Proceedings of Machine Learning Research vol 201:1–19, 2023 34th International Conference on Algorithmic Learning Theory

Projection-free Adaptive Regret with Membership Oracles

Zhou Lu
Princeton University and Google AI Princeton
Nataly Brukhim
Princeton University and Google AI Princeton
Paula Gradu
UC Berkeley
Elad Hazan
Princeton University and Google AI Princeton

Editors: Shipra Agrawal and Francesco Orabona

Abstract

In the framework of online convex optimization, most iterative algorithms require the computation of projections onto convex sets, which can be computationally expensive. To tackle this problem Hazan and Kale (2012) proposed the study of projection-free methods that replace projections with less expensive computations. The most common approach is based on the Frank-Wolfe method, that uses linear optimization computation in lieu of projections. Recent work by Garber and Kretzu (2022) gave sublinear adaptive regret guarantees with projection free algorithms based on the Frank Wolfe approach.

In this work we give projection-free algorithms that are based on a different technique, inspired by Mhammedi (2022), that replaces projections by set-membership computations. We propose a simple lazy gradient-based algorithm with a Minkowski regularization that attains near-optimal adaptive regret bounds. For general convex loss functions we improve previous adaptive regret bounds from $O(T^{3/4})$ to $O(\sqrt{T})$, and further to tight interval dependent bound $\tilde{O}(\sqrt{I})$ where I denotes the interval length. For strongly convex functions we obtain the first poly-logarithmic adaptive regret bounds using a projection-free algorithm.

1. Introduction

We consider the problem of efficient learning in changing environments as formalized in the framework of minimizing adaptive regret in online convex optimization Hazan and Seshadhri (2009). We focus on settings in which efficient computation is paramount, and computing projections is expensive.

This setting of projection-free adaptive regret minimization was recently considered in the work of Garber and Kretzu (2022). The latter paper gives efficient projection-free algorithms that guarantee $O(T^{3/4})$ adaptive regret. Their algorithm is based on the Frank-Wolfe method, that replaces projections by linear optimization computations.

We consider exactly the same problem, but use a different approach for projection free algorithms. Inspired by the recent work of Mhammedi (2022), we consider adaptive regret minimization using set membership computations. Set membership computation amounts to the decision problem of whether an input point x is in the decision set $\mathcal{K}$ or not. For a comprehensive comparison of the complexity of computing a membership oracle and linear optimization see Mhammedi (2022).

For this problem, we propose a simple gradient based algorithm that achieves improved, and tight, adaptive regret bounds for general convex and strongly-convex loss functions, detailed in the following sub-section.

1.1. Summary of results

Our main algorithm is a novel lazy online gradient descent (OGD) algorithm with a specialized regularization function which we call the Minkowski regularization, a variation of the Minkowski functional.

$$\gamma(x) = \inf\{c \geq 1 : \frac{x}{c} \in \mathcal{K}\}.$$

Assuming the decision set $\mathcal{K}$ contains a ball centered at origin, we prove that our algorithm is able to achieve optimal $O(\sqrt{T})$ and $O(\log T)$ regret bounds for general convex and strongly-convex loss functions respectively using only $O(d \log T)$ calls to the membership oracle per round, as a first step to adaptive regret.

Observing that OGD-based algorithms are easier to adapt for minimizing adaptive regret as compared to Frank-Wolfe type algorithms, we show that our algorithm improves previously known best adaptive regret bounds in the projection-free setting when combined with previous adaptive regret meta-algorithms. We derive an $O(\log^2 T)$ adaptive regret bound for strongly-convex loss functions, and an $\tilde{O}(\sqrt{I})$ strongly adaptive regret bound for general convex loss functions where I denotes the length of any sub-interval. Our results in comparison to previous work are summarized in table 1. Besides the improved regret bounds, our algorithm itself is simple.

As our main technical contribution, it's shown that the smoothness of $\mathcal{K}$'s boundary can yield a better approximation algorithm to $\nabla\gamma(x)$ whose error has a good uniform upper bound in contrast to previous general methods, by exploiting its special structure. This property is used to prove that our original algorithm already guarantees $O(\sqrt{T})$ adaptive regret for general convex loss functions when $\mathcal{K}$'s boundary is smooth.

Algorithm	Regret (convex)	Regret (strongly-convex)	Adaptive?	Projection-free?
Hazan and Seshadhri (2009)	$\tilde{O}(\sqrt{T})$	$O(\log^2 T)$	✓	×
Cutkosky (2020), Lu et al. (2022)	$\tilde{O}(\sqrt{\lvert I\rvert})$	×	✓	×
Hazan and Kale (2012)	$O(T^{\frac{3}{4}})$	×	×	✓, LOO
Kretzu and Garber (2021)	×	$O(T^{\frac{2}{3}})$	×	✓, LOO
Mhammedi (2022)	$O(\kappa\sqrt{T})$	$O(\kappa \log T)$	×	✓, MO
Garber and Kretzu (2022)	$O(T^{\frac{3}{4}})$	×	✓	✓, LOO
This paper	$\tilde{O}(\kappa\sqrt{\lvert I\rvert})$	$O(\kappa \log^2 T)$	✓	✓, MO

Table 1: Comparison of results. LOO and MO denote linear optimization oracle and membership oracle respectively. I denotes the length of any interval in adaptive regret. $\kappa = D/r$ denotes the ratio between enclosing and enclosed balls of the domain $\mathcal{K}$.

1.2. Related work

For a survey of the large body of work on online convex optimization, projection free methods and adaptive regret minimization see Hazan (2016).

The study of adaptive regret was initiated by Hazan and Seshadhri (2009), which proves $\tilde{O}(\sqrt{T})$ and $O(\log^2 T)$ adaptive regret bounds for general convex and strongly-convex loss functions respectively. The bound was later improved to $\tilde{O}(\sqrt{I})$ for any sub-interval with length I by Daniely et al. (2015) for general convex losses. Further improvements were made in Jun et al. (2017); Cutkosky (2020); Lu et al. (2022). Lu and Hazan (2022) also considers more efficient algorithms using only $O(\log \log T)$ number of experts, instead of the typical $O(\log T)$ experts.

Projection-free methods date back to the Frank Wolfe algorithm for linear programming Frank and Wolfe (1956). In the setting of online convex optimization, Hazan and Kale (2012) gave a the first sublinear regret algorithm with $O(T^{\frac{3}{4}})$ regret based on the Frank Wolfe algorithm. Later, Garber and Hazan (2015) improved the regret bound to the optimal $O(\sqrt{T})$ under the assumption that the decision set is a polytope. Further assuming the loss is smooth or strongly convex, it was shown that $O(T^{\frac{2}{3}})$ regret is attainable Hazan and Minasyan (2020); Kretzu and Garber (2021); Wan and Zhang (2021).

This paper considers a particularly efficient projection method which can be performed in logarithmic time (and so is essentially projection-free). Similar projection methods have been previously applied in the context of online boosting algorithms Hazan and Singh (2021); Brukhim and Hazan (2021); Brukhim et al. (2020), and projection-free OCO with membership oracles Mhammedi (2022); Levy and Krause (2019).

Recent work by Garber and Kretzu (2022) was the first to consider projection-free adaptive regret minimization. They consider using a linear optimization oracle, and achieve a $O(T^{\frac{3}{4}})$ adaptive regret bound. The work of Garber and Kretzu (2022) also considers using a separation oracle, and gets an $O(\sqrt{T})$ regret bound in this setting. We improve their bound to $\tilde{O}(\sqrt{I})$ in the general sense of projection-free. However, our result is based on membership oracle computations instead of linear optimization, which can be either more or less efficient depending on the underlying convex set (but still strictly more efficient than separation oracles).

