Reusing Combinatorial Structure: Faster Iterative Projections over Submodular Base Polytopes

Anonymous Author(s)
Affiliation
Address
email

Abstract

Optimization algorithms such as projected Newton's method, FISTA, mirror descent and its variants enjoy near-optimal regret bounds and convergence rates, but suffer from a computational bottleneck of computing "projections" in potentially each iteration (e.g., $O(T^{1/2})$ regret of online mirror descent) [1, 2, 3, 4]. On the other hand, conditional gradient variants solve a linear optimization in each iteration, but result in suboptimal rates (e.g., $O(T^{3/4})$ regret of online Frank-Wolfe) [5, 6, 7, 8]. Motivated by this trade-off in runtime v/s convergence rates, we consider iterative projections of close-by points over widely-prevalent submodular base polytopes $B(f)$. We develop a toolkit to speed up the computation of projections using both discrete and continuous perspectives (e.g., [9, 10, 11]). We subsequently adapt the away-step Frank-Wolfe algorithm to use this information and enable early termination. For the special case of cardinality based submodular polytopes, we improve the runtime of computing certain Bregman projections by a factor of $\Omega(n/\log(n))$. Our theoretical results show orders of magnitude reduction in runtime in preliminary computational experiments.

1 Introduction

Though the theory of discrete and continuous optimization methods has evolved independently over the last many years, machine learning applications have often brought the two regimes together to solve structured problems such as combinatorial online learning over rankings and permutations [12, 13, 14, 15], shortest-paths [16] and trees [17, 18], regularized structured regression [6], MAP inference, document summarization [19] (and references therein). One of the most prevalent forms of constrained optimization in machine learning is the use of iterative optimization methods such as online stochastic gradient descent, mirror descent variants, projected Newton's method, conditional gradient descent variants, fast iterative shrinkage-thresholding algorithm (FISTA). These methods repeatedly compute two main subproblems: either a projection (i.e., a convex minimization) or a linear optimization in each iteration. The former class of algorithms is known as projection-based optimization methods (e.g., projected Newton's method, see Table 1), and they enjoy near-optimal regret bounds in online optimization and near-optimal convergence rates in convex optimization compared to projection-free methods. These projection-based methods however suffer form high computational complexity per iteration due to the projection subproblem [1, 2, 20, 21, 4, 22]. E.g., online mirror descent is near-optimal in terms of regret (i.e., $O(\sqrt{T})$) for most online learning problems, however it is computationally restrictive for large scale problems [3]. On the other hand, online Frank-Wolfe is computationally efficient, but has a suboptimal regret of $O(T^{3/4})$ [23, 8].

Discrete optimizers, in parallel, have developed beautiful characterizations of properties of convex minimizers over interesting polytopes which results in typically non-iterative exact algorithms (upto solution of a univariate equation) over combinatorial polytopes. This theory however has not been

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Algorithm	Subproblem solved	Steps for ϵ-error
Vanilla Frank-Wolfe [6]	LO over polytope	$O\left(\frac{LD^2}{\epsilon}\right)$
Away-steps Frank-Wolfe [7]	LO over polytope and active sets	$O\left(\kappa\left(\frac{D}{\delta}\right)^2 \log\frac{1}{\epsilon}\right)$
*Projected gradient descent [24]	Euclidean projection over polytope	$O\left(\kappa \log\frac{1}{\epsilon}\right)$
*Mirror descent (MD) [25]	Bregman Projection	$O\left(\kappa\nu^2 \log\frac{1}{\epsilon}\right)$
*Projected Newton's method [24]	Euclidean projection over polytope scaled by (approximate) Hessian	$O\left((\kappa\beta)^3 \log\frac{1}{\epsilon}\right)$
*Accelerated Proximal Gradient [26]	Euclidean projection over polytope	$O\left(\sqrt{\kappa} \log\frac{1}{\epsilon}\right)$
*Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) [27]	Euclidean projection over polytope	$O\left(\sqrt{\kappa} \log\frac{1}{\epsilon}\right)$

Table 1: Some iterative optimization algorithms which solve a linear or convex optimization problem in each iteration. Here, $\kappa := L/\mu$ is the condition number of the main optimization, ν is condition number of the mirror map used in MD, D is the diameter of the domain, δ is the pyramidal width, $\beta \geq 1$ measures on how well the Hessian is approximated. Starred algorithms have dimension independent optimal convergence rates.

properly integrated within the iterative optimization framework. Each subproblem within the above-mentioned iterative methods is typically solved from scratch, using a black-box subroutine, leaving a significant opportunity to speed-up "perturbed" subproblems using combinatorial structure. Motivated by these trade-offs in convergence guarantees and computational complexity, we ask if:

Is it possible to speed up iterative subproblems of computing projections over combinatorial polytopes by reusing structural information from previous minimizers?

Let the overarching optimization problem be $(P1)$ where in each subproblem in iteration t, a point y_t needs to be projected over a constrained set $P \subseteq \mathbb{R}^n$ to find the iterate x_t. Typically, in regularized optimization methods such as mirror descent and its variants, iterates x_t are obtained by walking a small step in some approximation to gradient of the loss functions, and projecting to P. We will refer to a subproblem of computing a single projection as (P2), where an iterative method might compute a series of iterates $z^{(1)}, \ldots, z^{(k)}$ to converge to the exact projection x_t (depicted in Figure 1 (left, middle)). We would like to use combinatorial structure of $x_1, \ldots, x_{t-1}, z^{(1)}, \ldots, z^{(t-1)}, y_1, \ldots, y_t$ to speed up the computation of x_t. To the best of our knowledge, we are the first to consider using the combinatorial structure of previously projected points. To capture a broad class of interesting combinatorial polytopes, we focus on submodular base polytopes.

Submodularity is a discrete analogue of convexity, and captures the notion of diminishing returns. Submodular polytopes have been used in a wide variety of online and machine learning applications (see Table 2 in appendix). A typical example is when $B(f)$ is permutahedron, a polytope whose vertices are the permutations of $\{1, \ldots, n\}$, and is used for learning over rankings. Other machine learning applications include learning over spanning trees to reduce communication delays in networks, [13]), permutations to model scheduling delays [14], and k-sets for principal component analysis [28]. Other example applications of convex minimization over submodular polytopes include computation of densest subgraphs [29], bounds on the partition function of log-submodular distributions [30] and distributed routing [31].

Though (Bregman) projections can be computed efficiently in closed form for certain simple polytopes (such as the n-dimensional simplex), the submodular base polytopes pose a unique challenge since they are defined using 2^n linear inequalities [32], and their exist instances with exponential extension complexity as well [33] (i.e., there exists no extended formulation with polynomial number of constraints for some submodular polytopes). Existing combinatorial algorithms for minimizing separable convex functions over base polytopes typically require iterative submodular function minimizations (SFM) [10, 9, 15], which are quite expensive in practice [34]. However, these combinatorial methods highlight important structure in convex minimizers which can be exploited to speed up the continuous optimization methods. In this paper, we bridge discrete and continuous optimization insights to speed up projections over submodular polytopes as follows:

(i) *Bregman Projections over cardinality-based polytopes*: We first show that the results of [35] extend to all cardinality-based submodular polytopes (where $f(S) = g(|S|)$ for some concave function g) to give an $O(n \log n)$-time algorithm for computing a Bregman projection, improving the current best-known $O(n \log n + n^2)$ algorithm [15], in Section 3. These are exact algorithms (up to the solution of a univariate equation), compared to iterative continuous optimization methods.

