# Projection-free Algorithms for Online Convex Optimization with Adversarial Constraints

**Dhruv Sarkar**
IIT Kharagpur, India

**Aprameyo Chakrabartty**
IIT Kharagpur, India

**Subhamon Supantha**
IIT Bombay, India

**Palash Dey**
IIT Kharagpur, India

**Abhishek Sinha**
TIFR Mumbai, India

## Abstract

We study a generalization of the Online Convex Optimization (OCO) framework with time-varying adversarial constraints. In this setting, at each round, the learner selects an action from a convex decision set $\mathcal{X}$, after which both a convex cost function and a convex constraint function are revealed. The objective is to design a computationally efficient learning policy that simultaneously achieves low regret with respect to the cost functions and low cumulative constraint violation (CCV) over a horizon of length $T$. A major computational bottleneck in standard OCO algorithms is the projection operation onto the decision set $\mathcal{X}$. However, for many structured decision sets, linear optimization can be performed efficiently. Motivated by this, we propose a *projection-free* online conditional gradient (OCG)-based algorithm that requires only a single call to a linear optimization oracle over $\mathcal{X}$ per round. Our approach improves upon the state of the art for projection-free online learning with adversarial constraints, achieving $\tilde{O}(T^{3/4})$ bounds for both regret and CCV.

Our algorithm is conceptually simple. It constructs a surrogate cost function as a non-negative linear combination of the cost and constraint functions, and feeds these surrogate costs into a novel adaptive online conditional gradient subroutine introduced in this paper. We further extend our framework to the bandit setting, where we show that a new form of surrogate loss is necessary to properly handle bandit feedback—an issue overlooked in prior work. Finally, we develop an efficient Follow-the-Perturbed-Leader (FTPL)-based algorithm, particularly well-suited for online combinatorial optimization problems with discrete actions, which also achieves $O(T^{3/4})$ regret and CCV.

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300. Copyright 2026 by the author(s).

## 1 INTRODUCTION

Online Convex Optimization (OCO) provides a foundational framework for studying decision-making under uncertainty, especially in optimization and machine learning [Hazan, 2022]. In this problem, an online algorithm competes against an adaptive adversary to minimize *regret*, defined as the cumulative cost incurred by the algorithm and the cumulative cost incurred by the best-fixed action in hindsight, where the cost functions are assumed to be convex. We generalize this framework by incorporating adversarial constraints, where convex constraint functions are revealed online along with the cost functions. This problem, commonly known as *Constrained* Online Convex Optimization (COCO), has been extensively studied in the literature. Specifically, on round $t \geqslant 1$, the online algorithm first selects a feasible action $x_t$ from a convex decision set $\mathcal{X}$. After that, the adversary reveals a convex cost function $f_t : \mathcal{X} \mapsto \mathbb{R}$ and a convex constraint function $g_t : \mathcal{X} \mapsto \mathbb{R}$. As a result, the algorithm incurs an instantaneous cost of $f_t(x_t)$ and a constraint violation penalty (in case the current constraint is not satisfied) of $\max(0, g_t(x_t)) \equiv g_t(x_t)^+$. Let $T$ be the number of rounds for which the above game is played. The goal of the online policy is to choose a sequence of feasible actions $\{x_t\}_{t=1}^T$ so that both the regret ($\textsc{Regret}_T$) and the Cumulative Con-

straint Violations ($\text{CCV}_T$), are simultaneously minimized, where we define

$$\begin{aligned} \textsc{Regret}_T(x^\star) &\overset{\text{(def.)}}{=} \sum_{t=1}^{T} f_t(x_t) - \sum_{t=1}^{T} f_t(x^\star), \\ \text{CCV}_T &\overset{\text{(def.)}}{=} \sum_{t=1}^{T} g_t(x_t)^+. \end{aligned} \tag{1}$$

In the above definition, $x^\star \in \mathcal{X}$ is any fixed benchmark which satisfies all constraints, namely $g_t(x^\star) \leqslant 0, \forall t$. We have also derived results while considering a stronger benchmark, one where it does not have to satisfy all the constraints and some violation is allowed. We have dealt with this in more detail in Appendix E. The analysis and the definition of the benchmark follows from Sarkar et al. [2025].

Note that when all constraint functions are identically equal to zero, the COCO problem reduces to the standard OCO problem, where the only objective is to minimize the regret. The COCO problem has been the subject of extensive study, including works by Sinha and Vaze [2024], Garber and Kretzu [2024], Guo et al. [2022], Yi et al. [2021], Mahdavi et al. [2012]. The optimal $O(\sqrt{T})$ regret and $\tilde{O}(\sqrt{T})$ CCV bound for the COCO problem was recently obtained by Sinha and Vaze [2024], who proposed a first-order online algorithm using the AdaGrad subroutine. A major computational bottleneck of their algorithm, common to many standard OCO algorithms, *e.g.,* Online Gradient Descent (OGD), is the Euclidean projection step onto the decision set $\mathcal{X}$. This step requires solving a quadratic optimization problem over the decision set on every round and can be a computational bottleneck for high-dimensional problems, which are quite common in practice.

To get around the above computational inefficiency, many efficient *projection-free* algorithms have been proposed in the literature for the standard OCO problem, albeit with a worse regret guarantee. Instead of the projection step, these algorithms rely on a linear optimization step, which can be efficiently computed for many structured problems of interest, such as the shortest path problem on graphs, semidefinite optimization, optimization over matroid polytopes, optimal ranking problem, etc [Hazan, 2022, Garber and Kretzu, 2024]. Hazan and Kale [2012] proposed the first projection-free algorithm for the standard OCO problem, called Online Conditional Gradient (OCG), which achieves $O(T^{3/4})$ regret. Garber and Kretzu [2022] proposed a refined projection-free algorithm for the standard OCO problem, which achieves $O(T^{3/4})$ regret for all intervals provided that a uniform upper bound to the Lipschitz constants of the cost functions is known *a priori*. Furthermore, assuming access to an efficient membership oracle for the decision set, Mhammedi [2022] proposed an OCO policy that achieves $O(\sqrt{T})$ regret with $\tilde{O}(T)$ many calls to the membership oracle. Coming back to the COCO problem, Garber and Kretzu [2024] recently proposed a projection-free policy for the constrained OCO problem with $O(T^{3/4})$ regret and $O(T^{7/8})$ CCV. They proposed two projection-free algorithms, where one uses convex optimization while the other uses only linear optimization. See Table 1 for a brief review of the known results on COCO and Section A in the Appendix for more discussion on the related works. In this paper, we build upon the general regret decomposition framework developed by Sinha and Vaze [2024]. While their work establishes the foundational approach of using a surrogate loss to bound cumulative constraint violations, our transition to a projection-free setting introduces fundamental algorithmic and analytical challenges. Specifically, we propose a new Lipschitz-adaptive Online Conditional Gradient (OCG) policy. Because our step sizes dynamically adapt to the data-dependent Lipschitz constants of the surrogate losses, our regret bound inherently depends on a logarithmic ratio term involving $L_t/L_{t-1}$. To handle this price of adaptivity, which breaks the exponential potential function analysis used in prior projection-based work Sinha and Vaze [2024], we introduce a novel power-law potential function and a specifically tailored initialization strategy. Through this new analysis, we achieve bounds of $\tilde{O}(T^{3/4})$ for both regret and CCV in the full-information and bandit feedback settings.

We now highlight the major differences of the OCG-based algorithm presented in this paper with the independent and concurrent work of Wang et al. [2025], which addresses the same problem as ours. While both studies achieve similar theoretical guarantees, they are methodologically distinct. A key difference lies in how we address the unknown upper bound on the Lipschitz constant, $\tilde{G}$, for the surrogate loss functions. Wang et al. [2025] employs the **doubling trick**, a general technique for handling unknown parameters in online learning. This method partitions the time horizon into phases and restarts the entire algorithm with a revised estimate for $\tilde{G}$ for the next phase whenever the previous estimate is found to be small. Such restarting techniques do not leverage the information accumulated in prior phases. For a broader discussion on the pitfalls of the doubling trick, we refer the reader to Zhang et al. [2024], Kwon and Mertikopoulos [2014], and Luo and Schapire [2014], where it has been argued that continuously adaptive algorithms perform better in practice. In contrast, our work develops an **adaptive strategy** that continuously adjusts to the ob-

served gradient norms *without* requiring restarts. This allows our algorithm to retain and build upon its accumulated knowledge throughout the entire time horizon. The experiments in Section F suggest that our Lipschitz-adaptive method offers distinctive practical performance gains over methods that rely on the doubling trick.

Furthermore, we identify critical technical issues in the bandit analysis presented by Wang et al. [2025]. Firstly, their analysis assumes $Q(t)$ (the CCV up to round $t$) in the bandit setting to be a deterministic quantity instead of a random variable, which is flawed as the actions in the bandit setting are randomized. More importantly, their surrogate loss function on round $t$ depends on the current action of the algorithm $x_t$ (see Eqn. (6) of [Wang et al., 2025]). While this is permissible in the full-information setting, it is erroneous in the bandit setting, as without further assumptions, it would then be impossible to estimate the full gradient of the loss function through a randomized action. The latter is a subtle yet crucial point, and we provide a comprehensive discussion for the same in Section 4 and Section A.1 in the Appendix. Finally, we present a new FTPL-based algorithm in Section 5 for linear cost and constraint functions.

### 1.1 Our contributions

1. We propose an online learning policy (Algorithm 2) which makes exactly one call to an LP solver every round and yields $\tilde{O}(T^{3/4})$ regret and $\tilde{O}(T^{3/4})$ CCV bounds for COCO (Theorem 3.4). This result improves upon the bound of $O(T^{3/4})$ regret and $O(T^{7/8})$ CCV achieved by Garber and Kretzu [2024] and matches that of the concurrent and independent work by Wang et al. [2025] without using the doubling trick.
2. The proposed algorithm requires only first-order information of the cost and constraint functions (more precisely, $\nabla f_t(x_t), \nabla g_t(x_t)$, and $g_t(x_t)$ on round $t$ [1]), and hence, is efficient. We also extend our algorithm to the bandit feedback setting where only the values of the cost and constraint functions (*i.e.,* $f_t(x_t)$ and $g_t(x_t)$) are revealed. Here we achieve state-of-the-art guarantees of $\tilde{O}(T^{3/4})$ expected regret and $\tilde{O}(T^{3/4})$ expected violation. The bandit analysis also involves the emphasis on some novel and subtle points which have been overlooked in related works, and we provide a comprehensive discussion on the same for the sake of future works. Those ideas could be of independent interest.
3. As a pre-requisite for our algorithm, we propose a Lipschitz-adaptive Online Conditional Gradient (OCG) algorithm for the standard OCO problem (Algorithm 1), deriving the first known scale-free and Lipschitz adaptive regret bounds for OCG. This result might be of independent interest. Crucially, this result allows us to sidestep the use of the impractical doubling trick that related work relied on.
4. In the special case of online linear optimization, where both the cost and constraint functions are linear, we propose a Follow-the-Perturbed-Leader (FTPL)-based policy that attains slightly improved regret and cumulative constraint violation (CCV) bounds of $O(T^{3/4})$ compared to the previous OCG-based algorithm. Moreover, for a broad class of combinatorial problems, such as shortest path and maximum spanning tree, FTPL offers significant computational advantages over OCG. In particular, FTPL directly outputs a feasible combinatorial solution (e.g., a path or a spanning tree), whereas OCG requires the design of a non-trivial sampling procedure to extract such solutions from its (typically fractional) outputs.

[1] For a (not necessarily differentiable) convex function $h$, $\nabla h(x)$ refers to any arbitrary subgradient of $h$ at the point $x$. All logarithms in this paper are taken w.r.t. the natural base.

## 2 PRELIMINARIES: ADAPTIVE OCO ALGORITHMS

Our proposed online policy simply passes a sequence of surrogate cost functions (defined in Eqn. (7)) to an *adaptive* version of the Online Conditional Gradient (OCG) algorithm. The standard non-adaptive version of the OCG algorithm with a fixed step size requires a known uniform upper bound to the Lipschitz constants of all future cost functions [Hazan, 2022, Algorithm 26]. In this problem, we will see that the Lipschitz constants of the surrogate cost functions depend on the past actions of the algorithm. Thus, it is not possible to effectively upper bound the Lipschitz constants *a priori*. To address this issue, we propose an adaptive OCG algorithm that eliminates the need for a priori estimates of Lipschitz constants by leveraging adaptive step sizes. In Section 3, we invoke this algorithm as a subroutine to construct an online policy for the Constrained OCO problem, which enjoys $\tilde{O}(T^{3/4})$ regret and $\tilde{O}(T^{3/4})$ Cumulative Constraint Violations (CCV) bounds. Our algorithm is conceptually simpler than Garber and Kretzu [2024], which makes use of an approximately feasible projection oracle. Furthermore, our algorithm achieves a better CCV bound (see Table 1), which is also reflected in the numerical experiments (see Section F).

While the term *adaptive* is used in various contexts in online learning literature, here it specifically refers to

the above definition. A standard example of an adaptive OCO policy is AdaGrad [Duchi et al., 2011], which was used by Sinha and Vaze [2024] in their COCO policy. Although AdaGrad enjoys a data-dependent adaptive regret bound [Orabona, 2019, Theorem 4.14], it is not projection-free. To the best of our knowledge, tight adaptive regret bounds for the OCG policy have not appeared in the literature before. Closest to this problem is the work by Garber and Kretzu [2022, Theorem 9], who proposed a version of the OCG policy which achieves $O(T^{3/4})$ regret over any time interval. However, their algorithm needs to know a uniform upper bound to the Lipschitz constants of the cost functions and, hence, is not adaptive in our sense. As discussed previously, and also in section A.1, the algorithm proposed in Wang et al. [2025] does not make use of an adaptive OCG and instead runs multiple OCGs sequentially to account for the unknown Lipchitz constants, where each new OCG corresponds to a new estimate of the Lipchitz constant. This involves discarding the previous information whenever a new OCG run is started and thus is not very practically efficient (see Section F).

### 2.1 Adaptive Online Conditional Gradient Algorithm

Recall that the ubiquitous online gradient descent (OGD) algorithm takes a gradient descent step on the last revealed cost function on every round and then projects the resulting (possibly infeasible) action back on the convex decision set $\mathcal{X}$ [Hazan, 2022, Algorithm 8]. OGD uses Euclidean projection, which could be expensive to compute on every round for many decision sets of practical interest (*e.g.,* flow polytope on graphs). On the contrary, the Online Conditional Gradient Algorithm (*a.k.a.* Online Frank-Wolfe) algorithm avoids the expensive projection step altogether and replaces it with a cheaper linear optimization step over the decision set Hazan and Kale [2012]. For many structured decision sets of practical interest, well-known and efficient subroutines exist that can perform linear optimization over the decision set efficiently. Standard examples include Dijkstra's algorithm for linear optimization over the flow polytope, the maximum weight matching algorithm for linear optimization over the Birkhoff-von Neumann polytope, and the greedy algorithm for linear optimization over the Matroid polytope. Hazan and Kale [2012] showed that their proposed OCG algorithm achieves $O(T^{3/4})$ regret for the OCO problem. In Algorithm 1, we propose an adaptive version of the standard OCG policy.

The only difference between the standard non-adaptive version of the OCG policy, given in [Hazan, 2022, Algorithm 27], and Algorithm 1 is that the fixed learning rate $\eta = \frac{D}{2LT^{3/4}}$ of the former has been replaced with a sequence of adaptive learning rates $\eta_t = \frac{D}{2L_tT^{3/4}}$, $t \geqslant 1$, in the latter. Here $L_t$ is the Lipschitz constant of the $t^{\text{th}}$ cost function $\hat{f}_t$. However, because of the adaptive regularization, this simple change requires non-trivial modifications in the statement and the proof of the regret bound, as given by the following theorem.

**Algorithm 1** Adaptive Online Conditional Gradient

1: **Input** : Closed and convex decision set $\mathcal{X}$, Horizon length $T$, sequence of convex cost functions $\{\hat{f}_t\}_{t=1}^T$, where the function $\hat{f}_t$ is $L_t$-Lipschitz, $\text{diam}(\mathcal{X}) = D$
2: **Parameters**: $\eta_t = \frac{D}{2L_tT^{\frac{3}{4}}}, \sigma_t = \min\{1, \frac{2}{\sqrt{t}}\},\ t \geqslant 1$
3: Initialize $x_1 \in \mathcal{X}$ arbitrarily
4: **for** $t = 1 : T$ **do**
5: Play $x_t$, observe $\hat{f}_t$, and compute $\nabla_t \equiv \nabla \hat{f}_t(x_t), L_t = ||\nabla_t||$, and $\eta_t$
6: Define the regularized cumulative cost function

$$F_t(x) = \sum_{\tau=1}^{t} \nabla_\tau^\top x + \frac{\|x - x_1\|^2}{\eta_t}. \tag{2}$$

7: Solve a linear program over $\mathcal{X}$ to compute

$$v_t = \arg\min_{x \in \mathcal{X}} \left\{ \nabla F_t\left(x_t\right)^\top x \right\}. \tag{3}$$

8: Set

$$x_{t+1} = (1 - \sigma_t)x_t + \sigma_t v_t.$$

9: **end for**

**Theorem 2.1.** *Consider a sequence of convex cost functions $\{\hat{f}_t\}_{t\geqslant 1}$, where the function $\hat{f}_t$ is $L_t$-Lipschitz such that the Lipschitz constants $L_t$'s are monotone non-decreasing,* i.e., $L_t \leqslant L_{t+1}, \forall t \geqslant 1$. *Then, the Online Conditional Gradient algorithm with adaptive learning rates, described in Algorithm 1, achieves the following regret bound*

$$\begin{aligned} \textit{Regret}_T &\equiv \sum_{t=1}^{T} (\hat{f}_t(x_t) - \hat{f}_t(x^\star)) \\ &= 38DL_TT^{3/4}\sqrt{\log(TeL_T/L_1)})\Bigg(1 + \\ &\quad \frac{1}{304\sqrt{T}}(\max_{1\leqslant t\leqslant T} \frac{L_t}{L_{t-1}})\Bigg). \end{aligned} \tag{4}$$

*In particular, if $\max_{1\leqslant t\leqslant T} \frac{L_t}{L_{t-1}}$ is upper bounded by a constant, then the regret is dominated by the first term only.*

*Remark* 2.2. It is important to note that assuming the monotonicity of the Lipschitz constants incurs no loss of generality. One can always enforce this prop-

erty by defining a new Lipschitz constant as $\tilde{L}_t := \max_{1\leqslant\tau\leqslant t} L_\tau$. In particular, due to the non-decreasing nature of the CCV variable $Q(t)$, the monotonicity condition naturally holds in the case of the surrogate cost function.

