ONLINE CONVEX OPTIMIZATION: ALGORITHMS, LEARNING, AND DUALITY

VICTOR S. PORTELLA

ADVISOR: MARCEL K. DE CARLI SILVA

Abstract. In this project we plan to study the blossoming field of online convex optimization, a framework in the intersection of optimization and machine learning. Results and algorithms from this line of research have found applications in a wide range of fields such as game theory and theoretical computer science. In this text, we present a summarized, but representative, part of our current progress in the project through a brief introduction to the field of online convex optimization. First, we formally present the online learning and online convex optimization frameworks, and describe some applications and related areas. We also present two categories of online convex optimization algorithms: Follow the Regularized Leader and Online Mirror Descent. We further examine their connections and derive classical results. Finally, we discuss future directions of the project, and we present two topics as examples: bandit convex optimization and boosting. Additionally, we outline the main results from convex analysis needed for this text, with a focus on building intuition.

1. Introduction

Consider the following scenario: at each round of a multi-round game, nature reveals a question or query to a player and her adversary. The player then guesses an answer and, simultaneously, the adversary picks what she considers to be the "true answer" to nature's query. In the end of the round, the player suffers a loss that depends on how well she predicted the adversary's true answer, and a new round begins. For example, the player could be a spam detector that receives an email at each round to be classified as spam or not, and the user, who fits the role of adversary in this scenario, determines the true email classification. Another example is the case where the player is a weather forecaster who, at the beginning of each day, tries to predict if it is going to rain or not on that day. The problem of devising strategies for the player in this scenario has been actively researched in statistics and computer science.

One way to model this problem is to make statistical assumptions over the pairs of queries and answers picked by nature and adversary, an approach taken in the field of statistical learning [14, 62, 63]. That is, the queries and answers picked by nature and adversary are drawn from a probability distribution $\mathcal{D}$, and performance is measured with respect to the expected accuracy of the model over samples from $\mathcal{D}$. In spite of the success of statistical learning, mainly for machine learning problems, this strategy has some drawbacks. Besides the need of statistical assumptions, which might not be valid in some applications, techniques from statistical learning require access to a *training set*, a set of previously sampled examples, which is used to prepare/train the model. Only after training the model, usually a costly operation, one can use it to make predictions. Moreover, training algorithms usually do not adapt to new points in the training set, requiring execution of the algorithm from scratch if one adds new points to the training set.

In this text we will be interested in the case of online learning [37, 40, 57, 58], which frees the adversary from any statistical assumptions, allowing her to make choices that maximize the loss of the player. As one may expect, it is impossible for the player to minimize the number of mistakes, or the amount she loses, against an antagonistic adversary. A more sensible way to measure player efficiency in this case is the game-theoretic idea of regret. Essentially, regret measures how well the player performs compared to some strategy given by a function from queries to answers.

Still, there are plenty of problems from the online learning setting, such as the spam filtering example, on which it is impossible for the player to attain low regret [30]. One assumption that drastically changes the complexity of the problem is convexity of the function that determines the losses, as well as convexity of the sets whence the entities make their choices. In classic optimization theory, the convexity assumption causes optimization to be much more tractable, giving rise to interesting optimality conditions and efficient algorithms [12, 16, 49]. Thus, a reasonable idea is to specialize online learning to the case known as online convex optimization (OCO) [17, 37, 58]: at each round of a multi-round game, the player picks a point x from a convex set, and the adversary simultaneously picks a convex function f. At the end of the round, the player suffers the loss $f(x)$. In this context, we are interested in strategies for the player that perform as well as the the best fixed point for the player in hindsight. Namely, we want strategies which minimize the difference of the losses of the player and the losses of the best fixed point x^* in hindsight. This latter quantity is known as the regret of the player with respect to x^*. Hopefully (and it is actually the case), one can devise strategies that attain low regret in this scenario. However, one might say that this assumption makes us lose a lot of modeling power. After all, in the spam filtering example, the set of possible answers (which is binary) is clearly non-convex. Fortunately, there are techniques to model non-convex problems into the OCO framework [58]. One such technique is to allow the player to randomize her choices. For example, we can allow the spam detector to predict the *probability* of a certain email being spam, by making the set of possible answers of the player the $[0, 1]$ interval.

Online convex optimization, after a period of mainly *ad hoc* development, started to see more general and coherent progress recently [25, 26, 47, 57]. This was mainly due to the use of powerful concepts and tools from classical convex analysis and optimization, chiefly the concept of duality. The latter is based on the cornerstone result that two disjoint convex sets can be separated by a hyperplane. This gives rise to a duality theory which allows new and unifying convergence analysis of the algorithms [36, 47], as well as new algorithms fundamentally based on duality itself [59]. This unifying view of the field is giving us a deeper understanding of the key challenges in the area, and it is revealing new ways of applying online convex optimization techniques to different problems.

Even though the first and more intuitive applications of online convex optimization were indeed to learning problems, the theoretical computer science community started successfully using OCO techniques in other problems. One of the driving factors of this phenomenon was the emergence of the *Big data* trend, in which the instances are huge, thus rendering even quadratic algorithms prohibitively slow, but inexact solutions are acceptable, even more so if they can be computed quickly. For instance, we can mention the application of the *Multiplicative Weights Update (MWU) Method* (see [6] for a description of the method and many other applications) to the maximum flow problem by Christiano et al. [28], and the design of a matrix version of MWU and its application in primal-dual approach to many combinatorial problems by Kale [43].

In this project we plan to study the blossoming field of online convex optimization and its applications, while trying to grasp its core concepts and ideas. Shortly, our project can be broken down into the following steps:

(i) to study thoroughly the basics of convex analysis that we need to properly understand the online convex optimization framework and algorithms;
(ii) to solidify our knowledge about the basics of classic convex optimization, since many online convex optimization algorithms use classic algorithms as inspiration;
(iii) to dive into online convex optimization through surveys of the field, getting to know the main techniques, results, and algorithms;
(iv) to familiarize ourselves with the state of the art of online convex optimization;
(v) to focus on one topic of interest from online convex optimization.

Currently, we are on step (iv), and in this text we present what we believe to be a representative summary of our current progress, with possible ideas for the very important step (v) at the end.

1.1. **Text Outline.** In Section 2 we formalize the Online Learning and Online Convex Optimization frameworks, and discuss some of the connections and related fields. In Section 3 we present some algorithms for online convex optimization, together with their analysis and some discussion about their main ideas. Finally, in Section 4 we discuss some potential future directions for this project. Additionally, in Appendix A we present (without proof) the main results from convex analysis that we need, with a focus on building intuition. On Table 1 we present the basic notation that will be used throughout the text. The reader may skip it for now, and refer to it later as needed.

TABLE 1. Basic Notation

$[n]$	$\coloneqq$	$\{1, \dots, n\}$ for each $n \in \mathbb{N}$
$[P]$	$\coloneqq$	1 if the predicate P is true, and 0 otherwise
X^Y	$\coloneqq$	the set of functions from the set Y to the set X
X^n	$\coloneqq$	$X^{[n]}$ for every $n \in \mathbb{N}$; note that $X^0 = \{\varnothing\}$
$\operatorname{Seq}(X)$	$\coloneqq$	$\bigcup_{n=0}^{\infty} X^n$
$X + Y$	$\coloneqq$	$\{ x + y : x \in X, y \in Y \}$ for any subsets X, Y of an Euclidean space
αX	$\coloneqq$	$\{ \alpha x : x \in X \}$ for every $\alpha \in \mathbb{R}$ and every subset X of an Euclidean space
$\oplus$	$\coloneqq$	the direct sum of two vectors or two sets of vectors
Δ_E	$\coloneqq$	$\{ x \in [0,1]^E : \mathbb{1}^\mathsf{T} x = 1 \}$, the simplex in $\mathbb{R}^E$, for every set E

2. Online Learning and Online Convex Optimization

Consider a scenario where an investor has to pick a single stock from a set A to buy in the beginning of the day, and at the end of the day she sees how much she has gained (or lost) by choosing this stock. This "game" is played each day, during T days. Clearly her goal is to maximize her gains (i.e. minimize her losses). The investor/player, feeling insecure about her stock market knowledge, enlists the help of a set E of experts. Now, at the beginning of each day, each expert e suggests a stock $x_e \in A$ for her to buy. She then needs to choose an expert e^* and follow her advice. At the end of the day, the gains of the stocks recommended by the experts are revealed to the player, and she collects the gains (or incurs the losses) of x_{e^*}.

Even though one can keep in mind the stock market intuition, we can formulate this problem independently: in each of T rounds the experts make their suggestions, represented by a vector $x \in A^E$, where each point in A is some kind of "action". After this, the player picks an expert $e^* \in E$, and then the costs $y \in [-1, 1]^E$ of following each expert are revealed to her, and she suffers the loss $y(e^*)$. This problem is known as *prediction with expert advice*, and it will be better formalized later in the text. This problem is part of an even more general framework known as online learning, which can be loosely described in the following way: in each of T rounds, a player receives a query from "nature". The player then predicts a point based on this query, and an enemy picks the "true answer". At the end of the round, the player suffers a loss which depends on the player's prediction and on the answer given by the enemy. The goal of the player is to minimize, if possible, her cumulative loss during the game.

In this section we describe the frameworks of online learning and online convex optimization (OCO), and we discuss some problems that fit in these broad settings. Our description is more formal than usual, while also being careful not to clutter the comprehension of the reader. The reason for the currently chosen formalism is to be able to state (and prove) claims that are usually loosely stated by some authors, which in many cases makes it hard to understand what the claims mean.

In addition, even though the focus of this project is on the OCO framework, as well as its algorithms and applications, taking some time to describe online learning will be useful for two main reasons. The first one is that important problems studied in online convex optimization are more naturally described in the online learning setting, and our distinction makes clear which modifications are needed in order to use OCO algorithms on them. The second one is that online learning and OCO are often not clearly described, making it hard to understand their differences. We try to make the differences clearer.

2.1. **Online Learning Setting.** Let us recall the intuitive situation that the online learning setting aims to study: a player is participating in a game made of a sequence of rounds with a competitor, who we call enemy since it can, and will in the analysis of the algorithms, be adversarial to the player. In round t, the environment/nature presents a query x_t from a set X. The player then picks a prediction d_t from a set D, and the enemy simultaneously picks the "true label" y_t from a set Y. At the end of the round, the player suffers a loss of $L(d_t, y_t)$, where L is some function fixed throughout the game. One important aspect of this game is that, in round t, the player and the enemy know all the queries from nature until round t, besides knowing the points played by each other until the round $t-1$. That is, in round t, both the player and the enemy know $x_1, \ldots, x_t$, the player knows $y_1, \ldots, y_{t-1}$, and the enemy knows $d_1, \ldots, d_{t-1}$, so they may adapt to each others' previous choices. We are interested in studying strategies for the player that minimize, in some sense, her losses. Let us now formalize this game.

An **online learning problem** is a quadruple (X, Y, D, L), where X, Y, and D are arbitrary sets which we call the **query**, **label** and **decision sets** of the problem, respectively, and $L\colon D \times Y \to \mathbb{R}$ is a function, which we call the **loss function** of the problem. Let $\mathcal{P} \coloneqq (X, Y, D, L)$ be an online learning problem. We associate with $\mathcal{P}$ the function $\mathrm{OL}_{\mathcal{P}}$, which receives the following parameters:

- NATURE$\colon \mathbb{N} \to X$, which we call **nature oracle**;
- PLAYER$\colon \mathrm{Seq}(X) \times \mathrm{Seq}(Y) \to D$, which we call **player oracle**;
- ENEMY$\colon \mathrm{Seq}(X) \times \mathrm{Seq}(D) \to Y$, which we call **enemy oracle**;
- $T \in \mathbb{N}$, which we call the number of **rounds** or **iterations**.

We define $\mathrm{OL}_{\mathcal{P}}$ in an "iterative" way in Algorithm 2.1. Our description is mainly based on [58], while making use of other sources as well [17, 37, 40].

Algorithm 2.1 Definition of $\mathrm{OL}_{\mathcal{P}}(\mathrm{NATURE}, \mathrm{PLAYER}, \mathrm{ENEMY}, T)$

Input: NATURE, PLAYER, and ENEMY which are a nature, player, and enemy oracles for $\mathcal{P}$, respectively, and $T \in \mathbb{N}$.
Output: $(x, d, y) \in X^T \times D^T \times Y^T$.
  for $t = 1$ to T **do**
    $x_t \coloneqq \mathrm{NATURE}(t)$
    $d_t \coloneqq \mathrm{PLAYER}\big((x_1, \ldots, x_t), (y_1, \ldots, y_{t-1})\big)$
    $y_t \coloneqq \mathrm{ENEMY}\big((x_1, \ldots, x_t), (d_1, \ldots, d_{t-1})\big)$
    $\ell_t \coloneqq L(d_t, y_t)$
  return (x, d, y)

In the definition of $\mathrm{OL}_{\mathcal{P}}$, for each $t \in [T]$ we say that ℓ_t is the **loss** (of the player oracle) at round t, and $\sum_{t=1}^{T} \ell_t$ is the **cumulative loss** (of the player oracle). The function $\mathrm{OL}_{\mathcal{P}}$ defines formally the online learning setting. For example, we can formulate the prediction with expert advice example with a set of actions A and a set of experts E as the online learning problem $\mathcal{P} \coloneqq (A^E, [-1, 1]^E, E, L)$, where $L(e^*, y) \coloneqq y(e^*)$ for every $y \in [-1, 1]^E$ and $e^* \in E$. Any "strategy" of a player can be naturally transformed into a player oracle for $\mathcal{P}$. Moreover, any sequence of vectors of expert advice and their respective costs can be encoded through nature and enemy oracles, respectively.

Devising player oracles for the online learning setting which attain low (sublinear in the number of rounds) cumulative loss for some general classes of online learning problems may seem like an interesting idea, even more so in the case where the nature and enemy oracles are adversarial and colluded. Not surprisingly, this is usually an impossible mission. For any given player oracle, one can create an enemy oracle that picks, at each round, the point that maximizes the loss of the player. This is possible because one can create an enemy oracle that "knows" the strategy of the player oracle. For example, let PLAYER be a player oracle for the experts problem. Note that one can create an enemy oracle that simulates PLAYER. Thus, knowing the expert that PLAYER picked, the enemy can output a vector such that the cost of that expert is equal to 1, making the cumulative loss of PLAYER equal to T after T rounds. We can actually formally define this worst-case enemy for general online learning problems. If X is a set and $x \in X^t$, denote by $x_{i:j}$ the restriction of x to the set $\{i, i+1, \ldots, j\}$. With that, we can define the adversarial enemy oracle $\text{ENEMY}^*_{\text{PLAYER}}$ for an online learning problem (X, Y, D, L): for every $t \in \mathbb{N}$, and for any sequences $x \in X^t$ and $d \in D^{t-1}$, the point $\text{ENEMY}^*_{\text{PLAYER}}(x_{1:t}, d_{1:t-1})$ is an (arbitrary) point from the set

$$\arg\max\{\, L(d_t, y) : y \in Y\}, \qquad \text{where } d_i \coloneqq \text{PLAYER}(x_{1:i}, y_{1:i-1}) \text{ for every } i \in [t].$$

The above argument shows that the worst-case cumulative loss of a player does not give us much information about her performance: both complicated and simple-minded player oracles are usually indistinguishable if we look only at the cumulative loss. Intuitively, the problem is that we are posing, in some sense, an unrealistic goal: to obtain low cumulative loss when even a player with hindsight could perform no better. A more informative metric of performance originated from game theory [26] is the notion of regret, which measures how better or worse the player oracle performs when compared to some function of queries to predictions. The name comes from the idea that the regret with respect to some function measures how "sorry" the player oracle is for not using this function as its strategy throughout the whole game.

Formally, for every function $h\colon X \to D$ and $T \in \mathbb{N}$, define the **regret** of PLAYER with respect to the function h (and w.r.t. the oracles ENEMY and NATURE) as

$$\text{Regret}_T(h) \coloneqq \text{Regret}_T(\text{PLAYER}, h) \coloneqq \sum_{t=1}^{T} \big(L(d_t, y_t) - L(h(x_t), y_t)\big),$$

where $(x, d, y) \coloneqq \text{OL}_{\mathcal{P}}(\text{NATURE}, \text{PLAYER}, \text{ENEMY}, T)$, omitting PLAYER from the parameter list when it is clear from context. Moreover, for every $\mathcal{H} \subseteq D^X$, we define the **regret** of PLAYER with respect to $\mathcal{H}$ (and w.r.t. the oracles ENEMY and NATURE) as

$$\begin{aligned}\text{Regret}_T(\mathcal{H}) &\coloneqq \text{Regret}_T(\text{PLAYER}, \mathcal{H}) \coloneqq \sup_{h \in \mathcal{H}} \text{Regret}_T(\text{PLAYER}, h) \\ &= \sum_{t=1}^{T} L(d_t, y_t) - \inf_{h \in \mathcal{H}} \sum_{t=1}^{T} L\big(h(x_t), y_t\big),\end{aligned}$$

where $(x, d, y) \coloneqq \text{OL}_{\mathcal{P}}(\text{NATURE}, \text{PLAYER}, \text{ENEMY}, T)$, omitting PLAYER from the parameter list when it is clear from context. In the machine learning community, the function h and the set $\mathcal{H}$ in the above definitions are usually called **hypothesis** and **hypothesis set**, respectively. Let us take a step back and look at the intuitive meaning of these regret functions. The regret of a player oracle with respect to a function h measures how much better the player would have performed had she used as her strategy throughout the game the function h. Similarly, the regret with respect to a hypothesis set $\mathcal{H}$ measures how much better the player would have performed had she used as her strategy throughout the game the hypothesis from $\mathcal{H}$ best suited for this game, that is, the one which performs best.

Given a class of learning problems and a hypothesis set, we are interested in finding player oracles that attain low regret w.r.t. this hypothesis set for any online learning problem from this class. Here,

we consider the regret of the player w.r.t. a hypothesis set $\mathcal{H}$ to be low if it is sublinear in T, which happens only if, for any $h \in \mathcal{H}$, the difference of the loss of the player at round t (i.e. ℓ_t) and the loss of the function h in the same round (i.e. $L(h(x_t), y_t)$) goes to zero as the number t goes to infinity. Intuitively, if $\ell_t - L(h(x_t), y_t) \to 0$ as $t \to +\infty$ for any $h \in \mathcal{H}$, it means that the player is being able to "learn" how to perform as well as the best hypothesis $\mathcal{H}$ for the game. Formally, we consider the regret of a player oracle w.r.t. a hypothesis set $\mathcal{H}$ to be low if it grows sub-linearly in T, that is,

$$\lim_{T\to\infty} \frac{\text{Regret}_T(\mathcal{H})}{T} = 0.$$

We will usually be interested in player oracles that attain low regret for any pair of enemy and nature oracles for the OL problem at hand.

