Projection Efficient Subgradient Method and Optimal Nonsmooth Frank-Wolfe Method

Kiran Koshy Thekumparampil
University of Illinois at Urbana-Champaign
`thekump2@illinois.edu`

Prateek Jain
Microsoft Research, India
`prajain@microsoft.com`

Praneeth Netrapalli
Microsoft Research, India
`praneeth@microsoft.com`

Sewoong Oh
University of Washington, Seattle
`sewoong@cs.washington.edu`

Abstract

We consider the classical setting of optimizing a nonsmooth Lipschitz continuous convex function over a convex constraint set, when having access to a (stochastic) first-order oracle (FO) for the function and a projection oracle (PO) for the constraint set. It is well known that to achieve ε-suboptimality in high-dimensions, $\Theta(\varepsilon^{-2})$ FO calls are necessary [64]. This is achieved by the projected subgradient method (PGD) [11]. However, PGD also entails $\mathcal{O}(\varepsilon^{-2})$ PO calls, which may be computationally costlier than FO calls (e.g. nuclear norm constraints). Improving this PO calls complexity of PGD is largely unexplored, despite the fundamental nature of this problem and extensive literature. We present first such improvement. This only requires a mild assumption that the objective function, when extended to a slightly larger neighborhood of the constraint set, still remains Lipschitz and accessible via FO. In particular, we introduce MOPES method, which carefully combines Moreau-Yosida smoothing and accelerated first-order schemes. This is guaranteed to find a *feasible* ε-suboptimal solution using only $\mathcal{O}(\varepsilon^{-1})$ PO calls and optimal $\mathcal{O}(\varepsilon^{-2})$ FO calls. Further, instead of a PO if we only have a linear minimization oracle (LMO, à la Frank-Wolfe) to access the constraint set, an extension of our method, MOLES, finds a *feasible* ε-suboptimal solution using $\mathcal{O}(\varepsilon^{-2})$ LMO calls and FO calls—both match known lower bounds [54], resolving a question left open since [84]. Our experiments confirm that these methods achieve significant speedups over the state-of-the-art, for a problem with costly PO and LMO calls.

1 Introduction

In this paper, we consider the nonsmooth convex optimization (NSCO) problem with the First-order Oracle (FO) and the Projection Oracle (PO) defined as:

$$\text{NSCO}: \min_{x}\ f(x),\ \text{s.t.}\ x \in \mathcal{X}\ ,\ \ \text{FO}(x) \in \partial f(x),\ \text{and}\ \text{PO}(x) = \mathcal{P}_{\mathcal{X}}(x) = \operatorname*{argmin}_{y \in \mathcal{X}} \|y - x\|_2^2, \quad (1)$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a convex Lipschitz-continuous function, and $\mathcal{X} \subseteq \mathbb{R}^d$ is a convex constraint. When queried at a point x, FO returns a subgradient of f at x and PO returns the projection of x onto $\mathcal{X}$. NSCO is a fundamental problem with a long history and several important applications including support vector machines (SVM) [12], robust learning [44], and utility maximization in finance [82].

Finding an ε-suboptimal solution for this problem requires $\Omega(\varepsilon^{-2})$ FO calls in the worst case, when the dimension d is large [64]. This lower bound is tightly matched by the projected subgradient method (PGD). Unfortunately, PGD also uses one PO call after every FO call, resulting in a PO calls

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

	Randomized Smoothing dimension dependent	**State-of-the-art dimension-free**	**Our results** (Theorems 1 and 2)	**Lower bound**
SFO	$\mathcal{O}((G^2+\sigma^2)/\varepsilon^2)$ [27]	$\mathcal{O}((G^2+\sigma^2)/\varepsilon^2)$ [65]	$\boldsymbol{\mathcal{O}((G^2+\sigma^2)/\varepsilon^2)}$	$\Omega((G^2+\sigma^2)/\varepsilon^2)$ [64]
PO	$\mathcal{O}(d^{1/4}G/\varepsilon)$ [27]	$\mathcal{O}(G^2/\varepsilon^2)^\star$ [65]	$\boldsymbol{\mathcal{O}(G/\varepsilon)}$	Open problem
SFO	$\mathcal{O}(\sqrt{d}\,(G^2+\sigma^2)^2/\varepsilon^4)$ [54]	$\mathcal{O}((G^2+\sigma^2)/\varepsilon^2)^\dagger$	$\boldsymbol{\mathcal{O}((G^2+\sigma^2)/\varepsilon^2)}$	$\Omega((G^2+\sigma^2)/\varepsilon^2)$ [64]
LMO	$\mathcal{O}(\sqrt{d}\,G^2/\varepsilon^2)$ [54]*	$\mathcal{O}((G^2+\sigma^2)^2/\varepsilon^4)^\dagger$	$\boldsymbol{\mathcal{O}(G^2/\varepsilon^2)}$	$\Omega(G^2/\varepsilon^2)$ [54]

Table 1: Comparison of SFO (3), PO (1) & LMO (2) calls complexities of our methods and state-of-the-art algorithms, and corresponding lower-bounds for finding an approximate minimizer of a d-dimensional NSCO problem (1). We assume that f is convex and G-Lipschitz continuous, and is accessed through a stochastic subgradient oracle with a variance of σ^2. *requires using a minibatch of appropriate size, †approximates projections of PGD with FW method (FW-PGD, see Appendix B.2).

complexity (PO-CC)—the number of times PO needs to be invoked—of $\Theta(\varepsilon^{-2})$. This can be a major bottleneck in solving several practical problems like collaborative filtering [79], where the cost of a PO is often higher than the cost of an FO call. This begs the natural question, which surprisingly is largely unexplored in the general nonsmooth optimization setting: *Can we design an algorithm whose PO calls complexity is significantly better than the optimal FO calls complexity $O(\varepsilon^{-2})$?*

In this work, we answer the above question in the affirmative. Our first key contribution is MOreau Projection Efficient Subgradient method (MOPES), that obtains an ε-suboptimal solution using only $\mathcal{O}(\varepsilon^{-1})$ PO calls, while still ensuring that the FO calls complexity (FO-CC)—the number of times FO needs to be invoked—is optimal, i.e., $\mathcal{O}(\varepsilon^{-2})$. This requires a mild assumption that the function f extends to a slightly larger neighborhood of the constraint set $\mathcal{X}$. Concretely, we assume that f is Lipschitz continuous in this neighborhood and FO can be queried at points in this neighborhood. To the best of our knowledge, our result is the first improvement over the $\mathcal{O}(\varepsilon^{-2})$ PO calls of PGD for minimizing a general nonsmooth Lipschitz continuous convex function.

We achieve this by carefully combining Moreau-Yosida regularization with accelerated first-order methods [62, 81]. As accelerated methods cannot be directly applied to a nonsmooth f, we can instead apply them to minimize its Moreau envelope, which is smooth (as long as f is Lipschitz continuous). Although this idea has been explored, for example, in [25, 9], PO-CC has remained $\mathcal{O}(\varepsilon^{-2})$, unless a much stronger and unrealistic oracle is assumed [9] with a direct access to the gradient of Moreau envelope. The key idea in breaking this barrier is to separate out the dependence on FO calls of f from PO calls to $\mathcal{X}$ by: (a) using Moreau-Yosida regularization to *split* the original problem into a composite problem, where one component consists of an unconstrained optimization of the function f and the other consists of a simple constrained optimization over the set $\mathcal{X}$; and (b) applying the gradient sliding algorithm [55] on this joint problem to ensure the above mentioned bounds for both FO and PO calls. We note that our results are limited to the Euclidean norm, since our results crucially depend on smoothness of the Moreau envelope and its regularizer, which is not known for Moreau envelopes based on general Bregman divergences [7].

In some high-dimensional problems, even a single call to the PO can be computationally prohibitive. A popular alternative, pioneered by Frank and Wolfe [28], is to replace PO by a more efficient Linear Minimization Oracle (LMO), which returns a minimizer of any linear functional $\langle g, \cdot\rangle$ over the set $\mathcal{X}$.

$$\mathrm{LMO}\,(g) \in \operatorname*{argmin}_{s\in\mathcal{X}} \langle g, s\rangle \tag{2}$$

Linear minimization is much faster than projection in several practical ML applications such as a nuclear norm ball constrained problems [15], video-narration alignment [1], structured SVM [51], and multiple sequence alignment and motif discovery [89]. LMO based methods have an important additional benefit of producing solutions that preserve desired structures such as sparsity and low rank. For *smooth* f, there is a long history of conditional gradient (Frank-Wolfe) methods that use $\mathcal{O}(\varepsilon^{-1})$ LMO calls and $\mathcal{O}(\varepsilon^{-1})$ FO calls to achieve ε-suboptimality, which achieve optimal LMO-CC [45]. For *nonsmooth* functions, starting from the work of [84], several approaches have been proposed, some under more assumptions. The best known upper bound on LMO calls is $\mathcal{O}(\sqrt{d}\varepsilon^{-2})$ which is achieved at the expense of significantly larger $\mathcal{O}(\varepsilon^{-4})$ FO calls. Details of these are in Section 1.1.

Our second key contribution is the algorithm MOLES, which obtains an ε-suboptimal solution using the optimal $\mathcal{O}(\varepsilon^{-2})$ LMO and FO calls, without any additional dimension dependence. We achieve this result by extending MOPES to work with approximate projections and using the classical Frank-Wolfe (FW) method [28] to implement these approximate projections using LMO calls.

Finally, both of our methods extend naturally to the Stochastic First-order Oracle (SFO) setting, where we have access only to stochastic versions of the function's subgradients. Stochastic versions of MOPES and MOLES still achieve the the same PO/LMO calls complexities as deterministic counterparts, while the SFO calls complexity (SFO-CC) is $\mathcal{O}\left((1+\sigma^2)\epsilon^{-2}\right)$, where σ^2 is the variance in SFO. This again matches information theoretic lower bounds [64].

Contributions: We summarize our contributions below and in Table 1. We assume that the function f extends to a slightly larger neighborhood of the constraint set $\mathcal{X}$ i.e., f continues to be Lipschitz continuous and (S)FO can be queried in this neighborhood.

- We introduce MOPES and show that it is guaranteed to find an ε-suboptimal solution for any constrained nonsmooth convex optimization problem using $\mathcal{O}(\varepsilon^{-1})$ PO calls and optimal $\mathcal{O}(\varepsilon^{-2})$ SFO calls. To the best of our knowledge, for the general problem, this achieves the first improvement over $\mathcal{O}(\varepsilon^{-2})$ PO-CC and SFO-CC of stochastic projected subgradient method (PGD).
- For LMO setting, we extend our method to design MOLES, that achieves the optimal SFO-CC and LMO-CC of $\mathcal{O}(\varepsilon^{-2})$, and improves over the best known LMO-CC by $\sqrt{d}$.
- We also empirically evaluate MOPES and MOLES on the popular nuclear norm constrained Matrix SVM problem [85], where they achieve significant speedups over their corresponding baselines.
- Our main technical novelty is the use Moreau-Yosida regularization to separate out the constraint (PO/LMO) and function (SFO) accesses into two parts of a composite optimization problem. This enables a better control of how many times each of these oracles are accessed. This idea might be of independent interest, whenever a trade-off between PO-CC/LMO-CC and SFO-CC is desirable.

1.1 Related Work

Nonsmooth convex optimization: Nonsmooth convex optimization has been the focal point of several research works for past few decades. [64] provided information theoretic lower bound of FO calls $O(\varepsilon^{-2})$ to obtain ε-suboptimal solution, for the general problem. This bound is matched by the PGD method introduced independently by [34] and [59], which also implies a PO-CC of $O(\varepsilon^{-2})$. Recently, several faster PGD style methods [50, 78, 87, 48] have been proposed that exploit more structure in the given optimization function, e.g., when the function is a sum of a smooth and a nonsmooth function for which a *proximal* operator is available [8]. But, to the best of our knowledge, such works do not explicitly address PO-CC and are mainly concerned about optimizing FO-CC. Thus, for the worst case nonsmooth functions, these methods still suffer from $O(\varepsilon^{-2})$ PO-CC.

Smoothed surrogates: Smoothing of the nonsmooth function is another common approach in solving them [62, 66]. In particular, randomized smoothing [27, 9] techniques have been successful in bringing down FO-CC w.r.t. ε but such improvements come at the cost of dimension factors. For example, [27, Corollary 2.4] provides a randomized smoothing method that has $O(d^{1/4}/\varepsilon)$ PO-CC and $O(\varepsilon^{-2})$ FO-CC. Our MOPES method guarantees significantly better PO-CC than PGD that is still *independent* of dimension.

One or $\log(1/\epsilon)$ **projection methods**: Starting with the work of [61], several recent works [91, 17, 88] have proposed methods that require only *one* or $\log(1/\epsilon)$ projections, under a variety of conditions on the optimization function like smoothness and strong convexity. However, these methods require that the constraint set can be written as $c(x) \leq 0$ and they require access to $\nabla c(x)$—the gradient of c–in *each* iteration. Hence, for the general nonsmooth functions, they will require at least $O(\varepsilon^{-2})$ accesses to gradients of the set's functional form. On the other hand, our method is required to access the set at only $O(\varepsilon^{-1})$ points. Furthermore, for several practical problems, the computational complexities of computing $\nabla c(x)$ and projecting are similar. For example, when $c(x) = \|x\|_{\mathrm{nuc}} - r$ where $\|\cdot\|_{\mathrm{nuc}}$ denotes the nuclear norm (see Section 4), then both gradient of $c(x)$ as well as PO requires computation of a full-SVD of x.

Frank-Wolfe methods: FW or *conditional gradient* method [28, 59] for smooth convex optimization, which uses LMO, has found renewed interest in machine learning [92, 45] due to the efficiency of computing LMO over PO [33], and its ability to ensure atomic structure and provide coreset guarantees [22]. Over the last decade, several variants of FW method and their analyses have been proposed [54, 29, 3, 31, 58, 68, 14], and FW has been extended to stochastic nonconvex

[49, 39, 75, 76, 5, 37] and online [38, 30, 52, 18, 86, 40] settings. However these methods provide dimension-free LMO-CC and SFO-CC only for smooth functions, and further it is known that FW fails to converge if subgradients are used instead of gradients [68].

Nonsmooth Frank-Wolfe methods: [84] posed an interesting question in the domain of nonsmooth optimization with LMO: can LMO-CC be reduced from the $\mathcal{O}(\varepsilon^{-4})$ bound (achieved by PGD with PO implemented via LMO: FW-PGD, see Appendix B.2) without increasing FO-CC significantly. On the lower bound side, [54] showed that $\mathcal{O}(\varepsilon^{-2})$ LMO calls are necessary. On algorithmic side, several randomized smoothing approaches combined with Frank-Wolfe methods were proposed, and can reduce LMO-CC to $\mathcal{O}(d^{1/2}\varepsilon^{-2})$. But, they come at the expense of increased $\mathcal{O}(d^{1/2}\varepsilon^{-4})$ FO calls [54, improving Theorem 5][1]. If we allow stronger oracles or additional structure in the problem, the complexity can be significantly improved. Assuming a stronger than LMO oracle introduced in [84], [73] shows that $\mathcal{O}(1/\varepsilon^2)$ LMO-CC and FO-CC are achievable for a special class of problems with low curvatures. Another popular setting is when the nonsmooth problem admits a *smooth* convex-concave saddle point reformulation [35, 23, 72, 36, 41, 42, 32, 60]. Among these the best complexity is achieved by semi-proximal mirror-prox [41] which uses $\mathcal{O}(\varepsilon^{-2})$ LMO and $\mathcal{O}(\varepsilon^{-1})$ FO calls. However, for the general nonsmooth convex optimization problem with LMO, the problem posed by [84] remained open, and is resolved by our MOLES method that achieves the optimal $\mathcal{O}(\varepsilon^{-2})$ LMO-CC and FO-CC.

2 Preliminaries and Notations

We consider Nonsmooth Convex Optimization with FO and PO (1) or LMO (2) accesses. Let $\mathcal{X} \subset \mathbb{R}^d$ be a closed convex set of diameter $D_{\mathcal{X}} := \max_{x_1, x_1 \in \mathcal{X}} \|x_1 - x_2\|$, where $\|\cdot\|$ is the Euclidean norm which corresponds to the inner product $\langle \cdot, \cdot \rangle$. Let $\mathcal{X}$ be enclosed in a closed convex set $\mathcal{X}' \subseteq \mathbb{R}^d$ to which it is easy to project, i.e. $\mathcal{X} \subset \mathcal{X}'$. For simplicity, let $\mathcal{X}'$ be a Euclidean ball of radius $R \leq D_{\mathcal{X}}$ around origin. We can satisfy $R = D_{\mathcal{X}}$ by re-centering $\mathbb{R}^d$ around any feasible point of $\mathcal{X}$. We assume $f : \mathcal{X}' \to \mathbb{R}$ to be a proper, lower semi-continuous (l.s.c.), convex Lipschitz function.We use $\partial f(x)$ to denote sub-differential of f at x, and if f is differentiable we use $\nabla f(x)$ to denote its gradient at x. We assume a first-order oracle (FO) can provide access to some subgradient at any point in $\mathcal{X}'$, i.e. FO $(x) \in \partial f(x)$.

Definition 1. *A function $f : \mathcal{X}' \to \mathbb{R}$ is G-Lipschitz if and only if, $|f(y) - f(x)| \leq G\,\|y - x\|$ for all $x, y \in \mathcal{X}'$. For a convex f, this is equivalent to:* $\max_{x \in \mathcal{X}'} \max_{g \in \partial f(x)} \|g\| \leq G$.

Definition 2. *A function $f : \mathcal{X}' \to \mathbb{R}$ is μ-strongly convex if and only if, $\frac{\mu}{2}\|y - x\|^2 + \langle g, y - x\rangle + f(x) \leq f(y)$, for all $x, y \in \mathcal{X}'$ and $g \in \partial f(x)$. Similarly, a differentiable function $f : \mathcal{X}' \to \mathbb{R}$ is said to be L-smooth if and only if, $f(y) \leq f(x) + \langle \nabla f(x), y - x\rangle + \frac{L}{2}\|y - x\|^2$ for all $x, y \in \mathcal{X}'$.*

In addition to FO, we also consider problems with stochastic FO (SFO) access, which computes stochastic subgradient of a point x with variance σ^2, as defined below:

$$SFO(x) := \widehat{g}, \text{ where } \mathbb{E}[\widehat{g} \,|\, x] = g \text{ for some } g \in \partial f(x), \text{ and } \mathbb{E}[\|\widehat{g} - g\|^2 \,|\, x] \leq \sigma^2. \tag{3}$$

Moreau Envelope: The key idea behind our method is to use "smoothed" version of the function via its Moreau envelope [63, 90] defined below.

Definition 3. *For a proper l.s.c. convex function $f : \mathcal{X}' \to \mathbb{R} \cup \{\infty\}$ defined on a closed convex set $\mathcal{X}'$ and $\lambda > 0$, its* Moreau-(Yosida) envelope *function, $f_\lambda : \mathcal{X}' \to \mathbb{R}$, is given by*

$$f_\lambda(x) \;=\; \min_{x' \in \mathcal{X}'} f(x') + \frac{1}{2\lambda}\|x - x'\|^2, \quad \textit{for all } x \in \mathcal{X}' \;. \tag{4}$$

Furthermore, the prox *operator is defined:* $\operatorname{prox}_{\lambda f}(x) := \operatorname{argmin}_{x' \in \mathcal{X}'} f(x') + \frac{1}{2\lambda}\|x - x'\|^2$.

When f is clear from context, we will use $\hat{x}_\lambda(x)$ to denote $\operatorname{prox}_{\lambda f}(x)$. Note that this definition of Moreau envelope is not standard as x' is constrained to $\mathcal{X}' \subseteq \mathbb{R}^d$. However, the following lemma (whose proof is in Appendix C.3) shows that this Moreau envelope and the prox operator still satisfies most useful properties of the standard definition.