Projection free methods using a membership oracle. The approach of using a membership oracle rather than linear optimization started in the work of Levy and Krause (2019). This line of work was significantly extended in the important work of Mhammedi (2022), who introduced the use of the function $\gamma(x)$ as a barrier. The main idea behind the algorithm of Mhammedi (2022) is to first use the gauge function to reduce the problem to optimization over the unit ball by the geometric assumptions on the domain, then apply any black-box OCO algorithm over the unit ball.

Instead, our approach directly uses the gauge function as a regularization function in the classical OGD framework, yielding simpler and more natural algorithm and analysis. In addition, the OGD framework makes it suitable for the adaptive regret minimization framework, and in a later section we will show Algorithm 1 inherently exhibits $O(\sqrt{T})$ adaptive regret when $\partial\mathcal{K}$ is smooth.

2. Preliminaries

We consider the online convex optimization (OCO) problem. At each round t the player $\mathcal{A}$ chooses $x_t \in \mathcal{K}$ where $\mathcal{K} \subset \mathbb{R}^d$ is some convex domain, then the adversary reveals convex loss function

$f_t(x)$ and player suffers loss $f_t(x_t)$. The goal is to minimize regret:

$$\text{Regret}(\mathcal{A}) = \sum_{t=1}^{T} f_t(x_t) - \min_{x \in \mathcal{K}} \sum_{t=1}^{T} f_t(x)$$

which corresponds to the difference between the overall loss the player suffered and that of the best fixed point in hindsight.

2.1. Adaptive regret

It's often the case that in a changing environment we want to have regret guarantees not only globally but also locally. Hazan and Seshadhri (2009) introduced the notion of adaptive regret to capture this intuition to control the worst-case regret among all sub-intervals of $[1, T]$.

$$\text{Adaptive-Regret}(\mathcal{A}) = \max_{I=[s,t]} \left(\sum_{\tau=s}^{t} f_\tau(x_\tau) - \min_{x \in \mathcal{K}} \sum_{\tau=s}^{t} f_\tau(x) \right).$$

Daniely et al. (2015) extended this notion by adding dependence on the length of sub-intervals, and provided an algorithm that achieves an $\tilde{O}(\sqrt{|I|})$ regret bound for all sub-intervals I. In particular, they define strongly adaptive regret as follows:

$$\text{SA-Regret}(\mathcal{A}, k) = \max_{I=[s,t], t-s=k} \left(\sum_{\tau=s}^{t} f_\tau(x_\tau) - \min_{x \in \mathcal{K}} \sum_{\tau=s}^{t} f_\tau(x) \right).$$

The common strategy for minimizing adaptive regret is to hedge over some algorithm instances initiated on different intervals.

2.2. Optimization oracles

Most OCO algorithms require a projection operation per round for updates, which computes the following for some norm $\|\cdot\|$.

$$\Pi_{\mathcal{K}}(x) = \arg\min_{y \in K} \|x - y\|$$

However, computing the projection might be computationally expensive which motivates the study of optimization with projection-free methods, which make use of other (easier) optimization oracles instead.

The most commonly used optimization oracle for projection-free methods is the linear optimization oracle, which outputs

$$\arg\max_{x \in \mathcal{K}} v^\top x$$

given any vector v. It's known that linear optimization oracle is computationally equivalent (up to polynomial time) to separation oracle, which given any x outputs **yes** if $x \in \mathcal{K}$, or a separating hyperplane between x and $\mathcal{K}$.

In this paper we consider the membership oracle, that given an input $x \in \mathbb{R}^d$, outputs **yes** if $x \in \mathcal{K}$ and **no** otherwise. This oracle is considered dual of the linear optimization oracle used by the Frank-Wolfe algorithm, and we shall see it admits better adaptive regret bounds at the cost of only logarithmically more calls to the oracle.

$$\text{Oracle}(x) = \begin{cases} \textbf{yes} & x \in \mathcal{K} \\ \textbf{no} & x \notin \mathcal{K} \end{cases}$$

2.3. Assumptions

We make the following assumption on the loss f_t.

Assumption 1 *The loss f_t is convex, G-Lipschitz and non-negative. The domain $\mathcal{K}$ has diameter D. For simplicity, we assume $G, D \geq 1$.*

We also assume the loss functions are defined over $\mathbb{R}^d$, though we only optimize over $\mathcal{K}$.

Assumption 2 *The loss f_t is defined on $\mathbb{R}^d$ (with the same convexity and Lipschitzness).*

In fact, we can extend any convex Lipschitz function f to $\mathbb{R}^d$ while preserving the Lipschitz constant by defining

$$\hat{f}(x) = \min_{y \in \mathcal{K}} f(y) + G\|x - y\|_2$$

for details see Theorem 1 in Cobzas and Mustata (1978). To make use of the membership oracle, we assume the domain $\mathcal{K}$ contains a ball centered at origin.

Assumption 3 *$\mathcal{K}$ contains $r\mathcal{B}_d$ for some constant $r > 0$, where $\mathcal{B}_d$ is the unit ball in $\mathbb{R}^d$.*

Besides general convex loss functions, we will also consider strongly-convex loss functions, defined as follows:

Definition 1 *A function $f(x)$ is λ-strongly-convex if for any $x, y \in \mathcal{K}$ the following holds:*

$$f(y) \geq f(x) + \nabla f(x)^\top (y - x) + \frac{\lambda}{2}\|x - y\|_2^2$$

3. Projection-free Algorithm via a Membership Oracle

Algorithm 1 Projection-free algorithm via a membership oracle

1: Input: time horizon T, initialization $x_1, y_1 \in \mathcal{K}$, $\delta = \frac{1}{T^2}$.
2: **for** $t = 1, \ldots, T$ **do**
3: Play x_t and suffer loss $f_t(x_t)$
4: Observe loss f_t, define $\hat{f}_t(x) = f_t(x) + 3GD(\gamma(x) - 1)$
5: Compute $\tilde{\gamma}(y_t)$ and $\tilde{\nabla}\gamma(y_t)$, a δ-approximation of $\gamma(y_t)$ and $\nabla\gamma(y_t)$ (see Lemmas 2, 3)
6: Update x_t via the OGD rule and project via the Minkowski regularization:

$$y_{t+1} \leftarrow y_t - \eta_t \tilde{\nabla}\hat{f}_t(y_t) = y_t - \eta_t(\nabla f_t(y_t) + 2GD\tilde{\nabla}\gamma(y_t)).$$

$$x_{t+1} \leftarrow \frac{y_{t+1}}{\tilde{\gamma}(y_{t+1})}.$$

7: **end for**

Let $\mathcal{K} \subseteq \mathbb{R}^d$, and contains $r\mathcal{B}_d$ for some constant $r > 0$ as in Assumption 3. Define the Minkowski regularization for a convex set $\mathcal{K} \subseteq \mathbb{R}^d$ as

$$\gamma(x) = \inf\left\{c \geq 1 : \frac{x}{c} \in \mathcal{K}\right\}. \tag{1}$$

The "projection operation" (which we call the Minkowski projection) w.r.t. $\gamma(x)$ is particularly simple to compute via $O(\log \frac{1}{\delta})$ membership oracle calls up to precision δ, without the use of any projection operator. It is defined as

$$\Pi_\gamma(x) = \frac{x}{\gamma(x)}.$$

Our main algorithm 1 is simply a lazy version of OGD run on the original loss function with the Minkowski regularization, using the Minkowski projection after the gradient update. The algorithm uses $O(\log T)$ calls to the membership oracle to get $\frac{1}{T^2}$ approximate of $\gamma(x)$ and the gradient of $\gamma(x)$. We will introduce some important properties of the Minkowski regularization to establish the validity of using such approximation.

