2013 IEEE 54th Annual Symposium on Foundations of Computer Science

Playing Non-linear Games with Linear Oracles

Dan Garber
Technion - Israel Institute of Technology
Haifa 32000 Israel
dangar@tx.technion.ac.il

Elad Hazan
Technion - Israel Institute of Technology
Haifa 32000 Israel
ehazan@ie.technion.ac.il

***Abstract*—Linear optimization is many times algorithmically simpler than non-linear convex optimization. Linear optimization over matroid polytopes, matching polytopes and path polytopes are example of problems for which we have efficient combinatorial algorithms, but whose non-linear convex counterpart is harder and admit significantly less efficient algorithms. This motivates the computational model of online decision making and optimization using a linear optimization oracle.**

In this computational model we give the first *efficient* decision making algorithm with optimal regret guarantees, answering an open question of [1], [2], in case the decision set is a polytope. We also give an extension of the algorithm for the partial information setting, i.e. the "bandit" model.

Our method is based on a novel variant of the conditional gradient method, or Frank-Wolfe algorithm, that reduces the task of minimizing a smooth convex function over a domain to that of minimizing a linear objective. Whereas previous variants of this method give rise to approximation algorithms, we give such algorithm that converges exponentially faster and thus runs in polynomial-time for a large class of convex optimization problems over polyhedral sets, a result of independent interest.

***Keywords*-Online Algorithms, Regret Minimization, Convex Optimization**

I. Introduction

We consider the problem of playing an online repeated game in which a decision maker is iteratively required to choose a point in a fixed convex decision set. After choosing his point, an adversary chooses some convex function and the decision maker incurs a loss that is equal to the function evaluated at the point chosen. In this adversarial setting there is no hope to play as well as an optimal offline algorithm that has the benefit of hindsight. Instead the standard benchmark is an optimal naive offline algorithm that has the benefit of hindsight but that must play the same fixed point on each round. The difference between the cumulative loss of the decision maker and that of of this offline benchmark is known as *regret*. The setting described above is often termed *online convex optimization* in the machine learning community and it has two popular variants. In the first variant on each round, after the adversary chooses the function to play, the decision maker gains full knowledge of this function. This variant is known as the *full information* setting. In the second variant the decision maker only learns the value of the function evaluated at the point he chose (which is just the loss he incurs). This variant is known as the *partial information* or *bandit* setting.

While there are known algorithms for playing such games with guaranteed regret bounds, they usually require to optimize a non-linear convex function over the decision set on every iteration (many times referred to as the "projection") [3], [4], [5], [6]. In high dimensional machine learning applications this latter non-linear optimization problem is the de-facto computational bottleneck [2]. On the other hand, for many convex sets of interest, optimizing a linear objective over the domain could be done by a very efficient and simple combinatorial algorithm. Prominent examples for this phenomenon are the matroid polytope for which there is a simple greedy algorithm for linear optimization, and the flow polytope (convex hull of all $s-t$ paths in a directed acyclic graph) for which linear optimization amounts to finding a minimum-weight path [7]. Other important examples include the set of rotations for which linear optimization is very efficient using Wahba's algorithm [8], and the bounded positive semidefinite cone, for which linear optimization amounts to a leading eigenvector computation whereas projections require SVD decompositions.

This phenomena motivates the study of online algorithms that require only linear optimization steps over the domain and in particular algorithms that require only a constant number of such steps per iteration of the game.

The main contribution of this work is an algorithm for online games for the special case in which the decision set is a polytope. Our algorithm requires only a single linear optimization step over the domain per iteration and obtains an optimal regret bound in terms of the game length in the full information setting. Using existing techniques we give an extension of this algorithm to the partial information setting which obtains the best known regret bound for this setting [1].

Our online algorithms are based on a new extension

[1]Since we use only one linear optimization step per iteration, our regret bounds suffer from a certain blow-up in the form of a small polynomial in the dimension and certain quantities of the polytope, however these additional factors are independent of the game length and are all polynomial in the input representation for standard combinatorial optimization problems, i.e. matroid polytopes, matching polytopes.

0272-5428/13 $26.00 © 2013 IEEE
DOI 10.1109/FOCS.2013.52

of a method that dates back to the 1950s for the offline optimization of a smooth convex function, known as the Conditional Gradient method or the Frank-Wolfe algorithm [9], [10], [11]. This method basically reduces the task of optimizing a non-linear function over a domain to that of optimizing a linear objective over the domain. The main appeal of this method is due to its computational simplicity and its tendency to yield sparse solutions [11], [12], [13], however its convergence rate is relatively slow and thus it is only an approximation method. One of the key contributions of our work is in providing a new conditional gradient algorithm for non-linear smooth and strongly-convex optimization over polytopes with an improved convergence rate that is exponentially faster than the basic method. The new algorithm runs in polynomial time for a large class of optimization problems and thus is an exact optimization method for such problems. This new offline algorithm is of independent interest. Building on the new ideas for offline optimization we derive our online algorithms by analysing the regret of a meta-algorithm for online games known as *regularized follow the leader* [5], [4], but replace the non-linear optimization step on each round with a single linear optimization step.

The following table gives a short summary of our results with a comparison to previous linear oracle-based methods.

Setting	Previous	This paper
Offline, smooth and strongly convex	t^{-1} [11], [13]	$e^{-O(t)}$
Online, arbitrary convex losses	$T^{3/4}$ [2]	$\sqrt{T}$
Online, strongly convex losses	$T^{3/4}$ [2]	$\log T$

Table I
COMPARISON OF LINEAR-ORACLE BASED METHODS FOR OPTIMIZATION OVER POLYTOPES IN VARIOUS SETTINGS. IN THE OFFLINE SETTING WE GIVE THE APPROXIMATION ERROR AFTER t LINEAR OPTIMIZATION STEPS, OMITTING CONSTANTS. IN THE ONLINE SETTING WE GIVE THE ORDER OF THE REGRET AFTER T ROUNDS.