---

[1] Needs tightening of [54, Theorem 5], by reducing the number of SFO calls per step by a factor of $d^{-1/2}$, i.e. $T_k = \lceil k d^{-1/2} \rceil$

Lemma 1. *For a closed convex set $\mathcal{X}'$, a convex proper l.s.c. function $f : \mathcal{X}' \to \mathbb{R} \cup \{\infty\}$ and $\lambda > 0$, the following hold for any $x \in \mathcal{X}'$.*
(a) $\hat{x}_\lambda(x)$ is unique and $f(\hat{x}_\lambda(x)) \leq f_\lambda(x) \leq f(x)$,
(b) f_λ is convex, differentiable, $1/\lambda$-smooth and $\nabla f_\lambda(x) = (1/\lambda)(x - \hat{x}_\lambda(x))$, and,
(c) if f is G-Lipschitz continuous, then, $\|\hat{x}_\lambda(x) - x\| \leq G\lambda$, and $f(x) \leq f_\lambda(x) + G^2\lambda/2$.

This lemma implies that, to find an ε-approximate minima of a nonsmooth f, one can instead minimize f_λ and achieve a faster convergence by exploiting its smoothness. Concretely, if f is G-Lipschitz and $\lambda = O(\varepsilon/G^2)$, and Lemma 1(c) ensures that solving f_λ up to $O(\varepsilon)$ accuracy guarantees $O(\varepsilon)$ accuracy in the original minimization of f (Lemma 2). This insight allows us to design a simple method that can reduce PO-CC but at the cost of a higher FO-CC. Next section starts with this result as a warm-up and then presents our method, which ensures reduced PO-CC with optimal FO-CC.

3 Main Results

We present our main results in this section. We first present the main ideas in Section 3.1 and then the results for PO and LMO settings in Sections 3.2 and 3.3 respectively.

3.1 Main Ideas

We are interested in the NSCO problem (1). As discussed in the previous section, instead of optimizing $f(x)$ over $\mathcal{X}$, we can instead optimize the Moreau envelope function $f_\lambda(x)$ with $\lambda = O(\epsilon)$ to get ϵ-suboptimality. Since by Lemma 1, $f_\lambda(\cdot)$ is a $1/\lambda$-smooth convex function, a straightforward approach is to iteratively optimize $f_\lambda(x)$ using Nesterov's accelerated gradient descent (AGD) [69] method. But to get gradients of $f_\lambda(x)$, we will need to solve the *inner problem* (4) approximately.

A key insight is that since the inner problem does not involve the constraint set $\mathcal{X}$, PO calls are not required in inner steps for estimating $\nabla f_\lambda(x)$. So the total number of PO calls required is equal to the total number of outer steps in minimizing $f_\lambda(x)$, which for Nesterov's AGD is $\mathcal{O}(1/\sqrt{\lambda\varepsilon}) = \mathcal{O}(\varepsilon^{-1})$. We see that this already improves over the $\mathcal{O}(\varepsilon^{-2})$ projections of PGD. However, since $\nabla f_\lambda(x)$ needs to be estimated to a good accuracy, the total number of FO calls, including in the inner loop, turns out to be $\mathcal{O}(\varepsilon^{-3})$, which is worse than the optimal $\mathcal{O}(\varepsilon^{-2})$ FO calls of PGD.

Similarly, when we have access to LMO for $\mathcal{X}$, we could optimize f_λ using FW [28, 45], with total number of outer steps $= \mathcal{O}(1/\lambda\varepsilon) = \mathcal{O}(\varepsilon^{-2})$, and hence the total number of LMO calls is $\mathcal{O}(\varepsilon^{-2})$. However, this again leads to suboptimal $\mathcal{O}(\varepsilon^{-4})$ FO calls. We can improve the FO-CC to $\mathcal{O}(\varepsilon^{-3})$ by using the conditional gradient sliding algorithm [56] instead of FW method, but this is still worse than the optimal $\mathcal{O}(\varepsilon^{-2})$ FO calls.

In order to achieve optimal number of FO calls, we directly optimize the Moreau envelope through the following joint optimization.

$$\min_{x \in \mathcal{X}, x' \in \mathcal{X}'} \left[\Psi_\lambda(x, x') := f(x') + \psi_\lambda(x, x')\right] \quad \text{where} \quad \psi_\lambda(x, x') := \frac{1}{2\lambda}\|x' - x\|^2, \tag{5}$$

where the function $\Psi_\lambda : \mathcal{X}' \times \mathcal{X}' \to \mathbb{R}$ is convex in the joint variable (x, x'). The main advantage of this new form is that, this is a composite optimization problem with a nonsmooth part (corresponding to $f(x')$) and a $2/\lambda$-smooth part (corresponding to $(1/2\lambda)\|x' - x\|^2$) with the constrained variable $x \in \mathcal{X}$ only appearing in the smooth part. Now, by the following lemma, an approximate minimizer of Ψ_λ, is also an approximate minimizer of the Moreau envelope f_λ, and further if $\lambda = \varepsilon/G^2$, it is also an approximate minimizer of the original function f. A proof is provided in Appendix C.1.

Lemma 2. *Under the same assumptions as in Lemma 1, let $\mathcal{X} \subseteq \mathcal{X}'$ be a convex subset and Ψ_λ be defined as in* (5). *Then, (i) $\min_{x \in \mathcal{X}} \min_{x \in \mathcal{X}'} \Psi_\lambda(x, x') = \min_{x \in \mathcal{X}} f_\lambda(x) \leq \min_{x \in \mathcal{X}} f(x)$, and (ii) for any random vectors $(x_\varepsilon, x'_\varepsilon) \in \mathcal{X} \times \mathcal{X}'$, $\mathbb{E}[f(x_\varepsilon)] - G^2\lambda/2 \leq \mathbb{E}[f_\lambda(x_\varepsilon)] \leq \mathbb{E}[\Psi_\lambda(x_\varepsilon, x'_\varepsilon)]$.*

Our algorithm essentially solves (5) using Gradient Sliding [55] and Conditional Gradient Sliding [56] frameworks, which are optimal for minimizing composite problems of the form (5) for the PO and LMO settings respectively. The resulting algorithm for PO setting, called MOPES is given in Algorithm 1. The algorithm for LMO setting, called MOLES is presented in Algorithm 2. The only difference between MOPES and MOLES is that MOLES uses FW to compute approximate projections while MOPES uses exact projections. Finally, our algorithms extend straightforwardly

Algorithm 1: MOPES: MOreau Projection Efficient Subgradient method

```
Input: $f$, $\mathcal{X}$, $\mathcal{X}'$, $G$, $D_\mathcal{X}$, $R$, $x_0$, $K$, $\tilde{D}$, $c'$, $\lambda$,
1.1  Set $x'_0 = z'_0 = x_0 = z_0 = x_0$
1.2  for $k = 1, \ldots, K$ do
1.3      Set $\beta_k = \frac{4}{\lambda k}$ , $\gamma_k = \frac{2}{k+1}$ , and $T_k = \left\lceil \frac{(4G^2+\sigma^2)\lambda^2 K k^2}{2\tilde{D}} \right\rceil$
1.4      Set $(y_k, y'_k) = (1-\gamma_k)\cdot(x_{k-1}, x'_{k-1}) + \gamma_k \cdot (z_{k-1}, z'_{k-1})$
1.5      Set $z_k = \mathcal{P}_\mathcal{X}\big(z_{k-1} - \frac{1}{\beta_k}\cdot\nabla_{y_k}\Psi_\lambda(y_k, y'_k)\big)$ (1)    // Note $\nabla_{y_k}\Psi_\lambda(y_k, y'_k) = \frac{y_k - y'_k}{\lambda}$
1.6      Set $(z'_k, \tilde{z}'_k) =$ Prox-Slide$\big(\nabla_{y'_k}\psi_\lambda(y_k, y'_k), z'_{k-1}, \beta_k, T_k\big)$    // $\nabla_{y'_k}\psi_\lambda(y_k, y'_k) = \frac{y'_k - y_k}{\lambda}$
1.7      Set $(x_k, x'_k) = (1-\gamma_k)\cdot(x_{k-1}, x'_{k-1}) + \gamma_k \cdot (z_k, \tilde{z}'_k)$
Output: $(x_K, x'_K)$
1.8  Prox-Slide($g$, $u_0$, $\beta$, $T$) // Approx. resolve $\mathrm{prox}_{f/\beta}(u'_0 - g/\beta)$[55]:
1.9      Set $\widetilde{u}_0 = u_0$
1.10     for $t = 1, \ldots, T$ do
1.11         Set $\theta_t = \frac{2(t+1)}{t(t+3)}$, $\widehat{g}_{t-1} = \mathrm{SFO}\,(u_{t-1})$ (3)
1.12         Set $\widehat{u}_t = u_{t-1} - \frac{1}{(1+t/2)\beta}\cdot(\widehat{g}_{t-1} + \beta(u_{t-1} - (u_0 - g/\beta)))$
              // subgradient method step for $\phi(u) := f(u) + \frac{\beta}{2}\|u - \big(u_0 - \frac{g}{\beta}\big)\|^2$
1.13         Set $u_t = \widehat{u}_t \cdot \min(1, R/\|u_t\|)$    // projection of $\widehat{u}_t$ onto $\mathcal{X}'$: $\mathcal{P}'_\mathcal{X}(\mathbf{u}_t)$
1.14         Set $\widetilde{u}_t = \big(1-\theta_t\big)\cdot\widetilde{u}_{t-1} + \theta_t \cdot u_t$
1.15     return $(u_T, \widetilde{u}_T)$
```

to the case of stochastic subgradients through a stochastic first order oracle (SFO) and the resulting bounds depend on the variance of SFO in addition to the Lipschitz constant of $f(\cdot)$.

3.2 MOreau Projection Efficient Subgradient (MOPES) method

A pseudocode of our algorithm MOPES is presented in Algorithm 1. At a high level, MOPES is an inexact Accelerated Proximal Gradient method (APGD) [67, 8] scheme which tries to implement Nesterov's AGD algorithm on $\Psi_\lambda(x, x')$. Now, standard AGD updates for solving $\min_{x\in\mathcal{X},x'}\Psi_\lambda(x, x')$, *if* Ψ_λ *were smooth* are:

$$\boxed{\begin{aligned} \beta_k &\leftarrow 4/\lambda k\ ,\ \gamma_k \leftarrow 2/(k+1) \\ (y_k, y'_k) &\leftarrow (1-\gamma_k)\,(x_{k-1}, x'_{k-1}) + \gamma_k(z_{k-1}, z'_{k-1}) \\ z_k &\leftarrow \mathcal{P}_\mathcal{X}\left(z_{k-1} - \nabla_{y_k}\Psi_\lambda(y_k, y'_k)/\beta_k\right),\ \ z'_k \leftarrow z'_{k-1} - \nabla_{y'_k}\Psi_\lambda(y_k, y'_k)/\beta_k, \\ (x_k, x'_k) &\leftarrow (1-\gamma_k)\,(x_{k-1}, x'_{k-1}) + \gamma_k(z_k, z'_k). \end{aligned}} \tag{6}$$

MOPES essentially implements the above updates, but as Ψ_λ is nonsmooth in x', we use prox steps for the x' variable instead of the GD steps. The prox step—$\mathrm{prox}_{f/\beta_k}\big(z'_{k-1} - \nabla_{y'_k}\psi_\lambda(y_k, y'_k)/\beta_k\big)$—is implemented via `Prox-Slide` procedure (see Line 1.6), which is the standard subgradient method applied to a strongly convex function ϕ (see Line 1.12). Now, `Prox-Slide` procedure outputs two points $(z'_k, \tilde{z}'_k)$ which are the final and average iterates, respectively, of the subgradient method, This achieves optimal FO-CC by exploiting strong convexity of ϕ. If we were to use only the average of the iterates, the FO-CC would increase by a factor of $\mathcal{O}(\varepsilon^{-1})$ (see the failed attempt in Appendix A.1).

Note that MOPES needs only a PO call & no FO call in Line 1.5, and only a FO/SFO call in Line 1.11. Therefore, we bound below, the total number of PO calls K and the number of FO/SFO calls $K \cdot T$.

Theorem 1. *Let $f : \mathcal{X}' \to \mathbb{R}$ be a G-Lipschitz continuous proper l.s.c. convex function equipped with a SFO with variance σ^2, and $\mathcal{X} \subseteq \mathcal{X}' = B(0, R)$ be some convex subset equipped with a projection oracle $\mathcal{P}_\mathcal{X}$ and contained inside the Euclidean ball of radius R around origin. If we run MOPES (Algorithm 1) with inputs $\lambda = \varepsilon/G^2$, $\tilde{D} = c\|x_0 - x^*\|^2$ and $K = \lceil 2\sqrt{(10+8c)}G\|x_0 - x^*\|/\varepsilon\rceil$ for any absolute constant $c > 0$ and $x^* \in \operatorname{argmin}_{x\in\mathcal{X}} f(x)$, then, using $\mathcal{O}(\frac{G\|x_0-x^*\|}{\varepsilon})$ PO calls and $\mathcal{O}(\frac{(G^2+\sigma^2)\|x_0-x^*\|^2}{\varepsilon^2})$ FO calls, it outputs x_K satisfying $f(x_K) - \min_{x\in\mathcal{X}} f(x) \le \varepsilon$.*

Remarks: Note that FO-CC is same as that of PGD (up to constants) while PO-CC is significantly better. A natural open question is if PO-CC can be further reduced. Also, MOPES requires querying of SFO/FO at u_{t-1} which is not necessarily in $\mathcal{X}$ but is always in $\mathcal{X}'$ (Line 1.11). Recall from Section 2 that $\mathcal{X}'$ is a Euclidean ball of radius $R \leq D_{\mathcal{X}}$ around origin. Being able to query SFO/FO in $\mathcal{X}'$ seems like a mildly stricter requirement than the standard requirement of querying on $\mathcal{X}$ only, but for most practical problems this seems feasible. Even if f is unknown outside of $\mathcal{X}$, theoretically we could work with its convex extension to the entire space, which remains G Lipschitz (see Section 6). Also, notice that the guarantee only depends on the diameter $D_{\mathcal{X}}$ of the constraint set $\mathcal{X}$ and not the radius R of the enclosing set $\mathcal{X}'$. This is so because the first-order method only depends on the distance from initial point (x_0, x'_0) to the desired solution (x^*, x^*), which is $\mathcal{O}(\|x_0 - x^*\|) = \mathcal{O}(D_{\mathcal{X}})$, as $x'_0 = x_0$. Finally, for simplicity of exposition, we provide desired suboptimality ϵ as an input to MOPES–in practice, we can remove this assumption by using standard doubling trick [80, Algorithm 6].

See Appendices A.2 and C.2.1 for a proof sketch and a detailed proof, respectively, of Theorem1. At a high level, our proof uses a potential function [6] for analyzing APGD, combines it with Proposition 1 which provides a fast convergence guarantee on `Prox-Slide` iterates, and then apply standard APGD proof techniques [81] to obtain the final result.

Proposition 1 (Proposition 3, informal)**.** *For some* $0 < \tau_k \leq 1.5$*, output of* `Prox-Slide` *satisfies*

$$\phi_k(\tilde{z}'_k) - \phi_k(x') + \frac{\beta_k}{2}\|z'_{k-1} - x'\|^2 \leq \frac{\beta_k}{2}[\tau_k\|z'_{k-1} - x'\|^2 - \tau_{k+1}\|z'_k - x'\|^2] + \frac{16\,G^2}{\beta_k T_k}.$$

3.3 MOreau Linear minimization oracle Efficient Subgradient (MOLES) method

Algorithm 2: MOLES: MOreau Linear minimization oracle Efficient Subgradient method

```
Use the same steps as MOPES (Algorithm 1), but replace Line 1.5 with:
2.5  Set $z_k$ = FW-Based-Projection($z_{k-1} - \frac{1}{\beta_k} \cdot \nabla_{y_k}\Psi_\lambda(y_k, y'_k)$, $z_{k-1}$, $\lceil \frac{7KD_{\mathcal{X}}^2}{c'\tilde{D}} \rceil$)

2.16 FW-Based-Projection($z$, $u_0$, $\hat{T}$):
         // $\hat{T}$ steps of standard Frank-Wolfe for $\min_{u\in\mathcal{X}} \|u - z\|^2$
2.17     for $t = 1, \ldots, \hat{T}$ do
2.18         Set $s_t$ = LMO ($u_{t-1} - z$)
2.19         Set $u_t = ((t-1) \cdot u_{t-1} + 2 \cdot s_t)/(t+1)$
2.20     return $u_{\hat{T}}$
```

We now present our results for the LMO setting. A pseudocode of our algorithm, MOLES, is presented in Algorithm 2. MOLES does exactly the same steps as in MOPES (Algorithm 1), except that the projection in Line 1.5 of MOPES is estimated using the LMO and Frank-Wolfe algorithm. At the outer-step k, the output z_k of `FW-Based-Projection`, which uses $\hat{T} = \mathcal{O}(1/\varepsilon)$ LMO calls to approximately project, satisfies the following bound on the projection problem's Wolfe dual gap [45]:

$$\max_{s\in\mathcal{X}} \beta_k \left\langle z_k - \left(z_{k-1} - (1/\beta_k) \cdot \nabla_{y_k}\Psi_\lambda(y_k, y'_k)\right), z_k - s \right\rangle \leq 4c'\tilde{D}/\lambda K k \tag{7}$$

In practice we can use the above condition as a stopping criterion for `FW-Based-Projection`. The following theorem, a proof of which is in Appendix C.2.2, provides the convergence guarantee.

Theorem 2. *Let* $f : \mathcal{X}' \to \mathbb{R}$ *be a G-Lipschitz continuous proper l.s.c. convex function equipped with an SFO with variance* σ^2*, and* $\mathcal{X} \subseteq \mathcal{X}' = B(0, R)$ *be some convex subset of diameter* $D_{\mathcal{X}}$ *equipped with an LMO and contained inside the Euclidean ball of radius around origin. If we run MOLES (Algorithm 2) with inputs* $\lambda = \varepsilon/G^2$, $\tilde{D} = cD_{\mathcal{X}}$, $K = \lceil 2\sqrt{10 + 8c(1+c')}G\|x_0 - x^*\|/\varepsilon \rceil$*, for some absolute constants c,c'* > 0 *and* $x^* \in \operatorname{argmin}_{x\in\mathcal{X}} f(x)$*, then, using* $\mathcal{O}(\frac{G^2 D_{\mathcal{X}}^2}{\varepsilon^2})$ *LMO calls and* $\mathcal{O}(\frac{(G^2+\sigma^2)D_{\mathcal{X}}^2}{\varepsilon^2})$ *FO calls it outputs* x_K *satisfying* $f(x_K) - \min_{x\in\mathcal{X}} f(x) \leq \varepsilon$.

Remarks: Thus our algorithm obtains the optimal $\mathcal{O}(\varepsilon^{-2})$ dimension independent FO-CC and LMO-CC for general nonsmooth functions [54]. Similar to MOPES, here also, we require FO/SFO of f to be well-defined in $\mathcal{X}'$. If f is a maximum of smooth convex functions, then we can get similar PO-CC by applying min-max saddle point approaches [41]. But even for such functions, it is non-trivial to extend saddle point approaches to stochastic FO, which is important in practice. In contrast, our result matches the optimal FO-CC (on all key parameters) of unconstrained stochastic-PGD method.

4 Applications

We first explain the gain of MOPES in practical applications. One of the main applications of our method is Empirical Risk Minimization (ERM) with nonsmooth loss functions. For a nonsmooth loss f_i for the ith training example in a set of n examples, the general form of ERM is:

$$\min_{x\in\mathcal{X}\subseteq\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} f_i(x) \quad \text{For example,} \qquad \min_{X\in\mathbb{R}^{m\times p}; \|X\|_{\text{nuc}}\le r} \frac{1}{n}\sum_{i=1}^{n} \max(0, 1-b_i\langle X, A_i\rangle)\ , \tag{8}$$

which is known as the *low rank SVM* [85, 83, 74] as the nuclear norm constraint induces low rank solutions. As the cost of a single PO call involves a full SVD on a potentially full rank X, MOPES significantly improves over the competing baseline as we showcase in Fig. 1. There are numerous examples of ERMs with costly POs to a nuclear norm ball (e.g. max-margin collaborative filtering [79]), to an ℓ_1 norm ball (e.g. sparse SVM [13, 93, 4]), and to a large number of linear constraints (e.g. robust classification [10]). One notable example is *SVM with hard constraints* on a subset of the training data, so that some predictions are constrained to be always accurate [70] (See Appendix E.2).

In all these examples, PO calls can be more costly than FO calls, making MOPES attractive. In comparison, popular *accelerated proximal point methods*, such as FISTA [8], cannot handle general nonsmooth losses. The standard *projected subgradient methods* suffer from $\mathcal{O}(\varepsilon^{-2})$ PO-CC. *Mirror descent* [64] may give better d dependence, but it too requires $O(\varepsilon^{-2})$ (proximal) operations.

