Optimal regret algorithm for Pseudo-1d Bandit Convex Optimization

Aadirupa Saha [1] **Nagarajan Natarajan** [2] **Praneeth Netrapalli** [2 3] **Prateek Jain** [2 3]

Abstract

We study online learning with bandit feedback (i.e. learner has access to only zeroth-order oracle) where cost/reward functions f_t admit a "pseudo-1d" structure, i.e. $f_t(\mathbf{w}) = \ell_t(g_t(\mathbf{w}))$ where the output of g_t is one-dimensional. At each round, the learner observes context $\mathbf{x}_t$, plays prediction $g_t(\mathbf{w}_t; \mathbf{x}_t)$ (e.g. $g_t(\cdot) = \langle \mathbf{x}_t, \cdot \rangle$) for some $\mathbf{w}_t \in \mathbb{R}^d$ and observes loss $\ell_t(g_t(\mathbf{w}_t))$ where ℓ_t is a convex Lipschitz-continuous function. The goal is to minimize the standard regret metric. This pseudo-1d bandit convex optimization problem (PBCO) arises frequently in domains such as online decision-making or parameter-tuning in large systems. For this problem, we first show a lower bound of $\min(\sqrt{dT}, T^{3/4})$ for the regret of any algorithm, where T is the number of rounds. We propose a new algorithm OPTPBCO that combines randomized online gradient descent with a kernelized exponential weights method to exploit the pseudo-1d structure effectively, guaranteeing the *optimal* regret bound mentioned above, up to additional logarithmic factors. In contrast, applying state-of-the-art online convex optimization methods leads to $\tilde{O}\left(\min\left(d^{9.5}\sqrt{T}, \sqrt{d}T^{3/4}\right)\right)$ regret, that is significantly suboptimal in d.

1. Introduction

Online learning with bandit feedback is a cornerstone problem in the online learning literature and can be used to model a variety of practical systems where at each step t, the system takes an action $\mathbf{w}_t \in \mathbb{R}^d$ for which it incurs a loss of $f_t(\mathbf{w}_t)$. Now, often times in practice, the action space has significantly more structure. For example, in large-scale parameter tuning the reward/loss is computed on a *scalar* parameter predicted by an underlying ML model applied to the current context of system. That is, the problem has a "pseudo-1d" structure in the loss functions $f_t(\mathbf{w}) = \ell_t(g_t(\mathbf{w}; \mathbf{x}_t))$ where $g_t : \mathbb{R}^d \to \mathbb{R}$ is a one-dimensional function.

We formulate the Pseudo-1d Bandit Convex Optimization (in Section 2) as follows: given a data point, or context, $\mathbf{x}_t \in \mathcal{X}$ at round t, the prediction of the learner is given by $g_t(\mathbf{w}_t; \mathbf{x}_t)$ for some $\mathbf{w}_t \in \mathcal{W} \subseteq \mathbb{R}^d$ and *known* g_t, e.g. $g_t(\mathbf{w}_t; \mathbf{x}_t) = \langle \mathbf{w}_t, \mathbf{x}_t \rangle$. The learner then receives $\ell_t(g_t(\mathbf{w}_t; \mathbf{x}_t))$ from the adversary for some unknown convex, Lipschitz-continuous loss ℓ_t. The goal is to minimize regret, i.e. the excess cumulative loss suffered by the learner over the best, fixed, parameter $\mathbf{w}^* \in \mathcal{W}$ in hindsight. As mentioned above, the pseudo-1d structure arises naturally in online parameter tuning/decision making where the goal is to learn the optimal parameters $\mathbf{w}$ that govern the system, which can be very high-dimensional, but the dynamic reward ℓ_t depends only on a one-dimensional action g_t taken by the system based on parameters $\mathbf{w}$ and the observed context $\mathbf{x}_t$.

A concrete application that motivates our work is in the domain of large scale distributed services where programmers are required to set configuration parameters of services using hand-written heuristics for some control or decision logic; this is indeed an important research question at the intersection of Programming Languages and ML, see Natarajan et al. (2020). For example, a programmer wants to decide the minimum amount of RAM required for a service, which in turn depends on other configuration parameters such as the number of threads running on the VM (x_1) and the number of users served (x_2), say, via $g_t(\mathbf{w}; \mathbf{x}) = w_1 x_1 + w_2 x_2$. The ultimate validation of the choices of w_1 and w_2 is by observing overall failures and system throughput, which is the reward ℓ_t obtained in the bandit sense. This is an instance of the pseudo-1d problem, and arises as part of tuning configuration settings of large-scale services in Microsoft.

The problem is a special case of the standard bandit convex optimization for which the state-of-the-art methods have regret of $O(d^{9.5}\sqrt{T})$ (Bubeck et al., 2017) or $O(\sqrt{d}T^{3/4})$ (Flaxman et al., 2005). So, the key question we answer in this paper is if and when the pseudo-1d structure can help obtain learning algorithms with better sample complexity or regret guarantees. For example, can we design an algorithm that has the optimal $\sqrt{T}$ regret in terms of T, but its

[1]Microsoft Research, New York City [2]Microsoft Research, India [3]The authors are currently at Google Research, India. Correspondence to: Aadirupa Saha <aadirupa.saha@microsoft.com>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

regret is completely independent of d? Note that in the full-information setting, i.e., when full access to ℓ_t is available, the standard Online Gradient Descent (OGD) (Zinkevich, 2003) indeed obtains the optimal $\sqrt{T}$ regret independent of d.

Somewhat surprisingly, our first result (Theorem 1 in Section 3) shows that, even though the problem has a pseudo-1d structure, in the worst case, any algorithm will suffer a regret of $\min(\sqrt{dT}, T^{3/4})$ after T rounds. That is, for large T, optimal regret has to scale with d.

So, the next natural question is if we can design an algorithm to achieve the optimal regret. We answer that question in affirmative (Theorem 2 in Section 4) by designing an efficient algorithm that indeed achieves the optimal regret when the loss function ℓ_t is convex and Lipschitz. Our method critically utilizes the pseudo-1d structure to define the algorithm in two regimes: a) for $d \geq \sqrt{T}$, we present a modification of the randomized gradient descent method by Flaxman et al. (2005) to get the rate optimal in this regime, b) for $d \leq \sqrt{T}$ we exploit a kernelized exponential weighting scheme similar to that of Bubeck et al. (2017) to again obtain the optimal rate in this regime. A key contribution of our work is that exploiting the problem structure also greatly simplifies the analysis and the proofs become significantly clearer (presented in Section 4, Lemma 5), and much more palatable, than the general d-dimensional analysis by Bubeck et al. (2017).

Now, it is instructive to compare our results against those of contextual bandit (CB) algorithms as the high level goal of both the formulations is similar. But, there are certain key distinctions between the two formulations. CB formulations work with general loss/reward functions while we restrict our methods to *convex Lipschitz* functions only. On the other hand, CB methods are designed in general for discrete action and policy space (see Remark 3 in Section 2) unlike pseudo-1d bandit formulation that handles continuous prediction/action space and infinite policy space.