A. Related Work

The two closest works to ours, which also consider the setting of online games with linear oracles, are those of Kalai and Vempala [1], and Hazan and Kale [2]. In both only linear optimization steps over the domain are used, one per iteration. Kalai and Vempala [1] give a randomized algorithm for online games in the special case in which all loss functions are linear, this is some times referred to as *online linear optimization*. Their algorithm achieves a regret bound $O(\sqrt{T})$ where T is the length of the game, which is optimal [14]. Their algorithm plays on each iteration a point in the domain that minimizes the cumulative loss on all previous iterations plus some random noise. They leave open the question of handling non-linear loss functions. Hazan and Kale [2] give an algorithm that can handle arbitrary convex losses. Their technique builds on approximating the decisions of a meta-algorithm for online games known as *regularized follow the leader*, which plays a point that is the optimal solution to a regularized non-linear convex problem. They aim at approximating this non-linear objective using a single linear optimization step. However their approximation error is not guaranteed to be sufficiently small which results in a suboptimal regret bound of $O(T^{3/4})$ and they leave open the question of attaining the optimal regret bound in terms of T.

Also relevant to our setting is the work of Kakade, Kalai and Ligett [15], which as [1] also deals with the special case of *online linear optimization*. However, [15] consider the case in which we are not given an exact linear optimization oracle but only an oracle that returns a solution with some approximation guarantee.

The extension of our online algorithm to the bandit setting is based on the techniques developed by Flaxman, Kalai and McMahan [16] which gives an elegant way of converting an algorithm for the *full information* setting into an algorithm for the *partial information* setting by coupling exploration with exploitation on each round.

The primary technical tool we use for the derivation of our results is the Conditional Gradient method, or Frank-Wolfe algorithm for offline smooth, convex optimization which was first proposed by Frank and Wolfe in 1956 [9]. The basic method guarantees additive approximation error ϵ after $O(\epsilon^{-1})$ linear optimization steps. In case the domain is a polytope and the objective is a strongly convex function, GuéLat and Marcotte [10] showed that under a certain assumption on the distance of the constrained optimum from the boundary of the set, an error rate $O(\log \epsilon^{-1})$ is achievable. However their error rate depends directly on the distance of the optimum from the boundary, and thus the $\log \epsilon^{-1}$ rate holds only for instances in which the constraints are irrelevant, i.e. problems that are effectively unconstrained optimization for which much simpler techniques exist.

Migdalas [17] showed that for the strongly convex case, an error rate $O(\log \epsilon^{-1})$ is achievable if the linear optimization step on each iteration is replaced with a quadratic optimization step, which is computationally equivalent to computing projections. More recently the conditional gradient method regained interest due to its inherent sparsity with works such as that of Clarkson [11], Hazan [12] and Jaggi [13] to the simplex, semi-definite cone and arbitrary compact convex sets respectively.

B. Techniques

One of our main contributions is a new conditional gradient algorithm for smooth strongly convex optimization over polyhedra sets which guarantees ϵ additive approximation error after $O(\log \epsilon^{-1})$ linear optimization steps over the domain. Previous approaches require in the same setting $O(\epsilon^{-1})$ linear optimization steps. The slow convergence rate of previous approaches is due to the following reason: each

step of the general method combines, via a convex sum, the current iterate with a point in the domain that optimizes the inner product with the gradient vector of the function evaluated in the current iterate, which for polyhedral sets is without loss of generality a vertex. Unfortunately, the new added vertex may be far from the current iterate which forces taking decreasing step sizes in order to guarantee convergence.

In case the objective is a strongly-convex function, a possible remedy to the above problem is to consider as solutions to the inner product problem only points that are close to the current iterate. However this results in a non-linear optimization problem which is much less efficient to solve using a linear oracle than the original linear problem. The key technical contribution of this work is in constructing a procedure, which we call *a local linear oracle*, for optimizing a linear objective over the domain in the close proximity of a feasible point. This procedure requires only a single call to the linear oracle of the polytope and returns a feasible point with objective value at least as good as the optimal point in the intersection of the polytope with a small ball centred at the current iterate. The point returned is also close enough to the current iterate which allows the reduction of the approximation error by a constant factor on each iteration.

We use the new ideas for offline optimization to analyse the regret of a meta-algorithm for online games known as *regularized follow the leader* (RFTL) [5], [4], in case that the non-linear optimization step of RFTL is replaced with a single linear optimization step over the domain. This approach is also the one taken in [2], however at the time of their work no conditional gradient algorithm with exponentially-decreasing error was known which is essential for getting the optimal regret rates with such analysis. We also manage to obtain regret bounds that are logarithmic in the length of the game for strongly convex instances, which is also known to be tight.

A technical issue with the efficiency of our online algorithm is the size of the representation of the current iterate as a convex sum of vertices. We show that it is possible to compute a sufficiently small decomposition by bootstrapping our new offline algorithm to the problem of computing a nearest point in the domain. We require to compute such a decomposition only after sufficiently many iterations, and thus the amortized oracle complexity per iteration remains a constant.

II. Formal Definitions

A. Online repeated convex games

We consider an online repeated game in which on each iteration a decision maker needs to choose a point x_t in a fixed convex set $\mathcal{P}$ (a polytope in this work). An adversary then chooses a convex loss function $f_t(x) : \mathcal{P} \to \mathbb{R}$ and the decision maker incurs loss $f_t(x_t)$. The choice of the function $f_t(x)$ may be adversarial and based on the actual plays made by the decision maker in the past. In the *full information*, setting after suffering the loss, the decision maker gets full knowledge of the function f_t. In the *partial information* setting (*bandit*) the decision maker only learns the value $f_t(x_t)$ and does not gain any other knowledge about f_t.

The standard goal in such games is to have overall loss which is not much larger than that of the best fixed point in $\mathcal{P}$ in hindsight. Formally the goal is to minimize a quantity known has *regret* which is given by,

$$\text{regret}_T = \sum_{t=1}^{T} f_t(x_t) - \min_{x \in \mathcal{P}} \sum_{t=1}^{T} f_t(x)$$

In certain cases, such as in the bandit setting, the decision maker must use randomness in order to make his decisions. In this case we consider only the expected regret, where the expectation is taken over the randomness in the algorithm of the decision maker.

B. Technical definitions and notations

Given two vectors x, y we write $x \geq y$ if every entry of x is greater or equal to the corresponding entry in y. We denote by $\|x\|$ the l_2 norm of the vector x and by $\|A\|$ the spectral norm of the matrix A, that is $\|A\| = \max_{x:\|x\|=1} \|Ax\|$. Given a matrix A we denote by $A(i)$ the vector that corresponds to the ith row of A. We denote $\mathbb{B}_r(x)$ the euclidean ball of radius r centred at x and in particular we denote the unit ball around the origin by $\mathbb{B}$. We also denote by $\mathbb{S}$ the unit sphere around the origin, that is the set $\{x \,|\, \|x\| = 1\}$.