*Remark* 2.3. It is also worth emphasizing that the appearance of the ratio $\max_{1\leqslant t\leqslant T} \frac{L_t}{L_{t-1}}$ in the regret bound above results from the use of a non-proximal regularizer in the underlying FTRL subroutine. This is known as the so-called *off-by-one* problem in the literature [McMahan, 2014].

## 3 PROJECTION-FREE COCO

In this Section, we propose a projection-free online policy for the Constrained OCO (COCO) problem. We make the following assumptions common in the COCO literature [Neely and Yu, 2017, Guo et al., 2022, Sinha and Vaze, 2024, Garber and Kretzu, 2024].

**Assumption 3.1** (Convexity and Lipschitzness)**.** The decision set $\mathcal{X}$ is closed and convex with a finite Euclidean diameter $D$. Furthermore, each of the cost and constraint functions are convex and $G$-Lipschitz, *i.e.*,$|f_t(x) - f_t(y)| \leqslant G||x-y||$, $|g_t(x) - g_t(y)| \leqslant G||x-y||$, $\forall t, x, y \in \mathcal{X}$.

**Assumption 3.2** (Feasible Benchmark)**.** Regret is measured relative to a fixed benchmark action $x^\star$ that satisfies all constraints *i.e.,* $g_t(x^\star) \leqslant 0$, $\forall t \in [T]$.

The following computational assumption is specific to the projection-free literature, which puts an additional restriction on how the decision set $\mathcal{X}$ can be accessed by the algorithm [Garber and Kretzu, 2024].

**Assumption 3.3** (LP Oracle for the decision set $\mathcal{X}$)**.** The decision set $\mathcal{X}$ is accessible only through a linear optimization oracle $\mathcal{O}$, which takes a vector $v$ as input and returns $\arg\min_{x\in\mathcal{X}} \langle v, x\rangle$.

As discussed before, for many structured decision sets $\mathcal{X}$ of practical importance (*e.g.,* flow-polytope), the oracle $\mathcal{O}$ can be implemented efficiently.

### 3.1 Algorithm Design and Analysis

Our proposed algorithm, described in Algorithm 2, simply passes a sequence of surrogate cost functions to the adaptive OCG subroutine given in Algorithm 1.

We now state our main result. The proof is given subsequently.

**Theorem 3.4.** *For $T \geqslant e^2 \geqslant 8$, Algorithm 2 achieves $Regret_T = \tilde{O}(T^{3/4})$, $CCV_T = \tilde{O}(T^{3/4})$.*

**Algorithm 2** Projection-Free COCO

1: **Inputs:** Convex decision set $\mathcal{X}$ of diameter $D$, sequence of convex cost functions $\{f_t\}_{t\geqslant 1}$, and convex constraint functions $\{g_t\}_{t\geqslant 1}$ with Lipschitz constant $G$
2: **Parameters:** $\Phi(x) = x^p, V = (cp^2T^{3/4})^p, p = \log T, c = 144GD, Q(0) = GD\log T$.
3: Initialize $x_1 \in \mathcal{X}$ arbitrarily
4: **for** $t = 1 : T$ **do**
5: Play $x_t$, observe $f_t(\cdot)$, and $g_t(\cdot)$,
6: $Q(t) = Q(t-1) + g_t(x_t)^+$
7: Compute $\nabla \hat{f}_t(x_t)$ from Eqn. (7) and pass the (sub) gradient to Algorithm 1
8: **end for**

### 3.2 The Regret Decomposition Inequality

Let $\Phi : \mathbb{R}_{\geqslant 0} \mapsto \mathbb{R}_{\geqslant 0}$ be a convex, differentiable, non-decreasing function, referred to as the *Lyapunov* function. The function $\Phi(\cdot)$ will be used to control the CCV, which satisfies the following recursion:

$$Q(t) = Q(t-1) + g_t(x_t)^+, \quad t \geqslant 1.$$

For analytical convenience, $Q(0)$ is initialized as $GD\log T$, a choice explained in subsequent sections. Thus $Q(T)$ constitutes an upper bound to $\text{CCV}_T$, defined in (1). Using the convexity of the function $\Phi(\cdot)$, we have

$$\Phi(Q(t-1)) \geqslant \Phi(Q(t)) + \Phi'(Q(t))(Q(t-1) - Q(t)).$$

Hence the increase of the Lyapunov function on round $t$ can be bounded as follows:

$$\Phi(Q(t)) - \Phi(Q(t-1)) \leqslant \Phi'(Q(t))g_t(x_t)^+. \tag{5}$$

We now fix a feasible action $x^\star \in \mathcal{X}$ so that $g_t(x^\star) \leqslant 0$. Hence, from Eqn. (5), we have

$$\begin{aligned} & \Phi(Q(t)) - \Phi(Q(t-1)) + Vf_t(x_t) - Vf_t(x^\star) \\ \leqslant\ & Vf_t(x_t) + \Phi'(Q(t))g_t^+(x_t) \\ & -Vf_t(x^\star) - \Phi'(Q(t))g_t^+(x^\star). \end{aligned} \tag{6}$$

We now define the *surrogate cost* function $\hat{f}_t$ for round $t$ as:

$$\hat{f}_t(x) = Vf_t(x) + \Phi'(Q(t))g_t^+(x), \quad x \in \mathcal{X}, \tag{7}$$

where we have used the fact that since the adversary is allowed to be adaptive, the definition of the surrogate cost for round $t$ can depend on the action $x_t$. Thus inequality (6) takes the following form:

$$\begin{aligned} & \Phi(Q(t)) - \Phi(Q(t-1)) + V\big(f_t(x_t) - f_t(x^\star)\big) \\ & \leqslant \hat{f}_t(x_t) - \hat{f}_t(x^\star). \end{aligned}$$

Summing up the above, we arrive at the following Regret Decomposition Inequality which forms the basis for the subsequent analysis:

$$\Phi(Q(t)) + V\text{Regret}_t \leqslant \Phi(Q(0)) + \text{Regret}'_t, \tag{8}$$

where $\text{Regret}_t$ on the LHS where $\text{Regret}'_t$ on the RHS correspond to the Regrets of the original and surrogate cost functions up to round $t$ respectively.

### 3.3 Regret and CCV Bound

Since the cost and constraint functions are assumed to be $G$-Lipschitz, from Eqn. (7), it follows that the surrogate cost $\hat{f}_t$ is an $L_t$-Lipschitz convex function where

$$L_t = G(V + \Phi'(Q(t)). \tag{9}$$

Furthermore, since $\Phi(\cdot)$ is convex and $Q(t)$ is monotone non-decreasing, it follows that $\{L_t\}_{t\geqslant 1}$ is also a monotone non-decreasing sequence. From Eqn. (8), we have

$$\Phi(Q(T)) + V\text{Regret}_T \leqslant \Phi(Q(0)) + \text{Regret}'_T \tag{10}$$

We choose a power-law potential function of the form $\Phi(x) = x^p$ with $p = \log T$. From Lemma B.4, we have $\max_{1\leqslant t\leqslant T} \frac{L_t}{L_{t-1}} \leqslant 1 + e$. Hence, the regret of the OCG policy used on the surrogate cost functions, given by Theorem 2.1, can be upper bounded as:

$$\text{Regret}'_T \leqslant 48DL_T T^{3/4}\sqrt{\log(TeL_T/L_1)}).$$

Substituting for the expression of $L_T$ from Eqn. (9) and using Lemma B.5, stated below, we can further simplify the above regret bound as:

$$\text{Regret}'_T \leqslant 144GDT^{3/4}(V + pQ^{p-1}(T))\log T.$$

Combining the above regret bound with Eqn. (10) and letting $c \equiv 144GD$, we arrive at the following inequality which will be used for simultaneously bounding the Regret and CCV:

$$Q^p(T) + V\text{Regret}_T \leqslant (GDp)^p + cpT^{\frac{3}{4}}(V + pQ^{p-1}(T)), \tag{11}$$

where we have used the fact that $p = \log T$.

**Analysis:** From the Lipschitzness of the cost functions, we trivially have $\text{Regret}_T \geqslant -GDT$. This implies that

$$\begin{aligned} Q^p(t) &\leqslant 2cVT + (GDp)^p + cp^2T^{\frac{3}{4}}\,Q^{p-1}(T) \\ &\leqslant 2\max(2cVT + (GDp)^p, cp^2T^{\frac{3}{4}}\,Q^{p-1}(T)). \end{aligned}$$

Hence,

$$Q(T) \leqslant \max((4cVT)^{\frac{1}{p}} + 2GDp, 2cp^2T^{\frac{3}{4}}). \tag{12}$$

The above inequality bounds the CCV. To bound the Regret, using Eqn. (11) once again, we have

$$\begin{aligned} &\text{Regret}_T \\ &\leqslant cpT^{\frac{3}{4}} + \frac{(GDp)^p}{V} + \frac{cp^2T^{\frac{3}{4}}Q^{p-1}(T) - Q^p(T)}{V} \\ &\overset{(b)}{\leqslant} cT^{\frac{3}{4}}\log T + \frac{(GDp)^p}{V} + \frac{1}{V}(cp^2T^{\frac{3}{4}})^p. \end{aligned} \tag{13}$$

Inequality (b) follows by considering the possible cases $Q(T) > cp^2T^{3/4}$ and $0 \leqslant Q(T) \leqslant cp^2T^{3/4}$ and upper bounding the last term separately in each case. Finally, we set $V = (cp^2T^{\frac{3}{4}})^p$ and recall that $p = \log T$. Substituting the above into (12) and (13), and simplifying the algebraic expressions, we obtain $\tilde{O}(T^{3/4})$ bounds for both Regret and CCV, which proves Theorem 3.4.

## 4 PROJECTION-FREE CBCO

In this section, we extend our projection-free algorithm to the bandit-feedback setting against an oblivious adversary. This setting has previously been explored in both Garber and Kretzu [2024] and Wang et al. [2025] where they proposed projection-free algorithms for constrained bandit convex optimization (CBCO). Garber and Kretzu [2019] achieved an expected regret bound of $\tilde{O}(T^{3/4})$ and an expected CCV bound of $\tilde{O}(T^{7/8})$. The results of Wang et al. [2025] do not actually hold and we have discussed the same in Appendix A.1. We achieve expected regret bound of $\tilde{O}(T^{3/4})$ and expected CCV bound of $\tilde{O}(T^{3/4})$.

As in Flaxman et al. [2004] we make the standard assumption that the feasible set $\mathcal{X}$ is full dimensional, contains the origin, and that there exists scalars $r, R > 0$ such that $r\mathcal{B}^n \subseteq \mathcal{X} \subseteq R\mathcal{B}^n$, where $\mathcal{B}^n$ denotes the unit Euclidean ball centered at the origin in $\mathbb{R}^n$. Further note that $D = 2R$. Also, as in most bandit convex optimization literature, we assume that we have an oblivious adversary, that is, the cost and constraint functions are chosen ahead of the beginning of the play and are not affected by the randomness of the play.

The key algorithmic distinction in the bandit setting is that, at each round, the decision-maker observes only the incurred cost and constraint violation for the chosen action, rather than the entire cost and constraint functions. Consequently, one cannot compute exact gradients and must instead rely on gradient estimators whose bias and variance affect overall performance. To control the variance we use the *blocking* technique and to ensure that unbiased estimation of gradient is possible, we change the form of the surrogate loss. More details regarding the choice of estimator and notions like smoothing are given in Section C.1.

### 4.1 Adaptive Online Conditional Gradient under Bandit Feedback

**Algorithm 3** Adaptive Block Bandit Conditional Gradient

**Input** : Closed and convex decision set $\mathcal{X}$, Horizon length $T$, sequence of convex cost functions $\{\hat{f}_t\}_{t=1}^T$, where the function $\hat{f}_t$ is $L_t$-Lipschitz and has sup-norm of $M_t$, $\text{diam}(\mathcal{X}) = D$
**Parameters**: $\eta_m = \frac{D}{(\sqrt{n}M_{mK}+L_{mK})T^{\frac{3}{4}}}$, $m \geqslant 1$, $\epsilon = 16R^2T^{-1/2}\sqrt{\log T}$
Initialize $x_1 \in \mathcal{X}$ arbitrarily
**for** $m = 1 : \frac{T}{K}$ **do**
  **for** $s = 1, ....., K$ **do**
    $t = (m-1)K + s$ and $u_t \sim \mathcal{S}^n$
    Play $y_t \leftarrow x_m + \delta u_t$, observe $\hat{f}_t(y_t)$ and compute $\nabla_{m,s} \equiv \frac{n\hat{f}_t(y_t)}{\delta}$
  **end for**
  $\nabla_m \equiv \sum_{s=1}^K \nabla_{m,s}$
  Define the regularized cumulative cost function

$$F_m(x) = \sum_{\tau=1}^{m} \nabla_\tau^\top x + \frac{\|x - x_1\|^2}{\eta_m}. \tag{14}$$

  Run Algorithm 6 with set $\mathcal{X}_\delta$, tolerance $\epsilon$, initial vector $x_m$, and function $F_m(x)$.
  $x_{m+1} \leftarrow$ Output of Algorithm 3
**end for**

Like in the full information setting, we would have to design an adaptive algorithm for the unconstrained setting to which the surrogate loss function would be passed. Therefore, the first step is to design something akin to an adaptive OCG for the bandit setting.

To keep the estimator's variance under control, a naive extension of the OCG algorithm is insufficient. Instead, we employ a *blocking* technique: the time horizon is divided into disjoint blocks of length $K$, and the algorithm holds a single action fixed throughout each block $m$, updating only at the beginning of block $m+1$. By choosing $K$ appropriately, this approach balances variance reduction against any block-wise regret accumulation. We call the resulting method *blocked-OCG*, and we present it as Algorithm 3. We assume without loss of generality that $T$ is perfectly divisible by $K$.

It is important to note that the blocking technique has been used before [Hazan and Minasyan, 2020, Garber and Kretzu, 2019, 2024, Merhav et al., 2002]. However, we mention it here to highlight the distinction between our approach to handling CBCO as opposed to the COCO at an algorithmic level.

The above algorithm calls the subroutine Algorithm 6 as a subroutine at the end of every block and the goal of the subroutine is to approximately optimize $F_m$ using LOO calls. The error tolerance $\epsilon$ is a measure of our proximity to the exact solution. The subroutine works by terminating when the *Frank-Wolfe gap* is less than $\epsilon$. Till the time that a low enough Franke-Wolfe gap is achieved, we keep making LOO calls and improving our decision. More details about Algorithm 6 is given in section C.2.

Note that we are making multiple calls at the end of each block. Thus, we also have to bound the expected total number of LOO calls at the end of $T$ rounds.

We present Theorem 4.1 which presents the bound on both the expected regret and the expected number of LOO calls of the above algorithm.

Note that $N_m$ is the number of LOO calls made at the end of block $m$. Further we define $G_t = \sqrt{n}M_t + L_t$ for the sake of notational convenience. Where $M_t$ and $L_t$ are the sup-norm and Lipchitz constant of the $t$-th loss respectively.

**Theorem 4.1.** *Setting $\eta_m = \frac{D}{G_{mK}}T^{-\frac{3}{4}}$, $\delta = \sqrt{n}T^{-\frac{1}{4}}$, $K = \sqrt{T}$ in Algorithm 3, guarantees that the expected regret is upper-bounded by*

$$\mathbb{E}[Regret_T] \leqslant 4\sqrt{n}L_T T^{\frac{3}{4}} + 3L_T T\sqrt{\epsilon} + DG_T\big(1 + \max_{1\leqslant m\leqslant T/K} \frac{G_{mK}}{G_{(m-1)K}}\big)T^{3/4},$$

*and that the expected overall number of calls to the linear optimization oracle is upper-bounded by*

$$\mathbb{E}\left[\sum_{m=1}^{\frac{T}{K}} N_m\right] \leqslant \frac{16R^4}{\epsilon^2}\left(\frac{\sqrt{\epsilon}}{R}T^{1/4} + 1 + \sqrt{\log\frac{G_T}{G_1}}\right).$$

The proof has been deferred to Appendix C.3.

To achieve an amortized budget of one Linear Optimization Oracle (LOO) call per round (for a total of $T$ calls), we select an appropriate value for the error tolerance $\epsilon$. This choice is crucial as it governs the trade-off between the total number of oracle calls and the final regret bound.

### 4.2 The Regret Decomposition Inequality

One may be tempted to extend the approach in the full information setting and construct the surrogate loss as in there and pass that surrogate loss to an analogous adaptive bandit algorithm. This is essentially what Wang et al. [2025] tried to do. However, that would not be a correct approach. This is because doing so would break one of the most fundamental assumptions in bandit convex optimization - that we would be able to compute an unbiased estimate of the loss function.

Consider that we construct a surrogate loss of the form $\hat{f}_t(x_t) = Vf_t(x_t) + \Phi'(Q(t))g_t^+(x_t)$. Notice that, the definition of the function $\hat{f}_t$ involves $Q(t)$ in it which in itself contains $x_t$. For there to be an unbiased estimate of $\hat{f}_t$, we need to at least make sure that the loss at time $t$ does not depend on the action at time $t$. Ultimately, this is the reason that it is assumed that we have an oblivious adversary as opposed to an adaptive one in the bandit setting. An adaptive adversary in the bandit setting would have the power to decide the loss function based on the action of the player and thus unbiased estimation of gradient would be impossible. In contrast, for an oblivious adversary, the sequence of loss functions does not depend on the randomness of play. However, unbiased estimation is possible even if the loss function depends on the randomness of play as long as it does not depend on the latest action $x_t$. Thus, the loss function can depend on all the actions $x_{<t}$ but not on $x_t$ itself. Accordingly we modify the surrogate loss function, to be of the following form –

$$\hat{f}_t(x_t) = Vf_t(x_t) + \Phi'(Q(t-1) + GD)g_t^+(x_t) \quad (15)$$

In summary, we can say that the above equation implies that the surrogate loss $\hat{f}_t$ is $\mathcal{F}_{t-1}$ measurable, where $\{\mathcal{F}_\tau\}_{\tau\geqslant 1}$ is the natural filtration.

To derive the regret decomposition inequality, we would utilise the fact that the quantity $Q(t-1)+GD$ is an overestimate of $Q(t)$ because $g_t^+(x_t) \leqslant GD$.