Let us describe some examples of interesting classes online learning problems:

Prediction with expert advice: Consider the following problem, known as *online routing*: every day a driver has to pick one of many routes to go to work, wishing to minimize her total travel time[1]. A good measure of effectiveness of the driver's strategy is to compare the time she has spent driving with the time she would have spent had she chosen the best fixed route in hindsight. Even though it may not be clear at first, we can fit this problem into the prediction with expert advice problem described earlier in the text. The key idea is to model each route as being an expert whose cost at the end of the day/round is its travel time, and whose advice is void or meaningless. The crux of modeling problems as an experts problem is usually this choice of what the experts are. With this in mind, we can fit online routing in the prediction with expert advice problem: each route is an expert, the player suffers the loss of the expert she has chosen, and the cost of every expert is revealed after the player makes a choice. Formally, prediction with expert advice are the problems of the form $(A^E, [-1,1]^E, E, L)$, where A is some arbitrary set of possible "actions", E is a finite set of "experts", and $L(e^*, y) \coloneqq y(e^*)$ for every $e \in E$ and $y \in [-1,1]^E$. In this problem we are usually interested in devising player oracles that perform, in some sense, as well as the best expert in hindsight, even for adversarial enemy oracles. For example, in the online routing problem we want a strategy for the player that attains a total travel time close to the one of the best fixed route in hindsight. Formally, we want to construct player oracles that attain sub-linear regret with respect to the hypothesis set $\{\, x \in A^E \mapsto e : e \in E\}$.

Online classification: In this kind of problem the player oracle receives, in each round, a point x from an arbitrary set X. It then has to assign a class (or label) from a finite set Y, and then the enemy reveals the correct class where x belongs. The goal of the player oracle is to assign classes to the points of X in a way to approximately match the enemy oracle's classification. A concrete example is *spam filtering*, where at each round the player receives an e-mail, represented by a point x. The player has to classify it as spam or not, and the user reveals the true classification of the instance. Note that the user is not adversarial to the player in this case. Rather, the user is the one that wants the most for the predictions to be accurate. Instead the spammers, which in this case play the role of nature, are the ones that are adaptative and adversarial[2] to the player, the spam detector. Formally, an online learning problem is said to be an online classification problem when it has the form (X, Y, Y, L), where

[1] We suppose that, at the end of the day the driver gets to see the travel time of that day for each route, even though it is more realistic to suppose the player has no information about the routes except for the one she picked. Later in the text we will describe the bandit setting, which models situations where we do not have full information at the end of a round.

[2] In our framework nature cannot be adaptative to the players' choices since it is not aware of such decisions. However, spammers usually tend to modify the emails they send in a way that it does not look like a spam. Thus, in later rounds of the OL game, spammers will send emails which look like non-spam emails from previous rounds, still the user will classify it as spam. This way, from the point of view of the OL player, the enemy/user is the one who is being adversarial.

X is an arbitrary set, Y is some finite set, and $L\colon Y \times Y \to \mathbb{R}$ is some arbitrary function, although we usually have $L(d, y) \coloneqq [d \neq y]$. One specific type of online classification that deserves special attention is when $Y = \{0, 1\}$, which we call **online binary classification**. Note that the spam filtering problem is an example of online binary classification. For online classification the hypothesis sets used are very problem dependent.

Online regression: Consider the scenario in which a doctor has to determine the weight of her patients, who come one by one to her clinic, based on test results that she gets to see before receiving each patient. Even if she has no way to correctly guess the weight of the first patients, hopefully she starts to notice some dependence between the exam results and the patient weights. Not only that, she wants to shape the variable dependencies with some "simple" function, supposing that there is indeed some function of this type that approximately explains the weights of all her patients, including the ones she has not seen yet. This is an online regression problem, where one wants to predict some target value based on the value of a collection of variables, aiming to explain the data as well as the best function from some arbitrary set. Formally, online regression problems are the online learning problems of the form[3] $(\mathbb{R}^d, \mathbb{R}, \mathbb{R}, L)$, where $L\colon \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is some arbitrary function. Some common loss functions used for this type of problems are $(d, y) \in \mathbb{R}^2 \mapsto |d - y|$ and $(d, y) \in \mathbb{R}^2 \mapsto (d - y)^2$. The format of online regression problems may seem too broad, and indeed it is. The structure of these problems is usually contained into the hypothesis set $\mathcal{H}$. For example, if $\mathcal{H}$ had no structure whatsoever, e.g. $\mathcal{H}$ is the set of all functions from $\mathbb{R}^d$ to $\mathbb{R}$, it would be impossible for a player to have low regret. More frequently, one wants to compete against a more restricted class of functions. Some examples of hypothesis sets are $\{\, x \in X \mapsto w^\mathsf{T} x : w \in \mathbb{R}^d \}$ and $\{\, x \in X \mapsto x^\mathsf{T} W x + b^\mathsf{T} x : W \in \mathbb{R}^{d \times d}, b \in \mathbb{R}^d \}$, which we call **online linear regression** and **online quadratic regression**, respectively.

2.2. **Comparison with Statistical Learning Theory.** Until now we were considering that the points picked by nature and enemy were independent, or even adversarial to the player. Even though this is the focus of the text, a classical and related framework which changes this assumption is the statistical learning setting [15, 63]. In this case, we suppose that the pairs of query and label points are sampled from some probability distribution which is *unknown to the player*. More specifically, in statistical learning we have sets X, Y, and D, template random variables $\mathcal{X}$ and $\mathcal{Y}$, usually not independent, which take values on X and Y, respectively, and a class of functions $\mathcal{H} \subseteq D^X$. The task is, without knowledge of the distributions of $\mathcal{X}$ and $\mathcal{Y}$, and given a set $\{(x_1, y_1), \dots, (x_n, y_n)\}$ of n i.i.d. sampled copies of $\mathcal{X}$ and $\mathcal{Y}$, often called a *training set*, to pick a function $h \in \mathcal{H}$ that minimizes the *risk* (with respect to some function $L\colon D \times Y \to \mathbb{R}$)

$$\mathbb{E}[L(h(\mathcal{X}), \mathcal{Y})].$$

One example of a problem from this setting is a statistical version of the prediction with expert advice problem. Consider a set E of experts and a set A of actions. In this version of the problem, the predictions of the experts (and their respective costs) are bound to a probability distribution, that is, there is a distribution $\mathcal{Z}$ on $A^E \times [-1, 1]^E$ whence the training set is sampled. We are often interested in finding the expert that, in expectation, minimizes the loss given by the function $L(e, y) \coloneqq y(e^*)$ for every $(e, y) \in E \times [-1, 1]^E$. In the context of statistical learning, this is the same as picking a function in $\{\, x \in A^E \mapsto e : e \in E \}$ that minimizes the risk with respect to the loss function L.

There are two main differences between this setting and that of online learning. In the statistical case, we have strong statistical assumptions over the points sampled from $X \times Y$, while in the online case the points can be picked in an adversarial way. The other main difference is that in statistical

[3]Even though we define this problem using $\mathbb{R}^d$ as the query set, one can consider the more general case where the query space for the regression is an Euclidean space, e.g. the space of matrices with the trace inner product.

learning the player has access to all the points sampled from $X \times Y$ before picking a hypothesis, while in the online learning setting the player has to make a prediction (and thus suffer some kind of loss) for each round. Despite their similarities, note that the way that we measure progress in each of the settings (regret and risk) are not easily comparable. In the statistical case, the player tries to pick a *single* function that generalizes well, that is, that performs well in the "real world". In the online case, although the player suffers a loss at each round, she can, in some sense, change her strategy in the middle of the current game based on the points already seen. While strategies used in these frameworks can differ in approach, with some effort, algorithms for online learning that have low regret can be used for statistical learning in some cases to obtain low risk [24, 46].

2.3. **Impossibility of Sub-linear Regret and Randomization.** In the examples of online learning problems, online binary classifications may be one of the most natural problems from the online learning setting, even more so if one has already some experience with machine learning problems. However, the next proposition shows that it is usually impossible to have low regret in this problem, even for a very simple hypothesis set.

Proposition 2.1 (Cover's impossibility result [30])**.** Let X be a nonempty set. Define the learning problem $\mathcal{P} \coloneqq (X, \{0,1\}, \{0,1\}, L)$, where $L(d, y) \coloneqq |d - y|$ for every $d, y \in \{0,1\}$. Set $h_0(x) \coloneqq 0$ and $h_1(x) \coloneqq 1$ for every $x \in X$, and define $\mathcal{H} \coloneqq \{h_0, h_1\}$. Then, there are nature and enemy oracles of $\mathcal{P}$ such that, for any player oracle for $\mathcal{P}$ and $T \in \mathbb{N}$, we have $\operatorname{Regret}_T(\mathcal{H}) \geq T/2 - 1$.

The above proposition shows a problem in the model: binary classification is a simple and known problem, and being unable to solve it shows that the player is lacking some power. The problem arises from the possibility of the enemy predicting exactly which is the next player prediction. What if, instead of making predictions deterministically, the player decided only on a probability distribution over the choices, and left the actual choice for randomness to take care of? Let us take the online binary classification case as an example. Given a query, the player may think that there is a 60% chance of it being from the class 1, for example. If the player decides to deterministically pick the class in which she is more confident, the enemy will be able to exploit that. If, instead, we flip a biased coin which the enemy *does not have access to*, we take away part of this advantage. Randomizing the choices seems even more appealing when we have a greater number of choices and the confidence that the player in the information she has about each choice is small. In view of this discussion, we can change the model to allow the player oracle to randomize its predictions, and restrict the access to information of the enemy oracle: it will not have access to the "random bits" played. For example, consider a player oracle in the prediction with expert advice problem that, instead of choosing an expert deterministically, samples one from some probability distribution. The key here is that the enemy is able to simulate this player oracle and see the probability of each expert being sampled that round, but she does not know which expert gets sampled before making a decision.

Formally, let $\mathcal{P} \coloneqq (X, Y, D, L)$ be an online learning problem, let $(\Omega, \Sigma, \mathbb{P})$ be a probability space, and suppose D is equipped with a σ-algebra[4]. A **randomized player oracle** for $\mathcal{P}$ is a function $\overline{\text{PLAYER}}\colon \operatorname{Seq}(X) \times \operatorname{Seq}(Y) \to D^\Omega$ such that every function $F \in D^\Omega$ in the image of $\overline{\text{PLAYER}}$ is measurable, that is, F is a random variable over $(\Omega, \Sigma, \mathbb{P})$ that takes values in D. In Algorithm 2.2 we overload the definition of $\text{OL}_\mathcal{P}$ for randomized player oracles, making it clear which information each oracle has access to. Moreover, the definition of Regret_T for randomized player oracles is similar to the definition of Regret_T for deterministic player oracles seen earlier. Note, however, that Regret_T becomes a random variable for randomized player oracles.

[4]If D is a finite set or $\mathbb{R}$, there are natural σ-algebras over them. Namely, the power set of D and the (Borel) σ-algebra generated by the open intervals in the real line, respectively. Whenever we are in one of these cases, we assume D is equipped with such a σ-algebra, unless stated otherwise.

Algorithm 2.2 Definition of $[\mathrm{OL}_{\mathcal{P}}(\mathrm{NATURE}, \overline{\mathrm{PLAYER}}, \mathrm{ENEMY}, T)](\omega)$ (overloading $\mathrm{OL}_{\mathcal{P}}$)

Input: $T \in \mathbb{N}$, the functions NATURE, $\overline{\mathrm{PLAYER}}$, and ENEMY which are a nature, randomized player, and enemy oracles for $\mathcal{P}$, respectively, and $\omega \in \Omega$, where Ω is from the probability space $(\Omega, \Sigma, \mathbb{P})$.
Output: $(x, d, y) \in X^T \times D^T \times Y^T$.
  for $t = 1$ to T **do**
    $x_t \coloneqq \mathrm{NATURE}(t)$
    $D_t \coloneqq \overline{\mathrm{PLAYER}}\big((x_1, \ldots, x_t), (y_1, \ldots, y_{t-1})\big)$
    $y_t \coloneqq \mathrm{ENEMY}\big((x_1, \ldots, x_t), (D_1(\omega), \ldots, D_{t-1}(\omega))\big)$
    $\ell_t \coloneqq L\big(D_t(\omega), y_t\big)$
  return $(x, (D_1(\omega), \ldots, D_T(\omega)), y)$

At first glance, one may think that the player oracle does not gain much by randomizing its choices: the enemy can still simulate its behavior and pick a point that maximizes the loss of the player oracle. However, as already mentioned, even though the enemy oracle can simulate the behavior of the randomized player oracle, she does not have access to the "random bits" (represented by ω in the definition). Therefore, even though the enemy knows, in some sense, the probability distribution over the experts that the player oracle is using at each round, she does not know which expert was sampled.

The return value of the function $\mathrm{OL}_{\mathcal{P}}$ and the losses of the player oracle are random variables in the case of randomized player oracles, as well as the regret. Thus, bounds on the regret in this case are generally coupled with some kind of probabilistic information. In the randomized OL setting we are usually interested in bounding the worst-case expected regret $\mathbb{E}(\mathrm{Regret}_T(\mathcal{H}))$[5] or in bounds on the regret that hold with high probability. Later in the text we will show how to obtain $O(\sqrt{T})$ worst case expected regret in the case of online classification, where T is the number of rounds.

2.4. **Online Convex Optimization Setting.** A framework which has shown itself useful in the development of algorithms for online learning is online convex optimization, which we now describe in an intuitive way, leaving the formalization for later. Throughout the remaining of this section, $\mathbb{E}$ denotes an arbitrary euclidean space (finite-dimensional real vector space) equipped with an inner-product $\langle \cdot, \cdot \rangle$.

Similarly to the online learning setting, the OCO framework is a game played in rounds by a player and its enemy. At round t, the player picks a point x_t from a convex set $X \subseteq \mathbb{E}$, and the enemy picks, simultaneously, a convex function[6] $f_t \colon \mathbb{E} \to [-\infty, +\infty]$ from some set $\mathcal{F}$. At the end of the round, the player suffers the loss $f_t(x_t)$. Similarly to the online learning setting, at round t the player knows the previous functions $f_1, \ldots, f_{t-1}$ played by the enemy, and the enemy knows the previous points $x_1, \ldots, x_{t-1}$ played by the player. The goal of the player is to minimize, in some sense, the cumulative loss suffered along a sequence of T rounds. As one may already guess, minimizing the loss is impossible in the case of adversarial enemy oracles. Thus, we shall define regret for OCO in a way analogous to the regret of the online learning setting. Let us now formalize this setting.

An **online convex optimization (OCO) problem** is a pair $(X, \mathcal{F})$ where $X \subseteq \mathbb{E}$ is a convex set and $\mathcal{F} \subseteq [-\infty, +\infty]^X$ is a set of convex functions. Let $\mathcal{C} \coloneqq (X, \mathcal{F})$ be an online convex optimization problem. We associate with $\mathcal{C}$ the function $\mathrm{OCO}_{\mathcal{C}}$, which receives the following parameters:

- PLAYER: $\mathrm{Seq}(\mathcal{F}) \to X$, which we call **player oracle**;

[5] This is sometimes called *strong regret*. Some authors (e.g. [8]) bound the arguably easier-to-bound measure $\sup_{h \in \mathcal{H}} \mathbb{E}(\mathrm{Regret}_T(h))$. We skip the discussion of the differences among these (and others) measures of "regret" since it is not relevant for the purposes of this text.

[6] We will use extended-real-valued functions, a convention justified in Appendix A.

- ENEMY: $\mathrm{Seq}(X) \to \mathcal{F}$, which we call **enemy oracle**;
- $T \in \mathbb{N}$, which we call the number of **rounds** or **iterations**,

and outputs a point in $\mathrm{Seq}(X) \times \mathrm{Seq}(\mathcal{F})$. As in the case of online learning, we define the function $\mathrm{OCO}_{\mathcal{C}}$ in an iterative way in Algorithm 2.3.

Algorithm 2.3 Definition of $\mathrm{OCO}_{\mathcal{C}}(\mathrm{PLAYER}, \mathrm{ENEMY}, T)$

Input: Functions PLAYER and ENEMY which are player and enemy oracles for $\mathcal{C}$, respectively, and $T \in \mathbb{N}$.
Output: $(x, f) \in X^T \times \mathcal{F}^T$.
  for $t = 1$ to T **do**
    $x_t \coloneqq \mathrm{PLAYER}\big((f_1, \ldots, f_{t-1})\big)$
    $f_t \coloneqq \mathrm{ENEMY}\big((x_1, \ldots, x_{t-1})\big)$
    $\ell_t \coloneqq f_t(x_t)$
  return (x, f)

Again, in a similar way to the online learning case, we define the **regret** of a player oracle PLAYER in $\mathcal{C}$ with respect to a point $u \in \mathbb{E}$ (and w.r.t. the enemy oracle ENEMY) as

$$\mathrm{Regret}_T(u) \coloneqq \mathrm{Regret}_T(\mathrm{PLAYER}, u) \coloneqq \sum_{t=1}^{T} \big(f_t(x_t) - f_t(u)\big),$$

where $(x, f) \coloneqq \mathrm{OCO}_{\mathcal{C}}(\mathrm{PLAYER}, \mathrm{ENEMY}, T)$, omitting PLAYER from the parameter list when it is clear from context. Moreover, for every $U \subseteq \mathbb{E}$, we define the **regret** of PLAYER in $\mathcal{C}$ with respect to U (and w.r.t. the enemy oracle ENEMY) as

$$\mathrm{Regret}_T(U) \coloneqq \mathrm{Regret}_T(\mathrm{PLAYER}, U) \coloneqq \sup_{u \in U} \mathrm{Regret}_T(\mathrm{PLAYER}, u) = \sum_{t=1}^{T} f_t(x_t) - \inf_{u \in U} \sum_{t=1}^{T} f_t(u),$$

where $(x, f) \coloneqq \mathrm{OCO}_{\mathcal{C}}(\mathrm{PLAYER}, \mathrm{ENEMY}, T)$, omitting PLAYER from the parameter list when it is clear from context.

It is interesting to note that the regret for OCO is computed with respect to fixed points, whereas in the case of online learning regret is computed with respect to functions (or hypotheses). Although this may seem arbitrary at first, we can fit the OCO framework into the online learning setting (this reduction will be formalized soon) with a nature oracle which is just a constant function. On account of this nature oracle, each hypotheses in the regret from online learning will evaluate to only one point. Thus, the regret for these online learning problems is exactly the regret defined here for OCO problems.

2.5. **From Learning to Optimization.** The similarities between the frameworks of online learning and online convex optimization are not a coincidence: any online convex optimization problem can be written as an online learning problem. To see this, let $\mathcal{C} \coloneqq (X, \mathcal{F})$ be an online convex optimization problem, and set $\mathcal{P} \coloneqq (\{0\}, X, \mathcal{F}, L)$ where $L(x, f) \coloneqq f(x)$ for every $(x, f) \in \mathbb{E} \times \mathcal{F}$. Then it is easy to see that the regret in $\mathcal{C}$ of a player oracle with respect to a point $u \in X$ is exactly the same as the regret in $\mathcal{P}$ of the same player oracle (modifying it and the enemy oracle from $\mathcal{C}$ to ignore the points from the nature oracle) with respect to the hypothesis $0 \in \mathbb{N} \mapsto u$. Not only that, but since nature only picks the same point each round, the regret of the online learning problem w.r.t. a hypothesis set $\mathcal{H}$ is equivalent to the regret for the OCO problem w.r.t. the set $\{\, h(0) : h \in \mathcal{H} \} \subseteq \mathbb{E}$.

However, we are interested in going in the opposite way: we want to model problems from the online learning setting to the online convex optimization setting. The reason is that, as we are going to see later, there are algorithms for OCO that have sub-linear regret for very broad classes of problems. Thus, if we manage to model problems from online learning into the OCO framework,

we obtain sub-linear regret algorithms for these problems as well. Some problems from online learning, such as the online linear regression problem, fit seamlessly into the online convex optimization setting. The next proposition states the result formally, and we give a sketch of the proof.

