the next frontier is in the use of particular structures of our problems, i.e., in opening the black box. Combining these structures with the knowledge of general optimization techniques, we may obtain new tools and developments. In Chapter 4 we obtained a solution to our problem by following this philosophy, which allowed to obtain an algorithm with several desirable properties in this setting: fast optimization due to acceleration, determinism, distribution and width-independence.

Finally, a word on the format of this thesis. For most notations appearing in this dissertation, we included non-intrusive links pointing to their definitions. Mathematics is often read non-linearly or require contexts that frequently are inconveniently scattered. We need more ways and technological tools to facilitate the search, understanding, and dissemination of research and its ideas. We hope this concept helps moving in this direction and we hope to see it used in more works or that it inspires other support structures or tools.

Bibliography

Abernethy, Jacob D., Elad Hazan, and Alexander Rakhlin (2008). "Competing in the Dark: An Efficient Algorithm for Bandit Linear Optimization". In: *21st Annual Conference on Learning Theory - COLT 2008, Helsinki, Finland, July 9-12, 2008*. Ed. by Rocco A. Servedio and Tong Zhang. Omnipress, pp. 263–274 (cit. on p. 3).

Ahn, Kwangjun and Suvrit Sra (2020). "From Nesterov's Estimate Sequence to Riemannian Acceleration". In: *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*. Ed. by Jacob D. Abernethy and Shivani Agarwal. Vol. 125. Proceedings of Machine Learning Research. PMLR, pp. 84–118 (cit. on pp. 5, 47, 49, 50, 52–54, 65, 95).

Alimisis, Foivos, Antonio Orvieto, Gary Bécigneul, and Aurélien Lucchi (2020). "A Continuous-time Perspective for Modeling Acceleration in Riemannian Optimization". In: *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, pp. 1297–1307 (cit. on pp. 52, 62).

— (2021). "Momentum Improves Optimization on Riemannian Manifolds". In: *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, pp. 1351–1359 (cit. on p. 52).

Allen-Zhu, Zeyuan (2017a). "Katyusha: The First Direct Acceleration of Stochastic Gradient Methods". In: *J. Mach. Learn. Res.* 18, 221:1–221:51 (cit. on pp. 2, 16, 35, 47).

— (2017b). "Natasha: Faster Non-Convex Stochastic Optimization via Strongly Non-Convex Parameter". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 89–97 (cit. on p. 35).

— (2018a). "Katyusha X: Practical Momentum Method for Stochastic Sum-of-Nonconvex Optimization". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 179–185 (cit. on pp. 35, 47).

— (2018b). "Natasha 2: Faster Non-Convex Optimization Than SGD". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pp. 2680–2691 (cit. on pp. 35, 47).

Allen-Zhu, Zeyuan, Ankit Garg, Yuanzhi Li, Rafael Mendes de Oliveira, and Avi Wigderson (2018). "Operator scaling via geodesically convex optimization, invariant theory and polynomial identity testing". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*. Ed. by Ilias Diakonikolas, David Kempe, and Monika Henzinger. ACM, pp. 172–181 (cit. on p. 48).

Allen-Zhu, Zeyuan and Elad Hazan (2016). "Optimal Black-Box Reductions Between Optimization Objectives". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 1606–1614 (cit. on pp. 6, 51, 60, 63).

Allen-Zhu, Zeyuan and Yuanzhi Li (2018). "NEON2: Finding Local Minima via First-Order Oracles". In: *Advances in Neural Information Processing Systems 31: Annual Conference on*

Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada. Ed. by Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, pp. 3720–3730 (cit. on p. 35).

Allen-Zhu, Zeyuan, Yuanzhi Li, Rafael Mendes de Oliveira, and Avi Wigderson (2017). "Much Faster Algorithms for Matrix Scaling". In: *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pp. 890–901 (cit. on p. 47).

Allen-Zhu, Zeyuan and Lorenzo Orecchia (2015). "Using Optimization to Break the Epsilon Barrier: A Faster and Simpler Width-Independent Algorithm for Solving Positive Linear Programs in Parallel". In: *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*. Ed. by Piotr Indyk. SIAM, pp. 1439–1456 (cit. on p. 100).

— (2017). "Linear Coupling: An Ultimate Unification of Gradient and Mirror Descent". In: *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*. Ed. by Christos H. Papadimitriou. Vol. 67. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 3:1–3:22 (cit. on pp. 3, 29, 31, 51, 55, 57, 59, 60, 66, 101).

— (2019). "Nearly linear-time packing and covering LP solvers - Achieving width-independence and 1/epsilon-convergence". In: *Math. Program.* 175.1-2, pp. 307–353 (cit. on pp. 7, 16, 47, 100, 101, 107).

Allen-Zhu, Zeyuan, Zheng Qu, Peter Richtárik, and Yang Yuan (2016). "Even Faster Accelerated Coordinate Descent Using Non-Uniform Sampling". In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pp. 1110–1119 (cit. on p. 47).

Allybokus, Zaid, Konstantin Avrachenkov, Jérémie Leguay, and Lorenzo Maggi (2018). "Lower Bounds for the Fair Resource Allocation Problem". In: *CoRR* abs/1802.02932 (cit. on p. 101).

Anandkumar, Animashree, Nithin Michael, Ao Kevin Tang, and Ananthram Swami (2011). "Distributed Algorithms for Learning and Cognitive Medium Access with Logarithmic Regret". In: *IEEE J. Sel. Areas Commun.* 29.4, pp. 731–745 (cit. on pp. 114, 116).

Arioli, Mario and Jennifer A. Scott (2014). "Chebyshev acceleration of iterative refinement". In: *Numer. Algorithms* 66.3, pp. 591–608 (cit. on p. 115).

Arjevani, Yossi and Ohad Shamir (2016). "On the Iteration Complexity of Oblivious First-Order Optimization Algorithms". In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. Ed. by Maria-Florina Balcan and Kilian Q. Weinberger. Vol. 48. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 908–916 (cit. on p. 9).

Asi, Hilal and John C. Duchi (2019). "Modeling simple structures and geometry for better stochastic optimization algorithms". In: *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, pp. 2425–2434 (cit. on p. 19).

Atkinson, Anthony B (1970). "On the measurement of inequality". In: *Journal of economic theory* 2.3, pp. 244–263 (cit. on p. 98).

Audibert, Jean-Yves and Sébastien Bubeck (2009). "Minimax Policies for Adversarial and Stochastic Bandits". In: *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009* (cit. on p. 117).

