Projection-free Online Exp-concave Optimization

Dan Garber   DANGAR@TECHNION.AC.IL
Ben Kretzu   BENKRETZU@CAMPUS.TECHNION.AC.IL
Technion - Israel Institute of Technology

Abstract

We consider the setting of online convex optimization (OCO) with *exp-concave* losses. The best regret bound known for this setting is $O(n \log T)$, where n is the dimension and T is the number of prediction rounds (treating all other quantities as constants and assuming T is sufficiently large), and is attainable via the well-known Online Newton Step algorithm (ONS). However, ONS requires on each iteration to compute a projection (according to some matrix-induced norm) onto the feasible convex set, which is often computationally prohibitive in high-dimensional settings and when the feasible set admits a non-trivial structure. In this work we consider projection-free online algorithms for exp-concave and smooth losses, where by projection-free we refer to algorithms that rely only on the availability of a linear optimization oracle (LOO) for the feasible set, which in many applications of interest admits much more efficient implementations than a projection oracle. We present an LOO-based ONS-style algorithm, which using overall $O(T)$ calls to a LOO, guarantees in worst case regret bounded by $\widetilde{O}(n^{2/3}T^{2/3})$ (ignoring all quantities except for n, T). However, our algorithm is most interesting in an important and plausible low-dimensional data scenario: if the gradients (approximately) span a subspace of dimension at most ρ, $\rho << n$, the regret bound improves to $\widetilde{O}(\rho^{2/3}T^{2/3})$, and by applying standard deterministic sketching techniques, both the space and average additional per-iteration runtime requirements are only $O(\rho n)$ (instead of $O(n^2)$). This improves upon recently proposed LOO-based algorithms for OCO which, while having the same state-of-the-art dependence on the horizon T, suffer from regret/oracle complexity that scales with $\sqrt{n}$ or worse.

Keywords: online learning, online convex optimization, projection-free, exp-concave, linear optimization oracle

1. Introduction

This work contributes to the line of research on efficient projection-free algorithms for online convex optimization (OCO), which has received a significant amount of interest in the theoretical machine learning community in recent years, see for instance Hazan and Kale (2012); Garber and Hazan (2013); Chen et al. (2019); Garber and Kretzu (2020); Kretzu and Garber (2021); Hazan and Minasyan (2020); Levy and Krause (2019); Wan and Zhang (2021); Ene et al. (2021); Chen et al. (2018); Zhang et al. (2017); Garber and Kretzu (2022); Mhammedi (2021, 2022); Lu et al. (2022). We recall that in the setting of OCO (Hazan, 2019; Shalev-Shwartz et al., 2012) (see formal definition in Section 2.2) a decision maker is required to iteratively choose an action — a point in some convex and closed set $\mathcal{K} \subseteq \mathbb{R}^n$ (fixed throughout all iterations)[1], where after her selection, a convex loss function from $\mathcal{K}$ to $\mathbb{R}$ is revealed and the decision maker incurs a loss which equals the value of the loss function evaluated at the point chosen on that round. The performance of the decision maker

---

1. for ease of presentation we consider the underlying vector space to be $\mathbb{R}^n$, however any finite Euclidean space will work

is measured via the standard notion of regret which is the difference between her accumulated loss throughout all T rounds (where T here is assumed to be known in advanced) and that of the best fixed point in $\mathcal{K}$ in hindsight. Throughout this work we consider the full-information setting, where after each round, the decision maker gains full knowledge of the loss function used on that round. The term *projection-free* refers to algorithms which avoid the computation of orthogonal projections onto the feasible set $\mathcal{K}$, as required by most standard algorithms, and instead only access the feasible set via conceptually simpler computational primitives, such as an oracle for linear optimization over $\mathcal{K}$ (LOO). The motivation for such methods is that indeed for many feasible sets of interest and for high-dimensional problems, implementing the LOO can be much more efficient than projection, see for instance detailed examples in Jaggi (2013); Hazan and Kale (2012) (see in the sequel discussion on other projection-free oracles).

In this work we consider, to the best of our knowledge for the first time, efficient projection-free LOO-based algorithms for OCO in case all loss functions are *exp-concave*. We recall that a function $f(\mathbf{x})$ is α-exp-concave for some $\alpha > 0$, if the function $e^{-\alpha f(\mathbf{x})}$ is concave[2]. Exp-concavity is a property which is well known to allow for faster convergence rates (in terms of regret). In particular, exp-concave losses underly some of the most important applications of OCO such as online linear regression and online portfolio selection. More generally, any loss of the form $f(\mathbf{x}) = g(\mathbf{a}^\top \mathbf{x})$ with $g : \mathbb{R} \to \mathbb{R}$ strongly convex, is exp-concave. While for general convex functions the optimal regret bound attainable (by any algorithm) is $O(\sqrt{T})$ (treating all quantities except for n, T as constants), in case all losses are exp-concave, a regret bound of the form $O(n \log T)$ is attainable (Hazan et al., 2007; Hazan, 2019), which is faster for any fixed dimension n and T large enough. The latter regret bound is attainable via a well-known algorithm known as *Online Newton Step* (ONS), however, ONS requires on each iteration to compute a non-Euclidean projection onto the feasible set w.r.t. to some matrix-induced norm (this matrix aggregates all the gradients of the losses observed so far), and hence is often computationally prohibitive in high-dimensional settings and when the feasible set $\mathcal{K}$ admits non-trivial structure.

Our main contribution is a novel projection-free LOO-based variant of ONS for exp-concave and smooth (Lipschitz continuous gradient) losses. Using overall $O(T)$ calls to a LOO (throughout all rounds), our algorithm guarantees in worst case $\widetilde{O}(n^{2/3} T^{2/3})$ regret (where currently for ease of presentation we treat all quantitates except for n, T as constants, and $\widetilde{O}$ hides poly-logarithmic factors). However, our algorithm is most interesting in a highly popular and plausible scenario in high-dimensional analytics, namely when the observed gradients of the loss functions (the data fed into the algorithm), approximately, span only a low dimensional subspace. Denoting by ρ the (approximate) dimension of the subspace spanned by the gradients, by a simple tuning of parameters, our regret bound improves to $\widetilde{O}(\rho^{2/3} T^{2/3})$, which is independent of the ambient dimension n. Moreover, by leveraging well-known efficient deterministic sketching techniques, as was already proposed in Luo et al. (2016) (but not in the context of projection-free algorithms), we can also reduce the memory and additional average runtime per iteration from $O(n^2)$ to only $O(\rho n)$, i.e., linear in the dimension for a constant ρ.

To put our results in perspective, the best previous regret bound for a LOO-based algorithm for OCO (that holds for arbitrary convex losses and with no assumption on the span of the gradients) which is dimension-independent is $O(T^{3/4})$ and requires overall $O(T)$ calls to the LOO (Hazan and Kale, 2012; Garber and Kretzu, 2022). Two recently proposed LOO-based algorithms also improved

---

2. in linear regression one has $g(x) = (x - b)^2$ for some $b \in \mathbb{R}$, and in online portfolio selection one has $g(x) = -\log(x)$, which is strongly convex on $\mathbb{R}_{>0}$

the dependence on the horizon T from $T^{3/4}$ to $T^{2/3}$, however suffer from regret and/or oracle complexity which scales with $\sqrt{n}$ or worse: the regret bound of the Follow The Perturbed Leader-based algorithm of Hazan and Minasyan (2020) has a regret bound of the form $O(\sqrt{n}T^{2/3})$, while the Follow The Leader-based algorithm of Mhammedi (2022) (which is based on approximating the feasible set with a strongly convex set, which leads to the faster rate) requires overall $\widetilde{O}(nT)$ calls to the LOO and has a regret bound of the form $O((R/r)^{2/3}T^{2/3})$, where R/r is the ratio between an enclosing ball and an enclosed ball, which often scales with $\sqrt{n}$ and even with n (e.g., for the simplex or the spectrahedron, see Mhammedi (2021)). Moreover, the additional runtime per iteration of the algorithm in Mhammedi (2022) scales with n^3. Unfortunately, such explicit dependencies on the ambient dimension may be prohibitive for high-dimensional problems, which is indeed the typical setting of interest for projection-free methods. Thus, it is interesting whether it is possible to obtain a fast $T^{2/3}$ rate without explicit dependence on the ambient dimension.

A very popular approach to circumvent explicit dependencies on the ambient dimension, which underlies numerous models in statistics/high-dimensional analytics and is observed frequently in real-world scenarios, is the assumption that the data, at least approximately, lies only in a low-dimensional subspace. In our context of OCO with exp-concave losses, as discussed above, many losses of interest take the form $f_t(\mathbf{x}) = g_t(\mathbf{a}_t^\top \mathbf{x})$, $g_t : \mathbb{R} \to \mathbb{R}$, with the gradient vector being $\nabla f_t(\mathbf{x}) = g'_t(\mathbf{a}_t^\top \mathbf{x}_t)\mathbf{a}_t$. Thus, when the data vectors $\mathbf{a}_1, \ldots, \mathbf{a}_T$ approximately span only a low-dimensional subspace (in the sense that the eigenvalues of the unnormalized covariance $\lambda_i(\sum_{t=1}^T \mathbf{a}_t\mathbf{a}_t^\top)$ are sufficiently small for all $i \geq \rho+1$, for some $\rho << n$), our regret bound becomes dimension-independent and thus suitable for such popular high-dimensional settings. To the best of our knowledge, the fast (in terms of T, but not n) algorithms proposed in Hazan and Minasyan (2020); Mhammedi (2022) cannot efficiently leverage low-dimensionality of the gradients.

Table 1 gives a short summery of our results as well as a comparison to related LOO-based algorithms for OCO.

On the technical side, our work primarily builds on the recent approach of Garber and Kretzu (2022) which suggested a LOO-based projection-free variant of the well known Euclidean Online Projected Gradient Descent method (Zinkevich, 2003; Hazan, 2019), and is based on the concept of *approximately-feasible (Euclidean) projections*[3], which refers to the computation of points which on one-hand, while infeasible w.r.t. the decision set $\mathcal{K}$, still satisfy certain properties related to orthogonal projections and are sufficiently close to the feasible set, which drives the regret bound, and on the other-hand, could be computed efficiently using only a limited number of queries to the LOO of the feasible set via the classical Frank-Wolfe algorithm for *offline* convex minimization (Frank and Wolfe, 1956; Jaggi, 2013). Here we provide a non-trivial extension of this framework, from supporting only Euclidean (approximately feasible) projections, to supporting projections w.r.t. matrix-induced norms as employed by ONS. We also substantially improve the bound on the oracle complexity required to compute such approximately-feasible projections, which is crucial to obtaining our faster regret rate ($T^{2/3}$ instead of $T^{3/4}$ in Garber and Kretzu (2022)).

Other projection-free oracles: We mention in passing that while, as in this work, most literature on projection-free OCO assumes the feasible set is accessible through a LOO, some recent works have also considered other oracles such as a separation oracle or a membership oracle (Mhammedi, 2021; Garber and Kretzu, 2022; Lu et al., 2022). While each of these oracles could be implemented via the others (see for instance Tat Lee et al. (2017)), none of them is generically superior to the

---

3. Garber and Kretzu (2022) originally used the terminology *close infeasible projections*

Reference	Additional assumptions	Based on	Deter- ministic?	LOO calls	Additional runtime	Regret
Hazan and Kale (2012)	-	RFTL	✓	T	nT	$RGT^{\frac{3}{4}}$
Garber and Kretzu (2022)	-	OGD	✓	T	nT	$RGT^{\frac{3}{4}}$
Mhammedi (2022)	$r\mathcal{B} \subseteq \mathcal{K}$	FTL	✗	nT	n^3T	$GR(R/r)^{\frac{2}{3}}T^{\frac{2}{3}}$
Hazan and Mi- nasyan (2020)	β-smooth losses	FTPL	✗	T	nT	$R\left(G\sqrt{n} + \beta R\right)T^{\frac{2}{3}}$
Theorem 10	α-exp concave and β-smooth losses	ONS	✓	T	n^2T	$\left(G + \alpha^{-1}\right)Rn^{\frac{2}{3}}T^{\frac{2}{3}}$ $+\beta R^2 T^{\frac{2}{3}}$
Theorem 11	1. α-exp concave and β-smooth losses 2. gradients approx. span ρ-dim. subspace (*)	ONS	✓	T	ρnT	$\left(G + \alpha^{-1}\right)R\rho^{\frac{2}{3}}T^{\frac{2}{3}}$ $+\beta R^2 T^{\frac{2}{3}}$

Table 1: Summary of results and comparison to previous LOO-based methods (applicable to arbitrary convex and compact sets). G denotes an upper-bound on the ℓ_2 norm of the gradients, R denotes the radius of the feasible set $\mathcal{K}$, and $\mathcal{B}$ denotes the unit Euclidean ball centered at the origin. Condition (*) should be understood as $\sum_{i=\rho+1}^{n} \lambda_i\left(\sum_{t=1}^{T} \nabla_t \nabla_t^\top\right) = O(T^{2/3})$, where $\lambda_i(\cdot)$ denotes the i-th largest eigenvalue and $\nabla_t \in \mathbb{R}^n$ denotes the gradient of the loss observed on round t. The bounds omit constants and poly-logarithmic factors.

other (in terms of efficiency of implementation). Finally, the very recent work Mhammedi and Gatmiry (2022) considers an efficient variant of ONS which is based on accessing the feasible set only through a separation oracle, however it requires the feasible set $\mathcal{K}$ to by symmetric in the sense that $\mathcal{K} = -\mathcal{K}$, which is fairly restrictive.