Our solution enables modeling and solving pseudo-1d problems arising in practice (like the parameter tuning example cited in the beginning of this section) automatically with small sample complexity. Libraries like Vowpal Wabbit are extensively used by practitioners for such problems (characterized by Algorithm 1 in Bietti et al. (2018)) and there are frameworks used at enterprise scale based on bandit formulations (Agarwal et al., 2016), but their applicability to the general setting is limited.

We present simulations in Section 5 that demonstrate the regret bounds on simple synthetic problems. Our contributions are summarized below:
1) A novel problem formulation that captures practical online learning scenarios with bandit feedback and structure in the reward/loss function.
2) A lower bound for the pseudo-1d bandit convex optimization problem – in the worst case, any learning strategy suffers a regret of $O(\min(\sqrt{dT}, T^{3/4}))$.
3) A learning algorithm that is provably optimal, assuming the loss functions are convex and Lipschitz — with a regret bound that matches the lower bound up to logarithmic factors.

Related Work. Flaxman et al. (2005) initiated the study of bandit optimization for general convex functions and showed a regret guarantee of $O(\sqrt{d}T^{5/6})$ using online gradient-descent; with additional assumption of Lipschitzness, they improve the bound to $O(\sqrt{d}T^{3/4})$, and recently (Hazan & Li, 2016) and (Bubeck et al., 2017) showed $\sqrt{T}$-regret (optimal in terms of T, but highly suboptimal in terms of d) using two different types of algorithms. Due to the fundamental nature of the problem, there is a long line of work in this space (Bubeck & Eldan, 2016; Chen et al., 2018; Sahu et al., 2018), that look at certain types of losses (e.g. linear losses) (Abernethy et al., 2009; Y. Abbasi-Yadkori & Szepesvari, 2011), different types of feedback (e.g. two-point feedback, as against one-point feeback in our work) (Agarwal et al., 2011; Shamir, 2017), or different settings (stochastic vs adversarial) where improved regret bounds are possible (Ghadimi & Lan, 2013; Shamir, 2013; Yang & Mohri, 2016; Saha & Tewari, 2011).

On the contrary, in the full information (online convex optimization) setting, where the gradient information of the loss function is known, Zinkevich (2003) showed that online gradient descent achieves a regret of $O(\sqrt{T})$ (which can be improved under additional assumptions (Hazan et al., 2007)). Wang et al. (2017) consider a composite structure of the loss function ($H(\cdot)$ in Eqn. (1) of their paper) similar to our pseudo-1d structure. However, they (a) work in the stochastic optimization setting, i.e., the goal is to design an algorithm that minimizes $f(\cdot) = \mathbb{E}[\ell(\mathbb{E}[g(\cdot)])]$ (rewritten using our notation), where $\mathbb{E}$ is expectation w.r.t. underlying stochasticity, and (b) assume a *first-order* gradient feedback model, i.e., they require access to (noisy) $\nabla\ell(.)$ (in addition to $\nabla g(.)$). Thus, their problem setup is strictly simpler than ours. Consequently, they are able to obtain $O(\sqrt{T})$ convergence when g is linear, and ℓ is smooth but possibly non-convex (they do not explicitly quantify the dependence on d or the pseudo-dimension). In contrast, our goal is to design an algorithm with bounded *regret* in the online setting when f_t's are adversarially chosen, while the learner has only *zeroth-order* oracle access for the losses ℓ_t.

Contextual bandit learning has a vast literature and results focusing on finite/discrete action spaces (survey by Bubeck et al. (2012)). The state-of-the-art results for continuous action spaces (i.e. at each round, the learner receives context

$\mathbf{x}_t$ and plays a value from $[0, 1]$) is due to Krishnamurthy et al. (2019); Majzoubi et al. (2020); here, they work with a notion of "smoothed" regret, where each action is mapped to a smoothed action, and the learner also competes with a smoothed policy class (that maps context to action, akin to g_t). One key difference in the bandit learning literature is that typically there is no (or mild) assumption on the loss/reward function (See Remark 3).

2. Problem Setup and Preliminaries

The standard online (bandit) convex optimization framework proceeds in rounds: at round t, the learner *plays* $\mathbf{w}_t \in \mathcal{W} \subseteq \mathbb{R}^d$ and receives the incurred loss $f_t(\mathbf{w}_t)$ as feedback, for some convex f_t chosen adversarially. The "action space" $\mathcal{W}$ is restricted to be a closed convex set with diameter $W = \max_{\mathbf{w},\mathbf{w}' \in \mathcal{W}} \|\mathbf{w} - \mathbf{w}'\|_2$. The goal of the (possibly randomized) learner $\mathcal{A}$ is to have a bounded regret compared to a fixed $\mathbf{w}^* \in \mathcal{W}$ in hindsight that achieves the least cumulative loss, i.e. to minimize the regret defined as:

$$\mathcal{R}_T(\mathcal{A}) = \sum_{t=1}^{T} \mathbb{E}\big[f_t(\mathbf{w}_t)\big] - \sum_{t=1}^{T} f_t(\mathbf{w}^*), \tag{1}$$

where $\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w})$, and $\mathbb{E}[.]$ is wrt to any randomness in $\mathcal{A}$. In our formulation, at each round, the learner receives context $\mathbf{x}_t \in \mathcal{X}$, chooses parameters $\mathbf{w}_t$ and plays its prediction $g_t(\mathbf{w}_t; \mathbf{x}_t)$, and receives loss for this prediction; the loss functions chosen by the adversary at each round satisfies:

$$f_t(\mathbf{w}_t) = \ell_t(g_t(\mathbf{w}_t; \mathbf{x}_t)), \tag{2}$$

for some $g_t : \mathcal{W} \times \mathcal{X} \to \mathcal{G} \subseteq \mathbb{R}$, and bounded convex and L-Lipschitz $\ell_t : \mathcal{G} \to [0, C]$. Note that while the learner receives bandit feedback for ℓ_t, it has complete knowledge of g_t, for example, $g_t(\cdot) = \langle \cdot, \mathbf{x}_t \rangle$. Thus, in particular, the learner has access to both zeroth- and first-order information for g_t but only zeroth-order information for ℓ_t. We refer to $\mathcal{G}$ as the prediction space. With this set up, we formally state the problem of interest below.

Pseudo-1d Bandit Convex Optimization (PBCO): Minimize (1) where the functions $f_t(.)$ admit the structure in (2), and $\ell_t, \mathbf{x}_t$ are chosen adversarially.