Definition 1: We say that a function $f(x) : \mathbb{R}^n \to \mathbb{R}$ is Lipschitz with parameter L over the set $\mathcal{K} \subset \mathbb{R}^n$ if for all $x, y \in \mathcal{K}$ it holds that,

$$|f(x) - f(y)| \leq L\|x - y\|$$

Definition 2: We say that a function $f(x) : \mathbb{R}^n \to \mathbb{R}$ is β-smooth over the set $\mathcal{K}$ if for all $x, y \in \mathcal{K}$ it holds that,

$$f(y) \leq f(x) + \nabla f(x)^\top (y - x) + \beta\|x - y\|^2$$

Definition 3: We say that a function $f(x) : \mathbb{R}^n \to \mathbb{R}$ is σ-strongly convex over the set $\mathcal{K}$ if for all $x, y \in \mathcal{K}$ it holds that,

$$f(y) \geq f(x) + \nabla f(x)^\top (y - x) + \sigma\|x - y\|^2$$

The above definition together with first order optimality conditions imply that for a σ-strongly convex f, if $x^* = \arg\min_{x \in \mathcal{K}} f(x)$, then for all $x \in \mathcal{K}$

$$f(x) - f(x^*) \geq \sigma\|x - x^*\|^2$$

Note that a sufficient condition for a twice-differential function f to be β-smooth and σ-strongly convex over a domain $\mathcal{K}$ is that,

$$\forall x \in \mathcal{K} : \quad \beta \mathbf{I} \succeq \nabla^2 f(x) \succeq \sigma \mathbf{I}$$

Given a polytope $\mathcal{P} = \{x \in \mathbb{R}^n \mid A_1 x = b_1,\ A_2 x \leq b_2\}$ [2], A_2 is $m \times n$ [3], let $\mathcal{V}$ denote the set of vertices of $\mathcal{P}$ and let $N = |\mathcal{V}|$. We assume that $\mathcal{P}$ is bounded and we denote $D(\mathcal{P}) = \max_{x,y \in \mathcal{P}} \|x - y\|$. We denote $\xi(P) = \min_{v \in \mathcal{V}} \min\{b_2(j) - A_2(j)v \mid j \in [m], A_2(j)v < b_2(j)\}$. Let $r(A_2)$ denote the row rank of the matrix A_2. Let $\mathbb{A}(\mathcal{P})$ denote the set of all $r(A_2) \times n$ matrices whose rows are linearly independent vectors chosen from the rows of A_2 and denote $\psi(\mathcal{P}) = \max_{M \in \mathbb{A}(\mathcal{P})} \|M\|$. Finally denote $\mu(\mathcal{P}) = \frac{\psi(\mathcal{P}) D(\mathcal{P})}{\xi(\mathcal{P})}$. Henceforth we shall use the shorthand notation of D, ξ, ψ, μ when the polytope at hand is clear from the context.

We note that for many polytopes of interest, especially such that arise in combinatorial optimization, such as matorid polytopes, the unit flow polytope, the matching polytope and more, the quantities D, ψ are polynomial in the dimension and ξ is a constant.

III. Informal Statement of Results

In all of our results we assume that we are performing optimization (either offline or online) over a polytope $\mathcal{P}$ and that we have a linear optimization oracle $\mathcal{O}_{\mathcal{P}}$ for it:

$$\mathcal{O}_{\mathcal{P}}(c) \in \arg\min_{v \in \mathcal{V}} \{c^\top v\}$$

Offline optimization: We give an iterative algorithm that given a β-smooth and σ-strongly-convex function f returns after t iterations a point $x_{t+1} \in \mathcal{P}$ such that,

$$f(x_{t+1}) - \min_{y \in \mathcal{P}} f(y) \leq C \exp\left(-\frac{\sigma}{4\beta n \mu^2} t\right)$$

where $C = \max_{x,y \in \mathcal{P}} |f(x) - f(y)|$. The algorithm performs a single call to the oracle $\mathcal{O}_{\mathcal{P}}$ on each iteration.

This convergence rate is sometimes referred to as linear convergence rate in the continuous optimization community.

We note that for the case of minimizing the function $f(x) = \|x\|^2$ over the unit simplex it is known (see for example [13]) that for any $\epsilon \geq \frac{1}{n}$ in order to achieve additive approximation error ϵ, the cardinality of the solution (number of non zeros) must be $\Omega(\epsilon^{-1})$. In this specific problem all parameters C, σ, β, μ are constants, hence the dependence of the convergence rate on the dimension n seems mandatory.

Online optimization: Based on the technique used for the offline algorithm we derive online algorithms that require only a single call to the linear oracle $\mathcal{O}_{\mathcal{P}}$ per iteration of the game. Our algorithms achieve the following regret bounds. For ease of presentation here we only write the dependency on the length of the game T which is the primary concern. When we formally present the results we give the full dependency of the bound on all parameters.

1) For arbitrary convex loss functions we get regret bound $O(\sqrt{T})$ which is optimal [14].
2) In case all loss functions are H-strongly convex for some $H > 0$ we get a logarithmic regret bound $O(\log T)$ which is also optimal [18].
3) In the *partial information* setting we give a randomized algorithm with expected regret bound $O(T^{3/4})$. The expectation is taken over the randomness in the algorithm. To date this is the best known regret bound for this setting.

IV. A Linearly Convergent Conditional Gradient Algorithm

In this section we overview our two main technical tools which are the conditional gradient method and the construction of a new oracle we term *local linear oracle*, for finding a feasible point that dominates all points in the intersection of the polytope and a small ball with respect to some linear objective, such that the returned point is not too far from the center of the ball.

These two components combined yield a new algorithm for offline optimization of smooth strongly convex functions with error rate that is exponentially-decreasing (a linear convergence rate) which is also the technical key to our online algorithms.

A. The conditional gradient method and local linear oracles

The conditional gradient method is a simple algorithm for minimizing a smooth convex function f over a convex set $\mathcal{P}$, which in this work we assume to be a polytope, given only an oracle $\mathcal{O}_{\mathcal{P}}$ for minimizing a linear objective over $\mathcal{P}$. The basic algorithm is given in figure 1.