Now, several nonsmooth loss functions have a special structure where they can be written as a *smooth minimax problem*. Such (stochastic) problems can be solved using $\mathcal{O}_\varepsilon(\varepsilon^{-1})$ (S)FO and PO calls [66]. However, the resulting complexity scales up with the dimension d or the number of samples n. Thus the PO-CC of the minimax formulations becomes inefficient (even with variance reduction [71, 16]), whenever n or d gets large. In the deterministic setting, each step of the optimization problem requires gradient of the entire empirical risk function, so for problems with large n and small ε, total time complexity can be significantly higher than MOPES. See Appendix E.1 for exact complexities.

Further, beyond ERM, nice minimax representations might not always exist. For example, in reinforcement learning/optimal control setting, f could be an (already trained) *input-convex neural network* [2, 19] approximating the Q-function over a continuous constrained action space [20].

For several of the above examples, LMO might be preferred if it is significantly more efficient than a PO call e.g., for high-dimensional low rank SVM, a LMO call only requires computing top singular vector, as opposed to full SVD required by a PO. Further, LMO-based methods have an additional benefit of preserving the desired structure of the solution, such as sparse and low rank structures [22]. This makes MOLES particularly attractive, for example, in differentially private collaborative filtering [46], where structured updates lead to improved privacy guarantees. In Appendix E, we present the details of some these examples, and give analytical comparisons to competing methods.

5 Empirical Results

We experimentally evaluate[2] MOPES (Algorithm 1) and MOLES (Algorithm 2) methods on a low rank SVM problem [85] of the form (8) on a subset of the Imagewoof 2.0 dataset [43]. The training data contains $n = 400$ samples $\{(A_i, y_i)\}_{i=1}^n$ where A_i is a 224×224 grayscale image labeled using $y_i \in \{0, 1\}$. Note that the effective dimension is $d = 50176$. We use $r = 0.1$ as nuclear norm ball radius of $\mathcal{X}$. First, we compare the PO and FO efficiencies of MOPES with those of PGD with a fixed and PGD with a diminishing stepsize. In Figure 1 we plot the mean (over 10 runs) sub-optimality gap: $f(x_k) - \hat{f}^*$, of the iterates against the number of PO (top) and FO (bottom) calls, respectively, used to obtain that iterate. Next, we compare the LMO and FO efficiencies of MOLES with those of FW-PGD (see Algorithm 3 in Appendix B.2) and Randomized Frank-Wolfe (RandFW) [54, Theorem 5] methods with a fixed and diminishing stepsizes. In Figure 2 we plot the

---

[2] Code for the experiments is available at `https://github.com/tkkiran/MoreauSmoothing`

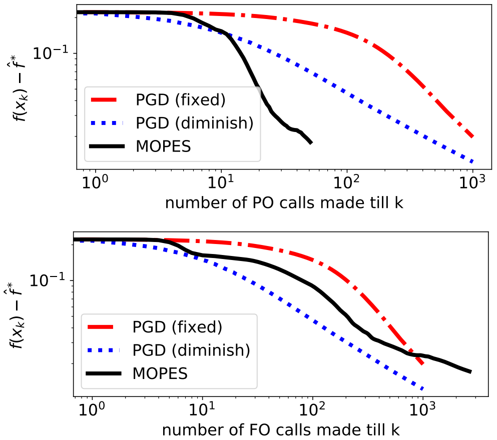

Figure 1: MOPES uses significantly fewer PO calls and comparable number of FO calls than PGD

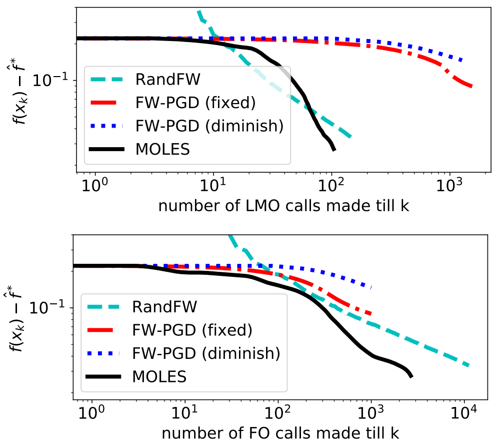

Figure 2: MOLES uses fewer LMO calls and similar number of FO calls than FW-PGD and RandFW

mean (over 10 runs) sub-optimality gap: $f(x_k) - \hat{f}^*$, of the iterates against the number of LMO (top) and FO (bottom) calls, respectively, used to obtain that iterate. In both these plots, while MOPES/MOLES and baselines have comparable FO-CC, MOPES/MOLES is significantly more efficient in the number of PO/LMO calls, matching our Theorems 1 and 2. As the nuclear norm ball has a non-trivial projection/LMO, PO-CC/LMO-CC will dominate the total run-time as m becomes larger for $X \in \mathbb{R}^{m \times m}$. Note that matrix mirror descent [47] would also require $O(\varepsilon^{-2})$ SVD based proximal operations. We provide additional experimental details in Appendix D.

6 Conclusion

We study a canonical problem in optimization: minimizing a nonsmooth Lipschitz continuous convex function over a convex constraint set. We assume that the function is accessed with a first-order oracle (FO) and the set is accessed with either a projection oracle (PO) or a linear minimization oracle (LMO). In this general setting, we address the fundamental question of reducing the number of accesses to the function and the set. When using projections, we introduce MOPES, and show that it finds an ε-suboptimal solution with $\mathcal{O}(\varepsilon^{-2})$ FO calls and $\mathcal{O}(\varepsilon^{-1})$ PO calls. This is optimal in the number of FO calls and significantly improves over competing methods in the number of PO calls (see Table 1). When using linear minimizations, we introduce MOLES, and show that it finds an ε-suboptimal solution with $\mathcal{O}(\varepsilon^{-2})$ FO and LMO calls. This is optimal in both the number of PO and the number of LMO calls. This resolves a question left open since [84] on designing the optimal Frank-Wolfe type algorithm for nonsmooth functions.

The two properties we need of the superset $\mathcal{X}' \supseteq \mathcal{X}$ are that (a) it is easy to project onto $\mathcal{X}'$ and (b) f is G-Lipschitz on $\mathcal{X}'$. In our paper, we choose $\mathcal{X}'$ to be a Euclidean ball (which is easy to project to) but any other choice of $\mathcal{X}'$ which satisfies the above properties works just as well. For example, if f is Lipschitz everywhere, we can set $\mathcal{X}' = \mathbb{R}^d$ and ignore the explicit projection to $\mathcal{X}'$ in line 1.13 of Algorithm 1. However, even if f is G-Lipschitz inside the constraint $\mathcal{X}$, f could (i) have unbounded Lipschitz constant, or (ii) be undefined just outside of $\mathcal{X}$. Thus an $\mathcal{X}'$ satisfying our requirements may not exist. In our experiments, we do not explicitly project onto $\mathcal{X}'$ (line 1.13) but still observed that $\|x_k - x'_k\| = \mathcal{O}(G\lambda) = \mathcal{O}(\varepsilon)$ and small, which hints that we may only need Lipschitzness over a much smaller set, say $\mathcal{X} + B(0, \mathcal{O}(G\lambda))$. Theoretically, we can work around the above issues by minimizing the convex extension $f_{\mathcal{X}} : \mathbb{R}^d \to \mathbb{R}$ of the function f from the set $\mathcal{X}$, defined as $f_{\mathcal{X}}(x') := \max_{x \in \mathcal{X}} \max_{g \in \partial f(x)} f(x) + \langle g, x' - x \rangle$. The extension $f_{\mathcal{X}}$ has the same value as f inside $\mathcal{X}$ and is G-Lipschitz everywhere. Therefore the minimization problems $\min_{x \in \mathcal{X}} f(x)$ and $\min_{x \in \mathcal{X}} f_{\mathcal{X}}(x')$ are equivalent. However, it is not clear if we can estimate the gradients of $f_{\mathcal{X}}$ efficiently. We did not find any relevant prior work and leave this question for future work.

Another possible direction of future work is developing ε-horizon oblivious algorithms, where we need not fix K and ε a priori. In our experiments, we observed that varying λ according to $\lambda_k = \mathcal{O}(\frac{D_{\mathcal{X}}}{Gk})$ and $\beta_k = \frac{4}{\lambda_k k}$ works just as well as fixing it.

Broader Impact

As this is foundational research that is theoretical in nature, it is hard to predict any foreseeable societal consequence.

Funding Disclosure

Funding in direct support of this work: NSF grant 1927712, NSF grant 1929955, NSF grant 2019844, and Google faculty research award.

References

[1] J.-B. Alayrac, P. Bojanowski, N. Agrawal, J. Sivic, I. Laptev, and S. Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4575–4583, 2016.

[2] B. Amos, L. Xu, and J. Z. Kolter. Input convex neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 146–155. JMLR. org, 2017.

[3] F. Bach. Duality between subgradient and conditional gradient methods. *SIAM Journal on Optimization*, 25(1):115–129, 2015.

[4] F. Bach, R. Jenatton, J. Mairal, G. Obozinski, et al. Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning*, 4(1):1–106, 2012.

[5] K. Balasubramanian and S. Ghadimi. Zeroth-order (non)-convex stochastic optimization via conditional gradient and gradient updates. In *Advances in Neural Information Processing Systems*, pages 3455–3464, 2018.

[6] N. Bansal and A. Gupta. Potential-function proofs for first-order methods. *arXiv preprint arXiv:1712.04581*, 2017.

[7] H. H. Bauschke, M. N. Dao, and S. B. Lindstrom. Regularizing with bregman–moreau envelopes. *SIAM Journal on Optimization*, 28(4):3208–3228, 2018.

[8] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[9] A. Beck and M. Teboulle. Smoothing and first order methods: A unified framework. *SIAM Journal on Optimization*, 22(2):557–580, 2012.

[10] A. Ben-Tal, S. Bhadra, C. Bhattacharyya, and A. Nemirovski. Efficient methods for robust classification under uncertainty in kernel matrices. *Journal of Machine Learning Research*, 13 (Oct):2923–2954, 2012.

[11] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific Belmont, 2 edition, 1999.

[12] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[13] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In *ICML*, volume 98, pages 82–90, 1998.

[14] G. Braun, S. Pokutta, and D. Zink. Lazifying conditional gradient algorithms. *Journal of Machine Learning Research*, 20(71):1–42, 2019.

[15] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on optimization*, 20(4):1956–1982, 2010.

[16] Y. Carmon, Y. Jin, A. Sidford, and K. Tian. Variance reduction for matrix games. In *Advances in Neural Information Processing Systems*, pages 11377–11388, 2019.

[17] J. Chen, T. Yang, Q. Lin, L. Zhang, and Y. Chang. Optimal stochastic strongly convex optimization with a logarithmic number of projections. *arXiv preprint arXiv:1304.5504*, 2013.

[18] L. Chen, C. Harshaw, H. Hassani, and A. Karbasi. Projection-free online optimization with stochastic gradient: From convexity to submodularity. In *International Conference on Machine Learning*, pages 814–823, 2018.

[19] Y. Chen, Y. Shi, and B. Zhang. Optimal control via neural networks: A convex approach. In *International Conference on Learning Representations*, 2018.

[20] Y. Chen, Y. Shi, and B. Zhang. Input convex neural networks for optimal voltage regulation. *arXiv preprint arXiv:2002.08684*, 2020.

[21] A. Clark and Contributors. Pillow: Python image-processing library, 2020. URL `https://pillow.readthedocs.io/en/stable/`. Documentation.

[22] K. L. Clarkson. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Transactions on Algorithms (TALG)*, 6(4):1–30, 2010.

[23] B. Cox, A. Juditsky, and A. Nemirovski. Decomposition techniques for bilinear saddle point problems and variational inequalities with affine monotone operators. *Journal of Optimization Theory and Applications*, 172(2):402–435, 2017.

[24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[25] O. Devolder, F. Glineur, and Y. Nesterov. First-order methods of smooth convex optimization with inexact oracle. *Mathematical Programming*, 146(1-2):37–75, 2014.

[26] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, pages 272–279, 2008.

[27] J. C. Duchi, P. L. Bartlett, and M. J. Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

[28] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[29] R. M. Freund and P. Grigas. New analysis and results for the frank–wolfe method. *Mathematical Programming*, 155(1-2):199–230, 2016.

[30] D. Garber and E. Hazan. A linearly convergent conditional gradient algorithm with applications to online and stochastic optimization. *arXiv preprint arXiv:1301.4666*, 2013.

[31] D. Garber and E. Hazan. Faster rates for the frank-wolfe method over strongly-convex sets. In *32nd International Conference on Machine Learning, ICML 2015*, 2015.

[32] G. Gidel, T. Jebara, and S. Lacoste-Julien. Frank-wolfe algorithms for saddle point problems. In *Artificial Intelligence and Statistics*, pages 362–371. PMLR, 2017.

[33] G. Gidel, F. Pedregosa, and S. Lacoste-Julien. Frank-wolfe splitting via augmented lagrangian method. In *International Conference on Artificial Intelligence and Statistics*, pages 1456–1465, 2018.

[34] A. A. Goldstein. Convex programming in hilbert space. *Bulletin of the American Mathematical Society*, 70(5):709–710, 1964.

[35] J. H. Hammond. *Solving asymmetric variational inequality problems and systems of equations with generalized nonlinear programming algorithms*. PhD thesis, Massachusetts Institute of Technology, 1984.

[36] Z. Harchaoui, A. Juditsky, and A. Nemirovski. Conditional gradient algorithms for norm-regularized smooth convex optimization. *Mathematical Programming*, 152(1-2):75–112, 2015.

[37] H. Hassani, A. Karbasi, A. Mokhtari, and Z. Shen. Stochastic conditional gradient++: (non-)convex minimization and continuous submodular maximization. *arXiv preprint arXiv:1902.06992*, 2019.

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

[39] E. Hazan and H. Luo. Variance-reduced and projection-free stochastic optimization. In *International Conference on Machine Learning*, pages 1263–1271, 2016.

[40] E. Hazan and E. Minasyan. Faster projection-free online learning. *arXiv preprint arXiv:2001.11568*, 2020.

[41] N. He and Z. Harchaoui. Semi-proximal mirror-prox for nonsmooth composite minimization. In *Advances in Neural Information Processing Systems*, pages 3411–3419, 2015.

[42] N. He and Z. Harchaoui. Stochastic semi-proximal mirror-prox. Workshop on Optimization for Machine Learning, 2015. URL `https://opt-ml.org/papers/OPT2015_paper_27.pdf`.

[43] J. Howard. Imagenette, 2019. URL `https://github.com/fastai/imagenette`. Github repository with links to dataset.

[44] P. J. Huber. *Robust statistical procedures*, volume 68. SIAM, 1996.

[45] M. Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th international conference on machine learning*, pages 427–435, 2013.

[46] P. Jain, O. D. Thakkar, and A. Thakurta. Differentially private matrix completion revisited. In *International Conference on Machine Learning*, pages 2215–2224. PMLR, 2018.

[47] B. Kulis, M. A. Sustik, and I. S. Dhillon. Low-rank kernel learning with bregman matrix divergences. *Journal of Machine Learning Research*, 10(Feb):341–376, 2009.

[48] A. Kundu, F. Bach, and C. Bhattacharya. Convex optimization over intersection of simple sets: improved convergence rate guarantees via an exact penalty approach. In *International Conference on Artificial Intelligence and Statistics*, pages 958–967. PMLR, 2018.

[49] S. Lacoste-Julien. Convergence rate of frank-wolfe for non-convex objectives. *arXiv preprint arXiv:1607.00345*, 2016.

[50] S. Lacoste-Julien, M. Schmidt, and F. Bach. A simpler approach to obtaining an $O(1/t)$ convergence rate for the projected stochastic subgradient method. *arXiv preprint arXiv:1212.2002*, 2012.

[51] S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-coordinate frank-wolfe optimization for structural svms. In *Proceedings of the 30th international conference on machine learning*, pages 53–61, 2013.

[52] J. Lafond, H.-T. Wai, and E. Moulines. On the online frank-wolfe algorithms for convex and non-convex optimizations. *arXiv preprint arXiv:1510.01171*, 2015.

[53] G. Lan. An optimal method for stochastic composite optimization. *Mathematical Programming*, 133(1-2):365–397, 2012.

[54] G. Lan. The complexity of large-scale convex programming under a linear optimization oracle. *arXiv preprint arXiv:1309.5550*, 2013.

[55] G. Lan. Gradient sliding for composite optimization. *Mathematical Programming*, 159(1-2): 201–235, 2016.

[56] G. Lan and Y. Zhou. Conditional gradient sliding for convex optimization. *SIAM Journal on Optimization*, 26(2):1379–1409, 2016.

[57] G. Lan, Z. Lu, and R. D. Monteiro. Primal-dual first-order methods with $O(1/\varepsilon)$ iteration-complexity for cone programming. *Mathematical Programming*, 126(1):1–29, 2011.

[58] G. Lan, S. Pokutta, Y. Zhou, and D. Zink. Conditional accelerated lazy stochastic gradient descent. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1965–1974, 2017.

[59] E. S. Levitin and B. T. Polyak. Constrained minimization methods. *USSR Computational mathematics and mathematical physics*, 6(5):1–50, 1966.

[60] F. Locatello, A. Yurtsever, O. Fercoq, and V. Cevher. Stochastic frank-wolfe for composite convex minimization. In *Advances in Neural Information Processing Systems*, pages 14246–14256, 2019.

[61] M. Mahdavi, T. Yang, R. Jin, S. Zhu, and J. Yi. Stochastic gradient descent with only one projection. In *Advances in Neural Information Processing Systems*, pages 494–502, 2012.

[62] J. J. Moreau. Functions convexes duales et points proximaux dans un espace hilbertien. *CR Acad. Sci. Paris Ser. A Math.*, 255:2897–2899, 1962.

[63] J.-J. Moreau. Proximité et dualité dans un espace hilbertien. *Bulletin de la Société mathématique de France*, 93:273–299, 1965.

[64] A. S. Nemirovski and D. B. Yudin. *Problem complexity and method efficiency in optimization.* Wiley-Interscience, 1 edition, 1983.

[65] Y. Nesterov. *Introductory lectures on convex programming volume I: Basic course*. Lecture notes, 1998.

[66] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical programming*, 103 (1):127–152, 2005.

[67] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[68] Y. Nesterov. Complexity bounds for primal-dual methods minimizing the model of objective function. *Mathematical Programming*, 171(1-2):311–330, 2018.

[69] Y. E. Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983.

[70] Q. Nguyen. *Efficient learning with soft label information and multiple annotators*. PhD thesis, University of Pittsburgh, 2014.

[71] B. Palaniappan and F. Bach. Stochastic variance reduction methods for saddle-point problems. In *Advances in Neural Information Processing Systems*, pages 1416–1424, 2016.

[72] F. Pierucci, Z. Harchaoui, and J. Malick. A smoothing approach for composite conditional gradient with nonsmooth loss. Technical report, [Research Report] RR-8662, INRIA Grenoble, 2014.

[73] S. N. Ravi, M. D. Collins, and V. Singh. A deterministic nonsmooth frank wolfe algorithm with coreset guarantees. *Informs Journal on Optimization*, 1(2):120–142, 2019.

[74] M. I. Razzak. *Sparse support matrix machines for the classification of corrupted data*. PhD thesis, Queensland University of Technology, 2019.

[75] S. J. Reddi, S. Sra, B. Póczos, and A. Smola. Stochastic frank-wolfe methods for nonconvex optimization. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1244–1251. IEEE, 2016.

[76] A. K. Sahu, M. Zaheer, and S. Kar. Towards gradient free and projection free stochastic optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3468–3477, 2019.

[77] M. Schmidt, N. L. Roux, and F. R. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in neural information processing systems*, pages 1458–1466, 2011.

[78] O. Shamir and T. Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International conference on machine learning*, pages 71–79, 2013.

[79] N. Srebro, J. Rennie, and T. S. Jaakkola. Maximum-margin matrix factorization. In *Advances in neural information processing systems*, pages 1329–1336, 2005.

[80] K. K. Thekumparampil, P. Jain, P. Netrapalli, and S. Oh. Efficient algorithms for smooth minimax optimization. In *Advances in Neural Information Processing Systems*, pages 12659–12670, 2019.