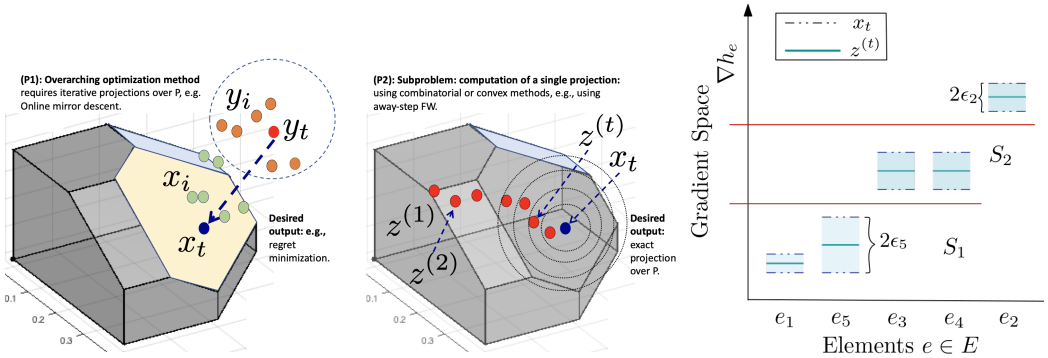

Figure 1: Left: **(P1)** represents an iterative optimization algorithm that computes projections x_i for points y_i in every iteration (see Table 1). Middle: **(P2)** represents subproblem of computing a single projection of y_t using an iterative method with easier subproblems, e.g., away-step Frank-Wolfe where $z^{(i)}$ are iterates during a single run of AFW and converge to projection x_t (of y_t). The goal is speed up the subproblems using both past projections $x_1, \ldots, x_{t-1}$, as well as iterates $z^{(1)}, \ldots, z^{(k)}$. Right: We show how to detect tight sets S_1 and S_2 for close-by points by looking at the maximum error in $\nabla h(x_t)$ (tools INFER1, INFER2).

(ii) *Toolkit for Exploiting Combinatorial Structure:* We next develop a toolkit (tools **T1-T6**) of provable ways for detecting tight inequalities, reusing active sets, restrict to optimal inequalities and rounding approximate projections to enable early termination:

(a) INFER: We first show that for "close" points $y, \tilde{y}$ where the projection $\tilde{x}$ of $\tilde{y}$ on $B(f)$ is known, we can infer some tight sets for x using the structure of $\tilde{x}$ without explicitly computing x (**T1**). Further, suppose that we use a convergent iterative optimization method to solve the projection subproblem (P2) for y_t to compute x_t, then given any iterate $z^{(k)}$ in such a method, we know that $\|z^{(k)} - x_t\| \leq \epsilon_k$ is bounded for strongly convex functions. Using this, we show how to infer some tight sets (provably) for x_t for small enough ϵ_k (**T2**), in Section 4.1.

(b) REUSE: Suppose we compute the projection $\tilde{x}$ of $\tilde{y}$ on $B(f)$ using AFW, and obtain an active set of vertices A for $\tilde{x}$. Our next tool (**T3**) gives conditions under which A is also an active set for x. Thus, x can be computed by projecting y onto $\mathrm{Conv}(A)$ instead of $B(f)$ in Section 4.2.

(c) RESTRICT: While solving the subproblem (P2), we show that discovered tight inequalities for the optimum solution can be incorporated into the linear optimization (LO) oracle over submodular polytopes, in Section 4.2. We modify Edmonds' greedy algorithm to do LO over any lower dimensional face of the submodular base polytope, while maintaining its efficient $O(n \log n)$ running time. Note that in general, while there may exist efficient algorithms to do LO over the entire polytope (e.g. shortest-paths polytope), restricting to lower dimensional faces may not be trivial.

(d) RELAX and ROUND: We give two approaches for rounding an approximate projection to an exact one in Section 4.3, which helps terminate iterative algorithms early. The first method uses INFER to iteratively finds tight sets at projection x_t, and then checks if we have found all such tight sets defining the optimal face by projecting onto the affine space of tight inequalities. If the affine projection x_0 is feasible in the base polytope, then this is optimal projection. The second one rounding tool is algebraic in nature, and useful for base polytopes of integral submodular functions. It only requires a guarantee that the approximate projection be within a (Euclidean) distance of $1/(2|E|^2)$ to the optimal for Euclidean projections.

(iii) *Adaptive Away-Step Frank-Wolfe (A^{2}FW):* We combine the above-mentioned tools to give a novel adaptive away-step Frank-Wolfe variant in Section 5. We first use INFER (**T1**) to detect tight inequalities using past projections of x_{t-1}. Next, we start away-step FW to compute projection x_t in iteration t by REUSING the optimal active set from computation of x_{t-1}. During the course of A^{2}FW, we INFER tight inequalities iteratively using distance of iterates $z^{(t)}$ from optimal (**T2**). To adapt to discovered tight inequalities, we use the modified greedy oracle (**T4**). We check in each iteration if RELAX allows us to terminate early (**T5**). In case of Euclidean projections, we also detect if rounding to lattice of feasible points is possible (**T6**). We finally show an order of magnitude reduction in running time of online mirror descent by using A^{2}FW as a subroutine for computing projections in Section 5.1 and conclude with limitations in Section 5.2.

Although we show that our toolkit can help speed up online mirror descent, the tools are more general and can be used to speed up other combinatorial algorithms like Groenvelt's Decomposition algorithm, Fujishige's minimum norm point, and Gupta et. al's Inc-Fix [36, 10, 15]. A special case

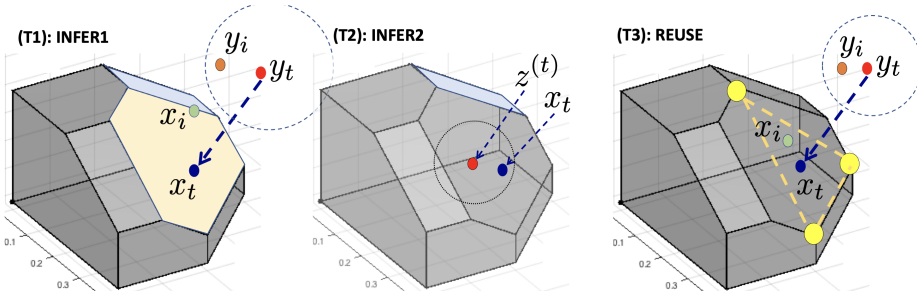

Figure 2: Toolkit to Speed Up Projections: INFER1 **(T1)** uses previously projected points to infer tight sets defining the optimal face of x_t and is formally described by Theorem 3 (see also Figure 1-Right). On the other hand, INFER2 **(T2)** uses the closeness of iterates $z^{(t)}$ of an algorithm solving the projection subproblems (e.g. AFW) to the optimal x_t, to find more tight sets at x_t (than those found by **(T1)** (Lemma 4). REUSE **(T3)** uses active sets of previous projections computed using AFW (Lemma 1).

of our rounding approach is used within the Fujishige-Wolfe minimum norm point algorithm to find approximate submodular function minimizers [37, 38].