**Algorithm 4** Projection-Free CBCO

1: **Inputs:** Convex decision set $\mathcal{X}$, sequence of convex cost functions $\{f_t\}_{t\geqslant 1}$, and convex constraint functions $\{g_t\}_{t\geqslant 1}$.
2: **Parameters:** $Q(0) = GD\sqrt{T}\log T, \Phi(x) = x^{\log T}$, $V = \left((30+\frac{R}{r})\sqrt{n}(\log T)^{1/4}T^{3/4}\right)^{\log T}$
3: Initialize $x_1 \in \mathcal{X}$ arbitrarily
4: **for** $t = 1 : T$ **do**
5: Play $x_t$, observe $f_t(x_t)$, and $g_t(x_t)$,
6: Compute $\hat{f}_t(x_t)$ using equation (15). Pass $\hat{f}_t(x_t)$. to Algorithm 3.
7: **end for**

$$\begin{aligned}\Phi(Q(t)) - \Phi(Q(t-1)) &\leqslant \Phi'(Q(t))g_t(x_t)^+ \\ &\leqslant \Phi'(Q(t-1)+GD)g_t(x_t)^+\end{aligned}$$

$$\begin{aligned}&\Phi(Q(t)) - \Phi(Q(t-1)) + Vf_t(x_t) - Vf_t(x^\star) \\ \leqslant &Vf_t(x_t) + \Phi'(Q(t-1)+GD)g_t^+(x_t) \\ &- Vf_t(x^\star) - \Phi'(Q(t-1)+GD)g_t^+(x^\star).\end{aligned}$$

Summing up inequalities for $1 \leqslant t \leqslant T$ and telescoping, we conclude that

$$\Phi(Q(T)) - \Phi(Q(0)) + V\text{Regret}_T \leqslant \text{Regret}'_T$$

Taking expectation on both sides,

$$\mathbb{E}\Phi(Q(T)) - \Phi(Q(0)) + V\mathbb{E}\text{Regret}_T \leqslant \mathbb{E}\text{Regret}'_T \quad (16)$$

Now we have got something analogous to the regret decomposition inequality for the full-information setting. The only difference is that we are appropriately taking expectation of the regret and violation over the randomness of the play.

$\mathbb{E}\text{Regret}'_T$ is the expected regret incurred by the bandit algorithm when we pass the surrogate loss in eqn. (15) to it. We have shown that in Algorithm 4.

Note that although the surrogate loss (15) on round $t$ depends on the actions of the online algorithm up to round $t-1$, the benchmark $x^\star$, as discussed above, is oblivious to the action of the algorithms and can be decided at the start of the play because $\{f_t\}_{t\geqslant 1}$ and $\{g_t\}_{t\geqslant 1}$ are known apriori by our oblivious adversary. Indeed, the benchmark $x^\star$ would be chosen such that the cumulative loss of $\{f_t\}_{t\geqslant 1}$ is minimized while ensuring feasibility on $\{g_t\}_{t\geqslant 1}$.

Since the action of the algorithm at round $t$ is independent of the losses at round $t$, it is clear that we can upper bound the regret of the surrogate problem against the benchmark $x^\star$ using any oblivious regret bound. Therefore, we take the help of Theorem 4.1 to help us bound $\mathbb{E}\text{Regret}'_T$ by using the form of the surrogate cost.

### 4.3 Expected Regret and Violation

**Theorem 4.2.** *Algorithm 4 achieves* $\mathbb{E}Regret_T = \tilde{O}(T^{3/4}), \mathbb{E}CCV_T = \tilde{O}(T^{3/4})$ *while using an expected total number of less than* $T$ *LOO calls.*

The proof has been deferred to Appendix C.4.

The key feature of the proof is that using the form of the surrogate loss, we can bound $\sqrt{\log\frac{G_T}{G_1}}$ and then set $\epsilon = 16R^2T^{-1/2}\sqrt{\log T}$ to make the expected number of LOO calls less than $T$. From that, we get the bound on $\mathbb{E}\text{Regret}'_T$ as,

$$\mathbb{E}\text{Regret}'_T \leqslant c(V + \mathbb{E}\Phi'(Q(T)))T^{3/4}$$

where $c = (30+\frac{R}{r})\sqrt{n}(\log T)^{1/4}$ and we then plug this in the regret decomposition inequality (16), alongside some manipulations with the Jensen's inequality to get our desired bounds.

The takeaway from our bandit analysis should be that while the extension to CBCO from COCO is quite simple from an algorithmic point of view, the subtleties associated with the form of the surrogate loss that

we assume makes the analysis non-trivial. As demonstrated in Wang et al. [2025] it is quite easy to miss this point and this further highlights our contribution. The points raised are of independent interest even beyond the projection-free setting. In particular, they will be important whenever one tries to use a reduction to BCO to analyse a CBCO problem along the lines of Sinha and Vaze [2024].

# 5 An EFFICIENT FTPL-BASED ALGORITHM IN THE FULL-INFORMATION SETTING

There exist efficient linear programming (LP) subroutines for many combinatorial optimization problems, such as shortest path, maximum matching, and maximum-weight spanning tree. While the OCG algorithm described in the previous sections can, in principle, be applied to the online versions of these problems, by convexifying the feasible set and designing an appropriate sampling procedure, this approach is indirect and often computationally inefficient.

In contrast, the Follow-the-Perturbed-Leader (FTPL) algorithm requires solving only a single LP at each round and directly outputs a feasible solution (e.g., a path, a matching, or a spanning tree), thereby completely bypassing the need for sampling [Abernethy et al., 2016]. In this section, we show that FTPL can be used to solve constrained online linear optimization problems while achieving the same regret and cumulative constraint violation (CCV) guarantees as OCG.

**Setup.** We consider a constrained online linear optimization (OLO) problem over an arbitrary compact (not necessarily convex) set $\mathcal{X}$. For example, $\mathcal{X}$ could be given by the set of all paths, or matchings, or spanning trees of some graph. At each round $t \geqslant 1$, the learner observes a linear cost function $f_t(x) = \langle f_t, x\rangle$ and a linear constraint function $g_t(x) = \langle g_t, x\rangle$, where $\|f_t\|, \|g_t\| \leqslant 1$ for all $t \geqslant 1$. For simplicity, we assume that both sequences are generated by an oblivious adversary. Unlike the previous section, we allow the learning algorithm to be randomized. The objective is to minimize both the expected regret and the expected (signed) cumulative constraint violation, defined as $\mathsf{CCV}_T = \mathbb{E}\left[\sum_{t=1}^T g_t(x_t)\right]^+$. Using a similar Lyapunov-based approach with a quadratic Lyapunov function (see Appendix D for details), we reduce the above constrained problem to an unconstrained OLO problem with modified cost functions. Specifically, at round $t$, we define the linear surrogate cost function

$$\hat{f}_t(x) = \langle V f_t + 2Q(t-1)\, g_t,\ x\rangle, \quad t \geqslant 1,$$

where $V > 0$ is a parameter to be specified later. Applying the FTPL algorithm with Gaussian perturbations to the sequence $\{\hat{f}_t\}_{t=1}^T$ yields the following result.

**Theorem 5.1** (Projection-Free OLO with Adversarial Constraints)**.** *Consider the above* FTPL*-based policy in the full-information setting with $V = \sqrt{T}$. It achieves the following bounds on the expected regret and expected cumulative constraint violation:*

$$\mathbb{E}\mathsf{Regret}_T \leqslant \mathcal{O}(T^{3/4}), \quad \mathbb{E}\mathsf{CCV}_T \leqslant \mathcal{O}(T^{3/4}).$$

The proof has been deferred to Appendix D.

# 6 NUMERICAL SIMULATIONS

To show the practical benefit of our algorithm, we demonstrate the efficacy of our algorithm in the problem of Online Shortest Path with Delays. We have detailed the experiments in section F. We compare our algorithm against three other baselines — the projection-based algorithm proposed by Sinha and Vaze [2024] and the projection-free algorithms of Garber and Kretzu [2024] and Wang et al. [2025]. The Sinha and Vaze [2024] algorithm has better regret and violation performance but orders of magnitude worse computational efficiency. Among projection-free algorithms, ours has the best regret and violation performance. This is theoretically supported by the suboptimal regret and violation bounds of Garber and Kretzu [2024] and the use of the inefficient doubling trick in the case of Wang et al. [2025].

# 7 CONCLUSION

This paper introduces an efficient projection-free policy for OCO problems with time-varying adversarial constraints for both full-information and bandit feedback settings. The proposed algorithm uses a new, Lipschitz-adaptive version of the Online Condition Gradient algorithm and achieves $\tilde{O}(T^{3/4})$ bound for both Regret and CCV metrics in the full information setting, thus improving upon the state-of-the-art results. Additionally, for constrained online linear optimization over combinatorial structures, we present an efficient Follow-the-Perturbed-Leader (FTPL) policy that directly outputs discrete feasible actions without requiring expensive post-hoc sampling, while achieving slightly improved Regret and CCV bounds of $O(T^{3/4})$. In the bandit setting, we use an OCG-type algorithm to obtain the same bounds of $\tilde{O}(T^{3/4})$ on expected regret and violation, and the analysis offers novel perspectives.

## 8 ACKNOWLEDGEMENT

The work of Abhishek Sinha was supported by the Department of Atomic Energy, Government of India, under project no. RTI4014 and by a Google India faculty research award. Palash Dey thanks the Anusandhan National Research Foundation (ANRF) (erstwhile Science, Education, and Research Board (SERB)), Government of India, for supporting this work through Core Research Grant under file no. CRG/2022/003294.

## References

Jacob Abernethy, Chansoo Lee, and Ambuj Tewari. Perturbation techniques in online learning and optimization. *Perturbations, Optimization, and Statistics*, page 233, 2016.

Lilian Besson and Emilie Kaufmann. What doubling tricks can and can't do for multi-armed bandits. *arXiv preprint arXiv:1803.06971*, 2018.

Lin Chen, Christopher Harshaw, Hamed Hassani, and Amin Karbasi. Projection-free online optimization with stochastic gradient: From convexity to submodularity, 2018. URL https://arxiv.org/abs/1802.08183.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. *arXiv preprint cs/0408007*, 2004.

Dan Garber. Efficient online linear optimization with approximation algorithms. *Mathematics of Operations Research*, 46(1):204–220, 2021.

Dan Garber and Ben Kretzu. Improved regret bounds for projection-free bandit convex optimization, 2019. URL https://arxiv.org/abs/1910.03374.

Dan Garber and Ben Kretzu. New projection-free algorithms for online convex optimization with adaptive regret guarantees. In *Conference on Learning Theory*, pages 2326–2359. PMLR, 2022.

Dan Garber and Ben Kretzu. New projection-free algorithms for online convex optimization with adaptive regret guarantees, 2023. URL https://arxiv.org/abs/2202.04721.

Dan Garber and Ben Kretzu. Projection-free online convex optimization with time-varying constraints. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=RnbobOgbn0.

Hengquan Guo, Xin Liu, Honghao Wei, and Lei Ying. Online convex optimization with hard constraints: Towards the best of two worlds and beyond. *Advances in Neural Information Processing Systems*, 35:36426–36439, 2022.

Elad Hazan. Introduction to online convex optimization. *CoRR*, abs/1909.05207, 2019. URL http://arxiv.org/abs/1909.05207.

Elad Hazan. *Introduction to online convex optimization*. MIT Press, 2022.

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

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

Elad Hazan, Alexander Rakhlin, and Peter Bartlett. Adaptive online gradient descent. *Advances in neural information processing systems*, 20, 2007.

Rodolphe Jenatton, Jim Huang, and Cédric Archambeau. Adaptive algorithms for online convex optimization with long-term constraints. In *International Conference on Machine Learning*, pages 402–411. PMLR, 2016.

Joon Kwon and Panayotis Mertikopoulos. A continuous-time approach to online optimization. *arXiv preprint arXiv:1401.6956*, 2014.

Nikolaos Liakopoulos, Apostolos Destounis, Georgios Paschos, Thrasyvoulos Spyropoulos, and Panayotis Mertikopoulos. Cautious regret minimization: Online optimization with long-term budget constraints. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3944–3952. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/liakopoulos19a.html.

Haipeng Luo and Robert Schapire. Towards minimax online learning with unknown time horizon. In *International Conference on Machine Learning*, pages 226–234. PMLR, 2014.

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

H Brendan McMahan. Analysis techniques for adaptive online learning. *arXiv preprint arXiv:1403.3465*, 6, 2014.

N. Merhav, E. Ordentlich, G. Seroussi, and M.J. Weinberger. On sequential strategies for loss functions with memory. *IEEE Transactions on Information Theory*, 48(7):1947–1958, 2002. doi: 10.1109/TIT.2002.1013135.

Zakaria Mhammedi. Efficient projection-free online convex optimization with membership oracle. In *Conference on Learning Theory*, pages 5314–5390. PMLR, 2022.

Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Conditional gradient method for stochastic submodular maximization: Closing the gap, 2017. URL https://arxiv.org/abs/1711.01660.

Michael J Neely and Hao Yu. Online convex optimization with time-varying constraints. *arXiv preprint arXiv:1702.04783*, 2017.

Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

Francesco Orabona and Dávid Pál. Scale-free online learning. *Theoretical Computer Science*, 716:50–69, 2018.

Dhruv Sarkar and Abhishek Sinha. Optimal anytime algorithms for online convex optimization with adversarial constraints. *arXiv preprint arXiv:2510.22579*, 2025.

Dhruv Sarkar, Samrat Mukhopadhyay, and Abhishek Sinha. Online learning for approximately-convex functions with long-term adversarial constraints. *arXiv preprint arXiv:2508.16992*, 2025.

Abhishek Sinha and Rahul Vaze. Optimal algorithms for online convex optimization with adversarial constraints. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=TxffvJMnBy.

RIPE NCC Staff. Ripe atlas: A global internet measurement network. *Internet Protocol Journal*, 18(3): 2–26, 2015.

Wen Sun, Debadeepta Dey, and Ashish Kapoor. Safety-aware algorithms for adversarial contextual bandit. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3280–3288. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/sun17a.html.

Yibo Wang, Yuanyu Wan, and Lijun Zhang. Revisiting projection-free online learning with time-varying constraints, 2025. URL https://arxiv.org/abs/2501.16046.

Julien Weibel, Pierre Gaillard, Wouter M. Koolen, and Adrien Taylor. Optimized projection-free algorithms for online learning: construction and worst-case analysis, 2025. URL https://arxiv.org/abs/2506.05855.

David P Williamson. *Network flow algorithms*. Cambridge University Press, 2019.

Xinlei Yi, Xiuxian Li, Tao Yang, Lihua Xie, Tianyou Chai, and Karl Johansson. Regret and cumulative constraint violation analysis for online convex optimization with long term constraints. In *International Conference on Machine Learning*, pages 11998–12008. PMLR, 2021.

Xinlei Yi, Xiuxian Li, Tao Yang, Lihua Xie, Yiguang Hong, Tianyou Chai, and Karl H Johansson. Distributed online convex optimization with adversarial constraints: Reduced cumulative constraint violation bounds under slater's condition. *arXiv preprint arXiv:2306.00149*, 2023.

Hao Yu and Michael J Neely. A low complexity algorithm with $o(\sqrt{T})$ regret and $o(1)$ constraint violations for online convex optimization with long term constraints. *Journal of Machine Learning Research*, 21(1):1–24, 2020.

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

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

Zhiyu Zhang, Heng Yang, Ashok Cutkosky, and Ioannis C Paschalidis. Improving adaptive online learning using refined discretization. In *International Conference on Algorithmic Learning Theory*, pages 1208–1233. PMLR, 2024.

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

## Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [No]
2. For any theoretical claim, check if you include:

   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [No]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Not Applicable]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

# Appendix

## A Related Work

**Unconstrained OCO:** In a pioneering work, Zinkevich [2003] demonstrated that the well-known projected online gradient descent (OGD) strategy achieves an $O(\sqrt{T})$ regret for OCO with bounded sub-gradients. Subsequently, various studies have introduced adaptive and parameter-free variations of OGD [Hazan et al., 2007, Orabona and Pál, 2018]. Textbook discussions on the OCO framework and related algorithms can be found in Orabona [2019], Hazan [2022].

| Reference | Regret | CCV | Complexity per round | Assumptions |
|---|---|---|---|---|
| Mahdavi et al. [2012] | $O(\sqrt{T})$ | $O(T^{3/4})$ | Proj. | Fixed Constraints |
| Jenatton et al. [2016] | $O(T^{\max(\beta,1-\beta)})$ | $O(T^{1-\beta/2})$ | Proj | Fixed Constraints |
| Yu et al. [2017] | $O(\sqrt{T})$ | $O(\sqrt{T})$ | Proj | Slater & Stochastic Constr. |
| Neely and Yu [2017] | $O(\sqrt{T})$ | $O(\sqrt{T})$ | Proj. | Slater |
| Yu and Neely [2020] | $O(\sqrt{T})$ | $O(1)$ | Conv-OPT | Slater & Fixed Constraints |
| Yi et al. [2021] | $O(\sqrt{T})$ | $O(T^{1/4})$ | Conv-OPT | Fixed Constraints |
| Guo et al. [2022] | $O(\sqrt{T})$ | $O(T^{3/4})$ | Conv-OPT | — |
| Yi et al. [2023] | $O(T^{\max(\beta,1-\beta)})$ | $O(T^{1-\beta})$ | Conv-OPT | Slater |
| Sinha and Vaze [2024] | $O(\sqrt{T})$ | $\tilde{O}(\sqrt{T})$ | Proj | — |
| Garber and Kretzu [2024] | $O(T^{3/4})$ | $O(T^{7/8})$ | LP | — |
| Garber and Kretzu [2024] | $O(T^{3/4})$ | $O(T^{7/8})$ | Conv-OPT | — |
| Wang et al. [2025] | $O(T^{3/4})$ | $\tilde{O}(T^{3/4})$ | LP | Use of doubling trick |
| **This paper** | $\tilde{O}(T^{3/4})$ | $\tilde{O}(T^{3/4})$ | **LP** | — |

Table 1: Summary of the results for the COCO problem with convex cost and convex constraint functions. The parameter $\beta \in [0,1]$ is any fixed constant. Abbreviations have the following meanings- Proj: Euclidean projection on the decision set, Conv-OPT: Solving a convex optimization problem over the decision set, LP: solving a linear optimization problem over the decision set.

**Constrained OCO (COCO):**

**Time-invariant constraints:** A number of papers have explored COCO with time-invariant constraints, where $g_{t,i} = g_i$ for all $t \geqslant 1$ [Yuan and Lamperski, 2018, Jenatton et al., 2016, Mahdavi et al., 2012, Yi et al., 2021]. These studies typically assume that the constraint functions are known beforehand. However, they permit the policy to be infeasible in any round to circumvent the expensive projection step involved in traditional projected OGD policy. The primary aim of this line of work is to design a computationally efficient policy by avoiding the projection step while maintaining small regret and cumulative constraint violation bounds.