Proposition 2.2. Let $\mathcal{P} \coloneqq (\mathbb{R}^d, \mathbb{R}, \mathbb{R}, L)$ be an online linear regression problem, where $L\colon \mathbb{R}\times\mathbb{R} \to \mathbb{R}$ is convex w.r.t. its first argument, and let $\mathcal{C} \coloneqq (\mathbb{R}^d, \mathcal{F})$ be the OCO problem where

$$\mathcal{F} \coloneqq \{\, w \in \mathbb{R}^d \mapsto L(x^\mathsf{T} w, y) : x \in \mathbb{R}^d, y \in \mathbb{R}\}.$$

If there is a player oracle $\text{PLAYER}_{\text{OCO}}$ for $\mathcal{C}$ and a function $R\colon \mathbb{N} \to \mathbb{R}$ such that, for any enemy oracle for $\mathcal{C}$, $\text{Regret}_T(\text{PLAYER}_{\text{OCO}}, \mathbb{R}^d) \le R(T)$, then there is a player oracle $\text{PLAYER}_{\text{OL}}$ for $\mathcal{P}$ such that, for any enemy and nature oracles for $\mathcal{P}$, we have $\text{Regret}_T(\text{PLAYER}_{\text{OL}}, \mathcal{H}) \le R(T)$, where $\mathcal{H} \coloneqq \{\, x \in X \mapsto w^\mathsf{T} x : w \in \mathbb{R}^d\}$.

Proof sketch. For every $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$, define the function $f_{(x,y)}$ given by $f_{(x,y)}(w) \coloneqq L(x^\mathsf{T} w, y)$ for every $w \in \mathbb{R}^d$. If $\text{PLAYER}_{\text{OCO}}$ is a player oracle for $\mathcal{C}$, we can define a player oracle $\text{PLAYER}_{\text{OL}}$ for $\mathcal{P}$ by

$$\text{PLAYER}_{\text{OL}}\big((x_1, \dots, x_T), (y_1, \dots, y_{T-1})\big) \coloneqq x_T^\mathsf{T} w_T,$$

for every $T \in \mathbb{N}$ and $(x, y) \in (\mathbb{R}^d)^T \times (\mathbb{R})^T$, where

$$w_T \coloneqq \text{PLAYER}_{\text{OCO}}\big(f_{(x_1,y_1)}, \dots, f_{(x_{T-1},y_{T-1})}\big).$$

Therefore,

$$\begin{aligned}
\text{Regret}_T(\text{PLAYER}_{\text{OL}}, \mathcal{H}) &= \sum_{t=1}^{T} L(d_t, y_t) - \inf_{h\in\mathcal{H}} \sum_{t=1}^{T} L(h(x_t), y_t) \\
&= \sum_{t=1}^{T} L(w_t^\mathsf{T} x_t, y_t) - \inf_{w\in\mathbb{R}^d} \sum_{t=1}^{T} L(w^\mathsf{T} x_t, y_t) \\
&= \sum_{t=1}^{T} f_{(x_t,y_t)}(w_t) - \inf_{w\in\mathbb{R}^d} \sum_{t=1}^{T} f_{(x_t,y_t)}(w).
\end{aligned}$$

Note that the formula in the right-hand side of the last equation is the regret of $\text{PLAYER}_{\text{OCO}}$ in $\mathcal{C}$ with a properly defined enemy oracle. Thus, if we have a bound on the worst case regret of $\text{PLAYER}_{\text{OCO}}$, we obtain a bound on the regret of $\text{PLAYER}_{\text{OL}}$. □

In contrast, there are some online learning problems that do not fit directly in the online convex optimization setting. One clear example is the online binary classification problem. At each round the player oracle needs to pick a point in the non-convex set $\{0, 1\}$. Besides, as we are going to see, there are player oracles for OCO that have sublinear regret. Hence, fitting online binary classification directly into the OCO setting would contradict Proposition 2.1.

Another example is the prediction with expert advice problem. Let $\mathcal{P} \coloneqq (A^E, [-1, 1]^E, E, L)$ be an instance of this problem. At each round, the player oracle in the online learning setting has to pick an expert from the non-convex set E. On the one hand, it is not clear how to model this problem directly into the OCO setting. On the other hand, the randomized version of the problem fits almost seamlessly. To sample an expert, the only information that we need is, essentially, the probability of each expert being sampled. This can be represented by a vector p in the simplex $\Delta_E \coloneqq \{\, x \in [0, 1]^E : \mathbb{1}^\mathsf{T} x = 1\}$. From this vector $p \in \Delta_E$, we can define the random variable R by $\mathbb{P}(R = e) = p_e$ for each $e \in E$, writing $R \sim p$ in this case. Moreover, note that for every $p \in \Delta_E$, by letting $R \sim p$ we have

$$\mathbb{E}[L(R, y)] = \sum_{e\in E} \mathbb{P}(R = e) y_e = \sum_{e\in E} p_e y_e = p^\mathsf{T} y, \qquad \forall y \in [-1, 1]^E.$$

Thus, one can note that if we have player oracles for OCO that have low regret for problems of the form $(\Delta_E, \{\, p \in \mathbb{R}^E \mapsto y^\mathsf{T} p : y \in [-1,1]^E\})$, we can devise randomized player oracles for the prediction with expert advice problem that have low expected regret.

3. Algorithms for OCO

In this section, we will describe some player oracles for OCO which have good regret bounds in some scenarios.

3.1. Follow the Regularized Leader.

Intuitively, at iteration t of a game for an OCO problem, a good choice of point by the player oracle would be one that minimizes the sum of the already-seen functions, that is, a point in $\arg\min_{x\in X} \sum_{i=1}^{t-1} f_i(x)$, where $X \subseteq \mathbb{E}$ is the set determined by the OCO problem, and $\{f_i\}_{i=1}^{t}$ are the functions already played by the enemy until iteration t. This approach is known as *Follow the Leader*, mainly due to its interpretation in the experts problems, where the point that minimizes the already-seen function is, in the deterministic case, the best expert so far. However, one can show that this strategy may fail even for an enemy playing only linear functions, and the reason is that the enemy can make the choices of the player oracle vary a lot by playing appropriate functions. Thus, a way to fix this is to add a special function R, a regularizer, which stabilizes our choices. This approach is known as *Follow the Regularized Leader* (FTRL) [37, 42, 58, 59], and the formal definition of the player oracle is made in Algorithm 3.1.

Before we continue, we need to address a technical problem. The function FTRL_R is not well-defined, since different choices of points in the set of minimizers could lead to different evaluations of FTRL_R given the same arguments. In order to fix that, not only for this definition but for other oracles that we will define throughout this section, we assume that every set A is equipped with an arbitrary relation $\leq_A$ which defines a total order of the elements on A. Thus, every time we have to pick a point from a set A, we will use the convention that such a point is the minimum with respect to $\leq_A$. This solves the technical problem, and, since the order is arbitrary, it does not restrict us in any way.

Algorithm 3.1 Definition of $\mathrm{FTRL}_R(f_1, \ldots, f_T)$

Input: Functions $R, f_1, \ldots, f_T$ for some $T \in \mathbb{N}$ from $\mathbb{E}$ to $[-\infty, +\infty]$ such that, for every $t \in [T+1]$, the function $R + \sum_{i=1}^{t-1} f_i$ is proper and its infimum over $\mathbb{E}$ is attained.
Output: $x_{T+1} \in \operatorname{dom} R \subseteq \mathbb{E}$
Let $x_{T+1} \in \arg\min_{x\in\mathbb{E}} \left(R(x) + \sum_{t=1}^{T} f_t(x) \right)$
return x_{T+1}

The next lemma is the fundamental result used in the analysis of FTRL algorithms.

Lemma 3.1 (Follow The Leader-Be The Leader Lemma, [42])**.** Let $T \in \mathbb{N}$, let $R, f_1, \ldots, f_T$ be proper functions from $\mathbb{E}$ to $[-\infty, +\infty]$ such that, for every $t \in [T+1]$, the function $R + \sum_{i=1}^{t-1} f_i$ is proper and its infimum over $\mathbb{E}$ is attained, and define $x_t \coloneqq \mathrm{FTRL}_R(f_1, \ldots, f_{t-1})$ for every $t \in [T+1]$. Then, for every $u \in \mathbb{E}$,

$$\sum_{t=1}^{T} (f_t(x_t) - f_t(u)) \leq \sum_{t=1}^{T} (f_t(x_t) - f_t(x_{t+1})) + R(u) - R(x_1). \tag{1}$$

Proof. First of all, note that by re-arranging (1) we get

$$R(x_1) + \sum_{t=1}^{T} f_t(x_{t+1}) \leq R(u) + \sum_{t=1}^{T} f_t(u), \qquad \forall u \in \mathbb{E}. \tag{2}$$

Thus, it suffices to prove that the above inequality holds, and we will do so by induction on T. For $T = 0$ the inequality holds since $x_1 \in \arg\min_{x\in\mathbb{E}} R(x)$ by definition. Suppose $T > 0$. Then, for every $u \in \mathbb{E}$,

$$R(x_1) + \sum_{t=1}^{T} f_t(x_{t+1}) = f_T(x_{T+1}) + R(x_1) + \sum_{t=1}^{T-1} f_t(x_{t+1}) \le f_T(x_{T+1}) + R(x_{T+1}) + \sum_{t=1}^{T-1} f_t(x_{T+1})$$
$$= R(x_{T+1}) + \sum_{t=1}^{T} f_t(x_{T+1}) \le R(u) + \sum_{t=1}^{T} f_t(u),$$

where in the first inequality we used the induction hypothesis with (2) specialized to $u = x_{T+1}$, and in the second we used that $x_{T+1} \in \arg\min_{x\in\mathbb{E}}(R(x) + \sum_{t=1}^{T} f_t(x))$ by definition. □

The above lemma is surprisingly general: it does not depend on convexity at all, so it can be used to analyze a wide variety of algorithms which fit the "FTRL framework". To obtain useful regret bounds from the above result, we need to bound the $f_t(x_t) - f_t(x_{t+1})$ terms. That is, FTRL has to balance two competing aspects of the algorithm. On the one hand, it has to minimize the raw value of the functions. On the other hand, it has to be stable in the sense that the values of f at consecutive iterates x_t and x_{t+1} shouldn't be too far apart. Here is where convexity starts to come in handy. If f_t is convex and subdifferentiable at x_t, we can bound this term by the subgradient inequality: if $g_t \in \partial f_t(x_t)$, then for any norm $\|\cdot\|$ on $\mathbb{E}$ we have

$$f_t(x_t) - f_t(x_{t+1}) \le \langle g_t, x_t - x_{t+1}\rangle \le \|g_t\|_*\|x_t - x_{t+1}\|,$$

where $\|\cdot\|_*$ is the dual norm as defined in Appendix A.7. This yields the following corollary.

Corollary 3.2. Let $R, f_1, \dots, f_T$ be proper convex functions from $\mathbb{E}$ to $[-\infty, +\infty]$ such that, for every $t \in [T+1]$, the function $R + \sum_{i=1}^{t-1} f_i$ is proper and its infimum over $\mathbb{E}$ is attained, and define $x_t := \mathrm{FTRL}_R(f_1, \dots, f_{t-1})$ for every $t \in [T+1]$. If $g_t \in \partial f_t(x_t)$ for every $t \in [T]$, then, for every $u \in \mathbb{E}$ and every norm $\|\cdot\|$ on $\mathbb{E}$,

$$\sum_{t=1}^{T}(f_t(x_t) - f_t(u)) \le \sum_{t=1}^{T}\|g_t\|_*\|x_t - x_{t+1}\| + R(u) - R(x_1).$$

3.2. **Subgradient Bounds and Lipschitz Continuity.** The crux of many proofs of convergence of online (and even classical) convex optimization algorithms is the bound of the $\|x_t - x_{t+1}\|$ terms, where x_t and x_{t+1} are consecutive iterates, and this part of the proof may vary wildly depending on the application. The following lemma encapsulates this part of the proof, and its own proof is quite elegant, relying on classic convex duality theory, mainly in a known dual relation between strongly convex and strongly smooth functions [41]. For details on these topics, we point the reader to Appendix A.

Lemma 3.3 ([47, Lemma 7])**.** Let $F\colon \mathbb{E} \to [-\infty, +\infty]$ and $f\colon \mathbb{E} \to [-\infty, +\infty]$ be closed proper convex functions such that $F + f$ is σ-strongly convex with $\sigma > 0$. Let $\bar{x} \in \arg\min_{x\in\mathbb{E}} F(x)$, and let $\bar{y} \in \arg\min_{x\in\mathbb{E}} F(x) + f(x)$. If $\mathrm{ri}(\mathrm{dom}\, F) \cap \mathrm{ri}(\mathrm{dom}\, f)$ is nonempty, then

$$\|\bar{x} - \bar{y}\| \le \frac{1}{\sigma}\|g\|_*, \qquad \forall g \in \partial f(\bar{x}).$$

Proof. Let $g \in \partial f(\bar{x})$, and define $\phi := F + f - \langle g, \cdot\rangle$. By Theorem A.8, we have

$$\partial\phi(x) = \partial F(x) + \partial f(x) - g, \qquad \forall x \in \mathbb{E}. \tag{3}$$

Since $\bar{x}$ minimizes F, we have $0 \in \partial F(\bar{x})$ by the definition of subgradient. Thus, since $g \in \partial f(\bar{x})$, we have $0 \in \partial\phi(\bar{x})$. Since F, f and $\langle g, \cdot\rangle$ are closed, and since $\mathrm{ri}(\mathrm{dom}\, F) \cap \mathrm{ri}(\mathrm{dom}\, f)$ is nonempty, we have that ϕ is closed as well (see [53, Theorem 9.3]). Thus, by Theorem A.7, we have $\bar{x} \in \partial\phi^*(0)$.

Similarly, since $\bar{y}$ minimizes $F + f$, we have $0 \in \partial(F + f)(\bar{y}) = \partial F(\bar{y}) + \partial f(\bar{y})$, where the equality holds by Theorem A.8. This with (3) yields $-g \in \partial\phi(\bar{y})$, and again by Theorem A.7 we have $\bar{y} \in \partial\phi^*(-g)$.

Since $F + f$ is σ-strongly convex and $-\langle g, \cdot\rangle$ is convex, we have that ϕ is also σ-strongly convex. Thus, by the strong convexity/smoothness duality (Theorem A.13), we have that ϕ^* is $(1/\sigma)$-strongly smooth w.r.t. $\|\cdot\|_*$. In particular, ϕ^* is differentiable on $\mathbb{E}$. Thus, Theorem A.9 together with the previous conclusions imply $\bar{x} = \nabla\phi^*(0)$ and $\bar{y} = \nabla\phi^*(-g)$. Finally, the fact that ϕ^* is $(1/\sigma)$-smooth implies $\|\bar{x} - \bar{y}\| = \|\nabla\phi^*(0) - \nabla\phi^*(-g)\| \leq \sigma^{-1}\|g\|_*$. □

With these lemmas, one can already expect the bounds on the regret of FTRL algorithms to depend on the (dual) norm of the subgradients of the functions given by the enemy. Thus, for these bounds to be meaningful, we may need to assume a bound on the norms of the subgradients. Although such an assumption may seem artificial at first glance, it happens to be somewhat natural due to an interesting connection with Lipschitz continuity, the latter being a traditional hypothesis in convergence proofs of many optimization algorithms. One may note at least one of the reasons why this is a sensible assumption: if the function can change drastically between two close points, intuitively, the algorithm will have a harder time optimizing over this function.

Let $\rho > 0$. A function $f\colon \mathbb{E} \to [-\infty, +\infty]$ is **ρ-Lipschitz continuous** on a set $X \subseteq \mathbb{E}$ w.r.t. a norm $\|\cdot\|$ if

$$|f(x) - f(y)| \leq \rho\|x - y\|, \qquad \forall x, y \in X.$$

The following theorem shows the effect of Lipschitz continuity on the norms of the subgradients.

Theorem 3.4. Let $X \subseteq \mathbb{E}$ be a convex set with nonempty interior, and let $f\colon \mathbb{E} \to [-\infty, +\infty]$ be a proper closed convex function which is ρ-Lipschitz continuous on X w.r.t. a norm $\|\cdot\|$. Then, for every $x \in X$ there is $g \in \partial f(x)$ such that $\|g\|_* \leq \rho$. Additionally, for every $x \in \operatorname{int} X$ we actually have $\partial f(x) \subseteq \{\, g \in \mathbb{E} : \|g\|_* \leq \rho\}$.

Proof. Let $x \in \operatorname{int} X$, let $g \in \partial f(x)$, and let $y \in \arg\max\{\, \langle g, u\rangle : u \in \mathbb{E}, \|u\| \leq 1\} + x$, that is, y is such that $\langle g, y - x\rangle = \|g\|_*$ and that $\|y - x\| = 1$. For every $\lambda \in [0, 1]$, define $z_\lambda \coloneqq x + \lambda(y - x)$. For $\lambda \in (0, 1)$ small enough, $z_\lambda \in \operatorname{int} X$, and then

$$\lambda\|g\|_* = \lambda\langle g, y - x\rangle = \langle g, z_\lambda - x\rangle \leq f(z_\lambda) - f(x) \leq \rho\|z_\lambda - x\| = \rho\lambda\|x - y\| \leq \rho\lambda.$$

Thus, $\|g\|_* \leq \rho$.

Now let $\bar{x} \in X$, let $\mathring{x} \in \operatorname{int} X$, and define $x_\lambda \coloneqq \mathring{x} + \lambda(\bar{x} - \mathring{x})$ for every $\lambda \in [0, 1]$. For every $\lambda \in [0, 1)$ we have $x_\lambda \in \operatorname{int} X$, and by Theorem A.6, there is $g_\lambda \in \partial f(x_\lambda)$. By the claim proved in the previous paragraph, $\|g_\lambda\|_* \leq \rho$ for every $\lambda \in [0, 1)$, thus $\{g_\lambda\}_{\lambda \in I}$ where $I \coloneqq \{\, 1 - \frac{1}{n} : n \in \mathbb{N} \setminus \{0\}\}$ is a bounded sequence. This implies that there is a sub-sequence that converges to a point $\bar{g} \in \mathbb{E}$ with $\|\bar{g}\|_* \leq \rho$. Moreover, one can prove that $\lim_{\lambda\uparrow 1} f(x_\lambda) = f(\bar{x})$ if f is closed (see [53, Thm. 7.5]). Thus, for every $z \in \mathbb{E}$, the limit of the subgradient inequality $\langle g_\lambda, z - x_\lambda\rangle + f(x_\lambda) \leq f(z)$ with λ going to 1 from the left yields $\langle \bar{g}, z - \bar{x}\rangle + f(\bar{x}) \leq f(z)$. That is, $\bar{g} \in \partial f(\bar{x})$. □

3.3. **A Sub-linear Regret Bound.** Finally, we are in place to prove the first meaningful bound on the regret of FTRL oracles.