Remark 1. *Note that the goal is to minimize cumulative regret* (1) *with respect to the best fixed d-dimensional parameter $\mathbf{w}^*$, though the learner plays in the prediction space $\mathcal{G}$ which is one-dimensional.*

Remark 2 (Applying bandit convex optimization)**.** *Ignoring the structure in* (2)*, one can apply bandit convex optimization algorithms to* PBCO *problem. The state-of-the-art result for online convex optimization with bandit feedback is by Bubeck et al. (2017); using their algorithm gives a significantly sub-optimal regret bound of $O(d^{9.5}\sqrt{T})$.*

Remark 3 (Applying continuous contextual bandits)**.** *The recent work by Krishnamurthy et al. (2019) provides optimal guarantees for contextual bandits with continuous actions (i.e. the learner plays an action from $[0, 1]$ at each round). Applying their algorithm to our setting yields a "smoothed" regret (which is a weaker notion of regret, and not directly comparable to ours) of $O(T^{2/3}(Ld)^{1/3})$, where L is Lipschitz constant of ℓ_t. Note, however, that their guarantees apply to general losses and in particular do* not *need convexity.*

In the (easier) setting of (bandit) stochastic convex optimization, there is a fixed unknown $f(.)$ for which the learner obtains noisy evaluations. The goal is to minimize the expected value of the function, i.e., to bound:

$$\bar{\mathcal{R}}(\mathcal{A}) := \min_{\mathbf{w}\in\mathcal{W}} \mathbb{E}_Z[f(\mathbf{w}; Z) - f(\mathbf{w}^*; Z)], \tag{3}$$

where $\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathcal{W}} \mathbb{E}_Z[f(\mathbf{w}; Z)]$. Naturally, we can pose a stochastic version of the PBCO problem where f admits the pseudo-1d structure.

Notation. Let $[n] = \{1, 2, \dots n\}$, for any $n \in \mathbb{N}$. For any $\delta > 0$, let $\mathcal{B}_d(\delta)$ and $\mathcal{S}_d(\delta)$ denote the ball and the surface of the sphere of radius δ in d dimensions respectively. Lower case bold letters denote vectors, upper case bold letters denote matrices. $\mathbf{P}_{\mathcal{X},\|\cdot\|}(\mathbf{x})$ denotes the nearest point projection of a point $\mathbf{x} \in \mathbb{R}^d$ on to set $\mathcal{X} \subseteq \mathbb{R}^d$ with respect to norm $\|\cdot\|$, i.e. $\mathbf{P}_{\mathcal{X}}(\mathbf{x}) := \arg\min_{\mathbf{z}\in\mathcal{X}} \|\mathbf{x} - \mathbf{z}\|$. For any vector $\mathbf{x} \in \mathbb{R}^d$, $\|\mathbf{x}\|_2$ denotes the ℓ_2 norm of vector $\mathbf{x}$. To be consistent with the literature, we will use f_t as a short-hand for $\ell_t(g_t(.))$ in this paper (as defined in (2)); and use $g_t(\mathbf{w})$ as a short-hand for $g_t(\mathbf{w}; \mathbf{x}_t)$ when $\mathbf{x}_t$ is implicit from the context.

Below we give definitions that will be used in the remainder of the paper.

(A1) Convexity: For all $\mathbf{w}_1, \mathbf{w}_2 \in \mathcal{W}$ and $\lambda \in [0, 1]$,
(i) $\ell_t\big(\lambda g_t(\mathbf{w}_1) + (1-\lambda) g_t(\mathbf{w}_2)\big) \le \lambda \ell_t(g_t(\mathbf{w}_1)) + (1-\lambda)\ell_t(g_t(\mathbf{w}_2))$
(ii) $f_t\big(\lambda \mathbf{w}_1 + (1-\lambda)\mathbf{w}_2\big) \le \lambda f_t(\mathbf{w}_1) + (1-\lambda) f_t(\mathbf{w}_2)$.

(A2) L-Lipschitzness: For all $\mathbf{w}_1, \mathbf{w}_2 \in \mathcal{W}$, $\|\ell_t(g_t(\mathbf{w}_1)) - \ell_t(g_t(\mathbf{w}_2))\|_2 \le L\|g_t(\mathbf{w}_1) - g_t(\mathbf{w}_2)\|_2$.

While we require the loss function to be convex, the learner can choose any bounded prediction function as stated below.

(A3) Boundedness of g_t: (i) $g_t \in \mathcal{G} = [\alpha_{\mathcal{W}}, \beta_{\mathcal{W}}] \subseteq \mathbb{R}$, **(ii)** $\|\nabla_{\mathbf{w}} g_t(\mathbf{w}; \mathbf{x})\| \le D$, for all $\mathbf{x} \in \mathcal{X}, \mathbf{w} \in \mathcal{W}$. Note A3(ii) implies g_t is D-Lipschitz.

Remark 4. *Note that when g_t is linear, i.e. $g_t(\mathbf{w}; \mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$, then the above assumptions simplify: In particular, (a)* **(A1) (i)** $\iff$ **(A1) (ii)***, (b) $\|\nabla_{\mathbf{w}} g_t(\mathbf{w}; \mathbf{x})\| = \|\mathbf{x}\| \le D$, where D denotes the diameter of $\mathcal{X}$, and $g_t \in [-DW, DW]$, where W is the diameter of $\mathcal{W}$.*

All detailed proofs are provided in the supplementary (Appendix A).

3. A lower bound for PBCO

It does appear that the PBCO problem introduced in Section 2 is effectively a one-dimensional problem because the loss function ℓ_t is computed on a scalar. This raises the natural question as to when and if one can get rid of dimension dependence in the regret. Recall that existing bandit convex optimization techniques (Remark 2 in Section 2) do suffer poly(d) dependence. In the following we show that, in general, one cannot avoid the dependence on d, and in particular, we show a lower bound that is $\Omega(\sqrt{dT})$, in the regime $d = O(\sqrt{T})$. For larger d, any algorithm must suffer a regret that is $\Omega(T^{3/4})$.

Theorem 1 (Lower bound for PBCO)**.** *For any algorithm $\mathcal{A}$ for the* PBCO *problem, there exists $\mathcal{W} \subseteq \mathcal{B}_d(1)$, and sequence of loss functions $f_1, \dots f_T : \mathcal{W} \mapsto \mathbb{R}$ where for any t, $\mathbb{E}[f_t(\cdot)] \in [0,1]$, the expected regret suffered by $\mathcal{A}$ satisfies:*

$$\mathbb{E}\big[R_T(\mathcal{A})\big] = \mathbb{E}\Bigg[\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w})\Bigg] \geq \frac{1}{32}\min\left(\sqrt{dT}, T^{3/4}\right).$$

*In particular, the lower bound holds under the assumptions **(A1)**, **(A2)** and **(A3)**.*

Proof Sketch. We give a simple construction of problem instance to show the desired lower bound. We will work with linear model, i.e. $g_t(.) = \langle \mathbf{x}_t, \cdot \rangle$, and $\mathcal{W} = \frac{1}{\sqrt{d}}\{\pm 1\}^d$ which suffices for a lower bound. The idea is to divide the max rounds $[T]$ into d equal length sub intervals (each of length T/d) $T_1, \dots, T_d$ (let $T_0 = \emptyset$). Now, for $i \in [d]$, choose $\sigma_i \sim \text{Ber}(\pm 1)$, and set $\mathbf{x}_i = \mathbf{e}_i$. At round $t \in T_i = \left\{\frac{T}{d}(i-1)+1, \dots, \frac{T}{d}i\right\}$, $i \in [d]$, adversary chooses $\mathbf{x}_t = \mathbf{x}_i$ and the loss function $f_t(\mathbf{w}) = \mu\sigma_i(\mathbf{w}^\top \mathbf{x}_i) + \varepsilon_t$, where $\varepsilon_t \sim \mathcal{N}(0, \frac{1}{16})$, for some constant $\mu > 0, \forall \mathbf{w} \in \mathcal{W}$. For this problem instance, it is easy to show that $\mathbf{w}^* = -\frac{\boldsymbol{\sigma}}{\sqrt{d}} \in \mathcal{W}$, where $\boldsymbol{\sigma} = (\sigma_1, \dots, \sigma_d)$. The learner's goal is then to figure out $\boldsymbol{\sigma}$. Now, we argue a lower bound for two regimes:

Case $d \leq 16\sqrt{T}$**.** We can show that any learning strategy must suffer an expected regret of at least $\frac{\sqrt{dT}}{32}$ if we set $\mu = \frac{d}{16\sqrt{T}} < 1$ (used by the adversary for constructing $f_t(\mathbf{w})$ mentioned above).