1: Let x_1 be an arbitrary point in $\mathcal{P}$.
2: **for** $t = 1...$ **do**
3: $\quad p_t \leftarrow \mathcal{O}_{\mathcal{P}}(\nabla f(x_t))$.
4: $\quad x_{t+1} \leftarrow x_t + \alpha_t(p_t - x_t)$ for $\alpha_t \in (0, 1)$.
5: **end for**

Figure 1. Conditional Gradient Algorithm

Denote $x^* = \arg\min_{x \in \mathcal{P}} f(x)$. The proof of convergence of the conditional gradient algorithm is due to the following simple observations.

$$\begin{aligned} & f(x_{t+1}) - f(x^*) \\ &= f(x_t + \alpha_t(p_t - x_t)) - f(x^*) \\ &\leq f(x_t) - f(x^*) + \alpha_t(p_t - x_t)^\top \nabla f(x_t) + \alpha_t^2 \beta \|p_t - x_t\|^2 \\ &\leq f(x_t) - f(x^*) + \alpha_t(x^* - x_t)^\top \nabla f(x_t) + \alpha_t^2 \beta \|p_t - x_t\|^2 \\ &\leq f(x_t) - f(x^*) + \alpha_t(f(x^*) - f(x_t)) + \alpha_t^2 \beta \|p_t - x_t\|^2 \\ &\leq (1 - \alpha_t)(f(x_t) - f(x^*)) + \alpha_t^2 \beta D^2 \end{aligned} \tag{1}$$

[2] we consider polytopes given in this most general representation since we are dealing with non-linear optimization. In this case transformations between polyhedra representations that are standard in linear programming do not apply in a straightforward manner since they may also change the objective.

[3] we don't impose any limit on the number of inequalities m and in fact it may be exponential.

where the first inequality follows from the β-smoothness of f, the second inequality follows from the optimality of p_t and the third inequality follows from convexity of f.

The relatively slow convergence of the conditional gradient algorithm is due to the term $\|p_t - x_t\|$ in the above analysis, that may remain as large as the diameter of $\mathcal{P}$ even when the error $f(x_t) - f(x^*)$ is small, thus forcing us to choose values of α_t that decrease like $\frac{1}{t}$ [11], [12], [13] which results in a poor convergence rate of $O(1/t)$.

Notice that if f is σ-strongly convex for some $\sigma > 0$ then knowing that for some iteration t it holds that $f(x_t) - f(x^*) \leq \epsilon$ implies that $\|x_t - x^*\|^2 \leq \frac{\epsilon}{\sigma}$. Thus when choosing the direction p_t, denoting $r = \sqrt{\epsilon/\sigma}$, it is enough to consider points that lie in the intersection set $\mathcal{P} \cap \mathbb{B}_r(x_t)$, since all we need for the above analysis to hold is that $p_t^\top \nabla f(x_t) \leq {x^*}^\top \nabla f(x_t)$. In this case the term $\|p_t - x_t\|^2$ will be of the same magnitude as $f(x_t) - f(x^*)$ (or even smaller) and as observable from (1) we will get

$$f(x_{t+1}) - f(x^*) \leq (f(x_t) - f(x^*)) \left(1 - \alpha_t + \frac{\alpha_t^2 \beta}{\sigma}\right) \quad (2)$$

which by an appropriate choice of α_t leads to an exponentially fast convergence.

However solving the problem $\min_{p \in \mathcal{P} \cap \mathbb{B}_r(x_t)} p^\top \nabla f(x_t)$ is much more difficult than solving the original linear problem $\min_{p \in \mathcal{P}} p^\top \nabla f(x_t)$, given a linear oracle. To overcome this difficulty we introduce the following definition which is a primary ingredient of our work.

Definition 4 (Local Linear Oracle): We say that a procedure $\mathcal{A}(x, r, c)$, $x \in \mathcal{P}$, $r \in \mathbb{R}^+$, $c \in \mathbb{R}^n$ is a Local Linear Oracle for the polytope $\mathcal{P}$ with parameter ρ, if $\mathcal{A}(x, r, c)$ returns a point $p \in \mathcal{P}$ such that:

1) $\forall y \in \mathbb{B}_r(x_t) \cap \mathcal{P}$ it holds that $c^\top y \geq c^\top p$.
2) $\|x - p\| \leq \rho \cdot r$.

The local linear oracle (LLO) relaxes the problem $\min_{p \in \mathcal{P} \cap \mathbb{B}_r(x_t)} p^\top \nabla f(x_t)$ by searching for a solution to the linear problem on a larger set, but one that still has a diameter that is not much larger than $\sqrt{f(x_t) - f(x^*)}$. One of the main technical contributions of this work is showing that for a polytope $\mathcal{P}$ a local linear oracle can be constructed such that the parameter ρ depends only on the dimension n and the quantity $\mu(\mathcal{P})$. Moreover the construction requires only a single call to the oracle $\mathcal{O}_\mathcal{P}$.

With a local linear oracle at hand, by replacing the oracle call in the conditional gradient algorithm with an LLO call, we get that for a single iteration it holds that,

$$f(x_{t+1}) - f(x^*) \leq (f(x_t) - f(x^*)) \left(1 - \alpha_t + \frac{\alpha_t^2 \beta \rho^2}{\sigma}\right) \quad (3)$$

and thus again by an appropriate choice of α_t, exponentially-fast convergence is attainable.

We conclude this subsection with the following theorem which we prove in the full version of this paper.

Theorem 1: There exists an iterative algorithm for minimizing a β-smooth and σ-strongly convex function f over a polytope $\mathcal{P}$ that on every iteration performs a single call to the linear oracle $\mathcal{O}_\mathcal{P}$ and such that after t iterations the iterate x_{t+1} satisfies:

$$f(x_{t+1}) - f(x^*) \leq C \exp\left(-\frac{\sigma}{4\beta n \mu^2} t\right)$$

where $x^* = \arg\min_{x \in \mathcal{P}} f(x)$ and C is a bound on the initial error $f(x_1) - f(x^*)$. Moreover, x_{t+1} is naturally given in the form of a convex combination of at most $t + 1$ vertices of $\mathcal{P}$.

One interesting application of theorem 1 which is also relevant to our online algorithms is the following. Assume that we are maintaining a point in the domain by its convex decomposition into vertices of the polytope and we are interested for the sake of efficiency to keep this representation relatively small. A remedy for this problem is to re-decompose the point into fewer vertices every time the size of the decomposition becomes too large. A generic way of achieving such a decomposition, that relies only on the use of the linear oracle, is by bootstrapping theorem 1 to solve the problem $\min_{x \in \mathcal{P}} \|x - x_0\|^2$ where x_0 is the point we would like to decompose. This would enable us to find a point x_1 close enough to x_0 (closeness depends on the error we are willing to incur) that is given by a smaller decomposition.