Auer, Peter, Nicolò Cesa-Bianchi, and Paul Fischer (2002). "Finite-time Analysis of the Multiarmed Bandit Problem". In: *Mach. Learn.* 47.2-3, pp. 235–256 (cit. on p. 124).

Auslender, Alfred and Marc Teboulle (2006). "Interior Gradient and Proximal Methods for Convex and Conic Optimization". In: *SIAM J. Optim.* 16.3, pp. 697–725 (cit. on p. 29).

Auzinger, W and J Melenk (2011). "Iterative solution of large linear systems". In: *Lecture notes, TU Wien* (cit. on p. 121).

Awerbuch, Baruch and Rohit Khandekar (2008). "Stateless distributed gradient descent for positive linear programs". In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. Ed. by Cynthia Dwork. ACM, pp. 691–700 (cit. on p. 100).

Awerbuch, Baruch and Robert Kleinberg (2008). "Competitive collaborative learning". In: *J. Comput. Syst. Sci.* 74.8, pp. 1271–1288 (cit. on p. 117).

Bai, Yu, Qijia Jiang, and Ju Sun (2019). "Subgradient Descent Learns Orthogonal Dictionaries". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net (cit. on p. 48).

Bar-On, Yogev and Yishay Mansour (2019). "Individual Regret in Cooperative Nonstochastic Multi-Armed Bandits". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, pp. 3110–3120 (cit. on p. 117).

Beck, Amir, Angelia Nedic, Asuman E. Ozdaglar, and Marc Teboulle (2014). "An O(1/k) Gradient Method for Network Resource Allocation Problems". In: *IEEE Trans. Control. Netw. Syst.* 1.1, pp. 64–73 (cit. on pp. 100, 101).

Beck, Amir and Marc Teboulle (2009). "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems". In: *SIAM J. Imaging Sci.* 2.1, pp. 183–202 (cit. on p. 2).

Berthier, Raphaël, Francis R. Bach, and Pierre Gaillard (2020). "Accelerated Gossip in Networks of Given Dimension Using Jacobi Polynomial Iterations". In: *SIAM J. Math. Data Sci.* 2.1, pp. 24–47 (cit. on p. 124).

Bertsekas, Dimitri, Angelia Nedic, and Asuman Ozdaglar (2003). "Convex analysis and optimization". In: vol. 1. Athena Scientific, pp. 245–247 (cit. on p. 11).

Bertsimas, Dimitris, Vivek F. Farias, and Nikolaos Trichakis (2011). "The Price of Fairness". In: *Oper. Res.* 59.1, pp. 17–31 (cit. on pp. 7, 98).

Bonald, Thomas and James W. Roberts (2015). "Multi-Resource Fairness: Objectives, Algorithms and Performance". In: *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, Portland, OR, USA, June 15-19, 2015*. Ed. by Bill Lin, Jun (Jim) Xu, Sudipta Sengupta, and Devavrat Shah. ACM, pp. 31–42 (cit. on p. 98).

Boyd, Stephen P., Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah (2006). "Randomized gossip algorithms". In: *IEEE Trans. Inf. Theory* 52.6, pp. 2508–2530 (cit. on pp. 114, 115).

Braca, Paolo, Stefano Maranò, and Vincenzo Matta (2008). "Enforcing Consensus While Monitoring the Environment in Wireless Sensor Networks". In: *IEEE Trans. Signal Process.* 56.7-2, pp. 3375–3380 (cit. on p. 119).

Brodén, Björn, Mikael Hammar, Bengt J. Nilsson, and Dimitris Paraschakis (2017). "Bandit Algorithms for e-Commerce Recommender Systems: Extended Abstract". In: *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*. Ed. by Paolo Cremonesi, Francesco Ricci, Shlomo Berkovsky, and Alexander Tuzhilin. ACM, p. 349 (cit. on p. 118).

Bubeck, Sébastien and Nicolò Cesa-Bianchi (2012). "Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems". In: *Found. Trends Mach. Learn.* 5.1, pp. 1–122 (cit. on p. 114).

Bubeck, Sébastien, Qijia Jiang, Yin Tat Lee, Yuanzhi Li, and Aaron Sidford (2019). "Complexity of Highly Parallel Non-Smooth Convex Optimization". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, pp. 13900–13909 (cit. on p. 1).

Bubeck, Sébastien, Yin Tat Lee, and Mohit Singh (2015). "A geometric alternative to Nesterov's accelerated gradient descent". In: *CoRR* abs/1506.08187 (cit. on p. 4).

Buccapatnam, Swapna, Atilla Eryilmaz, and Ness B. Shroff (2013). "Multi-armed bandits in the presence of side observations in social networks". In: *Proceedings of the 52nd IEEE Conference on Decision and Control, CDC 2013, Florence, Italy, December 10-13, 2013*. IEEE, pp. 7309–7314 (cit. on p. 114).

Bullins, Brian and Richard Peng (2019). "Higher-Order Accelerated Methods for Faster Non-Smooth Optimization". In: *CoRR* abs/1906.01621 (cit. on p. 1).

Bürgisser, Peter, Cole Franks, Ankit Garg, Rafael Mendes de Oliveira, Michael Walter, and Avi Wigderson (2019). "Towards a theory of non-commutative optimization: geodesic first and second order methods for moment maps and polytopes". In: *CoRR* abs/1910.12375 (cit. on p. 48).

Busemann, Herbert and Bhalchandra Phadke (1984). "A general version of Beltrami's theorem in the large". In: *Pacific Journal of Mathematics* 115.2, pp. 299–315 (cit. on p. 58).

Cambier, Léopold and Pierre-Antoine Absil (2016). "Robust Low-Rank Matrix Completion by Riemannian Optimization". In: *SIAM J. Scientific Computing* 38.5 (cit. on p. 48).

Carmon, Yair, John C. Duchi, Oliver Hinder, and Aaron Sidford (2017). ""Convex Until Proven Guilty": Dimension-Free Acceleration of Gradient Descent on Non-Convex Functions". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 654–663 (cit. on pp. 2, 48).

Carmon, Yair, Arun Jambulapati, Qijia Jiang, Yujia Jin, Yin Tat Lee, Aaron Sidford, and Kevin Tian (2020). "Acceleration with a Ball Optimization Oracle". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (cit. on pp. 1, 3, 46).

Cesa-Bianchi, Nicolò, Claudio Gentile, Yishay Mansour, and Alberto Minora (2016). "Delay and Cooperation in Nonstochastic Bandits". In: *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*. Ed. by Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir. Vol. 49. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 605–622 (cit. on p. 117).