Case $d > 16\sqrt{T}$**.** One can use an embedding trick, and simply ignore the $d - 16\sqrt{T}$ dimensions. In this setup, we can argue that any learner must suffer a regret of at least $\frac{T^{3/4}}{32}$ by falling back on the first case.

Together, we get the desired lower bound. See Appendix A for details.

Remark 5. *Note that in the lower bound instance of Theorem 1, ℓ_t and $\mathbf{x}_t$ are dependent random variables. In fact, this dependence is crucial for obtaining a lower bound that depends on the dimension d. It is indeed possible to design an algorithm that achieves $\tilde{O}\left(\sqrt{T}\right)$ regret for the stochastic setting where ℓ_t is independent of $\mathbf{x}_t$. The main idea is this: all one needs to estimate is the minimizer of the one-dimensional function $\mathbb{E}[\ell_t]$. However, this situation does not seem to be of much interest and hence we do not provide a proof of this claim.*

4. An optimal algorithm for PBCO

In this section, we develop a method for the PBCO problem in the adversarial setting, and show that it achieves a regret that matches the lower bound presented in Section 3, up to logarithmic factors. The proposed solution operates in two regimes, mirroring the lower bound analysis: in one regime, when $d = O(\sqrt{T})$, it relies on a kernelized exponential weights scheme, and in the other regime, when d is larger, it relies on an online gradient descent style algorithm. This method, called OPTPBCO, is presented in Algorithm 1.

Algorithm 1 OPTPBCO

1: **Input:**
2: max rounds T, dimensionality d
3: **if** $d \leq \frac{WLD\sqrt{T}}{C\log(L'T)}$ **then**
4: Run Kernelized Exponential Weights for PBCO (Algorithm 2) with $\eta = \frac{\sqrt{d\log(L'T)}}{C\sqrt{BT}}, T$
5: **else**
6: Run Online Gradient Descent for PBCO (Algorithm 3) with $\eta = \frac{W\delta}{DC\sqrt{T}}, \delta = \left(\frac{WDC}{3L\sqrt{T}}\right)^{1/2}, \alpha = \delta$, and T
7: **end if**

We now state our second key result of the paper — OPTPBCO achieves an optimal regret bound given below.

Theorem 2 (Regret bound for OPTPBCO (Algorithm 1))**.** *If the loss functions $\ell_t : \mathcal{G} \to [0, C]$, f_t satisfy* **(A1)**, **(A2)**, **(A3)**, *$\mathcal{W} = \mathcal{B}_d(W)$, the expected regret of the* PBCO *learner presented in Algorithm 1 can be bounded as:*

$$\mathbb{E}[\mathcal{R}_T(\mathcal{A}_{\text{OPTPBCO}})] \leq 2\sqrt{2}\min\left(C\sqrt{dT\log(L'T)}, \sqrt{WLDC}T^{3/4}\right)$$

where $L' = LDW$ and the expectation $\mathbb{E}[\cdot]$ is with respect to the algorithm's randomization.

Proof. The bound follows from Lemmas 5 and 7, the choice of parameters given in steps 4 and 6 of Algorithm 1, and

noticing that when d is larger than the threshold in step 3 of the Algorithm, OGD (Algorithm 3) achieves a smaller regret than Kernelized Exponential Weights (Algorithm 2). □

Corollary 3. *When g_t is linear, i.e. $g_t(.) = \langle \mathbf{x}_t, \cdot \rangle$, then D is the diameter of $\mathcal{X}$.*

A few remarks are in order.

Remark 6. OPTPBCO *requires the knowledge of the Lipschitz constant L (e.g. in Step 3) of unknown loss ℓ_t. This is a standard assumption made in the bandit convex optimization literature (Flaxman et al., 2005).*

Remark 7. *It is straight-forward to state a result similar to Theorem 2 for the stochastic version of the* PBCO *problem.*

4.1. Regime $d = \widetilde{O}(\sqrt{T})$: Kernelized Exp. Weights

The key idea in our approach is to use a kernelized exponential weights scheme that exploits the pseudo-1d structure in the loss function. Exponential weights is a popular online learning algorithm for contextual bandits. Recently (Bubeck et al., 2017) developed a meticulous kernel method that uses exponential weight update at its core to prove $O(\sqrt{T})$ regret for general convex (and Lipschitz) functions. Their approach hinges on using a smoothing operator (kernel) to obtain an estimator of the loss function f_t (the analogous estimator is fairly straight-forward in the multi-arm bandit setting) in the bandit convex optimization setting.

In the general d-dimensional setting, defining a kernel such that the resulting estimator of f_t is both (almost) unbiased and has bounded variance turns out to be extremely complicated and incurs large polynomial factors in dimension d. But, we can exploit the pseudo-1d structure in our setting to define a relatively simple kernel in the *one-dimensional* prediction space instead. A key benefit of using the simple 1-d kernel is that much of the analysis in (Bubeck et al., 2017) can be greatly simplified, and the proofs become significantly easier to follow.

Before describing the main ideas of the algorithm, we need some notation and definitions set up. Let $\mathbf{p}_t$ denote the distribution over parameters $\mathcal{W}$ maintained by the learner at round t. Also let $\mathcal{G}_t := \{g_t(\mathbf{w}, \mathbf{x}_t) \mid \mathbf{w} \in \mathcal{W}\} \subseteq \mathbb{R}$, for any $t \in [T]$, and $\mathcal{W}_t(y) := \{\mathbf{w} \in \mathcal{W} \mid g_t(\mathbf{w}, \mathbf{x}_t) = y\}$, for $y \in \mathcal{G}_t$. Given this, we obtain a one dimensional distribution $\mathbf{q}_t \in \mathcal{Q}_t$ over $\mathcal{G}_t$ from $\mathbf{p}_t$ as follows: $d\mathbf{q}_t(y) := \int_{\mathcal{W}_t(y)} d\mathbf{p}_t(\mathbf{w}), \forall y \in \mathcal{G}_t$.

