Project Report COMS W4232 Advanced Algorithms
Reading-based Project: Online Convex Optimization

Nilaksh Agarwal
Department of Computer Science
Columbia University
New York, USA
na2886@columbia.edu

Katie Jooyoung Kim
Department of Computer Science
Columbia University
New York, USA
jk4534@columbia.edu

1 Problem Statement

Online problems deal with incomplete information, especially over a period of time. In particular, online convex optimization (OCO) problems offer the following extension to traditional convex optimization problems (Yuan and Lamperski, 2018): at time t we choose a vector x_t in a convex set $S = \{x : g(x) \leq 0\}$. Then, we receive a convex loss function $f_t : S \to R$, which allows the definition of cumulative regret $Regret_T(x^*) = \sum_{t=1}^{T} f_t(x_t) - \sum_{t=1}^{T} f_t(x^*)$ where x^* is defined as the optimal solution to $min_{x \in S} \sum_{t=1}^{T} f_t(x)$. This can be interpreted as the difference between actual (realized) cumulative loss until time T and the cumulative loss under the "best possible choice". The goal is to obtain an algorithm which successively generates x_t which minimizes the cumulative regret.

We note a few aspects of the above problem: for one, it is analogous to linear programming as discussed in class. Instead of maximizing $c^T x$ subject to $Ax \leq b$ and $x \geq 0$, we minimize regret subject to S – with the additional challenge of varying x_t over time. For another, while in standard linear programming we focus solely on the feasible set of solutions that satisfy all constraints, in the OCO setting we also consider algorithms that generate x_t that may at some time steps t violate the constraint $g(x) \leq 0$. This leads to a natural extension of the problem: we minimize both the regret and the cumulative constraint violations. Finally, we note that we would ideally like to have cumulative regret to be sub-linear in T. Intuitively, we understand this as follows. As we advance in time, we "do better" in the sense that we have smaller relative difference with the minimum possible loss up to that point.

In addition, we note the different classes of feedback availability in OCO problems: in particular, a popular setting is bandit convex optimization in which we are not given the full loss function f_t, but rather only its value evaluated at x_t, i.e. $f_t(x_t)$.

Our goal is to examine and expose different recent approaches in handling online convex optimization problems in a variety of feedback availability settings, and to compare and contrast the main results and ideas.

2 Motivation

Online optimization problems have many applications, such as auctions, portfolio management, and online spam filtering. In addition, it is useful to consider situations in which we take into account violations of the constraint set S due to the following: in an offline setting where we have all the information, it is possible to determine what "feasible" is in advance and to consider only solutions that respect the corresponding constraints. One might think of the familiar LP setting – *given that product A requires m and n resources and is sold for x & product B requires p and q resources and is sold for y, what combination of A and B should we produce?*. However, in an online setting, it may not be possible to always guarantee that constraints will be satisfied – we can think of power supply constraints over time. There

are "surge" moments in which the constraints are violated, and they cannot be avoided, but they exist. Online problems provide a formalisation for such situations.

3 List of Papers

- Online convex optimization with stochastic constraints (Yu et al., 2017)
- Online convex optimization for cumulative constraints (Yuan and Lamperski, 2018)
- Boosting for online convex optimization (Hazan and Singh, 2021)
- Optimal Algorithms for Online Convex Optimization with Multi-Point Bandit Feedback (Agarwal and Dekel, 2010)

4 Initial Observations

Most of the papers we are studying perform some variation/relaxation in the initial convex optimization problem to achieve improvements in this space.

For example, (Agarwal and Dekel, 2010) changes the bandit feedback mechanism to allow for two special cases, (i) a 2 point query from the convex set; and (ii) a $d + 1$ point query from the convex set of dimension d. In these scenarios they are able to prove regret bounds close to the full information case.

In (Hazan and Singh, 2021) they develop a boosting based approach both for the partial-information (bandits) and full-information online convex optimization settings. They first start off with a weak-learner that guarantees multiplicative approximate regret against the experts. On this, they employ a boosting strategy that guarantees near optimal regret against the convex hull of the base class.

The authors of (Yu et al., 2017) introduce multiple stochastic functional constraints in the online convex optimization problem. This is done to model stochastic environments or deterministic environments with noisy observations. These constraints lead to special cases such as online convex optimization with long term constraints, stochastic constrained convex optimization, and deterministic constrained convex optimization. There are able to obtain a $O(\sqrt{T})$ expected regret and constraint violations.

(Yuan and Lamperski, 2018) changes the regret calculation, opting to use squared constraint violations ($\sum \left([g(x_t)]_+\right)^2$) over their regular summation ($\sum g(x_t)$). In the regular summation form, strictly feasible solutions can cancel out the effects of violated constraints. Furthermore, it heavily penalizes large constraint violations. They are able to improve the regret bound for strongly convex objectives.