[81] P. Tseng. Accelerated proximal gradient methods for convex optimization. Technical report, University of Washington, Seattle, 2008. URL `https://www.mit.edu/~dimitrib/PTseng/papers/apgm.pdf`.

[82] R. Vinter and H. Zheng. Some finance problems solved with nonsmooth optimization techniques. *Journal of optimization theory and applications*, 119(1):1–18, 2003.

[83] Z. Wang, X. He, D. Gao, and X. Xue. An efficient kernel-based matrixized least squares support vector machine. *Neural Computing and Applications*, 22(1):143–150, 2013.

[84] D. White. Extension of the frank-wolfe algorithm to concave nondifferentiable objective functions. *Journal of optimization theory and applications*, 78(2):283–301, 1993.

[85] L. Wolf, H. Jhuang, and T. Hazan. Modeling appearances with low-rank svm. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6. IEEE, 2007.

[86] J. Xie, Z. Shen, C. Zhang, B. Wang, and H. Qian. Efficient projection-free online methods with stochastic recursive gradient. In *AAAI*, pages 6446–6453, 2020.

[87] T. Yang and Q. Lin. RSG: Beating subgradient method without smoothness and strong convexity. *The Journal of Machine Learning Research*, 19(1):236–268, 2018.

[88] T. Yang, Q. Lin, and L. Zhang. A richer theory of convex constrained optimization with reduced projections and improved rates. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3901–3910. JMLR. org, 2017.

[89] I. E.-H. Yen, X. Lin, J. Zhang, P. Ravikumar, and I. Dhillon. A convex atomic-norm approach to multiple sequence alignment and motif discovery. In *International Conference on Machine Learning*, pages 2272–2280, 2016.

[90] K. Yosida. *Functional analysis*. Springer Verlag, 1965.

[91] L. Zhang, T. Yang, R. Jin, and X. He. $O(\log t)$ projections for stochastic optimization of smooth and strongly convex functions. In *International Conference on Machine Learning*, pages 1121–1129, 2013.

[92] T. Zhang. Sequential greedy approximation for certain convex optimization problems. *IEEE Transactions on Information Theory*, 49(3):682–691, 2003.

[93] J. Zhu, S. Rosset, R. Tibshirani, and T. J. Hastie. 1-norm support vector machines. In *Advances in neural information processing systems*, pages 49–56, 2004.

Appendix

A Supplementary results

A.1 Intuition behind the design of MOPES and a failed attempt

In this section we study the main ideas behind the design of MOPES method through a failed attempt. Only for the section, for simplicity, we assume that $\mathcal{X}'$ is the whole vector space, and f is G Lipschitz in $\mathcal{X}'$. Recall that we want to solve the problem (5),

$$\min_{x\in\mathcal{X},x'\in\mathcal{X}'}[\Psi_\lambda(x,x') = \psi_\lambda(x,x') + f(x')]\,. \tag{9}$$

Notice that this is a composite objective which is a sum of a $2/\lambda$-smooth function ψ_λ and a nonsmooth function f. This implies that, if we have access to the proximal operator (recall Definition 3) for f

$$\mathrm{prox}_{f/t}(z) := \arg\min_{x\in\mathcal{X}'} f(x) + \frac{t}{2}\|x-z\|^2\,, \tag{10}$$

then theoretically we can solve this problem using accelerated proximal gradient algorithm (APGD) [8, 67, 81], which has the following update rule

$$\boxed{\begin{aligned} \beta_k &\leftarrow 4/\lambda k\ ,\ \gamma_k \leftarrow 2/(k+1) \\ (y_k,y'_k) &\leftarrow (1-\gamma_k)\,(x_{k-1},x'_{k-1}) + \gamma_k(z_{k-1},z'_{k-1}) \\ z_k &\leftarrow \mathcal{P}_\mathcal{X}\left(z_{k-1} - \nabla_x\psi_\lambda(y_k,y'_k)/\beta_k\right) \\ z'_k &\leftarrow \mathrm{prox}_{f/\beta_k}\left(z'_{k-1} - \nabla_{x'}\psi_\lambda(y_k,y'_k)/\beta_k\right) \\ (x_k,x'_k) &\leftarrow (1-\gamma_k)\,(x_{k-1},x'_{k-1}) + \gamma_k(z_k,z'_k) \end{aligned}}\ , \tag{APGD}$$

for some stepsize $1/\beta_k$ and iterate weight γ_k.

Notice that this update rule is different from the standard accelerated schemes, because the latter either first update the primal variables (x,x') and then extrapolate the dual variables (z,z') [65] or simultaneously update them both [69, 57], whereas (APGD), which is fashioned along the lines of [81], first updates (z,z') using proximal[3] step and then extrapolates these to update (x,x'). Advantage of [81] over the standard rule are three fold; former only needs one proximal step per variable (as opposed to two in [69, 57]) per iteration (which makes it practically faster), or keeps the dual and middle iterates z_k, y_k feasible (as opposed to [65, (2.2.17)]), and can easily handle stochastic FO and constraints [53]. Another reason for the choice, which will be evident later on, is that, our update rule can simultaneously provide the optimal complexity for the smooth ψ_λ and nonsmooth f parts of the composite function, Ψ_λ (5) [55].

With the right choice of β_k, γ_k, (APGD) can find an ε-approximate solution to the problem (5), (x_K,x'_K), in $\mathcal{O}(\sqrt{2/\lambda\varepsilon})$ steps. Now if we choose $\lambda = \mathcal{O}(\varepsilon)$, we can show that x_K is also an $\mathcal{O}(\varepsilon)$ solution of our original nonsmooth constrained problem (1). This is formalized in the Lemma 2. Thus applying (APGD) on (5) with $\lambda = \varepsilon/G^2$ gives us an ε solution to the original problem (1) using only $K = \mathcal{O}(G/\varepsilon)$ projections, which is a significant improvement over the $\mathcal{O}(G^2/\varepsilon^2)$ PO calls used by the standard subgradient method.

For a general G-Lipschitz convex function f, we cannot solve $\mathrm{prox}_{f/\beta_k}$ exactly, and hence we resort to some approximate solution. We emphasize here that it is not immediately evident that we can implement an inexact prox operator, and still maintain that the total number of FO calls used by this inexact APGD method match the optimal lowerbound $\mathcal{O}(G^2/\varepsilon^2)$ [64, 65]. Perhaps surprisingly, this is achieved using the Gradient Sliding method [55], which proposes a specific form of an inexact APGD. Note that the constrained variable $x \in \mathcal{X}$ is not an input to the nonsmooth part f of Ψ_λ, which means that approximately resolving the prox operator $\mathrm{prox}_{f/t}$ does not require any projection.

As an intermediate algorithm we first present (IAPGD), which is derived from (APGD) but replaces the proximal update of z'_k with an inexact resolution for prox operator $\mathrm{prox}_{f/\beta_k}$ up to an approximation error of δ. Notice that $\delta = 0$, implies that z_k is an exact resolution of the operation $\mathrm{prox}_{f/\beta_k}(\hat{z}'_k)$.

---

[3] projection $\mathcal{P}_\mathcal{X}$ could be considered as a proximal step for the 0-∞ indicator function for the set $\mathcal{X}$

The specific choice of the approximation error is important, as other notions of approximation error of the proximal operator in the context of APGD (such as those in [77]) do not explicitly control the distance $\|x' - z'_k\|^2$ which is crucial for our guarantee. Although with $\delta = \varepsilon$, (IAPGD) would require $\mathcal{O}(1/\varepsilon^3)$ FO calls, we provide the details of its analysis, in the next theorem, as it showcases some of the ideas behind the design of our our main algorithm (Algorithm 4).

Use the update rule of (APGD), but replace prox step by the following:
find z'_k satisfying the following for all $x' \in \mathcal{X}'$

$$\frac{\beta_k}{2}\|x' - z'_k\|^2 + f(z'_k) + \frac{\beta_k}{2}\|z'_k - \hat{z}'_k\|^2 \ \le\ f(x') + \frac{\beta_k}{2}\|x' - \hat{z}'_k\|^2 + \delta\ , \tag{IAPGD}$$

where $\hat{z}'_k = (z'_{k-1} - \nabla_{x'}\psi_\lambda(y_k, y'_k)/\beta_k)$

Theorem 3. *Let $f : \mathcal{X}' \to \mathbb{R}$ be a G-Lipschitz continuous convex function and $\mathcal{X} \subseteq \mathcal{X}'$ be any convex set with a diameter $D_\mathcal{X}$ and projection oracle $\mathcal{P}_\mathcal{X}$. If we choose $\lambda = \varepsilon/G^2$ and $\delta = \varepsilon$, then after $K = \mathcal{O}(GD_\mathcal{X}/\varepsilon)$ iterations of the IAPGD update rule, initialized with $y'_0 = y_0 = x'_0 = x_0$, finds $x_K \in \mathcal{X}$ satisfying $f(x_K) - \min_{x\in\mathcal{X}} f(x) \le \varepsilon$. Further, $\mathcal{O}(G^2D_\mathcal{X}^2/\varepsilon^2)$ iterations of a standard subgradient method ensures the condition in* (IAPGD)*. In total, this algorithm requires $\mathcal{O}(G^3D_\mathcal{X}^3/\varepsilon^3)$ FO calls and $\mathcal{O}(GD_\mathcal{X}/\varepsilon)$ PO calls.*

Remarks: Even though IAPGD only achieves a FO-CC of $\mathcal{O}(1/\varepsilon^3)$, the main take away from this result should be that with this right choice for approximate resolvent of the proxoperator $\mathrm{prox}_{f/\beta_k}$ (IAPGD), we can achieve $\mathcal{O}(1/\varepsilon)$ PO-CC. This is exploited by our MOPES method (Algorithm 1) which uses a more efficient `Prox-Slide` procedure [55] to approximately resolve the prox operator, so as to obtain a PO-CC of $\mathcal{O}(1/\varepsilon)$ while still maintaining the optimal FO-CC o $\mathcal{O}(1/\varepsilon^2)$.

Proof of Theorem 3. Now consider the following potential (Lyapunov) function from [6] for arbitrary $x \in \mathcal{X}$ and $x' \in \mathcal{X}'$:

$$\Phi_k := k(k+1)(\Psi_\lambda(x_k, x'_k) - \Psi_\lambda(x, x')) + (4/\lambda)\|(z_k, z'_k) - (x, x')\|^2 \tag{11}$$

We will prove that this potential satisfy the following approximate descent condition $\Phi_k \le \Phi_{k-1} + k\varepsilon$ as follows. Notice that by $2/\lambda$-smoothness and convexity of ψ_λ

$$\begin{aligned}
\psi_\lambda(x_k, x'_k) &\le \psi_\lambda(y_k, y'_k) + \langle \nabla_k, (x_k, x'_k) - (y_k, y'_k)\rangle + \frac{1}{\lambda}\|(x_k, x'_k) - (y_k, y'_k)\|^2 \\
&\le (1-\gamma_k)\psi_\lambda(x_{k-1}, x'_{k-1}) + \\
&\qquad \gamma_k[\psi_\lambda(y_k, y'_k) + \langle \nabla_k, (z_k, z'_k) - (y_k, y'_k)\rangle + \frac{\gamma_k}{\lambda}\|(z_k, z'_k) - (z_{k-1}, z'_{k-1})\|^2]
\end{aligned} \tag{12}$$

where we use the shorthand $\nabla_k := [\nabla_{k,x}^T \nabla_{k,x'}^T]^T := [\nabla_x\psi_\lambda(y_k, y'_k)^T\ \nabla_{x'}\psi_\lambda(y_k, y'_k)^T]^T$. Now combining this with $f(x'_k) \le (1-\gamma_k)f(x'_{k-1}) + \gamma_k f(z'_k)$ (convexity) and $\gamma_k/\lambda \le \beta_k/2$ we get that

$$\begin{aligned}
& k(k+1)\Psi_\lambda(x_k, x'_k) \\
&\le k(k-1)\Psi_\lambda(x_{k-1}, x'_{k-1}) + 2k\psi_\lambda(y_k, y'_k) + 2k[\langle \nabla_{k,x}, z_k - y_k\rangle + \frac{\beta_k}{2}\|z_k - z_{k-1}\|^2] \\
&\quad 2k[f(z'_k) + \langle \nabla_{k,x'}, z'_k - y'_k\rangle + \frac{\beta_k}{2}\|z'_k - z'_{k-1}\|^2] \\
&\le k(k-1)\Psi_\lambda(x_{k-1}, x'_{k-1}) + 2k\psi_\lambda(y_k, y'_k) + \\
&\quad 2k[\langle \nabla_{k,x}, x - y_k\rangle + \frac{\beta_k}{2}(\|z_{k-1} - x\|^2 - \|z_k - x\|^2)] \\
&\quad 2k[f(x') + \langle \nabla_{k,x'}, x' - y'_k\rangle + \frac{\beta_k}{2}(\|z'_{k-1} - x'\|^2 - \|z'_k - x'\|^2) + \varepsilon] \\
&\le k(k-1)\Psi_\lambda(x_{k-1}, x'_{k-1}) + 2k\Psi_\lambda(x, x') + \\
&\quad (4/\lambda)(\|(z_{k-1}, z'_{k-1}) - (x, x')\|^2 - \|(z_k, z'_k) - (x, x')\|^2) + k\varepsilon\,,
\end{aligned} \tag{13}$$

where the second inequality uses the definition of projection and the ε-approximate resolution of the proximal operator (IAPGD), and the last inequality again uses convexity of ψ_λ. This proves that

$\Phi_k \le \Phi_{k-1} + k\varepsilon$, which directly implies that

$$\Psi_\lambda(x_K, x'_K) - \Psi_\lambda(x, x') \le \frac{4(\|x_0 - x\|^2 + \|x_0 - x'\|^2)}{\lambda K(K+1)} + \frac{1}{K(K+1)} \sum_{k=1}^{K} k\varepsilon \tag{14}$$

Setting $x' = x$, choosing $\lambda = \varepsilon/G^2$ and $K = \mathcal{O}(GD_{\mathcal{X}}/\varepsilon)$ gives us $\Psi_\lambda(x_k, x'_k) - f(x) \le \varepsilon/2$. Then by Lemma 2 we get that $f(x_k) - \min_{x\in\mathcal{X}} f(x) \le \varepsilon$. For each inner problem the standard (unconstrained) proximal subgradient method applied on $\min_{x'\in\mathcal{X}'} f(x') + (\beta_k/2)\|x' - (z'_{k-1} - \nabla_{k,x'}/\beta_k)\|^2$, initialized with x_0 (for ease of argument), can achieve this error using $\mathcal{O}(G^2\|x_0 - \widehat{x}_\lambda(x)\|^2/\varepsilon^2) = \mathcal{O}(G^2 D_{\mathcal{X}}^2/\varepsilon^2)$ FO calls (Lemma 3, in Appendix B.1). Thus the algorithm uses totally $\mathcal{O}(GD_{\mathcal{X}}/\varepsilon)$ projections and $\mathcal{O}(G^3 D_{\mathcal{X}}^3/\varepsilon^3)$ subgradients. □

A.2 A proof sketch for Theorem 1 (MOPES)

This section provides a short proof sketch for the Theorem 1—guarantee for the MOPES (Algorithm 1) method—to showcase the main analysis techniques used by the full proof in Appendix C.2.1. At a high level, our proof uses a potential function [6] for analyzing APGD, combines it with Proposition 3 which provides a fast convergence guarantee on `Prox-Slide` iterates, and then apply standard APGD proof techniques [81] to obtain the final result.

Proof sketch. Here we only consider the deterministic FO. We define the following potential (Lyapunov) function for some arbitrary $x \in \mathcal{X}$, by slightly modifying the standard AGD potential [6].

$$\Phi_k := k(k+1)(\Psi_\lambda(x_k, x'_k) - \Psi_\lambda(x, x)) + (4/\lambda)(\|z_k - x\|^2 + \tau_{k+1}\|z'_k - x\|^2) \tag{15}$$

where $\tau_k := (T_k + 1)(T_k + 2)/T_k(T_k + 3)$. We will prove that this potential satisfies the descent rule: $\Phi_k \le \Phi_{k-1} + k\eta'_k$, for some error η'_k. Using the fact that Ψ_λ is a sum of two convex functions: $2/\lambda$-smooth quadratic ψ_λ and G-Lipschitz f, and standard analysis techniques for AGD we can get

$$\begin{aligned} k(k+1)\Psi_\lambda(x_k, x'_k) \le\ & k(k-1)\Psi_\lambda(x_{k-1}, x'_{k-1}) + 2k\psi_\lambda(x, x) + \\ & 2k[\langle \nabla_{k,x}, z_k\rangle + (\beta_k/2)\|z_k - z_{k-1}\|^2] + 2k[\phi_k(\widetilde{z}'_k) - \phi_k(x) + (\beta_k/2)\|x - z'_{k-1}\|^2] \end{aligned} \tag{16}$$

where we use the short-hands $\nabla_k := \nabla\psi_\lambda(y_k, y'_k)$ and $\phi_k(x') := f(x') + \langle \nabla_{k,x'}, x'\rangle + (\beta_k/2)\|x' - z'_{k-1}\|^2$. Next, using definition of projection z_k, we bound the third term in the RHS of (16) as

$$2k[\langle \nabla_{k,x}, x - z_k\rangle + (\beta_k/2)\|z_k - z_{k-1}\|^2] \le 2k(\beta_k/2)[\|z_{k-1} - x\|^2 - \|z_k - x\|^2]. \tag{17}$$

The fourth term in the RHS of (16) corresponds to the ε-approximate resolution of the $\operatorname{prox}_{f/\beta_k}$ operator through the `Prox-Slide` procedure (Line 1.5), whose output satisfies the following guarantee.