**Time-varying constraints:** Tackling the COCO problem with time-varying constraint functions presents greater difficulties. Aside from Neely and Yu [2017] and Liakopoulos et al. [2019], most existing work involves constructing a Lagrangian function and subsequently updating primal and dual variables in an online fashion [Yu et al., 2017, Sun et al., 2017, Yi et al., 2023]. However, the resulting algorithms turn out to be inefficient with sub-optimal regret/CCV bounds. Both Neely and Yu [2017] and Liakopoulos et al. [2019] employ the drift-plus-penalty (DPP) methodology to address the constrained problem under additional assumptions. Specifically, Neely and Yu [2017] proposed a policy grounded on DPP for COCO with Slater's condition, which assumes strict feasibility in the sense that $g_t(x^\star) < -\eta$ for some $\eta > 0$ $\forall t$. This clearly precludes standard constraint functions without slack (such as functions of the form $\max(0, g_t(x))$). Additionally, the bounds derived under Slater's condition depend inversely on the Slater constant $\eta$ (generally hidden under the big-Oh notation). As $\eta$ may be arbitrarily small, these bounds might be excessively loose. Liakopoulos et al. [2019] augmented Neely and Yu [2017]'s work by incorporating a less stringent feasibility assumption without Slater's condition. In more recent work, Sinha and Vaze [2024] presented a simple COCO algorithm which uses a Euclidean projection step and achieves state-of-the-art performance guarantees. Sarkar et al. [2025] extended their work by considering a class of non-convex functions with a strengthened benchmark. Sarkar and Sinha [2025] also came up with an anytime

version of their policy by making innovative use of time-varying Lyapunov functions. They further derived an adaptive bound on the CCV.

**Projection-free OCO and COCO:**

For these algorithms to be practical, both performance and computational efficiency must be considered. It is well-established that the projection step commonly employed in OCO and COCO algorithms in every round is highly computationally expensive. This has motivated the development of the so-called *projection-free* algorithms that replace the expensive projection step with a linear optimization step.

One of the first works that studies this problem in the OCO setup is Hazan and Kale [2012]. Further contributions to the area of projection-free OCO have been made by Garber and Kretzu [2023], Garber [2021]. There also exist projection-free works that, instead of an LP step, assume some strong oracle, such as the access to a membership oracle in Garber and Kretzu [2022] and Mhammedi [2022]. While these works achieve strong regret guarantees, the overall algorithm still might be computationally expensive outside of specific use cases.

It is still an open question as to what the asymptotic lower bounds of Frank-Wolfe based projection-free methods are. Very recently, Weibel et al. [2025] provide strong numerical evidence that the lower bound on regret for these methods is $\Omega(T^{3/4})$. However, a formal proof for the same remains elusive.

Garber and Kretzu [2024] was the first to address the design of projection-free algorithms for general COCO. This paper also makes use of a drift-plus-penalty technique like other COCO works, except that it does not make use of a projection step. It uses the online Frank-Wolfe algorithm to iteratively move towards feasibility without explicit projection. It must be noted, however, that while the algorithm that they have proposed accesses the admissible set only via call to LP solvers, at one point, it has to do a convex optimization step over an Euclidean ball. While this is not the same as performing convex optimization over the admissible set, this is still a costly operation compared to linear optimization. They further propose a second algorithm where they perform online primal-dual (sub)gradient updates w.r.t. the dual-regularized Lagrangian functions. This algorithm does not make use of convex optimization, but the downside is that the regret and violation bounds do not hold for a subsequence of cost and constraint functions. Instead, they only hold when the entire sequence is considered at once. Both algorithms divide the horizon into disjoint blocks and only perform updates at the end of a block. They also extend their results into the bandit setting by employing the standard reduction using smoothed functions.

### A.1 Relation to Wang et al. [2025]

We dedicate this subsection to specifically situating our contributions with respect to concurrent work by Wang et al. [2025] regarding the OCG-based policy. At first glance, it would appear that our guarantees in the full-information and bandit settings are overlapping with theirs. However, we now argue, despite seemingly overlapping guarantees, our approach is fundamentally different from theirs and how we make a unique contribution to the literature.

At the outset, we must mention the similarity between our paper and Wang et al. [2025]. At a high level, both papers minimize regret and violation by constructing a special surrogate loss $\tilde{f}_t$ by additively combining the cost $f_t$ and constraint function $g_t$ in a proportion which depends on the violation $Q(t)$ till time $t$. After that, the surrogate loss is passed to the Online Conditional Gradient (OCG) algorithm. A major technical limitation of existing approaches to the analysis of the OCG algorithm is that they consider a uniform Lipchitz constant $G$ for all cost functions up to time $T$. However, in our setting, since $Q(t)$ at round $t$ is unknown, we cannot assign a Lipschitz constant to the surrogate cost $\tilde{f}_t$ at time $t$. This is the main technical challenge — to design an *adaptive* OCG policy where we don't need to know a uniform Lipchitz constant across all time steps. How we achieve this is a point of departure from Wang et al. [2025].

1. **Critique of the Doubling Trick Approach in Wang et al. (2025):** In the full information setting, Wang et al. [2025] makes use of a Franke-Wolfe type policy like us. However, to handle the unknown gradient norms required for setting the learning rate, they rely on a doubling trick. The doubling trick has not been viewed very favorably by the community. See the discussion in Zhang et al. [2024], Kwon and Mertikopoulos [2014], Luo and Schapire [2014] and Besson and Kaufmann [2018] for more details. These works have considered the doubling trick to be aesthetically inelegant, intuitively wasteful and practically suboptimal (as confirmed by our experiments) as it involves breaking up the time horizon into phases and then restarting the algorithm

from scratch at the beginning of every phase.

It is to be noted that while the typical use of the doubling trick is in estimating the length of the time horizon, which is a priori unknown, it is a general technique and can be used even when some other quantity is unknown. In Wang et al. [2025] it was used because the value of $\tilde{G}$, the Lipchitz constant which upper-bounds the gradient of all surrogate losses across timesteps was unknown. To do so, they estimated a value of $\tilde{G}$ and ran the algorithm till it was valid, once it was not, they doubled it and started the algorithm from scratch which means that it loses all the accumulated knowledge upto that point. This is intuitively wasteful because the foundational idea behind OCO is to improve our predictions by accumulating knowledge. By periodically throwing away the accumulated information, we are going against this intuition. This type of restarting technique, the doubling trick, is often considered a last resort and brute-force approach when all other adaptive strategies have failed, which is why it is also considered aesthetically inelegant, as its use provides limited inspiration to the community. In our paper, we come up with a more elegant adaptive strategy that removes the need to use the doubling trick. This makes our analysis both more novel and more impactful in the context of the online learning community. As we show in our experiments (see section F), the doubling trick is also practically suboptimal, underperforming our algorithm despite having similar theoretical guarantees.

2. **Our Intrinsically Adaptive Method:** We sidestep the use of the doubling trick by adapting the Frank-Wolfe algorithm to our setting in a more elegant manner. We too face the technical challenge of adapting the learning rate to the unknown and time-varying Lipschitz constants of the surrogate cost functions. In contrast to restarting, our Algorithm 1 introduces a novel, intrinsically adaptive version of the Online Conditional Gradient (OCG) method. This algorithm adjusts its learning rate at every single step based on the most recently observed gradient norm. This approach avoids the disruptive restarts of the doubling trick, resulting in a more stable algorithm. Furthermore, in developing this solution, we derive the first known scale-free adaptive regret bounds for the OCG policy, a theoretical result of independent interest to the online learning community.

   In summary, while Wang et al. [2025] divides the time-horizon into chunks and then runs the usual OCG in those chunks, our method differs more fundamentally from the usual OCG, as the Lipchitz-constants are taken so that they can be time-varying at every timestep. We don't reduce our *adaptive* OCG into multiple sequential OCGs started from scratch and instead consider one contiguous OCG-like algorithm with time-varying Lipchitz constants. It takes considerable technical novelty on our part to derive scale-free regret bounds for this adaptive algorithm, which we then use in our analysis.

   Next, we talk about the bandit section in Wang et al. [2025] and how the analysis is flawed and the results don't actually hold up. Thus, our guarantees in the bandit setting are the first of their kind.

3. **The erroneous assumption of non-stochasticity of $Q(t)$ in the bandit analysis of Wang et al. (2025):** It is well-known that in the bandit convex optimization setting, since we need to add a random perturbation to our actions, we do not bound our regret but the expected regret as the regret becomes as inherently stochastic quantity in that case. Due to similar reasoning $Q(t)$ is also an inherently stochastic quantity. Indeed, both Garber and Kretzu [2024] and our paper bound the expectation of $Q(t)$ instead of $Q(t)$ for this very reason. On the other hand, Wang et al. [2025] proceeds with their bandit analysis as if $Q(t)$ were a non-stochastic quantity. We are not being pedantic here — this is a fundamental flaw and renders their analysis invalid as the analysis would be non-trivially different if they were to deal with $\mathbb{E}[Q(t)]$ in their analysis.

4. **On the form of the surrogate loss and its unique significance in the bandit setting:** Notwithstanding the above point, there are other, more subtle flaws in the bandit analysis of Wang et al. [2025]. In that paper, they effectively reused the form of the surrogate loss used in the full-information setting in the bandit setting as well. However, as we point out in our bandit section (section 4), this is not acceptable, and we instead have to come up with a new surrogate loss $\tilde{f}_t$ that depends on $Q(t-1)$ instead of $Q(t)$. Put briefly, this is because making $Q(t)$ part of the loss implies that we are making the function itself dependent on the random action which is problematic for unbiased gradient estimation, as the only source of randomness during gradient estimation should be the random perturbation introduced at time $t$ and the function itself should be deterministic given all the randomness of previous rounds. Thus, under the surrogate loss assumed in Wang et al. [2025], bandit learning is impossible. These minutiae are important and must not be abstracted out in a superficial analysis. This is a more subtle point and easier to miss, and thus, this is also an independent contribution of ours to the field of constrained BCO.

# B Proofs for Projection-Free COCO

## B.1 Proof of Theorem 2.1

Our proof combines the arguments in Hazan [2022, Theorem 23] with the adaptive regret bound of the FTRL policy with a time-varying Euclidean regularizer [Orabona, 2019, Corollary 7.9]. For reference, the FTRL policy with a time-varying Euclidean regularizer is described in Algorithm 5. Since Hazan [2022, Theorem 23] considers time-invariant non-adaptive regularizers, non-trivial extensions are required to derive an adaptive regret bound for the OCG policy. We first consider the following fictitious action sequence:

$$x_t^\star = \arg\min_{x\in\mathcal{X}} F_{t-1}(x), \quad t \geqslant 1, \tag{17}$$

where the regularized cumulative loss function $F_t(\cdot)$ is defined in Eqn. (2). Recall that $\{x_t\}_{t\geqslant 1}$ is the action sequence generated by the OCG algorithm. On the other hand, it is easy to see that the sequence $\{x_t^\star\}_{t\geqslant 1}$ corresponds to the iterates of the Follow-the-Regularized-Leader (FTRL) algorithm [Hazan, 2022, Algorithm 13] when run with the auxiliary cost function sequence $\{\tilde{f}_t\}_{t\geqslant 1}$, where the function $\tilde{f}_t$ is a shifted version of the original cost function $\hat{f}_t$:

$$\tilde{f}_t(x) := \hat{f}_t(x + x_t - x_t^\star), \quad t \geqslant 1. \tag{18}$$

This can be verified by referring to the FTRL algorithm, described in Algorithm 5, along with the fact that $\nabla\tilde{f}_t(x_t^\star) = \nabla\hat{f}_t(x_t) \equiv \nabla_t$. Clearly, the function $\tilde{f}_t$ is also $L_t$-Lipschitz. We emphasize that the action sequence $\{x_t^\star\}_{t\geqslant 1}$ is not necessarily played by the algorithm and is introduced here purely for the sake of analysis. The regret with respect to any fixed benchmark action $x^\star \in \mathcal{X}$ can be decomposed into the sum of two terms, $(A)$ and $(B)$, as described below:

$$\begin{aligned}
\text{Regret}_T(\text{OCG}) &= \sum_{t=1}^{T}\left(\hat{f}_t(x_t) - \hat{f}_t(x^\star)\right) = \sum_{t=1}^{T}\left(\hat{f}_t(x_t) - \hat{f}_t(x_t^\star)\right) + \sum_{t=1}^{T}\left(\hat{f}_t(x_t^\star) - \hat{f}_t(x^\star)\right) \\
&\overset{(a)}{\leqslant} \underbrace{\sum_{t=1}^{T} L_t||x_t - x_t^\star||}_{(A)} + \underbrace{\sum_{t=1}^{T}\left(\hat{f}_t(x_t^\star) - \hat{f}_t(x^\star)\right)}_{(B)},
\end{aligned} \tag{19}$$

where inequality $(a)$ follows from the Lipschitz continuity of the functions $\{\hat{f}_t, t \geqslant 1\}$. We now bound the above two terms separately.

**Bounding term $(A)$:** Using triangle inequality, we can further upper-bound term $(A)$ with the sum of two terms, $(C)$ and $(D)$, as follows:

$$\underbrace{\sum_{t=1}^{T} L_t||x_t - x_t^\star||}_{(A)} \leqslant \underbrace{\sum_{t=1}^{T} L_t||x_t - x_{t+1}^\star||}_{(C)} + \underbrace{\sum_{t=1}^{T} L_t||x_{t+1}^\star - x_t^\star||}_{(D)}. \tag{20}$$

The following two lemmas give upper bounds to term $(C)$ and term $(D)$, respectively.

**Lemma B.1.** *Term (C) in Eqn. (20) can be upper bounded as*

$$(C) \equiv \sum_{t=1}^{T} L_t||x_t - x_{t+1}^\star|| \leqslant 12DL_T T^{3/4}\sqrt{\log(TeL_T/L_1)}).$$

See Appendix B.3 for the proof of Lemma B.1.

**Lemma B.2.** *Term (D) in Eqn. (20) can be upper bounded as*

$$(D) \equiv \sum_{t=1}^{T} L_t||x_{t+1}^\star - x_t^\star|| \leqslant 3DL_T\sqrt{T}.$$

The proof of the above result follows from the stability property of the FTRL policy. See Appendix B.4 for the proof of Lemma B.2. Lemma (B.1) and (B.2), taken together, upper bounds Term $(A)$ in Eqn. (19).

**Bounding term (B):** The $t^{\text{th}}$ term inside the summation of term (B) in Eqn. (19) can be upper bounded as follows:

$$\hat{f}_t(x_t^\star) - \hat{f}_t(x^\star) \overset{(c)}{=} \tilde{f}_t(x_t^\star - x_t + x_t^\star) - \tilde{f}_t(x^\star - x_t + x_t^\star) \overset{(d)}{\leqslant} \tilde{f}_t(x_t^\star) - \tilde{f}_t(x^\star) + 2L_t||x_t^\star - x_t||,$$

where (c) follows from the Definition (18) and (d) follows from the fact that the auxiliary cost function $\tilde{f}_t$ is $L_t$-Lipschitz. Hence, term $(B)$ can be upper bounded as:

$$(B) \leqslant \text{Regret}_T(\text{FTRL}) + 2(A), \tag{21}$$

where $\text{Regret}_T(\text{FTRL}) \equiv \sum_{t=1}^T \left(\tilde{f}_t(x_t^\star) - \tilde{f}_t(x^\star)\right)$ is the regret of the FTRL policy for the auxiliary cost function sequence $\{\tilde{f}_t\}_{t=1}^T$ with adaptive learning rates. The pseudocode of the FTRL policy is given in Algorithm 5 for reference [Orabona, 2019]. The following result gives an upper bound to the FTRL policy with adaptive learning

**Algorithm 5** Follow-the-Regularized-Leader

1: **Inputs:** Convex decision set $\mathcal{X}$, sequence of convex cost functions $\{l_t : \mathcal{X} \to \mathbb{R}\}_{t\geqslant 1}$, $\text{diam}(\mathcal{X}) = D$.
2: **Parameters:** Closed and 2-strongly convex function (w.r.t. the Euclidean norm) $\psi(x) \equiv ||x - x_1||^2$, adaptive regularizer sequence $\psi_t(x) = \frac{\psi(x)}{\eta_{t-1}}$, where $\eta_{t+1} \leqslant \eta_t, \forall t \geqslant 1$.
3: Initialize $x_1 \in \mathcal{X}$ arbitrarily
4: **for** $t = 1 : T$ **do**
5: Play $x_t$, observe $l_t(\cdot)$, and compute $\nabla_t \equiv \nabla l_t(x_t)$.
6: Compute

$$x_{t+1} = \arg\min_{x\in\mathcal{X}} \sum_{\tau=1}^{t} \nabla_\tau^\top x + \frac{||x - x_1||^2}{\eta_t}.$$

7: **end for**

rates.

**Theorem B.3.** *(Orabona [2019, Corollary 7.9]) The FTRL policy, described in Algorithm 5, achieves the following regret bound for any sequence of convex cost functions $\{l_t\}_{t\geqslant 1}$ and for any feasible action $u \in \mathcal{X}$ :*

$$\sum_{t=1}^T l_t(x_t) - \sum_{t=1}^T l_t(u) \leqslant \frac{D^2}{\eta_T} + \frac{1}{4}\sum_{t=1}^T \eta_{t-1}||\nabla_t||^2.$$

Setting $l_t = \tilde{f}_t$, we have the following regret bound in our case:

$$\begin{aligned}\text{Regret}_T(\text{FTRL}) &\leqslant \frac{D^2}{\eta_T} + \frac{1}{4}\sum_{t=1}^T \eta_{t-1}L_t^2 \leqslant 2DL_TT^{\frac{3}{4}} + \frac{D}{8T^{3/4}}\sum_{t=1}^T \frac{L_t^2}{L_{t-1}} \\ &\leqslant \quad 2DL_TT^{\frac{3}{4}} + \frac{D}{8T^{3/4}}\big(\max_{1\leqslant t\leqslant T}\frac{L_t}{L_{t-1}}\big)\sum_{t=1}^T L_t \leqslant 2DL_TT^{\frac{3}{4}} + \frac{DL_TT^{1/4}}{8}\big(\max_{1\leqslant t\leqslant T}\frac{L_t}{L_{t-1}}\big).\end{aligned} \tag{22}$$

The final result now follows upon combining Eqn. (19) with Lemma B.1, Eqn. (21), and the FTRL regret bound (22), yielding

$$\text{Regret}_T(\text{OCG}) \leqslant 3(A) + \text{Regret}_T(\text{FTRL}) \leqslant 47DL_TT^{3/4}\sqrt{\log(TeL_T/L_1)})\left(1 + \frac{1}{376\sqrt{T}}\big(\max_{1\leqslant t\leqslant T}\frac{L_t}{L_{t-1}}\big)\right).$$

### B.2 Supporting Lemmata

In the analysis of Section 3.1, we state the following two supporting Lemmas, which were used in the argument there.

**Lemma B.4.** *Choosing $\Phi(x) = x^p$ with $p = \log T$ and for any $V > 0$, we have*

$$\max_{1\leqslant t\leqslant T}\frac{L_t}{L_{t-1}} \leqslant 1 + e.$$

See Section B.5 for the proof of this Lemma.

**Lemma B.5.** *Choosing $\Phi(x) = x^p$ with $p = \log T$ and for any $V > 0$, we have*

$$\sqrt{\log \frac{L_T}{L_1}} \leqslant \log T.$$

See Section B.6 for the proof.