3.1. Properties of the Minkowski regularization

To execute Algorithm 1, one needs to compute $\gamma(x)$ and $\nabla\gamma(x)$ to high accuracy using only the membership oracle. For the analysis henceforth, we require the following properties on the approximation of the regularization function and its gradient.

Lemma 2 *(**Approximating** $\gamma(x)$) Using $\log_2 \frac{2D}{\delta}$ number of calls to the membership oracle, we can compute $\Pi_\gamma(x)$ to δ accuracy. By $\log_2 \frac{2D^2}{r^2\delta}$ number of calls to the membership oracle, we can compute $\gamma(x)$ to δ accuracy.*

Lemma 3 *[Proposition 11 in Mhammedi (2022)] (**Approximating** $\nabla\gamma(x)$) Using $O(d \log \frac{Dd}{r\delta})$ number of calls to the membership oracle, we can get an output denoted as s, such that $\mathbb{E}[s]$ is an δ-approximate of the (sub)gradient of $\gamma(x)$, and $\mathbb{E}[\|s\|_2^2] \leq \frac{2}{r^2}$.*

The methods to achieve the approximation in the above lemmas are intuitively simple: a binary line search is used to approximate $\gamma(x)$, based on which a random partial difference along different coordinates is used to approximate the (sub)gradient. We include the proof of Lemma 2 in appendix, and refer the readers to Mhammedi (2022) or Lee et al. (2018) for the more technical Lemma 3.

As a result, if we choose $\delta = \frac{1}{T^2}$, then by the Lipschitzness of loss functions, the overall difference on the accumulated loss is only $o(1)$ which is negligible compared with regret bounds. The number of calls to the membership oracle is only $O(d \log T)$.

Below we introduce two technical lemmas necessary for our analysis, whose proof can be found in the appendix. We first show the convexity and Lipschitzness of $\gamma(x)$ in the following lemma.

Lemma 4 *The Minkowski regularization $\gamma(x)$ is convex and $\frac{1}{r}$-Lipschitz.*

By the convexity of $\gamma(x)$ and the fact that $\gamma(x)$ is defined on $\mathbb{R}^d$, the well-known Alexandrov theorem implies that it has a second derivative almost everywhere, and without loss of generality we can consider only gradients of $\gamma(x)$ instead of subgradients. We also prove a useful property of $\hat{f}$ and the Minkowski projection.

Lemma 5 *The Minkowski projection only reduces the function value of $\hat{f}$ if $\gamma(x)$ can be computed exactly. When $\Pi_\gamma(x)$ is computed to $\frac{1}{T^2}$ precision, we have that $\hat{f}_t(x_t) \leq \hat{f}_t(y_t) + \frac{Gr+3GD}{rT^2}$*

This lemma is powerful in that it allows us to replace the harder problem of learning x_t over a constrained set $\mathcal{K}$ (which needs projection) to learning y_t over $\mathbb{R}^d$, by replacing the original loss f_t with $\hat{f}_t$. The only thing left to check is whether $\hat{f}_t$ preserves the nice properties of f_t. It follows from definitions that for general convex or strongly-convex Lipschitz f_t, such a modification of the loss is 'for free'. Unfortunately, for exp-concave loss functions $\hat{f}_t$ will lose the exp-concavity.

4. Regret Guarantees and Analysis of Algorithm 1

In this section we prove regret bounds of Algorithm 1 in the non-adaptive setting, in order to build adaptive regret guarantees upon it in the next section. We start with regret guarantee for general convex loss functions.

Theorem 6 *Under Assumptions 1, 2, 3, Algorithm 1 with* $\eta_t = \frac{r}{\sqrt{T}G}$ *can be implemented with* $O(d \log T)$ *calls to the membership oracle per iteration and has regret bounded by*

$$\mathbb{E}[\sum_t f_t(x_t) - \min_{x \in \mathcal{K}} f_t(x)] = O(\frac{D^2 G \sqrt{T}}{r})$$

Proof Denote $\eta_t = \eta$. By Lemma 5 we know that $\hat{f}_t(x_t) \leq \hat{f}_t(y_t) + \frac{Gr+3GD}{rT^2}$. Moreover, by Lemma 4, and assumption 1 that f_t is convex and G-Lipschitz, we get that $\hat{f}_t$ is convex and G-Lipschitz as well. Note that for any $x \in \mathcal{K}$, $\hat{f}_t(x) = f_t(x)$. Thus, we get,

$$\begin{aligned}\mathbb{E}[\sum_t f_t(x_t) - f_t(x^*)] &= \mathbb{E}[\sum_t \hat{f}_t(x_t) - \hat{f}_t(x^*)] \\ &\leq \mathbb{E}[\sum_t \hat{f}_t(y_t) - \hat{f}_t(x^*)] + \frac{Gr + 3GD}{rT}. \qquad \text{(by Lemma 5)}\end{aligned}$$

Next, we bound the above by showing that,

$$\mathbb{E}[\sum_t \hat{f}_t(y_t) - \hat{f}_t(x^*)] = O(D\tilde{G}\sqrt{T}) = O(\frac{D^2 G\sqrt{T}}{r}),$$

where $\tilde{G}$ is the lipschitz constant of $\hat{f}_t$. To show that, we denote $\tilde{\nabla}_t \doteq \nabla \hat{f}_t(y_t)$, $\triangle_t \doteq \tilde{\nabla}\gamma(y_t) - \nabla\gamma(y_t)$ and observe that:

$$\begin{aligned}\|y_{t+1} - x^\star\|_2^2 = \|y_t - x^\star\|_2^2 + \eta^2\|\tilde{\nabla}_t\|_2^2 - 2\eta\tilde{\nabla}_t^\top(y_t - x^\star) \\ + 3GD\eta\triangle_t^\top(y_t - x^* - \eta\tilde{\nabla}_t) + 9G^2D^2\eta^2\|\triangle_t\|_2^2,\end{aligned}$$

and by re-arranging terms and taking expectation we get,

$$\mathbb{E}[2\tilde{\nabla}_t^\top(y_t - x^\star)] \leq \frac{\mathbb{E}[\|y_t - x^\star\|_2^2] - \mathbb{E}[\|y_{t+1} - x^\star\|_2^2]}{\eta} + \eta\tilde{G}^2 + \frac{9GD^2\eta}{T^2} + \frac{36G^2D^2\eta^2}{r^2}.$$

Telescoping and the fact that $y_1 \in \mathcal{K}$ yields the stated bound since by convexity:

$$\begin{aligned}\mathbb{E}[\sum_t \hat{f}_t(y_t) - \hat{f}_t(x^*)] &\leq \mathbb{E}[\sum_t \tilde{\nabla}_t^\top(y_t - x^\star)] \\ &\leq \frac{\|y_1 - x^\star\|_2^2}{2\eta} + \frac{\eta T\tilde{G}^2}{2} + \frac{9GD^2\eta}{2T} + \frac{18G^2D^2\eta^2 T}{r^2}\end{aligned}$$

We find that $\tilde{G}$ is upper bounded by the sum of the gradient of f_t and that of $3GD\gamma(x)$. Recall that γ is $1/r$-Lipschitz (by Lemma 4), plugging in the appropriate $\eta = \frac{r}{\sqrt{T}G}$ gives the stated bound. ■

Similarly, we can get logarithmic regret guarantee for strongly-convex loss functions.