Proposition 2 (informal version of Proposition 3)**.** *Output of `Prox-Slide` satisfies*

$$\phi_k(\widetilde{z}'_k) - \phi_k(x) + \frac{\beta_k}{2}\|z'_k - x\|^2 \le \frac{\beta_k}{2}(\tau_k - 1)[\|z'_{k-1} - x\|^2 - \|z'_k - x\|^2] + \frac{16\,G^2}{\beta_k T_k}.$$

The above lemma guarantees the optimal $O(1/T_k)$ convergence rate for the strongly convex minimization problem: $\min_{z'\in\mathcal{X}'} \phi_k(z')$, corresponding to the proximal operator. By combining the inequalities (16) and (17) and the proposition we get: $\Phi_k \le \Phi_{k-1} + k\mathcal{O}(G^2/\beta_k T_k)$. Now, using Lemma 2, and setting $x = x^*$, $\lambda = \frac{\varepsilon}{G^2}$, $\beta_k = \frac{4}{\lambda k}$, $T_k = \mathcal{O}(k)$ and $K = \Theta(\frac{G\|x_0 - x^*\|}{\varepsilon})$ we get

$$\begin{aligned} f(x_K) - f(x^*) &\le \Psi_\lambda(x_K, x'_K) - \Psi_\lambda(x^*, x^*) + G^2\frac{\lambda}{2} \\ &\le \frac{8\|x_0 - x^*\|^2}{\lambda K(K+1)} + \frac{\sum_{k=1}^{K} k\,16G^2/\beta_k T_k}{\lambda K(K+1)} + G^2\frac{\lambda}{2} = \mathcal{O}(\varepsilon) \end{aligned}$$

Therefore, the total number of PO calls made is $K = \mathcal{O}(G\|x_0 - x^*\|/\varepsilon)$ and the total number of FO calls made is $\sum_{k=1}^{K} T_k = \mathcal{O}(K^2) = \mathcal{O}(G^2\|x_0 - x^*\|^2/\varepsilon^2)$. □

B Supporting results

B.1 Proximal Subgradient method

Lemma 3 (proximal subgradient descent)**.** *Consider the regularized optimization problem*

$$\min_u [f_{\beta,x}(u) := f(u) + (\beta/2)\|u-x\|^2] \tag{18}$$

and the proximal subgradient method's update rule

$$\begin{aligned} u_{t+1} &= \operatorname*{argmin}_u [F_t(u) := \langle g_t, u-x\rangle + (1/2\eta)\|u-u_t\|^2 + \beta/2\|u-x\|^2] \\ &= u_t - (\eta/(1+\eta\beta))(g_t + \beta(u_t - x)) \end{aligned} \tag{19}$$

where $g_t \in \partial f(u_t)$ and η is the effective stepsize. Now, if $\eta = 2\,G^2\|u_0-u\|/\sqrt{T}$ and $\widetilde{u}_T = \frac{1}{T}\sum_{t=0}^{T-1} u_{t+1}$, then for any u

$$\frac{\beta}{2}\|\widetilde{u}_T - u\|^2 + f_{\beta,x}(\widetilde{u}_T) - f_{\beta,x}(u) \le \frac{2\,G\,\|u_0-u\|}{\sqrt{T}} \tag{20}$$

Proof. Let u be an arbitrary feasible point. By convexity and G-Lipschitzness of f,

$$\begin{aligned} f(u_{t+1}) - f(u) &= f(u_{t+1}) - f(u_t) + f(u_t) - f(u) \\ &\le \langle g_{t+1}, u_{t+1} - u_t\rangle + \langle g_t, u_t - u\rangle \\ &= \langle g_t, u_{t+1} - u_t\rangle + \langle g_{t+1} - g_t, u_{t+1} - u_t\rangle + \langle g_t, u_t - u\rangle \\ &\le \langle g_t, u_{t+1} - u\rangle + 2G\,\|u_{t+1} - u_t\|\,, \end{aligned} \tag{21}$$

As u_{t+1} is the minimizer of a $(\beta + 1/\eta)$-strong convexity update objective F_t and since, we get that

$$(\frac{\beta}{2} + \frac{1}{2\eta})\|u_{t+1} - u\|^2 + F_t(u_{t+1}) \le F_t(u) \tag{22}$$

Now summing up (21), sand (22) we get

$$\begin{aligned} \frac{\beta}{2}\|u_{t+1} - u\|^2 + f_{\beta,x}(u_{t+1}) - f_{\beta,x}(u) &\le \frac{1}{2\eta}(\|u_t - u\|^2 - \|u_{t+1} - u\|^2) + \\ &\qquad 2G\,\|u_{t+1} - u_t\| - \frac{1}{2\eta}\|u_{t+1} - u_t\|^2 \\ &\le \frac{1}{2\eta}(\|u_t - u\|^2 - \|u_{t+1} - u\|^2) + 2\,G^2\eta \\ \implies \frac{1}{T}\sum_{t=0}^{T-1}\frac{\beta}{2}\|u_{t+1} - u\|^2 + f_{\beta,x}(u_{t+1}) - f_{\beta,x}(u) &\le \frac{1}{2\eta T}(\|u_0 - u\|^2 - \|u_T - u\|^2) + 2\,G^2\eta \\ \frac{\beta}{2}\|\widetilde{u}_T - u\|^2 + f_{\beta,x}(\widetilde{u}_T) - f_{\beta,x}(u) &\le \end{aligned} \tag{23}$$

where the second inequality follows from $ax - x^2/2b \le a^2b/2$, the third inequality is obtained by summing over $t = 0, \ldots, T-1$, and the third inequality uses Jensen's inequality. Choosing $T = 2\,G\,\|u_0 - u\|/\sqrt{T}$, we get the desired result

$$\frac{\beta}{2}\|\widetilde{u}_T - u\|^2 + f_{\beta,x}(\widetilde{u}_T) - f_{\beta,x}(u) \le \frac{2\,G\,\|u_0-u\|}{\sqrt{T}} \tag{24}$$

□

B.2 Frank-Wolfe projected subgradient method FW-PGD (Algorithm 3)

Here we provide the details of the Frank-Wolfe based projected subgradient method (Algorithm 3) used in the experiments. The main idea is to use some competitive LMO based method to approximate the projection step in the standard projected subgradient method. The following theorem gives some guarantees for the output of the Algorithm 3.

```
Algorithm 3: Frank-Wolfe projected subgradient method using LMO
Input: $f$, $\mathcal{X}$, $G$, $D_{\mathcal{X}}$, $x_0$, $K$,
for $k = 0, \ldots, K-1$ do
    Set $\widehat{g}_k = \mathrm{SFO}\left(x_k\right)$
    Using any competitive LMO based algorithm (e.g. Frank-Wolfe method [28] or CndG
      procedure [56, Algo. 1]), approximately solve the projection problem
        $$x_{k+1} \approx \underset{x \in \mathcal{X}}{\operatorname{argmin}} \left\langle \widehat{g}_k, x \right\rangle + \frac{1}{2\alpha_k}\|x - x_k\|^2 = \underset{x \in \mathcal{X}}{\operatorname{argmin}} \frac{1}{2\alpha_k}\|x - (x_k - \alpha_k \cdot \widehat{g}_k)\|^2 , \quad (25)$$
      ensuring that the Wolfe duality gap [45] of the above problem at $u_\Pi$ satisfies
        $$x \max_{s \in \mathcal{X}} \left\langle \widehat{g}_k + 1/\alpha_k \left(x_{k+1} - x_k\right), x_{k+1} - s \right\rangle \leq \eta_k \quad (26)$$
Output: $\bar{x}_K = \frac{\sum_{k=0}^{K-1} \alpha_k x_k}{\sum_{k=0}^{K-1} \alpha_k}$
```

Theorem 4. *Let $f : \mathcal{X}' \to \mathbb{R}$ be a G-Lipschitz continuous proper l.s.c. convex function, and $\mathcal{X} \subseteq \mathcal{X}'$ be some closed convex subset of $\mathbb{R}^d$ with diameter $D_{\mathcal{X}}$. Then after K iterations, the Algorithm 3 projection tolerance $\eta_k = (G^2 + \sigma^2)\alpha_k$, stepsize $\alpha_k = \frac{D_{\mathcal{X}}}{2\sqrt{G^2+\sigma^2}\sqrt{K}}$ and outputs $\bar{x}_K \in \mathcal{X}$ satisfying*

$$\mathbb{E}[f\left(\bar{x}_K\right)] - f(x^*) \leq \frac{2\sqrt{G^2 + \sigma^2} D_{\mathcal{X}}}{\sqrt{K}} \tag{27}$$

Further, the algorithm uses K SFO calls and $O(K^2)$ LMO calls.

Proof. Using the Wolfe duality gap guarantee we get that for any $x \in \mathcal{X}$

$$\left\langle \widehat{g}_k + \frac{1}{\alpha_k}\left(x_{k+1} - x_k\right), x_{k+1} - x \right\rangle \leq \eta_k \, . \tag{28}$$

By rearranging the terms above we get that

$$\begin{aligned}
\left\langle \widehat{g}_k, x_k - x \right\rangle &\leq \frac{1}{2\alpha_k}(\|x_k - x\|^2 - \|x_{k+1} - x\|^2) - \frac{1}{2\alpha_k}\|x_{k+1} - x_k\|^2 + \left\langle \widehat{g}_k, x_k - x_{k+1} \right\rangle + \eta_k \\
&\leq \frac{1}{2\alpha_k}(\|x_k - x\|^2 - \|x_{k+1} - x\|^2) - \frac{1}{2\alpha_k}\|x_{k+1} - x_k\|^2 + \|\widehat{g}_k\|\|x_k - x_{k+1}\| + \eta_k \\
&\leq \frac{1}{2\alpha_k}(\|x_k - x\|^2 - \|x_{k+1} - x\|^2) + \frac{\alpha_k}{2}\|\widehat{g}_k\|^2 + \eta_k \, ,
\end{aligned} \tag{29}$$

where the last inequality uses the fact that $-(a/2)z^2 + bz \leq b^2/2a$ for all $a, b, z \in \mathbb{R}$. Next, multiplying by α_k and summing the above inequality over $k = 0, \ldots, K-1$ and dividing by $\sum_{k'=0}^{K-1} \alpha_{k'}$ we get

$$\sum_{k=0}^{K-1} \alpha_k \left\langle \widehat{g}_k, x_k - x \right\rangle \leq \frac{1}{2}(\|x_0 - x\|^2 - \|x_K - x\|^2) + \sum_{k=0}^{K-1} \alpha_k^2 (\frac{\|\widehat{g}_k\|^2}{2} + \frac{\eta_k}{\alpha_k}) \, , \tag{30}$$

Now taking expectation w.r.t. all the stochasticity in $\{\widehat{g}_k\}_{k=0}^{K-1}$ on both sides and using, the towering conditional expectation property $\mathbb{E}[a] = \mathbb{E}[\mathbb{E}[a \,|\, x_k]]$, and $\mathbb{E}[\widehat{g}_k \,|\, x_k] = g_k \in \partial f(x_k)$ and $\mathbb{E}[\|\widehat{g}_k\|^2 \,|\, x_k] \leq 2(G^2 + \sigma^2)$ we get

$$\sum_{k=0}^{K-1} \alpha_k \mathbb{E}[\left\langle g_k, x_k - x \right\rangle] \leq \frac{1}{2}\|x_0 - x\|^2 + \sum_{k=0}^{K-1} \alpha_k^2 ((G^2 + \sigma^2) + \frac{\eta_k}{\alpha_k}) \, , \tag{31}$$

Next diving by $\sum_{k'=0}^{K-1} \alpha_{k'}$, using convex affine lower bound of f at x_k and Jensen's inequality we get

$$\begin{aligned}\sum_{k=0}^{K-1} \frac{\alpha_k}{\sum_{k'=0}^{K-1} \alpha_{k'}} \mathbb{E}[f(x_k) - f(x)] &\le \frac{\frac{1}{2}\|x_0 - x\|^2 + \sum_{k=0}^{K-1} \alpha_k^2((G^2+\sigma^2) + \frac{\eta_k}{\alpha_k})}{\sum_{k'=0}^{K-1} \alpha_{k'}} \\ \mathbb{E}\bigg[f\bigg(\sum_{k=0}^{K-1} \frac{\alpha_k \cdot x_k}{\sum_{k'=0}^{K-1} \alpha_{k'}}\bigg)\bigg] - f(x) &\le \end{aligned} \tag{32}$$

Next if we choose $\eta_k = \alpha_k(G^2+\sigma^2)$, and set $x = x^* \in \operatorname{argmin}_{x' \in \mathcal{X}} f(x')$ and $\alpha_k = \frac{D_{\mathcal{X}}}{2\sqrt{G^2+\sigma^2}\sqrt{K}}$ we get

$$\begin{aligned}\mathbb{E}\bigg[f\bigg(\sum_{k=0}^{K-1} \frac{\alpha_k \cdot x_k}{\sum_{k'=0}^{K-1} \alpha_{k'}}\bigg)\bigg] - f(x^*) &\le \frac{\frac{1}{2}D_{\mathcal{X}}^2 + \sum_{k=0}^{K-1} \alpha_k^2 \cdot 2(G^2+\sigma^2)}{\sum_{k'=0}^{K-1} \alpha_{k'}} \\ &= \frac{2\sqrt{G^2+\sigma^2}D_{\mathcal{X}}}{\sqrt{K}} \end{aligned} \tag{33}$$

Clearly the algorithm uses K SFO calls. At step k when approximating the projection using an LMO based method, after using $\hat{T}_k = \lceil \frac{7D_{\mathcal{X}}^2}{\alpha_k^2(G^2+\sigma^2)} \rceil$ LMO calls in the `Approx-Proj` procedure, the Wolfe duality gap (38) is at most $\lceil \frac{6(1/\alpha_k)D_{\mathcal{X}}^2}{\hat{T}_k} \rceil \le \alpha_k(G^2+\sigma^2)$ if we use CndG procedure [56, Theorem 2.2(c)] or $\lceil \frac{7(1/\alpha_k)D_{\mathcal{X}}^2}{\hat{T}_k} \rceil \le \alpha_k(G^2+\sigma^2)$ if we use the standard Frank-Wolfe algorithm [45, Theorem 2]. Therefore the total number of linear minimization oracle calls made by the algorithm is

$$\sum_{k=0}^{K-1} \hat{T}_k = \sum_{k=0}^{K-1} \frac{7D_{\mathcal{X}}^2}{\alpha_k^2(G^2+\sigma^2)} + K = 28K^2 + K = O(K^2) \tag{34}$$

where we use the given choice for $\alpha_k = \frac{D_{\mathcal{X}}}{2\sqrt{G^2+\sigma^2}\sqrt{K}}$. □

C Proofs of the main results

C.1 Proof of Lemma 2

Proof. First we prove part (i). By definitions of Ψ_λ (5) and f_λ (Definition 3), we have $\min_{x\in\mathcal{X}} \min_{x\in\mathcal{X}'} \Psi_\lambda(x, x') = \min_{x\in\mathcal{X}} f_\lambda(x)$. By Lemma 1(a), we also can show that $\min_{x\in\mathcal{X}} f_\lambda(x) \le \min_{x\in\mathcal{X}} f(x)$.

For part (ii), first we show the following.

$$\begin{aligned}\mathbb{E}f_\lambda(x_\varepsilon) = \mathbb{E}\Psi_\lambda(x_\varepsilon, \widehat{x}_\lambda(x_\varepsilon)) = \mathbb{E}\min_{x'(x_\varepsilon)} \Psi_\lambda(x_\varepsilon, x'(x_\varepsilon)) &\le \mathbb{E}_{\bar{x}_K} \Psi_\lambda(x_\varepsilon, \mathbb{E}_{x'_\varepsilon|x_\varepsilon} x'_\varepsilon) \\ &\le \mathbb{E}_{x_\varepsilon}\mathbb{E}_{x'_\varepsilon|x_\varepsilon} \Psi_\lambda(x_\varepsilon, x'_\varepsilon) \\ &= \mathbb{E}\Psi_\lambda(x_\varepsilon, x'_\varepsilon)\end{aligned} \tag{35}$$

Finally, combining the above inequality with Lemma 1(c) we get the desired result

$$\mathbb{E}f(x_\varepsilon) \le \mathbb{E}f_\lambda(x_\varepsilon) + G^2\lambda/2 \le \mathbb{E}\Psi_\lambda(x_\varepsilon, x'_\varepsilon) + G^2\lambda/2 \tag{36}$$

□

C.2 Analysis of MOPES (Algorithm 1) and MOLES (Algorithm 2) method

Instead of separately analyzing MOPES and MOLES, we first analyze a more general algorithm, Algorithm 4, which has the following guarantee.

Theorem 5. *Let $f : \mathcal{X}' \to \mathbb{R}$ be a G-Lipschitz continuous proper l.s.c. convex function, and $\mathcal{X} \subseteq \mathcal{X}' = B(0, R)$ be some convex subset contained inside the Euclidean ball of radius R around origin. Then after K iterations, the Algorithm 4 outputs $x_K \in \mathcal{X}$ satisfying*

$$\mathbb{E}[f(x_K)] - f(x^*) \le \frac{10\|x_0 - x^*\|^2 + 8\tilde{D}}{\lambda K(K+1)} + \frac{\sum_{k=1}^{K} 2k\,\eta_k}{K(K+1)} + G^2\frac{\lambda}{2} \tag{39}$$

for any choice of $\lambda > 0$, $\tilde{D} > 0$, and tolerance $\{\eta_k\}_{k\in[K]}$ (38).

Algorithm 4: Moreau subgradient method for nonsmooth convex optimization using PO or LMO

```
Input: $f$, $\mathcal{X}$, $\mathcal{X}'$, $G$, $D_{\mathcal{X}}$, $R$, $x_0$, $K$, $\tilde{D}$, $\lambda$, $\{\eta_k \in \mathbb{R}_+\}_{k\in[K]}$
Set $x'_0 = z'_0 = x_0 = z_0 = x_0$
for $k = 1, \ldots, K$ do
    Set $\lambda_k = \lambda$, $\beta_k = \frac{4}{\lambda_k k}$ , $\gamma_k = \frac{2}{k+1}$ , and $T_k = \left\lceil \frac{(4G^2+\sigma^2)\lambda^2 K k^2}{2\tilde{D}} \right\rceil$
    Set $(y_k, y'_k) = (1-\gamma_k)\cdot(x_{k-1}, x'_{k-1}) + \gamma_k \cdot (z_{k-1}, z'_{k-1})$
    Set $z_k =$ Approx-Proj $(\nabla_{y_k}\Psi_\lambda(y_k, y'_k), z_{k-1}, \beta_k, \eta_k)$ // Note $\nabla_{y_k}\Psi_\lambda(y_k, y'_k) = \frac{y_k - y'_k}{\lambda}$
    Set $(z'_k, \tilde{z}'_k) =$ Prox-Slide$\big(\nabla_{y'_k}\psi_\lambda(y_k, y'_k), z'_{k-1}, \beta_k, T_k\big)$   // $\nabla_{y'_k}\psi_\lambda(y_k, y'_k) = \frac{y'_k - y_k}{\lambda}$
    Set $(x_k, x'_k) = (1-\gamma_k)\cdot(x_{k-1}, x'_{k-1}) + \gamma_k \cdot (z_k, \tilde{z}'_k)$
Output: $(x_K, x'_K)$

Approx-Proj($g$, $u_0$, $\beta$, $\eta$) // Approx. resolve $\mathcal{P}_{\mathcal{X}}(u_0 - g/\beta)$[56]:
    Either using exact PO, $\mathcal{P}_{\mathcal{X}}$ (1) , or using any competitive LMO based algorithm
     (e.g. Frank-Wolfe method [28] or CndG procedure [56, Algo. 1]), approximately solve the
     projection problem

        $u_\Pi \approx \underset{u\in\mathcal{X}}{\operatorname{argmin}} \langle g, u\rangle + \frac{\beta}{2}\|u - u_0\|^2 = \underset{u\in\mathcal{X}}{\operatorname{argmin}} \frac{\beta}{2}\|u - (u_0 - g/\beta)\|^2 ,$   (37)

     ensuring that the Wolfe duality gap [45] of the above problem at $u_\Pi$ satisfies

        $\max_{s\in\mathcal{X}} \langle g + \beta(u_\Pi - u_0), u_\Pi - s\rangle \leq \eta_k$   (38)

     return $u_\Pi$

Prox-Slide($g$, $u_0$, $\beta$, $T$) // Approx. resolve $\operatorname{prox}_{f/\beta}(u_0 - g/\beta)$[55]:
    Set $\tilde{u}_0 = u_0$
    for $t = 1, \ldots, T$ do
        Set $\theta_t = \frac{2(t+1)}{t(t+3)}$, $\widehat{g}_{t-1} =$ SFO $(u_{t-1})$ (3)
        Set $\widehat{u}_t = u_{t-1} - \frac{1}{(1+t/2)\beta}\cdot(\widehat{g}_{t-1} + \beta(u_{t-1} - (u_0 - g/\beta)))$
         // subgradient method step for $\phi(u) := f(u) + \frac{\beta}{2}\|u - \big(u_0 - \frac{g}{\beta}\big)\|^2$
        Set $u_t = \widehat{u}_t \cdot \min(1, R/\|u_t\|)$   // projection of $\widehat{u}_t$ onto $\mathcal{X}'$:  $\mathcal{P}'_{\mathcal{X}}(\mathbf{u}_t)$
        Set $\tilde{u}_t = \big(1-\theta_t\big)\cdot\tilde{u}_{t-1} + \theta_t \cdot u_t$
    return $(u_T, \tilde{u}_T)$
```

Remarks: Before providing a proof for the above result we discuss some its implications. MOPES makes K PO calls, one per outer step, and $\sum_{k=1}^{K} T_k = \mathcal{O}(\lambda^2 K^4)$ SFO calls, one per inner step. The above analysis shows that we need to choose $\lambda = \varepsilon/G^2$, which is expected from Lemma 1. Since PO returns exact projections, the second term is zero with $\eta_k = 0$. The target accuracy of ε is achieved by tuning the first term, where we need to choose $K = \Theta(1/\sqrt{\lambda\varepsilon})$. Put together, this gives the desired $\mathcal{O}(\varepsilon^{-1})$ PO-CC and $\mathcal{O}(\varepsilon^{-2})$ SFO-CC for MOPES. A complete proof is provided in Section C.2.1.

When we have inexact projections in MOLES, we need $\eta_k = \Theta(1/k)$ to ensure that the second term is $\mathcal{O}(\varepsilon)$. At (outer) iteration k, this uses $\hat{T} = \Omega(K)$ iterations of Frank-Wolfe algorithm in `FW-Based-Projection` of Algorithm 2. MOLES makes $\sum_{k=1}^{K} \mathcal{O}(K) = \mathcal{O}(K^2)$ LMO calls, resulting in $\mathcal{O}(\varepsilon^{-2})$ LMO-CC as $K = \mathcal{O}(1/\sqrt{\lambda\varepsilon}) = \mathcal{O}(1/\varepsilon)$. A complete proof is in Section C.2.2.