2. Preliminaries

2.1. Notation

We let $\|\cdot\|$ denote the Euclidean norm over $\mathbb{R}^n$. For a positive semidefinite matrix $\mathbf{A}$, we let $\|\cdot\|_{\mathbf{A}}$ denote the induced semi-norm over $\mathbb{R}^n$, i.e., for any $\mathbf{x} \in \mathbb{R}^n$, $\|\mathbf{x}\|_{\mathbf{A}} = \sqrt{\mathbf{x}^\top \mathbf{A} \mathbf{x}}$. We let $\mathbb{S}^n, \mathbb{S}^n_+, \mathbb{S}^n_{++}$ denote the space of real symmetric $n \times n$ matrices, the set of all real $n \times n$ (symmetric) positive semidefinite matrices, and the set of all real $n \times n$ (symmetric) positive definite matrices, respectively. We use the standard notation $\mathbf{A} \succeq 0$ ($\mathbf{A} \succ 0$) to denote that $\mathbf{A} \in \mathbb{S}^n_+$ ($\mathbf{A} \in \mathbb{S}^n_{++}$). For a matrix $\mathbf{A} \in \mathbb{S}^n$ and $i \in [n]$, we let $\lambda_i(\mathbf{A})$ denote the i-th largest (signed) eigenvalue of $\mathbf{A}$. We denote by $\mathbf{A} \bullet \mathbf{B}$ the standard inner product between two matrices in $\mathbb{S}^n$, i.e., $\mathbf{A} \bullet \mathbf{B} = \sum_{i=1}^n \sum_{j=1}^n \mathbf{A}_{i,j} \mathbf{B}_{i,j} = \text{Tr}(\mathbf{A} \mathbf{B}^\top)$. We let $\mathcal{B}$ denote the unit Euclidean ball in $\mathbb{R}^n$ centered at the origin. Given a convex and compact set $\mathcal{C} \subset \mathbb{R}^n$, a point $\mathbf{x} \in \mathbb{R}^n$, and a positive definite matrix $\mathbf{A} \in \mathbb{S}^n_{++}$, we let $\text{dist}(\mathbf{x}, \mathcal{C})$ and $\text{dist}_{\mathbf{A}}(\mathbf{x}, \mathcal{C})$ denote the Euclidean distance of $\mathbf{x}$ from

$\mathcal{C}$ and the distance induced by $\mathbf{A}$ of $\mathbf{x}$ from $\mathcal{C}$, respectively. That is, $\text{dist}(\mathbf{x}, \mathcal{C}) = \min_{\mathbf{y} \in \mathcal{C}} \|\mathbf{x} - \mathbf{y}\|$, $\text{dist}(\mathbf{x}, \mathcal{C}) = \min_{\mathbf{y} \in \mathcal{C}} \|\mathbf{x} - \mathbf{y}\|_{\mathbf{A}}$.

2.2. Problem setup: online exp-concave and smooth optimization with a LOO

We recall the setting of OCO (Hazan, 2019; Shalev-Shwartz et al., 2012), in which, a decision maker is required for T rounds (T is assumed known in advanced for simplicity), to select on each round some point $\mathbf{x}^t \in \mathcal{K}$, where $\mathcal{K} \subset \mathbb{R}^n$ is convex and closed (and fixed throughout all rounds). After making her choice on round t, the decision maker observes a convex loss function $f_t : \mathcal{K} \to \mathbb{R}$ and incurs the loss $f_t(\mathbf{x}^t)$. The goal of the decision maker is to minimize her regret which is given by

$$\mathcal{R}_T = \sum_{t=1}^{T} f_t(\mathbf{x}^t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}),$$

i.e., it is the difference between her cumulative loss, and the cumulative loss of the best fixed point in $\mathcal{K}$ in hindsight.

Throughout this work we assume the feasible set is compact and accessible through a linear optimization oracle, which means that for any $\mathbf{g} \in \mathbb{R}^n$ we can efficiently compute some $\mathbf{v}^* \in \text{argmin}_{\mathbf{v} \in \mathcal{K}} \mathbf{v}^\top \mathbf{g}$.

We now turn to discuss our specific assumptions on the loss functions $f_1, \ldots, f_T$. In the following definitions we let $\mathcal{C}$ denote a convex and compact subset of $\mathbb{R}^n$.

Definition 1 *We say $f : \mathcal{C} \to \mathbb{R}$ is β-smooth over $\mathcal{C}$, for some $\beta \geq 0$, if for every $\mathbf{x}, \mathbf{y} \in \mathcal{C}$ it holds that $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{x})\| \leq \beta \|\mathbf{x} - \mathbf{y}\|$.*

Definition 2 *We say $f : \mathcal{C} \to \mathbb{R}$ is α-exp concave over $\mathcal{C}$, for some $\alpha > 0$, if $e^{-\alpha f(\mathbf{x})}$ is concave over $\mathcal{C}$.*

We recall that an exp-concave function is in particular convex (see Hazan (2019)). In fact, we shall consider a weaker condition than exp concavity, which we shall refer to as a curvature condition.

Definition 3 *Let R denote the radius of $\mathcal{C}$, i.e., $\max_{\mathbf{x}, \mathbf{y} \in \mathcal{C}} \|\mathbf{x} - \mathbf{y}\| \leq 2R$. A differentiable function $f : \mathcal{C} \to R$ with gradients upper-bounded in ℓ_2 norm by some $G > 0$ over $\mathcal{C}$, is said to satisfy the curvature condition over $\mathcal{C}$ with some parameter $\alpha > 0$, if for every $\eta \geq \max\{4GR, 2/\alpha\}$ and every $\mathbf{x}, \mathbf{y} \in \mathcal{C}$, it holds that $f(\mathbf{x}) - f(\mathbf{y}) \leq \nabla f(\mathbf{x})^\top (\mathbf{x} - \mathbf{y}) - \frac{1}{2\eta} (\mathbf{x} - \mathbf{y})^\top \nabla f(\mathbf{x}) \nabla f(\mathbf{x})^\top (\mathbf{x} - \mathbf{y})$.*

This condition is weaker than exp-concavity in the sense that an α-exp-convex function also satisfies the curvature condition with the same parameter α (Hazan, 2019).

The following assumption records all of our assumptions on the loss functions $f_1, \ldots, f_T$, which we assume to hold throughout the rest of the paper.

Assumption 1 *The loss functions $f_1, \ldots, f_T$, are all β-smooth, have gradients upper-bounded in ℓ_2 norm by some $G > 0$, and satisfy the curvature condition with some parameter $\alpha > 0$, over the set $3R\mathcal{B}$, where R denotes the radius of a ball enclosing $\mathcal{K}$ and centered at the origin.[4]*

---

4. The consideration of a set strictly containing $\mathcal{K}$ (the ball $3R\mathcal{B}$) in which these assumptions hold is required since our algorithm will query gradients of the loss functions at infeasible points. For ease of presentation we consider the enclosing set $3R\mathcal{B}$, however this could be very much relaxed to consider a set only slightly larger than $\mathcal{K}$ in which the assumption needs to hold, see discussion in Section D.

2.3. Online Newton step with approximately-feasible (matrix) projections

We now begin to discuss our high-level approach towards efficient LOO-based implementation of the Online Newton Step method. As discussed, our approach builds on the one in Garber and Kretzu (2022), which considered the Euclidean Online Gradient Descent method, and extends it to ONS which requires non-Euclidean projections according to matrix-induced norms.

One of our central algorithmic building blocks is an oracle for computing *approximately-feasible projections* onto the feasible set $\mathcal{K}$ w.r.t. to some matrix-induced norm, which we now define. In the sequel we show how such an oracle could be implemented efficiently using only a LOO for the feasible set $\mathcal{K}$.

Definition 4 *Given a convex and compact set $\mathcal{K} \subset \mathbb{R}^n$, a positive definite matrix $\mathbf{A} \in \mathbb{S}_{++}^n$, and a tolerance $\epsilon > 0$, we say a function $\mathcal{O}_{AFP}(\mathbf{y}, \mathbf{A}, \epsilon, \mathcal{K})$ is an approximately-feasible projection (AFP) oracle (for the set $\mathcal{K}$ with parameters $\mathbf{A}, \epsilon$), if for any input point $\mathbf{y} \in \mathbb{R}^n$, it returns some $(\mathbf{x}, \widetilde{\mathbf{y}}) \in \mathcal{K} \times \mathbb{R}^n$ such that i. for all $\mathbf{z} \in \mathcal{K}$, $\|\widetilde{\mathbf{y}} - \mathbf{z}\|_\mathbf{A} \leq \|\mathbf{y} - \mathbf{z}\|_\mathbf{A}$, and ii. $\|\mathbf{x} - \widetilde{\mathbf{y}}\|_\mathbf{A}^2 \leq \epsilon$.*

Equipped with the concept of an AFP oracle, we can now introduce our second central algorithmic building block — a template for ONS-style algorithms that only accesses the feasible set $\mathcal{K}$ through an AFP oracle. As opposed to the standard (projection-based) ONS which maintains a single sequence of feasible points, Algorithm 1 maintains two main sequences: one sequence ($\{\widetilde{\mathbf{y}}_m\}_{m\geq 1}$) which is infeasible and corresponds to an ONS-style update, and another sequence ($\{\mathbf{x}_m\}_{m\geq 1}$) which is feasible and point-wise close to the previous sequence. We refer to Algorithm 1 as a template since it does not explicitly state how to choose the matrices $\mathbf{A}_m, m = 1, 2, \ldots$, used in the algorithm, but only states some restrictions on them. This will be useful later on to derive our two variants: one in which $\mathbf{A}_m$ is based on exact aggregation of gradients (as in standard ONS), and the other which is only a certain approximation via a matrix sketching technique and useful for reducing memory and runtime requirements in case the gradients span (approximately) only a low-dimensional subspace. Finally, Algorithm 1 partitions the prediction rounds $1, \ldots, T$ into consecutive disjoint blocks of size K (denoted by a subscript of m). This will be important to make sure the AFP oracle is called only once every K iterations, which will allow to upper bound the number of LOO calls required to implement it according to our needs. The following lemma states the regret bound of Algorithm 1 that will be used to derive all following regret bounds.

Lemma 5 *Consider running Algorithm 1 with some block size $K \in [T]$[5] and with $\epsilon_I \geq G^2 K^2$, $\eta \geq \max\{12KGR, \frac{2K}{\alpha}\}$. Suppose further that for all m it holds that $\widetilde{\mathbf{y}}_m \in 3R\mathcal{B}$. Then, it holds that*

$$\forall \mathbf{x} \in \mathcal{K}: \ \sum_{t=1}^{T} f_t(\mathbf{x}^t) - f_t(\mathbf{x}) \leq \frac{3\beta\epsilon}{\epsilon_I}T + \sqrt{\frac{6\epsilon T}{K}\sum_{m=1}^{T/K}\|\bar{\nabla}_m\|_{\mathbf{A}_m^{-1}}^2 + \frac{2R^2\epsilon_I}{\eta}} + \frac{\eta}{2}\sum_{m=1}^{T/K}\|\nabla_m\|_{\mathbf{A}_m^{-1}}^2 .$$

The proof, at a high level, builds on coupling the standard ONS proof (Hazan, 2019) with the properties of the AFP oracle, to derive a regret bound on the infeasible sequence $\{\widetilde{\mathbf{y}}_m\}_{\mathbf{m}\geq 1}$. The smoothness assumption on the losses is then used (and only in this proof) to derive a regret bound on the feasible sequence $\{\mathbf{x}^t\}_{t\geq 1}$, without incurring terms which (eventually) will scale worse than $T^{2/3}$.

---

5. without loosing much generality, throughout this paper we assume that the chosen block size K is integer and divides T, which will ease the analysis. Waiving this convention will only add lower-order terms to our regret bounds

---

Algorithm 1: Template for Online Newton Step Without Feasibility

---

Data: horizon T, block length K, learning rate $\eta > 0$, initialization parameter $\epsilon_I > 0$, error tolerance $\epsilon > 0$, approximately-infeasible projection oracle $\mathcal{O}_{AFP}(\cdot, \cdot, \cdot, \mathcal{K})$

$\mathbf{x}_1 = \widetilde{\mathbf{y}}_1 \leftarrow$ arbitrary point in $\mathcal{K}$

$\mathbf{A}_0 = \epsilon_I \mathbf{I}_n$

for $m = 1, \ldots, T/K$ **do**

    Set $\bar{\nabla}_m = \mathbf{0}$

    for $s = 1, \ldots, K$ **do**

        Play $\mathbf{x}^t = \mathbf{x}_m$ for $t = (m-1)K + s$

        Set $\nabla_t = \nabla f_t(\widetilde{\mathbf{y}}_m)$ and update $\bar{\nabla}_m = \bar{\nabla}_m + \nabla_t$

    end

    Update $\mathbf{A}_m$ such that $\mathbf{A}_0 \preceq \mathbf{A}_m \preceq \mathbf{A}_{m-1} + \bar{\nabla}_m \bar{\nabla}_m^\top$

    Update $\mathbf{y}_{m+1} = \widetilde{\mathbf{y}}_m - \eta \mathbf{A}_m^{-1} \bar{\nabla}_m$

    Set $(\mathbf{x}_{m+1}, \widetilde{\mathbf{y}}_{m+1}) \leftarrow \mathcal{O}_{AFP}(\mathbf{y}_{m+1}, \mathbf{A}_m, 3\epsilon, \mathcal{K})$

end

Proof [Proof of Lemma 5] Fix some block $m \in \{1, \ldots, T/K\}$. Since $\bar{\nabla}_m = \sum_{t=(m-1)K+1}^{mK} \nabla_t$, it holds that $\bar{\nabla}_m \bar{\nabla}_m^\top \preceq K^2 G^2 \mathbf{I}_n$. Thus, for the value of ϵ_I stated in the lemma and the choice of $\mathbf{A}_{m-1}$, we have that $\bar{\nabla}_m \bar{\nabla}_m^\top \preceq \epsilon_I \mathbf{I}_n = \mathbf{A}_0 \preceq \mathbf{A}_{m-1}$. Additionally, since $\mathbf{A}_m \preceq \mathbf{A}_{m-1} + \bar{\nabla}_m \bar{\nabla}_m^\top$, we have that

$$\|\mathbf{x}_m - \widetilde{\mathbf{y}}_m\|_{\mathbf{A}_m}^2 \leq \|\mathbf{x}_m - \widetilde{\mathbf{y}}_m\|_{\mathbf{A}_{m-1}}^2 + \|\mathbf{x}_m - \widetilde{\mathbf{y}}_m\|_{\bar{\nabla}_m \bar{\nabla}_m^\top}^2 \leq 2\|\mathbf{x}_m - \widetilde{\mathbf{y}}_m\|_{\mathbf{A}_{m-1}}^2. \qquad (1)$$

Denote $\mathbf{x}^* \in \operatorname{argmin}_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x})$, $m(t) := \lceil \frac{t}{K} \rceil$, and $\mathbf{g}_t = \nabla f_t(\mathbf{x}_{m(t)})$ for all $t \in [T]$. Using the convexity of each $f_t(\cdot)$, it holds that

$$\sum_{t=1}^{T} f_t(\mathbf{x}_{m(t)}) - f_t(\mathbf{x}^*) \leq \sum_{t=1}^{T} \mathbf{g}_t^\top \left(\mathbf{x}_{m(t)} - \widetilde{\mathbf{y}}_{m(t)}\right) + \sum_{t=1}^{T} f_t(\widetilde{\mathbf{y}}_{m(t)}) - f_t(\mathbf{x}^*).$$

Since each $f_t(\cdot)$ is β-smooth we have that,

$$\sum_{t=1}^{T} f_t(\mathbf{x}_{m(t)}) - f_t(\mathbf{x}^*) \leq \sum_{t=1}^{T} (\mathbf{g}_t - \nabla_t)^\top \left(\mathbf{x}_{m(t)} - \widetilde{\mathbf{y}}_{m(t)}\right) + \nabla_t^\top \left(\mathbf{x}_{m(t)} - \widetilde{\mathbf{y}}_{m(t)}\right) + f_t(\widetilde{\mathbf{y}}_{m(t)}) - f_t(\mathbf{x}^*)$$

$$\leq \sum_{t=1}^{T} \beta \left\| \mathbf{x}_{m(t)} - \widetilde{\mathbf{y}}_{m(t)} \right\|^2 + \sum_{m=1}^{T/K} \bar{\nabla}_m \left(\mathbf{x}_m - \widetilde{\mathbf{y}}_m\right) + \sum_{t=1}^{T} f_t(\widetilde{\mathbf{y}}_{m(t)}) - f_t(\mathbf{x}^*).$$

$$(2)$$

Since for every block m, $(\mathbf{x}_m, \widetilde{\mathbf{y}}_m)$ are the output of $\mathcal{O}_{AFP}(\mathbf{y}_m, \mathbf{A}_{m-1}, 3\epsilon, \mathcal{K})$, we have that

$$\sum_{t=1}^{T} \beta \left\| \mathbf{x}_{m(t)} - \widetilde{\mathbf{y}}_{m(t)} \right\|^2 = K \sum_{m=1}^{T/K} \beta \left\| \mathbf{x}_m - \widetilde{\mathbf{y}}_m \right\|^2 \underset{(a)}{\leq} \frac{\beta K}{\epsilon_I} \sum_{m=1}^{T/K} \|\mathbf{x}_m - \widetilde{\mathbf{y}}_m\|_{\mathbf{A}_{m-1}}^2 \underset{(b)}{\leq} \frac{3\beta\epsilon}{\epsilon_I} T, \quad (3)$$

where (a) holds since, by the choice of $\mathbf{A}_{m-1}$, it holds that $\mathbf{A}_{m-1} \succeq \mathbf{A}_0 = \epsilon_I \mathbf{I}_n$, and so, for any $\mathbf{z} \in \mathbb{R}^n$ we have that $\|\mathbf{z}\|^2_{\mathbf{A}_{m-1}} \geq \epsilon_I \|\mathbf{z}\|^2$, and (b) holds due to the gurantee of the AFP oracle.