B. Constructing a local linear oracle

In this subsection we show how to construct an algorithm for the procedure $\mathcal{A}(x, r, c)$ given only an oracle that minimizes a linear objective over the polytope $\mathcal{P}$.

As an exposition for our construction of a *local linear oracle* for general polyhedral sets, we first consider the specific case of constructing a *local linear oracle* for the n-dimensional simplex, that is the set $\mathcal{S}_n = \{x \in \mathbb{R}^n \,|\, \forall i : x_i \geq 0 \,, \sum_{i=1}^n x_i = 1\}$. Given a point $x \in \mathcal{S}_n$, a radius r and a linear objective $c \in \mathbb{R}^n$, consider the optimization problem,

$$\begin{aligned} &\min_{y \in \mathcal{S}_n} y^\top c \\ &\text{s.t. } \|x - y\|_1 \leq \sqrt{n} r \end{aligned} \quad (4)$$

Observe that a solution p to problem (4) satisfies:

1) $\forall y \in \mathcal{S}_n \cap \mathbb{B}_r(x) : p^\top c \leq y^\top c$
2) $\|x - p\| \leq \sqrt{n} r$

Thus a procedure that returns solutions to problem (4) is a local linear oracle for the simplex with parameter $\rho = \sqrt{n}$.

Assume for now that $\frac{\sqrt{n} r}{2} \leq 1$. Problem (4) is solved optimally by the following very simple algorithm:

1) $i^* \leftarrow \arg\min_{i \in [n]} c(i)$
2) $p \leftarrow x$
3) let $i_1, ..., i_n$ be a permutation over $[n]$ such that $c(i_1) \geq c(i_2) \geq ... \geq c(i_n)$

4) let $k \in [n]$ be the smallest integer such that $\sum_{j=1}^{k} x(i_j) \geq \frac{\sqrt{n}r}{2}$
5) $\delta \leftarrow \sum_{j=1}^{k} x(i_j) - \frac{\sqrt{n}r}{2}$
6) $\forall j \in [k-1] : p(i_j) \leftarrow 0$
7) $p(i_k) \leftarrow \delta$
8) $p(i^*) \leftarrow p(i^*) + \frac{\sqrt{n}r}{2}$

Observe that step 1 of the above algorithm is equivalent to a single linear optimization step over the simplex.

We now turn to generalize the above simple construction for the simplex to arbitrary polyhedral sets.

Our algorithm for a local linear oracle (LLO) for arbitrary polyhedral sets is given in figure 2. Note that the algorithm assumes that the input point x is given in the form of a convex combination of vertices of the polytope. Later on we show how to maintain such a decomposition of the input point x efficiently.

1: Input: a point $x \in \mathcal{P}$ such that $x = \sum_{i=1}^{k} \lambda_i v_i$, $\lambda_i > 0, \sum_{i=1}^{k} \lambda_i = 1, v_i \in \mathcal{V}$, radius $r > 0$, linear objective $c \in \mathbb{R}^n$.
2: $\Delta \leftarrow \min\{\frac{\sqrt{n}\psi}{\xi} r, 1\}$.
3: $\forall i \in [k]$: $l_i \leftarrow c^\top v_i$.
4: Let $i_1, ...i_k$ be a permutation over $[k]$ such that $l_{i_1} \geq l_{i_2} \geq ...l_{i_k}$.
5: **for** $j = 1...k$ **do**
6: $\quad \lambda'_{i_j} \leftarrow \max\{0, \lambda_{i_j} - \Delta\}$.
7: $\quad \Delta \leftarrow \Delta - (\lambda_{i_j} - \lambda'_{i_k})$.
8: **end for**
9: $v \leftarrow \mathcal{O}_{\mathcal{P}}(c)$.
10: **return** $p \leftarrow \sum_{i=1}^{k} \lambda'_i v_i + \left(1 - \sum_{i=1}^{k} \lambda'_i\right) v$.

Figure 2. Local Linear Oracle Algorithm

Lemma 1: Let $x \in \mathcal{P}$ be the input to the LLO algorithm and let $y \in \mathcal{P}$. Write $y = \sum_{i=1}^{k}(\lambda_i - \Delta_i)v_i + (\sum_{i=1}^{k} \Delta_i)z$ for some $\Delta_i \in [0, \lambda_i]$ and $z \in \mathcal{P}$ such that the sum $\sum_{i=1}^{k} \Delta_i$ is minimized. Then $\forall i \in [k]$ there exists an index $j \in [m]$ such that $A_2(j)v_i < b_2(j)$ and $A_2(j)z = b_2(j)$.

The proof of the lemma is given in the full version of the paper. The idea is to show that if the condition in lemma 1 is not satisfied then one can express the point y as $y = \sum_{i=1}^{k}(\lambda_i - \Delta'_i)v_i + (\sum_{i=1}^{k} \Delta'_i)z'$, $\Delta'_i \in [0, \lambda_i]$ such that $\sum_i \Delta'_i < \sum_i \Delta_i$, contradicting the minimality of $\sum_i \Delta_i$.

The proof of the following claim is given in the full version of this paper.

Claim 1: Let $z \in \mathcal{P}$ and denote $C(z) = \{i \in [m] : A_2(i)z = b_2(i)\}$ and let $C_0(z) \subseteq C(z)$ be such that the set $\{A_2(i)\}_{i \in C_0(z)}$ is a basis for span $\left(\{A_2(i)\}_{i \in C(z)}\right)$. Then given $y \in \mathcal{P}$, if there exists $i \in C(z)$ such that $A_2(i)y < b_2(i)$ then there exists $i_0 \in C_0(z)$ such that $A_2(i_0)y < b_2(i_0)$.

Lemma 2: Let $x \in \mathcal{P}$ be the input to the LLO algorithm and let $y \in \mathcal{P}$ be such that $\|x - y\| \leq r$. Write $y = \sum_{i=1}^{k}(\lambda_i - \Delta_i)v_i + (\sum_{i=1}^{k} \Delta_i)z$ for some $\Delta_i \in [0, \lambda_i]$ and $z \in \mathcal{P}$ such that the sum $\sum_{i=1}^{k} \Delta_i$ is minimized. Then $\sum_{i=1}^{k} \Delta_i \leq \frac{\sqrt{n}\psi}{\xi} r$.