Minimizing separable convex functions over submodular base polytopes was first studied by Fujishige [11] in 1980, followed by a series of results by Groenevelt [10], Hochbaum [39], and recently by Nagano and Aihara [9], and Gupta et. al. [40]. Each of these approaches considers different problem classes, but uses $O(n)$ calls to either parametric submodular function or submodular function minimization, with each computation discovering a cut and reducing the subproblem size for future iterations. Both subroutines, however, can be expensive in practice. Frank-Wolfe variants on the other hand have attempted at incorporating geometry of the problem in various ways: restricting FW vertices to norm balls [41, 42, 43], or restricting away vertices to best possible active sets [44], or prioritizing in-face steps [45], or theoretical results such as [46] and [47] show that FW variants must use active sets that containing the optimal solution after crossing a polytope dependent radius of convergence. These results, however, do not use combinatorial properties of previous minimizers or detect tight sets with provable guarantees and round to those. To the best of our knowledge, we are the first to adapt away-step Frank-Wolfe to consider combinatorial structure from previous projections.

2 Preliminaries

Consider a compact and convex set $\mathcal{X} \subseteq \mathbb{R}^n$, and let $\mathcal{D} \subseteq \mathbb{R}^n$ be a convex set such that $\mathcal{X}$ is included in its closure. A mirror map $\phi : \mathcal{D} \to \mathbb{R}$ is a strictly (or $\mu-$ strongly) convex* and continuously differentiable function over $\mathcal{D}$, and satisfies additional properties of divergence of the gradient on the boundary of $\mathcal{D}$, i.e., $\lim_{x \to \partial\mathcal{D}} \|\nabla\phi(x)\| = \infty$ (see [1, 21] for more details). We further assume that the mirror map ϕ is *uniformly* separable: $\phi = \sum_e \phi_e$ where $\phi_e : \mathcal{D}_e \to \mathbb{R}$ is the same function for all $e \in E$.We use $\|\cdot\|$ to denote the Euclidean norm unless otherwise stated. We say ϕ is $L-$ smooth if $\|\nabla\phi(x) - \nabla\phi(z)\| \leq L\|x - z\|$ for all $x, z \in \mathcal{D}$. The Bregman divergence generated by a mirror map ϕ is defined as $D_\phi(x, y) := \phi(x) - \phi(y) - \langle \nabla(y), x - y\rangle$. For example, the Euclidean mirror map is given by $\phi = \frac{1}{2}\|x\|^2$, for $\mathcal{D} = \mathbb{R}^E$ and is 1-strongly convex with respect to the ℓ_2 norm. In this case $D_\phi(x, y) = \frac{1}{2}\|x - y\|_2^2$ reduces to the Euclidean squared distance (see Table 3).

Submodularity and Convex Minimizers over Base Polytopes

Let $f : 2^E \to \mathbb{R}$ be a submodular function defined on a ground set of elements E, i.e. $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$ for all $A, B \subseteq E$. Assume without loss of generality that $f(\emptyset) = 0$, $f(A) > 0$ for $A \neq \emptyset$ and that f is monotone†. We denote by EO the time taken to evaluate f on any set. For $x \in \mathbb{R}^E$, we use the shorthand $x(S)$ for $\sum_{e \in S} x(e)$, and by both $x(e)$ and x_e we mean the value of x on element e. Given such a submodular function f, the polymatroid is defined as $P(f) = \{x \in \mathbb{R}_+^E : x(S) \leq f(S) \, \forall \, S \subseteq E\}$ and the base polytope as $B(f) = \{x \in \mathbb{R}_+^E : x(S) \leq f(S) \, \forall \, S \subset E, \; x(E) = f(E)\}$ [49]. A typical example is when f is the rank function of a matroid, and the corresponding base polytope corresponds to the convex hull of its bases (see Table 2).

*A differentiable function h is said to be strictly convex over domain $\mathcal{D}$ if $h(y) \geq h(x) + \langle \nabla h(x), y - x\rangle$ for all $x, y \in \mathcal{D}$. Moreover, a differentiable function h is said to be μ-strongly convex over domain $\mathcal{D}$ with respect to a norm $\|\cdot\|$ if $h(y) \geq h(x) + \langle \nabla h(x), y - x\rangle + \frac{\mu}{2}\|y - x\|^2$ for all $x, y \in \mathcal{D}$.

†f is monotone if $f(A) \leq f(B) \; \forall A \subseteq B \subseteq E$. For any non-negative submodular function f, we can consider a corresponding monotone submodular function $\bar{f}$ such that $P(f) = P(\bar{f})$ (see Section 44.4 of [48].

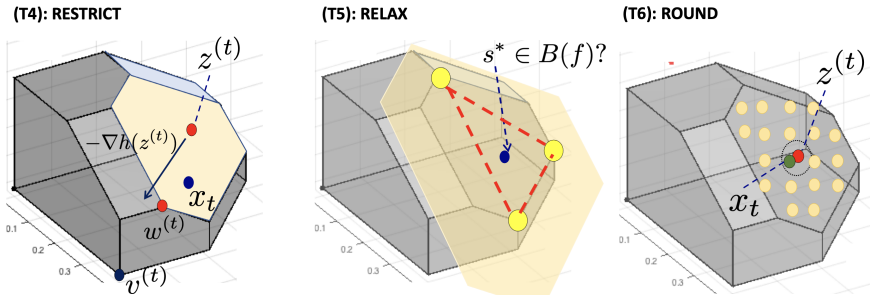

Figure 3: Proposed Toolkit (contd): RESTRICT **(T4)** restricts the LO oracle in AFW to the lower dimensional face defined by the tight cuts found by **(T1)** and **(T2)** (Theorems 5, 6). Note that the restricted vertex $w^{(t)}$ gives better progress than the orginal FW vertex $v^{(t)}$. RELAX **(T5)** enables early termination of algorithms solving projection subproblems (e.g. AFW) as soon as all tight sets defining the optimal face are found (Theorem 2). Finally, ROUND **(T6)** gives an integral rounding approach for special cases (Lemma 3).

Consider a submodular function $f : 2^E \to \mathbb{R}$ with $f(\emptyset) = 0$, and let $c \in \mathbb{R}^n$. Edmonds gave the greedy algorithm to perform linear optimization $\max c^T x$ over submodular base polytopes for monotone submodular functions. Order elements in $E = \{e_1, \ldots, e_n\}$ such that $c(e_i) \geq c(e_j)$ for all $i < j$. Define $U_i = \{e_1, \ldots, e_i\}$, $i \in \{0, \ldots, n\}$, and let $x^*(e_j) = f(U_j) - f(U_{j-1})$. Then, $x^* = \max_{x \in B(f)} c^\top x$. Further, for convex minimizers of strictly convex and separable functions, we will often use the following characterization of the convex minimizers.

Theorem 1 (Theorem 4 in [15])**.** *Consider any continuously differentiable and strictly convex function $h : \mathcal{D} \to \mathbb{R}$ and submodular function $f : 2^E \to \mathbb{R}$ with $f(\emptyset) = 0$. Assume that $B(f) \cap \mathcal{D} \neq \emptyset$. For any $x^* \in \mathbb{R}^E$, let $F_1, F_2, \ldots, F_l$ be a partition of the ground set E such that $(\nabla h(x^*))_e = c_i$ for all $e \in F_i$ and $c_i < c_l$ for $i < l$. Then $x^* = \arg\min_{x \in B(f)} h(x)$ if and only if x^* lies on the face H^* of $B(f)$ given by $H^* := \{x \in B(f) \mid x(F_1 \cup F_2 \cup \cdots \cup F_i) = f(F_1 \cup F_2 \cup \cdots \cup F_i) \, \forall \, 1 \leq i \leq l\}$.*

To see why this holds, note that the first order optimality condition states that at the convex minimizer x^*, we must have that $\nabla h(x^*) = c$ is minimized as a linear cost function over $B(f)$, i.e., $c^T x^* \leq c^T z$ for all $z \in B(f)$. However, linear optimization over submodular base polytopes is given by Edmonds' greedy algorithm, which simply raises elements of minimum cost as much as possible. This gives us the levels of the partial derivatives of x^* as $F_1, F_2, \ldots F_k$, which form the optimal face H^* of x^*. For separable convex functions like Bregman divergences (in Table 3), we can thus compute x^* by solving univariate equations in a single variable if the tight sets $F_1, \ldots, F_k$ of x^* are known. We equivalently refer to corresponding inequalities $x(F_i) = f(F_i)$ as the optimal tight inequalities.