Using Eq. (1) and the facts that for every block m, $\|\cdot\|_{\mathbf{A}_m}, \|\cdot\|_{\mathbf{A}_m^{-1}}$ are dual norms, and $(\mathbf{x}_m, \widetilde{\mathbf{y}}_m)$ are the outputs of $\mathcal{O}_{AFP}$, we have that

$$\sum_{m=1}^{T/K} \bar{\nabla}_m^\top (\mathbf{x}_m - \widetilde{\mathbf{y}}_m) \leq \sum_{m=1}^{T/K} \|\bar{\nabla}_m\|_{\mathbf{A}_m^{-1}} \|\mathbf{x}_m - \widetilde{\mathbf{y}}_m\|_{\mathbf{A}_m}$$
$$\leq \sum_{m=1}^{T/K} \|\bar{\nabla}_m\|_{\mathbf{A}_m^{-1}} \sqrt{2}\|\mathbf{x}_m - \widetilde{\mathbf{y}}_m\|_{\mathbf{A}_{m-1}} \leq \sqrt{6\epsilon} \sum_{m=1}^{T/K} \|\bar{\nabla}_m\|_{\mathbf{A}_m^{-1}},$$

where the last inequality is again due to the guarantee of the AFP oracle.

Using Jensen's inequality, we have that

$$\sum_{m=1}^{T/K} \bar{\nabla}_m^\top (\mathbf{x}_m - \widetilde{\mathbf{y}}_m) \leq \sqrt{6\epsilon} \sqrt{\left(\sum_{m=1}^{T/K} \|\bar{\nabla}_m\|_{\mathbf{A}_m^{-1}} \right)^2} \leq \sqrt{6\epsilon} \sqrt{\frac{T}{K}} \sqrt{\sum_{m=1}^{T/K} \|\bar{\nabla}_m\|^2_{\mathbf{A}_m^{-1}}}. \tag{4}$$

Now we turn to upper bound the third term in Eq. (2). For every block m, using the fact that $\widetilde{\mathbf{y}}_{m+1}$ is the output of $\mathcal{O}_{AFP}$ w.r.t. the input point $\mathbf{y}_{m+1}$, we have that

$$\forall \mathbf{x} \in \mathcal{K}: \quad \|\widetilde{\mathbf{y}}_{m+1} - \mathbf{x}\|^2_{\mathbf{A}_m} \leq \|\mathbf{y}_{m+1} - \mathbf{x}\|^2_{\mathbf{A}_m} = \|\widetilde{\mathbf{y}}_m - \eta \mathbf{A}_m^{-1} \bar{\nabla}_m - \mathbf{x}\|^2_{\mathbf{A}_m}$$
$$= \|\widetilde{\mathbf{y}}_m - \mathbf{x}\|^2_{\mathbf{A}_m} + \eta^2 \bar{\nabla}_m^\top \mathbf{A}_m^{-1} \bar{\nabla}_m - 2\eta \bar{\nabla}_m^\top (\widetilde{\mathbf{y}}_m - \mathbf{x}).$$

Rearranging, for every $m \in [T/K]$, we have that

$$\bar{\nabla}_m^\top (\widetilde{\mathbf{y}}_m - \mathbf{x}) \leq \frac{\|\widetilde{\mathbf{y}}_m - \mathbf{x}\|^2_{\mathbf{A}_m}}{2\eta} - \frac{\|\widetilde{\mathbf{y}}_{m+1} - \mathbf{x}\|^2_{\mathbf{A}_m}}{2\eta} + \frac{\eta}{2} \|\bar{\nabla}_m\|^2_{\mathbf{A}_m^{-1}}. \tag{5}$$

Before we continue, we upper bound the summation of $\|\widetilde{\mathbf{y}}_m - \mathbf{x}\|^2_{\mathbf{A}_m} - \|\widetilde{\mathbf{y}}_{m+1} - \mathbf{x}\|^2_{\mathbf{A}_m}$ over $m \in [T/K]$. Since $\widetilde{\mathbf{y}}_1 \in \mathcal{K}$ and $\mathbf{A}_0 = \epsilon_I \mathbf{I}_n$, we have that

$$\sum_{m=1}^{T/K} \|\widetilde{\mathbf{y}}_m - \mathbf{x}\|^2_{\mathbf{A}_m} - \|\widetilde{\mathbf{y}}_{m+1} - \mathbf{x}\|^2_{\mathbf{A}_m} \leq \|\widetilde{\mathbf{y}}_1 - \mathbf{x}\|^2_{\mathbf{A}_1} + \sum_{m=2}^{T/K} (\widetilde{\mathbf{y}}_m - \mathbf{x})^\top (\mathbf{A}_m - \mathbf{A}_{m-1}) (\widetilde{\mathbf{y}}_m - \mathbf{x})$$
$$= \|\widetilde{\mathbf{y}}_1 - \mathbf{x}\|^2_{\mathbf{A}_0} + \sum_{m=1}^{T/K} (\widetilde{\mathbf{y}}_m - \mathbf{x})^\top (\mathbf{A}_m - \mathbf{A}_{m-1}) (\widetilde{\mathbf{y}}_m - \mathbf{x})$$
$$\leq 4R^2 \epsilon_I + \sum_{m=1}^{T/K} (\widetilde{\mathbf{y}}_m - \mathbf{x})^\top (\mathbf{A}_m - \mathbf{A}_{m-1}) (\widetilde{\mathbf{y}}_m - \mathbf{x}).$$

Summing Eq. (5) over $m \in [T/K]$, and using the fact that for all blocks m, $\mathbf{A}_m - \mathbf{A}_{m-1} \preceq \bar{\nabla}_m \bar{\nabla}_m^\top$, we have that

$$\sum_{m=1}^{T/K} \bar{\nabla}_m^\top (\widetilde{\mathbf{y}}_m - \mathbf{x}) \leq \frac{2R^2 \epsilon_I}{\eta} + \sum_{m=1}^{T/K} \frac{(\widetilde{\mathbf{y}}_m - \mathbf{x})^\top \bar{\nabla}_m \bar{\nabla}_m^\top (\widetilde{\mathbf{y}}_m - \mathbf{x})}{2\eta} + \frac{\eta}{2} \sum_{m=1}^{T/K} \|\bar{\nabla}_m\|^2_{\mathbf{A}_m^{-1}}.$$

Since $\eta \geq \max\{12KGR, \frac{2K}{\alpha}\}$, and $\widetilde{\mathbf{y}}_m \in 3R\mathcal{B}$ for every $m \in [T/K]$ (by the assumption of the lemma), using Lemma 19 (sum of functions which satisfy the curvature condition in Definition 3,

8

also satisfies this condition) we have that for every $\mathbf{x} \in \mathcal{K} \subseteq 3R\mathcal{B}$, and every $m \in [T/K]$, it holds that

$$\sum_{t=(m-1)K+1}^{mK} f_t(\widetilde{\mathbf{y}}_m) - f_t(\mathbf{x}) \leq \bar{\nabla}_m^\top (\widetilde{\mathbf{y}}_m - \mathbf{x}) - \frac{1}{2\eta} (\widetilde{\mathbf{y}}_m - \mathbf{x})^\top \bar{\nabla}_m \bar{\nabla}_m^\top (\widetilde{\mathbf{y}}_m - \mathbf{x}).$$

Combining the last two inequalities we obtain that for every $\mathbf{x} \in \mathcal{K}$ it holds that,

$$\sum_{t=1}^{T} f_t(\widetilde{\mathbf{y}}_{m(t)}) - f_t(\mathbf{x}) \leq \frac{2R^2 \epsilon_I}{\eta} + \frac{\eta}{2} \sum_{m=1}^{T/K} \|\nabla_m\|_{\mathbf{A}_m^{-1}}^2.$$

Plugging-in the last equation, Eq. (4), and Eq. (3) into Eq. (2), we obtain

$$\sum_{t=1}^{T} f_t(\mathbf{x}_{m(t)}) - f_t(\mathbf{x}^*) \leq \frac{3\beta\epsilon}{\epsilon_I} T + \sqrt{6\epsilon} \sqrt{\frac{T}{K}} \sqrt{\sum_{m=1}^{T/K} \|\bar{\nabla}_m\|_{\mathbf{A}_m^{-1}}^2} + \frac{2R^2 \epsilon_I}{\eta} + \frac{\eta}{2} \sum_{m=1}^{T/K} \|\nabla_m\|_{\mathbf{A}_m^{-1}}^2.$$

Finally, the lemma follows from recalling that with the notation of Algorithm 1, we have that for all $t \in [T]$, $\mathbf{x}^t = \mathbf{x}_{m(t)}$. ∎

3. Efficient LOO-based Approximately-Feasible Projections

In this section we turn to discuss the technical heart of the paper — the efficient construction of an AFP oracle for the feasible set $\mathcal{K}$ (Definition 4) using only a linear optimization oracle for $\mathcal{K}$. As already discussed, we build on the approach of Garber and Kretzu (2022) for Euclidean projection, but expand on it in two ways: i. we extend it to projection w.r.t. matrix-induced norms, as employed by ONS, and ii. we critically improve certain parts of the analysis, which while not being a bottleneck in the analysis of Garber and Kretzu (2022) (which has a $T^{3/4}$ regret bound), are indeed crucial for our faster $T^{2/3}$ regret bounds.

At a high level, the construction relies on the following idea: given an infeasible point $\mathbf{y}$, using only the LOO, we can either construct a generalized hyperplane that separates $\mathbf{y}$ from $\mathcal{K}$ with sufficient margin (generalized in the sense that it separates w.r.t. to a given positive definite matrix $\mathbf{A}$, see in the sequel), or find a feasible point that is sufficiently close to $\mathbf{y}$ (in terms of the distance induced by the matrix $\mathbf{A}$). In case such a generalized hyperplane is found, it can then be used to "pull" the infeasible point closer to $\mathcal{K}$, and the process repeats itself.

We show that by applying the classical LOO-based Frank-Wolfe method (Frank and Wolfe, 1956; Jaggi, 2013) to the non-Euclidean projection problem $\min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}\|_{\mathbf{A}}^2$, we can indeed either find such a separating hyperplane, or find a close-enough feasible point, w.r.t. the matrix $\mathbf{A}$.

One may wonder: *if we can directly approximate matrix-based projections, arbitrarily well, using Frank-Wolfe, why do we need to go through the (conceptually more complex) approach of using separating hyperplanes?* The reason is that, has already discussed in Garber and Kretzu (2022), such a simplified approach will lead to a worse regret/oracle complexity tradeoff (mainly in terms of T). In particular, when applying Frank-Wolfe to the problem $\min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}\|_{\mathbf{A}}^2$, we will only compute a feasible point that is an approximated projection. On the other hand, with our approach (recall the definition of the AFP oracle) we always return a valid (though infeasible)

projection (and a feasible point that is sufficiently close to it), which allows for a tighter regret analysis.

The following lemma shows how given an infeasible point $\mathbf{y}$ and such a generalized separating hyperplane, we can "pull" $\mathbf{y}$ closer to the feasible set.

Lemma 6 *Let $\mathcal{K} \subset \mathbb{R}^n$ be convex and compact, let $\mathbf{A} \in \mathbb{S}_{++}^n$, and let $\mathbf{y} \in \mathbb{R}^n \setminus \mathcal{K}$. Let $\mathbf{g} \in \mathbb{R}^n$ be such that for all $\mathbf{z} \in \mathcal{K}$, $(\mathbf{y} - \mathbf{z})^\top \mathbf{A} \mathbf{g} \geq Q$, for some $Q \geq 0$. Consider the point $\widetilde{\mathbf{y}} = \mathbf{y} - \gamma \mathbf{g}$ for $\gamma = Q/C^2$, where $C \geq \|\mathbf{g}\|_{\mathbf{A}}$. It holds that*

$$\forall \mathbf{z} \in \mathcal{K}: \quad \|\widetilde{\mathbf{y}} - \mathbf{z}\|_{\mathbf{A}}^2 \leq \|\mathbf{y} - \mathbf{z}\|_{\mathbf{A}}^2 - (Q/C)^2.$$

Proof Fix some $\mathbf{z} \in \mathcal{K}$. It holds that

$$\|\widetilde{\mathbf{y}} - \mathbf{z}\|_{\mathbf{A}}^2 = \|\mathbf{y} - \mathbf{z} - \gamma \mathbf{g}\|_{\mathbf{A}}^2 = \|\mathbf{y} - \mathbf{z}\|_{\mathbf{A}}^2 - 2\gamma(\mathbf{y} - \mathbf{z})^\top \mathbf{A} \mathbf{g} + \gamma^2 \|\mathbf{g}\|_{\mathbf{A}}^2.$$

Since $(\mathbf{y} - \mathbf{z})^\top \mathbf{A} \mathbf{g} \geq Q$ and $C \geq \|\mathbf{g}\|_{\mathbf{A}}$, we indeed obtain

$$\|\widetilde{\mathbf{y}} - \mathbf{z}\|_{\mathbf{A}}^2 \leq \|\mathbf{y} - \mathbf{z}\|_{\mathbf{A}}^2 - 2\gamma Q + \gamma^2 C^2 = \|\mathbf{y} - \mathbf{z}\|_{\mathbf{A}}^2 - Q^2/C^2,$$

where the last equality follows from plugging-in the value of γ. ∎

Algorithm 2 given below, which simply applies the Frank-Wolfe method (with line-search) for smooth convex minimization over a convex and compact set (Jaggi, 2013) to the non-Euclidean projection problem $\min_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}\|_{\mathbf{A}}^2$, returns some feasible point $\widetilde{\mathbf{x}} \in \mathcal{K}$, that is either close enough (w.r.t. $\|\cdot\|_{\mathbf{A}}$) to the infeasible point $\mathbf{y}$, or can be used to construct a hyperplane which separates $\mathbf{y}$ from $\mathcal{K}$ w.r.t. $\mathbf{A}$ and with sufficient margin.