Chakraborty, Mithun, Kai Yee Phoebe Chua, Sanmay Das, and Brendan Juba (2017). "Coordinated Versus Decentralized Exploration In Multi-Agent Multi-Armed Bandits". In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*. Ed. by Carles Sierra. ijcai.org, pp. 164–170 (cit. on p. 116).

Cherian, Anoop and Suvrit Sra (2017). "Riemannian Dictionary Learning and Sparse Coding for Positive Definite Matrices". In: *IEEE Trans. Neural Networks Learn. Syst.* 28.12, pp. 2859–2871 (cit. on p. 48).

Cohen, Michael, Jelena Diakonikolas, and Lorenzo Orecchia (2018). "On Acceleration with Noise-Corrupted Gradients". In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 1018–1027 (cit. on pp. 29, 32, 48).

Criado, Francisco, David Martínez-Rubio, and Sebastian Pokutta (2021). "Fast Algorithms for Packing Proportional Fairness and its Dual". In: *arXiv preprint arXiv:2109.03678* (cit. on pp. 5, 101).

Criscitiello, Chris and Nicolas Boumal (2019). "Efficiently escaping saddle points on manifolds". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 5985–5995 (cit. on p. 48).

— (2020). "An accelerated first-order method for non-convex optimization on manifolds". In: *arXiv preprint arXiv:2008.02252* (cit. on p. 48).

Cutkosky, Ashok and Kwabena A. Boahen (2017). "Online Learning Without Prior Information". In: *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*. Ed. by Satyen Kale and Ohad Shamir. Vol. 65. Proceedings of Machine Learning Research. PMLR, pp. 643–677 (cit. on p. 20).

Cutkosky, Ashok and Tamás Sarlós (2019). "Matrix-Free Preconditioning in Online Learning". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 1455–1464 (cit. on p. 48).

d'Aspremont, Alexandre, Damien Scieur, and Adrien Taylor (2021). "Acceleration Methods". In: *CoRR* abs/2101.09545 (cit. on pp. 4, 42).

De Carvalho Bento, Glaydston, Orizon P. Ferreira, and Jefferson G. Melo (2017). "Iteration-Complexity of Gradient, Subgradient and Proximal Point Methods on Riemannian Manifolds". In: *J. Optim. Theory Appl.* 173.2, pp. 548–562 (cit. on p. 48).

Defazio, Aaron, Francis R. Bach, and Simon Lacoste-Julien (2014). "SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives". In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*. Ed. by Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, pp. 1646–1654 (cit. on p. 16).

Diakonikolas, Jelena, Maryam Fazel, and Lorenzo Orecchia (2020). "Fair Packing and Covering on a Relative Scale". In: *SIAM J. Optim.* 30.4, pp. 3284–3314 (cit. on pp. 7, 100, 101, 105, 110).

Diakonikolas, Jelena and Michael I. Jordan (2021). "Generalized Momentum-Based Methods: A Hamiltonian Perspective". In: *SIAM J. Optim.* 31.1, pp. 915–944 (cit. on p. 48).

Diakonikolas, Jelena and Lorenzo Orecchia (2017). "Solving Packing and Covering LPs in $\tilde{O}(1/\varepsilon^2)$ Distributed Iterations with a Single Algorithm and Simpler Analysis". In: *CoRR* abs/1710.09002 (cit. on p. 100).

— (2018). "Accelerated Extra-Gradient Descent: A Novel Accelerated First-Order Method". In: *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, 23:1–23:19 (cit. on pp. 3, 36, 48, 53, 55, 57, 59, 68).

— (2019a). "Conjugate Gradients and Accelerated Methods Unified: The Approximate Duality Gap View". In: *CoRR* abs/1907.00289 (cit. on p. 24).

— (2019b). "The Approximate Duality Gap Technique: A Unified Theory of First-Order Methods". In: *SIAM Journal on Optimization* 29.1, pp. 660–689 (cit. on pp. 3, 36, 39, 53, 55, 57–59, 66).

Dimakis, Alexandros G., Soummya Kar, José M. F. Moura, Michael G. Rabbat, and Anna Scaglione (2010). "Gossip Algorithms for Distributed Signal Processing". In: *Proc. IEEE* 98.11, pp. 1847–1864 (cit. on pp. 114, 115, 124).

Drori, Yoel (2017). "The exact information-based complexity of smooth convex minimization". In: *J. Complex.* 39, pp. 1–16 (cit. on pp. 4, 9, 42).

Drori, Yoel and Adrien Taylor (2020). "Efficient first-order methods for convex minimization: a constructive approach". In: *Math. Program.* 184.1, pp. 183–220 (cit. on p. 42).

— (2021). "On the oracle complexity of smooth strongly convex minimization". In: *arXiv preprint arXiv:2101.09740* (cit. on p. 42).

Drori, Yoel and Marc Teboulle (2014). "Performance of first-order methods for smooth convex minimization: a novel approach". In: *Math. Program.* 145.1-2, pp. 451–482 (cit. on pp. 4, 42).

Duchi, John C., Alekh Agarwal, and Martin J. Wainwright (2012). "Dual Averaging for Distributed Optimization: Convergence Analysis and Network Scaling". In: *IEEE Trans. Autom. Control.* 57.3, pp. 592–606 (cit. on pp. 114, 115, 119, 124, 133, 134).

Edelman, Alan, Tomás A. Arias, and Steven Thomas Smith (1998). "The Geometry of Algorithms with Orthogonality Constraints". In: *SIAM J. Matrix Analysis Applications* 20.2, pp. 303–353 (cit. on p. 48).

Fang, Cong, Chris Junchi Li, Zhouchen Lin, and Tong Zhang (2018). "SPIDER: Near-Optimal Non-Convex Optimization via Stochastic Path-Integrated Differential Estimator". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, pp. 687–697 (cit. on p. 35).

Ferreira, Orizon P., Mauricio S. Louzeiro, and Leandro da Fonseca Prudente (2019). "Gradient Method for Optimization on Riemannian Manifolds with Lower Bounded Curvature". In: *SIAM J. Optim.* 29.4, pp. 2517–2541 (cit. on p. 65).

Franceschelli, Mauro, Andrea Gasparri, Alessandro Giua, and Carla Seatzu (2013). "Decentralized estimation of Laplacian eigenvalues in multi-agent systems". In: *Autom.* 49.4, pp. 1031–1036 (cit. on pp. 114, 124, 133).