3 Bregman Projections over Cardinality-based Submodular Polytopes

We first improve the runtime of exact combinatorial algorithms for computing uniform Bregman projections over cardinality-based submodular polytopes. The key observation that allows us to do that is the following generalization of Lim and Wright's result [35]:

Theorem 2 (Dual of projection is isotonic optimization)**.** *Let $f : 2^E \to \mathbb{R}$ be a cardinality-based monotone submodular function, that is $f(S) = g(|S|)$ function for some nondecreasing concave function g. Let $c_i := g(i) - g(i-1)$ for all $i \in [E]$. Let $\phi : \mathcal{D} \to \mathbb{R}$ be a strictly convex and uniformly seperable mirror map. Let $B(f) \cap \mathcal{D} \neq \emptyset$ and consider any $y \in \mathbb{R}^n$. Assume without loss of generality that $y_1 \geq, \ldots, \geq y_n$. Then, the following problems are primal-dual pairs*

$$(P) \quad \begin{array}{l} \min \; D_\phi(x, y) \\ \textit{subject to } x \in B(f) \end{array} \qquad\qquad (D) \quad \begin{array}{l} \max \; -D^*_\phi(z, y) + z^T c \\ \textit{subject to } z_1 \leq \cdots \leq z_n \end{array}. \tag{1}$$

Moreover, from a dual optimal solution z^*, we can recover the optimal primal solution x^* via the following equation: $\nabla \phi(x^*) = z^* + \nabla \phi(y)$.

The proof of this result follows by the Fenchel Duality and exploiting the structure of cardinality-based polytopes to obtain the dual problem (D) (see Appendix C). Problem (D) in (1) is in fact a separable isotonic optimization problem[‡], which highlights an interesting connection between projections on

[‡]A separable isotonic optimization problem is of the form $\min \sum_{i=1}^n h_i(x_i)$ subject to $x_1 \leq x_2 \leq \cdots \leq x_n$, where h_i are univariate strictly convex functions

cardinality-based polytopes [50, 51, 19]. In particular, when $\phi(x) = \frac{1}{2}\|x\|^2$, the dual problem (D) in (1) becomes the following $\min_z\{\frac{1}{2}\|z-(c-y)\|^2 \mid z_1 \leq \cdots \leq z_n\}$ isotonic regression problem. Learning over projections is therefore dual to performing isotonic regression for perturbed data sets. Using the same algorithm as Lim and Wright's, i.e., the Pool Adjacent Violators (PAV) [52], we can solve the dual problem (D) with a faster running time of $O(n \log n + nEO)$ compared to $O(n^2 + nEO)$ of [40]. We include the details about the algorithm and correctness in Appendix C. It is worth noting that linear optimization over $B(f)$ also has a running time of $O(n \log n + nEO)$ using Edmonds' greedy algorithm [32]. Therefore, for cardinality-based polytopes, using a combinatorial algorithm is better than any iterative optimization method.

4 Toolkit to Adapt to Previous Combinatorial Structure

In the previous section, we gave an $O(n \log n)$ exact algorithm for computing Bregman projections over cardinality-based polytopes. However, the pool-adjacent-violator algorithm is very specific to the cardinality-based polytopes and does not extend to general submodular polyhedra. In this section, we construct a toolkit to speed up iterative projections over general submodular polytopes.

4.1 INFER tight inequalities

We first present our INFER tool **T1** that recovers some tight inequalities of projection of $\tilde{y}$ by using the tight inequalities of the projection of a close-by perturbed point $y \in \mathbb{R}^n$. The motivation of this result stems from the fact that projection-based optimization methods often move slowly, i.e., points $y, \tilde{y}$ to be projected are often close to each other, and so are their corresponding projections $x, \tilde{x}$. Our first result is specifically for Euclidean projections.

Theorem 3 (Recovering tight sets from previous projections **(T1)**). *Let $f : 2^E \to \mathbb{R}$ be a monotone submodular function with $f(\emptyset) = 0$. Further, let y and $\tilde{y} \in \mathbb{R}^E$ be such that $\|y - \tilde{y}\| \leq \epsilon$, and $x, \tilde{x}$ be the Euclidean projections of $y, \tilde{y}$ on $B(f)$ respectively. Let $F_1, F_2, \ldots, F_k$ be a partition of the ground set E such that $x_e - y_e = c_i$ for all $e \in F_i$ and $c_i < c_l$ for $i < l$. If $c_{j+1} - c_j > 4\epsilon$ for some $j \in [k-1]$, then the set $S = F_1 \cup \cdots \cup F_j$ is also a tight set for $\tilde{x}$, i.e. $\tilde{x}(S) = f(S)$.*

Note that $x_e - y_e$ is the partial derivative of the distance function from y at x. The proof shows that for $e \in E$, $\tilde{x}_e - \tilde{y}_e$ is close to $x_e - y_e$ and relies on the smoothness and non-expansivity of Euclidean projection. This helps us infer that the relative order of coordinates in $\tilde{x} - \tilde{y}$ (i.e., the coordinate-wise partial derivatives) is close to the relative order of coordinates in $x - y$. This relative order then determines tight sets for x, due to first-order optimality characterization of Theorem 1. See Appendix D.2 for a complete proof, where we also generalize the theorem to any Bregman projection that is L-smooth and non-expansive. In Section 5.1, we will show that this theorem infers a near-optimal number of tight inequalities computationally (see Figure 4-left).

Next, consider the subproblem (P2) of computing the projection x_t of a point y_t. Let $z^{(k)}$ be the iterates in the subproblem that are convergent to x_t. The points $z^{(k)}$ grow progressively closer to x_t, and our next tool INFER **T2** helps us recover tight sets for x_t using the gradients of points $z^{(k)}$.

Theorem 4 (Adaptively inferring the optimal face **(T2)**). *Let $f : 2^E \to \mathbb{R}$ be a monotone submodular with $f(\emptyset) = 0$, $h : \mathcal{D} \to \mathbb{R}$ be a strictly convex and L-smooth function, where $B(f) \cap \mathcal{D} \neq \emptyset$. Let $x := \arg\min_{z \in B(f)} h(z)$. Consider any $z \in B(f)$ such that $\|z - x\| \leq \epsilon$. Let $\tilde{F}_1, \tilde{F}_2, \ldots, \tilde{F}_k$ be a partition of the ground set E such that $(\nabla h(z))_e = \tilde{c}_i$ for all $e \in \tilde{F}_i$ and $\tilde{c}_i < \tilde{c}_l$ for $i < l$. Suppose $\tilde{c}_{j+1} - \tilde{c}_j > 2L\epsilon$ for some $j \in [k-1]$. Then, $S = F_1 \cup \cdots \cup F_j$ is tight for x, i.e. $x(S) = f(S)$.*

The proof of this theorem, similar to Theorem 3, relies on the L-smoothness of h to show that the relative order of coordinates in $\nabla h(x_t)$ is close to the relative order of coordinates in $\nabla h(z^{(k)})$, which helps infer some tight sets for x. See Appendix D.2 for a complete proof. Note that while Theorem 3 is restricted to Euclidean projections, Theorem 4 applies to smooth strictly convex functions.