5 Online convex optimization with stochastic constraints

5.1 Motivation

(Yu et al., 2017) proposes a new algorithm suitable for a generalization of (Zinkevich, 2003a). In the latter, we consider online convex optimization over a known simple fixed set. In other words, the constraints are known a priori, and only the convex loss functions $f^t(\cdot)$ vary arbitrarily over time. In this setting, we do not allow constraint violations; the sole aim is to develop a dynamic learning algorithm such that the regret grows sub-linearly with respect to the number of rounds T. In particular, Zinkevich's algorithm is as follows: $x(t+1) = \mathcal{P}_\chi[x(t) - \gamma \nabla f^t(x(t))]$ where $\mathcal{P}_\chi$ indicates the projection onto the feasible set (known a priori) χ and $\nabla f^t(\cdot)$ is the sub-gradient of $f^t(\cdot)$. It is evident that this will require the solution $x(t)$ at each time-step to be feasible, and that the projection operation in each time-step will effectively require a separate constrained optimization problem.

This framework can be impractical in real-life situations: for one, we may not have full knowledge of the feasible set χ in advance; χ may also vary over time. For another, the constrained optimization probelm posed by the projection operation may be a difficult problem in its own right – then the overall algorithm will end up being inefficient. However, if the constraint functions $g^t(x)$ vary entirely arbitrarily, then achieving sub-linear growth of regret and sub-linear growth of the cumulative constraint violation $\sum_{t=1}^{T} g^t(x(t))$ is impossible (Mannor et al., 2009).

Building upon these observations, the authors analyze OCO with constraint functions that are not entirely arbitrary. Instead, they consider *stochastic constraints*: the constraint functions $g_k^t(x)$ are generated i.i.d. from an unknown probability model (the subscript k indicates that there may be several constraint functions). Hence, at each round t, the decision-maker receives a loss function $f^t(\cdot)$ and i.i.d. constraint function realisations. This framework leads to a number of differences between the analysis to follow and the previous analysis by Zinkevich. For one, as we do not know the constraints in advance, we may not be able to satisfy them at every time step. Hence, the aim will be to satisfy the constraints in expectation. In addition, as we will now have constraint violations as a given, we will analyze the growth of cumulative constraint violation as well as the growth of regret with respect to T. Nevertheless, the goal remains the same: to bound the growth of both such that it is sub-linear with respect to T. In particular, (Yu et al., 2017) examines two different kinds of bounds: the bounds of regret and cumulative constraint violation *in expectation*, and the bounds *with high probability*.

5.2 Results

Within the described framework, the new algorithm proposed by (Yu et al., 2017) achieves the following.

- In expectation:
  - $O(\sqrt{T})$ regret
  - $O(\sqrt{T})$ cumulative constraint violation
- With probability $\geq 1 - \lambda$ for any $0 < \lambda < 1$:
  - $O(\sqrt{T}log(T)log^{1.5}(\frac{1}{\lambda}))$ regret
  - $O(\sqrt{T}log(T)log(\frac{1}{\lambda}))$ cumulative constraint violation

Hence, we can take for instance $\lambda = 0.1$ to obtain $O(\sqrt{T}log(T))$ regret and $O(\sqrt{T}log(T))$ constraint violations with probability at least 0.9.

5.3 Methodology

(Yu et al., 2017) introduces a new algorithm based on *virtual queues*. For each stochastic constraint function $g_k^t(\mathbf{x}) = g_k(\mathbf{x}; \omega(t))$, the authors introduce $Q_k(t)$, with $Q_k(1) = 0$ for all $k \in \{1, 2, ..., m\}$ and update rule $Q_k(t+1) = \max\{Q_k(t) + g_k^t(\mathbf{x}(t)) + [\nabla g_k^t(\mathbf{x}(t))]^T[\mathbf{x}(t+1) - \mathbf{x}(t)], 0\}$. Then, the authors define $\mathbf{Q}(t) = [Q_1(t), ..., Q_m(t)]^T$, $L(t) = \frac{1}{2}\|\mathbf{Q}(t)\|^2$, and $\Delta(t) = L(t+1) - L(t)$. Intuitively, $\Delta(t)$ gives a scalar measure of the change in $\mathbf{Q}(t)$, or the accumulation in $\mathbf{Q}(t)$. This in turn represents the accumulation of constraint violations $g_k^t(\mathbf{x})$. In addition, at each time step, the algorithm updates $\mathbf{x}(t+1)$ by choosing the minimizer in $\mathcal{X}_0$ of $V[\nabla f^t(\mathbf{x}(t))]^T[\mathbf{x} - \mathbf{x}(t)] + \sum_{k=1}^{m} Q_k(t)[\nabla g_k^t(\mathbf{x}(t))]^T[\mathbf{x} - \mathbf{x}(t)] + \alpha\|\mathbf{x} - \mathbf{x}(t)\|^2$ for constants V and α. We see in the sum of the second expression that the algorithm in effect chooses to minimize $\Delta(t)$.