Gai, Yi, Bhaskar Krishnamachari, and Rahul Jain (2010). "Learning multiuser channel allocations in cognitive radio networks: A combinatorial multi-armed bandit formulation". In: *Symposium on New Frontiers in Dynamic Spectrum*. IEEE, pp. 1–9 (cit. on p. 114).

Gasnikov, Alexander, Pavel Dvurechensky, Eduard Gorbunov, Evgeniya Vorontsova, Daniil Selikhanovych, César A. Uribe, Bo Jiang, Haoyue Wang, Shuzhong Zhang, Sébastien Bubeck, Qijia Jiang, Yin Tat Lee, Yuanzhi Li, and Aaron Sidford (June 2019a). "Near Optimal Methods for Minimizing Convex Functions with Lipschitz p-th Derivatives". In: *Proceedings of the Thirty-Second Conference on Learning Theory*. Ed. by Alina Beygelzimer and Daniel Hsu. Vol. 99. Proceedings of Machine Learning Research. PMLR, pp. 1392–1393 (cit. on p. 1).

Gasnikov, Alexander, Pavel E. Dvurechensky, Eduard A. Gorbunov, Evgeniya A. Vorontsova, Daniil Selikhanovych, César A. Uribe, Bo Jiang, Haoyue Wang, Shuzhong Zhang, Sébastien Bubeck, Qijia Jiang, Yin Tat Lee, Yuanzhi Li, and Aaron Sidford (2019b). "Near Optimal Methods for Minimizing Convex Functions with Lipschitz p-th Derivatives". In: *Conference on Learning Theory, COLT 2019, 25-28 June 2019, Phoenix, AZ, USA*, pp. 1392–1393 (cit. on p. 48).

Genicot, Matthieu, Wen Huang, and Nickolay T. Trendafilov (2015). "Weakly Correlated Sparse Components with Nearly Orthonormal Loadings". In: *Geometric Science of Information - Second International Conference, GSI 2015, Palaiseau, France, October 28-30, 2015, Proceedings*, pp. 484–490 (cit. on p. 48).

Glowacka, Dorota (2019). "Bandit algorithms in recommender systems". In: *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019*. Ed. by Toine Bogers, Alan Said, Peter Brusilovsky, and Domonkos Tikk. ACM, pp. 574–575 (cit. on p. 118).

Greenberg, Marvin J (1993). *Euclidean and non-Euclidean geometries: Development and history*. Macmillan. ISBN: 9781429281331 (cit. on pp. 54, 78, 85).

Grove, Karsten, Peter Petersen, and Silvio Levy (1997). *Comparison geometry*. Vol. 30. Cambridge University Press. ISBN: 0521592224 (cit. on pp. 49, 52, 58).

Güler, Osman (1992). "New Proximal Point Algorithms for Convex Minimization". In: *SIAM J. Optim.* 2.4, pp. 649–664 (cit. on p. 44).

Guminov, Sergey and Alexander Gasnikov (2017). "Accelerated Methods for alpha-Weakly-Quasi-Convex Problems". In: *arXiv preprint arXiv:1710.00797* (cit. on pp. 53, 56).

Guminov, SV, Yu E Nesterov, PE Dvurechensky, and AV Gasnikov (2019). "Primal-dual accelerated gradient descent with line search for convex and nonconvex optimization problems". In: *Doklady Akademii nauk*. Vol. 485. 1, pp. 15–18 (cit. on pp. 53, 56).

Hamilton, Linus and Ankur Moitra (2021). "A No-go Theorem for Acceleration in the Hyperbolic Plane". In: *arXiv preprint arXiv:2101.05657* (cit. on pp. 53, 95, 96).

Hazan, Elad and Satyen Kale (2008). "Extracting Certainty from Uncertainty: Regret Bounded by Variation in Costs". In: *21st Annual Conference on Learning Theory - COLT 2008, Helsinki,*

Finland, July 9-12, 2008. Ed. by Rocco A. Servedio and Tong Zhang. Omnipress, pp. 57–68 (cit. on p. 3).

Heidel, Gennadij and Volker Schulz (2018). "A Riemannian trust-region method for low-rank tensor completion". In: *Numerical Lin. Alg. with Applic.* 25.6 (cit. on p. 48).

Hillel, Eshcar, Zohar Shay Karnin, Tomer Koren, Ronny Lempel, and Oren Somekh (2013). "Distributed Exploration in Multi-Armed Bandits". In: *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*. Ed. by Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, pp. 854–862 (cit. on p. 117).

Hinder, Oliver, Aaron Sidford, and Nimit Sharad Sohoni (2020). "Near-Optimal Methods for Minimizing Star-Convex Functions and Beyond". In: *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*. Ed. by Jacob D. Abernethy and Shivani Agarwal. Vol. 125. Proceedings of Machine Learning Research. PMLR, pp. 1894–1938 (cit. on pp. 34, 53, 56).

Hosseini, Reshad and Suvrit Sra (2015). "Matrix Manifold Optimization for Gaussian Mixtures". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 910–918 (cit. on p. 48).

— (2020). "An alternative to EM for Gaussian mixture models: batch and stochastic Riemannian optimization". In: *Math. Program.* 181.1, pp. 187–223 (cit. on p. 48).

Hou, Thomas Y., Zhenzhen Li, and Ziyun Zhang (2020). "Fast Global Convergence for Low-rank Matrix Recovery via Riemannian Gradient Descent with Random Initialization". In: *CoRR* abs/2012.15467 (cit. on p. 48).

Huang, Wen and Ke Wei (2019). "Extending FISTA to Riemannian Optimization for Sparse PCA". In: *arXiv preprint arXiv:1909.05485* (cit. on p. 52).

— (2021). "Riemannian proximal gradient methods". In: *Mathematical Programming*, pp. 1–43 (cit. on p. 48).

Ireland, Kenneth and Michael Rosen (1982). *A classical introduction to modern number theory*. Vol. 84. Graduate texts in mathematics. Springer. ISBN: 978-0-387-90625-6 (cit. on p. 127).

Jain, Kamal and Vijay V. Vazirani (2007). "Eisenberg-Gale markets: algorithms and structural properties". In: *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*. Ed. by David S. Johnson and Uriel Feige. ACM, pp. 364–373 (cit. on p. 98).

— (2010). "Eisenberg-Gale markets: Algorithms and game-theoretic properties". In: *Games Econ. Behav.* 70.1, pp. 84–106 (cit. on p. 98).

Jin, Youngmi and Michiaki Hayashi (2018). "Trade-off between fairness and efficiency in dominant alpha-fairness family". In: *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops, INFOCOM Workshops 2018, Honolulu, HI, USA, April 15-19, 2018*. IEEE, pp. 391–396 (cit. on p. 98).