Proof of Theorem 5. We define the following potential (Lyapunov) function, for some arbitrary $x \in \mathcal{X}$, $x' \in \mathcal{X}'$:

$$\Phi_k := k(k+1)(\Psi_\lambda(x_k, x'_k) - \Psi_\lambda(x, x')) + \frac{4}{\lambda}(\|z_k - x\|^2 + \frac{(T_{k+1}+1)(T_{k+1}+2)}{T_{k+1}(T_{k+1}+3)}\|z'_k - x'\|^2) \tag{40}$$

This is a slightly modified version of the following potential function for the standard AGD setting with a $2/\lambda$-smooth function Ψ_λ [6]: $k(k+1)(\Psi_\lambda(x_k, x'_k) - \Psi_\lambda(x, x')) + \frac{4}{\lambda}(\|z_k - x\|^2 + \|z'_k - x'\|^2)$. Notice that the modification factor

$$\frac{(T_{k+1}+1)(T_{k+1}+2)}{T_{k+1}(T_{k+1}+3)} \le \frac{3}{2} = \mathcal{O}(1) \tag{41}$$

is upper-bounded by a constant when $1 \le T_k$. Below we prove that this potential satisfies the approximate descent guarantee: $\Phi_k \le \Phi_{k-1} + k\eta_k + k\eta'_k$, for some error η'_k. First, notice that by $2/\lambda$-smoothness and convexity of ψ_λ

$$\begin{aligned}
\psi_\lambda(x_k, x'_k) &\le \psi_\lambda(y_k, y'_k) + \langle \nabla_k, (x_k, x'_k) - (y_k, y'_k)\rangle + \frac{1}{\lambda}\|(x_k, x'_k) - (y_k, y'_k)\|^2 \\
&= (1-\gamma_k)[\psi_\lambda(y_k, y'_k) + \big\langle \nabla_k, (x_{k-1}, x'_{k-1}) - (y_k, y'_k)\big\rangle] \\
&\qquad \gamma_k[\psi_\lambda(y_k, y'_k) + \langle \nabla_k, (z_k, \tilde{z}'_k) - (y_k, y'_k)\rangle + \frac{\gamma_k}{\lambda}\|(z_k, \tilde{z}'_k) - (z_{k-1}, z'_{k-1})\|^2] \\
&\le (1-\gamma_k)\psi_\lambda(x_{k-1}, x'_{k-1}) + \\
&\qquad \gamma_k[\psi_\lambda(y_k, y'_k) + \langle \nabla_k, (z_k, \tilde{z}'_k) - (y_k, y'_k)\rangle + \frac{\gamma_k}{\lambda}\|(z_k, \tilde{z}'_k) - (z_{k-1}, z'_{k-1})\|^2]
\end{aligned} \tag{42}$$

where we use the shorthand $\nabla_k := [\nabla_{k,x}^T \nabla_{k,x'}^T]^T := [\nabla_x \psi_\lambda(y_k, y'_k)^T \ \nabla_{x'}\psi_\lambda(y_k, y'_k)^T]^T$, and the second inequality uses Lines 4.4 and 4.7. Now combining this with $f(x'_k) \le (1-\gamma_k)f(x'_{k-1}) + \gamma_k f(\tilde{z}'_k)$ (using convexity of f and Line 4.7) and $\gamma_k/\lambda = 2/(\lambda(k+1)) \le 2/(\lambda k) = \beta_k/2$ (using Line 4.3), and multiplying it with $k(k+1)$ we get that

$$\begin{aligned}
&k(k+1)\Psi_\lambda(x_k, x'_k) \\
&\le k(k-1)\Psi_\lambda(x_{k-1}, x'_{k-1}) + 2k\psi_\lambda(y_k, y'_k) + 2k[\langle \nabla_{k,x}, z_k - y_k\rangle + \frac{\beta_k}{2}\|z_k - z_{k-1}\|^2] \\
&\quad 2k[f(\tilde{z}'_k) + \langle \nabla_{k,x'}, \tilde{z}'_k - y'_k\rangle + \frac{\beta_k}{2}\|\tilde{z}'_k - z'_{k-1}\|^2] \\
&= k(k-1)\Psi_\lambda(x_{k-1}, x'_{k-1}) + 2k\psi_\lambda(y_k, y'_k) + 2k[\langle \nabla_{k,x}, z_k - y_k\rangle + \frac{\beta_k}{2}\|z_k - z_{k-1}\|^2] \\
&\quad 2k[\phi_k(\tilde{z}'_k) - \phi_k(x') + \frac{\beta_k}{2}\|x' - z'_{k-1}\|^2 + f(x') + \langle \nabla_{k,x'}, x' - y'_k\rangle]
\end{aligned} \tag{43}$$

where for brevity we use the notation

$$\phi_k(x') := f(x') + \langle \nabla_{k,x'}, x'\rangle + \frac{\beta_k}{2}\left\|x' - z'_{k-1}\right\|^2 . \tag{44}$$

Now using the approximate optimality of z_k through the bound on the Wolfe dual gap (38) we get,

$$\begin{aligned}
\frac{\beta_k}{2}\left\|z_k - z_{k-1}\right\|^2 &= \frac{\beta_k}{2}\left\|z_{k-1} - x\right\|^2 + \beta_k\left\langle z_k - z_{k-1}, z_k - x\right\rangle - \frac{\beta_k}{2}\left\|z_k - x\right\|^2 \\
&\le \frac{\beta_k}{2}\left\|z_{k-1} - x\right\|^2 + \langle \nabla_{k,x}, x - z_k\rangle + \eta_k - \frac{\beta_k}{2}\left\|z_k - x\right\|^2 .
\end{aligned} \tag{45}$$

When z_k is the exact projection (as in Line 1.5) then the above inequality is satisfied with above $\eta_k = 0$. Otherwise, with the LMO oracle we will later set $\eta_k = \mathcal{O}(\varepsilon)$. Next we state the following lemma which provides a guarantee for the `Prox-Slide` procedure [55]. Here for rigorousness, we denote the iterates of the `Prox-Slide` procedure (Line 4.10) called at the outer step k, with $\{u_{k,t}\}_t$. Similarly at the outer step k, we denote the stochastic subgradients used by the `Prox-Slide` procedure and the corresponding subgradient with $\{\widehat{g}_{k,t}\}_t$ and $\{g_{k,t}\}_t$, i.e. $g_{k,t} := \mathbb{E}[\widehat{g}_{k,t} | u_{k,t}] \in \partial f(u_{k,t})$ for all k and t. A proof for this lemma is provided in Section C.2.3.

Proposition 3 ([55, Similar to Proposition 1])**.** *Let ϕ_k (44) be the minimization objective solved by `Prox-Slide` procedure at step k of Algorithm 4. Then $(z'_k, \tilde{z}'_k)$ obtained after T_k iterations of the procedure satisfy the following for any $x' \in \mathcal{X}'$,*

$$\begin{aligned}
\phi_k(\tilde{z}'_k) - \phi_k(x') \le \frac{2}{T_k(T_k+3)}\frac{\beta_k}{2}\|z'_{k-1} - x'\|^2 - \frac{(T_k+1)(T_k+2)}{T_k(T_k+3)}\frac{\beta_k}{2}\|z'_k - x'\|^2 + \\
\frac{4\sum_{t=0}^{T_k-1}(2G + \|\delta_{k,t}\|)^2}{\beta_k T_k(T_k+3)} + \sum_{t=0}^{T_k-1}\frac{2(t+2)}{T_k(T_k+3)}\left\langle \delta_{k,t}, x' - u_{k,t}\right\rangle
\end{aligned} \tag{46}$$

where $\delta_{k,t} := \widehat{g}_{k,t} - g_{k,t}$ and $u_{k,t}$ are private inner variable of the `Prox-Slide` procedure.

Aside: Note that `Prox-Slide` procedure essentially applies T_k steps of the proximal standard subgradient method to the ϕ_k (44), which is a composite function of a G-Lipschitz function f and prox-friendly β_k-strongly convex quadratic. Finally the procedure outputs the average of its iterate $\widetilde{z}'_k$ and its last iterate z'_k. In the end we will set $T_k = \Theta(1/\varepsilon)$ and $K = \Theta(1/\varepsilon)$ so that total number of subgradients used by the algorithm be $\sum_{k=1}^{K} T_k = \mathcal{O}(1/\varepsilon^2)$.

Now substituting (45) and Proposition 3 into (43) we get

$$\begin{aligned} k(k+1)\Psi_\lambda(x_k, x'_k) \le\ & k(k-1)\Psi_\lambda(x_{k-1}, x'_{k-1}) + 2k\psi_\lambda(y_k, y'_k) + \\ & 2k[\langle \nabla_{k,x}, x - y_k\rangle + \frac{\beta_k}{2}(\|z_{k-1} - x\|^2 - \|z_k - x\|^2) + \eta_k] \\ & 2k[f(x') + \langle \nabla_{k,x'}, x' - y'_k\rangle] + \\ & 2k[\frac{(T_k+1)(T_k+2)}{T_k(T_k+3)}\frac{\beta_k}{2}\|z'_{k-1} - x'\|^2 - \frac{(T_k+1)(T_k+2)}{T_k(T_k+3)}\frac{\beta_k}{2}\|z'_k - u\|^2 + \eta'_k] \\ \le\ & k(k-1)\Psi_\lambda(x_{k-1}, x'_{k-1}) + 2k\Psi_\lambda(x, x') + 2k(\eta_k + \eta'_k) \\ & \frac{4}{\lambda}(\|z_{k-1} - x\|^2 - \|z_k - x\|^2) \\ & \frac{4}{\lambda}(\frac{(T_k+1)(T_k+2)}{T_k(T_k+3)}\|z'_{k-1} - x'\|^2 - \frac{(T_{k+1}+1)(T_{k+1}+2)}{T_{k+1}(T_{k+1}+3)}\|z'_k - x'\|^2)\,, \end{aligned} \tag{47}$$

where we use the shorthand

$$\eta'_k := \frac{4\sum_{t=0}^{T_k-1}(2G + \|\delta_{k,t}\|)^2}{\beta_k T_k(T_k+3)} + \sum_{t=0}^{T_k-1}\frac{2(t+2)}{T_k(T_k+3)}\langle \delta_{k,t}, x' - u_{k,t}\rangle\ , \tag{48}$$

and the last inequality uses convexity of ψ_λ and $\Psi_\lambda = f + \psi_\lambda$, definition of β_k (Line 4.3), and the fact that

$$T_k \le T_{k+1} \quad \text{(Line 4.3)} \quad \text{, and} \quad \frac{(T_{k+1}+1)(T_{k+1}+2)}{T_{k+1}(T_{k+1}+3)} \le \frac{(T_k+1)(T_k+2)}{T_k(T_k+3)} \tag{49}$$

This proves the approximate descent guarantee: $\Phi_k \le \Phi_{k-1} + k(\eta_k + \eta'_k)$, which along with the facts: $1 \le T_1$ and $z_0 = z'_0 = x_0$ gives

$$\Psi_\lambda(x_K, x'_K) - \Psi_\lambda(x, x') \le \frac{4(\|x_0 - x\|^2 + (3/2)\|x_0 - x'\|^2)}{\lambda K(K+1)} + \frac{\sum_{k=1}^{K} 2k(\eta_k + \eta'_k)}{K(K+1)} \tag{50}$$

Now we take expectation, with respect to randomness in all the stochastic subgradients $((\widehat{g}_{k,i})_{i=1}^{T_k})_{k=1}^{K}$ used in the algorithm, on both sides of (50). Then the expectation of the error from the `Prox-Slide` procedure can be bounded as follows

$$\begin{aligned} \sum_{k=1}^{K} 2k\mathbb{E}[\eta'_k] &= \sum_{k=1}^{K} 2k\mathbb{E}[\frac{4\sum_{t=0}^{T_k-1}(2G + \|\delta_{k,t}\|)^2}{\beta_k T_k(T_k+3)} + \sum_{t=0}^{T_k-1}\frac{2(t+2)}{T_k(T_k+3)}\langle \delta_{k,t}, u - u_t\rangle] \\ &\le \sum_{k=1}^{K} 2k\frac{8\,(4G^2 + \sigma^2)}{(\frac{4}{\lambda k})(\frac{(4G^2+\sigma^2)\lambda^2 K k^2}{2\tilde{D}})} + 0 \\ &= \frac{8\tilde{D}}{\lambda} \end{aligned} \tag{51}$$

where we use (48), linearity of expectation, $(a+b)^2 \le 2(a^2 + b^2)$, variance of stochastic gradient $\mathbb{E}[\|\delta_{k,t}\|^2 | u_{k,t}] = \mathbb{E}[\|\widehat{g}_{k,t} - g_{k,t}\|^2 | u_{k,t}] \le \sigma^2$ (3), the value of T_k from Line 4.3, and the fact that expectation of the second term becomes zero, since $\mathbb{E}[\widehat{g}_{k,i-1} \mid u_{k,i-1}] = g_{k,i-1}$, which in turn implies

$$\mathbb{E}[\langle \delta_{k,t}, x' - u_{k,t}\rangle] = \mathbb{E}\big[\,\mathbb{E}[\langle \widehat{g}_{k,t} - g_{k,t}, x' - u_{k,t}\rangle \mid u_{k,t}]\,\big]s = \mathbb{E}[\langle 0, x' - u_{k,i-1}\rangle] = 0\ . \tag{52}$$

Aside: Note that, in the final guarantee, when we set $\lambda = \varepsilon/G^2$ and $K = \mathcal{O}(1/\varepsilon)$, we are setting $T_k = \Theta(\varepsilon\, k^2) = \mathcal{O}(1/\varepsilon)$ and $1/\beta_k = \Theta(1/\varepsilon\, k) = \mathcal{O}(1)$, so that the error from the `Prox-Slide` procedure is small enough. For example, at $k = K$, $\mathbb{E}[\eta_K] = \mathcal{O}((G^2 + \sigma^2)/\beta_K T_K) = \mathcal{O}(\varepsilon)$.

Now taking expectation on both sides of (50) and using linearity of expectation and (51) we get that

$$\mathbb{E}[\Psi_\lambda(x_K, x'_K)] - \Psi_\lambda(x, x') \leq \frac{4(\|x_0 - x\|^2 + (3/2)\|x_0 - x'\|^2 + 2\tilde{D})}{\lambda K(K+1)} + \frac{\sum_{k=1}^{K} 2k\,\eta_k}{K(K+1)} \tag{53}$$

Next setting $x' = x = x^* \in \mathcal{X} \subseteq \mathcal{X}'$ and using Lemma 2 and (5) we get that

$$\mathbb{E}[f(x_K)] - f(x^*) \leq \frac{10\|x_0 - x^*\|^2 + 8\tilde{D}}{\lambda K(K+1)} + \frac{\sum_{k=1}^{K} 2k\,\eta_k}{K(K+1)} + G^2\frac{\lambda}{2} \tag{54}$$

Aside: Note that, in the final guarantee, when we set $\lambda = \varepsilon/G^2$, the third term, which is the error from the Moreau smoothing becomes $\varepsilon/2$. Additionally, when $K = \mathcal{O}(1/\varepsilon)$, first term above is $\mathcal{O}(\varepsilon)$. Further, when we set $T_k = \Theta(\varepsilon\,k^2) = \mathcal{O}(1/\varepsilon)$, we get $1/\beta_k = \mathcal{O}(1/\varepsilon\,k)$ and $\mathbb{E}[\eta_k] = \mathcal{O}(1/k)$ so that the second term is also $\mathcal{O}(\varepsilon)$. □

Next using the above result we derive the guarantees for MOPES (Algorithm 1) and MOLES (Algorithm 2) as corollaries of Theorem 5.

C.2.1 Proof of Theorem 1

Proof. Notice that exact projection on Line 1.5 of Algorithm 1 is equivalent to choosing $\eta_k = 0$ in Algorithm 4. Then setting, $\eta_k = 0$, and $\lambda = \varepsilon/G^2$ in Theorem 5 we get

$$\mathbb{E}[f(x_K)] - f(x^*) \leq \frac{G^2(10\|x_0 - x^*\|^2 + 8\tilde{D})}{\varepsilon K(K+1)} + 0 + \frac{\varepsilon}{2} \tag{55}$$

Now, by using the given choices: $\tilde{D} = c\|x_0 - x^*\|^2$ and $K = \lceil\frac{2\sqrt{10+8c}\,G\|x_0 - x^*\|}{\varepsilon}\rceil$, we get

$$\mathbb{E}[f(x_K)] - f(x^*) \leq \varepsilon \tag{56}$$

Then the number of PO calls made by the algorithm is $K = \mathcal{O}(\frac{G\|x_0 - x^*\|}{\varepsilon})$ and the total number of SFO calls made subgradients made is

$$\begin{aligned}\sum_{k=1}^{K} T_k \leq \sum_{k=1}^{K}\left(\frac{(4G^2+\sigma^2)\lambda^2 K k^2}{2\tilde{D}} + 1\right) &= \frac{(4G^2+\sigma^2)\varepsilon^2 K^2(K+1)(2K+1)}{12cG^4\|x_0 - x^*_\lambda\|^2} + K \\ &= \mathcal{O}\left(\frac{(G^2+\sigma^2)\|x_0 - x^*\|^2}{\varepsilon^2}\right),\end{aligned} \tag{57}$$

where we used Line 1.3 and the given choices for λ, K, and $\tilde{D}$. □

C.2.2 Proof of Theorem 2

Proof. Notice that at step k of Algorithm 2 choosing $\hat{T} = \lceil\frac{7KD_{\mathcal{X}}^2}{c'\tilde{D}}\rceil = \mathcal{O}(\frac{1}{\varepsilon})$ is equivalent to choosing $\eta_k = \frac{4c'\tilde{D}}{\lambda K k}$ in Algorithm 4 (see below). Therefore by setting, $\eta_k = \frac{4c'\tilde{D}}{\lambda K k} = \mathcal{O}(\frac{1}{k})$, and $\lambda = \varepsilon/G^2$ in Theorem 5 we get

$$\mathbb{E}[f(x_K)] - f(x^*) \leq \frac{G^2(10\|x_0 - x^*\|^2 + 8\tilde{D} + 8c'\tilde{D})}{\varepsilon K(K+1)} + \frac{\varepsilon}{2} \tag{58}$$

Now, by using the given choices: $\tilde{D} = c\|x_0 - x^*\|^2$ and $K = \lceil\frac{2\sqrt{10+8c(1+c')}G\|x_0 - x^*\|}{\varepsilon}\rceil$, in the (58) we get

$$\mathbb{E}[f(x_K)] - f(x^*) \leq \varepsilon \tag{59}$$

Then using the similar arguments as in proof of Theorem 2, we can show that $K = \mathcal{O}(\frac{G\|x_0 - x^*\|}{\varepsilon})$ and the total number of SFO calls made is $\sum_{k=1}^{K} T_k = \mathcal{O}(\frac{(G^2+\sigma^2)\|x_0 - x^*\|^2}{\varepsilon^2})$.