Theorem 7 *Under Assumptions 1, 2, 3 and further assume the loss functions are λ-strongly convex, Algorithm 1 with $\eta_t = \frac{1}{\lambda t}$ can be implemented with $O(d \log T)$ calls to the membership oracle per iteration and has regret bounded by*

$$\mathbb{E}[\sum_t f_t(x_t) - \min_{x \in \mathcal{K}} f_t(x)] = O(\frac{G^2 D^2 \log T}{r^2 \lambda})$$

Proof By Lemma 5 we know that $\hat{f}_t(x_t) \leq \hat{f}_t(y_t) + \frac{Gr+3GD}{rT^2}$. In addition, $\hat{f}_t$ is strongly-convex and Lipschitz by Lemma 4. We know that

$$\mathbb{E}[\sum_t f_t(x_t) - f_t(x^*)] \leq \mathbb{E}[\sum_t \hat{f}_t(y_t) - \hat{f}_t(x^*)] + \frac{Gr + 3GD}{rT}$$

Notice that $\hat{f}_t$ is still λ-strongly-convex. Apply the definition of strong convexity to y_t, x^*, we get

$$\hat{f}_t(y_t) - \hat{f}_t(x^*) \leq \tilde{\nabla}_t^\top (y_t - x^*) - \frac{\lambda}{2}\|y_t - x^*\|_2^2$$

We proceed to upper bound $\tilde{\nabla}_t^\top (y_t - x^*)$. Using the update rule of y_t, we have

$$\begin{aligned}\|y_{t+1} - x^\star\|_2^2 = \|y_t - x^\star\|_2^2 + \eta_t^2 \|\tilde{\nabla}_t\|_2^2 - 2\eta_t \tilde{\nabla}_t^\top (y_t - x^\star) \\ + 3GD\eta_t \triangle_t^\top (y_t - x^* - \eta_t \tilde{\nabla}_t) + 9G^2D^2\eta^2 \|\triangle_t\|_2^2\end{aligned}$$

and so,

$$\mathbb{E}[2\tilde{\nabla}_t^\top (y_t - x^\star)] \leq \frac{\mathbb{E}[\|y_t - x^\star\|_2^2] - \mathbb{E}[\|y_{t+1} - x^\star\|_2^2]}{\eta_t} + \eta_t \tilde{G}^2 + \frac{9GD^2\eta_t}{T^2} + \frac{36G^2D^2\eta_t^2}{r^2}.$$

hence the regret $\mathbb{E}[\sum_t f_t(x_t) - f_t(x^*)]$ can be upper bounded as follows,

$$\begin{aligned}\mathbb{E}[\sum_t f_t(x_t) - f_t(x^*)] &\leq \sum_t \mathbb{E}[\tilde{\nabla}_t^\top (y_t - x^*) - \frac{\lambda}{2}\|y_t - x^*\|_2^2] + \frac{Gr + 3GD}{rT} \\ &\leq \sum_t \frac{1}{2}\mathbb{E}[\|y_t - x^*\|_2^2](\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} - \lambda) + (\frac{\tilde{G}^2}{2} + \frac{9GD^2}{2T^2})\sum_t \eta_t \\ &\quad + \frac{Gr + 3GD}{rT} + \frac{18G^2D^2}{r^2}\sum_t \eta_t^2 \\ &= 0 + (\frac{\tilde{G}^2}{2\lambda} + \frac{9GD^2}{2\lambda T^2})\sum_t \frac{1}{t} + \frac{Gr + 3GD}{rT} + \frac{18G^2D^2}{r^2\lambda^2}\sum_t \frac{1}{t^2} \\ &\leq \frac{G^2D^2}{2r^2\lambda}(1 + \log T) + \frac{3\pi^2 G^2 D^2}{r^2\lambda^2} + o(1) \\ &= O(\frac{G^2D^2 \log T}{r^2\lambda}).\end{aligned}$$

■

5. Projection Free Methods with Adaptive regret Guarantees

In this section we consider projection-free adaptive regret algorithms based on Algorithm 1. We show how to achieve adaptive regret using membership oracles for both general convex and strongly-convex loss functions, by using Algorithm 1 as the base algorithm in adaptive regret meta algorithms (details in appendix).

For strongly-convex loss, we combine Algorithm 1 with the FLH algorithm to get $O(\log^2 T)$ adaptive regret bound, using $O(d\log^2 T)$ calls to the membership oracle. Next, we show that combining Algorithm 1 with the EFLH algorithm Lu and Hazan (2022) yields an $\tilde{O}(I^{\frac{1}{2}+\epsilon})$ strongly adaptive regret bound using only $O(d\log T \times \log\log T/\epsilon)$ calls to the membership oracle per round. We start with the case of strongly convex loss case:

Theorem 8 *Under Assumptions 1, 2, 3, and further assume the loss functions are λ-strongly convex, with $O(d\log^2 T)$ calls to the membership oracle Algorithm 3 achieves expected adaptive regret*

$$O(\frac{G^2 D^2 \log^2 T}{r^2 \lambda})$$

For general convex loss, we show how to get an improved $\tilde{O}(I^{\frac{1}{2}+\epsilon})$ strongly adaptive regret bound when I is the length of any sub-interval $[s,t]$, using only $O(d\log\log T/\epsilon)$ calls to the membership oracle per round for any $\epsilon > 0$. The main idea is to use Algorithm 1 as a black-box, and apply it under the framework of Lu and Hazan (2022).

Theorem 9 *Under Assumptions 1, 2, 3. By using Algorithm 1 as the black-box algorithm in the EFLH algorithm of Lu and Hazan (2022), Algorithm 4 achieves the following expected adaptive regret bound for all intervals $[s,t]$ with $O(d\log T \times \log\log T/\epsilon)$ calls to the membership oracle per round.*

$$\mathbb{E}[\sum_{i=s}^{t} f_i(x_i) - \min_{x\in\mathcal{K}} \sum_{i=s}^{t} f_i(x)] = O(\frac{D^2 G\sqrt{\log T}(t-s)^{\frac{1}{2}+\epsilon}}{r})$$

Notice that by choosing $\epsilon = \frac{1}{\log T}$, we have that

$$I^{\frac{1}{2}+\epsilon} \le \sqrt{I} T^{\frac{1}{\log T}} = O(\sqrt{I})$$

which recovers the optimal $\tilde{O}(\sqrt{I})$ strongly adaptive regret bound, but using $O(\log T)$ number of experts instead. The proofs are direct black-box reductions by replacing the expert OCO algorithms in adaptive regret algorithms by our algorithm, thus we leave the (simple) proofs to appendix.

Remark 10 *Adaptive regret bounds can be derived similarly from Mhammedi (2022), by choosing the subroutine in the reduction of Mhammedi (2022) to have an adaptive regret over the ball.*

6. A Better Gradient Approximation of the Minkowski Regularization

The approximation of $\nabla\gamma(x)$ in Lemma 3 is limited in two aspects: it doesn't have a uniform control of the approximation error like in Lemma 2, for which we can only achieve expected regret guarantees. Secondly, the method used by Lemma 3 is designed for general functions and does not exploit the special structure of $\gamma(x)$.

In this section we build a better method to obtain uniform control of the approximation error for estimating $\nabla\gamma(x)$ when $\partial\mathcal{K}$ is smooth. We later show how even non-smooth sets, such as polytopes, can be smoothed, and the techniques hereby still apply. Denote by $v(x)$ the normal vector to its tangent plane for any $x \in \partial\mathcal{K}$. We start by stating that Algorithm 1 without modification possesses an adaptive regret guarantee for smooth sets.