Joe-Wong, Carlee, Soumya Sen, Tian Lan, and Mung Chiang (2012). "Multi-resource allocation: Fairness-efficiency tradeoffs in a unifying framework". In: *Proceedings of the IEEE INFOCOM 2012, Orlando, FL, USA, March 25-30, 2012*. Ed. by Albert G. Greenberg and Kazem Sohraby. IEEE, pp. 1206–1214 (cit. on p. 98).

Johnson, Rie and Tong Zhang (2013). "Accelerating Stochastic Gradient Descent using Predictive Variance Reduction". In: *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*. Ed. by Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, pp. 315–323 (cit. on p. 16).

Jolliffe, Ian T, Nickolay T Trendafilov, and Mudassir Uddin (2003). "A modified principal component technique based on the LASSO". In: *Journal of computational and Graphical Statistics* 12.3, pp. 531–547 (cit. on p. 48).

Joulani, Pooria, András György, and Csaba Szepesvári (2013). "Online Learning under Delayed Feedback". In: *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*. Vol. 28. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 1453–1461 (cit. on p. 116).

Joulani, Pooria, Anant Raj, András György, and Csaba Szepesvári (2020). "A simpler approach to accelerated optimization: iterative averaging meets optimism". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 4984–4993 (cit. on pp. 2, 4, 43).

Kalathil, Dileep, Naumaan Nayyar, and Rahul Jain (2014). "Decentralized learning for multiplayer multiarmed bandits". In: *IEEE Transactions on Information Theory* 60.4, pp. 2331–2345 (cit. on p. 116).

Kar, Soummya, H. Vincent Poor, and Shuguang Cui (2011). "Bandit problems in networks: Asymptotically efficient distributed allocation rules". In: *50th IEEE Conference on Decision and Control and European Control Conference, 11th European Control Conference, CDC/ECC 2011, Orlando, FL, USA, December 12-15, 2011*. IEEE, pp. 1771–1778 (cit. on p. 116).

Karimi, Hamed, Julie Nutini, and Mark Schmidt (2016). "Linear Convergence of Gradient and Proximal-Gradient Methods Under the Polyak-Łojasiewicz Condition". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I*. Ed. by Paolo Frasconi, Niels Landwehr, Giuseppe Manco, and Jilles Vreeken. Vol. 9851. Lecture Notes in Computer Science. Springer, pp. 795–811 (cit. on pp. 47, 52).

Karimi, Sahar and Stephen A Vavasis (2016). "A unified convergence bound for conjugate gradient and accelerated gradient". In: *arXiv preprint arXiv:1605.00320* (cit. on p. 24).

Kasai, Hiroyuki, Pratik Jawanpuria, and Bamdev Mishra (2019). "Riemannian adaptive stochastic gradient algorithms on matrix manifolds". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 3262–3271 (cit. on p. 48).

Kavis, Ali, Kfir Y. Levy, Francis R. Bach, and Volkan Cevher (2019). "UniXGrad: A Universal, Adaptive Algorithm with Optimal Guarantees for Constrained Optimization". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, pp. 6257–6266 (cit. on p. 2).

Kelly, Frank (1997). "Charging and rate control for elastic traffic". In: *Eur. Trans. Telecommun.* 8.1, pp. 33–37 (cit. on p. 98).

Kelly, Frank and Elena Yudovina (2014). *Stochastic networks*. Vol. 2. Cambridge University Press. ISBN: 978-1-107-69170-4 (cit. on p. 100).

Khuzani, Masoud Badiei and Na Li (2017). "Stochastic Primal-Dual Method on Riemannian Manifolds of Bounded Sectional Curvature". In: *16th IEEE International Conference on Machine Learning and Applications, ICMLA 2017, Cancun, Mexico, December 18-21, 2017*, pp. 133–140 (cit. on p. 48).

Kim, Donghwan and Jeffrey A. Fessler (2016). "Optimized first-order methods for smooth convex minimization". In: *Math. Program.* 159.1-2, pp. 81–107 (cit. on pp. 4, 9, 42).

Korda, Nathan, Balázs Szörényi, and Li Shuai (2016). "Distributed Clustering of Linear Bandits in Peer to Peer Networks". In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. Ed. by Maria-Florina

Balcan and Kilian Q. Weinberger. Vol. 48. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 1301–1309 (cit. on p. 116).

Kreyszig, E. (1991). *Differential Geometry*. Dover Publications. ISBN: 9780486667218 (cit. on pp. 50, 58).

Krichene, Walid, Alexandre M. Bayen, and Peter L. Bartlett (2015). "Accelerated Mirror Descent in Continuous and Discrete Time". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. Ed. by Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, pp. 2845–2853 (cit. on pp. 3, 59).

Lai, Tze Leung and Herbert Robbins (1985). "Asymptotically efficient adaptive allocation rules". In: *Advances in Applied Mathematics* 6.1, pp. 4–22 (cit. on p. 127).

Lan, Guanghui (2012). "An optimal method for stochastic composite optimization". In: *Math. Program.* 133.1-2, pp. 365–397 (cit. on p. 2).

Lan, Tian, David T. H. Kao, Mung Chiang, and Ashutosh Sabharwal (2010). "An Axiomatic Theory of Fairness in Network Resource Allocation". In: *INFOCOM 2010. 29th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 15-19 March 2010, San Diego, CA, USA*. IEEE, pp. 1343–1351 (cit. on pp. 7, 98).

Landgren, Peter, Vaibhav Srivastava, and Naomi Ehrich Leonard (May 2016). "On distributed cooperative decision-making in multiarmed bandits". In: *Control Conference (ECC), 2016 European*. IEEE, pp. 243–248 (cit. on p. 8).

— (2019a). *Distributed Cooperative Decision-Making in Multiarmed Bandits: Frequentist and Bayesian Algorithms* (cit. on pp. 8, 115, 128, 129, 132).

— (2019b). "On distributed cooperative decision-making in multiarmed bandits". In: (cit. on pp. 115, 120, 128, 129).

Lemaréchal, Claude (2012). "Cauchy and the gradient method". In: *Doc Math Extra* 251.254, p. 10 (cit. on p. 1).

Lezcano-Casado, Mario (2019). "Trivializations for Gradient-Based Optimization on Manifolds". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 9154–9164 (cit. on p. 48).

— (2020). "Curvature-Dependant Global Convergence Rates for Optimization on Manifolds of Bounded Geometry". In: *arXiv preprint arXiv:2008.02517* (cit. on p. 62).