Algorithm 2: Generalized Separating Hyperplane via Frank-Wolfe

---

Data: LOO for the feasible set $\mathcal{K}$, error tolerance $\epsilon > 0$, initial point $\mathbf{x}_1 \in \mathcal{K}$, $\mathbf{A} \in \mathbb{S}_{++}^n$,
    infeasible point $\mathbf{y}$

for $i = 1, 2, \ldots$ **do**

　　$\mathbf{v}_i \in \operatorname*{argmin}_{\mathbf{x} \in \mathcal{K}} \{(\mathbf{x}_i - \mathbf{y})^\top \mathbf{A} \mathbf{x}\};$　　　　　　　　　　/* call to LOO of $\mathcal{K}$ */

　　if $(\mathbf{x}_i - \mathbf{y})^\top \mathbf{A}(\mathbf{x}_i - \mathbf{v}_i) \leq \epsilon$ *or* $\|\mathbf{x}_i - \mathbf{y}\|_{\mathbf{A}}^2 \leq 3\epsilon$ **then**

　　　　return $\widetilde{\mathbf{x}} \leftarrow \mathbf{x}_i$

　　$\sigma_i = \operatorname*{argmin}_{\sigma \in [0,1]} \{\|\mathbf{y} - \mathbf{x}_i - \sigma(\mathbf{v}_i - \mathbf{x}_i))\|_{\mathbf{A}}^2\}$

　　$\mathbf{x}_{i+1} = \mathbf{x}_i + \sigma_i(\mathbf{v}_i - \mathbf{x}_i)$

end

Lemma 7 *Algorithm 2 terminates after at most $\lceil (27R^2 \lambda_1(\mathbf{A})/\epsilon) - 2 \rceil$ iterations, and returns a point $\widetilde{\mathbf{x}} \in \mathcal{K}$ satisfying:*

1. *$\|\widetilde{\mathbf{x}} - \mathbf{y}\|_{\mathbf{A}}^2 \leq \|\mathbf{x}_1 - \mathbf{y}\|_{\mathbf{A}}^2$.*

2. *At least one of the following holds: $\|\widetilde{\mathbf{x}} - \mathbf{y}\|_{\mathbf{A}}^2 \leq 3\epsilon$ or $\forall \mathbf{z} \in \mathcal{K} : (\mathbf{y} - \mathbf{z})^\top \mathbf{A}(\mathbf{y} - \widetilde{\mathbf{x}}) > (2/3)\|\widetilde{\mathbf{x}} - \mathbf{y}\|_{\mathbf{A}}^2$.*

3. *If $\mathrm{dist}^2_{\mathbf{A}}(\mathbf{y}, \mathcal{K}) \leq \epsilon$, then $\|\widetilde{\mathbf{x}} - \mathbf{y}\|^2_{\mathbf{A}} \leq 3\epsilon$.*

Proof As discussed, Algorithm 2 is simply the well-known Frank-Wolfe method with line-search, see Algorithm 3 in Jaggi (2013), when applied to minimizing the convex and $\lambda_1(\mathbf{A})$-smooth function $g(\mathbf{x}) := \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2_{\mathbf{A}}$, whose gradient vector is given by $\nabla g(\mathbf{x}) = \mathbf{A}(\mathbf{x} - \mathbf{y})$, over the set $\mathcal{K}$. Thus, the upper-bound on the number of iterations executed by Algorithm 2 follows immediately from Theorem 2 in Jaggi (2013), which gives a convergence rate for the dual gap. For our choice of g, the dual gap on any iteration i is given precisely by $\nabla g(\mathbf{x}_i)^\top(\mathbf{x}_i - \mathbf{v}_i) = (\mathbf{x}_i - \mathbf{y})^\top\mathbf{A}(\mathbf{x}_i - \mathbf{v}_i)$, which corresponds to one of the stopping conditions is Algorithm 2.

Since the line-search guarantees that the function value $g(\mathbf{x}_i) = \frac{1}{2}\|\mathbf{x}_i - \mathbf{y}\|^2_{\mathbf{A}}$ does not increase when moving from iterate $\mathbf{x}_i$ to $\mathbf{x}_{i+1}$, Item 1 holds trivially.

Item 2 follows from the stopping condition of the algorithm and by noting that, if for some iteration i it holds that $(\mathbf{x}_i - \mathbf{y})^\top\mathbf{A}(\mathbf{x}_i - \mathbf{v}_i) \leq \epsilon$ and $\|\mathbf{x}_i - \mathbf{y}\|^2_{\mathbf{A}} > 3\epsilon$ (in which case the algorithm will return $\widetilde{\mathbf{x}} = \mathbf{x}_i$) then, for all $\mathbf{z} \in \mathcal{K}$ it holds that

$$(\mathbf{z} - \mathbf{y})^\top \mathbf{A}(\mathbf{x}_i - \mathbf{y}) = (\mathbf{z} - \mathbf{x}_i)^\top \mathbf{A}(\mathbf{x}_i - \mathbf{y}) + \|\mathbf{x}_i - \mathbf{y}\|^2_{\mathbf{A}} \geq (\mathbf{v}_i - \mathbf{x}_i)^\top \mathbf{A}(\mathbf{x}_i - \mathbf{y}) + \|\mathbf{x}_i - \mathbf{y}\|^2_{\mathbf{A}}$$
$$\geq -\epsilon + \|\mathbf{x}_i - \mathbf{y}\|^2_{\mathbf{A}} > -(\|\mathbf{x}_i - \mathbf{y}\|^2_{\mathbf{A}}/3) + \|\mathbf{x}_i - \mathbf{y}\|^2_{\mathbf{A}} = (2/3)\|\mathbf{x}_i - \mathbf{y}\|^2_{\mathbf{A}},$$

where the first inequality is due to the definition of $\mathbf{v}_i$.

Finally, to prove Item 3, denote $\mathbf{x}^* = \mathrm{argmin}_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}\|^2_{\mathbf{A}}$. Suppose by contradiction that $\mathrm{dist}^2_{\mathbf{A}}(\mathbf{y}, \mathcal{K}) = \|\mathbf{x}^* - \mathbf{y}\|^2_{\mathbf{A}} \leq \epsilon$, and $\|\widetilde{\mathbf{x}} - \mathbf{y}\|^2_{\mathbf{A}} > 3\epsilon$. By the stopping condition of the algorithm, on the last iteration executed i, it must hold that $(\widetilde{\mathbf{x}} - \mathbf{y})^\top\mathbf{A}(\widetilde{\mathbf{x}} - \mathbf{v}_i) = \max_{\mathbf{v} \in \mathcal{K}}\nabla g(\widetilde{\mathbf{x}})^\top(\widetilde{\mathbf{x}} - \mathbf{v}) \leq \epsilon$, which means that

$$\|\widetilde{\mathbf{x}} - \mathbf{y}\|^2_{\mathbf{A}} - \mathrm{dist}^2_{\mathbf{A}}(\mathbf{y}, \mathcal{K}) = 2g(\widetilde{\mathbf{x}}) - 2g(\mathbf{x}^*) \leq 2\nabla g(\widetilde{\mathbf{x}})^\top(\widetilde{\mathbf{x}} - \mathbf{x}^*) \leq 2\max_{\mathbf{v} \in \mathcal{K}}\nabla g(\widetilde{\mathbf{x}})^\top(\widetilde{\mathbf{x}} - \mathbf{v}) \leq 2\epsilon,$$

where the first inequality is due to the gradient inequality and the convexity of $g(\cdot)$. Thus, we have that $\|\widetilde{\mathbf{x}} - \mathbf{y}\|^2_{\mathbf{A}} \leq 2\epsilon + \mathrm{dist}^2_{\mathbf{A}}(\mathbf{y}, \mathcal{K}) \leq 3\epsilon$, which contradicts the assumption that $\|\widetilde{\mathbf{x}} - \mathbf{y}\|^2_{\mathbf{A}} > 3\epsilon$. ■

Our LOO-based implementation of a AFP oracle for the feasible set $\mathcal{K}$ is given as Algorithm 3. The algorithm builds on iteratively using separating hyperplanes generated by Algorithm 2 to "pull closer" the infeasible point $\mathbf{y}$ towards the feasible set $\mathcal{K}$ using the updates suggested in Lemma 6, until it is sufficiently close.

The proof of the following lemma is given in the appendix.

Lemma 8 *Setting $\gamma = 2/3$ in Algorithm 3 guarantees that it stops after at most*

$$\max\left\{2.25\log\left(\frac{\|\mathbf{y}_1 - \mathbf{x}_0\|^2_{\mathbf{A}}}{\epsilon}\right) + 1, 0\right\}$$

iterations, and returns $(\mathbf{x}, \mathbf{y}) \in \mathcal{K} \times \left(R + \sqrt{3\epsilon/\lambda_n(\mathbf{A})}\right)\mathcal{B}$ such that

$$\forall \mathbf{z} \in \mathcal{K}: \|\mathbf{y} - \mathbf{z}\|^2_{\mathbf{A}} \leq \|\mathbf{y}_1 - \mathbf{z}\|^2_{\mathbf{A}} \quad and \quad \|\mathbf{x} - \mathbf{y}\|^2_{\mathbf{A}} \leq 3\epsilon.$$

It is important to note that Lemma 8 significantly and critically improves upon its Euclidean counterpart in Garber and Kretzu (2022): while the number of iterations here scales only with $\log(1/\epsilon)$, in Garber and Kretzu (2022) it scales with $1/\epsilon^2$. This improvement is critical for obtaining our improved regret/oracle complexity tradeoffs.

Algorithm 3: Approximately-Feasible (matrix) Projection via a Linear Optimization Oracle

---

Data: LOO for the feasible set $\mathcal{K}$, feasible point $\mathbf{x}_0 \in \mathcal{K}$, initial point $\mathbf{y}_1 \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{S}_{++}^n$,
     error tolerance $\epsilon > 0$, step-size $\gamma > 0$

if $\|\mathbf{x}_0 - \mathbf{y}_1\|_{\mathbf{A}}^2 \leq 3\epsilon$ **then**
  | **Return** $\mathbf{x} \leftarrow \mathbf{x}_0, \mathbf{y} \leftarrow \mathbf{y}_1$
end
for $i = 1, 2, \dots$ **do**
  | $\mathbf{x}_i \leftarrow$ Output of Algorithm 2 when called with LLO of $\mathcal{K}$, tolerance ϵ, feasible point $\mathbf{x}_{i-1}$,
  | positive definite matrix $\mathbf{A}$, and initial point $\mathbf{y}_i$
  | **if** $\|\mathbf{x}_i - \mathbf{y}_i\|_{\mathbf{A}}^2 > 3\epsilon$ **then**
  | | $\mathbf{y}_{i+1} = \mathbf{y}_i - \gamma \left(\mathbf{y}_i - \mathbf{x}_i \right)$
  | **else**
  | | **Return** $\mathbf{x} \leftarrow \mathbf{x}_i, \mathbf{y} \leftarrow \mathbf{y}_i$
  | **end**
end

---

4. LOO-based Online Newton Step

In this section we present our main result — an efficient LOO-based ONS-style algorithm and its regret and complexity guarantees.

The following lemma builds on the combination of our ONS Without Feasibility template (Algorithm 1) together with our LOO-based construction for an AFP oracle (Algorithm 3). The proof is given in the appendix.

Lemma 9 *Fix block size $K \in [T]$. Consider running Algorithm 1 with parameters $\eta, \epsilon, \epsilon_I$ such that $\eta \geq \max\{12KGR, \frac{2K}{\alpha}\}, \epsilon_I \geq (KG)^2$, and $\frac{3\epsilon}{\epsilon_I} \leq 4R^2$, and when the $\mathcal{O}_{AFP}$ oracle is implemented via Algorithm 3, where the initial feasible input to Algorithm 3 (the point $\mathbf{x}_0$ in Algorithm 3), when called during block m in Algorithm 1, is the previous feasible output of Algorithm 3 — the point $\mathbf{x}_m$, if $m \geq 2$, and the initialization point of Algorithm 1 (the point $\mathbf{x}_1$), if $m = 1$. Then, the regret is upper bounded by*

$$\sum_{t=1}^{T} f_t(\mathbf{x}^t) - \min_{\mathbf{x}^* \in \mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}^*) \leq \frac{3\beta\epsilon}{\epsilon_I} T + \sqrt{\frac{6\epsilon T}{K} \sum_{m=1}^{T/K} \|\bar{\nabla}_m\|_{\mathbf{A}_m^{-1}}^2} + \frac{2R^2\epsilon_I}{\eta} + \frac{\eta}{2} \sum_{m=1}^{T/K} \|\nabla_m\|_{\mathbf{A}_m^{-1}}^2 \,,$$

and the overall number of calls to the LOO of $\mathcal{K}$ is upper bounded by

$$N_{calls} \leq 61R^2 \log\left(19 + 4\frac{\eta^2 K^2 G^2}{\epsilon \epsilon_I}\right) \frac{\epsilon_I + G^2 KT}{K\epsilon} T.$$

We are now ready to formally present our main result. Here for ease of presentation we present a concise version only. A fully detailed version which specifics all choices of parameters and all poly-logarithmic factors, as well as the proof, is given in the appendix.