### B.3 Proof of Lemma B.1

Recall the definition of the regularized cumulative cost function from Eqn. (2):

$$F_t(x) = \sum_{\tau=1}^{t} \nabla_\tau^T x + \frac{||x - x_1||^2}{\eta_t}, \tag{23}$$

with the adaptive learning rate sequence $\{\eta_t\}_{t\geqslant 1}$ given in Algorithm 1. Clearly, the quadratic function $F_t$ is $\frac{2}{\eta_t}$-smooth and $\frac{2}{\eta_t}$-strongly convex w.r.t. the standard Euclidean norm. Similar to the proof in [Hazan, 2022, Theorem 23], we define

$$h_t(x) \overset{\text{(def)}}{=} F_t(x) - F_t(x_{t+1}^\star), \quad h_t \overset{\text{(def)}}{=} h_t(x_t),$$

where $x_{t+1}^\star \equiv \arg\min_{x\in\mathcal{X}} F_t(x)$, as defined in Eqn. (17). Now recall that the OCG policy chooses the next action as $x_{t+1} = (1-\sigma_t)x_t + \sigma_t v_t$. Hence,

$$\begin{aligned}
h_t(x_{t+1}) &= F_t(x_{t+1}) - F_t(x_{t+1}^\star) \\
&= F_t(x_t + \sigma_t(v_t - x_t)) - F_t(x_{t+1}^\star) \\
&\overset{(a)}{\leqslant} F_t(x_t) - F_t(x_{t+1}^\star) + \sigma_t\langle \nabla F_t(x_t), v_t - x_t\rangle + \frac{\sigma_t^2||v_t - x_t||^2}{\eta_t} \\
&\overset{(b)}{\leqslant} F_t(x_t) - F_t(x_{t+1}^\star) + \sigma_t\langle \nabla F_t(x_t), x_{t+1}^\star - x_t\rangle + \frac{\sigma_t^2 D^2}{\eta_t} \\
&\overset{(c)}{\leqslant} (1-\sigma_t)\big(F_t(x_t) - F_t(x_{t+1}^\star)\big) + \frac{\sigma_t^2 D^2}{\eta_t},
\end{aligned}$$

where (a) uses the smoothness property of $F_t$, (b) follows from the definition of $v_t$ as given in Eqn. (3), and (c) follows from the convexity of $F_t$. Thus, the above yields

$$h_t(x_{t+1}) \leqslant (1-\sigma_t)h_t + \frac{\sigma_t^2 D^2}{\eta_t}. \tag{24}$$

Furthermore, we have

$$\begin{aligned}
h_{t+1} &= F_{t+1}(x_{t+1}) - F_{t+1}(x_{t+2}^\star) \\
&= F_t(x_{t+1}) + \nabla_{t+1}^T x_{t+1} + ||x_{t+1} - x_1||^2\big(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\big) - F_t(x_{t+2}^\star) - \nabla_{t+1}^T x_{t+2}^\star - ||x_{t+2}^\star - x_1||^2\big(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\big) \\
&\leqslant h_t(x_{t+1}) + L_{t+1}||x_{t+1} - x_{t+2}^\star|| + D^2\big(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\big),
\end{aligned} \tag{25}$$

where in the last step, we have used Cauchy-Schwarz inequality along with the fact that $||\nabla_t|| \leqslant L_t$. Since, $F_{t+1}$ is $\frac{2}{\eta_{t+1}}$-strongly convex, and $x_{t+2}^\star \equiv \arg\min_{x\in\mathcal{X}} F_{t+1}(x)$, we have

$$h_{t+1} = h_{t+1}(x_{t+1}) = F_{t+1}(x_{t+1}) - F_{t+1}(x_{t+2}^\star) \geqslant \frac{1}{\eta_{t+1}}||x_{t+1} - x_{t+2}^\star||^2.$$

This implies that

$$||x_{t+1} - x_{t+2}^\star|| \leqslant \sqrt{\eta_{t+1}h_{t+1}}. \tag{26}$$

By substituting this bound into Eqn. (25), we conclude that

$$h_{t+1} \leqslant h_t(x_{t+1}) + L_{t+1}\sqrt{\eta_{t+1}h_{t+1}} + D^2\big(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\big).$$

Substituting the upper bound from Eqn. (24) into the above, we obtain

$$h_{t+1} \leqslant (1-\sigma_t)h_t + \frac{\sigma_t^2 D^2}{\eta_t} + L_{t+1}\sqrt{\eta_{t+1}h_{t+1}} + D^2\big(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\big).$$

Multiplying both sides of the above equation by $\eta_{t+1}$, we obtain

$$\begin{aligned}\eta_{t+1}h_{t+1} &\leqslant (1-\sigma_t)\eta_{t+1}h_t + \eta_{t+1}\frac{\sigma_t^2 D^2}{\eta_t} + \eta_{t+1}L_{t+1}\sqrt{\eta_{t+1}h_{t+1}} + D^2(1-\frac{\eta_{t+1}}{\eta_t}) \\ &\leqslant (1-\sigma_t)\eta_t h_t + \sigma_t^2 D^2 + \eta_{t+1}L_{t+1}\sqrt{\eta_{t+1}h_{t+1}} + D^2(1-\frac{\eta_{t+1}}{\eta_t}),\end{aligned}$$

where in the last step, we have used the fact that $\eta_t \geqslant \eta_{t+1}, \forall t \geqslant 1$. Let us now define the sequence $u_t \equiv \eta_t h_t, t \geqslant 1$. From the above, we have

$$u_{t+1} \leqslant (1-\sigma_t)u_t + \sigma_t^2 D^2 + \underbrace{\eta_{t+1}L_{t+1}\sqrt{u_{t+1}}}_{\text{(c)}} + \underbrace{D^2(1-\frac{\eta_{t+1}}{\eta_t})}_{\text{(d)}}. \tag{27}$$

Now, term (c) can be upper bounded as follows:

$$\begin{aligned}\eta_{t+1}L_{t+1}\sqrt{u_{t+1}} &= (\sqrt{D}\eta_{t+1}L_{t+1})^{2/3}\left(\frac{\eta_{t+1}L_{t+1}}{D}\right)^{1/3}\sqrt{u_{t+1}} \\ &\overset{(e)}{\leqslant} \frac{1}{2}(\sqrt{D}\eta_{t+1}L_{t+1})^{4/3} + \frac{1}{2}\left(\frac{\eta_{t+1}L_{t+1}}{D}\right)^{2/3}u_{t+1} \\ &\overset{(f)}{\leqslant} \frac{1}{4}D^2\sigma_t^2 + \frac{1}{3}\sigma_t u_{t+1},\end{aligned}$$

where inequality $(e)$ follows from the AM-GM inequality and $(f)$ follows from the choice of the step size sequence $\eta_t = \frac{D}{2L_t T^{\frac{3}{4}}}$, and the fact that $\sigma_t \geqslant \frac{1}{\sqrt{t}}$. To bound term (d) from Eqn. (27), observe that,

$$D^2(1-\frac{\eta_{t+1}}{\eta_t}) = D^2\big(1-\frac{L_t}{L_{t+1}}\big).$$

Substituting the above bounds into Eqn. (27), we have

$$u_{t+1} \leqslant (1-\sigma_t)u_t + \frac{5}{4}D^2\sigma_t^2 + \frac{1}{3}\sigma_t u_{t+1} + D^2(1-\frac{L_t}{L_{t+1}})$$

This implies that

$$u_{t+1}\big(1-\frac{1}{3}\sigma_t\big) \leqslant (1-\sigma_t)u_t + \frac{5}{4}D^2\sigma_t^2 + D^2(1-\frac{L_t}{L_{t+1}}),$$

*i.e.,*

$$u_{t+1} \leqslant \big(1-\frac{2}{3}\sigma_t\big)u_t + \frac{15}{8}D^2\sigma_t^2 + \frac{3}{2}D^2\big(1-\frac{L_t}{L_{t+1}}\big), \tag{28}$$

where we have used the fact that $\sigma_t \leqslant 1$ and $\frac{1-\sigma_t}{1-\frac{1}{3}\sigma_t} \leqslant 1 - \frac{2}{3}\sigma_t$. Using the fact that the sequence $\{\sigma_t\}_{t\geqslant 1}$ is non-increasing and $u_{T+1} \geqslant 0$, summing up the inequalities in (28) in the range $t = 1$ to $t = T-1$, we conclude that

$$\frac{2}{3}\sigma_T\sum_{t=1}^{T}u_t \leqslant u_1 + \frac{15}{8}D^2\sum_{t=1}^{T-1}\sigma_t^2 + \frac{3}{2}D^2\sum_{t=1}^{T-1}\frac{L_{t+1}-L_t}{L_{t+1}}$$

$$\begin{aligned} &\leqslant \quad u_1 + \frac{15}{2}D^2 \sum_{t=1}^{T-1} \frac{1}{t} + \frac{3}{2}D^2 \int_{L_1}^{L_T} \frac{dx}{x} \\ &\leqslant \quad u_1 + \frac{15}{2}D^2(1+\ln T) + \frac{3}{2}D^2 \log \frac{L_T}{L_1}. \end{aligned}$$

Since $\sigma_T \geqslant \frac{1}{\sqrt{T}}$, the above bound implies that

$$\sum_{t=1}^{T} u_t \leqslant \frac{3\sqrt{T}}{2}u_1 + \frac{45D^2\sqrt{T}}{4}(1+\ln T) + \frac{9D^2\sqrt{T}}{4}\log\frac{L_T}{L_1}.$$

It can be easily verified that $u_1 = \eta_1 h_1 \leqslant D^2/2$. Finally, we have

$$\sum_{t=1}^{T} L_t ||x_t - x_{t+1}^\star|| \overset{(a)}{\leqslant} L_T \sum_{t=1}^{T} \sqrt{u_t} \overset{(b)}{\leqslant} L_T\sqrt{T}\sqrt{\sum_{t=1}^{T} u_t} \leqslant 12 L_T D T^{3/4}\sqrt{\log(TeL_T/L_1)}),$$

where inequality (a) follows upon combining the monotonicity of the Lipschitz constants, the bound from Eqn. (26) and the definition of the sequence $\{u_t\}_{t\geqslant 1}$ and (b) follows from Cauchy-Schwarz inequality.

### B.4 Proof of Lemma B.2

Since $F_t$ is $\frac{2}{\eta_t}$-strongly-convex and $x_{t+1}^\star \in \arg\min_{x\in\mathcal{X}} F_t(x)$, we have

$$\|x_t^\star - x_{t+1}^\star\|^2/\eta_t \leqslant F_t(x_t^\star) - F_t(x_{t+1}^\star) \leqslant \underbrace{F_{t-1}(x_t^\star) - F_{t-1}(x_{t+1}^\star)}_{\leqslant 0} + \nabla_t^\top(x_t^\star - x_{t+1}^\star) + \|x_t^\star - x_1^\star\|^2(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}),$$

where we have used the fact that $\{\eta_t\}_{t\geqslant 1}$ is non-increasing. Hence, using Cauchy-Schwarz inequality, and using the fact that $\eta_t = \frac{D}{2L_t T^{\frac{3}{4}}}, t \geqslant 1$, we have

$$\begin{aligned} &\|x_t^\star - x_{t+1}^\star\|^2 \leqslant \eta_t \|\nabla_t\| \|x_t^\star - x_{t+1}^\star\| + D^2(1 - \frac{L_{t-1}}{L_t}) \\ \implies \quad &\|x_t^\star - x_{t+1}^\star\| \leqslant 2\left(\eta_t\|\nabla_t\| + D\sqrt{1 - \frac{L_{t-1}}{L_t}}\right). \end{aligned}$$

Define $\xi \equiv \max_{1\leqslant t\leqslant T} \frac{L_t}{L_{t-1}}$. We can bound the summation as follows

$$\sum_{t=1}^{T} L_t\sqrt{1 - \frac{L_{t-1}}{L_t}} = \sum_{t=1}^{T}\sqrt{L_t(L_t - L_{t-1})} \leqslant \sqrt{L_T}\sum_{t=1}^{T}\sqrt{L_t - L_{t-1}} \leqslant L_T\sqrt{T},$$

where the last step follows from an application of the Cauchy-Schwarz inequality. Hence,

$$\sum_{t=1}^{T} L_t ||x_t^\star - x_{t+1}^\star|| \leqslant D L_T T^{1/4} + 2DL_T\sqrt{T} \leqslant 3DL_T\sqrt{T}. \tag{29}$$

### B.5 Proof of Lemma B.4

Recall that the sequence $\{Q(t)\}_{t\geqslant 1}$ is monotone non-deceasing with $Q(0) = GD\log T$. Furthermore, since $g_t(x^\star) \leqslant 0$, for any $t$, from the Lipschitz continuity of the constraint function $g_t$, we have $g_t(x_t) \leqslant g_t(x^\star) + G||x_t - x^\star|| \leqslant GD$. This shows that $Q(t) - Q(t-1) \leqslant GD$. From the expression of the Lipschitz constant of the surrogate cost functions, given by Eqn. (9), we have:

$$L_t = G\big(V + pQ^{p-1}(t)\big), \quad \text{and} \quad L_{t-1} = G\big(V + pQ^{p-1}(t-1)\big).$$

Hence, the ratio of the successive Lipschitz constants can be bounded as:

$$\frac{L_t}{L_{t-1}} = \frac{V + pQ^{p-1}(t)}{V + pQ^{p-1}(t-1)}$$

$$
\begin{aligned}
&= 1 + \frac{p(Q^{p-1}(t) - Q^{p-1}(t-1))}{V + pQ^{p-1}(t-1)} \\
&\overset{(a)}{\leqslant} 1 + GDp(p-1)\frac{Q^{p-2}(t)}{V + pQ^{p-1}(t-1)} \\
&\overset{(b)}{\leqslant} 1 + GDp\frac{\big(Q(t-1) + GD\big)^{p-2}}{Q^{p-1}(t-1)} \\
&= 1 + \frac{GDp}{Q(t-1)}\left(1 + \frac{GD}{Q(t-1)}\right)^{p-2} \\
&\overset{(c)}{\leqslant} 1 + \left(1 + \frac{1}{p}\right)^{p-2} \\
&\overset{(d)}{\leqslant} 1 + e^{1-\frac{2}{p}} \\
&\leqslant 1 + e
\end{aligned}
$$

where (a) follows from the convexity of the mapping $x \mapsto x^{p-1}$, which yields

$$Q^{p-1}(t-1) \geqslant Q^{p-1}(t) + (p-1)Q^{p-2}(t)(Q(t-1) - Q(t)),$$

*i.e.,*

$$Q^{p-1}(t) - Q^{p-1}(t-1) \leqslant (p-1)Q^{p-2}(t)(Q(t) - Q(t-1)) \leqslant (p-1)GDQ^{p-2}(t),$$

where we have used the fact that $Q(t) - Q(t-1) \leqslant GD$. Inequality (b) uses the previous bound and the fact that $V$ is non-negative. Inequality (c) follows from the fact that $Q(t-1) \geqslant Q(0) = GD\log T = GDp$. Finally, inequality (d) follows from the fact that $e^t \geqslant 1 + t, \forall t \in \mathbb{R}$. This shows that for any $T \geqslant 1$, we have

$$\max_{1 \leqslant t \leqslant T} \frac{L_t}{L_{t-1}} \leqslant 1 + e.$$

### B.6 Proof of Lemma B.5

Recall that the sequence $\{Q(t)\}_{t \geqslant 1}$ is monotone non-deceasing with $Q(0) = GD\log T$. Furthermore, the Lipschitz constants of the surrogate costs functions depend on the CCV as

$$L_T = G\big(V + pQ^{p-1}(T)\big), \quad \text{and} \quad L_1 = G\big(V + pQ^{p-1}(1)\big),$$

where $p = \log T$. Hence,

$$\frac{L_T}{L_1} = \frac{V + pQ^{p-1}(T)}{V + pQ^{p-1}(1)} \overset{(a)}{\leqslant} \frac{Q^{p-1}(T)}{Q^{p-1}(1)} \overset{(b)}{\leqslant} \frac{(GDT + GD\log T)^{p-1}}{(GD\log T)^{p-1}} \overset{(c)}{\leqslant} T^p.$$

where inequality (a) follows from the fact that $Q(T) \geqslant Q(1), V > 0$ and the map $x \mapsto \frac{x+\alpha}{x+\beta}$ is non-increasing for $\alpha \geqslant \beta$, (b) uses the fact that $Q(t) - Q(t-1) \leqslant GD, t \geqslant 1$, and $Q(0) = GD\log T$, and finally, (c) holds for $T \geqslant 2$. Hence, we have

$$\log\frac{L_T}{L_1} \leqslant p(\log T) \implies \sqrt{\log\frac{L_T}{L_1}} \leqslant \log T,$$

## C The Bandit Feedback Setting

### C.1 Adaptive Blocked Bandit Conditional Gradient algorithm

Hazan [2022] presents a simple reduction from the bandit feedback setting to the full information setting for first-order algorithms. The reduction guarantees the same regret bounds as the original first-order algorithm up to the magnitude of the estimated gradients. In other words, as long as we can reasonably bound the variance of the gradient estimator, we should expect the same bounds as the analogous full-information algorithm. It

turns out that while the reduction applies to the OCG algorithm by virtue of it being a first-order algorithm, the resulting bounds can be further improved by modifying the algorithm to control the variance of the gradient estimators better.

To achieve that end, we employ a standard tool known as the *blocking* technique Hazan and Minasyan [2020], Garber and Kretzu [2019, 2024], Merhav et al. [2002]. This is the main departure from section 2.1, apart from the obvious transition from full information to a bandit setting. The blocking technique is a general technique where the horizon is partitioned into non-overlapping blocks of length $K$. Instead of playing a new action at every timestep, the policy plays the same action throughout each block $m$ while switching to a new action at the start of block $m+1$.

*Remark* C.1. For the sake of analysis, it would be useful to remember that this technique is essentially the grouping of several iterations of the sequential game into one. This observation would enable a reduction from a blocking algorithm to a non-blocking algorithm.

The purpose of the blocking technique is variance-reduction and it is closely related with another technique known in literature as the averaging technique Mokhtari et al. [2017], Chen et al. [2018]. Perhaps the latter better alludes to the intuition behind this technique. As the gradients corresponding to the functions in the block are accumulated, the corresponding variance of the sum of their estimators only scales linearly with $K$ instead of $K^2$. This argument is formalized in Lemma C.8.