Theorem 11 *Under Assumptions 1, 2, 3. If $\partial\mathcal{K}$ is smooth and for any $x \in \partial\mathcal{K}$ we have $\frac{v(x)}{x^\top v(x)}$ is Lipschitz, Algorithm 1 with $\eta_t = \frac{D}{\sqrt{T}G}$ can be implemented with $O(d\log T)$ calls to the membership oracle per iteration and has adaptive regret bounded by*

$$\max_{1\le s\le t\le T}\left\{\sum_{i=s}^{t} f_i(x_i) - \min_{x\in\mathcal{K}}\sum_{i=s}^{t} f_i(x)\right\} = O(\frac{D^3G\sqrt{T}}{r^2})$$

The core technique is to show that the smoothness of $\partial\mathcal{K}$ implies the smoothness of $\gamma(x)$, by which we can obtain a uniform upper bound on the approximation error formalized in the following lemma.

Lemma 12 *Assume $\partial\mathcal{K}$ is a smooth manifold, such that for any $x \in \partial\mathcal{K}$, its normal vector $v(x)$ to its tangent plane is unique. For any $x \notin \mathcal{K}$, $\nabla\gamma(x)$ is along the same direction of $v(\Pi_\gamma(x))$. In addition, the gradient $\nabla\gamma(x)$ is*

$$\frac{v(\Pi_\gamma(x))}{\Pi_\gamma(x)^\top v(\Pi_\gamma(x))}$$

The intuition is that the Minkowski functional defines a norm, under which $\partial\mathcal{K}$ becomes the unit sphere, then the gradient of norm on the sphere should be the normal vector to the tangent plane. We are able to build a better estimation to $\nabla\gamma(x)$ based on this expression.

Lemma 13 *Assume the set $\mathcal{K}$ satisfies that for any $x \in \partial\mathcal{K}$ $\frac{v(x)}{x^\top v(x)}$ is β-Lipschitz, then using $O(d\log\frac{Dd}{r\delta})$ number of calls to the membership oracle, we can get an δ-approximate of the gradient of $\gamma(x)$.*

Remark 14 *If $\mathcal{K}$ is a polytope defined by $\{x|h(x)\le 0\}$ where $h(x) = \max_{i=1}^m(x^\top\alpha_i + b_i)$, we can smooth the set to get a similar result. Details can be found in appendix.*

Algorithm 2 Estimating $\nabla\gamma(x)$ with smooth $\partial\mathcal{K}$

1: Input: $\lambda = \frac{1}{\sqrt{d}T^{2.5}}$, $\delta = \frac{1}{dT^5}$ and smoothness factor β of $\partial\mathcal{K}$.
2: Estimate $\gamma(x)$ to δ accuracy.
3: **for** $i = 1, \ldots, d$ **do**
4:     Estimate $\gamma(x+\lambda e_i)$ to δ accuracy.
5:     Estimate $\nabla\gamma(x)_i$ by $\frac{\gamma(x+\lambda e_i)-\gamma(x)}{\lambda}$
6: **end for**

Lemma 13 also enables us to remove the expectation on the regret bound in previous theorems, because estimations to both $\gamma(x)$ and $\nabla\gamma(x)$ can be implemented deterministically, and we now have a uniform control on the estimation error of $\nabla\gamma(x)$.

When the set $\mathcal{K}$ is defined by $\{x|h(x) \leq 0\}$ for some smooth function $h(x)$, if $h(x)$ itself is smooth it also satisfies the above assumption where $\frac{\nabla h(x)}{x^\top \nabla h(x)}$ is $\frac{D\beta}{r}$-Lipschitz on $\partial\mathcal{K}$.

Proof [Proof of Theorem 11] The proof is straightforward by noticing Algorithm 1 is an active algorithm. In fact, using the same analysis as in Theorem 6, the telescoping gives

$$\sum_{i=s}^{t} \hat{f}_i(y_i) - \hat{f}_i(x^*) \ \leq \ \frac{\|y_s - x^\star\|_2^2}{2\eta} + \frac{\eta(t-s)\hat{G}^2}{2} + \frac{9GD^2\eta}{T}$$

The only question here is how to control the norm of y_s. We argue here that for any t

$$\|y_t\|_2 \leq \frac{r}{3}(2 + \frac{3D}{r})^2 + 3D + 2r = O(\frac{D^2}{r})$$

Use the definition of update rule, we get

$$\|y_{t+1}\|_2^2 - \|y_t\|_2^2 = \frac{r^2}{TG^2}\|\tilde{\nabla}\hat{f}_t\|_2^2 - 2y_t^\top \frac{r}{\sqrt{T}G}\tilde{\nabla}\hat{f}_t$$

Notice that when $y_t \notin \mathcal{K}$, $\frac{y_t^\top}{\|y_t\|_2}\nabla\gamma(y_t) = \frac{1}{\|\gamma(y_t)\|_2}$ which is in $[\frac{1}{D}, \frac{1}{r}]$, combine this with the fact that f_t is G-Lipschitz, we have that

$$y_t^\top \tilde{\nabla}\hat{f}_t \geq y_t^\top \nabla\hat{f}_t - \frac{3GD}{rT^2}\|y_t\|_2 \geq \|y_t\|_2(\frac{3GD}{\|\gamma(y_t)\|_2} - G - \frac{3GD}{rT^2}) \geq 1.5G\|y_t\|_2$$

We also need to upper bound $\|\tilde{\nabla}\hat{f}_t\|_2^2$, which is $(G + \frac{3GD}{r} + \frac{3GD}{T^2})^2$.

As a result, when

$$\|y_t\|_2 \geq \frac{r}{3}(2 + \frac{3D}{r})^2$$

we have that

$$2y_t^\top \frac{r}{\sqrt{T}G}\tilde{\nabla}\hat{f}_t \geq \frac{r^2}{\sqrt{T}}(2 + \frac{3D}{r})^2 \geq \frac{r^2}{TG^2}\|\tilde{\nabla}\hat{f}_t\|_2^2$$

and further $\|y_{t+1}\|_2 \leq \|y_t\|_2$.

Besides, the distance that y_t can move in a single step is upper bounded by $(\frac{3GD}{r} + \frac{3GD}{T^2} + G)\frac{r}{\sqrt{T}G} \leq 3D + 2r$, which concludes the proof of the argument. Now, the choice of $\eta = \frac{D}{\sqrt{T}G}$ which is independent of s, t gives an $O(\frac{D^3G\sqrt{T}}{r^2})$ regret bound over all intervals $[s, t]$ simultaneously because $t - s \leq T$. ■

Simply using the previous estimation of $\nabla\gamma(x)$ (Lemma 3) doesn't work, because its variance causes an unpleasant martingale representation of y_t, making it hard to bound its norm.

7. Conclusion

In this paper, we consider the problem of online convex optimization with membership oracles. We propose a simple lazy OGD algorithm that achieves $O(\sqrt{T})$ and $O(\log T)$ regret bounds for general convex and strongly-convex loss functions respectively, using $O(d \log T)$ number of calls to the membership oracle. We further utilize the active nature of our algorithm in adaptive regret minimization, achieving better bounds than known algorithms using linear optimization oracles. It remains open to try to match these regret bounds using linear optimization, or alternatively fewer membership oracle queries.

Acknowledgements

The authors thank Dan Garber and Zak Mhammedi for helpful suggestions and typo corrections in an earlier version of this manuscript. Zhou Lu, Nataly Brukhim and Elad Hazan are supported by NSF grant number 2134040.

References

Nataly Brukhim and Elad Hazan. Online boosting with bandit feedback. In *Algorithmic Learning Theory*, pages 397–420. PMLR, 2021.