Theorem 3.5. Let $\mathcal{C} \coloneqq (X, \mathcal{F})$ be an online convex optimization problem such that $X \subseteq \mathbb{E}$ is closed and nonempty, and that every $f \in \mathcal{F}$ is closed, proper, and subdifferentiable on X. Let $R\colon \mathbb{E} \to [-\infty, +\infty]$ be a proper σ-strongly convex function w.r.t. a norm $\|\cdot\|$ such that $\operatorname{dom} R \subseteq X$. Moreover, suppose that for any finite sequence $\{f_i\}_{i=1}^m$ with $f_i \in \mathcal{F}$ for each $i \in [m]$, the function $R + \sum_{i=1}^m f_i$ is proper and that its infimum over $\mathbb{E}$ is attained, and suppose that $(\bigcap_{f\in\mathcal{F}} \operatorname{ri}(\operatorname{dom} f)) \cap \operatorname{ri}(\operatorname{dom} R)$ is nonempty. Let $T \in \mathbb{N}$, let ENEMY be an enemy oracle for $\mathcal{C}$, let

$(x, f) \coloneqq \mathrm{OCO}_{\mathcal{C}}(\mathrm{FTRL}_R, \mathrm{ENEMY}, T)$, and let $g_t \in \partial f_t(x_t)$ for each $t \in [T]$. Then, for every $x^* \in X$,

$$\mathrm{Regret}_T(\mathrm{FTRL}_R, x^*) \le R(x^*) - \min_{x \in X} R(x) + \frac{1}{\sigma} \sum_{t=1}^{T} \|g_t\|_*^2.$$

In particular, if X has nonempty interior, each $f \in \mathcal{F}$ is also ρ-Lipschitz continuous on X w.r.t. $\|\cdot\|$, and $\theta \coloneqq \sup\{ \frac{1}{\sigma}(R(x) - R(y)) : x, y \in X\}$[7] is finite, then

$$\mathrm{Regret}(\mathrm{FTRL}_{R'}, X) \le 2\rho\sqrt{\theta T},$$

where $R' \coloneqq \big(\rho\sqrt{T}/(\sigma\sqrt{\theta})\big) R$.

Proof. First of all, note that $R + \sum_{i=1}^{t-1} f_i$ is proper and σ-strongly convex for each $t \in [T+1]$ by our assumption. Since X is closed, by Lemma A.11 we have that $\mathrm{FTRL}_R(f_1, \dots, f_{t-1})$ is well defined for each $t \in [T+1]$. Not only that, for each $t \in [T]$, since $(\bigcap_{f \in \mathcal{F}} \mathrm{ri}(\mathrm{dom}\, f)) \cap \mathrm{ri}(\mathrm{dom}\, R)$ is nonempty, Lemma 3.3 with $f = f_t$ and $F = R + \sum_{i=1}^{t-1} f_t$ yields

$$\|x_t - x_{t+1}\| \le \frac{1}{\sigma} \|g_t\|_*, \tag{4}$$

where we define $x_{T+1} \coloneqq \mathrm{FTRL}_R(f_1, \dots, f_T)$.

Define $\theta' \coloneqq \frac{1}{\sigma}(R(x^*) - R(x_1))$. By Corollary 3.2, for every $x^* \in X$ we have

$$\mathrm{Regret}(\mathrm{FTRL}_R, x^*) \le \sigma\theta' + \sum_{t=1}^{T} \|g_t\|_* \|x_t - x_{t+1}\| \overset{(4)}{\le} \sigma\theta' + \frac{1}{\sigma} \sum_{t=1}^{T} \|g_t\|_*^2.$$

This proves the first inequality, since x_1 attains $\min_{y \in X} R(y)$ by definition.

In particular, if X has nonempty interior, and if each $f \in \mathcal{F}$ is ρ-Lipschitz continuous, by Theorem 3.4 we have that for each $t \in T$ there is $g_t \in \partial f_t(x_t)$ such that $\|g_t\|_* \le \rho$. Using these subgradients with bounded dual norm in the above inequality yields

$$\mathrm{Regret}(\mathrm{FTRL}_R, x^*) \le \sigma\theta' + \frac{T\rho^2}{\sigma}.$$

Moreover, since R' is $(\rho\sqrt{T}/\sqrt{\theta})$-strongly convex, plugging R' into this inequality and taking the supremum with $x^* \in X$ yields the second inequality. □

Let us look at the problem of prediction with expert advice. As we have seen in the end of Section 2, in order to obtain a low-regret randomized player oracle for the experts problem $(A^E, [-1,1]^E, E, L)$, it suffices to devise a player oracle for the OCO problem $\mathcal{C} \coloneqq (\Delta_E, \mathcal{F})$, where we define the set $\mathcal{F}$ by $\mathcal{F} \coloneqq \{ p \in \mathbb{R}^E \mapsto y^\mathsf{T} p : y \in [-1,1]^E \}$. Let $y \in [-1,1]^E$ and define $f_y(x) \coloneqq y^\mathsf{T} x$ for every $x \in [-1,1]^E$. Note that, for every $u, v \in \mathbb{R}^E$ and for every norm $\|\cdot\|$ on $\mathbb{R}^E$,

$$|f_y(u) - f_y(v)| \le |y^\mathsf{T}(u - v)| \le \|y\|_* \|u - v\|. \tag{5}$$

Since the ℓ_2-norm is self-dual, from the above inequality we conclude that every function in $\mathcal{F}$ is $\sqrt{d}$-Lipschitz continuous w.r.t. $\|\cdot\|_2$, where $d \coloneqq |E|$. Let $T > 0$ and define the function $R \coloneqq (\sqrt{2dT})\big(\|\cdot\|_2^2 + \delta(\cdot \,|\, \Delta_E)\big)$. It is easy to verify that R is $\sqrt{2dT}$-strongly convex w.r.t. $\|\cdot\|_2$. Since $\inf_{x \in \Delta_E} R(x) = 1/2d$, by Theorem 3.5 we have, for every expert $i \in E$,

$$\mathrm{Regret}(\mathrm{FTRL}_R, e_i) \le \left(\frac{1}{2} - \frac{1}{2d}\right)\sqrt{2dT} + \frac{1}{2}\sqrt{2dT} \le \sqrt{2dT}.$$

where $\{ e_i \in \{0,1\}^E : i \in E \}$ is the set of canonical basis vectors in $\mathbb{R}^E$, and the regret is measured w.r.t. $\mathrm{OCO}_{\mathcal{C}}(\mathrm{FTRL}_R, \mathrm{ENEMY}, T)$ with ENEMY being an arbitrary enemy oracle for $\mathcal{C}$. It is natural to ask if this regret bound is optimal, especially since our choice of regularizer was arbitrary. It

[7] One may think of this value as the diameter of the set X measured through the lens of R.

turns out that the dependence on T on the bound given by Theorem 3.5 is optimal: there is an OCO problem $(X, \mathcal{F}')$ with each function in $\mathcal{F}'$ being Lipschitz continuous such that the worst-case regret of any player oracle in this problem is no better than $\Omega(\sqrt{T})$, where the constants hidden by the asymptotic notation may depend on other parameters of the problem, such as the dimension [2]. The fact that such an intuitive algorithm already attains optimal regret asymptotically (w.r.t. T) is surprising. Still, this lower bound says nothing about the dependence on the dimension, which can be high, even more so in machine learning applications.

However, a smarter choice of regularizer already improves (exponentially) the dependence of the regret bound for FTRL on the number of the experts problem, which can be interpreted as the dimension of the problem. Define $R(x) \coloneqq \sum_{i\in E}[x_i \neq 0]x_i \ln x_i + \delta(x \,|\, \Delta_E)$ for every $x \in \mathbb{R}^E$. One can prove[8] that R is 1-strongly convex w.r.t. $\|\cdot\|_1$. Moreover, since the dual norm of $\|\cdot\|_1$ is $\|\cdot\|_\infty$, by (5) we conclude that every function in $\mathcal{F}$ is 1-Lipschitz continuous w.r.t. $\|\cdot\|_1$. Finally, one may verify[9] that $R(x) - R(y)$ is at most $\ln d$ for any $x, y \in \Delta_E$. Define $R' \coloneqq \sqrt{(\ln d)T}R$. By Theorem 3.5, we have

$$\operatorname{Regret}(\operatorname{FTRL}_{R'}, e_i) \leq 2\sqrt{(\ln d)T}, \qquad \forall i \in E.$$

3.4. **Logarithmic Regret.** We have just seen that FTRL attains optimal regret (in the dependence of T) for OCO problems with Lipschitz continuous functions. Still, if we are dealing with a problem about which we know more about the functions the enemy is allowed to use, we may improve the regret bounds. Indeed, the next theorem shows a bound for OCO problems with strongly convex functions which is exponentially better than the one from Theorem 3.5.

Theorem 3.6. Let $\mathcal{C} \coloneqq (X, \mathcal{F})$ be an online convex optimization problem such that $X \subseteq \mathbb{E}$ is nonempty and closed, and that each $f \in \mathcal{F}$ is subdifferentiable on X and σ-strongly convex w.r.t. a norm $\|\cdot\|$ for some $\sigma > 0$. Moreover, suppose $\bigcap_{f\in\mathcal{F}} \operatorname{ri}(\operatorname{dom} f) \cap \operatorname{ri} X$ is nonempty and define $R \coloneqq \delta(\cdot \,|\, X)$. Let $T \in \mathbb{N}$, let ENEMY be an enemy oracle for $\mathcal{C}$, let $(x, f) \coloneqq \operatorname{OCO}_{\mathcal{C}}(\operatorname{FTRL}_R, \text{ENEMY}, T)$, and let $g_t \in \partial f_t(x_t)$ for each $t \in [T]$. Then

$$\operatorname{Regret}(\operatorname{FTRL}_R, X) \leq \sum_{t=1}^{T} \frac{1}{\sigma t}\|g_t\|_*^2.$$

In particular, if X has nonempty interior and every function in $\mathcal{F}$ is ρ-Lipschitz continuous on X w.r.t. $\|\cdot\|$, then $\operatorname{Regret}(\operatorname{FTRL}_R, X) \leq \rho^2(1 + \ln T)/\sigma$ for every enemy oracle for $\mathcal{C}$.

Proof. For each $t \in \{0\} \cup [T]$, define $F_t \coloneqq R + \sum_{i=1}^{t} f_i$. Since, for each $t \in [T]$, the function f_t is σ-strongly convex, we have that F_t is $t\sigma$-strongly convex. Moreover, by assumption we have that $\operatorname{ri}(\operatorname{dom} F_t) \cap \operatorname{ri}(\operatorname{dom} f_t)$ is nonempty for each $t \in [T]$. Thus, since $x_t \in \arg\min_{x\in\mathbb{E}} F_{t-1}$ and $x_{t+1} \in \arg\min_{x\in\mathbb{E}} F_{t-1} + f_t = \arg\min_{x\in\mathbb{E}} F_t$ for each $t \in [T]$, by Lemma 3.3 we have

$$\|x_t - x_{t+1}\| \leq \frac{1}{t\sigma}\|g_t\|_*, \qquad \forall t \in [T].$$

Therefore, by Corollary 3.2,

$$\operatorname{Regret}(\operatorname{FTRL}_R, X) \leq \sum_{t=1}^{T} \|g_t\|_*\|x_t - x_{t+1}\| \leq \sum_{t=1}^{T} \frac{1}{\sigma t}\|g_t\|_*^2.$$

In particular, if every function in $\mathcal{F}$ is ρ-Lipschitz continuous on X w.r.t. $\|\cdot\|$, Theorem 3.4 together with the bound of the harmonic series $\sum_{t=1}^{T} 1/t \leq (1 + \ln T)$ yields the stated regret bound. □

[8] In order to prove this result, one can show that $\nabla^2 R(x) - I$ is positive semidefinite for every $x \in \Delta_E$, which can be shown to imply 1-strong convexity.

[9] Since $R(x) \leq 0$ for any $x \in \Delta_E$, it suffices to show that $\bar{x} \coloneqq d^{-1}\mathbb{1}$ attains $\inf_{x\in\Delta_E} R(x)$, where $\mathbb{1}$ is the vector with 1 in each entry. Using Theorem A.10, the latter claim boils down to showing that $\nabla R(\bar{x})^{\mathsf{T}}(u - \bar{x}) \geq 0$ for every $u \in \Delta_E$. Interestingly, the latter inequality holds as an equation in this case.

3.5. **Lazy Online Mirror Descent.** We have shown that Follow the Regularized Leader algorithms yield good regret bounds. Not only that, the description of the method is fairly straightforward. In spite of that, it is not clear how to efficiently compute FTRL_R in general. Not only that, in many applications one has access to the functions through "first-order oracles", that is, given a point x and a function f, one may compute $f(x)$ together with a subgradient of f at x. In these cases, one usually wants to make a constant number of queries to the oracle at each iteration. Thus, it would be interesting to have an algorithm which deals with the subgradients directly, without requiring optimization over the given functions. In Algorithm 3.2 we define the Lazy Online Mirror Descent (LOMD) oracle. Mirror Descent is a generalization of the Gradient Descent technique for optimization that was first proposed by Nemirovski and Yudin [48]. There is a very interesting intuition behind Mirror Descent algorithms, but before diving into that, let us show a deep connection between Lazy Online Mirror Descent and Follow the Regularized Leader algorithms.

Algorithm 3.2 Definition of $\mathrm{LOMD}_R(f_1, \dots, f_T)$

Input: Proper convex functions $R, f_1, \dots, f_T$ from $\mathbb{E}$ to $[-\infty, +\infty]$ such that R is strongly convex, and that $f_1, \dots, f_T$ are subdifferentiable on $\operatorname{dom} R$.
Output: $x_{T+1} \in \operatorname{dom} R \subseteq \mathbb{E}$
  Let $\{x_1\} \coloneqq \arg\min_{x \in \mathbb{E}} R(x)$ and $y_1 \coloneqq 0$.
  for $t = 1$ to T **do**
    $y_{t+1} \coloneqq y_t - g_t$, where $g_t \in \partial f_t(x_t)$
    $x_{t+1} \coloneqq \nabla R^*(y_{t+1})$
  return x_{T+1}

Theorem 3.7. Let $f_1, \dots, f_T, R$ be proper convex functions from $\mathbb{E}$ to $[-\infty, +\infty]$, where R is strongly convex. Moreover, for every $t \in [T]$ let g_t be as defined in Algorithm 3.2 for $\mathrm{LOMD}_R(f_1, \dots, f_T)$, and define $h_t \coloneqq \langle g_t, \cdot \rangle$ for every $t \in [T]$. Then $\mathrm{LOMD}_R(f_1, \dots, f_T) = \mathrm{FTRL}_R(h_1, \dots, h_T)$.

Proof. For each $t \in [T+1]$, define y_t and x_t as in Algorithm 3.2, and define $s_t \coloneqq \sum_{i=1}^{t} g_t$. By a simple induction, we have $y_t \coloneqq -s_{t-1}$ for every $t \in T$. That is, $\nabla R^*(-s_T) = x_{T+1} = \mathrm{LOMD}_R(f_1, \dots, f_T)$. Thus, by Theorem A.7 we have that x_{T+1} attains

$$\sup_{x \in \mathbb{E}} \langle x, -s_T \rangle - R(x) = -\inf_{x \in \mathbb{E}} R(x) + \langle x, s_T \rangle = -\inf_{x \in X} R(x) + \sum_{t=1}^{T} h_t(x),$$

that is, $x_{T+1} \in \arg\min_{x \in \mathbb{E}} R(x) + \sum_{t=1}^{T} h_t(x)$. Since the minimizer is unique and attained by Lemma A.11, this concludes our proof. □

By Theorem 3.4, a proper convex function $f \colon \mathbb{E} \to [-\infty, +\infty]$ which is Lipschitz continuous on a set $X \subseteq \mathbb{E}$ with nonempty interior has, at each point of X, at least one subgradient with small dual norm. Thus, if we have access to such special subgradients, LOMD_R has the same regret bound of FTRL_R given by Theorem 3.5 since the gradient of a linear function $\langle a, \cdot \rangle$ is a for any $a \in \mathbb{E}$ and since, by the subgradient inequality,[10]

$$\mathrm{Regret}_T(\mathrm{LOMD}_R, u) = \sum_{t=1}^{T} (f_t(x_t) - f_t(u)) \leq \sum_{t=1}^{T} \langle g_t, x_t - u \rangle \overset{\text{Thm. } 3.5}{=} \mathrm{Regret}_T(\mathrm{FTRL}_R, u),$$

where $g_t \in \partial f_t(x_t)$ is the subgradient picked in Algorithm 3.2 for each $t \in [T]$.

[10] One should note that the regrets in the inequality are for different OCO problems. In the left-hand side of the inequality the regret is for some OCO problem $\mathcal{C}$ and a fixed enemy, and the regret in the right-hand side is for the OCO problem $(\mathbb{E}, \mathbb{E}^*)$, where $\mathbb{E}^*$ is the set of linear functionals on $\mathbb{E}$, with an enemy that chooses, at round t, the function $\langle g_t, \cdot \rangle$. The details were left out for the sake of conciseness.

Note that by Theorem 3.4, every subgradient of f over $\operatorname{int} X$ has small norm. Thus, the assumption that we have access to such subgradients is not too strong, since the only points such that the subgradients can have large norm are the ones at the boundary of X.

In the example application of FTRL in the prediction with expert advice problem, we have used two different functions as regularizers. Namely the functions defined, for every $x \in \mathbb{R}^d$ and some $\eta > 0$, by

$$R_1(x) \coloneqq \frac{1}{\eta}\sum_{i=1}^{d}[x_i \neq 0]x_i \ln x_i + \frac{1}{\eta}\delta(x \,|\, X) \qquad \text{and} \qquad R_2(x) \coloneqq \frac{1}{2\eta}\|x\|_2^2 + \frac{1}{\eta}\delta(x \,|\, X), \tag{6}$$

where in the example we had $X = \Delta_E$ for a finite set E, but here we consider an arbitrary closed convex set $X \subseteq \mathbb{R}^d$. Not only that, the factor $1/\eta$ is used to control the strong convexity of the regularizer, and may be set in a way to derive optimal regret bounds. Let us see the algorithms that are generated by plugging these regularizers into the LOMD oracle. For simplicity, let us first consider $X = \mathbb{R}^d$.

For any norm $\|\cdot\|$, one can verify that $\left(\frac{1}{2}\|\cdot\|^2\right)^* = \frac{1}{2}\|\cdot\|_*^2$. Since the ℓ_2-norm is dual to itself, Theorem A.5 yields, for every $x \in \mathbb{R}^d$,

$$R_2^*(x) = \frac{1}{2\eta}\|\eta x\|_2^2 \implies \nabla R_2^*(x) = \eta x.$$

Using $R = R_2$ in Algorithm 3.2 yields $x_{t+1} = x_t - \eta g_t$ for each $t \in [T]$, which is a subgradient descent step with step size η. Let us look now at the case where R_1 is used as a regularizer with $X = \mathbb{R}_+^d$, where $\mathbb{R}_+ \coloneqq \{\, \alpha \in \mathbb{R} : \alpha \geq 0\}$, since the function $h_\beta(\alpha) \coloneqq \alpha\beta - [\alpha \neq 0]\alpha \ln \alpha$, where $\beta > 0$, is not defined for non-positive values. Using Theorem A.5 together with the fact that, for any $\beta > 0$, the function h_β attains its supremum at $e^{\beta-1}$, we have, for every $x \in \mathbb{R}^d$,

$$R_1^*(x) = \frac{1}{\eta}\sum_{i=1}^{d} e^{\eta(x_i-1)} \implies (\nabla R_1^*(x))_i = \frac{e^{\eta(x_i-1)}}{\eta} \text{ for each } i \in [d].$$

By a simple induction, one can see that, for each $t \in [T]$ and $i \in E$, we have

$$x_{t+1}(i) = \eta^{-1}e^{\eta(y_{t+1}(i)-1)} = \eta^{-1}e^{\eta(y_t(i)-1)-\eta g_t(i)} = x_t(i)e^{-\eta g_t(i)}.$$

That is, Online Mirror Descent with R_1 as a regularizer becomes the algorithm known as *Exponentiated Gradient Descent*, *Hedge* or *Multiplicative Weights Update Method with exponential updates* [6, 35]. The update rules derived by these regularizers are simple to implement and very well known in the optimization community. Thus, one may hope to unify convergence proofs of many existent algorithms with a general convergence proof of Online Mirror Descent or Follow the Regularized Leader type algorithms. We will discuss this point in more details at the end of this section.