Proof: Denote $C(z) = \{j \in [m] : A_2(j)z = b_2(j)\}$ and let $C_0(z) \subseteq C(z)$ such that the set of vectors $\{A_2(i)\}_{i \in C_0(z)}$ is a basis for span $\left(\{A_2(i)\}_{i \in C(z)}\right)$. Denote $A_{2,z} \in \mathbb{R}^{|C_0(z)| \times n}$ the matrix A_2 after deleting from it every row $i \notin C_0(z)$ and recall that by definition $\|A_{2,z}\| \leq \psi$. Then it holds that,

$$\begin{aligned}
\|x - y\|^2 &= \left\| \sum_{i=1}^{k} \Delta_i (v_i - z) \right\|^2 \\
&\geq \frac{1}{\|A_{2,z}\|^2} \left\| A_{2,z} \left(\sum_{i=1}^{k} \Delta_i (v_i - z) \right) \right\|^2 \\
&\geq \frac{1}{\psi^2} \left\| \sum_{i=1}^{k} \Delta_i A_{2,z} (v_i - z) \right\|^2 \\
&= \frac{1}{\psi^2} \sum_{j \in C_0(z)} \left(\sum_{i=1}^{k} \Delta_i (A_2(j)v_i - b_2(j)) \right)^2
\end{aligned}$$

Note that $|C_0(z)| \leq n$ and that for any vector $x \in \mathbb{R}^{|C_0(z)|}$ it holds that $\|x\| \geq \frac{1}{\sqrt{|C_0(z)|}} \|x\|_1$. Thus we have that,

$$\begin{aligned}
\|x - y\|^2 &\geq \frac{1}{n\psi^2} \left(\sum_{j \in C_0(z)} \left| \sum_{i=1}^{k} \Delta_i (A_2(j)v_i - b_2(j)) \right| \right)^2 \\
&= \frac{1}{n\psi^2} \left(\sum_{j \in C_0(z)} \sum_{i=1}^{k} \Delta_i (b_2(j) - A_2(j)v_i) \right)^2
\end{aligned}$$

Combining lemma 1 and claim 1, we have that for all $i \in [k]$ such that $\Delta_i > 0$ there exists $j \in C_0(z)$ such that $A_2(j)v_i \leq b_2(j) - \xi$. Hence,

$$\|x - y\|^2 \geq \frac{1}{n\psi^2} \left(\sum_{i=1}^{k} \Delta_i \xi \right)^2 = \frac{\xi^2}{n\psi^2} \left(\sum_{i=1}^{k} \Delta_i \right)^2$$

Since $\|x - y\|^2 \leq r^2$ we conclude that $\sum_{i=1}^{k} \Delta_i \leq \frac{\sqrt{n}\psi}{\xi} r$. ∎

The following lemma establishes that our LLO algorithm is indeed a local linear oracle for $\mathcal{P}$ with parameter $\rho = \sqrt{n}\mu$.

Lemma 3: Assume that the input to the LLO algorithm is $x = \sum_{i=1}^{k} \lambda_i v_i$ such that $\forall i \in [k]$, $\lambda_i > 0, v_i \in \mathcal{V}$ and $\sum_{i=1}^{k} \lambda_i = 1$. Let p be the point returned by the algorithm. Then the following conditions hold:

1) $p \in \mathcal{P}$.
2) $\|x - p\| \leq \sqrt{n}\mu r$.
3) $\forall y \in \mathbb{B}_r(x) \cap \mathcal{P}$ it holds that $c^\top y \geq c^\top p$.

Note that the LLO algorithm assumes that the input point x is given by its convex decomposition into vertices. Our

online algorithm uses the LLO algorithm in the following way: it sends as input to the LLO the current iterate x_t and then given the output of the LLO, p_t it produces the next iterate x_{t+1} by a taking a convex combination $x_{t+1} \leftarrow (1-\alpha)x_t + \alpha p_t$ for some parameter $\alpha \in (0,1)$. Thus if the convex decomposition of x_t is given, updating it to the convex decomposition of x_{t+1} is straightforward. Moreover, denoting $\mathcal{V}_t \subseteq \mathcal{V}$ the set of vertices that form the convex decomposition of x_t, it is clear from the LLO algorithm that $|\mathcal{V}_{t+1} \setminus \mathcal{V}_t| \leq 1$, since at most a single vertex, v is added to the decomposition.

Lemma 4: The LLO algorithm has an implementation such that each invocation of the algorithm requires a single call to the oracle $\mathcal{O}_{\mathcal{P}}$ and additional $O(T(n+\log T))$ time where T is the total number of calls to the LLO.

Proof: Clearly the LLO algorithm calls $\mathcal{O}_{\mathcal{P}}$ only once. The complexity of all other operations depends on k - the number of vertices in the convex decomposition of the input point x. As we discussed, if we denote by x_t, x_{t+1} the inputs to the algorithm on calls number $t, t+1$ to the algorithm and by k_t, k_{t+1} the number of vertices in the convex decomposition of x_t, x_{t+1} respectively then $k_{t+1} \leq k_t + 1$. Thus if the algorithm is called a total number of T times and the initial point, x_1 is a vertex, then at all times $k \leq T$. Since all other operations except for calling $\mathcal{O}_{\mathcal{P}}$ consist of computing k inner products between vectors in $\mathbb{R}^n$ and sorting k scalars, the lemma follows. ∎

Note that we can get rid of the linear dependence on T in the bound in lemma 4 by decomposing the iterate x_t into a convex sum of fewer vertices in case the number of vertices in the current decomposition, k becomes too large. From Caratheodory's theorem we know that there exists a decomposition with at most $n+1$ vertices and for many polytopes of interest there is an efficient algorithm for computing such a decomposition. From previous discussions, we will need to invoke this decomposition algorithm only every $O(n)$ iterations which will keep the amortized iteration complexity low.

Another approach for the above problem that relies only on the use of the oracle $\mathcal{O}_{\mathcal{P}}$ is the following. If k is too large we can compute a new decomposition of the input point x by finding an approximated solution to the optimization problem $\min_{y \in \mathcal{P}} \|x-y\|^2$ using theorem 1 up to precision r^2. The result will be a point x' given by a decomposition into $O(n\mu^2 \log(1/r))$ vertices such that $\|x'-x\| \leq r$. Thus this gives us a construction of a local linear oracle with parameter $\rho = 3\sqrt{n}\mu$. We will need to invoke this decomposition procedure only every $O(n\mu^2 \log(1/r))$ iterations which leads to the following lemma.