Theorem 10 (short version) *Consider the implementation of Algorithm 1 as described in Lemma 9 and when using the (standard ONS) update rule: $\mathbf{A}_m = \mathbf{A}_{m-1} + \bar{\nabla}_m \bar{\nabla}_m^\top$ for every block m.*

1. *If $T \geq T_0 = \widetilde{O}(1)$, there exists a choice for the parameters $K, \eta, \epsilon, \epsilon_I$ in Algorithm 1 which depends only on the quantities T, n, G, R, α and satisfies the assumptions of Lemma 9, such that the regret is upper-bounded by*

$$\mathcal{R}_T = \widetilde{O}\left((\beta R^2 + (GR + \alpha^{-1})n^{2/3})T^{2/3}\right). \tag{6}$$

2. *In continue to the previous item and under the same choice of parameters, for any $\rho \in [n]$, denoting $\Omega_\rho = \sum_{i=\rho+1}^{n} \lambda_i(\sum_{t=1}^{T} \nabla_t \nabla_t^\top)$ (∇_t is as defined in Algorithm 1), the regret is upper-bounded by*

$$\mathcal{R}_T = \widetilde{O}\left((\beta R^2 + GR(\rho^{1/2}n^{1/6} + n^{1/3}) + \alpha^{-1}n^{-1/3}\rho)T^{2/3}\right)$$
$$+ \widetilde{O}\left(RT^{1/3}\sqrt{\Omega_\rho} + G^{-2}n^{-2/3}(GR + \alpha^{-1})\Omega_\rho\right). \tag{7}$$

3. *Fix $\rho \in [n]$. If $T \geq T_0 = \widetilde{O}(1)$, there exists a choice for the parameters $K, \eta, \epsilon, \epsilon_I$ in Algorithm 1 which depends only on the quantities T, n, G, R, α and ρ, and satisfies the assumptions of Lemma 9, such that the regret is upper-bounded by*

$$\mathcal{R}_T = \widetilde{O}\left((\beta R^2 + (GR + \alpha^{-1})\rho^{2/3})T^{2/3} + RT^{1/3}\sqrt{\Omega_\rho} + G^{-2}\rho^{-2/3}(GR + \alpha^{-1})\Omega_\rho\right). \tag{8}$$

   Note this bound is not explicitly dependent on the ambient dimension n.

In all cases, the overall number of calls to the LOO of $\mathcal{K}$ is upper-bounded by $O(T + n^{1/3}T^{2/3})$, the additional space requirement is $O(n^2)$, and using the Sherman-Morrison formula for fast matrix inversion, the overall additional runtime is $O(n^2(T + n^{1/3}T^{2/3}))$.

Let us make a few comments regarding Theorem 10. The regret bounds (7), (8) may significantly improve upon the worst case bound (6) in case the observed gradients approximately span a subspace of dimension at most ρ, for some $\rho \in [n]$, in the sense that $\Omega_\rho = O(T^{2/3})$ (note that $\Omega_\rho = 0$ implies that the dimension of the subspace spanned by the gradients is at most ρ). In particular, the bound (7) holds simultaneously for all values of ρ (i.e., the algorithm is independent of the choice of ρ), but still depends on the ambient dimension n (though with milder dependence than (6)), while the bound (8) is completely independent of n, but requires a priori knowledge of ρ. In case it indeed holds that $\Omega_\rho = O(T^{2/3})$ for some known $\rho << n$, (8) translates into a $\widetilde{O}\left((\beta R^2 + (GR + \alpha^{-1})\rho^{2/3})T^{2/3}\right)$ regret bound.

5. Leveraging Frequent Directions Sketching for Low-dimensional Data

While Theorem 10 yields a regret bound for Algorithm 1 which is independent of the ambient dimension n and depends only on the (approximate) dimension of the subspace spanned by the gradients (guarantee (8)), the space and average additional runtime requirements still scale with n^2. Following the approach of Luo et al. (2016), who considered the coupling of ONS with matrix sketching techniques to reduce space and runtime requirements in case of low-dimensional data (but not in a projection-free setting), in this section we discuss the implications of such coupling to our LOO-based algorithm.

Similarly to Luo et al. (2016), we consider the use of the well known deterministic *Frequent Directions* sketching method (Ghashami et al., 2016). The idea is that instead of taking the matrix $\mathbf{A}_m$ for each block m in Algorithm 1 to be the exact aggregation of gradients as in Theorem 10 and maintain it (and its inverse $\mathbf{A}_m^{-1}$) explicitly, we shall only maintain a certain approximation of this gradient information in a low-rank factorized form, see Algorithm 4 which shows how the Frequent Directions sketch is used in synergy with Algorithm 1.

Algorithm 4: Frequent Directions Sketch for Algorithm 1

Data: sketch size $\rho \in [n]$, $\epsilon_I > 0$

Initialization: Set $\mathbf{S}_0 = \mathbf{0}_{(\rho+1) \times n}$, and $\mathbf{A}_0 = \epsilon_I \mathbf{I}_n$

for $m = 1$ *to* T/K **do**

  Receive $\bar{\nabla}_m \in \mathbb{R}^n$ from Algorithm 1 and insert it as the last row of $\mathbf{S}_{m-1}$

  Compute eigendecomposition of $\mathbf{S}_{m-1}^\top \mathbf{S}_{m-1}$: $\mathbf{V}_m^\top \widehat{\Sigma}_m \mathbf{V}_m = \mathbf{S}_{m-1}^\top \mathbf{S}_{m-1}$

  Set $\sigma_m = \widehat{\Sigma}_m (\rho+1, \rho+1)$ and $\Sigma_m = \widehat{\Sigma}_m - \sigma_m \mathbf{I}_{\rho+1}$ ▷ $\Sigma_m(\rho+1, \rho+1) = 0$

  Set $\mathbf{S}_m = (\Sigma_m)^{\frac{1}{2}} \mathbf{V}_m$ ▷ last row of $\mathbf{S}_m$ is now $\mathbf{0}$

  Set $\mathbf{H}_m = \mathbf{diag}\left(\frac{1}{\epsilon_I + \Sigma_m(1,1)}, \ldots, \frac{1}{\epsilon_I + \Sigma_m(\rho,\rho)}, \frac{1}{\epsilon_I}\right)$ ▷ $\mathbf{H}_m = \left(\epsilon_I \mathbf{I}_{\rho+1} + \mathbf{S}_m \mathbf{S}_m^\top\right)^{-1}$

  Set $\mathbf{A}_m = \mathbf{A}_0 + \mathbf{S}_m^\top \mathbf{S}_m$, $\mathbf{A}_m^{-1} = \epsilon_I^{-1} \left(\mathbf{I}_n - \mathbf{S}_m^\top \mathbf{H}_m \mathbf{S}_m\right)$ ▷ not to be explicitly computed;

  the expression for $\mathbf{A}_m^{-1}$ follows from the Woodbury matrix identity

end

The full version of the following theorem, as well as the proof and additional details regarding Algorithm 4, are given in the appendix.

Theorem 11 *Fix $\rho \in [n]$. Consider the implementation of Algorithm 1 as described in Lemma 9, and when the matrix $\mathbf{A}_m$ for every block m in Algorithm 1 is generated by Algorithm 4. Denote $\Omega_\rho = \sum_{i=\rho+1}^n \lambda_i \left(\sum_{t=1}^T \nabla_t \nabla_t^\top\right)$. If $T \geq T_0 = \widetilde{O}(1)$, then there exists a choice for the parameters $K, \eta, \epsilon, \epsilon_I$ in Algorithm 1 which depends only on the quantities T, ρ, G, R, α and satisfies the assumptions of Lemma 9, such that the regret is upper bounded by*

$$\mathcal{R}_T = \widetilde{O}\left(\left(\beta R^2 + \left(GR + \alpha^{-1}\right)\rho^{2/3}\right)T^{2/3} + R\rho^{1/2}T^{1/3}\sqrt{\Omega_\rho} + G^{-2}\rho^{1/3}\left(GR + \alpha^{-1}\right)\Omega_\rho\right).$$

The overall number of calls to the LOO is upper bounded by $O\left(\rho^{1/3}T^{2/3} + T\right)$, the additional space requirement is $O(\rho n)$, and the overall additional runtime is $O(\rho n T + \rho^{4/3} n T^{2/3} + \rho^{7/3} n T^{1/3})$.

6. Discussion

We provided the first projection-free LOO-based algorithm for exp-concave and smooth losses that in the case of (approximately) low-dimensional gradients, using $O(T)$ queries to the LOO, guarantees regret that both scales only with $T^{2/3}$, and is independent of the ambient dimension.

It is interesting if a similar result could be obtained when removing one or more of the above assumptions: smoothness of the losses, exp-concavity of the losses, low-dimensionality of the gradients. In particular, the two recent works Hazan and Minasyan (2020); Mhammedi (2022) achieve fast LLO-based regret bounds that scale with $T^{2/3}$ (but also with the dimension) without curvature assumptions on the losses. It is thus interesting whether the exp-concavity assumption, or even strong convexity (Kretzu and Garber, 2021), could lead to even faster rates than $T^{2/3}$.

References

Lin Chen, Christopher Harshaw, Hamed Hassani, and Amin Karbasi. Projection-free online optimization with stochastic gradient: From convexity to submodularity. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 814–823. PMLR, 10–15 Jul 2018.

Lin Chen, Mingrui Zhang, and Amin Karbasi. Projection-free bandit convex optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2047–2056. PMLR, 2019.

Alina Ene, Huy L Nguyen, and Adrian Vladu. Projection-free bandit optimization with privacy guarantees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7322–7330, 2021.

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:149–154, 1956.

Dan Garber and Elad Hazan. Playing non-linear games with linear oracles. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 420–428. IEEE, 2013.

Dan Garber and Ben Kretzu. Improved regret bounds for projection-free bandit convex optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 2196–2206. PMLR, 2020.

Dan Garber and Ben Kretzu. New projection-free algorithms for online convex optimization with adaptive regret guarantees. In Po-Ling Loh and Maxim Raginsky, editors, *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pages 2326–2359. PMLR, 02–05 Jul 2022.

Mina Ghashami, Edo Liberty, Jeff M Phillips, and David P Woodruff. Frequent directions: Simple and deterministic matrix sketching. *SIAM Journal on Computing*, 45(5):1762–1792, 2016.

Elad Hazan. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207*, 2019.

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

Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2):169–192, 2007.

Elad E Hazan and Satyen Kale. Projection-free online learning. In *29th International Conference on Machine Learning, ICML 2012*, pages 521–528, 2012.

Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML (1)*, pages 427–435, 2013.

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.

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, 2019.

Zhou Lu, Nataly Brukhim, Paula Gradu, and Elad Hazan. Projection-free adaptive regret with membership oracles. *arXiv preprint arXiv:2211.12638*, 2022.

Haipeng Luo, Alekh Agarwal, Nicolo Cesa-Bianchi, and John Langford. Efficient second order online learning by sketching. *Advances in Neural Information Processing Systems*, 29, 2016.

Zakaria Mhammedi. Efficient projection-free online convex optimization with membership oracle. *arXiv preprint arXiv:2111.05818*, 2021.

Zakaria Mhammedi. Exploiting the curvature of feasible sets for faster projection-free online learning, 2022. URL https://arxiv.org/abs/2205.11470.

Zakaria Mhammedi and Khashayar Gatmiry. Quasi-newton steps for efficient online exp-concave optimization. *arXiv preprint arXiv:2211.01357*, 2022.

Shai Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

Yin Tat Lee, Aaron Sidford, and Santosh S Vempala. Efficient convex optimization with membership oracles. *arXiv e-prints*, pages arXiv–1706, 2017.

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.

Wenpeng Zhang, Peilin Zhao, Wenwu Zhu, Steven CH Hoi, and Tong Zhang. Projection-free distributed online learning in networks. In *International Conference on Machine Learning*, pages 4054–4062. PMLR, 2017.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936, 2003.

Appendix A. Proof of Lemma 8

Proof First, we note that in the special case that $\|\mathbf{x}_0 - \mathbf{y}_1\|_{\mathbf{A}}^2 \leq 3\epsilon$, since $\|\mathbf{y}_1\| - \|\mathbf{x}_0\| \leq \|\mathbf{x}_0 - \mathbf{y}_1\| \leq \sqrt{\lambda_n^{-1}(\mathbf{A})}\|\mathbf{x}_0 - \mathbf{y}_1\|_{\mathbf{A}}$, and since $\mathbf{x} \in \mathcal{K}$, it holds that $\|\mathbf{y}_1\| \leq R + \sqrt{3\epsilon\lambda_n^{-1}(\mathbf{A})}$, and the lemma holds trivially.

For the remaining of the proof we shall assume that $\|\mathbf{x}_0 - \mathbf{y}_1\|_{\mathbf{A}}^2 > 3\epsilon$. Let us denote by $k \geq 1$ the overall number of for loop iterations executed in Algorithm 3, i.e., $\|\mathbf{y}_k - \mathbf{x}_k\|_{\mathbf{A}}^2 \leq 3\epsilon$ and $\|\mathbf{y}_i - \mathbf{x}_i\|_{\mathbf{A}}^2 > 3\epsilon$ for all $i < k$. Using Lemma 7 we have that for all $i < k$ it holds that $(\mathbf{y}_i - \mathbf{z})^\top \mathbf{A} (\mathbf{y}_i - \mathbf{x}_i) \geq (2/3)\|\mathbf{y}_i - \mathbf{x}_i\|_{\mathbf{A}}^2$ for every $\mathbf{z} \in \mathcal{K}$. Thus, using Lemma 6 with $\mathbf{g} = (\mathbf{y}_i - \mathbf{x}_i), C = \|\mathbf{y}_i - \mathbf{x}_i\|_{\mathbf{A}}, Q = (2/3)\|\mathbf{y}_i - \mathbf{x}_i\|_{\mathbf{A}}^2$, and $\gamma = Q/C^2 = 2/3$, we have that for every $i < k$,

$$\forall \mathbf{z} \in \mathcal{K}: \quad \|\mathbf{y}_{i+1} - \mathbf{z}\|_{\mathbf{A}}^2 \leq \|\mathbf{y}_i - \mathbf{z}\|_{\mathbf{A}}^2 - (4/9)\|\mathbf{y}_i - \mathbf{x}_i\|_{\mathbf{A}}^2. \tag{9}$$

Thus, we obtain that for all $\mathbf{z} \in \mathcal{K}$ it holds that $\|\mathbf{y}_k - \mathbf{z}\|_{\mathbf{A}}^2 \le \|\mathbf{y}_1 - \mathbf{z}\|_{\mathbf{A}}^2$.