Finally we calculate the total number of LMO calls made. At outer step k of Algorithm 2, after using $\hat{T} = \lceil\frac{7KD_{\mathcal{X}}^2}{c'\tilde{D}}\rceil$ LMO calls in the `Approx-Proj` procedure, we can find a feasible point z_k whose

Wolfe duality gap (38) is at most $\lceil \frac{6\beta_k D_{\mathcal{X}}^2}{\hat{T}} \rceil \leq \frac{4c'\tilde{D}}{\lambda K k} = \eta_k$ if we use CndG procedure [56, Theorem 2.2(c)] or $\lceil \frac{7\beta_k D_{\mathcal{X}}^2}{\hat{T}} \rceil \leq \frac{4c'\tilde{D}}{\lambda K k} = \eta_k$ if we use the standard Frank-Wolfe algorithm [45, Theorem 2]. Therefore the total number of linear minimization oracle calls made by the algorithm is

$$K\hat{T} = \frac{7K^2 D_{\mathcal{X}}^2}{cc'\|x_0 - x^*\|^2} + K = \mathcal{O}\Big(\frac{G^2 D_{\mathcal{X}}^2}{\varepsilon^2}\Big), \tag{60}$$

where we used Line 1.3 and the given choices for K and $\tilde{D}$. □

C.2.3 Proof of Proposition 3: Analysis of `Prox-Slide` (Line 4.10) procedure

Proof. We analyze the `Prox-Slide` procedure for a fixed k, therefore we drop k from ϕ_k, $u_{k,t}$, $\widehat{g}_{k,t}$, and $\delta_{k,t}$, which are denoted here with ϕ, u_t, $\widehat{g}_t$, and δ_t. `Prox-Slide` has the following update steps.

$$\theta_t = \frac{2(t+1)}{t(t+3)}, \ \ \widehat{g}_{t-1} = \text{SFO}\,(u_{t-1}) \tag{61}$$

$$\widehat{u}_t = u_{t-1} - \frac{1}{(1+t/2)\beta} \cdot (\widehat{g}_{t-1} + \beta(u_{t-1} - (u' - g/\beta))) \tag{62}$$

$$u_t = \widehat{u}_t \cdot \min(1, 2R/\|\widehat{u}_t\|) \tag{63}$$

$$\widetilde{u}_t = (1-\theta_t)\,\widetilde{u}_{t-1} + \theta_t u_t \tag{64}$$

By convexity and G-Lipschitzness of f in $\mathcal{X}'$, for any $u \in \mathcal{X}'$, we get

$$\begin{aligned} f(u_{t+1}) - f(u) &= f(u_{t+1}) - f(u_t) + f(u_t) - f(u) \\ &\leq \langle g_{t+1}, u_{t+1} - u_t\rangle + \langle g_t, u_t - u\rangle \\ &= \langle \widehat{g}_t, u_{t+1} - u_t\rangle + \langle g_{t+1} - g_t - \delta_t, u_{t+1} - u_t\rangle + \langle \widehat{g}_t, u_t - u\rangle - \langle \delta_t, u_t - u\rangle \\ &\leq \langle \widehat{g}_t, u_{t+1} - u\rangle + (2G + \|\delta_t\|)\,\|u_{t+1} - u_t\| + \langle \delta_t, u - u_t\rangle\ , \end{aligned} \tag{65}$$

where we used the fact that $\delta_t = \widehat{g}_t - g_t$. Notice that $u_t = \widehat{u}_t \cdot \min(1, 2R/\|\widehat{u}_t\|)$ is the projection of $\widehat{u}_t$ onto $\mathcal{X}' = B(0, 2R)$. Therefore, using Line 4.14, we can re-write `Prox-Slide` update as

$$\begin{aligned} u_{t+1} &= \operatorname*{argmin}_{u\in\mathcal{X}'} \frac{(t+3)\beta}{4}\ \|u - \widehat{u}_t\|^2 \\ &= \operatorname*{argmin}_{u\in\mathcal{X}'} \frac{(t+3)\beta}{4}\ \Big\|u - \Big(u_{t-1} - \frac{1}{(1+((t+1)/2)\beta} \cdot (\widehat{g}_{t-1} + \beta(u_{t-1} - (u_0 - g/\beta)))\Big)\Big\|^2 \\ &= \operatorname*{argmin}_{u\in\mathcal{X}'} \Big[F_t(u) := \langle g, u\rangle + \langle \widehat{g}_t, u\rangle + \frac{(t+1)\beta}{4}\|u - u_t\|^2 + \frac{\beta}{2}\|u - u_0\|^2\Big] \end{aligned} \tag{66}$$

By $\beta(t+3)/2$-strong convexity of the quadratic update objective $F_t(u)$ and the optimality of $u_{t+1} \in \operatorname{argmin}_{u\in\mathcal{X}'} F_t(u)$, we get that for any $u \in \mathcal{X}'$

$$\frac{\beta(t+3)}{4}\|u_{t+1} - u\|^2 + F_t(u_{t+1}) \leq F_t(u) \tag{67}$$

We want to provide a lower bound on $\phi(u)$ 44 which is defined as follows, when using the private notation of the `Prox-Slide` procedure by setting $u = x'$, $u' = z_{k-1}$, $g = \nabla_{k,x'}$, $\beta = \beta_k$.

$$\phi(u) := f\,(u) + \langle g, u\rangle + \frac{\beta}{2}\,\|u - u_0\|^2\ . \tag{68}$$

Now adding together (65) and (67) and using the definitions of F_t and ϕ we get

$$\begin{aligned} \phi(u_{t+1}) - \phi(u) \leq &\ \frac{\beta}{2}(\frac{t+1}{2}\|u_t - u\|^2 - \frac{t+3}{2}\|u_{t+1} - u\|^2) + \\ &\ (2G + \|\delta_t\|)\,\|u_{t+1} - u_t\| - \frac{\beta\,(t+1)}{4}\|u_{t+1} - u_t\|^2 + \langle \delta_t, u - u_t\rangle \\ \leq &\ \frac{\beta}{2}(\frac{t+1}{2}\|u_t - u\|^2 - \frac{t+3}{2}\|u_{t+1} - u\|^2) + \frac{(2G + \|\delta_t\|)^2}{\beta\,(t+1)} + \langle \delta_t, u - u_t\rangle \end{aligned} \tag{69}$$

where the second inequality follows from $ax - bx^2/2 \leq a^2/2b$. Now multiplying the above inequality is by $2(t+2)/T(T+3)$ and then summing over $t = \{0, \dots, T-1\}$, we get

$$\begin{aligned}\sum_{t=0}^{T-1} \frac{2(t+2)}{T(T+3)}(\phi(u_{t+1}) - \phi(u)) \leq \frac{\beta}{2}\frac{2}{T(T+3)}(\|u_0 - u\|^2 - \frac{(T+1)(T+2)}{2}\|u_T - u\|^2) + \\ \frac{2}{T(T+3)}\frac{2\sum_{t=0}^{T-1}(2G + \|\delta_t\|)^2}{\beta} + \sum_{t=0}^{T-1}\frac{2(t+2)}{T(T+3)}\langle \delta_t, u - u_t\rangle \\ \phi(\widetilde{u}_T) - \phi(u) \leq \end{aligned} \tag{70}$$

where the last inequality uses Jensen's inequality and $\widetilde{u}_T = \sum_{t=1}^{T}\frac{2(t+1)}{T(T+3)}u_t$, last of which follows from Lines 4.13 and 4.16 as follows

$$\begin{aligned}\widetilde{u}_T &= (1 - \theta_T)\widetilde{u}_{T-1} + \theta_T u_T \\ &= \frac{(T-1)(T+2)}{T(T+3)}((1 - \theta_{T-1})\widetilde{u}_{T-2} + \theta_{T-1}u_{T-1}) + \frac{2(T+1)}{T(T+3)}u_T \\ &= \frac{(T-2)(T+1)}{T(T+3)}\widetilde{u}_{T-2} + \frac{2(T)}{T(T+3)}u_{T-1} + \frac{2(T+1)}{T(T+3)}u_T \\ &\vdots \\ &= \sum_{t=1}^{T}\frac{2(t+1)}{T(T+3)}u_t \end{aligned} \tag{71}$$

Finally we get the desired result by setting $\phi = \phi_k$, $\beta = \beta_k$, $T = T_k$, $u_0 = \widetilde{z}'_{k-1}$, $u = x'$, $u_t = u_{k,t}$, $\widetilde{u}_T = \widetilde{z}'_k$, and $u_T = z_k$ we get the desired inequality

$$\begin{aligned}\phi_k(\widetilde{z}'_k) - \phi_k(x') \leq \frac{2}{T_k(T_k+3)}\frac{\beta_k}{2}\|z'_{k-1} - x'\|^2 - \frac{(T_k+1)(T_k+2)}{T_k(T_k+3)}\frac{\beta_k}{2}\|z'_k - x'\|^2 + \\ \frac{4\sum_{t=0}^{T_k-1}(2G + \|\delta_{k,t}\|)^2}{\beta_k T_k(T_k+3)} + \sum_{t=0}^{T_k-1}\frac{2(t+2)}{T_k(T_k+3)}\langle \delta_{k,t}, x' - u_{k,t}\rangle \end{aligned} \tag{72}$$

□

C.3 Proof of Lemma 1

We re-write $f_\lambda(x)$ as minimum value of a $\frac{1}{\lambda}$-strong convex function $\phi_{\lambda,x}$ as follows

$$f_\lambda(x) = \min_{x' \in \mathcal{X}'}\left[\phi_{\lambda,x}(x') := f(x') + \frac{1}{2\lambda}\|x - x'\|^2\right]. \tag{73}$$

Note that $\phi_{\lambda,x}(\cdot)$ is a $(1/\lambda)$-strongly convex function as f is convex and $(1/\lambda)\|\cdot - x\|^2$ is strongly convex, and $f_\lambda(x) = \min_{x'\in\mathcal{X}'}\phi_{\lambda,x}(x')$.
(a) The existence and uniqueness of $\hat{x}_\lambda(x) \in \mathcal{X}'$ follows from the strong convexity of $\phi_{\lambda,x}(\cdot)$ and the fact that f is a proper convex function. Then $f(\hat{x}_\lambda(x)) \leq \phi_{\lambda,x}(\hat{x}_\lambda(x)) = \min_{x'\in\mathcal{X}'}\phi_{\lambda,x}(x') = f_\lambda(x) \leq \phi_{\lambda,x}(x) = f(x)$.
(b) Let $g_x := (x - \hat{x}_\lambda(x))/\lambda$ for any $x \in \mathbb{R}^d$. By $(1/\lambda)$-strong convexity of $\phi_{\lambda,x}(x')$ and $\hat{x}_\lambda(x) = \operatorname{argmin}_{x'\in\mathcal{X}'}\phi_{\lambda,x}(x')$, we have, for any $x' \in \mathcal{X}'$, that

$$\begin{aligned}\phi_{\lambda,x}(x') \geq \phi_{\lambda,x}(\hat{x}_\lambda(x)) + \|x' - \hat{x}_\lambda(x)\|^2/2\lambda \\ \iff f(x') + \|x' - x\|^2/2\lambda \geq f(\hat{x}_\lambda(x)) + \|x' - \hat{x}_\lambda(x)\|^2/2\lambda + \|x' - \hat{x}_\lambda(x)\|^2/2\lambda \\ \iff f(x') \geq f(\hat{x}_\lambda(x)) + \langle g_x, x' - \hat{x}_\lambda(x)\rangle \end{aligned} \tag{74}$$

Using this, for any $x, y \in \mathbb{R}^d$ we get

$$\begin{aligned} f_\lambda(y) - f_\lambda(x) &= f(\hat{x}_\lambda(y)) - f(\hat{x}_\lambda(x)) + (\|\hat{x}_\lambda(y) - y\|^2 - \|\hat{x}_\lambda(x) - x\|^2)/2\lambda \\ &\geq \langle g_x, \hat{x}_\lambda(y) - \hat{x}_\lambda(x)\rangle + \lambda/2(\|g_y\|^2 - \|g_x\|^2) = \langle g_x, y - x\rangle + \lambda/2\|g_x - g_y\|^2 \end{aligned} \tag{75}$$

By instantiating the above for $y \leftarrow x, x \leftarrow y$, we also get $f_\lambda(y) - f_\lambda(x) \leq \langle g_y, y - x\rangle - \lambda/2\|g_x - g_y\|^2$. Combining these two inequalities

$$\begin{aligned} 0 \leq \lambda/2\|g_y - g_x\|^2 \leq f_\lambda(y) - f_\lambda(x) - \langle g_x, y - x\rangle &\leq -\lambda/2\|g_y - g_x\|^2 + \langle g_y - g_x, y - x\rangle \\ &\leq -\lambda/2\|g_y - g_x\|^2 + \|g_y - g_x\|\|y - x\| \\ &\leq \|y - x\|^2/2\lambda \end{aligned} \tag{76}$$

This implies that $\lim_{y\to x}(f_\lambda(y) - f_\lambda(x) - \langle g_x, y - x\rangle)/\|y - x\| = 0$. Thus f_λ is Frechet differentiable with gradient $\nabla f_\lambda(x) = g_x = (x - \hat{x}_\lambda(x))/\lambda$. The above inequality also implies f_λ is convex and $1/\lambda$-smooth.
(c) Let $x \in \mathcal{X}'$. Using $1/\lambda$-strong convexity of $\phi_{\lambda,x}$ and $\hat{x}_\lambda(x) \in \operatorname{argmin}_{x'\in\mathcal{X}'} \phi_{\lambda,x}(x')$, and G-Lipschitzness of f in $\mathcal{X}'$, we get

$$\begin{aligned} \|x - \hat{x}_\lambda(x)\|^2/2\lambda \leq \phi_{\lambda,x}(x) - \phi_{\lambda,x}(\hat{x}_\lambda(x)) &= f(x) - f_\lambda(x) \\ &= f(x) - f(\hat{x}_\lambda(x)) - \|x - \hat{x}_\lambda(x)\|^2/2\lambda \\ &\leq G\|\hat{x}_\lambda(x) - x\| - \|x - \hat{x}_\lambda(x)\|^2/2\lambda \leq G^2\lambda/2\ . \quad \square \end{aligned}$$

D Additional details for the experiments in Section 5

For all the experiments we randomly and uniformly sample a point x_0 from the surface of the nuclear norm ball of radius r. For all the figures where we plot the estimated sub-optimality gap: $f(x_k) - \hat{f}^*$, where $\hat{f}^*$ is the estimated minimum function value calculated by running the PGD method for a large number of iterations. We plot the mean (standard error is negligible) of the sub-optimality gap over 10 runs using 10 different initial points x_0's (same 10 initial points for all algorithms).

For experiments in Figures 1 and 2, we use a subset of the Imagewoof 2.0 dataset [43], which in itself is a subset of the Imagenet dataset [24]. The training data, contains $n = 400$ samples $\{(A_i, y_i)\}_{i=1}^n$ where A_i is a 224×224 grayscale image of one of the two types of dogs (classes n02087394 and n02115641 in Imagenet dataset) labeled using $y_i \in \{0, 1\}$. Note that the effective dimension is $d = 224 \times 224 = 50176$). These grayscale images are generated from the raw 8-bit RGB Imagewoof images using the Pillow python image-processing library [21], by (i) resizing to 256×256 pixels: `resize(256,256)`, (ii) cropping to the central 224×224 pixels: `crop(16,16,240,240)`, (iii) converting to the grayscale: `convert(mode='L')`, and (iv) normalizing by 255.0 so that the pixel values lie in range $[0, 1]$. For incorporating bias scalar into the SVM model we also zero-pad the training images with an additional column and row of zeros to the right and the bottom of the image array A_i. We use $r = 0.1$ as nuclear norm ball radius of $\mathcal{X}$, thus $D_\mathcal{X} = 0.2$. We have access to a deterministic FO.

In Figure 1, we use a Lipschitz constant of $G = 50$. For MOPES we set $c = 40$ and $\varepsilon = 5.0$, and for PGD we use two stepsize schemes: (i) fixed stepsize $D_\mathcal{X}/(G\sqrt{K})$ with $K = 10^3$ and (ii) diminishing stepsize $D_\mathcal{X}/(G\sqrt{k})$ with $K = 10^3$.

In Figure 2, we use a Lipschitz constant of $G = 50$. For MOLES we set $c = 40$, $c' = 1$ and $\varepsilon = 5.0$. For FW-PGD we use two stepsize schemes: (i) fixed stepsize $D_\mathcal{X}/(G\sqrt{K})$ with $K = 10^3$ and (ii) diminishing stepsize $D_\mathcal{X}/(G\sqrt{k})$ with $K = 10^3$. Both of these stepsize schemes use a projection tolerance of $\eta_k G^2/2$. For RandFW we use the standard parameter choices as given in [54, Theorem 5] with $K = 150$.

In practice, in the deterministic setup with FO, at outer-step k, we can use the following criterion for stopping the `Prox-Slide` (Line 1.8) procedure early at some $t \geq \widehat{T}_{k-1}$ (defined recursively below with $\widehat{T}_0 = 1$) and $t \leq T_k$. Let $\phi_k(x') := f(x') + \langle \nabla_{k,x'}, x'\rangle + \frac{\beta_k}{2}\left\|x' - z'_{k-1}\right\|^2$ and $\widetilde{g}_t \in \partial f(\widetilde{u}_t)$. Now if

$$\begin{aligned} &\max_{x'\in\mathcal{X}} \langle \widetilde{g}_t + \nabla_{k,x'}, \widetilde{u}_t - x'\rangle - \frac{(t+1)(t+2)}{t(t+3)}\beta_k \left\langle u_t - z'_{k-1}, x'\right\rangle \\ &\quad\leq \frac{8\,(4G^2 + \sigma^2)}{\beta_k(T_k + 3)} - \frac{\beta_k}{2}\|\widetilde{u}_t - z'_{k-1}\|^2 + \frac{(t+1)(t+2)}{t(t+3)}\frac{\beta_k}{2}(\|z'_{k-1}\|^2 - \|u_t\|^2) \end{aligned} \tag{77}$$

then we stop the procedure, set $\widehat{T}_k = t$ and return $(u_t, \widetilde{u}_t)$. This implies that for $(z'_k, \widetilde{z}'_k) = (u_t, \widetilde{u}_t)$

$$\phi_k(\widetilde{u}_t) - \phi_k(x') \leq \frac{2}{\widehat{T}_k(\widehat{T}_k + 3)} \frac{\beta_k}{2} \|z'_{k-1} - x'\|^2 - \frac{(\widehat{T}_k + 1)(\widehat{T}_k + 2)}{\widehat{T}_k(\widehat{T}_k + 3)} \frac{\beta_k}{2} \|u_t - x'\|^2 + \\ \frac{4 \sum_{t=0}^{T_k - 1} (2G)^2}{\beta_k T_k (T_k + 3)} \tag{78}$$

for all $x' \in \mathcal{X}$. Now the only change we need to make in the analysis of Theorem 5 is the change of the potential (40) to

$$\Phi_k := k(k+1)(\Psi_\lambda(x_k, x'_k) - \Psi_\lambda(x, x')) + \frac{4}{\lambda}(\|z_k - x\|^2 + \frac{(\widehat{T}_{k+1} + 1)(\widehat{T}_{k+1} + 2)}{\widehat{T}_{k+1}(\widehat{T}_{k+1} + 3)} \|z'_k - x'\|^2) \tag{79}$$

The LHS of (77) is an linear optimization problem whose solution can be easily found as

$$\begin{aligned} &\text{LMO}\left(\widetilde{g}_t + \nabla_{k,x'} + \frac{(t+1)(t+2)}{t(t+3)} \beta_k (u_t - z'_{k-1})\right) \\ &= \lim_{\alpha \to \infty} \text{PO}\left(-\alpha\left(\widetilde{g}_t + \nabla_{k,x'} + \frac{(t+1)(t+2)}{t(t+3)} \beta_k (u_t - z'_{k-1})\right)\right) . \end{aligned} \tag{80}$$

We also use a slightly modified $T_k = \left\lceil \frac{2G^2 \lambda^2 K k^2}{2\tilde{D}} \right\rceil$ for our experiments, since the deterministic FO we use, ensures this choice gets the same guarantees as given in our theorems. Also, in our implementation we do not explicitly project z_k onto $\mathcal{X}'$, as in practice this does not seem needed.

In practice, we can eliminate the need for selecting ε of MOPES by employing ε-doubling trick with warm restarts, which can increase the worse-case iteration complexity by a factor of at most 2 but oftentimes will accelerate the convergence [80, Algorithm 6].

E Additional details for applications

We refer to [73] for some more nonsmooth problems which can be solved using an LMO. In the following subsections, we compare the analytical complexities for solving some of the applications mentioned in Section 4, using different algorithms.