The kernelized exponential weights scheme crucially uses a kernel map to obtain a smooth estimate of the loss function on the action space based on a single point evaluation. The key observation we make is that, in our setting, it suffices to define such a kernel over the *scalar prediction* space than over the d-dimensional action space as in (Bubeck et al., 2017). This 1-dimensional kernel map, denoted $\mathbf{K}'_t$, is carefully constructed at each round t based on $\mathbf{q}_t$ and the observed context $\mathbf{x}_t$ as given below:

Definition 4 (Bubeck et al. (2017))**.** *Given a distribution $\mathbf{q}_t$ over $\mathcal{G}_t$, and $\epsilon > 0$, we define a one-dimension kernel $\boldsymbol{K}'_t : \mathcal{G}_t \times \mathcal{G}_t \mapsto \mathbb{R}_+$ as:*

$$\mathbf{K}'_t(y, y') = \begin{cases} \dfrac{I\big(y \in [y', \bar{y}]\big)}{|y' - \bar{y}|}, & \text{if } |y' - \bar{y}| \geq \epsilon, \\ \dfrac{I\big(y \in [\bar{y} - \epsilon, \bar{y}]\big)}{\epsilon}, & \text{when } y' \in [\bar{y} - \epsilon, \bar{y} + \epsilon] \end{cases},$$

where $\bar{y} := \mathbb{E}_{\mathbf{q}_t}[y]$, and $I(.)$ is the indicator function.

For the kernel $\mathbf{K}'_t$ defined above, we can verify that $\int_{\mathcal{G}_t} \mathbf{K}'_t(y, y')dy = 1$ for every $y' \in \mathcal{G}_t$. Further we define a linear operator on any $\mathbf{q} \in \mathcal{Q}_t$ (a smoothing of $\mathbf{q}$ w.r.t. $\mathbf{K}'_t$) as:

$$\mathbf{K}'_t\mathbf{q}(y) := \int_{y' \in \mathcal{G}_t} \mathbf{K}'_t(y, y')d\mathbf{q}(y') \quad \forall y \in \mathcal{G}_t. \tag{4}$$

This operator is particularly useful because for any valid probability measure $\mathbf{q} \in \mathcal{Q}_t$, the map $\mathbf{K}'_t\mathbf{q}$ also defines a valid probability distribution over $\mathcal{G}_t$ (a precise statement is proved in Lem. 8, Appendix A.2).

Even though we leverage the (above) definition of the 1-d kernel from Bubeck et al. (2017), the crucial difference in our analysis is that while they optimize the regret over just a scalar parameter (in their 1-d case), we leverage the same 1-d kernel for learning d-dimensional parameters, exploiting the pseudo-1d structure. Consequently, our algorithm design and the corresponding regret analysis is different than Bubeck et al. (2017), as outlined next.

Algorithm (main ideas). We start with maintaining uniform weight over the $\mathcal{W}$: $\mathbf{p}_1 \leftarrow \frac{1}{\text{vol}(\mathcal{W})}$. At any time $t \in [T]$, upon receiving $\mathbf{x}_t$, we first compute the effective scalar decision space $\mathcal{G}_t$ and sample a $y_t \in \mathcal{G}_t$ according to the smoothed distribution of $\mathbf{K}'_t\mathbf{q}_t$. However, since the task is to choose a prediction point from the d-dimensional space $\mathcal{W}$, we pick any (uniformly) random $\mathbf{w}_t$ that maps to y_t, i.e. $\mathbf{w}_t \in \mathcal{W}_t(y_t)$ uniformly at random (Line 7 in Algorithm 2). Upon receiving the zeroth-order feedback $f_t(\mathbf{w}_t)$, we estimate the loss at each point $\mathbf{w} \in \mathcal{W}$ as follows:

$$\tilde{f}_t(\mathbf{w}) \leftarrow \frac{f_t(\mathbf{w}_t)}{\mathbf{K}'_t\mathbf{q}_t(y_t)}\mathbf{K}'_t(y_t, y), \ \ \forall \mathbf{w} \in \mathcal{W}.$$

Note the above loss estimate $\tilde{f}_t$ ensures for a fixed $y \in \mathcal{G}_t$, $\tilde{f}_t(\mathbf{w})$ is same for all $\mathbf{w} \in \mathcal{W}_t(y)$ (as justified by the structure: $f_t(\cdot) = \ell_t(g_t(\cdot))$). Finally, using the (estimated) loss $\tilde{f} : \mathcal{W} \mapsto \mathbb{R}$, we update $\mathbf{p}_t$ identical to the standard exponential weights algorithm:

$$\mathbf{p}_{t+1}(\mathbf{w}) \leftarrow \frac{\mathbf{p}_t(\mathbf{w}) \exp\big(-\eta\tilde{f}_t(\mathbf{w})\big)}{\int_{\tilde{\mathbf{w}}} \mathbf{p}_t(\tilde{\mathbf{w}}) \exp\big(-\eta\tilde{f}_t(\tilde{\mathbf{w}})\big)d\tilde{\mathbf{w}}}, \ \ \forall \mathbf{w} \in \mathcal{W}.$$

Algorithm 2 summarizes the proposed kernelized exponential weights scheme for PBCO.

Algorithm 2 Kernelized Exponential Weights for PBCO

Input: learning rate: $\eta > 0$, $\epsilon > 0$, max rounds T.
Initialize: $\mathbf{w}_1 \leftarrow \mathbf{0}, \mathbf{p}_1 \leftarrow \frac{1}{\text{vol}(\mathcal{W})}$.
for $t = 1, 2, \cdots T$ **do**
  Receive $\mathbf{x}_t$, and define $\mathcal{G}_t := \{g_t(\mathbf{w}, \mathbf{x}_t) \mid \mathbf{w} \in \mathcal{W}\} \subseteq \mathbb{R}$
  Define $\mathbf{q}_t$ such that $d\mathbf{q}_t(y) := \int_{\mathcal{W}_t(y)} d\mathbf{p}_t(\mathbf{w}), \forall y \in \mathcal{G}_t$, where $\mathcal{W}_t(y) := \{\mathbf{w} \in \mathcal{W} \mid g_t(\mathbf{w}, \mathbf{x}_t) = y\}$
  Using $\mathbf{x}_t$ and $\mathbf{q}_t$, and given ϵ, define kernel $\mathbf{K}'_t : \mathcal{G}_t \times \mathcal{G}_t \mapsto \mathbb{R}$ (according to Definition 4)
  Sample $y_t \sim \mathbf{K}'_t\mathbf{q}_t$ and pick any $\mathbf{w}_t \in \mathcal{W}_t(y_t)$ uniformly at random
  Play $g_t(\mathbf{w}_t; \mathbf{x}_t)$ and receive loss $f_t(\mathbf{w}_t) = \ell_t(g_t(\mathbf{w}_t; \mathbf{x}_t))$
  $\tilde{f}_t(\mathbf{w}) \leftarrow \frac{f_t(\mathbf{w}_t)}{\mathbf{K}'_t\mathbf{q}_t(y_t)}\mathbf{K}'_t(y_t, y)$, for all $\mathbf{w} \in \mathcal{W}(y), \forall y \in \mathcal{G}_t$ ▷ estimator of f_t
  $\mathbf{p}_{t+1}(\mathbf{w}) \leftarrow \frac{\mathbf{p}_t(\mathbf{w}) \exp\big(-\eta \tilde{f}_t(\mathbf{w})\big)}{\int_{\tilde{\mathbf{w}}} \mathbf{p}_t(\tilde{\mathbf{w}}) \exp\big(-\eta \tilde{f}_t(\tilde{\mathbf{w}})\big) d\tilde{\mathbf{w}}}$, for all $\mathbf{w} \in \mathcal{W}$
end for

We show in the following Lemma that the regret bound for Algorithm 2 is bounded by $\tilde{O}(\sqrt{dT})$. Exploiting the problem structure gets us significantly improved dependence on d compared to the original result by Bubeck et al. (2017) for the general case (as stated in Remark 2).