Now we continue to upper-bound the number of iterations until Algorithm 3 stops. Denote $\mathbf{x}_i^* = \operatorname{argmin}_{\mathbf{x} \in \mathcal{K}} \|\mathbf{y}_i - \mathbf{x}\|_{\mathbf{A}}^2$ for every iteration $i < k$. Using Eq. (9), for every iteration $i < k$ it holds that

$$\operatorname{dist}_{\mathbf{A}}^2(\mathbf{y}_{i+1}, \mathcal{K}) = \|\mathbf{y}_{i+1} - \mathbf{x}_{i+1}^*\|_{\mathbf{A}}^2 \le \|\mathbf{y}_{i+1} - \mathbf{x}_i^*\|_{\mathbf{A}}^2$$
$$\le \|\mathbf{y}_i - \mathbf{x}_i^*\|_{\mathbf{A}}^2 - (4/9)\|\mathbf{y}_i - \mathbf{x}_i\|_{\mathbf{A}}^2 \le e^{-(4/9)}\operatorname{dist}_{\mathbf{A}}^2(\mathbf{y}_i, \mathcal{K}),$$

where the last inequality is since $\operatorname{dist}_{\mathbf{A}}^2(\mathbf{y}_i, \mathcal{K}) = \|\mathbf{y}_i - \mathbf{x}_i^*\|_{\mathbf{A}}^2 \le \|\mathbf{y}_i - \mathbf{x}_i\|_{\mathbf{A}}^2$, and by using the inequality $1 - x \le e^{-x}$. Unrolling the recursion we have,

$$\operatorname{dist}_{\mathbf{A}}^2(\mathbf{y}_k, \mathcal{K}) \le e^{-(4/9)(k-1)}\operatorname{dist}_{\mathbf{A}}^2(\mathbf{y}_1, \mathcal{K}) \le e^{-(4/9)(k-1)}\|\mathbf{y}_1 - \mathbf{x}_0\|_{\mathbf{A}}^2.$$

Thus, after at most $k - 1 = 2.25 \log\left(\|\mathbf{y}_1 - \mathbf{x}_0\|_{\mathbf{A}}^2 / \epsilon\right)$ iterations, we obtain $\operatorname{dist}_{\mathbf{A}}^2(\mathbf{y}_k, \mathcal{K}) \le \epsilon$, which by using Lemma 7, implies that the next iteration will be the last one, and the points $\mathbf{x}_k, \mathbf{y}_k$ will indeed satisfy $\|\mathbf{x}_k - \mathbf{y}_k\|_{\mathbf{A}}^2 \le 3\epsilon$. This proves the upper-bound on the overall number of for loop iterations.

Finally, note that $\mathbf{x} \in \mathcal{K}$ since it is the output of Algorithm 2, and since $\|\mathbf{y}\| - \|\mathbf{x}\| \le \|\mathbf{x} - \mathbf{y}\| \le \sqrt{\lambda_n^{-1}(\mathbf{A})}\|\mathbf{x} - \mathbf{y}\|_{\mathbf{A}}$, we obtain $\|\mathbf{y}\| \le R + \sqrt{3\epsilon\lambda_n^{-1}(\mathbf{A})}$ as required. ∎

Appendix B. Full Version of Theorem 10 and Proof

Before we state the full version of Theorem 10 and prove it, we first prove Lemma 9 and an additional lemma.

Proof [Proof of Lemma 9] First, note that since for every block m in Algorithm 1 we have that $\mathbf{A}_m \succeq \mathbf{A}_0 = \epsilon_I \mathbf{I}_n$, it holds that $\lambda_n(\mathbf{A}_m) \ge \epsilon_I$. By our assumption that $\frac{3\epsilon}{\epsilon_I} \le 4R^2$, using Lemma 8, it follows that for every block m in Algorithm 1 we have that $\widetilde{\mathbf{y}}_m \in 3R\mathcal{B}$, which is in accordance with the assumption of Lemma 5. Since $\eta \ge \max\{12KGR, \frac{2K}{\alpha}\}$, and $\epsilon_I \ge (KG)^2$, the regret bound stated in the lemma follows immediately from the one in Lemma 5.

Now we move on to upper-bound the overall number of calls to the LOO. We note that Eq. (1) holds here as well from the same arguments stated in the proof of Lemma 5. Recall that the update step of Algorithm 1 is $\mathbf{y}_{m+1} = \widetilde{\mathbf{y}}_m - \eta \mathbf{A}_m^{-1} \bar{\nabla}_m$. Thus, by using Eq. (1), and the fact that the points $\mathbf{x}_m, \widetilde{\mathbf{y}}_m$ are the outputs of $\mathcal{O}_{AFP}$ when called from Algorithm 1 with the input point $\mathbf{y}_m$, the positive definite matrix $\mathbf{A}_{m-1}$, and an error tolerance of 3ϵ, we have that

$$\|\mathbf{x}_m - \mathbf{y}_{m+1}\|_{\mathbf{A}_m} \le \|\mathbf{x}_m - \widetilde{\mathbf{y}}_m\|_{\mathbf{A}_m} + \|\widetilde{\mathbf{y}}_m - \mathbf{y}_{m+1}\|_{\mathbf{A}_m} \le \sqrt{6\epsilon} + \eta\|\mathbf{A}_m^{-1}\bar{\nabla}_m\|_{\mathbf{A}_m}.$$

Using $(a + b)^2 \le 2a^2 + 2b^2$ and since $\lambda_1(\mathbf{A}_m^{-1}) = \lambda_n^{-1}(\mathbf{A}_m) \le \lambda_n^{-1}(\mathbf{A}_0) = 1/\epsilon_I$, we have that for any block m in Algorithm 1,

$$\|\mathbf{x}_m - \mathbf{y}_{m+1}\|_{\mathbf{A}_m}^2 \le 12\epsilon + 2\eta^2\|\bar{\nabla}_m\|_{\mathbf{A}_m^{-1}}^2 \le 12\epsilon + 2\eta^2 K^2 G^2/\epsilon_I, \tag{10}$$

where the last inequality also uses the fact that $\|\bar{\nabla}_m\| \le KG$.

Using Lemma 8, each call to Algorithm 3 on some block m of Algorithm 1, makes at most $\max\{2.25 \log\left(\|\mathbf{x}_m - \mathbf{y}_{m+1}\|_{\mathbf{A}_m}^2 / \epsilon\right) + 1, 0\}$ iterations. On each such iteration it calls Algorithm 2

which, according to Lemma 7, makes at most $\left\lceil \frac{27R^2\lambda_1(\mathbf{A}_m)}{\epsilon} - 2 \right\rceil$ calls to the LOO. Recall that by the update rule of Algorithm 1: $\lambda_1(\mathbf{A}_m) \leq \lambda_1(\mathbf{A}_{m-1}) + \lambda_1(\bar{\nabla}_m \bar{\nabla}_m^\top) \leq \cdots \leq \epsilon_I + G^2 KT$ for every m, where we again used the fact that $\|\bar{\nabla}_m\| \leq KG$ for every block m. Thus, Algorithm 1, on each block m, makes at most

$$
\begin{aligned}
n_m &\leq \left(2.25 \log \left(\frac{12\epsilon + 2\eta^2 K^2 G^2/\epsilon_I}{\epsilon} \right) + 1 \right) \frac{27R^2(\epsilon_I + G^2 KT)}{\epsilon} \\
&= 2.25 \left(\log \left(\frac{12\epsilon + 2\eta^2 K^2 G^2/\epsilon_I}{\epsilon} \right) + \log \left(e^{\frac{4}{9}} \right) \right) \frac{27R^2(\epsilon_I + G^2 KT)}{\epsilon} \\
&\leq 2.25 \log \left(1.56 \frac{12\epsilon + 2\eta^2 K^2 G^2/\epsilon_I}{\epsilon} \right) \frac{27R^2(\epsilon_I + G^2 KT)}{\epsilon}
\end{aligned}
$$

calls to the LOO. Thus, the overall number of calls to the LOO throughout the run of Algorithm 1 is

$$
N_{calls} = \sum_{m=1}^{T/K} n_m \leq 61R^2 \log \left(19 + 4 \frac{\eta^2 K^2 G^2}{\epsilon \epsilon_I} \right) \frac{T}{K\epsilon} (\epsilon_I + G^2 KT).
$$

∎

Lemma 12 *Consider Algorithm 1 with the (standard ONS) update rule: $\mathbf{A}_m = \mathbf{A}_{m-1} + \bar{\nabla}_m \bar{\nabla}_m^\top$ for every block m. Then, for every $\rho \in [n]$ it holds that,*

$$
\sum_{m=1}^{T/K} \left\| \bar{\nabla}_m \right\|_{\mathbf{A}_m^{-1}}^2 \leq \rho \log \left(\frac{TKG^2 + \epsilon_I}{\epsilon_I} \right) + \frac{K}{\epsilon_I} \sum_{i=\rho+1}^{n} \lambda_i \left(\sum_{t=1}^{T} \nabla_t \nabla_t^\top \right).
$$

Proof For every $m \geq 1$ it holds that

$$
\left\| \bar{\nabla}_m \right\|_{\mathbf{A}_m^{-1}}^2 = \bar{\nabla}_m^\top \mathbf{A}_m^{-1} \bar{\nabla}_m = \mathbf{A}_m^{-1} \bullet \bar{\nabla}_m \bar{\nabla}_m^\top = \mathbf{A}_m^{-1} \bullet (\mathbf{A}_m - \mathbf{A}_{m-1}).
$$

Since $\mathbf{A}_m, \mathbf{A}_{m-1} \succ 0$, using Lemma 20, for every $m \geq 1$ we have that,

$$
\left\| \bar{\nabla}_m \right\|_{\mathbf{A}_m^{-1}}^2 \leq \log \frac{|\mathbf{A}_m|}{|\mathbf{A}_{m-1}|}.
$$

Summing over $m \in [T/K]$, we have

$$
\sum_{m=1}^{T/K} \|\bar{\nabla}_m\|_{\mathbf{A}_m^{-1}}^2 \leq \sum_{m=1}^{T/K} \log \frac{|\mathbf{A}_m|}{|\mathbf{A}_{m-1}|} = \log \frac{|\mathbf{A}_{T/K}|}{|\mathbf{A}_0|}.
$$

Since according to the update rule listed in the lemma we have that $\mathbf{A}_{T/K} = \epsilon_I \mathbf{I}_n + \sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top$, and $\|\bar{\nabla}_m\|^2 \leq K^2 G^2$ then, $\lambda_1(\mathbf{A}_{T/K}) \leq (\epsilon_I + TKG^2)$. Using Weyl's inequality for the eigenvalues, we have that for every $\rho \in [n]$ it holds that,

$$
\begin{aligned}
|\mathbf{A}_{T/K}| &= \prod_{i=1}^{\rho} \lambda_i \left(\epsilon_I \mathbf{I}_n + \sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top \right) \prod_{i=\rho+1}^{n} \lambda_i \left(\epsilon_I \mathbf{I}_n + \sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top \right) \\
&\leq \left(TKG^2 + \epsilon_I \right)^\rho \prod_{i=\rho+1}^{n} \left(\epsilon_I + \lambda_i \left(\sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top \right) \right).
\end{aligned}
$$

Recall that $\mathbf{A}_0 = \epsilon_I \mathbf{I}_n$. Since $\bar{\nabla}_m = \sum_{t=(m-1)K+1}^{mK} \nabla_t$, it holds that $K \sum_{t=1}^T \nabla_t \nabla_t^\top \succeq \sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top$ (Lemma 18). Thus, we have that

$$
\begin{aligned}
\frac{|\mathbf{A}_{T/K}|}{|\mathbf{A}_0|} &\leq \left(1 + \frac{TKG^2}{\epsilon_I}\right)^\rho \prod_{i=\rho+1}^n \left(1 + \frac{\lambda_i \left(K \sum_{t=1}^T \nabla_t \nabla_t^\top\right)}{\epsilon_I}\right) \\
&\leq \left(1 + \frac{TKG^2}{\epsilon_I}\right)^\rho \left(e^{(K/\epsilon_I) \sum_{i=\rho+1}^n \lambda_i\left(\sum_{t=1}^T \nabla_t \nabla_t^\top\right)}\right),
\end{aligned}
$$

where the last inequality follows from using $1 + x \leq e^x$.

Thus, we obtain

$$
\sum_{m=1}^{T/K} \|\bar{\nabla}_m\|^2_{\mathbf{A}_m^{-1}} \leq \log\left(\frac{|\mathbf{A}_{T/K}|}{|\mathbf{A}_0|}\right) \leq \rho \log\left(1 + \frac{TKG^2}{\epsilon_I}\right) + \frac{K}{\epsilon_I} \sum_{i=\rho+1}^n \lambda_i \left(\sum_{t=1}^T \nabla_t \nabla_t^\top\right).
$$

∎

Theorem 13 (Full version of Theorem 10) *Consider the implementation of Algorithm 1 as described in Lemma 9 and when using the (standard ONS) update rule: $\mathbf{A}_m = \mathbf{A}_{m-1} + \bar{\nabla}_m \bar{\nabla}_m^\top$ for every block m.*

1. *Suppose $T \geq T_0 = c \log^3\left(c + \left(c + \frac{c}{R^2 G^2 \alpha^2}\right) n^{-\frac{4}{3}} T^{\frac{1}{3}}\right)$, where $c > 0$ is a certain universal constant. Setting*

$$
\eta = 8 \max\{6GR, \frac{1}{\alpha}\} n^{-\frac{1}{3}} T^{\frac{2}{3}}, \quad K = 4n^{-\frac{1}{3}} T^{\frac{2}{3}}, \quad \epsilon_I = 32G^2 T^{\frac{4}{3}},
$$

$$
\epsilon = 96 G^2 R^2 \log\left(19 + 8\left(12 + \frac{1}{3R^2 G^2 \alpha^2}\right) n^{-\frac{4}{3}} T^{\frac{1}{3}}\right) T
$$

in Algorithm 1, the regret is upper-bounded by

$$
\begin{aligned}
\sum_{t=1}^T f_t(\mathbf{x}^t) - \min_{\mathbf{x}^* \in \mathcal{K}} \sum_{t=1}^T f_t(\mathbf{x}^*) \leq\, &9\beta R^2 T^{\frac{2}{3}} \log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right) + 2RGn^{\frac{1}{3}} T^{\frac{2}{3}} \\
&+ \left(36GR + \frac{4}{\alpha}\right) n^{\frac{2}{3}} T^{\frac{2}{3}} \log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right).
\end{aligned}
$$

2. *In continue to the previous item and under the same choice of parameters, for any $\rho \in [n]$, denoting $\Omega_\rho = \sum_{i=\rho+1}^n \lambda_i(\sum_{t=1}^T \nabla_t \nabla_t^\top)$ (∇_t is as defined in Algorithm 1), the regret is*

upper-bounded by

$$\sum_{t=1}^{T} f_t(\mathbf{x}^t) - \min_{\mathbf{x}^* \in \mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}^*) \leq 9\beta R^2 T^{\frac{2}{3}} \log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right) + 2RGn^{\frac{1}{3}} T^{\frac{2}{3}}$$
$$+ 36GR\rho^{\frac{1}{2}} n^{\frac{1}{6}} T^{\frac{2}{3}} \log\left(\left(c + \frac{c}{2G^2 \alpha^2}\right) T^{\frac{1}{3}}\right)$$
$$+ \frac{4}{\alpha} \rho n^{-\frac{1}{3}} T^{\frac{2}{3}} \log\left(cT^{\frac{1}{3}}\right)$$
$$+ 5RT^{\frac{1}{3}} \sqrt{\Omega_\rho} \sqrt{\log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right)}$$
$$+ \frac{\left(3GR + \frac{1}{2\alpha}\right)}{G^2 n^{\frac{2}{3}}} \Omega_\rho.$$

3. *Fix* $\rho \in [n]$. *Suppose* $T \geq T_0 = c \log^3\left(c + \left(c + \frac{c}{R^2 G^2 \alpha^2}\right) \rho^{-\frac{4}{3}} T^{\frac{1}{3}}\right)$, *where* c *is as in the previous items. Setting*

$$\eta = 8 \max\{6GR, \frac{1}{\alpha}\} \rho^{-\frac{1}{3}} T^{\frac{2}{3}}, \quad K = 4\rho^{-\frac{1}{3}} T^{\frac{2}{3}}, \quad \epsilon_I = 32G^2 T^{\frac{4}{3}},$$
$$\epsilon = 96G^2 R^2 \log\left(19 + 8\left(12 + \frac{1}{3R^2 G^2 \alpha^2}\right) \rho^{-\frac{4}{3}} T^{\frac{1}{3}}\right) T$$

in Algorithm 1, the regret is upper-bounded by

$$\sum_{t=1}^{T} f_t(\mathbf{x}^t) - \min_{\mathbf{x}^* \in \mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}^*) \leq 9\beta R^2 T^{\frac{2}{3}} \log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right) + 2RG\rho^{\frac{1}{3}} T^{\frac{2}{3}}$$
$$+ \left(36GR + \frac{4}{\alpha}\right) \rho^{\frac{2}{3}} T^{\frac{2}{3}} \log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right)$$
$$+ 5RT^{\frac{1}{3}} \sqrt{\Omega_\rho} \sqrt{\log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right)}$$
$$+ \frac{\left(3GR + \frac{1}{2\alpha}\right)}{\rho^{\frac{2}{3}} G^2} \Omega_\rho.$$

Note this bound is not explicitly dependent on the ambient dimension n.