Finally, the authors show that $\forall T \geq 1$, the algorithm guarantees that $\forall k \in \{1, ..., m\}$, $\sum_{t=1}^{T} g_k^t(\mathbf{x}(t))$ is bounded by $\|\mathbf{Q}(T+1)\| + O(\sum_{t=1}^{T} \|\mathbf{Q}(t)\|)$. Hence, minimizing $\Delta(t)$ leads to smaller $\|\mathbf{Q}(t)\|$, which bounds the cumulative constraint violation.

5.4 Applications

(Yu et al., 2017) discusses a specific application of the new learning algorithm to a real-life situation: online job scheduling in distributed data centers. In particular, the authors consider a system consisting

of one front-end job router and 100 servers distributed geographically in 10 different zones; we must note that the price of electricity can vary significantly across zones and across time. This leads to the natural formulation of a loss function to be minimized (i.e. the total cost of electricity) and stochastic constraints that should be satisfied in expectation (i.e. the jobs that must be served by the system). Formally, the stochastic constraints to be satisfied are given for each time step t by $g^t(x(t)) = \omega(t) - \sum_{i=1}^{100} h_i(x_i(t))$, where $\omega(t)$ corresponds to the incoming job (i.e. demand for electricity) at time t and each of the $h_i(x_i(t))$ correspond to the job served by the server indexed by i (i.e. provided electricity).

Hence, we obtain a real-life setting which corresponds to the problem of online convex optimization with stochastic constraints: we have a loss function (cost of electricity) varying over time, as well as constraint functions that are generated i.i.d. from an unknown probability model. Finally, we can accept that the requested jobs may not be served at all times; our constraint is not always satisfied. It is sufficient for the constraint to be satisfied in expectation, or at least with high probability: exactly what the proposed algorithm is useful for. Upon experimentation, the authors discover that the proposed algorithm performs similarly to the best fixed strategy in hindsight in terms of both the loss (running average electricity cost) and the constraint violations (running average unserved jobs).

6 Online convex optimization for cumulative constraints

6.1 Motivation

As discussed previously, we may examine the online convex optimization problem in many different kinds of constraint settings. In particular, instead of stochastic constraints, we may consider cumulative violations of a fixed constraint function $g(x)$ over time: we take the sum $\sum_{t=1}^{T} g(x_t)$. The constraint violation analysis differs in this setting as we no longer observe randomness. In addition, analysis of a sum – as opposed to simply analyzing the constraint violations at each time step – allows the following notion: it is possible for feasible solutions that are "far from the borders" (i.e. strictly feasible solutions) to cancel out the effect of past or future constraint violations.

However, this approach can be impractical for situations in which *large constraint violations followed by feasible solutions "far enough from the constraint boundaries" to compensate* are not desirable. We may want to avoid large constraint violations at any point in time, regardless of whether we are able to compensate later on or not.

Hence, (Yuan and Lamperski, 2018) focuses instead on $\sum_{t=1}^{T}([g(x_t)]_+)^2$, where $[g(x_t)]_+ = max\{g(x_t), 0\}$. The authors call this term *"square-clipped long-term constraint"* or *"square-cumulative constraint"*. When a solution is strictly feasible (i.e. $g(x_t) < 0$), its contribution to the sum is 0. On the other hand, when $g(x_t) > 0$, the constraint violation is heavily penalized.

6.2 Results

In this framework, (Yuan and Lamperski, 2018) achieves the following results.

- For convex $f_t(x)$ and user-defined trade-off parameter $\beta \in (0, 1)$:
  - $O(T^{max\{\beta, 1-\beta\}})$ regret
  - $O(T^{1-\beta})$ square-cumulative constraint
  - $O(T^{1-\beta/2})$ cumulative constraint
- For strongly convex $f_t(x)$:
  - $O(log(T))$ regret
  - $O(\sqrt{log(T)T})$ cumulative constraint violation

6.3 Methodology

The authors start by observing the methodology of two previous papers: the basic projected gradient algorithm defined by (Zinkevich, 2003a) and the augmented Lagrangian function defined by (Mahdavi et al., 2012). In the latter, we enhance Zinkevich's algorithm by taking the sub-gradient of $\mathcal{L}_t(x, \lambda) = f^t(x) + \sum_{i=1}^{m}\{\lambda_i g_i(x) - \frac{\sigma\eta}{2}\lambda_i^2\}$ instead of taking the sub-gradient of the loss function $f^t(\cdot)$. In addition, instead of projecting onto the feasible set $\mathcal{X}$, the update rule projects onto a fixed ball $\mathcal{B}$ which is a superset of $\mathcal{X}$. This allows lower computational complexity for the projection operation, with the trade-off that the constraints may not be respected at every time step t. Then, at each time step, we update $x_{t+1} = \Pi_{\mathcal{B}}(x_t - \eta\nabla_x\mathcal{L}_t(x_t, \lambda_t))$ and $\lambda_{t+1} = \Pi_{[0,+\infty)^m}(\lambda_t + \eta\nabla_\lambda\mathcal{L}_t(x_t, \lambda_t))$ where η is a pre-determined step size and σ is a pre-determined constant.