4.2 ReUse and Restrict

We now consider computing a single projection (P2) using Frank-Wolfe variants, that have two main advantages: (i) they maintain an active set for their iterates as a (sparse) convex combination of vertices, (ii) they only solve LO every iteration. Our first REUSE tool gives conditions under which a new projection has the same active set A as a point previously projected, which allows for a faster projection onto the convex hull of A (proof is included in Appendix D.2).

Algorithm 1 Greedy algorithm for faces of $B(f)$

Input: Monotone submodular $f : 2^E \to \mathbb{R}$, objective $c \in \mathbb{R}^n$, face $F = \{x \in B(f) \mid x(S_i) = f(S_i)$, where $S_1 \subset \cdots \subset S_k = E$ where S_i form a chain$\}$.

1: Consider an ordering on the ground set of elements $E = \{e_1, \ldots, e_n\}$ such that (i) it respects the given chain, i.e., $S_i = \{e_1, \ldots, e_{s_i}\}$ for all i, and (ii) each set $S_i \setminus S_{i-1} = \{e_{s_{i-1}+1}, \ldots, e_{s_i}\}$ is in decreasing order of cost, i.e., $c(e_{s_{i-1}+1}) \geq \ldots \geq c(e_{s_i})$.

2: Let $x^*(e) := f(\{e_1, \ldots, e_j\}) - f(\{e_1, \ldots, e_{j-1}\})$, for $i \in [n]$.

Return: $x^* = \arg\max_{x \in F} \langle c, x \rangle$

Lemma 1 (Reusing active sets **(T3)**). *Let $\mathcal{P} \subseteq \mathbb{R}^n$ be a polytope with vertex set $\mathrm{vert}(\mathcal{P})$. Let x be the Euclidean projection of some $y \in \mathbb{R}^n$ on $\mathcal{P}$. Let $\mathcal{A} = \{v_1, \ldots, v_k\} \subseteq \mathrm{vert}(\mathcal{P})$ be an active set for x, i.e., $x = \sum_{i \in [k]} \lambda_i v_i$ for $\|\lambda\| = 1$ and $\lambda > 0$. Let F be the minimal face of x and $\Delta := \min_{v \in \partial \mathrm{conv}(\mathcal{A})} \|x - v\|$ be the minimum distance between x and the boundary of $\mathrm{conv}(\mathcal{A})$. Then, $\mathcal{A}$ is also an active set for the Euclidean projection of any point $\tilde{y} \in \mathbb{B}_\Delta(y) \cap \mathrm{span}(F)$, where $\mathbb{B}_\Delta(y) = \{\tilde{y} \in \mathbb{R}^n \mid \|\tilde{y} - y\| \leq \Delta\}$ is a closed ball of radius Δ.*

In the previous section, we presented combinatorial tools to detect tight sets at the optimal solution. We now use our RESTRICT tool to strengthen the LO oracle in FW by doing restricting it to the lower dimensional faces defined by the tight sets we found (instead of doing LO over the whole polytope). Note that doing linear optimization over lower dimensional faces of polytopes, in general, is significantly harder (e.g., for shortest paths polytope). For submodular polytopes however, we show that we can do LO over any face of $B(f)$ efficiently using a modified greedy algorithm. Given a set of tight inequalities, one can uncross these to form a *chain* of tight sets, i.e., any facet of $B(f)$ can be written using a chain of subsets that are tight (see e.g. Section 44.6 in [53]). Given such a chain, our modified greedy algorithm then orders the cost vector in decreasing order so that it respects a given tight laminar family of subsets. Once it has that ordering, it proceeds in the same way as in Edmonds' greedy algorithm [32]. We include the full proof in Appendix D.2).

Theorem 5 (Linear optimization over faces of $B(f)$ **(T4)**). *Let $f : 2^E \to \mathbb{R}$ be a monotone submodular function with $f(\emptyset) = 0$. Further, let $F = \{x \in B(f) \mid x(S_i) = f(S_i)$ for $S_i \in \mathcal{S}\}$ be a face of $B(f)$, where $\mathcal{S} = \{S_1, \ldots S_k | S_1 \subseteq S_2 \ldots \subseteq S_k\}$. Then the modified greedy algorithm (Alg. 1) returns $x^* = \arg\max_{x \in F} \langle c, x \rangle$ in $O(n \log n + nEO)$ time.*

4.3 Rounding

Approximation errors in projection subproblems often impact (adversely) the convergence rate of the overarching iterative method unless the errors decrease at a sufficient rate [54, 55]. Our goal in this section is to detect if all tight sets at the optimum have been inferred, and enable early termination by computing the exact minimizer. In 2020, [56] gave primal gap bounds after which away-step FW reaches the optimal face, assuming strict complementarity assumption which need not hold even for computing a Euclidean projection. Further, [57], showed that there exists some convergence radius R such that for any iterate $z^{(t)}$ of AFW, if $\|z^{(t)} - x^*\| \leq R$, then any active set for $z^{(t)}$ must contain x^*, but parameter R existential and is non-trivial to compute. We complement these results by rounding our approximate projections to an exact one based on structure in partial derivatives.

Suppose that we have a candidate chain $\mathcal{S} = \{S_1, \ldots S_k\}$ of tight sets (e.g., using INFER). We observe that if the affine minimizer over $\mathcal{S}$, i.e., $\tilde{x} := \arg\min\{h(x) \mid x(S) = f(S) \, \forall S \in \mathcal{S}\}$ is feasible in $B(f)$, then this is indeed the optimum solution $\tilde{x} = x^*$.

Lemma 2 (Rounding to optimal face **(T5)**). *Let $f : 2^E \to \mathbb{R}$ be a monotone submodular function with $f(\emptyset) = 0$. Let $h : \mathcal{D} \to \mathbb{R}$ be a strictly convex, where $B(f) \cap \mathcal{D} \neq \emptyset$. Let $x^* := \arg\min_{x \in B(f)} h(x)$ and let $\mathcal{S} = \{S_1, \ldots S_k\}$ contain some of the tight sets at x^*, i.e. $x^*(S_i) = f(S_i)$ for all $i \in [k]$. Further, let $\tilde{x} := \arg\min\{h(x) \mid x(S) = f(S) \, \forall S \in \mathcal{S}\}$. If $\tilde{x} \in B(f)$, then $\tilde{x} = x^*$.*

The proof of this lemma can be found in Appendix D.3, and as a subroutine in Appendix B. To check whether $\tilde{x} \in B(f)$ in general requires an expensive submodular function minimization, but we instead we just check whether $\tilde{x}$ is in the convex hull of $\{v^{(1)}, \ldots, v^{(t)}\}$, where $v^{(i)}$ are the FW vertices of $B(f)$ that we have computed in Line 3 of Algorithm A^{2}FW up to iteration t. Using [57], we know that there will be a point at which the optimal solution is contained in the current active set.