In all cases, the overall number of calls to the LOO is upper-bounded by $0.65\left(8n^{\frac{1}{3}} T^{\frac{2}{3}} + T\right)$, *the additional space requirement is* $O(n^2)$, *and using the Sherman-Morrison formula for fast matrix inversion, the overall additional runtime is* $O(n^2(T + n^{1/3} T^{2/3}))$.

Proof The regret bound for each of the three cases is obtained directly by combining Lemma 9 with Lemma 12, and plugging-in the values of the parameters listed in the theorem. The bound on the overall number of calls follows also from the bound in Lemma 9 and plugging-in the values of the parameters listed in the theorem. The space requirement is dominated by the storage of $\mathbf{A}_m, \mathbf{A}_m^{-1}$ on each block m of Algorithm 1, and is thus upper-bounded by $O(n^2)$. Finally, in terms of additional runtime, it can be seen that the most expensive arithmetic operation preformed is the

multiplication of a $n \times n$ matrix ($\mathbf{A}_m$ or $\mathbf{A}_m^{-1}$ for some block m of Algorithm 1) with some vector (including when updating $\mathbf{A}_m^{-1}$ from $\mathbf{A}_{m-1}^{-1}$ via the Sherman-Morrison formula for rank-one update of the inverse), which requires $O(n^2)$ time. It can thus be seen that the overall additional runtime is dominated by the overall number of calls to the LOO (i.e., the overall number of iterations executed by Algorithm 2 throughout the run of Algorithm 1) times $O(n^2)$, which by plugging-in the values of the parameters in the theorem, gives the listed upper-bound on the overall additional runtime. ∎

Appendix C. Missing Details from Section 5

In this section we provide additional details on the sketching algorithm, and give the full version of Theorem 11 and its proof.

C.1. Properties of the sketching algorithm

The following observation shows that Algorithm 4 produces matrices $\mathbf{A}_m$, $m = 0, \ldots, T/K$, that indeed satisfy the requirements of Algorithm 1.

Observation 1 *Fix iteration $m \geq 1$ of Algorithm 4. It holds that $\mathbf{A}_0 \preceq \mathbf{A}_m \preceq \mathbf{A}_{m-1} + \bar{\nabla}_m \bar{\nabla}_m^\top$.*

Proof The first inequality $\mathbf{A}_m \succeq \mathbf{A}_0$ holds trivially due to the definition of $\mathbf{A}_m$ in the algorithm. We thus focus on the proof of the second inequality. Let $\mathbf{S}_{m-1}$ be as at the beginning of the m-th iteration of the for loop, and let $\mathbf{S}_{m-1}^+$ denote its value after setting its last row to $\bar{\nabla}_m$ (instead of $\mathbf{0}$). It holds that $\mathbf{S}_{m-1}^{+\top}\mathbf{S}_{m-1}^+ = \mathbf{S}_{m-1}^\top\mathbf{S}_{m-1} + \bar{\nabla}_m \bar{\nabla}_m^\top$. By the definition of $\mathbf{S}_m$ in the algorithm we have that,

$$\mathbf{S}_m^\top\mathbf{S}_m = \mathbf{S}_{m-1}^{+\top}\mathbf{S}_{m-1}^+ - \sigma_m\mathbf{V}_m^\top\mathbf{V}_m = \mathbf{S}_{m-1}^\top\mathbf{S}_{m-1} + \bar{\nabla}_m\bar{\nabla}_m^\top - \sigma_m\mathbf{V}_m^\top\mathbf{V}_m.$$

Since $\mathbf{A}_{m-1} = \epsilon_I\mathbf{I}_n + \mathbf{S}_{m-1}^\top\mathbf{S}_{m-1}$ and $\mathbf{A}_m = \epsilon_I\mathbf{I}_n + \mathbf{S}_m^\top\mathbf{S}_m$, it indeed holds that

$$\mathbf{A}_m - \mathbf{A}_{m-1} = \mathbf{S}_m^\top\mathbf{S}_m - \mathbf{S}_{m-1}^\top\mathbf{S}_{m-1} = \bar{\nabla}_m\bar{\nabla}_m^\top - \sigma_m\mathbf{V}_m^\top\mathbf{V}_m \tag{11}$$
$$\preceq \bar{\nabla}_m\bar{\nabla}_m^\top.$$

∎

We now state several results regarding Algorithm 4 which will be required in order to prove Theorem 11.

Theorem 14 (Theorem 1.1 in Ghashami et al. (2016)) *Consider Algorithm 4 and let $\mathbf{B} \in \mathbb{R}^{(T/K) \times n}$ be the matrix which is received by the algorithm row by row (i.e., the ith row of $\mathbf{B}$ is $\bar{\nabla}_i$). It holds that*

$$\mathbf{0} \preceq \mathbf{B}^\top\mathbf{B} - \mathbf{S}_{T/K}^\top\mathbf{S}_{T/K} \preceq \mathbf{I}_n \sum_{i=\rho+1}^n \lambda_i\left(\mathbf{B}^\top\mathbf{B}\right).$$

Lemma 15 *Consider Algorithm 4 and let $\mathbf{B} \in \mathbb{R}^{(T/K)\times n}$ be the matrix which is received by the algorithm row by row (i.e., the ith row of $\mathbf{B}$ is $\bar{\nabla}_i$). Denote $\mathbf{B}_\rho$ its best rank-ρ approximation, i.e. $\mathbf{B}_\rho = \operatorname{argmin}_{\mathbf{C}:rank(\mathbf{C})\leq\rho} \|\mathbf{B} - \mathbf{C}\|_F$. It holds that*

$$\sum_{m=1}^{T/K} \sigma_m \leq \|\mathbf{B} - \mathbf{B}_\rho\|_F^2 \, ,$$

where σ_m is as defined in Algorithm 4.

Proof Let $\mathbf{z}_i$, $i = 1,\ldots,n$ denote the left singular vector of $\mathbf{B}$ corresponding to the singular value $\sigma_i(\mathbf{B})$. We denote $\mathbf{S} = \mathbf{S}_{T/K}$, where $\mathbf{S}_{T/K}$ is as defined in Algorithm 4, i.e., the last sketched matrix. Using Property 3 in Ghashami et al. (2016) we have that,

$$(\rho + 1) \sum_{m=1}^{T/K} \sigma_m \leq \|\mathbf{B}\|_F^2 - \|\mathbf{S}\|_F^2 = \sum_{i=1}^{n} \|\mathbf{B}\mathbf{z}_i\|^2 - \|\mathbf{S}\|_F^2 = \sum_{i=1}^{\rho} \|\mathbf{B}\mathbf{z}_i\|^2 + \sum_{i=\rho+1}^{n} \|\mathbf{B}\mathbf{z}_i\|^2 - \|\mathbf{S}\|_F^2 \, .$$

Since $\mathbf{B}_\rho$ denotes the best rank-ρ approximation of $\mathbf{B}$, the above inequality implies that,

$$(\rho + 1) \sum_{m=1}^{T/K} \sigma_m \leq \|\mathbf{B} - \mathbf{B}_\rho\|_F^2 + \sum_{i=1}^{\rho} \|\mathbf{B}\mathbf{z}_i\|^2 - \|\mathbf{S}\|_F^2 \, . \tag{12}$$

Note that

$$\sum_{i=1}^{\rho} \|\mathbf{S}\mathbf{z}_i\|^2 = \sum_{i=1}^{\rho} \mathbf{z}_i^\top \mathbf{S}^\top \mathbf{S}\mathbf{z}_i \leq \operatorname{Tr}(\mathbf{S}^\top \mathbf{S}) = \|\mathbf{S}\|_F^2.$$

Thus, using Property 2 in Ghashami et al. (2016) we have that,

$$\sum_{i=1}^{\rho} \|\mathbf{B}\mathbf{z}_i\|^2 - \|\mathbf{S}\|_F^2 \leq \sum_{i=1}^{\rho} \left(\|\mathbf{B}\mathbf{z}_i\|^2 - \|\mathbf{S}\mathbf{z}_i\|^2 \right) \leq \rho \sum_{m=1}^{T/K} \sigma_m \tag{13}$$

The lemma follows from plugging Eq. (13) into Eq. (12). ∎

Lemma 16 *Consider the run of Algorithm 4 with a sketch size ρ, and denote $\Omega_\rho = \sum_{i=\rho+1}^{n} \lambda_i \left(\sum_{t=1}^{T} \nabla_t \nabla_t^\top \right)$, where ∇_t is as defined in Algorithm 1. It holds that,*

$$\sum_{m=1}^{T/K} \left\| \bar{\nabla}_m \right\|_{\mathbf{A}_m^{-1}}^2 \leq \rho \log \left(1 + \frac{G^2 KT}{\epsilon_I} \right) + \frac{(\rho+1)K\Omega_\rho}{\epsilon_I}.$$

Proof Using Eq. (11) it holds that,

$$\sum_{m=1}^{T/K} \left\| \bar{\nabla}_m \right\|_{\mathbf{A}_m^{-1}}^2 = \sum_{m=1}^{T/K} \mathbf{A}_m^{-1} \bullet \bar{\nabla}_m \bar{\nabla}_m^\top = \sum_{m=1}^{T/K} \mathbf{A}_m^{-1} \bullet (\mathbf{A}_m - \mathbf{A}_{m-1}) + \sum_{m=1}^{T/K} \sigma_m \mathbf{A}_m^{-1} \bullet \mathbf{V}_m^\top \mathbf{V}_m.$$

Since $\lambda_1\left(\mathbf{A}_m^{-1}\right) \leq \epsilon_I^{-1}$ and $\operatorname{Tr}\left(\mathbf{V}_m^\top \mathbf{V}_m\right) = \rho + 1$, it holds that $\mathbf{A}_m^{-1} \bullet \mathbf{V}_m^\top \mathbf{V}_m \leq \epsilon_I^{-1}(\rho + 1)$ for every $m \in [T/K]$. Since $\mathbf{A}_m \succ 0$ for every m, using Lemma 20 we have that,

$$
\sum_{m=1}^{T/K} \left\|\bar{\nabla}_m\right\|_{\mathbf{A}_m^{-1}}^2 \leq \sum_{m=1}^{T/K} \log\left(\frac{|\mathbf{A}_m|}{|\mathbf{A}_{m-1}|}\right) + \frac{(\rho + 1)}{\epsilon_I} \sum_{m=1}^{T/K} \sigma_m
$$

$$
= \log\left(\frac{|\mathbf{A}_{T/K}|}{|\mathbf{A}_0|}\right) + \frac{(\rho + 1)}{\epsilon_I} \sum_{m=1}^{T/K} \sigma_m. \tag{14}
$$

Since $\operatorname{rank}(\mathbf{S}_{T/K}) \leq \rho$ (recall the last row is $\mathbf{0}$), we have that

$$
|\mathbf{A}_{T/K}| = \prod_{i=1}^n \lambda_i\left(\epsilon_I \mathbf{I}_n + \mathbf{S}_{T/K}^\top \mathbf{S}_{T/K}\right) = \epsilon_I^{(n-\rho)} \prod_{i=1}^\rho \left(\epsilon_I + \lambda_i\left(\mathbf{S}_{T/K}^\top \mathbf{S}_{T/K}\right)\right).
$$

Since $\mathbf{A}_0 = \epsilon_I \mathbf{I}_n$ and $\mathbf{S}_{T/K}^\top \mathbf{S}_{T/K} \preceq \sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top$ (Theorem 14), we have that

$$
\log\left(\frac{|\mathbf{A}_{T/K}|}{|\mathbf{A}_0|}\right) \leq \sum_{i=1}^\rho \log\left(1 + \frac{\lambda_i\left(\sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top\right)}{\epsilon_I}\right) \leq \rho \log\left(1 + \frac{G^2 KT}{\epsilon_I}\right), \tag{15}
$$

where the last inequality is since $\lambda_i\left(\sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top\right) \leq \sum_{m=1}^{T/K} \left\|\bar{\nabla}_m\right\|^2 \leq (T/K)(KG)^2 = G^2 KT$.