With this background, the authors proceed to return to the very premise of this paper. Instead of simply using the constraint functions $g_i(x)$ in the augmented Lagrangian function, they use $[g_i(x)]_+$ to solve the problem $\min_{x_1,x_2,\ldots,x_T\in\mathcal{B}} \sum_{t=1}^{T} f_t(x_t) - \min_{x\in S} \sum_{t=1}^{T} f_t(x)$ s.t. $\sum_{t=1}^{T}([g_i(x_t)]_+)^2 \leq O(T^\gamma), \forall i$ and for $\gamma \in (0, 1)$. Hence, we aim to achieve a sub-linear bound for the square-cumulative constraint, as opposed to the cumulative constraint. In addition, the authors introduce a time-varying constant θ_t to change the augmented Lagrangian function to $\mathcal{L}_t(x, \lambda) = f_t(x) + \sum_{i=1}^{m}\{\lambda_i[g_i(x)]_+ - \frac{\theta_t}{2}\lambda_i^2\}$.

Finally, the algorithm performs at each time step t the updates $x_{t+1} = \Pi_{\mathcal{B}}(x_t - \eta\partial_x\mathcal{L}_t(x_t, \lambda_t))$ and $\lambda_{t+1} = \frac{[g(x_{t+1})]_+}{\sigma\eta}$. We make two observations: the projection onto a ball implies that the constraints are not necessarily satisfied at each time step, and $\partial_x\mathcal{L}_t(x_t, \lambda_t)$ now only contains the sub-gradient with respect to constraints $g_i(x)$ that are violated.

This algorithm improves upon existing bounds for the special case when $f_t(x)$ is strongly convex. We recall the notion of strong convexity as follows: a function f is strongly convex if $\forall x, y$ and for some $\mu > 0$, $f(y) \geq f(x) + \nabla f(x)^T(y - x) + \frac{\mu}{2}\|y - x\|^2$. Intuitively, this provides a quadratic lower bound on the growth of f. In this setting, the authors use time-varying step sizes η_t and constants θ_t to obtain lower bounds on regret and cumulative constraint violation as detailed in 6.2.

6.4 Applications

The authors perform numerical experiments on two examples: a doubly-stochastic matrix approximation problem and an economic dispatch problem from power systems. We will focus on the latter as it is similar to the application seen in (Yu et al., 2017) and thus illustrates the differences between the two papers.

In the economic dispatch problem, we aim to allocate power generation among the units such that the constraints imposed are satisfied and the energy requirements are minimized (Karthikeyan et al., 2013). The loss function to be minimized is now the cost of electricity *plus* the power demand that is not fulfilled. This corresponds to the online convex optimization setting as the loss function is not known a priori. On the other hand, the constraints are no longer stochastic: we impose constraints on the emission levels of each generator and also require each generator to stay within its power generation limit. Both the emission levels and the power generation limit are known in advance. As such, we see the application of different constraint frameworks to two similar problems.

Through numerical experimentation, the authors show that the proposed algorithm indeed encourages small constraint violations for each time step – in comparison with previous algorithms – while maintaining running average objective cost very close to the best fixed strategy in hindsight. Hence, in situations in which it is desirable to minimize constraint violations at each time step, the algorithm introduced by (Yuan and Lamperski, 2018) provides an improvement upon previous approaches.

7 Boosting for Online Convex Optimization

7.1 Motivation

In the classical setting of predicting from expert advice, a learner iteratively makes decisions and receives loss/reward values for it's decisions. Since the loss functions are adversarially decided, a bound on the absolute loss is not possible, however there exists a pool of experts, which have the least loss/most reward for each loss function. Hence, the goal now becomes to minimize regret, i.e., the difference between the total loss of the learner and the expert. For this problem, it has been established in (Littlestone and Warmuth, 1994) to be bounded by $O(\sqrt{T \log |\mathcal{H}|})$ and this bound is tight. Here $|\mathcal{H}|$ is the number of experts. However, these algorithms have a running time linear in $O(|\mathcal{H}|)$, which is computationally infeasible.