Nataly Brukhim, Xinyi Chen, Elad Hazan, and Shay Moran. Online agnostic boosting via regret minimization. *Advances in Neural Information Processing Systems*, 33:644–654, 2020.

S Cobzas and C Mustata. Norm-preserving extension of convex lipschitz functions. *J. Approx. Theory*, 24(3):236–244, 1978.

Ashok Cutkosky. Parameter-free, dynamic, and strongly-adaptive online learning. In *International Conference on Machine Learning*, pages 2250–2259. PMLR, 2020.

Amit Daniely, Alon Gonen, and Shai Shalev-Shwartz. Strongly adaptive online learning. In *International Conference on Machine Learning*, pages 1405–1411. PMLR, 2015.

Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

Dan Garber and Elad Hazan. Faster rates for the frank-wolfe method over strongly-convex sets. In *International Conference on Machine Learning*, pages 541–549. PMLR, 2015.

Dan Garber and Ben Kretzu. New projection-free algorithms for online convex optimization with adaptive regret guarantees. *arXiv preprint arXiv:2202.04721*, 2022.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Elad Hazan and Satyen Kale. Projection-free online learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1843–1850, 2012.

Elad Hazan and Edgar Minasyan. Faster projection-free online learning. In *Conference on Learning Theory*, pages 1877–1893. PMLR, 2020.

Elad Hazan and Comandur Seshadhri. Efficient learning algorithms for changing environments. In *Proceedings of the 26th annual international conference on machine learning*, pages 393–400, 2009.

Elad Hazan and Karan Singh. Boosting for online convex optimization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4140–4149. PMLR, 18–24 Jul 2021.

Kwang-Sung Jun, Francesco Orabona, Stephen Wright, and Rebecca Willett. Improved strongly adaptive online learning using coin betting. In *Artificial Intelligence and Statistics*, pages 943–951. PMLR, 2017.

Ben Kretzu and Dan Garber. Revisiting projection-free online learning: the strongly convex case. In *International Conference on Artificial Intelligence and Statistics*, pages 3592–3600. PMLR, 2021.

Yin Tat Lee, Aaron Sidford, and Santosh S Vempala. Efficient convex optimization with membership oracles. In *Conference On Learning Theory*, pages 1292–1294. PMLR, 2018.

Kfir Levy and Andreas Krause. Projection free online learning over smooth sets. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1458–1466. PMLR, 2019.

Zhou Lu and Elad Hazan. Efficient adaptive regret minimization. *arXiv preprint arXiv:2207.00646*, 2022.

Zhou Lu, Wenhan Xia, Sanjeev Arora, and Elad Hazan. Adaptive gradient methods with local guarantees. *arXiv preprint arXiv:2203.01400*, 2022.

Zakaria Mhammedi. Efficient projection-free online convex optimization with membership oracle. In *Conference on Learning Theory*, pages 5314–5390. PMLR, 2022.

Yuanyu Wan and Lijun Zhang. Projection-free online learning over strongly convex sets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10076–10084, 2021.

Algorithm 3 Adaptive regret for strongly-convex loss without projection

Input: OCO algorithm $\mathcal{A}$, active expert set S_t, horizon T, $\alpha = \frac{\lambda}{G^2}$ and constant $\epsilon > 0$.
Set $\mathcal{A}_j$ to be Algorithm 1 with $\eta_t = \frac{1}{\lambda t}$.
Pruning rule: the horizon of $\mathcal{A}_j$ is $2^{k+2} + 1$ if $j = r2^k$ where r is an odd number.
Initialize: $S_1 = \{1\}$, $p_1^1 = 1$.
for $t = 1, \ldots, T$ **do**
  Set $\forall j \in S_t$, x_t^j to be the prediction of $\mathcal{A}_j$.
  Play $x_t = \sum_{j \in S_t} p_t^j x_t^j$.
  Perform multiplicative weight update. For $j \in S_t$

$$\tilde{p}_{t+1}^j = \frac{p_t^j e^{-\alpha f_t(x_t^j)}}{\sum_{i \in S_t} p_t^i e^{-\alpha f_t(x_t^i)}}$$

  Prune S_t and add $\{t+1\}$ to get S_{t+1}. Initialize $\tilde{p}_{t+1}^{t+1} = \frac{1}{t}$, then

$$\forall j \in S_{t+1}, p_{t+1}^j = \frac{\tilde{p}_{t+1}^j}{\sum_{i \in S_{t+1}} \tilde{p}_{t+1}^i}$$

end for

Appendix A. Adaptive Regret Meta Algorithms

We include here the algorithm boxes of the FLH and EFLH algorithms. Both Algorithm 3 and Algorithm 4 are meta expert algorithms, with some black-box base algorithms run on specific intervals.

Appendix B. Omitted Proofs

B.1. Proof of Lemma 4

Proof For any two points $x_1, x_2 \in \mathbb{R}^d$, we let $\gamma_1 = \gamma(x_1), \gamma_2 = \gamma(x_2)$. For any $\lambda \in (0, 1)$, let $z = \lambda x_1 + (1 - \lambda)x_2$. To establish convexity, we would like to prove that

$$\gamma(z) \leq \lambda\gamma_1 + (1 - \lambda)\gamma_2. \tag{2}$$

Since $\gamma(z)$ is the minimal value (greater than 1) such that $z/\gamma(z) \in \mathcal{K}$, we get that proving that Equation (2) holds is equivalent to showing that,

$$\frac{z}{\gamma(z)} = \frac{\lambda x_1 + (1 - \lambda)x_2}{\lambda\gamma_1 + (1 - \lambda)\gamma_2} \in \mathcal{K}. \tag{3}$$

First, notice that the term $z/\gamma(z)$ can be rewritten as follows,

$$\frac{z}{\gamma(z)} = \frac{x_1}{\gamma_1} \cdot \left(\frac{1}{1 + (\frac{1}{\lambda} - 1)\frac{\gamma_2}{\gamma_1}} \right) + \frac{x_2}{\gamma_2} \cdot \left(\frac{1}{1 + \frac{\lambda}{1-\lambda}\frac{\gamma_1}{\gamma_2}} \right). \tag{4}$$

Then, it is easy to verify that for any $\lambda \in (0, 1)$, and any $\gamma_1, \gamma_2 \geq 1$, both terms in the brackets are non-negative and sum to 1. Moreover, since by definition we have that $\frac{x_1}{\gamma_1}, \frac{x_2}{\gamma_2} \in \mathcal{K}$, we get that $\frac{z}{\gamma(z)}$

Algorithm 4 Strongly adaptive regret for general convex loss without projection

Input: OCO algorithm $\mathcal{A}$, active expert set S_t, horizon T and constant $\epsilon > 0$.

Let $\mathcal{A}_{t,k}$ be an instance of $\mathcal{A}$ initialized at t with lifespan $4l_k = 4\lfloor 2^{(1+\epsilon)^k}/2 \rfloor + 4$, for $2^{(1+\epsilon)^k}/2 \leq T$. Here $\mathcal{A}_{t,k}$ is set to be Algorithm 1 with horizon $4l_k$ and $\eta = \frac{r}{2\sqrt{l_k}G}$.

Initialize: $S_1 = \{(1,1),(1,2),...\}$, $w_1^{(1,k)} = \min\left\{\frac{1}{2}, \sqrt{\frac{\log T}{l_k}}\right\}$.

for $t = 1, \ldots, T$ **do**

  Let $W_t = \sum_{(j,k)\in S_t} w_t^{(j,k)}$.

  Play $x_t = \sum_{(j,k)\in S_t} \frac{w_t^{(j,k)}}{W_t} x_t^{(j,k)}$, where $x_t^{(j,k)}$ is the prediction of $\mathcal{A}_{(j,k)}$.