One may be wondering what happens when the set X in the definition of R_1 and R_2 is not the entire euclidean space (or the non-negative orthant). The following lemma shows that the effect of adding an indicator function of X to a differentiable[11] regularizer only changes the gradient of the conjugate by a Bregman projection onto X.

Proposition 3.8. Let $X \subseteq \mathbb{E}$ be nonempty, closed, and convex, let $\psi\colon \mathbb{E} \to [-\infty, +\infty]$ be a closed proper convex function such that ψ^* is differentiable on $\mathbb{E}$ and that ψ is differentiable on an open set $D \supseteq X$. Moreover, suppose that $\operatorname{ri} X \cap \operatorname{ri}(\operatorname{dom}\psi)$ is nonempty, and define $R \coloneqq \psi + \delta(\cdot \,|\, X)$. Then, for every $x^* \in \mathbb{E}$,

$$\nabla R^*(x^*) = \Pi_X^\psi(\nabla\psi^*(x^*)).$$

[11]The differentiability assumption is needed since it does not make sense to talk about Bergman divergence w.r.t. a non-differentiable function.

Proof sketch. Let $x, x^* \in \mathbb{E}$. By Theorem A.7 and Theorem A.8, $x = \nabla R^*(x^*)$ if and only if $x^* \in \partial R(x) = \nabla\psi(x) + N_X(x)$. This condition is equivalent to $\nabla\psi(\nabla\psi^*(x^*)) - \nabla\psi(x) \in N_X(x)$ since, by Theorem A.7, we have $x^* = \nabla\psi(\nabla\psi^*(x^*))$. Finally, we can apply Theorem A.10 to the Bregman projector applied to $\nabla\psi^*(x^*)$ since $X \subseteq D \subseteq \operatorname{ri}(\operatorname{dom}\psi)$. Therefore, we conclude that $\nabla\psi(\nabla\psi^*(x^*)) - \nabla\psi(x) \in N_X(x)$ if and only if $v = \Pi_X^\psi(\nabla\psi^*(x^*))$. □

To see the above theorem in action, let us look what happens to LOMD when X in (6) is an arbitrary nonempty set. We had seen that when we use R_2 as a regularizer in LOMD, the update of the iterates in Algorithm 3.2 were basic subgradients steps. With the above lemma, when X is an arbitrary convex set, the updates in Algorithm 3.2 become $\{x_t\} \coloneqq \arg\min_{x\in X}\|x - \eta y_t\|_2$ for every $t \in [T]$. One may also check that $y_t = \sum_{i=1}^t -g_i$ for each $t \in [T]$ in this case. Thus, when X is not the entire space, the algorithm does not exactly make a subgradient step at each iteration. The oracle is making gradient steps in the non-projected iterates y_t, but the gradients are based on the projected iterates x_t. This kind of projection is the reason why this is the *lazy* version of mirror descent. In the case of the function R_1 one can verify that the Bregman projector amounts to a normalization w.r.t. the ℓ_1-norm.

Algorithm 3.3 Definition of $\operatorname{EOMD}_\psi^X(f_1, \dots, f_T)$

Input: A nonempty closed convex set $X \subseteq \mathbb{E}$, proper convex functions $\psi, f_1, \dots, f_T$ from $\mathbb{E}$ to $[-\infty, +\infty]$ such that ψ is strongly convex and differentiable on an open set $D \supseteq X$, and that $f_1, \dots, f_T$ are subdifferentiable on $\operatorname{dom}\psi$.
Output: $x_{T+1} \in \operatorname{dom} R \subseteq \mathbb{E}$
Let $\{x_1\} \coloneqq \arg\min_{x\in X}\psi(x)$.
for $t = 1$ to T **do**
$y_{t+1} \coloneqq \nabla\psi(x_t) - g_t$, where $g_t \in \partial f_t(x_t)$
$x_{t+1} \coloneqq \Pi_X^\psi(\nabla\psi^*(y_{t+1}))$
return x_{T+1}

3.6. **Mirror Descent Intuition.** Let us now look at the intuition behind Online Mirror Descent, and in order to do so, we define the *eager* version of OMD in Algorithm 3.3, which will be the first to be discussed due to its similarity to the original idea of Nemirovski and Yudin [48]. The idea is that gradients are representations of the derivatives, which are linear functionals on $\mathbb{E}$. That is, the gradients represent elements from the dual space $\mathbb{E}^*$ of $\mathbb{E}$, the space of linear functions from $\mathbb{E}$ to $\mathbb{R}$. Thus, a gradient step style update such as $x_t - \eta\nabla f(x_t)$, where $x_t \in \mathbb{E}$ and $f\colon \mathbb{E} \to \mathbb{R}$ is differentiable, can be seen as actually dealing with the functionals $\langle x_t, \cdot\rangle$ and $\langle \nabla f(x_t), \cdot\rangle$. Still, the choice of corresponding x_t with $\langle x_t, \cdot\rangle$ was more or less arbitrary: for each differentiable function ψ, we can build the linear function $\langle \nabla\psi(x), \cdot\rangle$. Thus, we should chose a function ψ, our regularizer, to make such correspondences. With that connection between $\mathbb{E}$ and $\mathbb{E}^*$ made, the gradient step update becomes $\nabla\psi(x_t) - \eta\nabla f(x_t)$. Yet, this update produces a point y_{t+1} "in" the dual space $\mathbb{E}^*$, and we need to correspond it back to a point in the primal space. Such a correspondence should, intuitively, be the inverse of the mapping $x \in \mathbb{E} \mapsto \nabla\psi(x)$. If ψ is closed and strongly convex, by Proposition A.12 we know that ψ^* is differentiable everywhere, and by Theorem A.7 we have that $\nabla\psi^*$ is the inverse mapping of $\nabla\psi$. That is the reason (or at least one of them) why we use strongly convex functions as regularizers, even though the original specification of the algorithm does not require explicitly for the regularizer to be strongly convex. Thus, we correspond the point y_{t+1} in the dual with the point $\nabla\psi^*(y_{t+1})$ in the primal. There is, still, one last factor to discuss: the point $\nabla\psi^*(y_{t+1})$ may be outside of a set $X \subseteq \mathbb{E}$ where our optimization is actually taking place. Thus, we project $\nabla\psi^*(y_{t+1})$ onto X with the Bregman projector based on ψ, yielding a point $x_{t+1} \in X$. One may see why this is at least partially intuitive when we look at Proposition 3.8. After this projection,

we can again correspond x_{t+1} with a dual point through $\nabla\psi$ and make a gradient step, starting the process again.

The lazy version of mirror descent, also known by Nesterov's Dual Averaging, works exactly in the same way as the eager version when we do not need to project the primal points onto a set X. The difference happens when X is not the entire Euclidean space. In the eager version, from a primal point $x_t \in \mathbb{E}$ we have a dual point $\nabla\psi(x_t)$ from where we make the gradient step, which generates a dual point y_{t+1}. We then take y_{t+1} back to the primal space through $\nabla\psi^*$ and a Bregman projector. In the lazy version, the gradients of the objective function f are still calculated at the primal point x_t, but the gradient step is made from a current dual point y_t instead of doing it from $\nabla\psi^*(x_t)$. This can be seen by comparing the definitions of y_t in Algorithm 3.2 and in Algorithm 3.3.

3.7. **An Unifying Analysis.** Not surprisingly, one can derive regret bounds for EOMD that are similar to the ones the hold for LOMD [10]. However, differently from LOMD, the eager version of online mirror descent does not fit the FTRL framework smoothly. The ideal scenario would be to find a unique framework to fit these, and others, algorithms in order to unify convergence proofs and reveal interesting patterns. A recent survey [47] generalizes the FTRL to use different regularizers at each iteration. In this survey, the author proves a stronger version of Lemma 3.1 (which he calls by *Strong FTRL Lemma*) to fit his more general FTRL framework, and from that the author proves regret bounds for a wide variety of algorithms. For example, in this section we have derived player oracles which need knowledge of the number of round T to achieve optimal regret bounds. In spite of that, there are choices of step sizes which change at each iteration that eliminate this issue, and an adaptative FTRL analysis can capture this step size strategy option. One additional motivation to find a unified framework for OCO algorithms is to shed light into their inner workings in order to find interesting new applications and algorithms.

4. Future Directions

The content of this text is not a thorough description of everything that was studied in the project. Nevertheless, the topics presented here were selected in the hope of giving a feeling of what was covered until this point, so the reader can ponder with us the choices that lie ahead. At the moment, we are in a position which is both exciting and daunting. The field of online convex optimization is broad, with nice practical and theoretical applications, and benefits from the rich theory of convex analysis. Still, at the same time that this is exciting, it is impossible to hope to grasp the whole area and its ramifications in a couple of years. Thus, an important decision now is to choose some portion of the OCO field, or of some of its ramifications, to focus our attention on. We give two suggestions here, briefly talking about what each topic is about, and showing some of the current state of the area in each case. We do not intend to be thorough in neither of the topics that follow, since we have not taken the time to look more carefully at any one of them. Moreover, we are open to other exciting directions.

4.1. **Bandit Convex Optimization.** In the Online Learning and Online Convex Optimization settings the player has access to the points/functions picked by the enemy after each round. Yet, there are situations where such a "generous" assumption is unrealistic. For example, in many scenarios of Reinforcement Learning [27, 51], the agent has access only to the feedback of the action she chose, having no information of the feedback had she chosen another action. This is the scenario of Bandit Convex Optimization (BCO), a modification of the OCO framework. The BCO framework can be loosely described as the following multi-round game between a player and an adversary: at round t, the player has to pick a point x_t from a convex set X while, simultaneously, the adversary picks a convex function f_t. At the end of the round, the player suffers (and thus gets to know) the loss $f_t(x_t)$, *without having access to the function f_t itself.* One of the most famous problems from

BCO is the *Multi-armed Bandit (MAB) Problem*[12] (see [18]), where a player has to pick one of K choices at each round, observing its payoff afterwards. The name has its origin on the colloquial term "one-armed bandit" used for slot machines. Thus, the name "multi-armed bandit" comes from picturing each of the K choices the player has as being one slot machine. The multi-armed bandit problem received a lot of attention from web companies since it models quite well some problems of the area. One example is the problem of advertisement (ad) placement, where a website has to choose which ad to show to the next visitor, aiming to maximize the number of ad clicks.

The attentive reader may have noticed that the multi-armed bandit problem does not quite fit the BCO framework as we described it, since the set of choices for the player in this case is $[K]$, a non-convex set. Truthfully, the MAB problem is identical to the prediction with expert advice problem, but with different feedback and no kind of "advice" given by the arms[13]. Thus, in a similar way to the experts problem, the multi-armed bandit is more naturally described in a limited feedback version of the Online Learning setting, and we can *almost* model it as a BCO when the player is randomized. The problem in this case is that the feedback given to the player at each round is the cost of the sampled arm/expert, not the expected cost of the distribution over arms as in the experts case. Thus, one need to rely on random estimates of the arm costs, but we skip this discussion for conciseness sake.

For the sake of comparison, let us quickly formalize the BCO framework. A **bandit convex optimization (BCO) problem** is a pair $(X, \mathcal{F})$ where $X \subseteq \mathbb{E}$ is a convex set, and $\mathcal{F} \subseteq \mathbb{R}^X$ is a set of convex functions. Let $\mathcal{B} \coloneqq (X, \mathcal{F})$ be a bandit convex optimization problem. We associate with $\mathcal{B}$ the function $\mathrm{BCO}_{\mathcal{B}}$, which receives the following arguments:

- PLAYER: $\mathrm{Seq}(\mathbb{R}) \to X$, which we call **player oracle**;
- ENEMY: $\mathrm{Seq}(X) \to \mathcal{F}$, which we call **enemy oracle**;
- $T \in \mathbb{N}$, which we call the number of **rounds** or **iterations**,

and outputs a point in $\mathrm{Seq}(X) \times \mathrm{Seq}(\mathcal{F})$. In Algorithm 4.1 we define $\mathrm{BCO}_{\mathcal{C}}$ in an iterative way. Additionally, for any $x^* \in X$, and any convex set $U \subseteq X$, define $\mathrm{Regret}_T(\mathrm{PLAYER}, x^*)$ and

Algorithm 4.1 Definition of $\mathrm{BCO}_{\mathcal{B}}(\mathrm{PLAYER}, \mathrm{ENEMY}, T)$

Input: Functions PLAYER and ENEMY which are player and enemy oracles for $\mathcal{B}$, respectively, and $T \in \mathbb{N}$.
Output: $\mathrm{BCO}_{\mathcal{B}}(\mathrm{PLAYER}, \mathrm{ENEMY}, T)$.
  for $t = 1$ to T **do**
    $x_t \coloneqq \mathrm{PLAYER}\big((\ell_1, \dots, \ell_{t-1})\big)$
    $f_t \coloneqq \mathrm{ENEMY}\big((x_1, \dots, x_{t-1})\big)$
    $\ell_t \coloneqq f_t(x_t)$
  return (x, f)

$\mathrm{Regret}_T(\mathrm{PLAYER}, U)$ in a way analogous to the definition of regret in OCO.

Bandit problems deal with a dilemma faced in many reinforcement learning problems, the *exploration-exploitation trade-off*: at the same time the player wants to pick the choice she currently believes is the best, and exploit it, she also wants to explore other possible choices that might have a better payoff. Additionally, one may note that if the enemy could predict if the player would either exploit or explore, she could render the exploration useless. This, together with the inherent random nature of exploration, implies that player strategies for BCO problems are almost always randomized. Thus, in BCO we are usually interested in devising player oracles that attain low expected regret, or low regret with high probability. Sometimes bounding the expected regret can be hard, so we may

[12]Interestingly, the multi-armed bandit problem was first (formally) proposed in [52], initially with statistical assumptions over the payoffs of each choice, before the general framework of BCO was proposed.

[13]In a variation called "contextual bandits" [18], the arms do give "advice/context" vectors to the players.

try to bound what we call the *oblivious expected regret* of a player oracle with respect to a set U, defined as $\sup_{u\in U}\mathbb{E}[\text{Regret}_T(u)]$, which is less than or equal to the expected regret $\mathbb{E}[\text{Regret}_T(U)]$.

By comparing Algorithms 2.3 and 4.1 we can see the main difference, about which we already briefly talked about, between the OCO and BCO settings: in the latter the player only receives the values of the losses, while in the former the player had access to the functions themselves. Therefore, BCO is at least as hard as OCO for the player with respect to regret minimization. In this case, some natural questions are: is BCO strictly harder than OCO? If so, how much harder? The answer to the first question is positive, but obtaining matching upper and lower bounds for the (expected) regret of BCO has proved to be quite hard.

Let us look at the history of regret bounds for BCO, mainly based on [22]. We skip the history of the multi-armed bandit for conciseness, although we point the reader interested in the MBA history to [18] for a thorough survey and good introductions to other MBA and BCO variants. We will state bounds for an arbitrary BCO problem $\mathcal{B} \coloneqq (X, \mathcal{F})$ in T rounds, where $X \subseteq \mathbb{E}$, and $\mathbb{E}$ is d-dimensional. The first upper bounds on expected regret for the general version of BCO were done in [44], with $\tilde{O}(d^3T^{3/4})$ expected regret, and in [34], with $\tilde{O}(\sqrt{d}T^{3/4})$ expected regret. Both of them use random estimates of the gradients, and then utilize it in some OCO algorithm. For example, [34] uses online gradient descent with random estimates of the gradient. Some restricted cases of BCO received more attention and witnessed significant progress. For example, the case where $\mathcal{F}$ is a class of linear functions with bounded norm already models very interesting problems and was studied in many papers [1, 13, 19, 31, 32], achieving matching lower bounds (from [1, 32]) and upper bounds (from [19]) of $\tilde{O}(d\sqrt{T})$. When the functions in $\mathcal{F}$ are simultaneously strongly convex and smooth, [60] shows a $\Omega(d\sqrt{T})$ lower bound on the expected regret, and [38] shows a $\tilde{O}(d^{3/2}\sqrt{T})$ upper bound. Despite some progress in other cases with different assumptions on the class $\mathcal{F}$ (for examples, see [3, 4, 33, 55]), the bounds on the general case remained unchanged until 2014, when the authors of [20] proved a $\tilde{O}(\sqrt{T})$ bound on the case $d = 1$, although they use information-theoretic arguments, not showing any algorithm. Note that before this result, it was not even known if it was possible to achieve only a $\sqrt{T}$ dependence on T on the worst-case expected regret. This result was extended in [21], showing a $\tilde{O}(d^{11}\sqrt{T})$ upper bound for BCO problems. Only recently new upper bounds, with algorithms, were given to the general BCO problem. The authors in [39] show an (exponential-time) algorithm that attains $\tilde{O}(2^{d^4}\log^{2d}(T)\sqrt{T})$ expected regret, and [22] shows a $\tilde{O}(d^{9.5}\sqrt{T})$ upper bound on the expected regret, together with a polynomial-time algorithm, even though the latter may achieve slightly worse bounds. In [22] the authors conjecture a lower bound of $\Omega(d^{1.5}\sqrt{T})$ on the expected regret of the general BCO case, and that there is a version of their algorithm which attains this lower bound.

Deriving regret bounds for BCO seems hard, and there is room for improvement. Interesting paths to follow are to try to improve the bounds themselves for the general or some restricted case, or to look for more efficient algorithms for BCO, maybe even for a more restricted case.

4.2. **Boosting.** Boosting [56] is an extremely useful technique of machine learning, where one combines several *weak learners*, predictors which are only moderately accurate, in order to devise a *strong learner*, that is, a predictor with good accuracy or generalization capacity. Boosting is usually applied in an iterative way, where at each iteration one has several weak learners, and each makes good predictions only in some (small) subset of the instances. The trick is that the miss-classified examples give information about the weaknesses of the weak learners, and we can use this information to train a new (weak) model that performs well where the others were lacking.

Even though boosting is a machine learning technique, we may think about "boosting style" algorithms for non-learning problems: given an oracle that approximately solves a problem poorly, one calls it repeatedly in smart ways in order to produce a good approximate solution. This idea is not new, as can be seen, for instance, in [6], where several examples of applications of the Multiplicative Weights Update (MWU) Method are given, many of them fitting the boosting style. Interestingly,

the key ideas of these applications are not exactly the use of MWU, but the *way* they use it. Indeed, in these cases we have to be both the player and the enemy in the OCO framework. Not only that, usually the more adversarial the enemy is, the more the player will have to adapt, which implies in better final solutions. Since MWU is an OCO algorithm, we may think about applying other OCO algorithms in the same manner.