To fix the problem, the authors come up with a method of using a weak-learner, that is slightly better than random guessing, and employ boosting to achieve a more computationally efficient algorithm and regret bound. They perform this boosting on these weak learners which perform comparably to the convex hull of the base hypothesis class, with near optimal regret. The authors consider different information feedback models such as full gradient feedback and linear bandit (function value) feedback.

Online Convex Optimization (OCO) generalizes the problem of prediction from expert advice to a general convex decision set $\mathcal{K} \subseteq \mathbb{R}^d$, and adversarially chosen convex loss functions $f_t : \mathbb{R}^d \mapsto \mathbb{R}$. The authors consider a hypothesis class $\mathcal{H} \subseteq \mathcal{C} \mapsto \mathcal{K}$, where each hypothesis $h \in \mathcal{H}$ maps a context (or feature vector) in the context set $c_t \in \mathcal{C}$ to an action $h(c_t) \in \mathcal{K}$.

Assuming we have $\mathcal{W}$, an algorithm which is a γ-weak learner for the hypothesis class $\mathcal{H}$ which performs better than a random guess, so for unit linear loss functions f (unit linear implies it's range of values is within 1) it's total loss function is bounded by:

$$\sum_{t=1}^{T} f_t(\mathcal{W}(c_t)) \leq \gamma \cdot \min_{h \in \mathcal{H}} \sum_{t=1}^{T} f_t(h(c_t)) + (1-\gamma) \sum_{t=1}^{T} f_t^{\mu} + \text{Regret}_T(\mathcal{W}),$$

Where f_t^{μ} is the average value of this function under the uniform distribution, which is equivalent to randomly guessing.

7.2 Results

In this paper, the authors show the the regret bound of the boosting algorithm for the full gradient feedback as:

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) \; - \min_{h \in CH(\mathcal{H})} \sum_{t=1}^{T} f_t(h(c_t)) \; \leq \; \frac{4GDT}{\gamma\sqrt{N}} + \frac{2GD}{\gamma} \text{Regret}_T(\mathcal{W}).$$

In this way, it is dependant only on the parameter N-number of weaker learners and γ-advantage of the weak learner. Similarly, for the linear bandits feedback, the bound is:

$$\mathbb{E}\left[\sum_{t=1}^{T} f_t(\mathbf{x}_t)\right] \; - \inf_{h \in CH(\mathcal{H})} \sum_{t=1}^{T} f_t(h(c_t)) \; \leq \; GD\sqrt{T\left(\frac{16dT}{\gamma\sqrt{N}} + \frac{8d}{\gamma}\text{Regret}_T\left(\mathcal{W}\right)\right)}$$

Thirdly, they do the analysis for the stochastic context as well, getting the following bound:

$$F(\mathcal{A}) \; - F(h^{\star}) \; \leq \; 2\delta G^2 + \frac{2D^2}{\delta\gamma^2 N} + \frac{\hat{G}D}{\gamma}\varepsilon$$

Lastly, they run their algorithms on 3 real-world datasets, and use 3 weak learners (ridge regression, decision stumps, and tiny MLPs) to obtain improvement in loss functions with upto 5 boosted learners, gaining more than 15% improvement in this loss w.r.t a single weak learner.

7.3 Methodology

Similar to a regular boosting algorithm for learning theory, the authors determine an algorithm for these weak learners.

Full Gradient Feedback

Firstly, in the full gradient setting, in every round, they define points

$$\mathbf{x}_t^i = (1 - \eta_i)\mathbf{x}_t^{i-1} + \eta_i \frac{1}{\gamma} \mathcal{W}^i(c_t)$$

, where $\eta_i = \min\{\frac{2}{i}, 1\}$, the initial x_t^0 is sampled from convex set $\mathcal{K}$ at random, c_t is the contextual input for that round, and $\mathcal{W}^i(c_t)$ is the action obtained from the context using the weak learner. The final action to be taken is

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

Now using this action, we incur a loss of $f_t(x_t)$. The main challenge here is that the weak learner only has a γ-approximate regret guarantee, which is scaled by a factor of $\frac{1}{\gamma}$. However, the action space is still $\mathcal{K}$ and not $\frac{1}{\gamma}\mathcal{K}$. To combat this, they modify the loss function to extend outside $\mathcal{K}$

$$\hat{f}_t = X_{\mathcal{K},\delta,\kappa}[f_t](x) = M_\delta[f_t(x) + \kappa \cdot \mathbf{Dist}(\mathbf{x}, \mathcal{K})]$$

Here $\mathbf{Dist}(\mathbf{x}, K) = \min_{\mathbf{y} \in \mathcal{K}} \|\mathbf{y} - \mathbf{x}\|$ and $M_\delta[f] = \inf_{y \in \mathbb{R}^d} \left\{ f(y) + \frac{1}{2\delta}\|x - y\|^2 \right\}$ is the Moreau-Yosida regularization, which is a smoothing operator.