From Lemma 15 we have that $\sum_{m=1}^{T/K} \sigma_m \leq \sum_{i=\rho+1}^n \lambda_i\left(\sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top\right)$. Since $\bar{\nabla}_m = \sum_{t=(m-1)K+1}^{mK} \nabla_t$, using Lemma 18 it holds that $K \sum_{t=1}^T \nabla_t \nabla_t^\top \succeq \sum_{m=1}^{T/K} \bar{\nabla}_m \bar{\nabla}_m^\top$. Plugging these observations and Eq. (15) into Eq.(14), we indeed obtain

$$
\sum_{m=1}^{T/K} \left\|\bar{\nabla}_m\right\|_{\mathbf{A}_m^{-1}}^2 \leq \rho \log\left(1 + \frac{G^2 KT}{\epsilon_I}\right) + \frac{(\rho + 1)K}{\epsilon_I} \sum_{i=\rho+1}^n \lambda_i\left(\sum_{t=1}^T \nabla_t \nabla_t^\top\right).
$$

∎

C.2. Full version of Theorem 11 and its proof

Theorem 17 *[Full version of Theorem 11] Consider the implementation of Algorithm 1 as described in Lemma 9 and when using the update rule described in Algorithm 4:* $\mathbf{A}_m = \mathbf{A}_0 + \mathbf{S}_m^\top \mathbf{S}_m$ *for every block m. Fix $\rho \in [n]$ and denote $\Omega_\rho = \sum_{i=\rho+1}^n \lambda_i\left(\sum_{t=1}^T \nabla_t \nabla_t^\top\right)$. Suppose $T \geq T_0 = c \log^3\left(c + \left(c + \frac{c}{R^2 G^2 \alpha}\right) \rho^{-\frac{4}{3}} T^{\frac{1}{3}}\right)$, where $c > 0$ is a certain universal constant. Setting*

$$
\eta = 8 \max\{6GR, \frac{1}{\alpha}\} \rho^{-\frac{1}{3}} T^{\frac{2}{3}}, \quad K = 4\rho^{-\frac{1}{3}} T^{\frac{2}{3}}, \quad \epsilon_I = 32 G^2 T^{\frac{4}{3}},
$$

$$
\epsilon = 96 G^2 R^2 \log\left(19 + 8\left(12 + \frac{1}{3R^2 G^2 \alpha^2}\right) \rho^{-\frac{4}{3}} T^{\frac{1}{3}}\right) T
$$

in Algorithm 1, the regret is upper bounded by

$$\sum_{t=1}^{T} f_t(\mathbf{x}^t) - \min_{\mathbf{x}^*\in\mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}^*) \leq 9\beta R^2 T^{\frac{2}{3}} \log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right) + 2RG\rho^{\frac{1}{3}} T^{\frac{2}{3}}$$

$$+ \left(36GR + \frac{4}{\alpha}\right)(\rho+1)^{\frac{2}{3}} T^{\frac{2}{3}} \log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right)$$

$$+ 5R\rho^{\frac{1}{2}} T^{\frac{1}{3}} \sqrt{\Omega_\rho}\sqrt{\log\left(\left(c + \frac{c}{R^2 G^2 \alpha^2}\right) T^{\frac{1}{3}}\right)}$$

$$+ \left(6GR + \frac{1}{\alpha}\right)\frac{\rho^{\frac{1}{3}}\Omega_\rho}{G^2}.$$

The overall number of calls to the LOO is upper bounded by $0.65\left(8\rho^{\frac{1}{3}}T^{\frac{2}{3}} + T\right)$*, the additional space requirement is* $O(\rho n)$*, and the overall additional runtime is* $O(\rho nT + \rho^{4/3}nT^{2/3} + \rho^{7/3}nT^{1/3})$*.*

Proof The upper-bound on the regret and number of calls to the LOO follows directly from combining Lemma 9, Lemma 16, and plugging-in the values for parameters listed in the theorem. In terms of space requirement, by only explicitly maintaining the $(\rho+1)\times n$ matrices $\mathbf{S}_m, \mathbf{V}_m$ and the diagonal matrices $\mathbf{H}_m, \Sigma_m, \widehat{\Sigma}_m$ on each block m of Algorithm 1, i.e., $\mathbf{A}_m, \mathbf{A}_m^{-1}$ are never computed explicitly, the space is upper-bounded by $O(\rho n)$. Finally, for upper-bounding the overall additional runtime we note that the two most expensive arithmetic operations are i. computing a matrix-vector product with either $\mathbf{A}_m$ or $\mathbf{A}_m^{-1}$ during some block m of Algorithm 1, and ii. computing the eigen-decomposition of $\mathbf{S}_{m-1}^\top \mathbf{S}_{m-1} \in \mathbb{R}^{n\times n}$ for some iteration m of Algorithm 4. Using the low-rank factorizations of $\mathbf{A}_m, \mathbf{A}_m^{-1}$ in Algorithm 4 (i.e., by only explicitly maintaining $\mathbf{S}_m \in \mathbb{R}^{(\rho+1)\times n}$ and the diagonal matrix $\mathbf{H}_m$), computing a matrix-vector product with either $\mathbf{A}_m$ or $\mathbf{A}_m^{-1}$, could be carried out in $O(\rho n)$ time. Computing the eigen-decomposition of each $\mathbf{S}_{m-1}^\top \mathbf{S}_{m-1}$, could be done in $O(\rho^2 n)$ time by computing the SVD of $\mathbf{S}_{m-1} \in \mathbb{R}^{(\rho+1)\times n}$. Note however that such an SVD is computed only once during each block m of Algorithm 1. Thus, the overall runtime associated with these SVD computations is $O((T/K)\rho^2 n)$, which by plugging the value of K in the theorem, is only $O(\rho^{7/3}nT^{1/3})$. The overall additional runtime, excluding these SVD computations, is thus dominated by the number of matrix-vector products, times the runtime required for each such product which, as discussed, is $O(\rho n)$. As discussed in the proof of Theorem 13, the number of matrix-vector products is dominated by the overall number of calls to the LOO (i.e., the overall number of iterations executed by Algorithm 2). Combining these two contributions (matrix-vector products and SVD computations) to the overall additional runtime, yields the bound listed in the theorem. ∎

Appendix D. Discussion of Assumption 1

We recall that while the feasible set is $\mathcal{K}$, Assumption 1 assumes the losses are defined and satisfy the various properties listed in the (potentially much) larger set $3R\mathcal{B}$, where R is such that $\mathcal{K} \subseteq R\mathcal{B}$. This is because our Algorithm 1 queries gradients at infeasible points w.r.t. $\mathcal{K}$, and thus we must make sure these assumptions hold in these infeasible points, and for ease of presentation we simply

make sure in our analysis that indeed all infeasible points $\widetilde{\mathbf{y}}_m$ in the instantiations of Algorithm 1, lie inside the ball $3R\mathcal{B}$.

First, we note that our consideration of enclosing balls centered at the origin is w.l.o.g. since one can apply translation. Second, with a slightly more involved analysis it suffices to require that Assumption 1 holds only in the set $\widehat{\mathcal{K}}_\delta := \{\mathbf{x} \in \mathbb{R}^n \mid \mathrm{dist}(\mathbf{x}, \mathcal{K}) \leq \delta\}$, for some small $\delta > 0$, as we now explain. Note in particular that Lemma 8 guarantees that our LOO-based implementation of the AFP oracle (Algorithm 3) returns an infeasible point $\mathbf{y}$ and a corresponding feasible point $\mathbf{x} \in \mathcal{K}$, such that $\|\mathbf{x} - \mathbf{y}\|_{\mathbf{A}}^2 \leq 3\epsilon$. This implies that $\mathrm{dist}(\mathbf{y}, \mathcal{K}) \leq \|\mathbf{x} - \mathbf{y}\| \leq \sqrt{\frac{3\epsilon}{\lambda_n(\mathbf{A})}}$. Thus, when used with our Algorithm 1 (as described in Lemma 9), for every block m we have that,

$$\mathrm{dist}(\widetilde{\mathbf{y}}_m, \mathcal{K}) \leq \sqrt{\frac{3\epsilon}{\lambda_n(\mathbf{A}_{m-1})}} \leq \sqrt{\frac{3\epsilon}{\lambda_n(\mathbf{A}_0)}} = \sqrt{\frac{3\epsilon}{\epsilon_I}} = \widetilde{O}(T^{-1/6}),$$

where the second inequality is due to the constraints on the matrices $\{\mathbf{A}_m\}_{m\geq 1}$ in Algorithm 1, and the last equality follows from plugging-in the values of ϵ, ϵ_I listed in our main theorems — Theorem 13 and Theorem 11.

Thus, already for $\delta = \widetilde{O}(T^{-1/6})$, all the points in which our algorithm queries gradients lie in $\widehat{\mathcal{K}}_\delta$, and it suffices to require that the assumptions listed in Assumption 1 hold only in this set, which becomes tighter around $\mathcal{K}$ as T increases.

Appendix E. Auxiliary Lemmas

Lemma 18 *Let $\{\mathbf{v}_i\}_{i=1}^k \subset \mathbb{R}^n$, and $\mathbf{u} = \sum_{i=1}^k \mathbf{v}_i$. Then, $k \sum_{i=1}^k \mathbf{v}_i \mathbf{v}_i^\top \succeq \mathbf{u}\mathbf{u}^\top$.*

Proof From Jensen's inequality we have that for every sequence of scalars $\{a_i\}_{i=1}^k \subset \mathbb{R}$, it holds that $k \sum_{i=1}^k a_i^2 \geq \left(\sum_{i=1}^k a_i\right)^2$. Thus, for every $\mathbf{z} \in \mathbb{R}^n$ we have that,

$$\mathbf{z}^\top \left(\sum_{i=1}^k \mathbf{v}_i \mathbf{v}_i^\top\right) \mathbf{z} = \sum_{i=1}^k \left(\mathbf{v}_i^\top \mathbf{z}\right)^2 \geq \frac{1}{k}\left(\sum_{i=1}^k \mathbf{v}_i^\top \mathbf{z}\right)^2 = \frac{1}{k}\left(\mathbf{u}^\top \mathbf{z}\right)^2 = \frac{1}{k}\mathbf{z}^\top \mathbf{u}\mathbf{u}^\top \mathbf{z}.$$

$\blacksquare$

Lemma 19 *Let $\mathcal{C} \subset \mathbb{R}^n$ be convex and compact and such that $\mathcal{C} \subseteq R\mathcal{B}$. Let $f_1, \ldots, f_k$ be functions $\mathcal{C} \to \mathbb{R}$ that are differentiable over $\mathcal{C}$ and have gradients upper-bounded in ℓ_2 norm by some $G > 0$ over $\mathcal{C}$, and satisfy the curvature condition (Definition 3) over $\mathcal{C}$ with some parameter $\alpha > 0$. Define $h(\mathbf{x}) = \sum_{i=1}^k f_i(\mathbf{x})$. For all $\eta \geq \max\{4kGR, 2k/\alpha\}$ and every $\mathbf{x}, \mathbf{y} \in \mathcal{C}$ it holds that,*

$$h(\mathbf{x}) - h(\mathbf{y}) \leq \nabla h(\mathbf{x})^\top (\mathbf{x} - \mathbf{y}) - \frac{1}{2\eta}(\mathbf{y} - \mathbf{x})^\top \nabla h(\mathbf{x}) \nabla h(\mathbf{x})^\top (\mathbf{y} - \mathbf{x})$$

Proof Let $\eta' = \max\{4GR, 2/\alpha\}$ and fix some $\mathbf{x}, \mathbf{y} \in \mathcal{C}$. It holds that

$$
\begin{aligned}
h(\mathbf{x}) - h(\mathbf{y}) &= \sum_{i=1}^{k} f_i(\mathbf{x}) - f_i(\mathbf{y}) \\
&\leq \sum_{i=1}^{k} \left(\nabla f_i(\mathbf{x})^\top (\mathbf{x} - \mathbf{y}) \right) - \frac{1}{2\eta'} \sum_{i=1}^{k} \left((\mathbf{y} - \mathbf{x})^\top \nabla f_i(\mathbf{x}) \nabla f_i(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) \right) \\
&= \nabla h(\mathbf{x})^\top (\mathbf{x} - \mathbf{y}) - \frac{1}{2\eta'} (\mathbf{y} - \mathbf{x})^\top \left(\sum_{i=1}^{k} \nabla f_i(\mathbf{x}) \nabla f_i(\mathbf{x})^\top \right) (\mathbf{y} - \mathbf{x}).
\end{aligned}
$$

Using Lemma 18 we have that $k \sum_{i=1}^{k} \nabla f_i(\mathbf{x}) \nabla f_i(\mathbf{x})^\top \succeq \left(\sum_{i=1}^{k} \nabla f_i(\mathbf{x}) \right) \left(\sum_{i=1}^{k} \nabla f_i(\mathbf{x}) \right)^\top$ and thus,

$$
\begin{aligned}
h(\mathbf{x}) - h(\mathbf{y}) &\leq \nabla h(\mathbf{x})^\top (\mathbf{x} - \mathbf{y}) - \frac{1}{2\eta' k} (\mathbf{y} - \mathbf{x})^\top \left(\left(\sum_{i=1}^{k} \nabla f_i(\mathbf{x}) \right) \left(\sum_{i=1}^{k} \nabla f_i(\mathbf{x}) \right)^\top \right) (\mathbf{y} - \mathbf{x}) \\
&\leq \nabla h(\mathbf{x})^\top (\mathbf{x} - \mathbf{y}) - \frac{1}{2\eta} (\mathbf{y} - \mathbf{x})^\top \nabla h(\mathbf{x}) \nabla h(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}),
\end{aligned}
$$

where the last inequality holds since $\eta \geq k\eta'$. ■

Lemma 20 *Let $\mathbf{A}, \mathbf{B} \in \mathbb{S}^n$ be positive definite matrices. Then, $\mathbf{A}^{-1} \bullet (\mathbf{A} - \mathbf{B}) \leq \ln \frac{|\mathbf{A}|}{|\mathbf{B}|}$.*

Proof It holds that,

$$
\mathbf{A}^{-1} \bullet (\mathbf{A} - \mathbf{B}) = \mathrm{Tr}\left(\mathbf{A}^{-\frac{1}{2}} (\mathbf{A} - \mathbf{B}) \mathbf{A}^{-\frac{1}{2}} \right) = \sum_{i=1}^{n} \left(\lambda_i \left(\mathbf{I}_n - \mathbf{A}^{-\frac{1}{2}} \mathbf{B} \mathbf{A}^{-\frac{1}{2}} \right) \right).
$$

Since $\lambda_i (\mathbf{I}_n - \mathbf{A}) = 1 - \lambda_{n-i+1}(\mathbf{A})$, $1 - x \leq -\ln x$ for every $x \in \mathbb{R}_+$, and $\mathbf{A}^{-\frac{1}{2}} \mathbf{B} \mathbf{A}^{-\frac{1}{2}} \succ \mathbf{0}$, it holds that

$$
\begin{aligned}
\mathbf{A}^{-1} \bullet (\mathbf{A} - \mathbf{B}) &= \sum_{i=1}^{n} \left(1 - \lambda_i \left(\mathbf{A}^{-\frac{1}{2}} \mathbf{B} \mathbf{A}^{-\frac{1}{2}} \right) \right) \leq - \sum_{i=1}^{n} \ln \left(\lambda_i \left(\mathbf{A}^{-\frac{1}{2}} \mathbf{B} \mathbf{A}^{-\frac{1}{2}} \right) \right) \\
&= -\ln \left(\prod_{i=1}^{n} \lambda_i \left(\mathbf{A}^{-\frac{1}{2}} \mathbf{B} \mathbf{A}^{-\frac{1}{2}} \right) \right) = -\ln |\mathbf{A}^{-\frac{1}{2}} \mathbf{B} \mathbf{A}^{-\frac{1}{2}}| \\
&= \ln \left(|\mathbf{A}| / |\mathbf{B}| \right).
\end{aligned}
$$

■