It is important to note that algorithms based on this idea usually yield approximate solutions. Still, even if we are looking at a problem for which there are efficient polynomial-time algorithms to solve it, one may be interested in devising an approximation algorithm if the latter is significantly more efficient. Such super-efficient algorithms became even more desirable with the emergence of the *Big data* phenomenon, in which the problem instances have a huge size, making even quadratic algorithms impractical.

Let us look at some interesting examples of the boosting idea. In [29], the authors devise an algorithm to find an approximately maximum flow in sub-quadratic time in the number of vertices. To do so, they iteratively compute *electrical flows*, flows which may not respect edge capacities but can be (approximately) computed in nearly-linear time (see [29, 61]), using MWU to penalize edges which have their capacities violated. In [7, 43], the authors devise a matrix version of the MWU method, and then apply it to obtain an approximate solver of semidefinite programs in a specified format. This algorithm heavily uses conic optimization duality ideas, iteratively building a primal solution, and relies on the existence of an oracle which either certifies the feasibility of the solution, or outputs information on how to improve the current candidate primal solution. They then build oracles for three graph problems, yielding good approximation algorithms for each of them. In [11], the authors want to solve robust optimization (RO) problems, which are optimization problems with additional parameters that determine some uncertainty over the constraints. As they explain, this problem is well studied, but the robust counterparts of optimization problems are usually much harder to solve than the original ones. A natural question then arises: is it possible to solve a RO problem if one only knows how to solve the original problem? They answer this question affirmatively, and they show two meta-algorithms based on gradient descent and Follow the (Perturbed)[14] Leader algorithms, respectively, that repeatedly call an oracle that solves the original optimization problem to build an approximate solution to the RO problem.

Let us look at one final, but astounding example. Batson, Spielman, and Srivastava, in their seminal work [9], give an algorithm to construct a spectral sparsifier of a graph with a number of edges linear in the number n of vertices in the graph. In [23], the authors point to a deep connection between the Batson-Spielman-Srivastava (BSS) algorithm and the Matrix MWU method (MMWUM) of Arora and Kale [7, 43]. Citing [23]:

> "Another virtue of our second solution is that it illustrates that the surprising Batson-Spielman-Srivastava (BSS) algorithm is actually closely related to MMWUM. In particular, the algorithms underlying our two solutions are identical, except for the use of slightly different potential functions. (...)
>
> It is remarkable that Batson, Spielman and Srivastava developed their algorithm from first principles, apparently without knowing this connection to established algorithmic techniques."

In spite of the striking similarities, the BSS algorithm was not a clean application of the MMWUM, so there was a missing piece in this connection. The authors of [5] completed the puzzle: they showed that MMWUM is a matrix version of FTRL/Mirror Descent with a certain regularizer, and that the BSS algorithm could also be seen as using FTRL with a different regularizer. With this view on the algorithm, they were able to choose a better regularizer and to improve the running time from $\Omega(n^4)$ to $O(n^{2+\varepsilon})$, where ε is an error parameter. It was key for their analysis a clean and general view of

[14] Follow the Perturbed Leader algorithms use, instead of a regularizer as in FTRL, a random perturbation to stabilize its choices.

FTRL and Mirror Descent, together with tighter analysis with "local norms", norms which depend on the iterates of the algorithms. Later, the authors of [45] used and improved ideas from [5] to achieve an almost-linear running time algorithm.

As pointed in [6], the MWU method was rediscovered many times, or some algorithms which could be seen as applications of MWU were developed obliviously to its existence. The example we gave of the progress made by [5] is a great example of how a good understanding of the OCO framework allowed for a major algorithmic development. Thus, we believe that there may be many other problems to be studied through the lens of online convex optimization, and contributions can range from the creation of brand new algorithms, to the analysis and refinement of existing ones.

References

[1] J. D. Abernethy, E. Hazan, and A. Rakhlin. "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*. Edited by R. A. Servedio and T. Zhang. Omnipress, 2008, pages 263–274. URL: http://colt2008.cs.helsinki.fi/papers/123-Abernethy.pdf (cited on page 22).

[2] J. D. Abernethy, P. Bartlett, A. Rakhlin, and A. Tewari. "Optimal Strategies and Minimax Lower Bounds for Online Convex Games". In: UCB/EECS-2008-19 (February 2008). URL: https://www2.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-19.pdf (cited on page 16).

[3] A. Agarwal, O. Dekel, and L. Xiao. "Optimal Algorithms for Online Convex Optimization with Multi-Point Bandit Feedback". In: *COLT 2010 - The 23rd Conference on Learning Theory, Haifa, Israel, June 27-29, 2010*. Edited by A. T. Kalai and M. Mohri. Omnipress, 2010, pages 28–40. URL: http://alekhagarwal.net/bandits-colt.pdf (cited on page 22).

[4] A. Agarwal, D. P. Foster, D. Hsu, S. M. Kakade, and A. Rakhlin. "Stochastic convex optimization with bandit feedback". In: *SIAM J. Optim.* 23.1 (2013), pages 213–240. URL: https://doi.org/10.1137/110850827 (cited on page 22).

[5] Z. Allen-Zhu, Z. Liao, and L. Orecchia. "Spectral sparsification and regret minimization beyond matrix multiplicative updates [extended abstract]". In: *STOC'15—Proceedings of the 2015 ACM Symposium on Theory of Computing*. ACM, New York, 2015, pages 237–245 (cited on pages 23, 24).

[6] S. Arora, E. Hazan, and S. Kale. "The multiplicative weights update method: a meta-algorithm and applications". In: *Theory Comput.* 8 (2012), pages 121–164 (cited on pages 2, 18, 22, 24).

[7] S. Arora and S. Kale. "A combinatorial, primal-dual approach to semidefinite programs [extended abstract]". In: *STOC'07—Proceedings of the 39th Annual ACM Symposium on Theory of Computing*. ACM, New York, 2007, pages 227–236. URL: https://doi.org/10.1145/1250790.1250823 (cited on page 23).

[8] J.-Y. Audibert, S. Bubeck, and G. Lugosi. "Regret in online combinatorial optimization". In: *Math. Oper. Res.* 39.1 (2014), pages 31–45. URL: http://dx.doi.org/10.1287/moor.2013.0598 (cited on page 9).

[9] J. D. Batson, D. A. Spielman, and N. Srivastava. "Twice-Ramanujan sparsifiers". In: *STOC'09—Proceedings of the 2009 ACM International Symposium on Theory of Computing*. ACM, New York, 2009, pages 255–262 (cited on page 23).

[10] A. Beck and M. Teboulle. "Mirror descent and nonlinear projected subgradient methods for convex optimization". In: *Oper. Res. Lett.* 31.3 (2003), pages 167–175. URL: https://doi.org/10.1016/S0167-6377(02)00231-6 (cited on page 20).

[11] A. Ben-Tal, E. Hazan, T. Koren, and S. Mannor. "Oracle-based robust optimization via online learning". In: *Oper. Res.* 63.3 (2015), pages 628–638. URL: https://doi.org/10.1287/opre.2015.1374 (cited on page 23).

[12] A. Ben-Tal and A. Nemirovski. *Optimization III*. 2013. URL: http://www2.isye.gatech.edu/~nemirovs/OPTIII_LectureNotes2015.pdf (cited on page 2).

[13] A. Blum and H. B. McMahan. "Online geometric optimization in the bandit setting against an adaptive adversary". In: *Learning theory*. Volume 3120. Lecture Notes in Comput. Sci. Springer, Berlin, 2004, pages 109–123. URL: https://doi.org/10.1007/978-3-540-27819-1_8 (cited on page 22).

[14] O. Bousquet, S. Boucheron, and G. Lugosi. "Introduction to Statistical Learning Theory". In: *Advanced Lectures on Machine Learning, ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*. Edited by O. Bousquet, U. von Luxburg, and G. Rätsch. Volume 3176. Lecture Notes in Computer Science. Springer, 2003, pages 169–207. URL: https://doi.org/10.1007/978-3-540-28650-9_8 (cited on page 1).

[15] O. Bousquet, U. von Luxburg, and G. Rätsch, editors. *Advanced Lectures on Machine Learning, ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*. Volume 3176. Lecture Notes in Computer Science. Springer, 2004. URL: https://doi.org/10.1007/b100712 (cited on page 7).

[16] S. Bubeck. "Convex Optimization: Algorithms and Complexity". In: *Foundations and Trends in Machine Learning* 8.3-4 (2015), pages 231–357. URL: https://doi.org/10.1561/2200000050 (cited on page 2).

[17] S. Bubeck. *Introduction to Online Optimization*. Princeton University, December 14, 2011. URL: http://www.cse.iitd.ac.in/~naveen/courses/CSL866/BubeckLectureNotes.pdf (cited on pages 2, 4).

[18] S. Bubeck and N. Cesa-Bianchi. "Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems". In: *Foundations and Trends in Machine Learning* 5.1 (2012), pages 1–122. URL: http://dx.doi.org/10.1561/2200000024 (cited on pages 21, 22).

[19] S. Bubeck, N. Cesa-Bianchi, and S. M. Kakade. "Towards Minimax Policies for Online Linear Optimization with Bandit Feedback". In: *COLT 2012 - The 25th Annual Conference on Learning Theory, June 25-27, 2012, Edinburgh, Scotland*. Edited by S. Mannor, N. Srebro, and R. C. Williamson. Volume 23. JMLR Proceedings. JMLR.org, 2012, pages 41.1–41.14. URL: http://www.jmlr.org/proceedings/papers/v23/bubeck12a/bubeck12a.pdf (cited on page 22).

[20] S. Bubeck, O. Dekel, T. Koren, and Y. Peres. "Bandit Convex Optimization: $\sqrt{T}$ Regret in One Dimension". In: *Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015*. Edited by P. Grünwald, E. Hazan, and S. Kale. Volume 40. JMLR Workshop and Conference Proceedings. JMLR.org, 2015, pages 266–278. URL: http://jmlr.org/proceedings/papers/v40/Bubeck15a.html (cited on page 22).

[21] S. Bubeck and R. Eldan. "Multi-scale exploration of convex functions and bandit convex optimization". In: *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*. Edited by V. Feldman, A. Rakhlin, and O. Shamir. Volume 49. JMLR Workshop and Conference Proceedings. JMLR.org, 2016, pages 583–589. URL: http://jmlr.org/proceedings/papers/v49/bubeck16.html (cited on page 22).

[22] S. Bubeck, R. Eldan, and Y. T. Lee. "Kernel-based methods for bandit convex optimization". In: *STOC'17—Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, New York, 2017, pages 72–85 (cited on page 22).

[23] M. K. de Carli Silva, N. J. A. Harvey, and C. M. Sato. *Sparse Sums of Positive Semidefinite Matrices*. October 2011. arXiv: 1107.0088 [cs.DM]. URL: http://arxiv.org/abs/1107.0088 (cited on page 23).

[24] N. Cesa-Bianchi, A. Conconi, and C. Gentile. "On the generalization ability of on-line learning algorithms". In: *IEEE Trans. Inform. Theory* 50.9 (2004), pages 2050–2057. URL: https://doi.org/10.1109/TIT.2004.833339 (cited on page 8).

[25] N. Cesa-Bianchi, K. Crammer, and F. Orabona. "A generalized online mirror descent with applications to classification and regression". In: *Mach. Learn.* 99.3 (2015), pages 411–435. URL: https://doi.org/10.1007/s10994-014-5474-8 (cited on page 2).

[26] N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006 (cited on pages 2, 5).

[27] Z. Chen, J. D. Johnson, and J. Li. "Reinforcement Learning: An Introduction: R.S. Sutton, A.G. Barto, MIT Press, Cambridge, MA 1998, 322 pp. ISBN 0-262-19398-1". In: *Neurocomputing* 35.1-4 (2000), pages 205–206. URL: https://doi.org/10.1016/S0925-2312(00)00324-6 (cited on page 20).

[28] P. Christiano, J. A. Kelner, A. Mądry, D. A. Spielman, and S.-H. Teng. *Electrical Flows, Laplacian Systems, and Faster Approximation of Maximum Flow in Undirected Graphs*. October 2010. arXiv: 1010.2921 [cs.DS]. URL: http://arxiv.org/abs/1010.2921 (cited on page 2). Complete version of "Electrical flows, Laplacian systems, and faster approximation of maximum flow in undirected graphs". In: *STOC'11—Proceedings of the 43rd ACM Symposium on Theory of Computing*. ACM, 2011, pages 273–281. STOC Best Paper Award.

[29] P. Christiano, J. A. Kelner, A. Mądry, D. A. Spielman, and S.-H. Teng. "Electrical flows, Laplacian systems, and faster approximation of maximum flow in undirected graphs". In: *STOC'11—Proceedings of the 43rd ACM Symposium on Theory of Computing*. ACM, 2011, pages 273–281. STOC Best Paper Award (cited on page 23).

[30] T. M. Cover. "Behavior of sequential predictors of binary sequences". In: *Trans. Fourth Prague Conf. on Information Theory, Statistical Decision Functions, Random Processes (Prague, 1965)*. Academia, Prague, 1967, pages 263–272 (cited on pages 2, 8).

[31] V. Dani and T. P. Hayes. "Robbing the bandit: less regret in online geometric optimization against an adaptive adversary". In: *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*. ACM, New York, 2006, pages 937–943. URL: https://doi.org/10.1145/1109557.1109660 (cited on page 22).

[32] V. Dani, T. P. Hayes, and S. Kakade. "The Price of Bandit Information for Online Optimization". In: *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*. Edited by J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis. Curran Associates, Inc., 2007, pages 345–352. URL: http://papers.nips.cc/paper/3371-the-price-of-bandit-information-for-online-optimization (cited on page 22).

[33] O. Dekel, R. Eldan, and T. Koren. "Bandit Smooth Convex Optimization: Improving the Bias-Variance Tradeoff". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. Edited by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett. 2015, pages 2926–2934. URL: http://papers.nips.cc/paper/5842-bandit-smooth-convex-optimization-improving-the-bias-variance-tradeoff (cited on page 22).

[34] A. D. Flaxman, A. T. Kalai, and H. B. McMahan. "Online convex optimization in the bandit setting: gradient descent without a gradient". In: *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. ACM, New York, 2005, pages 385–394 (cited on page 22).

[35] Y. Freund and R. E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting". In: *J. Comput. System Sci.* 55.1, part 2 (1997). Second Annual European Conference on Computational Learning Theory (EuroCOLT '95) (Barcelona, 1995), pages 119–139. URL: https://doi.org/10.1006/jcss.1997.1504 (cited on page 18).

[36] V. Gupta, T. Koren, and Y. Singer. *A Unified Approach to Adaptive Regularization in Online and Stochastic Optimization*. June 2017. arXiv: 1706.06569 [cs.LG]. URL: http://arxiv.org/abs/1706.06569 (cited on page 2).

[37] E. Hazan. "Introduction to online convex optimization". In: *Foundations and Trends® in Optimization* 2.3-4 (2016), pages 157–325. URL: http://ocobook.cs.princeton.edu/OCObook.pdf (cited on pages 1, 2, 4, 12).

[38] E. Hazan and K. Y. Levy. "Bandit Convex Optimization: Towards Tight Bounds". In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*. Edited by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. 2014, pages 784–792. URL: http://papers.nips.cc/paper/5377-bandit-convex-optimization-towards-tight-bounds (cited on page 22).

[39] E. Hazan and Y. Li. *An optimal algorithm for bandit convex optimization*. March 2016. arXiv: 1603.04350 [cs.LG]. URL: http://arxiv.org/abs/1603.04350 (cited on page 22).

[40] S. C. H. Hoi, D. Sahoo, J. Lu, and P. Zhao. *Online Learning: A Comprehensive Survey*. February 2018. arXiv: 1802.02871 [cs.LG]. URL: http://arxiv.org/abs/1802.02871 (cited on pages 1, 4).

[41] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari. "Regularization techniques for learning with matrices". In: *J. Mach. Learn. Res.* 13 (2012), pages 1865–1890 (cited on pages 13, 35).

[42] A. Kalai and S. Vempala. "Efficient algorithms for online decision problems". In: *J. Comput. System Sci.* 71.3 (2005), pages 291–307. URL: http://dx.doi.org/10.1016/j.jcss.2004.10.016 (cited on page 12).

[43] S. Kale. "Efficient Algorithms Using The Multiplicative Weights Update Method". PhD thesis. Princeton University, 2007 (cited on pages 2, 23).

[44] R. D. Kleinberg. "Nearly Tight Bounds for the Continuum-Armed Bandit Problem". In: *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*. 2004, pages 697–704. URL: http://papers.nips.cc/paper/2634-nearly-tight-bounds-for-the-continuum-armed-bandit-problem (cited on page 22).

[45] Y. T. Lee and H. Sun. *Constructing Linear-Sized Spectral Sparsification in Almost-Linear Time*. August 2015. arXiv: 1508.03261 [cs.DS]. URL: https://arxiv.org/abs/1508.03261 (cited on page 24).

[46] N. Littlestone. "From on-line to batch learning". In: *Proceedings of the Second Annual Workshop on Computational Learning Theory (Santa Cruz, CA, 1989)*. Morgan Kaufmann, San Mateo, CA, 1989, pages 269–284 (cited on page 8).

[47] H. B. McMahan. "A survey of algorithms and analysis for adaptive online learning". In: *J. Mach. Learn. Res.* 18 (2017), Paper No. 90, 50 (cited on pages 2, 13, 20).

[48] A. S. Nemirovsky and D. B. a. Yudin. *Problem complexity and method efficiency in optimization*. A Wiley-Interscience Publication. Translated from the Russian and with a preface by E. R. Dawson, Wiley-Interscience Series in Discrete Mathematics. John Wiley & Sons, Inc., New York, 1983, pages xv+388 (cited on pages 17, 19).

[49] Y. Nesterov. *Introductory lectures on convex optimization. A basic course*. Volume 87. Applied Optimization. Kluwer Academic Publishers, Boston, MA, 2004, pages xviii+236. URL: http://dx.doi.org/10.1007/978-1-4419-8853-9 (cited on pages 2, 35).

[50] Y. Nesterov. "Primal-dual subgradient methods for convex problems". In: *Math. Program.* 120.1, Ser. B (2009), pages 221–259. URL: https://doi.org/10.1007/s10107-007-0149-x (cited on page 34).

[51] P. Norvig and S. J. Russell. *Artificial Intelligence - A Modern Approach (3. internat. ed.)* Pearson Education, 2010. URL: https://doi.org/10.1016/j.artint.2011.01.005 (cited on page 20).

[52] H. Robbins. "Some aspects of the sequential design of experiments". In: *Bull. Amer. Math. Soc.* 58 (1952), pages 527–535. URL: https://doi.org/10.1090/S0002-9904-1952-09620-8 (cited on page 21).

[53] R. T. Rockafellar. *Convex analysis*. Princeton Landmarks in Mathematics. Reprint of the 1970 original, Princeton Paperbacks. Princeton, NJ: Princeton University Press, 1997, pages xviii+451 (cited on pages 13, 14, 28–34).

[54] R. T. Rockafellar and R. J.-B. Wets. *Variational analysis*. Volume 317. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. Springer-Verlag, Berlin, 1998, pages xiv+733. URL: https://doi.org/10.1007/978-3-642-02431-3 (cited on page 35).

[55] A. Saha and A. Tewari. "Improved Regret Guarantees for Online Smooth Convex Optimization with Bandit Feedback". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*. Edited by G. J. Gordon, D. B. Dunson, and M. Dudik. Volume 15. JMLR Proceedings. JMLR.org, 2011, pages 636–642. URL: http://www.jmlr.org/proceedings/papers/v15/saha11a/saha11a.pdf (cited on page 22).