We first introduce the *smoothed* version of the cost function whose value at a point $x$ is the same as the expected value of the original cost function over all points in the $\delta$-neighbourhood of the point $x$. These play a crucial role in the analysis. The reason this becomes important is because we do not play a constant action throughout a block $m$ but a $\delta$-perturbation of it. This is essential as it is a well-known fact that a bandit algorithm has to be randomised. So the value of the smoothed function at the point $x$ is a better representative of the expected cost than the value of the actual cost function would be. Note that these functions don't appear in the algorithm and are only defined for the sake of analysis. Formally,

$$\hat{f}_\delta(\mathbf{x}) = \mathbb{E}_{\mathbf{u}\sim\mathcal{B}^n}\left[f(\mathbf{x}+\delta\mathbf{u})\right] \tag{30}$$

$$\nabla\hat{f}_\delta(\mathbf{x}) = \mathbb{E}_{\mathbf{u}\sim\mathcal{B}^n}\left[\frac{n}{\delta}f(\mathbf{x}+\delta\mathbf{u})\mathbf{u}\right] \tag{31}$$

Also, as in Flaxman et al. [2004] we make the standard assumption that the feasible set $\mathcal{X}$ is full dimensional, contains the origin, and that there exists scalars $r, R > 0$ such that $r\mathcal{B}^n \subseteq \mathcal{X} \subseteq R\mathcal{B}^n$, where $\mathcal{B}^n$ denotes the unit Euclidean ball centered at the origin in $\mathbb{R}^n$.

We denote by $\mathcal{S}^n$ the unit sphere in $\mathbb{R}^n$, and we write $u \sim S^n$ and $u \sim \mathcal{B}^n$ to denote a random vector $u$ sampled uniformly from $\mathcal{S}^n$ and $\mathcal{B}^n$, respectively. We denote by $\|\mathbf{x}\|$ the $\ell_2$ norm of the vector $\mathbf{x}$.

For a compact and convex set $\mathcal{X} \subset \mathbb{R}^n$, which satisfies the above assumptions (i.e., $r\mathcal{B}^n \subseteq \mathcal{X} \subseteq R\mathcal{B}^n$), and a scalar $0 < \delta \leqslant r$, we define the set $\mathcal{X}_\delta := (1-\delta/r)\mathcal{X} = \{(1-\delta/r)x \mid x \in \mathcal{X}\}$. In particular, it holds that $\mathcal{X}_\delta \subseteq \mathcal{X}$ and for all $x \in \mathcal{X}_\delta$, $x + \delta\mathcal{B}^n \subseteq \mathcal{X}$ (see Hazan [2019]). This is important as we want to make sure that a $\delta$-perturbation around $x_m$, that is, the constant expected action in a block, does not land us outside $\mathcal{X}$. This means that $x_m$ should belong to $\mathcal{X}_\delta$ and that the optimization in Algorithm 6 is over $\mathcal{X}_\delta$ as opposed to $\mathcal{X}$.

Finally, note that while the conditional gradient method is run over the shrunk set $\mathcal{X}_\delta$, solving the linear optimization problem over $\mathcal{X}_\delta$ is identical, up to scaling, to solving it over the original set $\mathcal{X}$.

*Remark* C.2. In line with Hazan [2019] much of the analysis of a BCO algorithm can be reduced to that of an OCO algorithm by simply replacing the cost and constraint functions by their $\delta-$smoothed versions.

### C.2 Conditional Gradient with Stopping Condition

The above Algorithm 6 serves as a subroutine to Algorithm 4 which is our adaptive bandit algorithm.

The algorithm tries to find the approximate minimizer of the function $F$ by using only LOO calls. The process repeats until the Frank-Wolfe gap, defined as $g_\tau = \nabla F_m(z_\tau)^\top(z_\tau - v_\tau)$, falls below a specified tolerance threshold, $\epsilon$. The gap $g_\tau$ provides a computable upper bound on the suboptimality of the current iterate, i.e., $F_m(z_\tau) - F_m(x^\star) \leqslant g_\tau$, where $x^\star$ is the true optimum.

**Algorithm 6** Conditional Gradient with Stopping Condition

**Inputs:** Convex decision set $\mathcal{X}_\delta$, initial vector $x_{\text{in}}$, objective function $F_m(x)$.
**Parameters:** error tolerance $\epsilon$.
Initialize $z_1 \leftarrow x_{in}$ and $\tau \leftarrow 0$
**repeat**
  $\tau \leftarrow \tau + 1$
  $v_\tau \in \underset{x \in \mathcal{X}_\delta}{\text{argmin}}\{\nabla F_m(z_\tau)^\top x\}$
  $\sigma_\tau = \underset{\sigma \in [0,1]}{\text{argmin}}\{F_m(z_\tau + \sigma(v_\tau - z_\tau))\}$
  $z_{\tau+1} = z_\tau + \sigma_\tau(v_\tau - z_\tau)$
**until** $\nabla F_m(z_\tau)^\top (z_\tau - v_\tau) < \frac{\epsilon}{\eta_m}$
$x_{\text{out}} \leftarrow z_\tau$

### C.3 Proof of Theorem 4.1

Define $x^\star \in \underset{x \in \mathcal{X}}{\text{argmin}} \sum_{t=1}^T f_t(x)$, $\tilde{x}^\star = (1 - \delta/r)x^\star$, $\forall t : m(t) := \lceil \frac{t}{K} \rceil$. Recall that throughout any block $m$, Algorithm 3 predicts according to $x_m$.

It is clear from Algorithm 3 that there is a lot of randomness involved which if separated would considerably ease our analysis. Note that what we are about to do next can be considered an implicit reduction from the bandit to full information setting.

To that end we define a proxy regret -

$$\text{Regret}_T(\text{blocked-OCG}) = \sum_{t=1}^{T} \left( \mathbb{E}[\hat{f}_{t,\delta}(x_{m(t)})] - \hat{f}_{t,\delta}(\tilde{x}^\star) \right) \tag{32}$$

The following lemma relates the true expected regret with the regret defined in Eqn. (32).

**Lemma C.3.** *Under Algorithm 3, the true expected regret can be upper bounded as*

$$\mathbb{E}[\textit{Regret}_T(\textit{Algorithm 3})] \leqslant \quad 3L_T\delta T + \tfrac{\delta R L_T T}{r} + \textit{Regret}_T(\textit{blocked-OCG})$$

The proof of Lemma C.3 is given in Appendix C.5.

**Bounding Regret$_T$(blocked-OCG) :**

**Lemma C.4.** *Under Algorithm 3, Regret$_T$(blocked-OCG) can be upper bounded as*

$$\frac{4R^2}{\eta_{T/K}} + \frac{1}{4}\sum_{m=1}^{T/K} \eta_{m-1}\mathbb{E}[||\nabla_{m,\delta}||^2] + 3L_T T\sqrt{\epsilon}$$

The proof of this lemma is given in C.6.

We next set $\eta_m = \frac{2R}{K(\frac{nM_{mK}}{\delta} + L_{mK}\sqrt{K})}$ and expanding $\mathbb{E}[||\nabla_{m,\delta}||^2]$ using Lemma C.8. We also set $\delta = \sqrt{n}T^{-1/4}$ and $K = \sqrt{T}$.

Overall, we get,

$$\begin{aligned}\mathbb{E}[\text{Regret}_T(\text{Algorithm 3})] \leqslant & \sqrt{n}L_T(3 + \frac{R}{r})T^{3/4} + 3L_T T\sqrt{\epsilon} \\ & + 2R(\sqrt{n}M_T + L_T)\big(1 + \max_{1 \leqslant m \leqslant T/K} \frac{\sqrt{n}M_{mK} + L_{mK}}{\sqrt{n}M_{(m-1)K} + L_{(m-1)K}}\big)T^{3/4}\end{aligned} \tag{33}$$

**Bounding the total number of calls to the oracle:** We begin by mentioning the following lemma-

**Lemma C.5.** *Under Algorithm 3, the expected overall number of calls to the linear optimization oracle is upper-bounded by*

$$\mathbb{E}\left[\sum_{m=1}^{\frac{T}{K}} N_m\right] \leqslant \frac{16R^4}{\epsilon^2}\left(\frac{\sqrt{\epsilon}}{R}T^{1/4} + 1 + \sqrt{\log\frac{\sqrt{n}M_T + L_T}{\sqrt{n}M_1 + L_1}}\right). \tag{34}$$

The proof of the lemma is given in C.7.

We can further bound the RHS of Eqn. (34) by $T$ by choosing $\epsilon$ appropriately.

### C.4 Proof of Theorem 4.2

We have to bound $\mathbb{E}\text{Regret}'_T$.

$$\mathbb{E}\text{Regret}'_T \leqslant \mathbb{E}[(3+\frac{R}{r})\sqrt{n}L_T T^{\frac{3}{4}} + 3L_T T\sqrt{\epsilon} + 2RG_T\big(1 + \max_{1\leqslant m\leqslant T/K}\frac{G_{mK}}{G_{(m-1)K}}\big)T^{3/4}]$$

and also, we have to bound

$$\mathbb{E}\left[\sum_{m=1}^{\frac{T}{K}} N_m\right] \leqslant \frac{16R^4}{\epsilon^2}\mathbb{E}\left(\frac{\sqrt{\epsilon}}{R}T^{1/4} + 1 + \sqrt{\log\frac{G_T}{G_1}}\right)$$

where $G_t = \sqrt{n}M_t + L_t$.

Now, note that $M_t = (V + pQ^{p-1}(T))F$ where $F$ is the sup-norm of cost and constraint functions. $L_t = (V + pQ^{p-1}(T))G$ where $G$ is the Lipchitz constant of cost and constraint functions. So, we have $G_t = (V + pQ^{p-1}(T))(\sqrt{n}F + G)$

We must first find the appropriate value of $\epsilon$ for which the expected total number of calls to the LOO is $T$. First, note that $\log\frac{G_{T/K}}{G_1} \leqslant \log\frac{V+pQ^{p-1}(T)}{V+pQ^{p-1}(1)} \leqslant \log\frac{Q^{p-1}(T)}{Q^{p-1}(1)} \leqslant \log Q^{p-1}(T) \leqslant \log T^{p-1} \leqslant (\log T)^2$. Therefore, the RHS of the above inequality is bounded by,

$$\frac{16R^4}{\epsilon^2}\left(\frac{\sqrt{\epsilon}}{R}T^{1/4} + 1 + \log T\right)$$

we can bound this by $T$, by taking $\epsilon = 16R^2T^{-1/2}\sqrt{\log T}$.

Now, let us look at the $\max_{1\leqslant m\leqslant T/K}\frac{G_m}{G_{m-1}}$ term in the inequality bounding $\mathbb{E}\text{Regret}'_T$. $\frac{G_m}{G_{m-1}} \leqslant \frac{V+pQ^{p-1}(mK)}{V+pQ^{p-1}((m-1)K)} \leqslant \frac{Q^{p-1}(mK)}{Q^{p-1}((m-1)K)} \leqslant (1+\frac{KGD}{Q(0)})^p \leqslant e$. The last inequality follows because $Q(0) = KGD\log T = \sqrt{T}GD\log T$. Therefore, we get,

$$\mathbb{E}\text{Regret}'_T \leqslant \mathbb{E}[(3+\frac{R}{r})\sqrt{n}L_T T^{\frac{3}{4}} + 12L_T RT^{3/4}(\log T)^{1/4} + 2RG_T\big(1+e\big)T^{3/4}]$$

We overall get,

$$\mathbb{E}\text{Regret}'_T \leqslant (30+\frac{R}{r})\sqrt{n}(V + \mathbb{E}\Phi'(Q(T)))T^{3/4}(\log T)^{1/4}$$

Therefore, from the regret decomposition inequality 16 and setting $c = (30+\frac{R}{r})\sqrt{n}$,

$$\mathbb{E}\Phi(Q(T)) - \Phi(Q(0)) + V\mathbb{E}\text{Regret}_T \leqslant c(V + \mathbb{E}\Phi'(Q(T)))T^{3/4}$$

Substituting, $\Phi(x) = x^p$, we get,

$$\mathbb{E}Q^p(T) - Q^p(0) + V\mathbb{E}\text{Regret}_T \leqslant cVT^{3/4} + cp\mathbb{E}Q^{p-1}(T)T^{3/4}$$

Bounding the Regret,

$$\mathbb{E}\text{Regret}_T \leqslant cT^{3/4} + \frac{1}{V}\mathbb{E}[\underbrace{cpQ^{p-1}(T)T^{3/4} - Q^p(T)}_{\leqslant (cpT^{3/4})^p}] + \frac{Q^p(0)}{V}$$

If we choose $V = (cpT^{3/4})^p$ then we get,

$$\mathbb{E}\text{Regret}_T \leqslant cT^{3/4} + 2$$

To bound the CCV, consider that $\text{Regret}_T \geqslant -T$,

$$\mathbb{E}Q^p(T) \leqslant 2cVT + cm\mathbb{E}Q^{p-1}(T)T^{3/4} + Q^p(0)$$

Either of the following two would hold in that case,

$$\begin{aligned}\mathbb{E}Q^p(T) \leqslant 4cVT + 2Q^p(0) \overset{(a)}{\implies} \mathbb{E}Q(T)^p \leqslant 4cVT + 2Q^p(0) \implies \mathbb{E}Q(T) &\leqslant (4cVT)^{1/p} + 2Q(0) \\ &\leqslant cpT^{3/4}(4cT)^{1/p} + 2GD\sqrt{T}\log T\end{aligned}$$

$$\begin{aligned}\mathbb{E}Q^p(T) \leqslant 2cp\mathbb{E}Q^{p-1}T^{3/4} \overset{(b)}{\implies} [\mathbb{E}Q^{p-1}(T)]^{\frac{p}{p-1}} \leqslant 2cp\mathbb{E}Q^{p-1}T^{3/4} &\implies [\mathbb{E}Q^{p-1}(T)]^{\frac{1}{p-1}} \leqslant 2cpT^{3/4} \\ &\overset{(c)}{\implies} \mathbb{E}Q(T) \leqslant 2cpT^{3/4}\end{aligned}$$

where (a) follows from the convexity of the map $x \mapsto x^p$ and using Jensen's inequality and $(b)$ follows from applying Jensen's inequality on the LHS to the convex map $x \mapsto x^{\frac{p}{p-1}}$, resulting in

$$\mathbb{E}(Q^p(T)) = \mathbb{E}\big[\big(Q^{p-1}(T)\big)^{\frac{p}{p-1}}\big] \geqslant \mathbb{E}\big[Q^{p-1}(T)\big]^{\frac{p}{p-1}},$$

(c) follows from the convexity of the map $x \mapsto x^{p-1}$ and using Jensen's inequality.

Thus, both expected regret and violation are $\tilde{O}(T^{3/4})$.

### C.5 Proof of Lemma C.3

$$\begin{aligned}&\mathbb{E}[\text{Regret}_T(\text{Algorithm 3})] \\ =\ &\sum_{t=1}^{T}\mathbb{E}[\big(\hat{f}_t(y_t)] - \hat{f}_t(x^\star)\big) \\ =\ &\underbrace{\sum_{t=1}^{T}\big(\mathbb{E}[\hat{f}_t(y_t)] - \mathbb{E}[\hat{f}_t(x_{m(t)})]\big)}_{(A)} + \underbrace{\sum_{t=1}^{T}\big(\hat{f}_t(\tilde{x}^\star) - \hat{f}_t(x^\star)\big)}_{(B)} + \underbrace{\sum_{t=1}^{T}\big(\mathbb{E}[\hat{f}_t(x_{m(t)})] - \hat{f}_t(\tilde{x}^\star)\big)}_{(C)},\end{aligned} \tag{35}$$

Bounding term (A):

$$\begin{aligned}&\sum_{t=1}^{T}\big(\mathbb{E}[\hat{f}_t(y_t)] - \mathbb{E}[\hat{f}_t(x_{m(t)})]\big) = \sum_{t=1}^{T}\big(\mathbb{E}[\hat{f}_t(x_{m(t)} + \delta u_t) - \hat{f}_t(x_{m(t)})]\big) \leqslant \sum_{t=1}^{T} L_t\mathbb{E}[\|\delta u_t\|] \\ \leqslant\ &L_T\delta T\end{aligned} \tag{36}$$

Bounding term (B):

$$\sum_{t=1}^{T}\big(\hat{f}_t(\tilde{x}^\star)-\hat{f}_t(x^\star)\big)=\sum_{t=1}^{T}\big(\hat{f}_t((1-\frac{\delta}{r})x^\star)-\hat{f}_t(x^\star)\big)\leqslant\sum_{t=1}^{T}L_t\|(1-\frac{\delta}{r})x^\star-x^\star\|$$
$$\leqslant\quad\frac{\delta R L_T T}{r} \tag{37}$$

Bounding term (C):

$$\sum_{t=1}^{T}\big(\mathbb{E}[\hat{f}_t(x_{m(t)})]-\hat{f}_t(\tilde{x}^\star)\big)$$
$$\leqslant\quad\sum_{t=1}^{T}\big(\mathbb{E}[\hat{f}_{t,\delta}(x_{m(t)})]-\hat{f}_{t,\delta}(\tilde{x}^\star)\big)+\sum_{t=1}^{T}\mathbb{E}[\underbrace{\big(\hat{f}_t(x_{m(t)})-\hat{f}_{t,\delta}(x_{m(t)})\big)}_{\leqslant L_T\delta}]+\sum_{t=1}^{T}\mathbb{E}[\underbrace{\big(\hat{f}_{t,\delta}(\tilde{x}^\star)-\hat{f}_t(\tilde{x}^\star)\big)}_{\leqslant L_T\delta}]$$
$$\leqslant\quad\sum_{t=1}^{T}\big(\mathbb{E}[\hat{f}_{t,\delta}(x_{m(t)})]-\hat{f}_{t,\delta}(\tilde{x}^\star)\big)+2\delta L_T T=\text{Regret}_T(\text{blocked-OCG})+2\delta L_T T \tag{38}$$

### C.6 Proof of Lemma C.4

Define $x_m^\star=\underset{x}{\text{argmin}}\,F_{m-1}(x)$.

Where $F_m$ is defined in Eqn. (14). Using similar arguments as in the proof of Theorem 2.1, we get the following -

$$\sum_{t=1}^{T}\big(\mathbb{E}[\hat{f}_{t,\delta}(x_{m(t)})]-\hat{f}_{t,\delta}(\tilde{x}^\star)\big)\leqslant 3\underbrace{\sum_{t=1}^{T}L_t\mathbb{E}[\|x_{m(t)}-x^\star_{m(t)}\|]}_{(A)}+\underbrace{\sum_{t=1}^{T}\big(\mathbb{E}[\tilde{f}_{t,\delta}(x^\star_{m(t)})]-\tilde{f}_{t,\delta}(\tilde{x}^\star)\big)}_{(B)} \tag{39}$$

It is easy to observe the similarity between the above equation and the Eqns. (19) and (21). The only difference is the use of the expectation operator which is an artefact of the randomised nature of the bandit algorithm. $\tilde{f}_{t,\delta}$ is the smoothed analogue of $\tilde{f}_t$ which is as defined in Eqn. (18).