We now present our second rounding tool ROUND for base polytopes of integral submodular functions. It only requires a guarantee that the approximate projection be within a (Euclidean) distance of $1/(2|E|^2)$ to the optimal projection:

Lemma 3 (Combinatorial Integer Rounding Euclidean Projections **(T6)**). *Let $f : 2^E \to \mathbb{Z}$ be a monotone submodular function with $f(\emptyset) = 0$. Consider $y \in \mathbb{Z}^E$ and let $h(x) = \frac{1}{2}\|x - y\|^2$. Let $x^* := \arg\min_{x \in B(f)} h(x)$. Consider any $x \in B(f)$ such that $\|x - x^*\| < \frac{1}{2|E|^2}$. Define $Q := \mathbb{Z} \cup \frac{1}{2}\mathbb{Z} \cup \ldots \cup \frac{1}{|E|}\mathbb{Z}$, and for any $r \in \mathbb{R}$, let $q(r) := \arg\min_{s \in Q} |r - s|$. Then, $q(x_e)$ is unique for all $e \in E$, and the optimal solution is given by $x^*_e = q(x_e)$ for all $e \in E$.*

This rounding algorithm runs in time $O(n^2 \log n)$ and is given in Algorithm 4 in Appendix B. The proof proceeds by showing that $x^*_e \in S$ for all $e \in E$, and that the distance between two points in S is at least $\frac{1}{|E|^2}$, so that one can always round to x^* correctly (complete proof is in Appendix D.3).

5 Adaptive Away-steps Frank-Wolfe (A^{2}FW)

We are now ready to present our Adaptive AFW (Alg. 5) by combining tools presented in the previous section. First using the INFER1, we detect some of the tight sets $\mathcal{S}$ at the optimal solution before even running A^{2}FW, and accordingly warm-start A^{2}FW with z_0 in the tight face of $\mathcal{S}$. A^{2}FW operates similar to the away-step Frank-Wolfe, but during the course of the algorithm it restricts to tight faces as it discovers them (using INFER2), adapts the linear optimization oracle (using RESTRICT), and attempts to round to optimum (using ROUND, RELAX). To apply INFER2 (subroutine included as Algorithm 2), we consider an iteration t of A^{2}FW, where we have computed the FW gap $g_t^{\text{FW}} := \max_{v \in B(f)} \left\langle -\nabla h(z^{(t)}), v - z^{(t)} \right\rangle$ (see line 5 in Alg. 5). When h is μ-strongly convex:

$$\frac{\mu}{2}\|z^{(t)} - x^*\|^2 \leq h(z^{(t)}) - h(x^*) \leq \max \left\langle -\nabla h(z^{(t)}), v - z^{(t)} \right\rangle = g_t^{\text{FW}}, \tag{2}$$

and so $\|z^{(t)} - x^*\| \leq \frac{2}{\mu}\sqrt{g_t^{\text{FW}}}$. Let $\tilde{F}_1, \tilde{F}_2, \ldots, \tilde{F}_k$ be a partition of the ground set E such that $(\nabla h(z^{(t)}))_e = \tilde{c}_i$ for all $e \in F_i$ and $\tilde{c}_i < \tilde{c}_l$ for all $i < l$. If $\tilde{c}_{j+1} - \tilde{c}_j > 4L\sqrt{g_t^{\text{FW}}}/\mu$ for some $j \in [k-1]$, then Theorem 4 implies that $S = F_1 \cup \cdots \cup F_j$ is tight for x^*, i.e. $x^*(S) = f(S)$.

Overall in A^{2}FW, we maintain a set $\mathcal{S}$ containing all such tight sets S at the optimal solution that we have found so far. We use those tight sets as follows: (i) we restrict our LO oracle to the lower dimensional face we identified using the modified greedy algorithm (RESTRICT- **(T4)**). (ii) We use our RELAX (**(T5)**) tool to check weather we have identified all the tight-sets defining the optimal face (Lemma 2). If yes, then we round the current iterate to the optimal face and terminate the algorithm early. For (Euclidean) projections over an integral submodular polytope, we can also use our ROUND **(T6)** tool to round an iterate close to optimal without knowing the tight sets. Each time we find a new tight inequality, we start A^{2}FW from a vertex on the tight face. Since this can only happen at most n times, it does not impact the convergence rate. Pseudocode of A^{2}FW is included in Algorithm 5.

Convergence Rate: As depicted in **(T4)** in Figure 3, restricting FW vertices to the optimal face results in better progress per iteration during the latter runs of the algorithm. The convergence rate of A^{2}FW depends on a geometric constant δ called the pyramidal width [7]. This constant is computed over the worst case face of the polytope. By iterative restricting the linear optimization oracle to optimal faces, we improve this worst case dependence in the convergence rate (proof in Appendix F):

Theorem 6 (Convergence rate of A^{2}FW). *Let $f : 2^E \to \mathbb{R}$ be a monotone submodular function with $f(\emptyset) = 0$ and f monotone. Consider any separable convex function $h = \sum_{e \in E} h_e : B(f) \to \mathbb{R}$ that is μ-strongly convex and L-smooth. Let $x^* := \arg\min_{x \in B(f)} h(x)$. Consider iteration t of A^{2}FW and let $\mathcal{S}$ be the tight sets found up to iteration t and $F(\mathcal{S})$ be face defined by these tight sets. Then, the primal gap $w(z^{(t+1)}) := h(z^{(t+1)}) - h(x^*)$ of A^{2}FW decreases geometrically at each step that is not a drop step[§] nor a restart step: $w(z^{(t+1)}) \leq \left(1 - \frac{\mu \rho^2_{F(\mathcal{S})}}{4LD^2}\right) w(z^{(t)})$, where D is the diameter of $B(f)$ and $\rho_{F(\mathcal{S})}$ is the pyramidal width of $B(f)$ restricted to $F(\mathcal{S})$. Moreover, the number of drop and restart steps up to iteration t is at most $t/2$ and $|E|$ respectively.*

[§] A drop step is when we take an away step with a maximal step size so that we drop a vertex from the current active set, i.e. $\gamma_t = \gamma_{\max}$

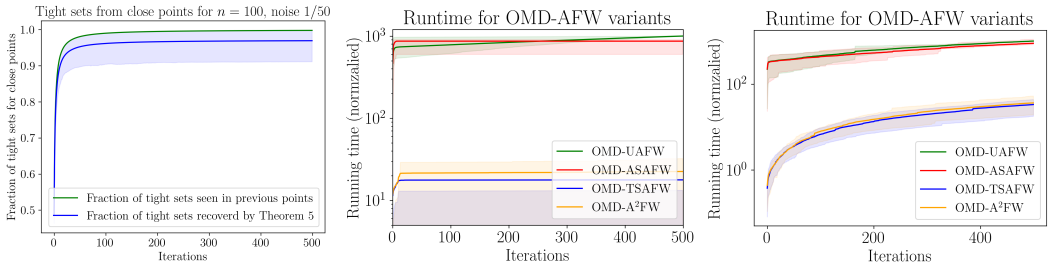

Figure 4: (left) 15-85% percentile plot of fraction of tight sets inferred by using INFER1 (blue) v/s highest number of tight sets common for ith iterate compared to previous $i-1$ iterates (in green) for close points generated randomly using Gaussian noise, over 500 runs. (middle) 25-75 percentile plots of normalized run times for OMD-AFW variants for first loss setting averaged over 20 runs. (right) 25-75 percentile plots of normalized run times for OMD-AFW variants with second loss setting averaged over 20 runs.

Note that $\rho_{F(\mathcal{S})}$ can be strictly larger than the worst-case pyramidal width over the entire polytope, e.g., $\delta \geq 1/\sqrt{dim(F)} > 1/\sqrt{n}$ for lower dimensional faces F of the unit cube [7].