[56] R. E. Schapire. "The Strength of Weak Learnability (Extended Abstract)". In: *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*. IEEE Computer Society, 1989, pages 28–33. URL: https://doi.org/10.1109/SFCS.1989.63451 (cited on page 22).

[57] S.-S. Shai. "Online Learning: Theory, Algorithms, and Applications". PhD thesis. The Hebrew University of Jerusalem, 2007 (cited on pages 1, 2).

[58] S. Shalev-Shwartz. "Online Learning and Online Convex Optimization". In: *Foundations and Trends® in Machine Learning* 4.2 (2011), pages 107–194. URL: http://dx.doi.org/10.1561/2200000018 (cited on pages 1, 2, 4, 12).

[59] S. Shalev-Shwartz and Y. Singer. "A primal-dual perspective of online learning algorithms". In: *Machine Learning* 69.2-3 (2007), pages 115–142. URL: http://dx.doi.org/10.1007/s10994-007-5014-x (cited on pages 2, 12).

[60] O. Shamir. "On the Complexity of Bandit and Derivative-Free Stochastic Convex Optimization". In: *COLT 2013 - The 26th Annual Conference on Learning Theory, June 12-14, 2013, Princeton University, NJ, USA*. Edited by S. Shalev-Shwartz and I. Steinwart. Volume 30. JMLR Workshop and Conference Proceedings. JMLR.org, 2013, pages 3–24. URL: http://jmlr.org/proceedings/papers/v30/Shamir13.html (cited on page 22).

[61] D. A. Spielman and S.-H. Teng. "Nearly-linear Time Algorithms for Graph Partitioning, Graph Sparsification, and Solving Linear Systems". In: *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*. STOC '04. New York, NY, USA: ACM, 2004, pages 81–90 (cited on page 23).

[62] L. G. Valiant. "A Theory of the Learnable". In: *Commun. ACM* 27.11 (1984), pages 1134–1142. URL: http://doi.acm.org/10.1145/1968.1972 (cited on page 1).

[63] V. Vapnik. "An overview of statistical learning theory". In: *IEEE Trans. Neural Networks* 10.5 (1999), pages 988–999. URL: https://doi.org/10.1109/72.788640 (cited on pages 1, 7).

Appendix A. Convex Analysis Prerequisites

Most of the algorithms for online convex optimization rely on concepts from convex analysis. Albeit for a complete grasp of the intuition and proofs of the next section we do recommend some familiarity with convex analysis, we will describe in this section the main definitions and results of the area used in this text. This section is heavily, but not uniquely, based on [53]. We do not intend to give proofs, but to describe the main ideas that we need. Our goal is twofold. For the reader not familiar with convex analysis, we aim to introduce the main concepts and intuition used in the algorithms described later in the text. For the more experienced reader, we aim to recall and inform about the main results needed. It is worth noting that we intend to write in the final dissertation a full chapter on the convex analysis prerequisites and to make it mostly self-contained, while maintaining readability.

A.1. **Basic Definitions.** Throughout this section, $\mathbb{E}$ is an euclidean space (finite dimensional real vector space) equipped with an inner-product $\langle \cdot, \cdot \rangle$.

A set $C \subseteq \mathbb{E}$ is **convex** if $\lambda C + (1-\lambda)C \subseteq C$ for any $\lambda \in [0,1]$, and C is **affine** if $\lambda C + (1-\lambda)C \subseteq C$ for any $\lambda \in \mathbb{R}$. That is, if the line segment between any two points in a set C is contained in C, then the set C is convex, and if, additionally, the line that passes through these points is contained in C, then the set C is affine. Interestingly, note that intersections of convex (affine) sets is convex (affine). Using the definition of convex sets we can define convex functions. This is useful to derive results (and intuition) about convex sets when dealing with convex functions.

Let $f \colon S \to [-\infty, +\infty]$ where $S \subseteq \mathbb{E}$. The **epigraph** of f is the set

$$\operatorname{epi} f \coloneqq \{\, x \oplus \mu \in S \oplus \mathbb{R} : f(x) \leq \mu \},$$

and f is **convex** if $\operatorname{epi} f$ is convex. One can prove that f is convex if and only if it satisfies the more familiar condition $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$ for every $x, y \in S$ and $\lambda \in [0,1]$. Note that the epigraph of f "lives" in a space with one extra dimension, namely $\mathbb{E} \oplus \mathbb{R}$. Intuitively, the epigraph is the set obtained from extruding upwards the graph of f indefinitely.

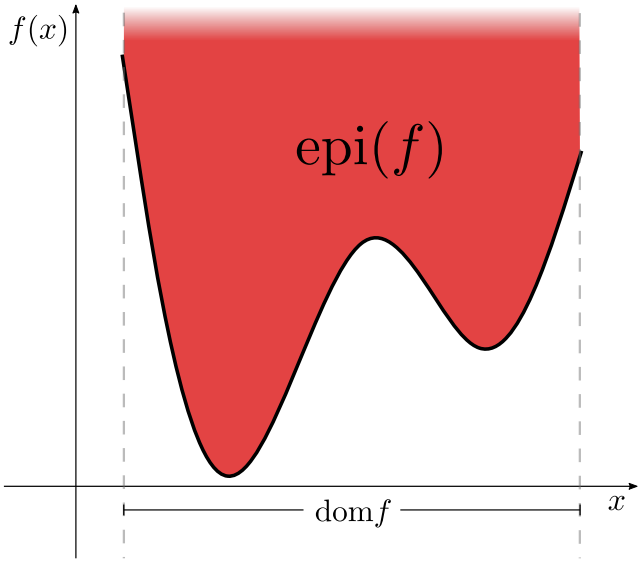

FIGURE 1. Illustration of the epigraph of a (non-convex) function f.

In this text we will follow the same convention used in [53]: all functions we deal with can be evaluated everywhere, even though they can take infinity values. We do not lose any generality with this assumption since a convex function g defined only in a subset $S \subseteq \mathbb{E}$ can be extended by setting $g(x) \coloneqq +\infty$ for every $x \in \mathbb{E} \setminus S$. This extension preserves the epigraph and, thus, convexity. The usefulness of this convention is that it makes many proofs and results less technical, just needing, in some cases, some care with the (non-)finiteness at some points. The **(effective) domain** of f is $\operatorname{dom} f \coloneqq \{\, x \in \mathbb{E} : f(x) \neq +\infty \}$, and f is **proper** if $\operatorname{dom} f$ is nonempty and $f(x) \neq -\infty$ for every $x \in \mathbb{E}$.

Some functions are very useful in stating and proving convex analysis results. For any set $C \subseteq \mathbb{E}$, define

- the **indicator function** $\delta(\cdot \,|\, C)$ of C by $\delta(x \,|\, C) \coloneqq 0$ for every $x \in C$ and $\delta(x \,|\, C) \coloneqq +\infty$ for every $x \in \mathbb{E} \setminus C$, and
- the **support function** $\delta^*(\cdot \,|\, C)$ of $C \subseteq \mathbb{E}$ by $\delta^*(x \,|\, C) \coloneqq \sup\{\, \langle x, y \rangle : y \in C \}$.

Given a set $S \subseteq \mathbb{E}$, we are sometimes interested in the smallest set with some property that contains S. For example, the smallest affine set that contains S tells us, in some sense, if we could fit the points of S in a space of smaller dimension. For any $S \subseteq \mathbb{E}$, define

- $\operatorname{aff} S \coloneqq \bigcap\{ M \subseteq \mathbb{E} : S \subseteq M \text{ and } M \text{ is affine}\}$, called the **affine hull** of S;
- $\operatorname{conv} S \coloneqq \bigcap\{ C \subseteq \mathbb{E} : S \subseteq C \text{ and } C \text{ is convex}\}$, called the **convex hull** of S.

A.2. **Basic Topological Properties.** Many results and ideas of convex analysis depend on the topology of the set (or function) we are dealing with. Set $\mathbb{B} \coloneqq \{ x \in \mathbb{E} : \langle x, x\rangle \leq 1\}$. For every $C \subseteq \mathbb{E}$, define

- the **closure** of C by $\operatorname{cl} C \coloneqq \bigcap_{\varepsilon>0}(C + \varepsilon\mathbb{B})$;
- the **interior** of C by $\operatorname{int} C \coloneqq \{ x \in C : \text{there exists } \varepsilon > 0 \text{ s.t. } x + \varepsilon\mathbb{B} \subseteq C\}$;
- the **relative interior** of C by $\operatorname{ri} C \coloneqq \{ x \in C : \text{there exists } \varepsilon > 0 \text{ s.t. } (x + \varepsilon\mathbb{B}) \cap \operatorname{aff} C \subseteq C\}$.

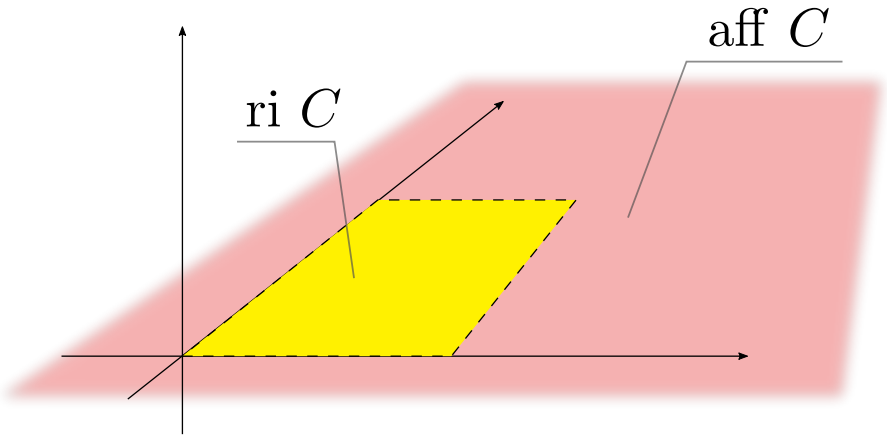

FIGURE 2. Illustration of the relative interior (yellow) and affine hull (red) of the set C, a closed two-dimensional rectangle in $\mathbb{R}^3$, whose interior is empty.

A set $C \subseteq \mathbb{E}$ is **relatively open** if $\operatorname{ri} C = C$. On the one hand, the concepts of closure and interior are classic topology concepts. On the other hand, one may not be familiar with the concept of relative interior, which plays a central role in many results of convex analysis. Loosely saying, the relative interior of a set would be its interior if it lived in a space with the "correct" dimension. For example, the interior of the line segment $[0, 1]$ is $(0, 1)$. However, the interior of $\{ (\lambda, \lambda) \in \mathbb{R}^2 : \lambda \in [0, 1]\}$ is empty. Geometrically these sets are similar. The issue is that the segment in $\mathbb{R}^2$ is not a two-dimensional object (its affine hull is not $\mathbb{R}^2$). One may be interested in the "correct" interior of the set, which in our latter case is $\{ (\lambda, \lambda) \in \mathbb{R}^2 : \lambda \in (0, 1)\}$, and this is exactly the relative interior of the set.

A concept similar to closure but for functions is the idea of lower semi-continuity. A function f is **lower semi-continuous** at $x \in \mathbb{E}$ if $f(x) = \liminf_{y\to x} f(y)$. Despite the definition not resembling anything about topology, the next theorem shows the connection to the concept of closure of sets.

Theorem A.1 ([53, Theorem 7.1])**.** Let $f \colon \mathbb{E} \to [-\infty, +\infty]$. The following are equivalent:

(i) f is lower semi-continuous throughout $\mathbb{E}$;
(ii) For every $\alpha \in \mathbb{R}$ the set $\{ x \in \mathbb{E} : f(x) \leq \alpha\}$ is closed;
(iii) The epigraph of f is a closed set in $\mathbb{E} \oplus \mathbb{R}$.

Thus, we can think about defining the closure of a function in a way analogous to the definition of set closure. Let $f \colon \mathbb{E} \to [-\infty, +\infty]$ be convex. The **closure** of f is the function $\operatorname{cl} f$ whose epigraph is $\operatorname{cl}(\operatorname{epi} f)$ if f is proper, and is the constant function $-\infty$ otherwise. Moreover, f is

closed if $\operatorname{cl} f = f$. We have to treat improper functions differently because taking the closure of the epigraph of improper functions may cause some unwanted abnormalities.

The next theorem shows the special usefulness of lower semi-continuity in the study of convex functions. It shows that (proper) convex functions are very close to being lower semi-continuous, possibly failing only at the boundary of the domain. This already hints that a convex function can behave more wildly on the boundary of the domain, and that it may require more attention at these boundary points.

Theorem A.2 ([53, Theorem 7.4])**.** Let $f\colon \mathbb{E} \to [-\infty, +\infty]$ be proper and convex. Then $\operatorname{cl} f$ is proper and convex, and $f(x) = (\operatorname{cl} f)(x)$ for every $x \in \operatorname{ri}(\operatorname{dom} f)$.

A.3. **Separation Theorem.** Separation is a simple, yet surprisingly powerful tool of convex analysis. It states that if the relative interior of two convex sets do not intersect, there is a hyperplane that passes between them, that is, there is a hyperplane which *separates* these two sets.

Theorem A.3 (Hyperplane Separation; see [53, Theorem 11.3])**.** Let $S, T \subseteq \mathbb{E}$ be nonempty and convex. If $\operatorname{ri}(S) \cap \operatorname{ri}(T) = \varnothing$, then there is $a \in \mathbb{E} \setminus \{0\}$ which satisfies the following properties:

(i) $\sup_{s\in S}\langle a, s\rangle \le \inf_{t\in T}\langle a, t\rangle$, and
(ii) $\inf_{s\in S}\langle a, s\rangle < \sup_{t\in T}\langle a, t\rangle$.

Even though we are not going to use this theorem directly, it gives the basis for a duality theory for convex sets. Given a convex set $C \subseteq \mathbb{E}$ and any point $x \in \mathbb{E} \setminus \operatorname{ri} C$, the above theorem states that there is a hyperplane separating them. With this idea, we can think about "describing" a convex set C only by the hyperplanes that separate it from the other points, that is, a kind of external representation of the set. For example, one can prove that any nonempty closed convex set is the intersection of all closed half-spaces that contain it. From all separating hyperplanes of C, the ones that are usually the most important are the *supporting hyperplanes*, that is, the hyperplanes that intersect at least one point of the boundary $C \setminus \operatorname{ri} C$. The application of the ideas of external representation of convex sets and of supporting hyperplanes to epigraphs of functions gives rise to the ideas of conjugate functions and subgradients, which are very useful in the creation and analysis of algorithms for (online and classical) convex optimization.

A.4. **Fenchel Conjugate.** Let $f\colon \mathbb{E} \to [-\infty, +\infty]$ be convex. The **(Fenchel) conjugate** of f is the function $f^*\colon \mathbb{E} \to [-\infty, +\infty]$ defined by

$$f^*(x^*) \coloneqq \sup_{x\in\mathbb{E}} \left(\langle x^*, x\rangle - f(x)\right), \qquad \forall x^* \in \mathbb{E}.$$

Note that $\operatorname{epi} f^*$ can be written as follows:

$$\begin{aligned}\operatorname{epi} f^* &= \bigcap_{x\in\mathbb{E}} \{\, x^* \oplus \mu^* \in \mathbb{E} \oplus \mathbb{R} : \langle x^*, x\rangle - f(x) \le \mu^* \}\\ &= \bigcap_{x\in\mathbb{E}} \{\, x^* \oplus \mu^* \in \mathbb{E} \oplus \mathbb{R} : \langle x^* \oplus (-1), x \oplus f(x)\rangle \le \mu^* \}.\end{aligned}$$

That is, $\operatorname{epi} f^*$ is the intersection of closed half-spaces. Thus, $\operatorname{epi} f^*$ is closed, and by Theorem A.1 we have that f^* is closed. To see the connection of conjugates with separating hyperplanes, note that if $x^* \oplus \mu^* \in \operatorname{epi} f^*$, by the right-hand side of the last equation above, we have that the hyperplane $\{\, x \oplus \mu \in \mathbb{E} \oplus \mathbb{R} : \langle x^* \oplus (-1), x \oplus \mu\rangle = \mu^* \}$ includes in one of its closed half-spaces the set $\operatorname{epi} f$. The idea is that each point of $\operatorname{epi} f^*$ comes from a hyperplane which, in one of its half-spaces, contains $\operatorname{epi} f$. One may already imagine that special properties may appear when we look in particular at the supporting hyperplanes, and this will be explored in the next subsession.

If $f\colon \mathbb{E} \to [-\infty, +\infty]$ is a proper convex function, then by the definition of conjugate we can easily get the *Fenchel-Young inequality*:

$$\langle x^*, x\rangle \le f^*(x^*) + f(x), \qquad \forall x^*, x \in \mathbb{E}.$$

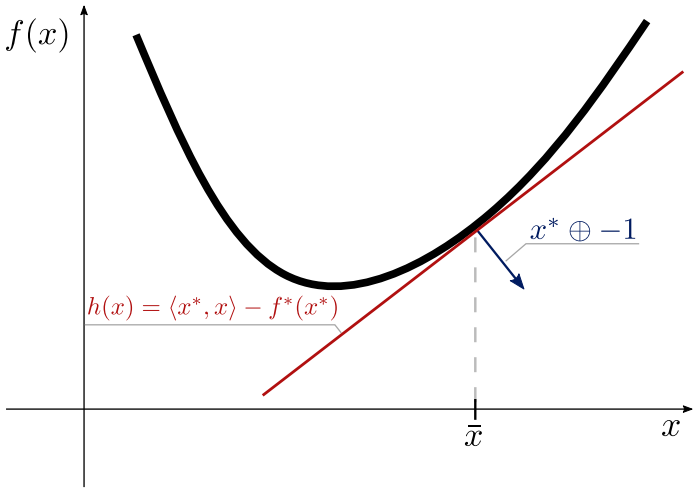

FIGURE 3. Illustration of the relation of supporting hyperplanes of epi f and the values of f^*.

The next theorem shows one expected property of a duality theory: the dual of the dual is, usually, the primal. In the case of convex functions, this holds smoothly if the function is closed.

Theorem A.4 ([53, Theorem 12.2])**.** Let $f\colon \mathbb{E} \to [-\infty, +\infty]$ be a convex function. Then f^* is a closed convex function, and proper if and only if f is proper. Moreover, $(\operatorname{cl} f)^* = f^*$ and $f^{**} = \operatorname{cl} f$.

Apart from their intuition, it is useful to know how to compute conjugate functions. For example, if $C \subseteq \mathbb{E}$ is convex, we can compute the conjugate of its indicator function. For each $x^* \in \mathbb{E}$, we have

$$(\delta(\cdot \,|\, C))^*(x^*) = \sup_{x\in\mathbb{E}} \langle x^*, x\rangle - \delta(x \,|\, C) = \sup_{x\in C} \langle x^*, x\rangle = \delta^*(x^* \,|\, C).$$

To compute conjugates of more complex functions, one can use results that relate the conjugates of such functions with the conjugates of simpler ones that compose them (see [53, Section 16]). Even though we will not dive into this rabbit role for the sake of conciseness, the following simple result in this direction will be useful in the text to compute conjugates of scaled functions.