Lezcano-Casado, Mario and David Martínez-Rubio (2019). "Cheap Orthogonal Constraints in Neural Networks: A Simple Parametrization of the Orthogonal and Unitary Group". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 3794–3803 (cit. on p. 48).

Lin, Hongzhou, Julien Mairal, and Zaid Harchaoui (2015). "A Universal Catalyst for First-Order Optimization". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. Ed. by Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, pp. 3384–3392 (cit. on pp. 2, 3, 46).

Lin, Lizhen, Bayan Saparbayeva, Michael Minyi Zhang, and David B. Dunson (2020). "Accelerated Algorithms for Convex and Non-Convex Optimization on Manifolds". In: *CoRR* abs/2010.08908 (cit. on p. 52).

Lin, Tianyi, Zeyu Zheng, Elynn Y. Chen, Marco Cuturi, and Michael I. Jordan (2021). "On Projection Robust Optimal Transport: Sample Complexity and Model Misspecification". In: *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, pp. 262–270 (cit. on p. 48).

Liu, Ji, Brian D. O. Anderson, Ming Cao, and A. Stephen Morse (2013). "Analysis of accelerated gossip algorithms". In: *Autom.* 49.4, pp. 873–883 (cit. on p. 124).

Liu, Keqin and Qing Zhao (2010). "Distributed learning in multi-armed bandit with multiple players". In: *IEEE Trans. Signal Process.* 58.11, pp. 5667–5681 (cit. on p. 116).

Liu, Yuanyuan, Fanhua Shang, James Cheng, Hong Cheng, and Licheng Jiao (2017). "Accelerated First-order Methods for Geodesically Convex Optimization on Riemannian Manifolds". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, pp. 4868–4877 (cit. on pp. 50, 51).

Loizou, Nicolas and Peter Richtárik (2019). "Revisiting Randomized Gossip Algorithms: General Framework, Convergence Rates and Novel Block and Accelerated Protocols". In: *CoRR* abs/1905.08645 (cit. on p. 124).

Luby, Michael and Noam Nisan (1993). "A parallel approximation algorithm for positive linear programming". In: *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*. Ed. by S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal. ACM, pp. 448–457 (cit. on p. 100).

Marašević, Jelena, Clifford Stein, and Gil Zussman (2016). "A Fast Distributed Stateless Algorithm for alpha-Fair Packing Problems". In: *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*. Ed. by Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi. Vol. 55. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 54:1–54:15 (cit. on pp. 7, 100, 101).

Martínez-Rubio, David (2020). "Global Riemannian Acceleration in Hyperbolic and Spherical Spaces". In: *arXiv preprint arXiv:2012.03618* (cit. on p. 5).

Martínez-Rubio, David, Varun Kanade, and Patrick Rebeschini (2019). "Decentralized Cooperative Stochastic Bandits". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, pp. 4531–4542 (cit. on p. 5).

Mary, Jérémie, Romaric Gaudel, and Philippe Preux (2015). "Bandits and Recommender Systems". In: *Machine Learning, Optimization, and Big Data - First International Workshop, MOD 2015, Taormina, Sicily, Italy, July 21-23, 2015, Revised Selected Papers*. Ed. by Panos M. Pardalos, Mario Pavone, Giovanni Maria Farinella, and Vincenzo Cutello. Vol. 9432. Lecture Notes in Computer Science. Springer, pp. 325–336 (cit. on p. 118).

McCormick, Bill, Frank Kelly, Patrice Plante, Paul Gunning, and Peter Ashwood-Smith (2014). "Real time alpha-fairness based traffic engineering". In: *Proceedings of the third workshop on Hot topics in software defined networking, HotSDN '14, Chicago, Illinois, USA, August 22, 2014*. Ed. by Aditya Akella and Albert G. Greenberg. ACM, pp. 199–200 (cit. on p. 98).

Mishra, Bamdev and Rodolphe Sepulchre (2014). "R3MC: A Riemannian three-factor algorithm for low-rank matrix completion". In: *53rd IEEE Conference on Decision and Control, CDC 2014, Los Angeles, CA, USA, December 15-17, 2014*, pp. 1137–1142 (cit. on p. 48).

Mo, Jeonghoon and Jean C. Walrand (2000). "Fair end-to-end window-based congestion control". In: *IEEE/ACM Trans. Netw.* 8.5, pp. 556–567 (cit. on p. 98).

Monteiro, Renato D. C. and Benar Fux Svaiter (2013). "An Accelerated Hybrid Proximal Extragradient Method for Convex Optimization and Its Implications to Second-Order Methods". In: *SIAM J. Optim.* 23.2, pp. 1092–1125 (cit. on pp. 1, 3, 44–46).

Nagarajan, Vaishnavh and J Zico Kolter (2019). "Generalization in deep networks: The role of distance from initialization". In: *arXiv preprint arXiv:1901.01672* (cit. on pp. 95, 97).

Nash, John F. (1950). "The Bargaining Problem". In: *Econometrica* 18.2, pp. 155–162. ISSN: 00129682, 14680262 (cit. on p. 98).

Nayyar, Naumaan, Dileep M. Kalathil, and Rahul Jain (2018). "On Regret-Optimal Learning in Decentralized Multiplayer Multiarmed Bandits". In: *IEEE Trans. Control. Netw. Syst.* 5.1, pp. 597–606 (cit. on p. 116).

Nedic, Angelia and Asuman E. Ozdaglar (2009). "Distributed Subgradient Methods for Multi-Agent Optimization". In: *IEEE Trans. Autom. Control.* 54.1, pp. 48–61 (cit. on p. 115).

Nemirovski, A and D Yudin (1983a). "Information-based complexity of mathematical programming". In: *Izvestia AN SSSR, Ser. Tekhnicheskaya Kibernetika (the journal is translated to English as Engineering Cybernetics. Soviet J. Computer & Systems Sci.)* 1 (cit. on pp. 3, 9, 24, 27, 28).

Nemirovski, Arkadi (1982a). [Online; accessed 20-September-2021], https://blogs.princeton.edu/imabandit/wp-content/uploads/sites/122/2019/06/Nemirovski81_EnglishDiscussion.pdf (cit. on p. 24).

— (1982b). "Orth-method for smooth convex optimization". In: *Izvestia AN SSSR, Transl.: Eng. Cybern. Soviet J. Comput. Syst. Sci* 2, pp. 937–947 (cit. on p. 24).