In this case bounding term (A) wouldn't be as tedious as it was in the case of Algorithm 1. This is because Algorithm 6 (line 9) ensures that it remains bounded by $\sqrt{\epsilon}$. This is demonstrated in the following inequalities.

$$\frac{\epsilon}{\eta_{m-1}}\geqslant g_{m-1}^\top(x_m-v_{L_m})\geqslant g_{m-1}^\top(x_m-x_m^\star)\geqslant F_{m-1}(x_m)-F_{m-1}(x_m^\star)\geqslant\frac{\|x_m-x_m^\star\|^2}{\eta_{m-1}} \tag{40}$$

Algorithm 6 ensures the first inequality by virtue of its loop condition. Where $g_{m-1}=\nabla F_{m-1}(x_m)$ and $v_{L_m}$ denotes the value of $v_\tau$ in the last iteration of the loop in Algorithm 6. The second inequality follows from the fact that $v_{L_m}$ is the minimiser of $g_{m-1}^\top x$. The third inequality follows from the convexity of $F_m$. The last inequality follows from the $\frac{2}{\eta_m}$-strong convexity of $F_m$. Now we bound term (B) -

$$(B)\equiv\sum_{m=1}^{T/K}\big(\mathbb{E}[\tilde{f}_{(m-1)K+1:mK,\delta}(x_m^\star)]-\tilde{f}_{(m-1)K+1:mK,\delta}(\tilde{x}^\star)\big) \tag{41}$$

The : notation is used to denote accumulation and $f_{t_1:t_2}$ denotes the sum of losses from $t_1$ to $t_2$ (both inclusive). Eqn. (41) makes it clear that term (B) is essentially the regret of the blocked FTRL algorithm where the same notion of blocking is used. This is essentially a reduction from a blocking algorithm to a non-blocking one as mentioned in Remark C.1. We appeal to Theorem B.3.

$$(B)\leqslant\frac{D^2}{\eta_{T/K}}+\frac{1}{4}\sum_{m=1}^{T/K}\eta_{m-1}\mathbb{E}[||\nabla_m||^2] \tag{42}$$

In this case, $\nabla_m$ is used the denote the accumulated gradients of the block. We further appeal to Lemma C.8 and substitute the value of $\eta_m$, $\delta$ and $K$ and also noting that both $\{M_t\}_{1\leqslant t\leqslant T}$ and $\{L_t\}_{1\leqslant t\leqslant T}$ are non-decreasing sequences.

### C.7 Proof of Lemma C.5

*Remark* C.6. Note that the function $\eta_m F_m$ is directly applicable to Lemma C.9 and scaling $F_m$ by $\eta_m$ does not affect the optimization in Algorithm 6 either. Henceforth, by $F_m$ we would denote the scaled version of the previous $F_m$. Thus it will be 2-smooth and 2-strongly convex.

Let $z_{m,\tau}$ be the iterate of Algorithm 6 after completing $\tau - 1$ iterations of the do-while loop, when invoked on iteration (block) $m$ of Algorithm 3. Also, for all $m, \tau$, define $h_{m,\tau} := F_m(z_{m,\tau}) - F_m(x^\star_{m+1})$. Recall that for any iteration $m$ of Algorithm 3, we have $z_{m,1} = x_m$.

Using the triangle inequality and the fact $F_{m-1}(x^\star_{m+1}) \geqslant F_{m-1}(x^\star_m)$, we have

$$\mathbb{E}[h_{m,1}] = \mathbb{E}[F_m(z_{m,1}) - F_m(x^\star_{m+1})] \leqslant \mathbb{E}[F_{m-1}(x_m) - F_{m-1}(x^\star_m) + \eta_m \|\nabla_m\| \; \|x_m - x^\star_{m+1}\|].$$

Since $h_{m-1,N_{m-1}} = F_{m-1}(x_m) - F_{m-1}(x^\star_m) \leqslant \epsilon$, using the triangle inequality, we have

$$\mathbb{E}[h_{m,1}] \leqslant \epsilon + \eta_m \mathbb{E}[\|\nabla_m\| \|x_m - x^\star_m + x^\star_m - x^\star_{m+1}\|] \leqslant \epsilon + \eta_m \mathbb{E}[\|\nabla_m\| \left(\|x_m - x^\star_m\| + \|x^\star_m - x^\star_{m+1}\|\right)].$$

Since $F_m(x)$ is 2-strongly convex and $h_{m,L_m} \leqslant \epsilon$, we have that $\|x_m - x^\star_m\| \leqslant \sqrt{\epsilon}$. Also, from Eq. (46), we have $\|x^\star_m - x^\star_{m+1}\| \leqslant \eta_m \|\nabla_m\| + D\sqrt{1 - \frac{G_{m-1}}{G_m}}$. Thus, we have

$$\mathbb{E}[h_{m,1}] \leqslant \epsilon + \eta_m \sqrt{\epsilon}\; \mathbb{E}[\|\nabla_m\|] + \eta_m^2 \mathbb{E}[\|\nabla_m\|^2] + \eta_m \mathbb{E}[\|\nabla_m\|] D \sqrt{1 - \frac{G_{m-1}}{G_m}}.$$

Using Lemma C.8 and the fact that for all $a, b \in \mathbb{R}^+$ it holds that that $\sqrt{a+b} \leqslant \sqrt{a} + \sqrt{b}$, we have

$$\begin{aligned}\mathbb{E}[h_{m+1,1}] \leqslant \epsilon + \eta_m \sqrt{\epsilon}\; \left(\sqrt{K}\left(\frac{nM_{mK}}{\delta}\right) + KL_{mK}\right) + \eta_m^2 \left(K\left(\frac{nM_{mK}}{\delta}\right)^2 + K^2 L_{mK}^2\right) \\ + \eta_m D \sqrt{1 - \frac{G_{m-1}}{G_m}} \left(\sqrt{K}\left(\frac{nM_{mK}}{\delta}\right) + KL_{mK}\right). \end{aligned} \tag{43}$$

Using Lemma C.9 with $\tilde{\epsilon}$ as the RHS of (43) for $m = 1, \ldots, \frac{T}{K}$, we have that on each iteration (block) $m$, the number of calls to the linear optimization oracle is $N_m \leqslant \max\left\{\frac{16R^2}{\epsilon^2}(h_{m,1} - \epsilon),\; \frac{2}{\epsilon}(h_{m,1} - \epsilon)\right\}$. Plugging-in $\epsilon$, we have $N_m \leqslant \frac{16R^2}{\epsilon^2}(h_{m,1} - \epsilon)$. Following Eq. (43) we have

$$\begin{aligned}&\mathbb{E}[N_m] \\ &\leqslant \frac{16R^2}{\epsilon^2}(\mathbb{E}[h_{m,1}] - \epsilon) \\ &\leqslant \frac{16R^2}{\epsilon^2}\eta_m \sqrt{\epsilon}\; \left(\sqrt{K}\left(\frac{nM_{mK}}{\delta}\right) + KL_{mK}\right) \\ &+ \frac{16R^2}{\epsilon^2}\eta_m^2 \left(K\left(\frac{nM_{mK}}{\delta}\right)^2 + K^2 L_{mK}^2\right) + \frac{16R^2}{\epsilon^2}\eta_m D \sqrt{1 - \frac{G_{m-1}}{G_m}} \left(\sqrt{K}\left(\frac{nM_{mK}}{\delta}\right) + KL_{mK}\right). \end{aligned} \tag{44}$$

Thus, overall on all blocks, we obtain

$$\begin{aligned}&\mathbb{E}\left[\sum_{m=1}^{\frac{T}{K}} N_m\right] \\ &\leqslant \frac{16R^2}{\epsilon^2}\left(\sum_{m=1}^{\frac{T}{K}} \eta_m \sqrt{\epsilon}\; \left(\sqrt{K}\left(\frac{nM_{mK}}{\delta}\right) + KL_{mK}\right) + \eta_m^2 \left(K\left(\frac{nM_{mK}}{\delta}\right)^2 + K^2 L_{mK}^2\right)\right. \\ &\left. + \sqrt{\sum_{m=1}^{\frac{T}{K}} \eta_m^2 \left(K\left(\frac{nM_{mK}}{\delta}\right)^2 + K^2 L_{mK}^2\right)}\, D \sqrt{\log \frac{G_{T/K}}{G_1}}\right) \end{aligned}$$

$$\leqslant \frac{16R^4}{\epsilon^2}\left(\frac{\sqrt{\epsilon}}{R}T^{1/4}+1+\sqrt{\log\frac{G_{T/K}}{G_1}}\right) \tag{45}$$

Where the last inequality follows because we take $\eta_m = \frac{2R}{K(\frac{nM_{mK}}{\delta}+L_{mK}\sqrt{K})}$.

### C.8 Auxiliary Lemmas

**Lemma C.7.** *Under Algorithm 3, the following holds*

$$\|x_m^\star - x_{m+1}^\star\| \leqslant \eta_m\|\nabla_m\| + D\sqrt{(1-\frac{G_{m-1}}{G_m})}$$

*Proof.*

$$\begin{aligned}
&\|x_m^\star - x_{m+1}^\star\|^2/\eta_m \leqslant F_m(x_m^\star) - F_m(x_{m+1}^\star) \leqslant \underbrace{F_{m-1}(x_m^\star) - F_{m-1}(x_{m+1}^\star)}_{\leqslant 0} + \nabla_m^\top(x_m^\star - x_{m+1}^\star)\\
&+\|x_m^\star - x_1^\star\|^2(\frac{1}{\eta_m}-\frac{1}{\eta_{m-1}})\\
\implies\quad &\|x_m^\star - x_{m+1}^\star\|^2 \leqslant \eta_m\|\nabla_m\|\|(x_m^\star - x_{m+1}^\star)\| + D^2(1-\frac{G_{m-1}}{G_m})\\
\implies\quad &\|x_m^\star - x_{m+1}^\star\| \leqslant \eta_m\|\nabla_m\| + D\sqrt{(1-\frac{G_{m-1}}{G_m})}.
\end{aligned} \tag{46}$$

□

**Lemma C.8.** *For any iteration (block) $m$ of the outer-loop in Algorithm 3 it holds that*

$$\mathbb{E}\left[\|\nabla_m\|\right]^2 \leqslant \mathbb{E}\left[\|\nabla_m\|^2\right] \leqslant K\left(\frac{nM_{mK}}{\delta}\right)^2 + K^2L_{mK}^2.$$

*Proof.*

$$\begin{aligned}
\mathbb{E}\left[\|\nabla_m\|^2\right] &= \mathbb{E}\left[\|\sum_{s=1}^K \nabla_{m,s}\|^2\right]\\
&=\mathbb{E}\left[\sum_{s=1}^K\|\nabla_{m,s}\|^2 + \sum_{s_2=1}^K\sum_{s_1=1,s_1\neq s_2}^K \nabla_{m,s_1}^\top\nabla_{m,s_2}\right]\\
&=\mathbb{E}\left[\sum_{s=1}^K\|\nabla_{m,s}\|^2\right] + \sum_{s_2=1}^K\sum_{s_1=1,s_1\neq s_2}^K \mathbb{E}[\nabla_{m,s_1}^\top\nabla_{m,s_2}].
\end{aligned}$$

Since, conditioned on the iterate $x_{m-1}$, $\forall s_1 \neq s_2$ $\nabla_{m,s_1}$, $\nabla_{m,s_2}$ are independent random vectors, we have

$$\mathbb{E}\left[\|\nabla_m\|^2\right] = \mathbb{E}\left[\sum_{s=1}^K\|\nabla_{m,s}\|^2\right] + \sum_{s_2=1}^K\sum_{s_1=1,s_1\neq s_2}^K \mathbb{E}[\nabla_{m,s_1}^\top]\mathbb{E}[\nabla_{m,s_2}].$$

We have that for all $(m-1)K+1 \leqslant t \leqslant mK$, $\|\mathbb{E}[\nabla_t|x_m]\| = \|\nabla\hat{f}_{t,\delta}(\mathbf{x}_m)\| \leqslant L_{mK}$. Since $\max_{\mathbf{x}\in\mathcal{K}}\|f(\mathbf{x})\| \leqslant M$, we also have $\|\nabla_{m,s}\| \leqslant \frac{n}{\delta}\|f_t(\mathbf{y}_t)\|\|\mathbf{u}_t\| \leqslant \frac{nM_{mK}}{\delta}$, and thus,

$$\begin{aligned}
&\mathbb{E}\left[\sum_{s=1}^K\|\nabla_{m,s}\|^2\right] + \sum_{s_2=1}^K\sum_{s_1=1,s_1\neq s_2}^K \mathbb{E}[\nabla_{m,s_1}^\top]\mathbb{E}[\nabla_{m,s_2}]\\
&\leqslant K\left(\frac{nM_{mK}}{\delta}\right)^2 + \left(K^2-K\right)L_{mK}^2 \leqslant K\left(\frac{nM_{mK}}{\delta}\right)^2 + K^2L_{mK}^2.
\end{aligned}$$

Finally, the inequality $\mathbb{E}\left[\|\nabla_m\|\right]^2 \leqslant \mathbb{E}\left[\|\nabla_m\|^2\right]$ stated in the lemma follows from using Jensen's inequality. □

**Lemma C.9.** *Given a function $F(\mathbf{x})$, 2-smooth and 2-strongly convex, and $\mathbf{x}_1 \in \mathcal{X}_\delta$ such that $h_1 = F(x_1) - F(x^\star) \leqslant \tilde{\epsilon}$, where $x^\star = \operatorname*{argmin}_{x \in \mathcal{X}_\delta} F(x)$, Algorithm 6 produces a point $x_{L+1} \in \mathcal{X}_\delta$ such that $F(x_{L+1}) - F(x^\star) \leqslant \epsilon$ after at most $L = \max\left\{\frac{16R^2}{\epsilon^2}(h_1 - \epsilon),\ \frac{2}{\epsilon}(h_1 - \epsilon)\right\}$ iterations.*

*Proof.* For any iteration $\tau$ of Algorithm 6, define $h_\tau = F(x_\tau) - F(x^\star)$ and denote $\nabla_\tau = \nabla F(x_\tau)$. From the definition of $v_\tau$ (in Line 6 of Algorithm 6)and the convexity of $F(\cdot)$, it follows that

$$\begin{aligned}\nabla_\tau^\top (x_\tau - v_\tau) \geqslant & \nabla_\tau^\top (x_\tau - x^\star) \\ \geqslant & F(x_\tau) - F(x^\star) = h_\tau\end{aligned} \tag{47}$$

Now, we establish the convergence rate of Algorithm 6. It holds that

$$\begin{aligned}h_{\tau+1} =& F(x_{\tau+1}) - F(x^\star) \\ =& F(x_\tau + \sigma_\tau(v_\tau - x_\tau)) - F(x^\star).\end{aligned}$$

For our analysis we define the step-size $\hat{\sigma}_\tau = \min\left\{\frac{\nabla_\tau^\top (x_\tau - v_\tau)}{8R^2}, 1\right\}$. Since $\sigma_\tau$ is chosen via line-search, we have that

$$\begin{aligned}h_{\tau+1} =& F(x_\tau + \sigma_\tau(v_\tau - x_\tau)) - F(x^\star) \\ \leqslant& F(x_\tau + \hat{\sigma}_\tau(v_\tau - x_\tau)) - F(x^\star).\end{aligned}$$

The above step was done to ensure that we can use our own $\hat{\sigma}_\tau$ to reach our bound.

Since $F(x)$ is 2-smooth it holds that

$$\begin{aligned}F(x_\tau + \hat{\sigma}_\tau(v_\tau - x_\tau)) \leqslant & F(x_\tau) + \hat{\sigma}_\tau \nabla_\tau^\top (v_\tau - x_\tau) \\ & + \hat{\sigma}_\tau^2 \|v_\tau - x_\tau\|^2,\end{aligned}$$

and we obtain

$$h_{\tau+1} \leqslant h_\tau + \hat{\sigma}_\tau^2 (2R)^2 - \hat{\sigma}_\tau \nabla_\tau^\top (x_\tau - v_\tau).$$

We now consider several cases.
*Case 1*: If $\nabla_\tau^\top (x_\tau - v_\tau) \leqslant \epsilon$ for some $\tau < L$, the algorithm will stop after less than $L$ iterations. Moreover, from Eq. (47) we have $h_\tau \leqslant \epsilon$.
*Case 2*: Else, $\nabla_\tau^\top (x_\tau - v_\tau) \geqslant \epsilon$ for all $\tau < L$. We have 2 cases:
*Case 2.1*: If $\nabla_\tau^\top (x_\tau - v_\tau) \geqslant 8R^2$ then $\hat{\sigma}_\tau = 1$ and we have

$$\begin{aligned}h_{\tau+1} \leqslant & h_\tau + \hat{\sigma}_\tau^2 (2R)^2 - \hat{\sigma}_\tau \nabla_\tau^\top (x_\tau - v_\tau) \\ \leqslant & h_\tau - \frac{\nabla_\tau^\top (x_\tau - v_\tau)}{2}.\end{aligned}$$

*Case 2.2*: Else, $\nabla_\tau^\top (x_\tau - v_\tau) \leqslant 8R^2$, and then $\hat{\sigma}_\tau = \frac{\nabla_\tau^\top (x_\tau - v_\tau)}{8R^2}$, and we have

$$\begin{aligned}h_{\tau+1} \leqslant & h_\tau + \hat{\sigma}_\tau^2 (2R)^2 - \hat{\sigma}_\tau \nabla_\tau^\top (x_\tau - v_\tau) \\ \leqslant & h_\tau - \left(\frac{\nabla_\tau^\top (x_\tau - v_\tau)}{4R}\right)^2.\end{aligned}$$

From both cases, we have

$$\begin{aligned}h_{\tau+1} \leqslant & h_\tau - \min\left\{\left(\frac{\nabla_\tau^\top (x_\tau - v_\tau)}{4R}\right)^2, \frac{\nabla_\tau^\top (x_\tau - v_\tau)}{2}\right\} \\ \leqslant & h_1 - \tau \min_{i=1,\ldots,\tau}\left\{\left(\frac{\nabla_\tau^\top (x_i - v_i)}{4R}\right)^2, \frac{\nabla_\tau^\top (x_i - v_i)}{2}\right\}\end{aligned}$$

$$\leqslant h_1 - \tau \min\left\{ \left(\frac{\epsilon}{4R}\right)^2, \frac{\epsilon}{2} \right\}. \tag{48}$$

Thus, for all cases, after a maximum of $L$ iterations, when

$$L = \max\left\{ \frac{16R^2}{\epsilon^2}(h_1 - \epsilon),\ \frac{2}{\epsilon}(h_1 - \epsilon) \right\},$$

we obtain $h_{L+1} \leqslant \epsilon$.

So, when the loop naturally terminates, the condition is met - even if the loop doesn't terminate, we need to run it for a maximum of $L$ times to ensure that the condition is met. □

## D Proof of Theorem 5.1

The following pseudocode, given in Algorithm 7, describes the proposed FTPL-based projection-free algorithm for the OLO problem with constraints.