Theorem A.5 ([53, Theorem 16.1])**.** If $f\colon \mathbb{E} \to [-\infty, +\infty]$ is proper and convex, then for any $\lambda \in \mathbb{R} \setminus \{0\}$ and $x^* \in \mathbb{E}$ we have $(\lambda f)^*(x^*) = \lambda f^*(\lambda^{-1} x^*)$.

A.5. **Subgradients.** Subgradients are a generalization of gradients for non-differentiable functions which is specially fruitful for convex functions. Let $f\colon \mathbb{E} \to \mathbb{R}$. A point $x^* \in \mathbb{E}$ is a **subgradient** of f at x if x^* satisfies, for every $z \in \mathbb{E}$, the *subgradient inequality*, that is,

$$f(z) \geq f(x) + \langle x^*, z - x\rangle.$$

The **subdifferential** of f at x is the set $\partial f(x)$ comprised of all the subgradients of f at x, and the **subdifferential** of f is the mapping $\partial f\colon x \in \mathbb{E} \mapsto \partial f(x)$. Moreover, f is **subdifferentiable** at a point $x \in \mathbb{E}$ if $\partial f(x) \neq \varnothing$. Before diving into the intuition of subgradients, one may wonder what is necessary for a function to have, at a given point, a subgradient. If the conditions for subdifferentiability are, in some sense, as strong as the ones for differentiability, we do not gain much in comparison. Luckily, the next theorem shows that convexity guarantees the existence of subgradients in a large portion of a function's domain.

Theorem A.6 ([53, Theorem 23.4])**.** Let $f\colon \mathbb{E} \to \mathbb{R}$ be proper and convex, and let $x \in \mathbb{E}$. If $x \notin \operatorname{dom} f$, then $\partial f(x) = \varnothing$, and if $x \in \operatorname{ri}(\operatorname{dom} f)$, then $\partial f(x) \neq \varnothing$. Finally, $\partial f(x)$ is nonempty and bounded if and only if $x \in \operatorname{int}(\operatorname{dom} f)$.

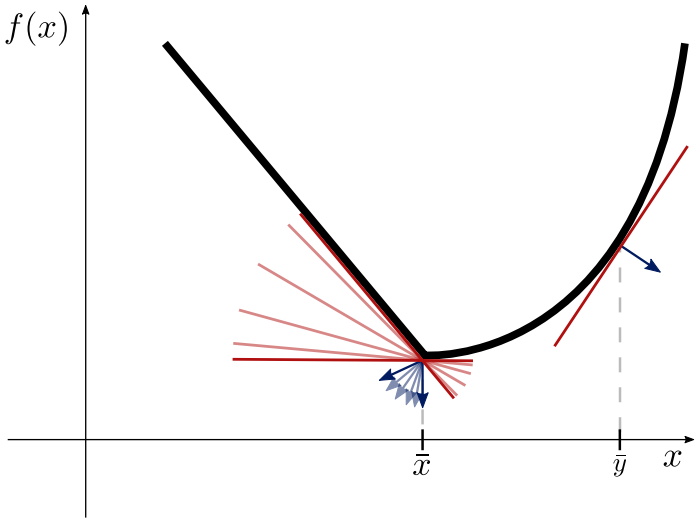

FIGURE 4. Illustration of the subgradients of f at the points $\bar{x}$, where f has multiple subgradients, and $\bar{y}$, where f is differentiable.

Similarly to classical gradients, subgradients have a strong relationship with tangent/supporting hyperplanes. Let $f\colon \mathbb{E} \to \mathbb{R}$, let $x \in \mathbb{E}$ be such that f is subdifferentiable at x, and let $x^* \in \partial f(x)$. By rewriting the subgradient inequality in a suitable way and by defining $\beta \coloneqq \langle x^*, x\rangle - f(x)$, we conclude that $\operatorname{epi} f$ is contained in one of the closed half-spaces define by the hyperplane

$$H \coloneqq \{\, z \oplus \mu \in \mathbb{E} \oplus \mathbb{R} : \beta = \langle x^* \oplus (-1), z \oplus \mu\rangle\}.$$

Not only that, note that $x \oplus f(x) \in H \cap \operatorname{epi} f$, that is, H is a supporting hyperplane of $\operatorname{epi} f$. With this intuition in mind, it is of no surprise that there is a rich connection between subgradients and Fenchel conjugates. The next theorem enumerates the most fundamental of these connections.

Theorem A.7 ([53, Theorem 23.5])**.** For any proper convex function $f\colon \mathbb{E} \to \mathbb{R}$ and any $x, x^* \in \mathbb{E}$, the following are equivalent:

(i) $x^* \in \partial f(x)$;
(ii) x attains the supremum $\sup_{z\in\mathbb{E}}(\langle z, x^*\rangle - f(z)) = f^*(x^*)$;
(iii) $f(x) + f^*(x^*) \le \langle x, x^*\rangle$;
(iv) $f(x) + f^*(x^*) = \langle x, x^*\rangle$.

Moreover, if $(\operatorname{cl} f)(x) = f(x)$, the following can be added to the list:

(v) $x \in \partial f^*(x^*)$;
(vi) x^* attains the supremum $\sup_{z\in\mathbb{E}}(\langle z^*, x\rangle - f^*(z^*)) = f(x)$;
(vii) $x^* \in \partial(\operatorname{cl} f)(x)$.

As an important example, let us compute the subdifferential of $\delta(\cdot \,|\, C)$, where $C \subseteq \mathbb{E}$ is nonempty and convex. By definition, $x^* \in (\partial\delta(\cdot \,|\, C))(x)$ if and only if $\delta(z \,|\, C) \ge \delta(x \,|\, C) + \langle x^*, z - x\rangle$ for every $z \in \mathbb{E}$. Since C is nonempty, this is equivalent to $x \in C$ and $0 \ge \langle x^*, z - x\rangle$. That is, $\partial\delta(\cdot \,|\, C))(x)$ is the set $N_C(x) \coloneqq \{\, d \in \mathbb{E} : \langle d, z - x\rangle \le 0 \text{ for every } z \in C\}$, which we call the **normal cone** of C at x. As in the case of conjugates, we may be interested in computing the subdifferential of complicated functions. The following theorem can be very useful in this task, and we will use it later in this section.

Theorem A.8 ([53, Theorem 23.8])**.** Let $f_1, \dots, f_m$ be proper convex functions on $\mathbb{E}$, and set $f \coloneqq f_1 + \cdots + f_m$. Then

$$\partial f_1(x) + \cdots + \partial f_m(x) \subseteq \partial f(x),$$

and equality holds if $\bigcap_{i\in[m]} \operatorname{ri}(\operatorname{dom} f_i) \neq \varnothing$.

Finally, one may be wondering if there are any connections between gradients and subgradients. The next theorem answers this natural question.

Theorem A.9 ([53, Theorem 25.1])**.** Let $f\colon \mathbb{E} \to \mathbb{R}$ be convex, and let $x \in \operatorname{dom} f$. Then f is differentiable at x if and only if $\partial f(x) = \{\nabla f(x)\}$.

A.6. **Optimality Conditions with Subgradients.** Arguably, one of the main reasons for convexity to be such a desirable element in optimization problems is that, for convex functions, local minimality is sufficient for global minimality. To see this, let $f\colon \mathbb{E} \to [-\infty, +\infty]$ be a proper convex function, let $x^* \in \mathbb{E}$ be a local minimum of f over $\mathbb{E}$, and let $y \in \operatorname{dom} f$. By the convexity of f, for every $\lambda \in (0, 1)$ we have

$$\frac{f(x^* + \lambda(y - x^*)) - f(x^*)}{\lambda} \leq f(y) - f(x^*).$$

For $\lambda \in (0, 1)$ small enough, the left hand side of the above inequality is non-negative, which implies $f(x^*) \leq f(y)$, that is, x^* is a global minimum.

Further, one is usually interested in necessary and sufficient conditions for minimality. Even though there are conditions of this type from Calculus, they depend on the differentiability of the functions. Since we are going to face non-differentiable functions quite often, it is useful to look at optimality conditions which do not depend on differentiability. As a warm up, let us consider the problem of minimizing a proper convex function $f\colon \mathbb{E} \to [-\infty, +\infty]$ over $\mathbb{E}$. A necessary and sufficient condition for a point $x^* \in \mathbb{E}$ to be a minimizer of f is $0 \in \partial f(x^*)$ by the mere definition of subgradient. A more interesting case is when we want to minimize f over a convex set $C \subseteq \mathbb{E}$. Note that this constrained problem is equivalent to minimizing the function $f + \delta(\cdot \,|\, C)$ over $\mathbb{E}$. Using this idea together with the subgradient condition for unconstrained minimization, the fact that the subdifferential of the indicator function is the normal cone and Theorem A.8, one can prove the following important theorem.

Theorem A.10 ([53, Theorem 27.4])**.** Let $C \subseteq \mathbb{E}$ be nonempty and convex, and let $f\colon \mathbb{E} \to [-\infty, +\infty]$ be proper and convex. A sufficient condition for a point $x \in \mathbb{E}$ to attain $\inf_{z\in C} f(z)$ is that $\partial f(x) \cap (-N_C(x))$ is nonempty. If $\operatorname{ri}(\operatorname{dom} f) \cap \operatorname{ri} C \neq \varnothing$, this condition is also necessary.

A.7. **Strongly Convex and Smooth Functions.** Sometimes, we will need to ensure that the functions we are handling are not "flat" or, in other words, we require them to have some curvature.

Let $\sigma > 0$. A function $\psi\colon \mathbb{E} \to [-\infty, +\infty]$ is **σ-strongly convex** on a convex set $X \subseteq \mathbb{E}$ (with respect to a norm $\|\cdot\|$) if for every $x, y \in X$ and $\lambda \in [0, 1]$ we have

$$\psi(\lambda x + (1 - \lambda)y) \leq \lambda\psi(x) + (1 - \lambda)\psi(y) - \lambda(1 - \lambda)\frac{\sigma}{2}\|x - y\|^2.$$

If the set X is not explicitly stated, we assume $X = \mathbb{E}$. Before looking at the intuition, let us look at one useful property of infima of strongly convex functions.

Lemma A.11 ([50, Appendix, Lemma 6][15])**.** Let $\psi\colon \mathbb{E} \to [-\infty, +\infty]$ be a closed and proper σ-strongly convex function with $\sigma > 0$. If $X \subseteq \operatorname{dom} \psi$ is closed and convex, then $\inf_{x\in X} \psi(x)$ is attained by a unique point.

The above result will allow us to define later the Bregman projector, a function which depends on the existence and uniqueness of the minimizer to be well defined. Even though the above lemma starts to show us how strong convexity can be helpful in optimization problems, one may find its definition cryptic. The following proposition states some interesting properties of strongly convex functions which may help us understand their behavior a little better.

[15]The result in the reference requires continuity of the function, but lower semi-continuity suffices.

Proposition A.12 ([54, Section 12H])**.** Let $\sigma > 0$ and $\psi\colon \mathbb{E} \to [-\infty, +\infty]$ be a closed σ-strongly convex function w.r.t. a norm $\|\cdot\|$. Then, for every $x, y \in \mathbb{E}$, the following properties hold:

(i) for every $u \in \partial\psi(x)$, we have $\psi(y) \geq \psi(x) + \langle u, y - x\rangle + \frac{\sigma}{2}\|x - y\|^2$;
(ii) for every $u \in \partial\psi(x)$ and $v \in \partial\psi(y)$, we have $\langle u - v, x - y\rangle \geq \sigma\|x - y\|^2$;
(iii) ψ^* is finite everywhere and differentiable.

The last property of the above proposition is specially useful for us in the description of the online mirror descent algorithm. At the moment, let us look at item (i), which one may have found familiar. Indeed, it is the same as the subgradient inequality, but with an added quadratic term. If ψ is a σ-strongly convex function, $x \in \mathbb{E}$ is a point where ψ is subdifferentiable, and $u \in \partial\psi(x)$, by the inequality from item (i) there is a (filled) paraboloid, namely the epigraph of the function q defined by $q(z) \coloneqq \psi(x) + \langle u, z - x\rangle + \frac{\sigma}{2}\|x - z\|^2$ for each $z \in \mathbb{E}$, which contains $\operatorname{epi}\psi$. Moreover, equality holds on item (i) for $y = x$, that is, the boundaries of $\operatorname{epi} q$ and $\operatorname{epi}\psi$ intersect (at least) at $(x, \psi(x))$. If ψ is σ-strongly convex, it is in particular *strictly convex*, which intuitively means that there are no line segments in the boundary of $\operatorname{epi}\psi$. It is also worth noting that the higher the value of σ, the higher the contribution of the quadratic term, which implies in paraboloids with "steeper curvature". Usually, higher values of σ in the definition of strong convexity yield better bounds in optimization results.

One question which happens to be quite fruitful to ask is if there are any special properties to the conjugates of strongly convex functions. In order to investigate this relationship, we first define dual norms. If $\|\cdot\|$ is a norm on $\mathbb{E}$, the **dual norm** of $\|\cdot\|$ is the norm $\|\cdot\|_*$ on $\mathbb{E}$ defined by

$$\|x^*\|_* \coloneqq \max\{\langle x, x^*\rangle : x \in \mathbb{E} \text{ s.t. } \|x\| \leq 1\}, \qquad \forall x^* \in \mathbb{E}.$$

As expected, it is easy to see that $\|\cdot\|_{**} = \|\cdot\|$. Now, let $\beta > 0$. A function $\psi\colon \mathbb{E} \to [-\infty, +\infty]$ is **β-strongly smooth** (with respect to a norm $\|\cdot\|$) if ψ is everywhere differentiable and if $\nabla\psi$ is β-Lipschitz continuous with respect to $\|\cdot\|$, that is, for every $x, y \in \mathbb{E}$,

$$\|\nabla\psi(x) - \nabla\psi(y)\|_* \leq \beta\|x - y\|.$$

Again, one may find it difficult to gain much intuition about the impact of smoothness from its definition. Strongly smooth functions have a property dual, in some sense, to the property of item (i) from Proposition A.12 (for a proof, see [49, Lemma 1.2.3]): if $\psi\colon \mathbb{E} \to [-\infty, +\infty]$ is a β-strongly smooth function, then for every $x, y \in \mathbb{E}$ we have

$$\psi(y) \leq \psi(x) + \langle \nabla\psi(x), y - x\rangle + \frac{\beta}{2}\|y - x\|^2.$$

The above inequality tells us that if ψ is a strongly smooth function, then $\operatorname{epi}\psi$ *contains* at each $x \in \mathbb{E}$ a (filled) paraboloid whose boundary intersects the boundary of $\operatorname{epi}\psi$ (at least) at $(x, \psi(x))$. It is interesting to note how this property is, in some sense, dual to the property of strongly convex functions discussed in the last paragraph: if ψ is strongly convex, then for every $x \in \mathbb{E}$ the set $\operatorname{epi}\psi$ is *contained* in paraboloid whose boundary meets the boundary of $\operatorname{epi}\psi$ at $(x, \psi(x))$. Intuitively, smoothness restricts how abruptly a function can change its slope. Not only that, lower values of β in the definition impose stricter restrictions, and usually yield better bounds in optimization results.

The next theorem shows the surprising dual connection between strongly convex and strongly smooth functions.

Theorem A.13 ([41, Theorem 3])**.** Let $\psi\colon \mathbb{E} \to \mathbb{R}$ be closed and convex, let $\|\cdot\|$ be a norm on $\mathbb{E}$, and let $\sigma > 0$. Then ψ is σ-strongly convex w.r.t. $\|\cdot\|$ if and only if ψ^* is $(1/\sigma)$-strongly smooth w.r.t. $\|\cdot\|_*$.

Let us look now at a way of measuring distances through the lens of convex functions. Let $\psi\colon \mathbb{E} \to [-\infty, +\infty]$ be a proper convex function, and let $D \subseteq \operatorname{dom}\psi$ be the set where ψ is differentiable. The

Bregman divergence associated with ψ is the function

$$B_\psi(x, y) \coloneqq \psi(x) - \psi(y) - \langle \nabla\psi(y), x - y\rangle, \qquad \forall x \in \mathbb{E}, y \in D.$$

In words, if $x \in \mathbb{E}$ and $y \in D$, then $B_\psi(x, y)$ measures how far away $\psi(x)$ is from the hyperplane

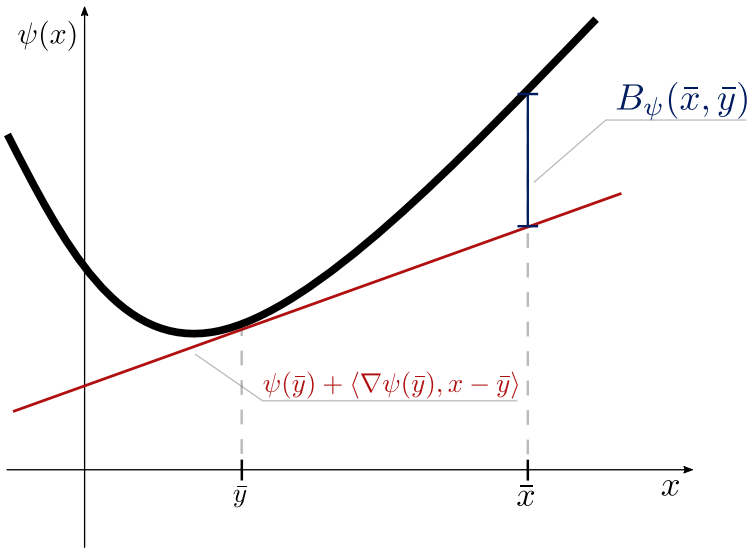

FIGURE 5. Illustration of the Bregman divergence.

tangent to the graph of ψ at y. As one can see, the curvature of ψ drives much of the behavior of B_ψ. Thus, it is not surprising that many applications of Bregman divergences are associated with the use of strongly convex functions.

Lastly, let $X \subseteq \mathbb{E}$ be a closed and convex set. If $\psi\colon \mathbb{E} \to [-\infty, +\infty]$ is a strongly convex[16] function, the **Bregman projector onto** X (with respect to ψ) is the function $\Pi_X^\psi\colon D \to X$ such that

$$\{\Pi_X^\psi(\bar{x})\} \coloneqq \operatorname*{arg\,min}_{x \in X} B_\psi(x, \bar{x}), \qquad \forall \bar{x} \in D.$$

It is worth noting that Lemma A.11 ensures that the Bregman projector is well defined. As an example, if $\psi \coloneqq \frac{1}{2}\|\cdot\|_2^2$ and $X \subseteq \mathbb{E}$, we have $B_\psi(\bar{x}, \bar{y}) = \frac{1}{2}\|\bar{x} - \bar{y}\|^2$ and $\Pi_X^\psi(\bar{x}) = \arg\min_{x \in X}\|x - \bar{x}\|_2$ for every $\bar{x}, \bar{y} \in \mathbb{E}$, that is, the Bregman divergence boils down to the squared euclidean distance and the Bregman projector is the known euclidean projector. Thus, one may think of Bregman divergences and Bregman projectors as a generalization of euclidean distances and projectors[17].

INSTITUTO DE MATEMÁTICA E ESTATÍSTICA, UNIVERSIDADE DE SÃO PAULO, R. DO MATÃO 1010, 05508-090, SÃO PAULO, SP
E-mail address: victorsp@ime.usp.br

[16] Strong convexity is needed here for the existence and uniqueness of the minimum.

[17] Note, however, that the Bregman divergence is not a metric since it need not satisfy the triangle inequality.