Nemirovski, Arkadi and Yurii Nesterov (1985). "Optimal methods of smooth convex minimization". In: *USSR Computational Mathematics and Mathematical Physics* 25.2, pp. 21–30 (cit. on p. 1).

Nemirovski, Arkadi and David Borisovich Yudin (1983b). "Problem complexity and method efficiency in optimization". In: (cit. on pp. 20, 21).

Nesterov, Yurii (1983). "A method for solving the convex programming problem with convergence rate O $(1/k^2)$". In: *Dokl. akad. nauk Sssr*. Vol. 269. 3, pp. 543–547 (cit. on pp. 1, 3, 28–30, 47, 54, 55, 57, 59, 66).

— (1998). "Introductory lectures on convex programming volume I: Basic course". In: *Lecture notes* 3.4, p. 5 (cit. on pp. 3, 28, 29, 31).

— (2005). "Smooth minimization of non-smooth functions". In: *Math. Program.* 103.1, pp. 127–152 (cit. on pp. 3, 28, 29).

Nocedal, Jorge and Stephen Wright (2006). *Numerical optimization*. Springer Science & Business Media. ISBN: 9780387227429 (cit. on p. 25).

Orabona, Francesco and Dávid Pál (2016). "Coin Betting and Parameter-Free Online Learning". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. Ed. by Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, pp. 577–585 (cit. on p. 20).

Parikh, Neal and Stephen P. Boyd (2014). "Proximal Algorithms". In: *Found. Trends Optim.* 1.3, pp. 127–239 (cit. on p. 44).

Perchet, Vianney, Philippe Rigollet, Sylvain Chassang, and Erik Snowberg (2015). "Batched Bandit Problems". In: *Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015*. Ed. by Peter Grünwald, Elad Hazan, and Satyen Kale. Vol. 40. JMLR Workshop and Conference Proceedings. JMLR.org, p. 1456 (cit. on p. 129).

Petersen, Peter, S Axler, and KA Ribet (2006). *Riemannian geometry*. Vol. 171. Springer. ISBN: 978-0-387-29403-2 (cit. on p. 50).

Raval, Siraj (2016). *Decentralized applications: harnessing Bitcoin's blockchain technology*. " O'Reilly Media, Inc." ISBN: 9781491924549 (cit. on p. 118).

Robbins, Herbert and Sutton Monro (1951). "A stochastic approximation method". In: *The annals of mathematical statistics*, pp. 400–407 (cit. on p. 1).

Sato, Hiroyuki, Hiroyuki Kasai, and Bamdev Mishra (2019a). "Riemannian Stochastic Variance Reduced Gradient Algorithm with Retraction and Vector Transport". In: *SIAM J. Optim.* 29.2, pp. 1444–1472 (cit. on p. 48).

— (2019b). "Riemannian Stochastic Variance Reduced Gradient Algorithm with Retraction and Vector Transport". In: *SIAM Journal on Optimization* 29.2, pp. 1444–1472 (cit. on p. 48).

Scaman, Kevin, Francis R. Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié (2017). "Optimal Algorithms for Smooth and Strongly Convex Distributed Optimization in Networks". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 3027–3036 (cit. on pp. 114, 115, 120, 124).

Scieur, Damien, Francis R. Bach, and Alexandre d'Aspremont (2017). "Nonlinear Acceleration of Stochastic Algorithms". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, pp. 3982–3991 (cit. on p. 4).

Scieur, Damien, Alexandre d'Aspremont, and Francis R. Bach (2020). "Regularized nonlinear acceleration". In: *Math. Program.* 179.1, pp. 47–83 (cit. on p. 4).

Shah, Devavrat (2009). "Gossip Algorithms". In: *Found. Trends Netw.* 3.1, pp. 1–125 (cit. on p. 115).

Shahrampour, Shahin, Alexander Rakhlin, and Ali Jadbabaie (2017). "Multi-armed bandits in multi-agent networks". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*. IEEE, pp. 2786–2790 (cit. on p. 116).

Shalev-Shwartz, Shai and Yoram Singer (2006). "Online Learning Meets Optimization in the Dual". In: *Learning Theory, 19th Annual Conference on Learning Theory, COLT 2006, Pittsburgh, PA, USA, June 22-25, 2006, Proceedings*. Ed. by Gábor Lugosi and Hans Ulrich Simon. Vol. 4005. Lecture Notes in Computer Science. Springer, pp. 423–437 (cit. on p. 3).

— (2007). "Online learning: Theory, algorithms, and applications". In: (cit. on p. 3).

Stranders, Ruben, Long Tran-Thanh, Francesco Maria Delle Fave, Alex Rogers, and Nicholas R. Jennings (2012). "DCOPs and bandits: exploration and exploitation in decentralised coordination". In: *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012, Valencia, Spain, June 4-8, 2012 (3 Volumes)*. Ed. by Wiebe van der Hoek, Lin Padgham, Vincent Conitzer, and Michael Winikoff. IFAAMAS, pp. 289–296 (cit. on p. 114).

Streeter, Matthew J. and H. Brendan McMahan (2012). "No-Regret Algorithms for Unconstrained Online Convex Optimization". In: *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. Ed. by Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, pp. 2411–2419 (cit. on p. 20).

Su, Weijie, Stephen P. Boyd, and Emmanuel J. Candès (2016). "A Differential Equation for Modeling Nesterov's Accelerated Gradient Method: Theory and Insights". In: *J. Mach. Learn. Res.* 17, 153:1–153:43 (cit. on p. 3).

Sun, Ju, Qing Qu, and John Wright (2017). "Complete Dictionary Recovery Over the Sphere II: Recovery by Riemannian Trust-Region Method". In: *IEEE Trans. Inf. Theory* 63.2, pp. 885–914 (cit. on p. 48).

Sun, Yue, Nicolas Flammarion, and Maryam Fazel (2019). "Escaping from saddle points on Riemannian manifolds". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 7274–7284 (cit. on p. 48).

Szörényi, Balázs, Róbert Busa-Fekete, István Hegedűs, Róbert Ormándi, Márk Jelasity, and Balázs Kégl (2013). "Gossip-based distributed stochastic bandit algorithms". In: *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*. Vol. 28. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 19–27 (cit. on p. 116).

Tan, Mingkui, Ivor W. Tsang, Li Wang, Bart Vandereycken, and Sinno Jialin Pan (2014). "Riemannian Pursuit for Big Matrix Recovery". In: *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 1539–1547 (cit. on p. 48).