Now on obtaining a compatible loss function with our weak learners space, we can calculate the gradient to be passed back to the learner as $f_t^i(\mathbf{x}) = \nabla \hat{f}_t(\mathbf{x}_t^{i-1}) \cdot \mathbf{x}$. In this way, the authors create an algorithm to boost weak learners.

For the proof of the regret bound of this algorithm, the authors first go about proving a regret bound for the converted loss function (the chosen x_t^N vs optimal x^*). They do this by using the smoothness and linearity of $\hat{f}_t$ and f_t^i. Lastly, they use a recurrence summation to arrive at the required proof.

Linear Bandits Feedback

In the bandits setting, the cost function is linear and the feedback information is only the cost incurred (not the gradient). For this they modify the algorithm by first sampling b_t from a bernoulli distribution with parameter η. If b_t = 0, they play the same x_t from the previous algorithm and pass $f_t = 0$ back. Else they play $x_t = i_t$ where i_t is a random coordinate basis vector $\in \mathbb{R}^d$. They then calculate the modified loss function $\hat{f}_t$ which is 0 everywhere except at i_t.

$$\hat{f}_t(i_t) = \frac{d}{\eta} \times f_t(i_t)$$

The expectation of this is equal to the loss function.

This algorithm passes the full gradient to the weak learner (rather than just the loss value), and hence it will be sub-linear in the size of the hypothesis class whenever the weak-learner's regret bound is sub-linear as well.

The proof for this follows similar approach to the first algorithm, however we now have an upper bound on the gradients of the loss function $\hat{f}_t$, i.e., $\frac{dG}{\eta}$. This provides us with the required claim, and we see the reduction of the regret bound is slower in the bandits setting as $O(1/\sqrt{N})$ vs $O(1/N)$ for the full gradient information case.

Furthermore, the authors describe one section on stochastic context optimization, which has an analogous derivation but can be useful in certain contexts. Lastly, they also showcase experimental results of their boosting algorithm on 3 real world datasets, showing significant improvement at times (18%) for certain cases.

8 Optimal Algorithms for Online Convex Optimization with Multi-Point Bandit Feedback

8.1 Motivation

The authors define Online Convex Optimization as a game between a player and an adversary, where the player picks a move x_t from a convex set $\mathcal{K} \subseteq \mathbb{R}^d$ for round t. The adversary then picks a loss function $l_t : \mathcal{K} \mapsto \mathbb{R}$ and the player incurs a loss $l_t(x_t)$. The goal of the player is to minimize his regret, defined as:

$$\sum_{t=1}^{T} \ell_t\left(x_t\right) - \min_{x \in \mathcal{K}} \sum_{t=1}^{T} \ell_t(x).$$

This is a measure of the difference between the players strategy and the best possible point (chosen in hindsight). The authors talk about (Zinkevich, 2003b) who proves a regret bound of $O(\sqrt{T})$ when $\mathcal{K}$ is compact and the loss functions are Lipschitz continuous. Similarly, (Hazan et al., 2007) prove a regret bound of $O(\log T)$ if the loss functions are strongly convex.

However, if the player only has access to partial information, such as in the bandits scenario, where the adversary only reveals the value of the loss function at x_t instead of the entire loss function l_t, so the player doesn't know the gradient of l_t and cannot use any gradient descent techniques. Furthermore, since the adversary plays second, they can always pick a loss function which will make the regret bound linear (Since multiple loss functions exist for the given point, and the adversary can choose to play all of them).

So, they create a fair competition for the player, the adversary can only use the players past moves $\{x_1, x_2, \ldots, x_{t-1}\}$ to decide the loss function l_t and will not be able to see x_t. For this setting, (Abernethy and Rakhlin, 2009) prove a $O(\sqrt{T})$ regret bound that holds with a high probability. Furthermore, even if the loss functions are strongly convex, (Dani et al., 2008) proved that the regret bound will still be $O(\sqrt{T})$.

The authors compare the regret bounds of the Online Convex Optimization problem for the full information setting $O(\log T)$ and the bandits case $O(\sqrt{T})$, and seeing significant discrepancy, look to find methods to bring the bandits bound closer to the full information case.

For this, they introduce a multi-point bandits problem, where the player can query the loss function at k randomized points, rather than just one. So the loss function now becomes:

$$\mathbb{E}\frac{1}{k}\sum_{t=1}^{T}\sum_{i=1}^{k} l_t(y_{t,i}) - \min_{x \in \mathcal{K}} \mathbb{E}\sum_{t=1}^{T} l_t(x)$$

Where the expectation is over the randomness of the player.