E.1 ℓ_1 norm constrained SVM

For simplicity, we work with the vector version of the matrix problems and replace nuclear norm constraint with the ℓ_1 norm constraint. The standard ℓ_1 norm constrained soft-margin SVM can be formulated as the the following optimization problem:

$$\begin{aligned} &\min_{x \in \mathbb{R}^d} f(x) = \frac{1}{n} \sum_{i=1}^{n} [f_i(x) = \max(0, 1 - \langle x, a_i \rangle)] \\ &\text{subject to} \quad \|x\|_1 \leq \lambda \end{aligned} \tag{81}$$

where $a_i \in \mathbb{R}^d$ captures the d-dimensional feature vector multiplied by a binary class value in $\{-1, 1\}$ and $\mathcal{X} = \{x \mid \|x\|_1 \leq 1\}$ is the constraint set. We do not include any explicit bias term above, because it can always be incorporated into the model by augmenting a_i with a constant dimension. We assume that n is large and therefore we only have access to minibatched stochastic subgradients obtained through minibatching d $(b = o(n))$ uniformly sampled (with replacement) training samples. We assume that f is G_p-Lipschitz continuous and the variance of any stochastic subgradient is upperbounded by σ_p^2, both calculated in ℓ_p norm $\|\cdot\|_p$, for $p = 1, 2$. We define $q := (1 - 1/p)^{-1} \in \{\infty, 2\}$. Then

$$G_p = \max_{\|x\|_1 \leq \lambda} \|\frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\{\langle x, a_i \rangle < 1\} a_i\|_q, \text{ and}$$

$$\sigma_p^2 = \max_{\|x\|_1 \leq \lambda} \mathbb{E}_{\{I_j\}_{j=1}^b} \|\frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\{\langle x, a_i \rangle < 1\} a_i - \frac{1}{b} \sum_{j=1}^{b} \mathbb{I}\{\langle x, a_{I_j} \rangle < 1\} a_{I_j}\|_q^2 \tag{82}$$

PO based methods (using ℓ_p norm)		
Nonsmooth methods ($p = 2$)	**PO**: $\mathcal{O}(d \ln\, d)$	**SFO**: $\mathcal{O}(d + n)$
Our MOPES ($p = 2$) [Theorem 1]	$\mathcal{O}\big(\frac{G_2}{\varepsilon}\lambda\big)$	$\mathcal{O}\big(\frac{G_2^2+\sigma_2^2}{\varepsilon^2}\lambda^2\big)$
PGD ($p = 2$)	$\mathcal{O}\big(\frac{G_2^2}{\varepsilon^2}\lambda^2\big)$	$\mathcal{O}\big(\frac{G_2^2+\sigma_2^2}{\varepsilon^2}\lambda^2\big)$
Randomized smoothing ($p = 2$) [27]	$\mathcal{O}\big(d^{1/4}\frac{G_2}{\varepsilon}\lambda\big)$	$\mathcal{O}\big(\frac{G_2^2+\sigma_2^2}{\varepsilon^2}\lambda^2\big)$
Nonsmooth methods ($p = 1$)	**MO**: $\mathcal{O}(d)$	**SFO**: $\mathcal{O}(d + n)$
Mirror descent ($p = 1$) [64]	$\mathcal{O}\big(\ln(d+1)\frac{G_1^2}{\varepsilon^2}\lambda^2\big)$	$\mathcal{O}\big(\ln(d+1)\frac{G_1^2+\sigma_1^2}{\varepsilon^2}\lambda^2\big)$
Randomized smoothing ($p = 1$) [27]	$\mathcal{O}\big(\sqrt{d\ln(d+1)}\frac{G_1}{\varepsilon}\lambda\big)$	$\mathcal{O}\big(\ln(d+1)\frac{G_1^2+\sigma_1^2}{\varepsilon^2}\lambda^2\big)$
Minimax methods: $\mathcal{O}(n)$ extra memory	**PO+MO**: $\mathcal{O}(d \ln\, d + n)$	**SFO**: $\mathcal{O}(d + n)$
Variance reduced Mirror-Prox ($p = 1$)[16]	$\mathcal{O}\Big(\frac{dn}{d+n} + \frac{L_{12}}{\varepsilon}\sqrt{\frac{dn}{d+n}}(\lambda\sqrt{n\ln d})\Big)$	

Table 2: Projection: Comparison of PO/MO and SFO calls complexities (PO-CC and SFO-CC) of various methods for d-dimensional ℓ_1 norm constrained SVM with n training samples. SFO uses a batchsize of $b = o(n)$. Our MOPES outperforms other nonsmooth methods in PO-CC/MO-CC while still maintaining $\mathcal{O}(1/\varepsilon^2)$ SFO-CC. Complexities of methods based on smooth minimax reformulation adversely scale with n or d.

PO: First we study the case of PO (or MO: Mirror descent step oracle) in the high-dimensional ($\text{poly}(G_p, \sigma_p, \lambda, 1/\varepsilon) \ll d$) and large-scale ($1 \ll \text{poly}(n)$) regime. In Table 2 we provide the PO-CC and SFO-CC of MOPES ($p = 1$, Algorithm 1) and competing nonsmooth methods: PGD ($p = 2$) [34, 59], Mirror descent ($p = 1$) [64], Randomized smoothing ($p = 1$ or $p = 2$) [27]. The p value in brackets marks which ℓ_p norm the method uses. By definition $G_1 \leq G_2 \leq \sqrt{d}G_1$ and $\sigma_1 \leq \sigma_2 \leq \sqrt{d}\sigma_1$. Therefore, in this high-dimensional and large-scale regime, and when $G_2 = o(\sqrt{d}G_1)$ and $\sigma_2 = o(\sqrt{d}\sigma_1)$, MOPES has a more efficient PO-CC than other competing nonsmooth first-order methods, while still maintaining $\mathcal{O}(1/\varepsilon^2)$ SFO-CC. Note that PO has a computational complexity of $O(d \log d)$ because it involves sorting [26], MO has a computational complexity of $\mathcal{O}(d)$, and SFO has a computational complexity of $\mathcal{O}(b(d+n))$ because it involves sampling b vectors from a set of n d-dimensional vectors. In practice, sorting could contribute to a significant part of the wall-clock time.

Many nonsmooth convex objectives in machine learning like the hinge loss here can be written as smooth convex-concave minimax objectives of the form

$$\min_{x\in\mathcal{X}} \frac{1}{n}\sum_{i=1}^{n} \max_{y_i\in\mathcal{Y}_i} g_i(x, y_i) \tag{83}$$

where $g_i(x, \cdot)$ is concave and g_i is L-smooth for all $i \in [n]$. However, the iteration/projection complexities of even the best variance reduced algorithms could have a dependence on the number n of additionally introduced dual variables $\{y_i\}_{i=1}^n$ [71, 16]. Therefore in the regime when comparatively n is large and ε is moderate ($\text{poly}(1/\varepsilon) \ll n$), it is more efficient to optimize the original stochastic nonsmooth formulation than the smooth minimax reformulation.

Concretely, the soft-margin SVM problem with a hinge loss, can be reformulated as a saddle point problem of the following form

$$\min_{\|x\|_1\leq 1} \max_{y\in[0,1]^n} \frac{1}{n}(y^T\mathbf{1} - y^T Ax)\,. \tag{84}$$

This smooth saddle point problem is an ℓ_1-ℓ_2 matrix game (ignoring possibility of ℓ_∞ optimization due to limited literature) which is L_{12}-smooth, where

$$L_{12} = \max_{\|x\|_1\ \leq 1} \max_{\|y\|_2\leq 1} \frac{1}{n} y^T Ax = \frac{1}{n} \max_{i=1,\ldots,n} \|a_i\|_2\,. \tag{85}$$

Note that the primal (in ℓ_1 norm) and dual (in ℓ_2 norm) space diameters are $D_{\mathcal{X}} = \mathcal{O}(\lambda)$ and $D_{\mathcal{Y}} = \mathcal{O}(\sqrt{n})$ respectively.

Next we derive the MO and SFO calls complexities (MO-CC and SFO-CC) of the variance reduced Mirror-prox method [16]. For any stepsize $\alpha \leq \frac{\varepsilon}{D_{\mathcal{X}} D_{\mathcal{Y}} \sqrt{\ln d}}$, this algorithm runs for $K = \mathcal{O}(\frac{\alpha D_{\mathcal{X}} D_{\mathcal{Y}} \sqrt{\ln d}}{\varepsilon})$ outer iterations, each of which uses $T = 1 + \frac{L_{12}^2}{\alpha^2}$ SFO calls and one FO call, and $T + 1$ primal and dual MO calls. Computational complexity of

- Primal MO is $O(d \log d)$ since it involves sorting,
- Dual MO is $O(n)$ since it involves normalization of each dual dimension,
- FO is $\mathcal{O}(dn)$ since it involves $d \times n$-matrix vector products, and,
- SFO is $\mathcal{O}(d+n)$ because it involves sampling from two set of n and d (d and n-dimensional, respectively) vectors.

We assume that the algorithm uses $\widetilde{T} = 1 + \frac{L_{12}^2}{\alpha^2} + \frac{dn}{d+n} = \mathcal{O}(\frac{L_{12}^2}{\alpha^2} + \frac{dn}{d+n})$ SFO calls per outer iteration, because computationally it is equivalent to $T = 1 + \frac{L_{12}^2}{\alpha^2}$ SFO calls and one FO call per outer iteration. Using the suggested stepsize $\alpha = \max(\frac{\varepsilon}{D_{\mathcal{X}} D_{\mathcal{Y}} \sqrt{\ln d}}, L_{12}\sqrt{\frac{d+n}{dn}})$, we get that

$$[\text{MO-CC} = \mathcal{O}(K \cdot T)] = \mathcal{O}\Big(\frac{dn}{d+n} + \frac{L_{12}}{\varepsilon}\sqrt{\frac{dn}{d+n}}(\lambda\sqrt{n \ln d})\Big) = [K \cdot \widetilde{T} = \text{SFO-CC}]\,. \quad (86)$$

In very high dimensional regime ($n \ll d$) or very large-scale regime ($d \ll n$), MO-CC of this smooth minimax formulation is $\mathcal{O}(d)$ or $\mathcal{O}(n)$ larger than PO-CC for MOPES. Further more the former method uses extra $\Theta(n)$ extra space for storing the dual variables.

LMO: Next we study the case of LMO in the high-dimensional ($\text{poly}(G_p, \sigma_p, \lambda, 1/\varepsilon) \ll d$) and large-scale ($1 \ll \text{poly}(n)$) regime. In Table 3 we provide the LMO and SFO calls complexities of MOLES ($p = 1$, Algorithm 1) and competing nonsmooth methods: FW-PGD ($p = 2$)—projection approximated with Frank-Wolfe method (Appendix B.2), and Randomized Frank-Wolfe method ($p = 1$ or $p = 2$) [54]. The p value in brackets marks which ℓ_p norm the method uses. By definition $G_1 \leq G_2 \leq \sqrt{d} G_1$ and $\sigma_1 \leq \sigma_2 \leq \sqrt{d}\sigma_1$. Therefore, in this high-dimensional and large-scale regime, MOLES has a more efficient dimension-free LMO-CC $\mathcal{O}(G_2^2 \lambda^2/\varepsilon^2)$ than other competing nonsmooth first-order methods, while still maintaining optimal $\mathcal{O}(1/\varepsilon^2)$ SFO-CC. Note that LMO has a computational complexity of $O(d)$ because it uses just one pass over a d-dimensional vector.

A competing method based on the smooth minimax reformulation is SP+VR-MP which combines ideas from Semi-Proximal [41] and Variance reduced [16] Mirror-Prox methods. Here SP+VR-MP uses the variance reduced Mirror-prox method [16] in the ℓ_2-ℓ_2 setting to optimize (84) and then approximates the projection steps with Frank-Wolfe (FW) method. This is an L_{22}-smooth minimax problem with

$$L_{22} = \max_{\|x\|_2 \leq 1} \max_{\|y\|_2 \leq 1} \frac{1}{n} y^T A x = \frac{1}{n}\|A\|_2\,. \quad (87)$$

where $\|A\|_2$ is the spectral norm of the matrix A, but the algorithm we are discussing will depend on

$$\widetilde{L}_{22} = \frac{1}{n}\|A\|_F\,. \quad (88)$$

where $\|A\|_F$ is the Frobenius norm of the matrix A. Note that $\frac{1}{\sqrt{\min(n,d)}}\|A\|_F \leq \|A\|_2 \leq \|A\|_F$. The primal and dual space diameters are again $D_{\mathcal{X}} = \mathcal{O}(\lambda)$ and $D_{\mathcal{Y}} = \mathcal{O}(\sqrt{n})$, respectively.

For each of the projection steps, the Frank-Wolfe method solves an $(\alpha + 10\frac{L_{22}^2}{\alpha})$-smooth convex optimization problem up to an error $\mathcal{O}(\varepsilon)$. Therefore each of these uses at most $\widehat{T} = \lceil(\alpha + 10\frac{L_{22}^2}{\alpha})\lambda^2/\varepsilon\rceil$ LMO calls. Thus using similar arguments as the PO setting and using the suggested stepsize $\alpha = \max(\frac{\varepsilon}{D_{\mathcal{X}} D_{\mathcal{Y}}}, \widetilde{L}_{22}\sqrt{\frac{d+n}{dn}})$, $K = \mathcal{O}(\frac{\alpha D_{\mathcal{X}} D_{\mathcal{Y}}}{\varepsilon})$ outer iterations, $\widetilde{T} = 1 + \frac{\widetilde{L}_{22}^2}{\alpha^2} = \mathcal{O}(\frac{\widetilde{L}_{22}^2}{\alpha^2})$ SFO calls per outer iteration, and $\widetilde{T} = 1 + \frac{\widetilde{L}_{22}^2}{\alpha^2} + \frac{dn}{d+n} = \mathcal{O}(\frac{\widetilde{L}_{22}^2}{\alpha^2} + \frac{dn}{d+n})$ effective number of SFO calls per outer iteration, we get that

$$[\text{SFO-CC} = K \cdot \widetilde{T}] = \mathcal{O}\Big(\frac{dn}{d+n} + \frac{\widetilde{L}_{22}}{\varepsilon}\sqrt{\frac{dn}{d+n}}(\lambda\sqrt{n})\Big)\,, \quad (89)$$

LMO based methods (using ℓ_p norm)		
Nonsmooth methods ($p = 2$)	**LMO**: $\mathcal{O}(d)$	**SFO**: $\mathcal{O}(d+n)$
MOLES ($p = 2$) [Theorem 2]	$\mathcal{O}\big(\frac{G_2^2}{\varepsilon^2}\lambda^2\big)$	$\mathcal{O}\big(\frac{G_2^2+\sigma_2^2}{\varepsilon^2}\lambda^2\big)$
FW-PGD ($p = 2$) [Theorem 4]	$\mathcal{O}\big(\frac{G_2^4+\sigma_2^4}{\varepsilon^4}\lambda^4\big)$	$\mathcal{O}\big(\frac{G_2^2+\sigma_2^2}{\varepsilon^2}\lambda^2\big)$
Rand. Frank-Wolfe ($p = 2$) [54]	$\mathcal{O}\big(d^{1/2}\frac{G_2^2}{\varepsilon^2}\lambda^2\big)$	$\mathcal{O}\big(\frac{G_2^4+\sigma_2^4}{\varepsilon^4}\lambda^4\big)$
Nonsmooth methods ($p = 1$)	**LMO**: $\mathcal{O}(d)$	**SFO**: $\mathcal{O}(d+n)$
Rand. Frank-Wolfe ($p = 1$) [54]	$\mathcal{O}\big(d\ln(d+1)\frac{G_1^2}{\varepsilon^2}\lambda^2\big)$	$\mathcal{O}\big(\ln^2(d+1)\frac{G_1^4+\sigma_1^4}{\varepsilon^4}\lambda^2\big)$
Minimax methods: $\mathcal{O}(n)$ extra memory	**LMO**: $\mathcal{O}(d)$	**SFO**: $\mathcal{O}(d+n)$
SP [41]+VR [16]-MP ($p = 2$)	**SFO-CC** $+\, \mathcal{O}\Big(\frac{d\sqrt{n}\,\lambda}{d+n}+$ $\frac{\widetilde{L}_{22}\,\lambda^2}{\varepsilon}\Big(\frac{dn}{d+n}\Big)^{\frac{3}{2}}+\frac{\widetilde{L}_{22}^2\,\lambda^3\sqrt{n}}{\varepsilon^2}\Big(\frac{dn}{d+n}\Big)\Big)$	$\mathcal{O}\Big(\frac{dn}{d+n}+$ $+\,\frac{\widetilde{L}_{22}}{\varepsilon}\sqrt{\frac{dn}{d+n}}(\lambda\sqrt{n})\Big)$

Table 3: Linear minimization oracle: LMO and SFO calls complexity (LMO-CC and SFO-CC) of various methods for d-dimensional ℓ_1 norm constrained SVM with n training samples. SFO uses a batchsize of $b = o(n)$. SP+VR-MP combines ideas from Semi-Proximal [41] and Variance reduced [16] Mirror-Prox methods. Our MOLES outperforms other nonsmooth methods in LMO-CC while still maintaining $\mathcal{O}(1/\varepsilon^2)$ SFO-CC. Complexities of method based on smooth minimax reformulation adversely scale with n or d.

and

$$
\begin{aligned}
&[\text{LMO-CC} = \mathcal{O}(K\cdot T)\cdot\widehat{T}]\\
&= \mathcal{O}\Big(\Big[\frac{dn}{d+n}+\frac{\widetilde{L}_{22}}{\varepsilon}\sqrt{\frac{dn}{d+n}}(\lambda\sqrt{n})\Big]\cdot\Big[1+(\alpha+\frac{L_{22}^2}{\alpha})\frac{D_{\mathcal{X}}^2}{\varepsilon}\Big]\Big)\\
&= \mathcal{O}\Big(\Big[\frac{dn}{d+n}+\frac{\widetilde{L}_{22}}{\varepsilon}\sqrt{\frac{dn}{d+n}}(\lambda\sqrt{n})\Big]\cdot\Big[1+\frac{\lambda}{\sqrt{n}}+\frac{\widetilde{L}_{22}\,\lambda^2}{\varepsilon}\sqrt{\frac{dn}{d+n}}\Big]\Big)\\
&= \text{SFO-CC}+\mathcal{O}\Big(\frac{d\sqrt{n}\,\lambda}{d+n}+\frac{\widetilde{L}_{22}\,\lambda^2}{\varepsilon}\Big(\frac{dn}{d+n}\Big)^{\frac{3}{2}}+\frac{\widetilde{L}_{22}^2\,\lambda^3\sqrt{n}}{\varepsilon^2}\Big(\frac{dn}{d+n}\Big)\Big)
\end{aligned}
\tag{90}
$$

In very high dimensional regime ($n \ll d$) or very large-scale regime ($d \ll n$), LMO-CC of this smooth minimax reformulation is $\mathcal{O}(d)$ or $\mathcal{O}(n)$ larger than LMO-CC for MOLES. Further more the former method uses extra $\Theta(n)$ extra space for storing the dual variables.

Similar arguments hold for the nuclear norm constrained Matrix SVM [85], so that MOPES/MOLES outperforms other nonsmooth methods in some regime, where n or d is large and ε is relatively moderate, and complexities of smooth minimax reformulation based methods scales adversely with d and n. For this case, the gain in the actual wall-clock time would be even more stark than vector SVM due to the computation of SVD/largest eigenvalue, which is required for implementing PO/LMO.

E.2 SVM with hard constraints

Soft-margin SVM could be provided with some hard constraints [70], so that the classifier is forced to always predict the correct labels for a subset (of size k) of important "gold" training examples. This problem can be formulated as a nonsmooth constrained optimization problem with a large number of linear constraints, as follows

$$
\begin{aligned}
\min_{x\in\mathbb{R}^d}\quad & \frac{1}{n}\sum_{i=1}^{n}\max(0, 1-\langle x, a_i\rangle)\\
\text{subject to,}\quad & 1 \le \langle x, \widetilde{a}_j\rangle\ ,\ \forall j = 1,\ldots,k\\
& \|x\|_1 \le \lambda
\end{aligned}
\tag{91}
$$

We can solve this nonsmooth convex problem using first-order methods using projection onto hard constraints set: $\mathcal{X} = \{x \mid \|x\|_1 \le \lambda \text{ and } 1 \le \langle x, \widetilde{a}_j\rangle\ , \forall j = 1,\ldots,k\}$. This projection can be can be implemented using linear programming methods, however it is computationally costly. Therefore PO-CC efficiency is critical, and just as in the case of SVM with ℓ_1 norm constraint (Section E.1), our MOPES method achieves smallest $\mathcal{O}(1/\varepsilon)$ dimension-free PO-CC which is better than other competing methods. PO-CC and SFO-CC are the same as given in Table 2.

Note that, first-order methods using one projection [61] cannot be applied here, since they need the constraint set to be written in the functional form: $c(x) \leq 0$, such that $\rho \leq \|g\|$ for all $g \in \partial c(x)$, for some $\rho > 0$. This is not true for general set of linear constraints, where a pathological case can occur when two linear constraints have almost identical normal vectors.