Taylor, Adrien B. and Yoel Drori (2021). "An optimal gradient method for smooth (possibly strongly) convex minimization". In: *CoRR* abs/2101.09741 (cit. on p. 42).

Taylor, Adrien B., Julien M. Hendrickx, and François Glineur (2017). "Smooth strongly convex interpolation and exact worst-case performance of first-order methods". In: *Math. Program.* 161.1-2, pp. 307–345 (cit. on pp. 4, 42).

Tekin, Cem and Mingyan Liu (2012). "Online learning in decentralized multi-user spectrum access with synchronized explorations". In: *31st IEEE Military Communications Conference, MILCOM 2012, Orlando, FL, USA, October 29 - November 1, 2012*. IEEE, pp. 1–6 (cit. on p. 114).

Tran-Thanh, Long, Alex Rogers, and Nicholas R. Jennings (2012). "Long-term information collection with energy harvesting wireless sensors: a multi-armed bandit based approach". In: *Auton. Agents Multi Agent Syst.* 25.2, pp. 352–394 (cit. on p. 114).

Tripuraneni, Nilesh, Nicolas Flammarion, Francis R. Bach, and Michael I. Jordan (2018). "Averaging Stochastic Gradient Descent on Riemannian Manifolds". In: *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018*. Ed. by Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet. Vol. 75. Proceedings of Machine Learning Research. PMLR, pp. 650–687 (cit. on p. 48).

Tseng, Paul (2008). "On accelerated proximal gradient methods for convex-concave optimization". In: *submitted to SIAM Journal on Optimization* 2.3 (cit. on pp. 2, 3, 29).

Vandereycken, Bart (2013). "Low-Rank Matrix Completion by Riemannian Optimization". In: *SIAM Journal on Optimization* 23.2, pp. 1214–1236 (cit. on p. 48).

Walker, Homer F. and Peng Ni (2011). "Anderson Acceleration for Fixed-Point Iterations". In: *SIAM J. Numer. Anal.* 49.4, pp. 1715–1735 (cit. on p. 25).

Wang, Di, Satish Rao, and Michael W. Mahoney (2016). "Unified Acceleration Method for Packing and Covering Problems via Diameter Reduction". In: *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, 50:1–50:13 (cit. on p. 48).

Wang, X. M., C. Li, and J. C. Yao (2015). "Subgradient Projection Algorithms for Convex Feasibility on Riemannian Manifolds with Lower Bounded Curvatures". In: *J. Optim. Theory Appl.* 164.1, pp. 202–217 (cit. on p. 65).

Wang, Xi, Zhipeng Tu, Yiguang Hong, Yingyi Wu, and Guodong Shi (2021). "No-regret Online Learning over Riemannian Manifolds". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual* (cit. on p. 48).

Wang, Yuanhao, Jiachen Hu, Xiaoyu Chen, and Liwei Wang (2020). "Distributed Bandit Learning: Near-Optimal Regret with Efficient Communication". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net (cit. on p. 116).

Weber, Melanie and Suvrit Sra (2017). "Frank-Wolfe methods for geodesically convex optimization with application to the matrix geometric mean". In: *CoRR* abs/1710.10770 (cit. on p. 48).

— (2019). "Nonconvex stochastic optimization on manifolds via Riemannian Frank-Wolfe methods". In: *CoRR* abs/1910.04194 (cit. on p. 48).

Wei, Ke, Jian-Feng Cai, Tony F Chan, and Shingyu Leung (2016). "Guarantees of Riemannian optimization for low rank matrix recovery". In: *SIAM Journal on Matrix Analysis and Applications* 37.3, pp. 1198–1222 (cit. on p. 48).

Wibisono, Andre, Ashia C. Wilson, and Michael I. Jordan (2016). "A Variational Perspective on Accelerated Methods in Optimization". In: *CoRR* abs/1603.04245 (cit. on pp. 3, 4).

Wiesel, Ami (2012). "Geodesic Convexity and Covariance Estimation". In: *IEEE Trans. Signal Process.* 60.12, pp. 6182–6189 (cit. on p. 48).

Xiao, Lin (2010). "Dual Averaging Methods for Regularized Stochastic Learning and Online Optimization". In: *J. Mach. Learn. Res.* 11, pp. 2543–2596 (cit. on p. 2).

Xiao, Lin and Stephen P. Boyd (2004). "Fast linear iterations for distributed averaging". In: *Syst. Control. Lett.* 53.1, pp. 65–78 (cit. on pp. 118, 119).

Xu, Jie, Cem Tekin, Simpson Zhang, and Mihaela van der Schaar (2015). "Distributed Multi-Agent Online Learning Based on Global Feedback". In: *IEEE Trans. Signal Process.* 63.9, pp. 2225–2238 (cit. on p. 116).

Xu, Yi, Jing Rong, and Tianbao Yang (2018). "First-order Stochastic Algorithms for Escaping From Saddle Points in Almost Linear Time". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, pp. 5535–5545 (cit. on p. 35).

Zhang, Hongyi, Sashank J. Reddi, and Suvrit Sra (2016). "Riemannian SVRG: Fast Stochastic Optimization on Riemannian Manifolds". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 4592–4600 (cit. on p. 48).

Zhang, Hongyi and Suvrit Sra (2016). "First-order Methods for Geodesically Convex Optimization". In: *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, pp. 1617–1638 (cit. on pp. 48–50, 52, 58, 62, 65).

— (2018). "An Estimate Sequence for Geodesically Convex Optimization". In: *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018*. Ed. by Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet. Vol. 75. Proceedings of Machine Learning Research. PMLR, pp. 1703–1723 (cit. on pp. 5, 47, 49, 50, 52–54, 58, 65, 95).

Zhang, Jingzhao, Hongyi Zhang, and Suvrit Sra (2018). "R-SPIDER: A Fast Riemannian Stochastic Optimization Algorithm with Curvature Independent Rate". In: *CoRR* abs/1811.04194 (cit. on p. 48).

Zhang, Lijun, Mehrdad Mahdavi, and Rong Jin (2013). "Linear Convergence with Condition Number Independent Access of Full Gradients". In: *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*. Ed. by Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, pp. 980–988 (cit. on p. 16).

Zhou, Pan, Xiao-Tong Yuan, and Jiashi Feng (2019). "Faster First-Order Methods for Stochastic Non-Convex Optimization on Riemannian Manifolds". In: *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, pp. 138–147 (cit. on p. 48).

Zhou, Xingyu (2018). "On the fenchel duality between strong convexity and lipschitz continuous gradient". In: *arXiv preprint arXiv:1803.06573* (cit. on p. 88).