Lemma 5 (Regret bound for Algorithm 2)**.** *If the losses $\ell_t : \mathcal{G} \to [0, C]$ and g_t, $t \in [T]$ satisfy* **(A1) (i)**, **(A2)**, *and* **(A3)**, *then for the choice of $\{\boldsymbol{K}_t\}_{t\in[T]}$ as defined in Definition 4, the expected regret of Algorithm 2, with learning rate $\eta = \left(\frac{2d\log(L'T)}{BC^2T}\right)^{\frac{1}{2}}$ and $\epsilon = \frac{1}{3LT}$, can be bounded as:*

$$\mathbb{E}[\mathcal{R}_T] \leq 4 + 2\sqrt{2}\left(\sqrt{dBC^2T\log(L'T)}\right) = O\left(C\sqrt{dT\log(L'T)}\right)$$

where $B = 2\Big(1+\ln(3LT)+\ln\left(\beta_{\mathcal{W}}-\alpha_{\mathcal{W}}\right)\Big)$, $L' = LDW$, $W = Diam(\mathcal{W})$ and the expectation $\mathbb{E}[\cdot]$ is with respect to the algorithm's randomization.

Proof sketch. Detailed proof (and supporting lemmas) is presented in Appendix A. Here, we sketch all its key constituents. The proof relies on key properties of the aforementioned 1-d kernel map, shown in Lemma 11. We start by analyzing the expected regret w.r.t. the optimal point $\mathbf{w}^* \in \mathcal{W}$ (denote $y_t^* = g_t(\mathbf{w}^*)$ for all $t \in [T]$). Define $\forall y \in \mathcal{G}_t$, $\tilde{\ell}_t(y) := \tilde{f}_t(\mathbf{w})$, for any $\mathbf{w} \in \mathcal{W}(y)$. Also let $\mathcal{H}_t = \boldsymbol{\sigma}\big(\{\mathbf{x}_\tau, \mathbf{p}_\tau, \mathbf{w}_\tau, f_\tau\}_{\tau=1}^{t-1} \cup \{\mathbf{x}_t, \mathbf{p}_t\}\big)$ denote the sigma algebra generated by the history till time t. Then the expected cumulative regret of Algorithm 2 over T time steps can be bounded as:

$$\begin{aligned}
\mathbb{E}[R_T(\mathbf{w}^*)] &:= \mathbb{E}\left[\sum_{t=1}^{T}\Big(\ell_t(g_t(\mathbf{w}_t;\mathbf{x}_t)) - \ell_t(g_t(\mathbf{w}^*;\mathbf{x}_t))\Big)\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{T}\Big(\ell_t(y_t) - \ell_t(y_t^*)\Big)\right] = \mathbb{E}\left[\sum_{t=1}^{T}\left\langle \mathbf{K}'_t\mathbf{q}_t - \boldsymbol{\delta}_{y_t^*}, \ell_t\right\rangle\right] \\
&\leq 6\epsilon LT + 2\sum_{t=1}^{T}\mathbb{E}\left[\left\langle \mathbf{K}'_t(\mathbf{q}_t - \boldsymbol{\delta}_{y_t^*}), \ell_t\right\rangle\right] \\
&\overset{(a)}{=} 6\epsilon LT + 2\sum_{t=1}^{T}\mathbb{E}\left[\sum_{t=1}^{T}\mathbb{E}_{y_t\sim\mathbf{K}'_t\mathbf{q}_t}\left[\left\langle \mathbf{q}_t - \boldsymbol{\delta}_{y_t^*}, \tilde{\ell}_t\right\rangle \mid \mathcal{H}_t\right]\right] \\
&= 6\epsilon LT + 2\sum_{t=1}^{T}\mathbb{E}\left[\sum_{t=1}^{T}\mathbb{E}_{y_t\sim\mathbf{K}'_t\mathbf{q}_t}\left[\left\langle \mathbf{p}_t - \boldsymbol{\delta}_{\mathbf{w}^*}, \tilde{f}_t\right\rangle \mid \mathcal{H}_t\right]\right]
\end{aligned} \tag{5}$$

where the last equality follows by Lemma 9, and by $\left\langle \boldsymbol{\delta}_{\mathbf{w}^*}, \tilde{f}_t\right\rangle = \tilde{f}_t(\mathbf{w}^*) = \tilde{\ell}_t(y_t^*) = \left\langle \boldsymbol{\delta}_{y_t^*}, \tilde{\ell}_t\right\rangle$; (a) and the first inequality rely on the properties of the kernel in Lemma 11. Let us denote by $\mathbf{p}^*$ a uniform measure on the set $\mathcal{W}_\kappa := \{\mathbf{w} \mid \mathbf{w} = (1-\kappa)\mathbf{w}^* + \kappa\mathbf{w}',$ for any $\mathbf{w}' \in \mathcal{W}\}$ for some $\kappa \in (0, 1)$. We can then show that the inner expectation in (5) can be bounded by $\sum_{t=1}^{T}\mathbb{E}_{y_t\sim\mathbf{K}'_t\mathbf{q}_t}[\left\langle \mathbf{p}_t, \tilde{f}_t\right\rangle - \left\langle \mathbf{p}^*, \tilde{f}_t\right\rangle] + \kappa LDWT$ using the assumption that g_t is D-Lipschitz, and a certain adjoint operator on the kernel map is L-Lipschitz. The term $\sum_{t=1}^{T}\left\langle \mathbf{p}_t - \mathbf{p}^*, \tilde{f}_t\right\rangle$ can be bounded (via Lemma 10) by $\frac{KL(\mathbf{p}^*||\mathbf{p}_1)}{\eta} + \frac{\eta}{2}\left\langle \mathbf{q}_t, \tilde{\ell}_t^2\right\rangle$. Now, the second term $\mathbb{E}_{y_t\sim\mathbf{K}'_t\mathbf{q}_t}\left[\left\langle \mathbf{q}_t, \tilde{\ell}_t^2\right\rangle\right]$ relates to the variance of the loss estimator, and can be bounded by a constant, ensured by our choice of the 1d-kernel; and the first, KL divergence, term can be bounded by $d\log\frac{1}{\kappa}$ by the definition of $\mathbf{p}^*$. Plugging these bounds in (5), letting $L' = LDW$, and setting $\kappa = \frac{1}{L'T}$, $\epsilon = \frac{1}{3LT}$, (5) yields:

$$\mathbb{E}[R_T(\mathbf{w}^*)] \leq \quad O(1) + 2\left(\frac{d\log L'T}{\eta} + \frac{\eta BC^2T}{2}\right)$$

By choosing η to minimize the RHS above, the proof is complete.

We observe from Lemma 5 that when d is small and constant, the bound behaves like $\sqrt{T}$ but when d is large, say, $d = T^{2/3}$, the bound behaves like $T^{5/6}$. In what follows, we show that an online gradient descent style algorithm achieves a regret that scales as $T^{3/4}$ *independent of* d.