8.2 Results

Building on the randomized gradient descent algorithm of (Flaxman et al., 2005), the authors show a variant of it for $k = 2$ case which has a high probability regret bound of $\tilde{O}(\sqrt{T})$ for convex Lipschitz-continuous loss functions chosen by an adaptive adversary. They also prove an expected regret

bound of $O(\log(T))$ for strongly convex loss functions chosen by an adaptive adversary. This bound is comparable to the one for the full gradient information setting, implying that observing two points is almost as powerful as observing the full loss function.

For $k = d + 1$ where d is the dimension of the space, the authors show a deterministic algorithm that can obtain a regret of $O(\sqrt{T})$ against convex Lipschitz and smooth loss functions, and a regret of $O(\log(T))$ against strongly convex and smooth loss functions. Moreover, these bounds hold even when the adversary knows the action(s) of the player at timestep t as well, before picking the loss function, similar to the full gradient information case. They conclude that having the ability to evaluate the loss function at $d + 1$ points on each round is as powerful as observing the entire function when the functions are smooth.

8.3 Methodology

For each of the two cases ($k = 2$ and $k = d + 1$), the authors provide an algorithm to take actions.

2 queries per round

In this proposed algorithm, the factors are the learning rate η_t, exploration parameter δ and shrinkage coefficient ξ. They initialize $x_1 = 0$.

In every round, they pick a unit vector u_t at random (random point on a origin centered unit ball), and set the two points to be queried as $x_t + \delta u_t$ and $x_t - \delta u_t$. After quering the loss function at these two points, they calculate the estimator for the gradient $\tilde{g}_t = \frac{d}{2\delta}\left(l_t(x_t + \delta u_t) - l_t(x_t - \delta u_t)\right) u_t$. This is similar to (Flaxman et al., 2005), which evaluates this gradient for a single point. The authors here take the difference over two points. Lastly, they get the x_{t+1} for the next round as $x_{t+1} = \Pi_{(1-\xi)\mathcal{K}}\left(x_t - \eta_t \tilde{g}_t\right)$ where $\Pi_{(1-\xi)\mathcal{K}}$ is the projection of the point onto a shrunk convex set. This set is shrunk so that the two points next round still belong to $\mathcal{K}$.

The key insight in this proof is that for the entire class of Lipschitz continuous functions, one can use two function evaluations to construct gradient estimators that have a bounded norm, which leads to much improved regret bounds. They get a bound for $\|\tilde{g}_t\| \leq Gd$ using the property of G-Lipschitz functions.

Furthermore, the authors obtain the regret bound with probability $\geq (1 - \delta_1)$

$$\sum_{t=1}^{T} \frac{1}{2}\left(\ell_t\left(y_{t,1}\right) + \ell_t\left(y_{t,2}\right)\right) - \sum_{t=1}^{T} \ell_t(x) \leq \left(d^2G^2 + D^2\right)\sqrt{T} + G\log(T)\left(3 + \frac{D}{r}\right) + 8dGD\sqrt{2T\log\left(1/\delta_1\right)}$$

which is of order $O(\sqrt{T})$. Here, $\|x\| \leq D \forall x \in \mathcal{K}$.

Similarly, if we take the assumption that l_ts are strongly convex, we get the following bound

$$\mathbb{E}\sum_{t=1}^{T} \frac{1}{2}\left(\ell_t\left(y_{t,1}\right) + \ell_t\left(y_{t,2}\right)\right) - \min_{x\in\mathcal{K}} \mathbb{E}\sum_{t=1}^{T} \ell_t(x) \leq G\log(T)\left(\frac{d^2G}{\sigma} + 3 + \frac{D}{r}\right)$$

which is of order $O(\log T)$.

The main idea behind this proof is to use the G-Lipschitz continuous property of these functions and derive bounds for them. Moreover, since we know $\|x\| \leq D \forall x \in \mathcal{K}$, we can use this property to derive a bound for the difference of the expectation. Lastly, they use the Hoeffding-Azuma inequality to bound the probability as $(1 - \delta_1)$, to acheive the required bound.

An interesting open problem that the authors propose is to find a high probability bound for the strongly-convex case, since their proof only calculates the bound in expectation.

General Smooth Functions

In the next section, the authors move past the unit ball constraint to general estimators based on random sampling with other probability distributions. They prove the same, but with an additional L-smoothness assumption, i.e., their gradient is L-Lipschitz continuous. They do this by bounding the difference between the expectation of the gradient estimator and the true gradient.

$$\|\mathbb{E}_t \tilde{g}_t - \nabla \ell_t(x_t)\| \leq \frac{dL\delta}{4}$$

Furthermore, the authors give some general conditions the generalized gradient estimator must fulfill, mainly (i) $\|x_t - y_{t,i}\| \leq \delta$ for $i = 1, \ldots, k$, (ii) $\|\tilde{g}_t\| \leq G_1$ for some constant G_1 and(iii) $\|\mathbb{E}_t \tilde{g}_t - \nabla \ell_t(x_t)\| \leq c\delta$ for some constant c. If these hold then:

$$\mathbb{E}\sum_{t=1}^{T} \frac{1}{k}\sum_{i=1}^{k} \ell_t(y_{t,i}) - \mathbb{E}\sum_{t=1}^{T} \ell_t(x) \leq \frac{G_1^2}{2}\sum_{t=1}^{T} \frac{1}{\sigma_{1:t}} + G\log(T)\left(1 + 2c + \frac{D}{r}\right).$$