  Perform multiplicative weight update to get w_{t+1}. For $(j,k) \in S_t$

$$w_{t+1}^{(j,k)} = w_t^{(j,k)} \left(1 + \min\left\{\frac{1}{2}, \sqrt{\frac{\log T}{l_k}}\right\} (f_t(x_t) - f_t(x_t^{(j,k)}))\right)$$

  Update S_t according to the pruning rule. Initialize

$$w_{t+1}^{(t+1,k)} = \min\left\{\frac{1}{2}, \sqrt{\frac{\log T}{l_k}}\right\}$$

  if $(t+1,k)$ is added to S_{t+1} (when $l_k | t-1$).

end for

is a convex combination of elements in $\mathcal{K}$, and thus must be in $\mathcal{K}$ as well. This concludes the proof that $\gamma(x)$ is convex.

Next, to prove Lipschitzness, notice that when $x_1, x_2 \in \mathcal{K}$ the argument is trivially true. Otherwise, assume without loss of generality that $x_2 \notin \mathcal{K}$ (and any x_1). Decompose x_2 as,

$$x_2 = x_1 + r\frac{x_2 - x_1}{\|x_2 - x_1\|_2} \times \frac{\|x_2 - x_1\|_2}{r}. \tag{5}$$

Then, we define $\alpha = \frac{r}{\|x_2 - x_1\|_2 + r\gamma_1}$, where r is the radius of a ball $r\mathcal{B}_d$ that is contained in $\mathcal{K}$ (recall assumption 1). Using out decomposition of x_2 above, we get that,

$$\alpha \cdot x_2 = (\gamma_1\alpha) \cdot \frac{x_1}{\gamma_1} + (1 - \gamma_1\alpha) \cdot r\frac{x_2 - x_1}{\|x_2 - x_1\|_2}. \tag{6}$$

Observe that since $r\frac{x_2 - x_1}{\|x_2 - x_1\|_2} \in r\mathcal{B}_d$ then is it also contained in $\mathcal{K}$, by assumption 1. In addition, by definition we have $\frac{x_1}{\gamma_1} \in \mathcal{K}$. Thus, the term $\alpha \cdot x_2$ can be re-written as a convex combination of elements in $\mathcal{K}$, and is therefore contained in $\mathcal{K}$ as well. Since γ_2 is the minimal scalar such that division of x_2 by it is in $\mathcal{K}$, we get that $\gamma_2 \leq 1/\alpha$. That is,

$$|\gamma_2 - \gamma_1| \leq \frac{1}{r}\|x_2 - x_1\|_2, \tag{7}$$

which concludes the proof. ■

B.2. Proof of Lemma 5

Proof In the exact case, denote $\gamma_t := \gamma(y_{t-1})$. Then, we have that

$$\begin{aligned}
\hat{f}_t(x_t) - \hat{f}_t(y_t) &= f_t(x_t) - f_t(y_t) + 3GD(\gamma(x_t) - 1) - 3GD(\gamma_t - 1) && \text{(definition of } \hat{f}_t) \\
&= f_t(x_t) - f_t(y_t) - 3GD(\gamma_t - 1) && (\gamma(x_t) = 1) \\
&\le G\|y_t - x_t\|_2 - 3GD(\gamma_t - 1) && (f_t \text{ is } G\text{-Lipschitz}) \\
&= G(\gamma_t - 1)\|x_t\|_2 - 3GD(\gamma_t - 1) && (y_t = \gamma_t x_t) \\
&= G(\gamma_t - 1)\big(\|x_t\|_2 - 3D\big) \\
&\le 0 && (\|x_t\|_2 \le 3D, \gamma_t \ge 1)
\end{aligned}$$

We notice that a similar argument also holds for the approximate version of γ_t: just replace $x_t = \frac{y_t}{\gamma_t}$ by $\tilde{x}_t = \frac{y_t}{\tilde{\gamma}_t}$ and use the fact that $\|x_t - \tilde{x}_t\|_2 \le \frac{1}{T^2}$, which gives $\hat{f}_t(\tilde{x}_t) - \hat{f}_t(y_t) \le \frac{Gr+3GD}{rT^2}$. This supports our previous argument that the overall difference on the accumulated loss is only $O(1)$ and is negligible. ■

B.3. Proof of Lemma 2

Proof Without loss of generality we only consider the case $x \notin \mathcal{K}$. We use the simple binary search method: the first call to the oracle gives $x \notin \mathcal{K}$, and we save two points x_{in} and x_{out}, initialized as 0 and x. For the $t+1$-th call to the oracle, we query $\frac{1}{2}(x_{in} + x_{out})$ and set it as x_{in} or x_{out} according to whether it's in $\mathcal{K}$ or not.

It follows that after the t-th call to the oracle, $\|x_{in} - x_{out}\|_2 \le \frac{2D}{2^t}$ which implies that x_{in} is an $\frac{2D}{2^t}$-approximation to $\Pi_\gamma(x)$. In other words, one can use $\log_2(\frac{2D}{\epsilon})$ calls to the membership oracle to get an ϵ-approximation to $\frac{x}{\gamma(x)}$. To put it more general, one can use $\log_2(\frac{2D^2}{r^2\epsilon})$ calls to the membership oracle to get an ϵ-approximation to $\gamma(x)$. ■

B.4. Proofs of Theorem 8 and Theorem 9

Proof Two theorems from previous works are introduced first, in which we will use our Algorithm 1 as a black-box to obtain the desired regret bounds.

Theorem 15 (Theorem 3.1 in Hazan and Seshadhri (2009)) *If all loss functions are α-exp concave then the FLH algorithm attains adaptive regret of $O(\frac{\log^2 T}{\alpha})$. The number of experts per iteration is $O(n^3 \log T)$.*

Theorem 16 (Theorem 6 in Lu and Hazan (2022)) *Given an OCO algorithm $\mathcal{A}$ with regret bound $cGD\sqrt{T}$ for some constant $c \ge 1$, the adaptive regret of EFLH is bounded by $40cGD\sqrt{\log T}|I|^{\frac{1+\epsilon}{2}}$ for any interval $I \subset [1, T]$. The number of experts per round is $O(\log\log T/\epsilon)$.*

We first define the exp-concavity of loss function.

Definition 17 *A function $f(x)$ is α-exp-concave if $e^{-\alpha f(x)}$ is convex.*

A notable fact is that under Assumption 1, any λ-strongly-convex function is also $\frac{\lambda}{G^2}$-exp-concave Hazan (2016).

For Theorem 8, we use the fact that the loss functions are also $\frac{\lambda}{G^2}$-exp-concave. By Theorem 3.1 in Hazan and Seshadhri (2009) and Theorem 7, if we treat our Algorithm 1 as the experts in FLH, the total regret is bounded by $O(\frac{G^2 D^2 \log^2 T}{r^2 \lambda} + \frac{G^2 \log^2 T}{\lambda})$. Noticing $r = O(D)$, we reach our conclusion.