4.2. Larger d: Online Gradient Descent

Consider the standard online gradient descent algorithm of (Zinkevich, 2003), but with an estimator in lieu of the true gradient as in (Flaxman et al., 2005) to deal with bandit feedback. The key observation here is that we can perform the gradient estimation much more accurately exploiting the pseudo-1d structure. In particular, using the chain rule, one can write the gradient of the loss function wrt to $\mathbf{w}$ as:

$$\nabla_{\mathbf{w}} f_t(\mathbf{w}) = \nabla_{\mathbf{w}} \ell_t(g_t(\mathbf{w};\mathbf{x}_t)) = \ell_t'(g_t(\mathbf{w};\mathbf{x}_t)) \nabla_{\mathbf{w}} g_t(\mathbf{w};\mathbf{x}_t) \tag{6}$$

Notice that because we have access to g_t, we know the d-dimensional gradient part accurately. The only unknown part in the equation above is the *scalar* quantity which is $\ell_t'(g(\mathbf{w}_t;\mathbf{x}_t))$. For this, we can use the one-point estimator as in (Flaxman et al., 2005), which in expectation gives the gradient wrt to not the actual loss ℓ_t but wrt to a smoothed loss, as stated in the following lemma.

Lemma 6. *Fix $\delta > 0$ and let u take 1 or -1 with equal probability. Define the one-point gradient estimator, $\hat{\nabla}\ell_t(a) := \frac{1}{\delta}\ell_t\big(a+\delta u\big)u$. Then:*

$$\nabla_{\mathbf{w}} \mathbb{E}_u\big[\ell_t\big(g_t(\mathbf{w}_t;\mathbf{x}_t)+\delta u\big)\big] = \mathbb{E}_u\big[\hat{\nabla}\ell_t\big(g_t(\mathbf{w}_t;\mathbf{x}_t)\big)\big] \cdot \nabla_{\mathbf{w}} g_t(\mathbf{w}_t;\mathbf{x}_t)$$

The resulting online gradient descent method for PBCO is given in Algorithm 3. In Lemma 7, we give the $O(T^{3/4})$ regret bound for the algorithm.

Algorithm 3 Online Gradient Descent for PBCO

Input:
  Perturbation parameter: $\delta > 0$, $\alpha \in (0,1]$, learning rate: $\eta > 0$, max rounds T
Initialize:
  $\mathbf{w}_1 \leftarrow 0$
for $t = 1, 2, \cdots T$ **do**
  Sample $u \sim \mathbf{U}\big(\mathcal{S}_1(1)\big)$ (i.e. select u uniformly from $\{-1, 1\}$)
  Receive $\mathbf{x}_t$
  Project $\mathbf{w}_t \leftarrow \mathbf{P}_{\mathcal{W}_\alpha}(\mathbf{w}_t)$, where $\mathcal{W}_\alpha = \{\mathbf{w} \in \mathcal{W} \mid g_t(\mathbf{w};\mathbf{x}_t) \in \mathcal{G} - \alpha\}$
  Play $a_t = g_t(\mathbf{w}_t;\mathbf{x}_t) + \delta u$ and receive loss $\ell_t(a_t)$
  Update $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta\Big[\frac{1}{\delta}\ell_t(a_t)u\nabla g_t(\mathbf{w}_t;\mathbf{x}_t)\Big]$ ▷ One-point estimator of $\nabla f_t(\mathbf{w}_t)$
end for

Lemma 7 (Regret bound for Algorithm 3)**.** *Consider $\mathcal{W} = \mathcal{B}_d(W)$. If the losses $f_t : \mathcal{W} \to [0, C]$ and g_t, $t \in [T]$ satisfy* **(A1) (ii)**, **(A2)**, *and* **(A3) (ii)**, *then setting $\eta = \frac{W\delta}{DC\sqrt{T}}$, $\delta = \left(\frac{WDC}{3L\sqrt{T}}\right)^{1/2}$, and $\alpha = \delta$, the expected regret of Algorithm 3 can be bounded as:*

$$\mathbb{E}[\mathcal{R}_T(\mathcal{A})] \leq 2\sqrt{3WLDC}T^{3/4},$$

where the expectation $\mathbb{E}[\cdot]$ is with respect to the algorithm's randomization.

Thus, we are able to guarantee optimal regret bound for OPTPBCO matching the lower bound, by falling back on a suitably modified OGD algorithm when d is sufficiently large.

Remark 8 (Assumptions for OGD vs Kernelized Exponential Weights)**.** *To show the regret bound for Algorithm 2, we only need convexity of the one-dimensional function ℓ_t unlike in the OGD case (Algorithm 3) where we need convexity of f_t in the d-dimensional parameter $\mathbf{w}$. In particular, our analysis of kernelized exponential weights method (in Lemma 5) does not need other assumptions on g_t other than boundedness, which may be counter-intuitive (for example, consider when g_t is possibly non-convex and ℓ_t is the identity function). But note that the analysis relies on the complete knowledge of g_t and ignores the computational complexity. To be able to implement Algorithm 2 efficiently, we will need some nice property of g_t like convexity.*

The following remark shows that pseudo-1d structure helps improve known bounds for bandit convex optimization by a factor of $\sqrt{d}$ at least.

Remark 9. *Consider the simple setting of bandit convex optimization when the loss functions are linear, $f_t(\mathbf{w}_t) = \langle \mathbf{w}_t, \xi_t \rangle$, where ξ_t is the cost vector chosen by the adversary, not revealed to the learner. It is known that, for bandit linear optimization, the minimax optimal regret is $\Theta(d\sqrt{T})$ (Shamir, 2015). Note that, in contrast, the context vector $\mathbf{x}_t$ is revealed to the learner in our setting, and only the (scalar) loss computed on the linear model $\langle \mathbf{w}_t, \mathbf{x}_t \rangle$ is not revealed, which captures typical online decision making setting. This way of posing the problem helps us leverage the structure, and get a better dependence on d.*

5. Simulations

We present synthetic experiments that showcase the regret bounds established in Section 4. We work with a linear g_t for all the experiments. We fix $\mathcal{W} = \mathcal{B}_d(1)$, context vectors from $\{\|\mathbf{x}_t\|_2 \leq 1\}$, and the two loss functions (a) $f_t(\mathbf{w}) = (\langle \mathbf{w}, \mathbf{x}_t \rangle - y_t^*)^2$ where $y_t^* = \langle \mathbf{w}^*, \mathbf{x}_t \rangle$, for a fixed $\mathbf{w}^* \in \mathcal{B}_d(1)$, and (b) $f_t(\mathbf{w}) = |\langle \mathbf{w}, \mathbf{x}_t \rangle - y_t^*|$. The details on implementing Algorithm 2 are given in Appendix B.