The proof of this involves taking the gradient bound and then the expectation of the expression to arrive at the result.

d+1 queries per round

In this algorithm, to query $(d+1)$ points, the authors query the point x_t (same as the previous algorithm) and d points as $x_t + \delta e_i$ where e_i are the coordinate axis (for each of the d dimensions). Lastly, the gradient estimator is calculated as

$$\tilde{g}_t = \frac{d}{2\delta}\left(\ell_t(y_{t,1}) - \ell_t(y_{t,2})\right) e_{i_t}.$$

So they can apply the theorem from the previous question to get an $O(\log T)$ bound.

Deterministic algo for adaptive adversaries

In the last section, the authors extend their results to adversaries who can decide their loss function l_t after seeing the query point(s) $x_{t,i}$. So, in this section, rather than a gradient descent based algorithm, the authors describe a deterministic algorithm for achieving the optimal regret bound. It is a modification of the Newton Step Algorithm, called the e Bandit Online Newton-Step algorithm (BONES).

Here, the authors showcase a matrix projection using $A_t = \tilde{g}\tilde{g}^T$ and use this to find the next x_{t+1}. Their proof involves proving that the update step for the BONES algorithm is equivalent to the Newton Step algorithm, and using the bounds proven from it. They follow the proof from (Hazan et al., 2007) to get the final bounds as

$$\sum_{t=1}^{T} \frac{1}{d+1}\left(\ell_t(x_t) + \sum_{i=1}^{d} \ell_t(x_t + \delta e_i)\right) - \min_{x\in\mathcal{K}} \sum_{t=1}^{T} \ell_t(x) \leq 4d\left(GD + \frac{1}{\alpha}\right)\log\left(1 + \frac{Td^4D^2}{64}\right) + \log(T)\left(\frac{DL\sqrt{d}}{2} + \frac{2GD}{r}\right) + o(\log(T))$$

The main idea involves using a second-order property of exp-concave functions ((Hazan et al., 2007) Lemma 3).

9 Conclusions

Online convex optimization (OCO) problems come in many flavors depending on the formulation of the constraints as well as the decision maker's access to information. In this survey, we have examined four such flavors: OCO with stochastic constraints, with cumulative constraints, with boosting, and with multi-point bandit feedback. Each provides a unique approach to addressing OCO problems in different settings, and leads to reflection on the strategies one may use to best adapt to the problem in hand.

References

Jacob Abernethy and Alexander Rakhlin. 2009. Beating the adaptive bandit with high probability. In *2009 Information Theory and Applications Workshop*, pages 280–289. IEEE.

Alekh Agarwal and Ofer Dekel. 2010. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *Colt*, pages 28–40. Citeseer.

Varsha Dani, Thomas P Hayes, and Sham M Kakade. 2008. Stochastic linear optimization under bandit feedback. In *Proceedings of the 21st Annual Conference on Learning Theory*.

Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. 2005. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 385–394.

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

Elad Hazan and Karan Singh. 2021. Boosting for online convex optimization. In *International Conference on Machine Learning*, pages 4140–4149. PMLR.

V. Karthikeyan, S. Senthilkumar, and V. J. Vijayalakshmi. 2013. A new approach to the solution of economic dispatch using particle swarm optimization with simulated annealing. *CoRR*, abs/1307.3014.

N. Littlestone and M.K. Warmuth. 1994. The weighted majority algorithm. *Information and Computation*, 108(2):212 – 261.

Mehrdad Mahdavi, Rong Jin, and Tianbao Yang. 2012. Trading regret for efficiency: online convex optimization with long term constraints. *The Journal of Machine Learning Research*, 13(1):2503–2528.

Shie Mannor, John N Tsitsiklis, and Jia Yuan Yu. 2009. Online learning with sample path constraints. *Journal of Machine Learning Research*, 10(3).

Arkadi Nemirovski. 2004. 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*, 15(1):229–251.

Hao Yu, Michael Neely, and Xiaohan Wei. 2017. Online convex optimization with stochastic constraints. *Advances in Neural Information Processing Systems*, 30.

Jianjun Yuan and Andrew Lamperski. 2018. Online convex optimization for cumulative constraints. *Advances in Neural Information Processing Systems*, 31.

Martin Zinkevich. 2003a. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, page 928–935. AAAI Press.

Martin Zinkevich. 2003b. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pages 928–936.