Lemma 5: Assume that on all iterations the input r to the LLO algorithm is lower-bounded by $r_1 > 0$. Then there exists an implementation for a local linear oracle with parameter $\rho = 3\sqrt{n}\mu$ such that the amortized linear oracle complexity per iteration is 2 and the additional amortized complexity per iteration is $O\left(n\mu^2 \log(1/r_1)\left(n + \log(n\mu^2 \log(1/r_1))\right)\right)$.

In our online algorithm the bound r_1 will always satisfy $\frac{1}{r_1} = O(T)$ where T is the length of the game, and thus the running time per iteration depends only logarithmically on T.

V. Online Algorithms

In this section we present and analyse our algorithm for playing online convex games. We initially focus on the *full information* setting. Later on we show how to convert the *full information* algorithm into an algorithm for the *partial information* setting. The algorithm is given in figure 3. The functions $F_t(x)$, used by the algorithm (in line 6), will be specified precisely in the analysis. For now assume that $F_t(x)$ sums the loss functions on times $1,...,t$ plus some regularization term of the form $\|x-x_0\|^2$.

1: Input: horizon T, set of radii $\{r_t\}_{t=1}^T$ $t \in [T]$, optimization parameter $\alpha \in (0,1)$, LLO $\mathcal{A}$ with parameter ρ.
2: Let x_1 be an arbitrary vertex in $\mathcal{V}$.
3: **for** $t = 1...T$ **do**
4: play x_t.
5: receive f_t.
6: $p_t \leftarrow \mathcal{A}(x_t, r_t, \nabla F_t(x_t))$.
7: $x_{t+1} \leftarrow x_t + \alpha(p_t - x_t)$.
8: **end for**

Figure 3. Algorithm for Full Information Setting

We have the following two main results.

Denote $G = \sup_{x \in \mathcal{P}, t \in [T]} \|\nabla f_t(x)\|$ and recall that we have a construction for a local linear oracle with parameter $\rho = O(\sqrt{n}\mu)$.

Theorem 2: Given a LLO with parameter $\rho = O(\sqrt{n}\mu)$, there exists a choice for the parameters $\eta, \alpha, \{r_t\}_{t=1}^T$ such that for general convex losses the regret of the algorithm in figure 3 is $O(GD\mu\sqrt{nT})$.

Theorem 3: Given a LLO with parameter $\rho = O(\sqrt{n}\mu)$, there exists a choice for the parameters $\eta, \alpha, \{r_t\}_{t=1}^T$ such that if all loss functions $f_t(x)$ are H-strongly convex then the regret of the algorithm in figure 3 is $O((G+HD)^2 n\mu^2/H)\log T)$.

A. Analysis for general convex losses

For time $t \in \{0,1,...,T-1\}$ we define the function $F_t(x) = \eta\left(\sum_{\tau=1}^t \nabla f_\tau(x_\tau)^\top x\right) + \|x-x_1\|^2$ where η is a parameter that will by determined in the analysis.

Denote $x_1^* = x_1$ and for all $t \in \{1,2,...,T-1\}$ denote $x_{t+1}^* = \arg\min_{x \in \mathcal{P}} F_t(x)$. Denote also $x^* = \arg\min_{x \in \mathcal{P}} \sum_{t=1}^T f_t(x)$. Observe that $F_t(x)$ is 1-smooth and 1-strongly convex and thus fits the assumptions of our results for offline smooth and strongly convex optimization (see theorem 1).

The proof of the following lemma is basically outlined in subsection IV-A and it is also given in full version of this paper.

Lemma 6: Given time $t \in [T-1]$, let $x^*_{t+1} = \arg\min_{x\in\mathcal{P}} F_t(x)$ and assume that $\|x_t - x^*_{t+1}\| \le r_t$. Then for every $\alpha \in (0,1)$ it holds that,

$$F_t(x_{t+1}) - F_t(x^*_{t+1}) \le (1-\alpha)\left(F_t(x_t) - F_t(x^*_{t+1})\right) + \beta\alpha^2 \min\{\rho^2 r_t^2, D^2\}$$

Our online algorithm basically finds approximated minimizers to the sequence of functions $F_1(x), F_2(x), ..., F_T(x)$ where on each time t, the point x_t - the approximated minimizer of $F_{t-1}(x)$ is produced by using the previous iterate x_{t-1} as a "warm start" for the problem $\min_{x\in\mathcal{P}} F_{t-1}(x)$ and then performing a single improvement step of the LLO-based conditional gradient algorithm. Lemma 6 gives a guarantee for such an improvement step.

The following lemma bounds for every time t the distance between the approximated minimizer of F_{t-1} - the point x_t that our algorithm plays on time t, and the true minimizer x^*_t which is the point that we would have liked to play on time t. A proof is given in the full version of this paper.

Lemma 7: For any $\epsilon > 0$, there is a choice for the parameters $\eta, \alpha, \{r_t\}_{t=1}^T$ such that for all $t \in [T]$: $\|x_t - x^*_t\| \le \sqrt{\epsilon}$.

Proof of theorem 2: Observe that playing the point $x^*_{t+1} = \arg\min_{x\in\mathcal{P}} F_t(x)$ on each time t is equivalent to playing the leader on each time with respect to the loss functions $f'_1(x) = \nabla f_1(x_1)^\top x + \frac{1}{\eta}\|x - x_1\|^2$ and $f'_t(x) = \nabla f_t(x)^\top x$ for every $t > 1$. This strategy of playing on each time according to the leader is known to achieve overall zero regret, see [1]. Thus,

$$\begin{aligned}\sum_{t=1}^T \nabla f_t(x_t)^\top (x^*_{t+1} - x^*) &\le \frac{1}{\eta}(\|x^* - x_1\|^2 - \|x^*_1 - x_1\|^2) \\ &\le \frac{D^2}{\eta}\end{aligned}$$

By the definition of $F_t(x)$, x^*_t and the use of strong-convexity we have that,

$$\|x^*_t - x^*_{t+1}\|^2 \le F_t(x^*_t) - F_t(x^*_{t+1}) \le \eta G\|x^*_t - x^*_{t+1}\|$$

Which implies by the triangle inequality that,

$$\sum_{t=1}^T \nabla f_t(x_t)^\top (x^*_t - x^*) \le \frac{D^2}{\eta} + T\eta G^2$$

Given a choice for ϵ, setting $\eta, \alpha, \{r_t\}_{t=1}^T$ to the values determined in the proof of lemma 7, plugging lemma 7 and by the convexity of $f_t()$ we get that,

$$\begin{aligned}\sum_{t=1}^T f_t(x_t) - f_t(x^*) &\le \sum_{t=1}^T \nabla f_t(x_t)^\top (x_t - x^*) \\ &\le \frac{18GD^2\rho^2}{\sqrt{\epsilon}} + \frac{TG\sqrt{\epsilon}}{18\rho^2} + TG\sqrt{\epsilon}\end{aligned}$$

The theorem now follows from plugging $\epsilon = \frac{(D\rho)^2}{T}$ and the fact that $\rho \ge 1$. ■

The analysis for the case of strongly convex losses follows the same lines with some relatively minor modifications such as approximating $f_t(x)$ with the function $\tilde{f}_t(x) = \nabla f_t(x_t)^\top x + H\|x - x_t\|^2$ instead of just $\nabla f_t(x_t)^\top x$ as done in the analysis above. The full details are given in the full version of this paper.