5.1 Computations

We first consider 500 clouds of points generated by perturbing a cloud-specific random point using multivariate Gaussian noise with mean zero and standard deviation $\epsilon = 1/50$. We compute the projections of each point in the cloud exactly, and plot percentile plots in Figure 4-left. The fraction of tight inequalities for each point y_i that were already tight for some other previous point $y_0, \ldots, y_{i-1}$ is in green, the fraction of tight sets for y_i inferred by using Theorem 3 is in blue. Note that our theoretical results give almost tight computational results, that is, we can recover most of the tight sets common between close points using Theorem 3.

Next, motivated by the trade-off in regret versus time for online mirror descent and online Frank-Wolfe variants, we conduct an experiment on the permutahedron P with $n = 25$ elements. We consider a time horizon of $T = 500$, and construct two types of loss functions dependent on an underlying preference model of customers (detailed in Appendix G). We then run preliminary experiments on the two problem settings dependent on loss, Online Frank Wolfe (OFW) and Online Mirror Descent (OMD) variants using the toolkit proposed: (1) OMD-UAFW: OMD with projection using vanilla away-step Frank-Wolfe (baseline), (2) OMD-ASAFW: OMD with AFW with reused active sets, (3) OMD-TSAFW: OMD with AFW with INFER, RESTRICT, and ROUNDING, (4) OMD-A^{2}FW OMD with A^{2}FW, (5) OMD-PAV: OMD with PAV, and (6) OFW. We call the first four variants as OMD-AFW variants. We averaged over 20 runs of this experiment on both the settings.

To depict the significant impact on running time of our toolkit, we normalized each OMD-UAFW run time to be 1000, and run times for all other variants in this run are corresponding scaled in Figures 4-middle and 4-right. We see roughly two orders of magnitude improvement in run time for OMD-AFW variants compared to the unoptimized OMD-AFW. Each iteration of OMD involves projecting a point on the permutahedron, and the cumulative run times for these projections are plotted. We remark that OMD-PAV and OFW are much faster than the OMD-AFW variants (with runtimes 6 to 7 orders of magnitudes respectively faster on average than OMD-UAFW); however, OMD-PAV suffers from the limitation that it only applies to cardinality-based submodular polytopes, while OFW has significantly higher regret (on average $8-25$ times the regret of OMD). The regret for all OMD variants was observed to be nearly the same. We include more details in Appendix G.

5.2 Limitations and Open questions

We proposed speeding up iterative projections using combinatorial structure inferred from past projections. We developed a toolkit to infer tight inequalities, reuse active sets, restrict linear optimization to faces of submodular polytopes and round solutions so that errors in projections do not significantly impact the overarching iterative optimization. This work focuses on theoretical results for speeding up Bregman projections over submodular polytopes. Though we speed up OMD by orders of magnitude in our preliminary experiments, this still is a long way from closing the computational gap with Online Frank Wolfe. Our work inspires many future research questions, e.g., procedures to infer tight sets on non-submodular polytopes such as matchings and procedures to round iterates to the nearest tight face for combinatorial polytopes. Nevertheless, we hope that our results can inspire future work that goes beyond looking at projection subroutines as black boxes. We believe that our work does not have any foreseeable negative ethical or societal impact.

References

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

[2] A. Beck and M. Teboulle, "Mirror descent and nonlinear projected subgradient methods for convex optimization," *Operations Research Letters*, vol. 31, no. 3, pp. 167–175, 2003.

[3] N. Srebro, K. Sridharan, and A. Tewari, "On the universality of online mirror descent," *Advances in Neural Information Processing Systems*, 2011.

[4] J. Audibert, S. Bubeck, and G. Lugosi, "Regret in online combinatorial optimization," *Mathematics of Operations Research*, vol. 39, no. 1, pp. 31–45, 2013.

[5] E. Hazan, "Survey: The convex optimization approach to regret minimization," *Optimization for machine learning*, p. 287, 2012.

[6] M. Jaggi, "Revisiting Frank-Wolfe: Projection-free sparse convex optimization," in *Proceedings of the 30th international conference on machine learning*, 2013, pp. 427–435.

[7] S. Lacoste-Julien and M. Jaggi, "On the global linear convergence of Frank-Wolfe optimization variants," in *Advances in Neural Information Processing Systems (NIPS)*, 2015, pp. 496–504.

[8] E. Hazan and E. Minasyan, "Faster projection-free online learning," in *Conference on Learning Theory*. PMLR, 2020, pp. 1877–1893.

[9] K. Nagano and K. Aihara, "Equivalence of convex minimization problems over base polytopes," *Japan journal of industrial and applied mathematics*, pp. 519–534, 2012.

[10] H. Groenevelt, "Two algorithms for maximizing a separable concave function over a polymatroid feasible region," *European journal of operational research*, vol. 54, no. 2, pp. 227–236, 1991.

[11] S. Fujishige, "Principal structures of submodular systems," *Discrete Applied Mathematics*, vol. 2, pp. 77–79, 1980.

[12] D. P. Helmbold and M. K. Warmuth, "Learning permutations with exponential weights," *The Journal of Machine Learning Research*, vol. 10, pp. 1705–1736, 2009.

[13] W. M. Koolen, M. K. Warmuth, and J. Kivinen, "Hedging structured concepts," *COLT*, 2010.

[14] S. Yasutake, K. Hatano, S. Kijima, E. Takimoto, and M. Takeda, "Online linear optimization over permutations," in *Algorithms and Computation*. Springer, 2011, pp. 534–543.

[15] S. Gupta, M. Goemans, and P. Jaillet, "Solving combinatorial games using products, projections and lexicographically optimal bases," *arXiv preprint arXiv:1603.00522*, 2016.

[16] A. György, T. Linder, G. Lugosi, and G. Ottucsák, "The on-line shortest path problem under partial monitoring." *Journal of Machine Learning Research*, vol. 8, no. 10, 2007.

[17] N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella, "Active learning on graphs via spanning trees," in *NIPS Workshop on Networks Across Disciplines*. Citeseer, 2010, pp. 1–27.

[18] H. Rahmanian, D. P. Helmbold, and S. Vishwanathan, "Online learning of combinatorial objects via extended formulation," in *Algorithmic Learning Theory*. PMLR, 2018, pp. 702–724.

[19] F. Bach *et al.*, "Learning with submodular functions: A convex optimization perspective," *Foundations and Trends® in Machine Learning*, vol. 6, no. 2-3, pp. 145–373, 2013.

[20] A. Nemirovski, "Prox-method with rate of convergence o (1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems," *SIAM Journal on Optimization*, vol. 15, no. 1, pp. 229–251, 2004.

[21] A. Beck, *First-order methods in optimization*. SIAM, 2017.

[22] S. Bubeck, "Theory of Convex Optimization for Machine Learning," *arXiv preprint arXiv:1405.4980*, 2014.

[23] E. Hazan and S. Kale, "Projection-free online learning," *arXiv preprint arXiv:1206.4657*, 2012.

[24] H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the polyak-lojasiewicz condition," in *European Conference on Machine Learning and Knowledge Discovery in Databases - Volume 9851*, ser. ECML PKDD 2016. Springer-Verlag, 2016, p. 795–811.

[25] A. Radhakrishnan, M. Belkin, and C. Uhler, "Linear convergence and implicit regularization of generalized mirror descent with time-dependent mirrors," *arXiv preprint arXiv:2009.08574*, 2020.

[26] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2003, vol. 87.