Theorem 9 is a direct consequence of Theorem 6 and Theorem 6 from Lu and Hazan (2022) by treating our Algorithm 1 as the experts in EFLH. Notice that the EFLH is a multiplicative weight algorithm over experts (instances of Algorithm 1) and doesn't require extra projections. ■

B.5. Proof of Lemma 12

Proof To see this, we compare the values of $\gamma(x)$ and $\gamma(x + \epsilon v)$ given a small constant ϵ and a unit vector v, and see which direction increases the function value the most. We have that

$$\Pi_\gamma(x + \epsilon v) = \Pi_\gamma(x) + \epsilon \frac{\|\Pi_\gamma(x)\|_2}{\|x\|_2}(v - v^\top v(\Pi_\gamma(x))v(\Pi_\gamma(x))) + O(\epsilon^2)$$

due to the fact that $\partial\mathcal{K}$ is smooth and at a neighborhood of x, its tangent plane is a good linear approximation of the boundary. As a result

$$\gamma(x + \epsilon v) = \frac{\|x + \epsilon v\|_2}{\|\Pi_\gamma(x) + \epsilon \frac{\|\Pi_\gamma(x)\|_2}{\|x\|_2}(v - v^\top v(\Pi_\gamma(x))v(\Pi_\gamma(x))) + O(\epsilon^2)\|_2}$$

We are actually dealing with the 2-dimensional plane spanned by v and $v(\Pi_\gamma(x))$, therefore without loss of generality we can do a coordinate change to only consider this 2-dimensional plane and use $x = (\|x\|_2, 0), \Pi_\gamma(x) = (\|\Pi_\gamma(x)\|_2, 0)$, and parameterize v and $v(\Pi_\gamma(x))$ by θ_1, θ_2. After standard calculation and simplification we have that

$$\gamma(x + \epsilon v)^2 = \frac{\|x\|_2^2 + 2\epsilon\|x\|_2 \cos\theta_1}{\|\Pi_\gamma(x)\|_2^2 + 2\epsilon \frac{\|\Pi_\gamma(x)\|_2^2}{\|x\|_2} \sin\theta_2(\sin\theta_2\cos\theta_1 - \cos\theta_2\sin\theta_1)}$$

Maximizing it is equivalent to maximizing

$$\frac{\|x\|_2 + 2\epsilon\cos\theta_1}{\|x\|_2 + 2\epsilon\sin\theta_2(\sin\theta_2\cos\theta_1 - \cos\theta_2\sin\theta_1)}$$

which is further equivalent to maximizing

$$\sin\theta_2\sin\theta_1 + \cos\theta_2\cos\theta_1$$

by ignoring second-order terms. It is maximized when taking $\theta_1 = \theta_2$, or in other words, $\nabla\gamma(x)$ is along the same direction of $v(\Pi_\gamma(x))$. An easier way to think of it is by slightly modifying the definition of $\gamma(x)$ to be the original Minkowski functional

$$\inf\{\gamma > 0 : \frac{x}{\gamma} \in \mathcal{K}\}$$

then $\gamma(x)$ defines a norm and $\partial\mathcal{K}$ is exactly the solution set of $\gamma(x) = 1$. Then the gradient of $\gamma(x)$ is the normal vector to this norm-1 level set.

We haven't determine the magnitude of the gradients yet. From Lemma 6 in Mhammedi (2022), we know that for any $x \notin \mathcal{K}$

$$\nabla\gamma(x) = \arg\max_{s\in\hat{\mathcal{K}}} s^\top x$$

where $\hat{\mathcal{K}}$ is the polar set of $\mathcal{K}$ defined as

$$\hat{\mathcal{K}} = \{s \in \mathbb{R}^d | s^\top x \leq 1, \forall x \in \mathcal{K}\}$$

From this we know the magnitude has to be $\frac{1}{\Pi_\gamma(x)^\top v(\Pi_\gamma(x))}$, and the gradient $\nabla\gamma(x)$ is really just

$$\frac{v(\Pi_\gamma(x))}{\Pi_\gamma(x)^\top v(\Pi_\gamma(x))}$$

■

B.6. Proof of Lemma 13

Proof From Lemma 2 we already know one can use $\log_2(\frac{2D^3}{r^4\epsilon})$ calls to the membership oracle to get an ϵ-approximation to $\gamma(x)$. We now use it to estimate $\nabla\gamma(x)$: for any $i \in [d]$ and a small constant λ, we have the following inequality for some x'.

$$|\gamma(x + \lambda e_i) - \gamma(x) - \lambda\nabla\gamma(x)_i| \leq \frac{\lambda^2}{2}\|\nabla^2\gamma(x')\|_2^2$$

From the smoothness assumption we know that $\|\nabla^2\gamma(x')\|_2^2 \leq \beta^2$. Then the estimation error of $\nabla\gamma(x)_i$ can be upper bounded by $\frac{\lambda\beta^2}{2} + \frac{2\delta}{\lambda}$ where δ is the precision of $\gamma(x)$. Take $\delta = \frac{1}{dT^5}$ and $\lambda = \frac{1}{\sqrt{d}T^{2.5}}$, the estimation error to $\nabla\gamma(x)$ is bounded by $\frac{1}{\sqrt{d}T^2}$ while we only use $O(d\log T)$ calls to the membership oracle by Lemma 2, that's $O(\log T)$ calls for each coordinate i. Since the estimation error for each coordinate is now bounded by $\frac{1}{\sqrt{d}T^2}$, the overall error is bounded by

$$\sqrt{d} \times \frac{1}{\sqrt{d}T^2} = \frac{1}{T^2}$$

■

Appendix C. Set Smoothing for Polytopes

Let $\mathcal{K} \subset \mathbb{R}^d$ be defined as a level set of a function h,

$$\mathcal{K} = \{x \in \mathbb{R}^d | h(x) \leq 0\}.$$

where $h(x) = \max_{i=1}^m(x^\top \alpha_i + b_i)$ and each α_i is a unit vector. Consider the smoothed set

$$\mathcal{K}_a = \{x \in \mathbb{R}^d | h_a(x) \leq 0\}$$

where $h_a(x)$ is defined as

$$h_a(x) = \frac{1}{a}\log(\sum_{i=1}^{m} e^{a(x^\top \alpha_i + b_i)})$$

We additionally assume that we have access to the linear constraints of $h(x)$, so that the oracle to $\mathcal{K}$ is stronger than just membership, but still weaker than projection.

We will show several properties of this smoothing method in order to determine the value of a. First, the new function $h_a(x)$ is $O(ma)$-smooth. Second, we notice that $\mathcal{K}_a$ is a subset of $\mathcal{K}$, while it also contains

$$\{x \in \mathbb{R}^d | h(x) \ \leq \ -\frac{\log m}{a}\}$$

because

$$h_a(x) \leq \frac{1}{a}\log(me^{ah(x)}) \leq h(x) + \frac{\log m}{a}.$$

In addition, for any x such that $h(x) = 0$, there exists x' such that $h(x) \ \leq \ -\frac{\log m}{a}$ and $\|x - x'\|_2 \leq \frac{D \log m}{ra}$, simply by shrinking x along the same direction and using the geometric assumptions on $\mathcal{K}$.

Now we choose $a = T^3$. Because the loss function is Lipschitz, replacing $\mathcal{K}$ by $\mathcal{K}_a$ can only hurt the regret by $O(T\tilde{G}\frac{D\log m}{rT^3}) = O(\frac{1}{T^2})$ which is negligible, using a similar argument as in the proof of Theorem 6.

Meanwhile, the smoothness constant being now $O(T^3)$ isn't an issue because we can multiply an additional $\frac{1}{T^3}$ factor to the method in Lemma 13 to get a similar guarantee, i.e. choosing $\delta = \frac{1}{dT^{11}}$ and $\lambda = \frac{1}{\sqrt{d}T^{5.5}}$ using still $O(d \log T)$ membership oracle calls.

Finally, because $h_a(x)$ has a clear closed form and can be exactly computed by m computations of the linear constraints in $h(x)$, we can implement an exact membership oracle to $\mathcal{K}_a$ using m times more calls. In all, we can achieve the same guarantee as in Lemma 13 using $O(md \log T)$ number of calls to the membership oracle instead.