B. Bandit algorithm

Our online algorithm in figure 3 can be converted to the bandit setting in an almost straightforward manner using the techniques described in [16]. The major difference between our algorithm and that of [16] is that our bandit algorithm is based on our algorithm for the *full information* setting, whereas the algorithm in [16] is based on Zinkevich's online gradient descent algorithm [3].

We now assume that without loss of generality the polytope $\mathcal{P}$ contains the origin and it holds that $r\mathbb{B} \subseteq \mathcal{P} \subseteq R\mathbb{B}$ for some positive scalars r, R. We assume that the loss function $f_t(x)$ chosen by the adversary on time t, is chosen with knowledge of the history of the game but without any knowledge of possible randomization used by the decision maker on time t. We also assume that $f_t(x)$ is L-Lipschitz for some positive scalar L and $|f_t(x)| \le C$ for some positive scalar C.

Our bandit algorithm and the proof of the following theorem are given in the full version of this paper.

Theorem 4: Our bandit algorithm generates a sequence of points $x_1, x_2, ..., x_T \in \mathcal{P}$ such that:

$$\mathbb{E}\left[\sum_{t=1}^T f_t(x_t) - \min_{x\in\mathcal{P}} \sum_{t=1}^T f_t(x)\right] = O\left(\frac{\sqrt{n^{3/2}CL\mu}R}{\sqrt{r}} T^{3/4}\right)$$

where the expectation is taken over the randomness in the algorithm.

VI. Conclusions and Open Problems

We have introduced the first *efficient* algorithm for playing repeated convex games with optimal regret in the linear oracle model, and the first linearly-converging conditional gradient algorithm for smooth and strongly-convex optimization over polyhedral sets.

The most important open problem remaining is to give an efficient decision making algorithm in the linear oracle model for general convex sets. Of particular practical interest is the bounded positive semidefinite cone, and a specific algorithm for this setting would be of particular interest in machine learning.

Another interesting research direction is to obtain an efficient polynomial time algorithm in the linear oracle

model for general convex optimization, possibly using similar techniques as Dunagan and Vempala [19]. [4]

Finally, it will be interesting to extend our techniques to the setting of Kakade, Kalai and Ligett [15] in which we are only given an approximated linear oracle and not an exact one, as assumed here. Their algorithm requires to call the approximated oracle $O(T)$ times per iteration of the game. Perhaps by further developing our techniques, this oracle complexity could be reduced dramatically, resulting in a more practical algorithm. Moreover, their algorithm applies only to the case in which all loss functions are linear. Extending their result to the non-linear case is also very interesting.

ACKNOWLEDGMENT

Work carried out and supported by the Technion-Microsoft Electronic Commerce Research Center and ISF grant 810/11.

REFERENCES

[1] A. T. Kalai and S. Vempala, "Efficient algorithms for online decision problems," *J. Comput. Syst. Sci.*, vol. 71, no. 3, pp. 291–307, 2005.

[2] E. Hazan and S. Kale, "Projection-free online learning," in *ICML*, 2012.

[3] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *ICML*, 2003, pp. 928–936.

[4] S. Shalev-Shwartz, "Online learning and online convex optimization," *Foundations and Trends in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2012.

[5] E. Hazan, "A survey: The convex optimization approach to regret minimization," in *Optimization for Machine Learning*, S. Sra, S. Nowozin, and S. J. Wright, Eds. MIT Press, 2011, pp. 287–302.

[6] W. M. Koolen, M. K. Warmuth, and J. Kivinen, "Hedging structured concepts," in *COLT*, 2010, pp. 93–105.

[7] A. Schrijver, *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003.

[8] G. Wahba, "A least squares estimate of satellite attitude," *SIAM Rev.*, vol. 7, no. 3, 1965.

[9] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics Quarterly*, vol. 3, pp. 149–154, 1956.

[10] J. GuéLat and P. Marcotte, "Some comments on Wolfe's 'away step'," *Mathematical Programming*, vol. 35, no. 1, 1986.

[11] K. L. Clarkson, "Coresets, sparse greedy approximation, and the frank-wolfe algorithm," in *SODA*, 2008, pp. 922–931.

[4] Polynomial time convex optimization is known to be possible even in the more difficult membership oracle model, here we ask for more efficient running times.

[12] E. Hazan, "Sparse approximate solutions to semidefinite programs," in *LATIN*, 2008, pp. 306–316.

[13] M. Jaggi, "Sparse convex optimization methods for machine learning," Ph.D. dissertation, ETH Zurich, Oct. 2011.

[14] N. Cesa-Bianchi and G. Lugosi, *Prediction, Learning, and Games*. New York, NY, USA: Cambridge University Press, 2006.

[15] S. M. Kakade, A. T. Kalai, and K. Ligett, "Playing games with approximation algorithms," in *STOC*, 2007, pp. 546–555.

[16] A. Flaxman, A. T. Kalai, and H. B. McMahan, "Online convex optimization in the bandit setting: gradient descent without a gradient," in *SODA*, 2005, pp. 385–394.

[17] A. Migdalas, "A regularization of the frankwolfe method and unification of certain nonlinear programming methods," *Mathematical Programming*, vol. 65, pp. 331–345, 1994.

[18] E. Hazan and S. Kale, "Beyond the regret minimization barrier: an optimal algorithm for stochastic strongly-convex optimization," *Journal of Machine Learning Research - Proceedings Track*, vol. 19, pp. 421–436, 2011.

[19] J. Dunagan and S. Vempala, "A simple polynomial-time rescaling algorithm for solving linear programs," *Math. Program.*, vol. 114, no. 1, pp. 101–114, 2008.