[27] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[28] M. K. Warmuth and D. Kuzmin, "Randomized PCA algorithms with regret bounds that are logarithmic in the dimension," in *Advances in Neural Information Processing Systems*, 2006, pp. 1481–1488.

[29] K. Nagano, Y. Kawahara, and K. Aihara, "Size-constrained submodular minimization through minimum norm base," in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011, pp. 977–984.

[30] J. Djolonga and A. Krause, "From MAP to marginals: Variational inference in bayesian submodular models," in *Advances in Neural Information Processing Systems*, 2014, pp. 244–252.

[31] W. Krichene, S. Krichene, and A. Bayen, "Convergence of mirror descent dynamics in the routing game," in *European Control Conference (ECC)*. IEEE, 2015, pp. 569–574.

[32] J. Edmonds, "Matroids and the greedy algorithm," *Mathematical Programming*, vol. 1, no. 1, pp. 127–136, 1971.

[33] T. Rothvoß, "Some 0/1 polytopes need exponential size extended formulations," *Mathematical Programming*, vol. 142, no. 1-2, pp. 255–268, 2013.

[34] S. Jegelka, H. Lin, and J. A. Bilmes, "On fast approximate submodular minimization," in *Advances in Neural Information Processing Systems*, 2011, pp. 460–468.

[35] C. H. Lim and S. J. Wright, "Efficient bregman projections onto the permutahedron and related polytopes," in *Artificial Intelligence and Statistics*. PMLR, 2016, pp. 1205–1213.

[36] S. Fujishige, "Lexicographically optimal base of a polymatroid with respect to a weight vector," *Mathematics of Operations Research*, 1980.

[37] S. Fujishige and S. Isotani, "A submodular function minimization algorithm based on the minimum-norm base," *Pacific Journal of Optimization*, vol. 7, no. 1, pp. 3–17, 2011.

[38] D. Chakrabarty, P. Jain, and P. Kothari, "Provable submodular minimization using wolfe's algorithm," in *Advances in Neural Information Processing Systems*, 2014, pp. 802–809.

[39] D. S. Hochbaum, "Lower and upper bounds for the allocation problem and other nonlinear optimization problems," *Mathematics of Operations Research*, 1994.

[40] S. Gupta *et al.*, "Combinatorial structures in online and convex optimization," Ph.D. dissertation, Massachusetts Institute of Technology, 2017.

[41] E. Hazan and T. Koren, "The computational power of optimization in online learning," *arXiv preprint arXiv:1504.02089*, 2015.

[42] D. Garber and E. Hazan, "A linearly convergent variant of the conditional gradient algorithm under strong convexity, with applications to online and stochastic optimization," *SIAM Journal on Optimization*, vol. 26, no. 3, p. 1493–1528, 2016.

[43] G. Lan, "The complexity of large-scale convex programming under a linear optimization oracle," *arXiv preprint arXiv:1512.06142*, 2013.

[44] M. A. Bashiri and X. Zhang, "Decomposition-invariant conditional gradient for general polytopes with line search," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, p. 2687–2697.

[45] R. Freund, P. Grigas, and R. Mazumder, "An extended Frank–Wolfe method with "in-face" directions, and its application to low-rank matrix completion," *SIAM Journal on Optimization*, vol. 27, no. 1, p. 319–346, 2015.

[46] J. GuéLat and P. Marcotte, "Some comments on wolfe's 'away step'," *Mathematical Programming*, vol. 35, pp. 110–119, 1986.

[47] A. Carderera and S. Pokutta, "Second-order conditional gradient sliding," *arXiv preprint arXiv:2002.08907*, 2020.

[48] A. Schrijver, *Combinatorial optimization: polyhedra and efficiency*. Springer Science & Business Media, 2003, vol. 24.

[49] J. Edmonds, "Submodular functions, matroids, and certain polyhedra," *Combinatorial Structures and Their Applications*, pp. 69–87, 1970.

[50] A. K. Menon, X. J. Jiang, S. Vembu, C. Elkan, and L. Ohno-Machado, "Predicting accurate probabilities with a ranking loss," in *Proceedings of the... International Conference on Machine Learning. International Conference on Machine Learning*, vol. 2012. NIH Public Access, 2012, p. 703.

[51] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 625–632.

[52] M. J. Best, N. Chakravarti, and V. A. Ubhaya, "Minimizing separable convex functions subject to simple chain constraints," *SIAM Journal on Optimization*, vol. 10, no. 3, pp. 658–672, 2000.

[53] A. Schrijver, "A combinatorial algorithm minimizing submodular functions in strongly polynomial time," *Journal of Combinatorial Theory, Series B*, vol. 80, no. 2, pp. 346–355, 2000.

[54] M. Schmidt, N. L. Roux, and F. Bach, "Convergence rates of inexact proximal-gradient methods for convex optimization," *arXiv preprint arXiv:1109.2415*, 2011.

[55] A. d'Aspremont, "Smooth optimization with approximate gradient," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1171–1183, 2008.

[56] D. Garber, "Revisiting frank-wolfe for polytopes: Strict complementarity and sparsity," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[57] J. Diakonikolas, A. Carderera, and S. Pokutta, "Locally accelerated conditional gradients," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 1737–1747.

[58] R. T. Rockafellar, *Convex analysis*. Princeton University Press, 1970.

[59] D. Suehiro, K. Hatano, S. Kijima, E. Takimoto, and K. Nagano, "Online prediction under submodular constraints," in *International Conference on Algorithmic Learning Theory (ALT)*. Springer, 2012, pp. 260–274.

[60] D. Bertsekas, A. Nedic, and O. AE, *Convex Analysis and Optimization*. Athena Scientific, 2003.

[61] G. Optimization, "Gurobi optimizer reference manual version 7.5," 2017, uRL: https://www.gurobi.com/documentation/7.5/refman.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] *All our main contributions are stated in detail in the introduction and summarized in the abstract.*
   (b) Did you describe the limitations of your work? [Yes] *Yes, we have described limitations of our work in Section 5.2.*
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] *We present the broader impact of our work in Section 5.2. We believe that our work does not have any foreseeable negative ethical or societal impact.*
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] *We ensured that our paper conforms to the ethics review guidelines.*
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] *All our assumptions are stated in the preliminaries section (Section 2). Moreover for each theorem/lemma we again state the necessary assumptions needed for these results to hold.*

(b) Did you include complete proofs of all theoretical results? [Yes] *Detailed proofs for all results presented in this paper can be found in the appendix.*

3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] *We uploaded the code along with the supplementary material in a zip file. We also included a* `ReadMe` *note explaining how to reproduce our results in the paper.*

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] *We explained the computational setup in the main body of the paper in Section 5.1. We also include more details on the computations in Appendix G. All the specifications are highlighted in our code files as well as the* `ReadMe` *note we created.*

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] *All our plots contain confidence intervals representing the variability in the results obtained across multiple plots. The number of times each experiment was run is documented in the paper and the seeds used are highlighted in the code.*

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] *We include all our computing details, i.e. total amount of compute and the type of resources used, in Appendix G*

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [N/A] *Our work does not use existing assets. We instead use synthetic data that we created*

(b) Did you mention the license of the assets? [N/A] Our work does not use existing assets.

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A] *Our setup is online learning setup that is not data based. However, our setup is in the code provided and can be used for other experiments*

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] *Our work does not use existing data.*

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] *Our work does not use existing data*

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] *Our work does not crowdsource any data.*

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] *Not applicable as our work does not crowdsource any data.*

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] *Not applicable as our work does not crowdsource any data.*