**Algorithm 7** Adaptive FTPL with Adversarial Constraints

**Require:** Compact decision set $\mathcal{X} \subseteq \mathbb{R}^N$, parameter $V = \sqrt{T}$, linear cost and constraint functions with coefficients $\{f_t, g_t\}_{t \geqslant 1}$, Gaussian Perturbation $\mathcal{N}(0, I_N)$

**Initialize:** Virtual queue $Q(0) = 0$, Cumulative surrogate loss vector $L_0 = \mathbf{0}$
**for** $t = 1 : T$ **do**
  Calculate adaptive scaling parameter:

$$\eta_t = \sqrt{\frac{1 + \sum_{\tau=1}^{t-1} \|\hat{f}_\tau\|_2^2}{N}}$$

  Sample perturbation vector $p_t \sim \mathcal{N}(0, I)$
  The learner plays $x_t = \arg\min_{x \in \mathcal{X}} \langle L_{t-1} + \eta_t p_t, x \rangle$
  The adversary reveals cost vector $f_t$ and constraint vector $g_t$
  Update the virtual queue: $Q(t) = (Q(t-1) + \langle g_t, x_t \rangle)^+$
  Construct surrogate loss vector: $\hat{f}_t = V f_t + 2Q(t-1) g_t$
  Update cumulative surrogate loss: $L_t = L_{t-1} + \hat{f}_t$
**end for**

**Regret and Constraint Violation Analysis:** Recall that we consider the online linear optimization setting with linear cost and constraint functions $f_t(x) = \langle f_t, x \rangle$ and $g_t(x) = \langle g_t, x \rangle$ such that $\|f_t\|, \|g_t\| \leqslant 1,\ \forall t \geqslant 1$, and the action $x$ belongs to some compact set $\mathcal{X}$. Our analysis closely follows the derivation in Section 3.2.

Define the virtual queue for Cumulative Constraint Violation (CCV) as:

$$Q(t) = (Q(t-1) + \langle g_t, x_t \rangle)^+ \tag{49}$$

and choose the quadratic Lyapunov function $\Phi(x) = \frac{x^2}{V}$, where $V > 0$ is a constant to be determined later. Eqn. (49) yields the following drift inequality:

$$Q^2(t) \leqslant Q^2(t-1) + 1 + 2Q(t-1)\langle g_t, x_t \rangle. \tag{50}$$

Let $x^\star$ be a feasible solution to the constraints. Adding $V(f_t(x_t) - f_t(x^\star))$ to both sides of the above and then summing up the resulting inequalities, we obtain the following regret decomposition inequality:

$$Q^2(t) + V \sum_{\tau=1}^{t} \big(f_\tau(x_\tau) - f_\tau(x^\star)\big) \leqslant t + \sum_{\tau=1}^{t} \big(\hat{f}_\tau(x_\tau) - \hat{f}_\tau(x^\star)\big), \tag{51}$$

where we have defined the surrogate linear cost function for round $t$ as:

$$\hat{f}_t(x) = \langle V f_t + 2Q(t-1) g_t, x \rangle. \tag{52}$$

Finally, running the FTPL policy with Gaussian noise and using the standard adaptive regret bound, given by [Abernethy et al., 2016, Theorem 1.10], on these surrogate costs, we obtain the following regret bound on the surrogate cost functions:

$$\text{Regret}'_t \leqslant 2\sqrt{1+\sum_{\tau=1}^{t}(V^2+4Q^2(\tau-1))} \leqslant 2+V\sqrt{t}+4\sqrt{\sum_{\tau=1}^{t-1}Q^2(\tau)}. \tag{53}$$

By scaling the Gaussian perturbation $p_t$ at each round $t$ by the adaptive parameter $\eta_t = \sqrt{(1+\sum_{\tau=1}^{t-1}\|\hat{f}_\tau\|_2^2)/N}$, the algorithm dynamically adjusts to the magnitude of the observed surrogate losses.

Combining this with the Lyapunov drift yields the following fundamental regret-decomposition inequality:

$$\mathbb{E}Q^2(t) + V\,\mathbb{E}\text{Regret}_t \leqslant 2+t+V\sqrt{t}+4\sqrt{\sum_{\tau=1}^{t-1}\mathbb{E}Q^2(\tau)}. \tag{54}$$

Following an inductive analysis similar to Sinha and Vaze [2024], this decomposition guarantees sublinear bounds, which we rigorously show in the following.

Summing from $t=1$ to $T$ we have,

$$\sum_{t=1}^{T}\mathbb{E}Q^2(t) + V\sum_{t=1}^{T}\mathbb{E}\text{Regret}_t \leqslant VT^{3/2}+2T+T^2+4T\sqrt{\sum_{t=1}^{T}\mathbb{E}Q^2(t)}. \tag{55}$$

Let us now define the variable $R(T) := \sqrt{\sum_{t=1}^{T}\mathbb{E}Q^2(t)}$. Note that we trivially have $\mathbb{E}\text{Regret}_t \geqslant -T$. Plugging in this bound in inequality (55), we conclude:

$$R^2(T) \leqslant VT^2+T+T^2+4VT^{3/2}+4TR(T).$$

Noticing that the above inequality is of the form $x^2 \leqslant ax+b$ where $x \equiv R(T)$, and because this implies $x \leqslant a+\sqrt{b}$, we obtain the following upper bound on $R(T)$:

$$R(T) \leqslant \sqrt{V}T+\sqrt{T}+2\sqrt{V}T^{3/4}+5T. \tag{56}$$

We further note that inequality (54) can be rewritten in term of the variables $R(T)$ as follows:

$$\mathbb{E}Q^2(T) + V\mathbb{E}\mathsf{Regret}_T \leqslant V\sqrt{T}+2+T+4R(T). \tag{57}$$

Plugging in the upper bound on $R(T)$ from (56) into the above inequality, we obtain

$$\mathbb{E}Q^2(T) + V\mathbb{E}\text{Regret}_T \leqslant 4V\sqrt{T}+2+T+4\sqrt{V}T+8\sqrt{T}+8\sqrt{V}T^{3/4}+20T. \tag{58}$$

Hence, using $\mathbb{E}(Q^2(T)) \geqslant 0$, we obtain the following regret bound:

$$\mathbb{E}\mathsf{Regret}_T \leqslant O\Big(\max(\sqrt{T}, \sqrt{\frac{T}{V}}, \frac{T^{3/4}}{\sqrt{V}}, \frac{T}{V})\Big).$$

Furthermore, substituting the trivial regret lower bound $\mathbb{E}\text{Regret}_T \geqslant -T$ into (58) and using Jensen's inequality $\mathbb{E}Q^2(T) \geqslant (\mathbb{E}(Q(T)))^2$, we obtain the following CCV bound

$$\mathbb{E}Q(T) \leqslant O\Big(\max(\sqrt{VT}, \sqrt{V}(T)^{1/4}, (V)^{1/4}\sqrt{T}, V^{1/4}T^{3/8}, \sqrt{T})\Big). \tag{59}$$

Choosing $V=\sqrt{T}$, we conclude $\mathbb{E}\mathsf{Regret}_T = O(T^{3/4})$, $\mathbb{E}\mathsf{CCV}_T = O(T^{3/4})$.

## E Budget Constrained Benchmark

Let $B_T$ denote the long-term budget and let us denote the set of all budget-feasible benchmarks by $\mathcal{X}^\star$, i.e.,

$$\mathcal{X}^\star = \{x \in \mathcal{X} : \sum_{t=1}^{T} g_t^+(x) \leqslant B_T\}.$$

As before,

$$\Phi(Q(t)) - \Phi(Q(t-1)) \leqslant \Phi'(Q(t))g_t(x_t)^+$$

$$\Phi(Q(t))-\Phi(Q(t-1))+Vf_t(x_t)-Vf_t(x^\star)-\Phi'(Q(t))g_t^+(x^\star) \leqslant Vf_t(x_t)+\Phi'(Q(t))g_t^+(x_t)-Vf_t(x^\star)-\Phi'(Q(t))g_t^+(x^\star)$$

where we added $Vf_t(x_t)-Vf_t(x^\star)-\Phi'(Q(t))g_t^+(x^\star)$ and re-arranged the terms on the RHS to make the structure of the surrogate loss apparent.

Noting that, $\hat{f}_t(x) = Vf_t(x) + \Phi'(Q(t))g_t^+(x)$.

$$\Phi(Q(t)) - \Phi(Q(t-1)) + Vf_t(x_t) - Vf_t(x^\star) \leqslant \hat{f}_t(x_t) - \hat{f}_t(x^\star) + \Phi'(Q(t))g_t^+(x^\star)$$

Further summing from $t = 1$ to $T$,

$$\Phi(Q(T)) - \Phi(Q(0)) + V\text{Regret}_T(x^\star) \leqslant \text{Regret}'(x^\star) + \sum_{t=1}^{T} \Phi'(Q(t))g_t^+(x^\star)$$

$$\Phi(Q(T)) - \Phi(Q(0)) + V\text{Regret}_T(x^\star) \leqslant \text{Regret}'(x^\star) + \Phi'(Q(T)) \sum_{t=1}^{T} g_t^+(x^\star)$$

$$\Phi(Q(T)) - \Phi(Q(0)) + V\text{Regret}_T(x^\star) \leqslant \text{Regret}'(x^\star) + \Phi'(Q(T))B_T.$$

Noting that,

$$\text{Regret}'_T \leqslant T^{3/4}(V + \Phi'(Q(T)))\log T.$$

Then we get,

$$\Phi(Q(T)) - \Phi(Q(0)) + V\text{Regret}_T(x^\star) \leqslant VT^{3/4}\log T + \Phi'(Q(T))(T^{3/4}\log T + B_T)$$

This is the new regret-decomposition inequality, which extends Eq. (10) to the arbitrary budget setting. Calculations similar to those in our main paper with the power-law potential would yield,

$$\text{Regret}_T(x^\star) = \tilde{O}(T^{3/4})$$

$$Q(T) = \tilde{O}(T^{3/4}) + B_T \log T$$

It should be noted that for $B_T \leqslant T^{3/4}$, the strengthening of the benchmark does not make the violation worse.

## F Experiments: Online Shortest Path Problem with Constraints

**Setup:** We consider a constrained version of the online shortest path problem [Hazan, 2022], where the length corresponds to the latency and the constraint is on the long-term cumulative bandwidth across the path. In this problem, on each round, the online algorithm first selects a route connecting a source $s$ to a destination $d$ on a graph $G(V, E)$. The latency and bandwidth of each edge vary across rounds, reflecting dynamic network conditions. The objective is to minimize the cumulative latency subject to a long-term lower bound on the cumulative bandwidth. Formally, the following sequence of events takes place on the $t^{\text{th}}$ round:

1. The algorithm first chooses a route (randomly or otherwise) $p_t \in P_{s,d}$, where $P_{s,d}$ is the set of all $s-d$ routes in the graph.

2. A latency of $\tau_e(t)$ and a bandwidth of $l_e(t)$ is chosen by an adversary for each edge $e \in E$.
3. The algorithm incurs a latency cost of $\sum_{e \in p_t} \tau_e(t)$ and a bandwidth cost of $-\sum_{e \in p_t} l_e(t)$ on round $t$.

We represent each route $p$ by its corresponding $|E|$-dimensional binary incidence vector where $p_e = 1$ if the edge $e \in E$ belongs to the route or $p_e = 0$ otherwise. On each round, our online policy returns an element from the convex hull of $P_{s,d}$, also known as the unit flow polytope [Hazan, 2022]. We use Dijkstra's algorithm for computing the weighted shortest route in Eqn. (3). It is well-known that any element in the unit flow polytope can be efficiently decomposed into a convex combination of at most $|E|$ number of $s-d$ routes using the flow decomposition lemma [Williamson, 2019, Lemma 2.20]. These convex combinations can be used to randomly select a single route on each round, incurring the same expected cost. The experiments are performed on a quad-core CPU with 8 GB RAM.

**Dataset:** For the experiments, we construct a synthetic dataset from the raw data collected from RIPE Atlas - a global network measurement platform [Staff, 2015]. The dataset provides real-time internet measurements, including latency and bandwidth, with temporal variations introduced through random scaling. We construct a graph with $n = 191$ nodes and $m = 1200$ edges where each edge is characterized by its latency and bandwidth values. We consider the horizon length to be $T = 1600$ for our experiments.

**Results:** We compare our algorithm with three other competing algorithms which consider similar problem settings. The first is the COCO algorithm proposed by Sinha and Vaze [2024], and the second is the Algorithm 3 of Garber and Kretzu [2024] and finally we have the algorithm proposed by Wang et al. [2025]. The parameters have been set as specified in the corresponding papers. We show the CCV plot in Figure 1, Regret plot in Figure 2, and execution times in Figure 3 for all three algorithms.

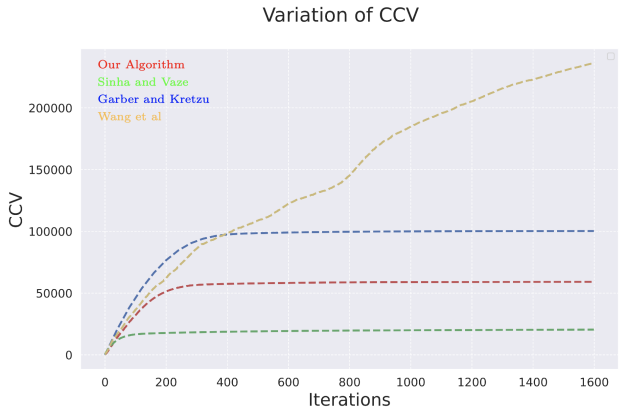


Figure 1: CCV comparison between our policy, Algorithm 1 of Sinha and Vaze [2024], Algorithm 3 of Garber and Kretzu [2024] and Algorithm 1 of Wang et al. [2025].

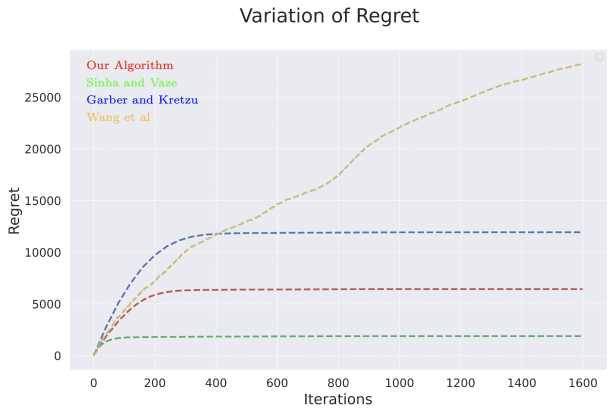


Figure 2: Regret comparison between our policy, Algorithm 1 of Sinha and Vaze [2024], Algorithm 3 of Garber and Kretzu [2024] and Algorithm 1 of Wang et al. [2025].

Figures 1 and 2 demonstrate that, in terms of regret and CCV, the algorithm proposed by Sinha and Vaze [2024] performs best when the computational cost is disregarded. This is not surprising because it is a computationally expensive projection-based algorithm whose theoretical guarantees of regret and CCV are much stronger ($O(\sqrt{T})$ and $\tilde{O}(\sqrt{T})$ respectively). However, this is more than compensated by the computational efficiency of our proposed Algorithm 2. Furthermore, Algorithm 2 clearly outperforms the projection-free algorithm proposed in Garber and Kretzu [2024] as demonstrated by the plots in Figure 2 and 3. This is not surprising as the theoretical guarantees of Garber and Kretzu [2024] are worse than ours. However, the performance of the algorithm proposed in Wang et al. [2025] may come as a surprise since it had comparable theoretical guarantees as ours and better than those in Garber and Kretzu [2024]. As discussed previously, this is because the method in the paper relies on the impractical doubling trick. If one looks closely at the nature of the plot, it becomes clear that while the curves of the other algorithms reach a flat region, the curve corresponding to the algorithm in Wang et al. [2025] has a wavy nature and never really seems to reach a flat portion. This is due to the restarting technique that they employ where an the algorithm has to discard the information it has upto that point and start from

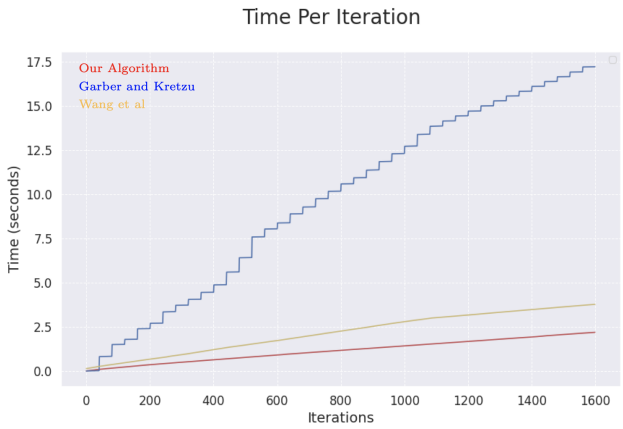


Figure 3: Comparison of execution times of the proposed Algorithm 2 and Algorithm 3 of Garber and Kretzu [2024] and also the algorithm proposed by Wang et al. [2025]. The run-time for Sinha and Vaze [2024]'s algorithm is two orders of magnitude larger (more than 1000s) and hence, is not shown on the figure.

scratch. Our observations match those in Besson and Kaufmann [2018] where they evaluated several versions of the doubling trick and concluded that performance is substantially worse than that of an intrinsically anytime algorithm, even one with less favorable theoretical guarantees. Moreover, this performance gap becomes larger each time the algorithm is restarted.

The projection-free algorithms mostly have similar computational efficiencies. The algorithm by Garber and Kretzu [2024] has slightly worse computational efficiency since at one point it has to do convex optimization over a ball besides using LOO calls. However, compared to the algorithm by Sinha and Vaze [2024] which has to employ extremely computationally expensive steps, all three projection-free algorithms are much more efficient.

### F.1 Data collection

Our data collection is done using public probes from RIPE Atlas, a global network measurement platform. Specifically, we collect HTTPS data that includes bandwidth and latency measurements between different network nodes. This data is gathered from a public probe, with the measurements taken between 20:30 and 21:30 on December 14th, 2024, and saved in the standard JSON format. The file contains fields such as IP addresses, timestamps, and corresponding latency and bandwidth values.

Once the data is collected, we proceed to construct a network graph. The graph is centered around a hub node, which connects to other nodes based on the latency and bandwidth data obtained from RIPE Atlas. Additional edges are then randomly added between other nodes, with the latency of each edge being the sum of the individual latencies to the central hub. The bandwidth of an edge is determined by the minimum bandwidth of the two nodes involved in the connection. This creates a network where each edge represents a connection between two nodes, characterized by their latency and bandwidth values.

To simulate real-world fluctuations and enhance the dataset, we introduce random scaling factors. Specifically, the latency values are scaled randomly between 0.5x and 1.5x, while the bandwidth values are scaled between 0.8x and 1.2x for each iteration. This variability mimics the dynamic nature of communication networks, ensuring that the dataset reflects more realistic network conditions. The scaled latency and bandwidth values are then used to construct latency and bandwidth matrices, which serve as the basis for the performance evaluation of the algorithms.