OGD vs Kernelized Exponential Weights for PBCO. In Figure 1 (a)-(b), we show the expected regret of Algorithm 3 on the synthetic problem (averaged over 50 problem instances), scaled by $1/t^{3/4}$ at round t, for the two loss functions; this, according to Lemma 7, ensures that the expected regret converges to a numerical constant, independent of d, with increasing rounds. We observe this is indeed the case for different d values. In Figure 1 (c)-

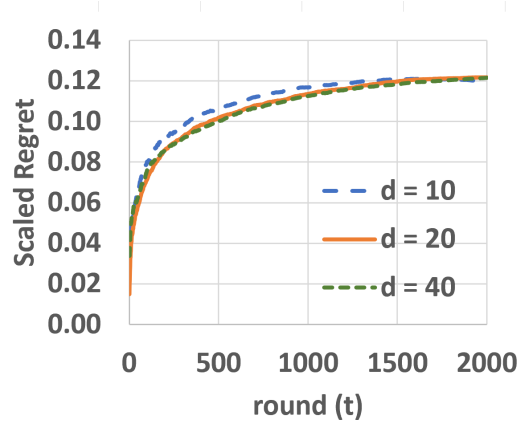
(d), we show the expected regret of Algorithm 2 on this problem (averaged over 50 problem instances), scaled by $1/\sqrt{t}$ at round t, for the two loss functions; this, according to Lemma 5, ensures that the regret converges to $O(\sqrt{d})$, with increasing rounds; notice that, e.g., in (c), for different d values, the converged scaled regret is $\gamma\sqrt{d}$ where $\gamma \approx 0.02/\sqrt{40} \approx 0.015/\sqrt{20} \approx 0.01/\sqrt{10} \approx 0.003$.

Comparison to (Flaxman et al., 2005). We present comparisons to the bandit OGD algorithm of (Flaxman et al., 2005) that does not exploit the pseudo-1d structure of the loss, achieving a regret of $O(\sqrt{d}T^{3/4})$, as against our Algorithm 3 that achieves a regret of $O(T^{3/4})$. In Figure 1 (e)-(f), we show the expected regret of the bandit OGD algorithm of (Flaxman et al., 2005) on the same data as earlier (averaged over 50 problem instances), scaled by $1/t^{3/4}$ at round t, for the two loss functions; this, according to (Flaxman et al., 2005), ensures that the regret converges to $O(\sqrt{d})$, with increasing rounds; notice that, e.g., in (e), for different d values, we can infer that the ratio of the converged regrets of (Flaxman et al., 2005) and our algorithm (corresponding to plot (a)) is at most $3\sqrt{d}$; the additional constant factor also appears in the analysis of (Flaxman et al., 2005).

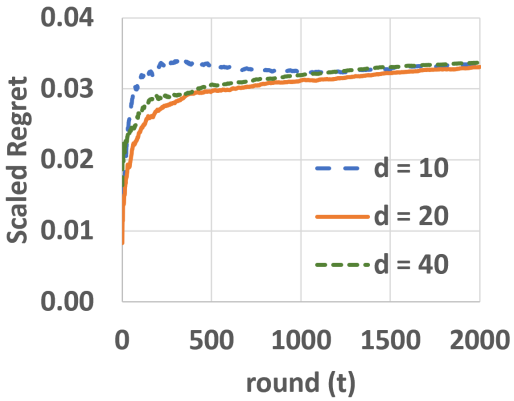
6. Conclusions and Future Work

We have formulated a novel bandit convex optimization problem with pseudo-1d structure motivated by its applications in online decision making and large-scale parameter tuning in systems. We provide optimal minimax regret bounds for the pseudo-1d bandit convex optimization problem. An open question here is if there is a single algorithm that achieves the regret trade-off we show in the lower bound (as against our method, that relies on two schemes in two regimes of dimensionality of the problem). Another follow-up direction is to extend the results in this work to settings when g_t is high-dimensional (when one needs to take multiple decisions based on the observed context), say $g_t(\mathbf{W}; \mathbf{x}) = \mathbf{W}\mathbf{x}$, where the parameters to estimate are $\mathbf{W} \in \mathbb{R}^{m \times d}$.

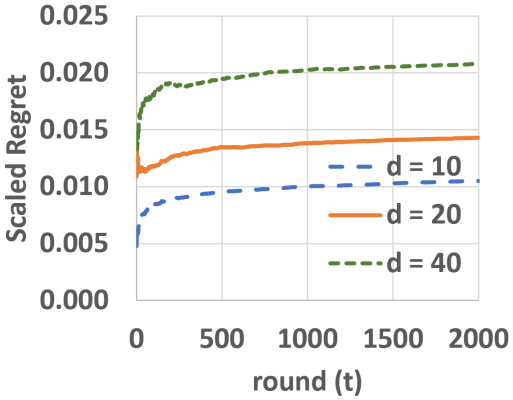
Acknowledgments: Most of the work was completed while AS was a graduate student at IISc, Bangalore and interning at MSR, India. AS thanks Qualcomm Innovation Fellowship 2019-20 for supporting this work.



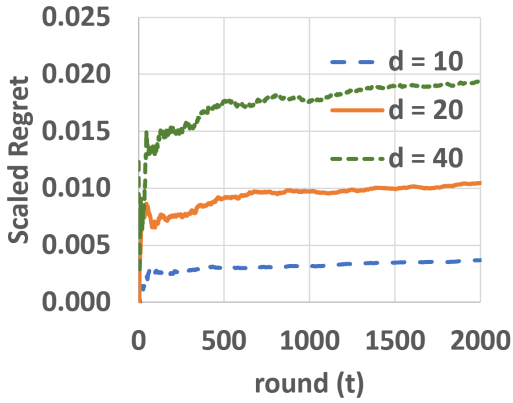
(a) Alg 3 (squared ℓ)



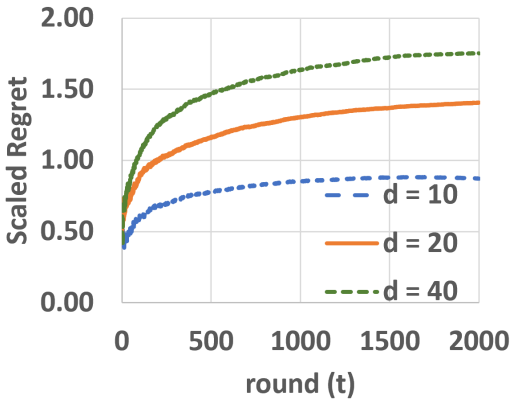
(b) Alg 3 (abs. ℓ)



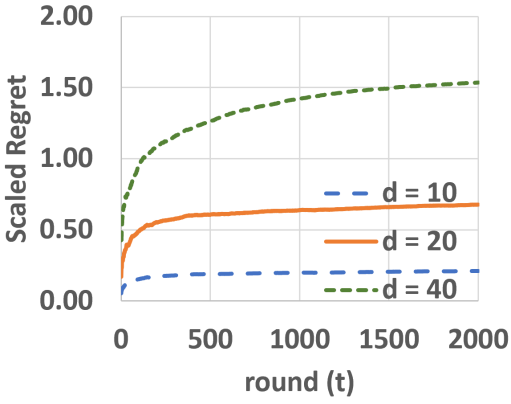
(c) Alg 2 (squared ℓ)



(d) Alg 2 (abs. ℓ)



(e) OGD of (Flaxman et al., 2005) (squared ℓ)



(f) OGD of (Flaxman et al., 2005) (abs. ℓ)

Figure 1. (a)-(b): Algorithm 3: Scaled cumulative regret $\mathcal{R}_t/t^{3/4}$ vs. t for the squared loss (a) and the absolute deviation loss (b). By Lemma 7, the (scaled) regret converges to a numerical constant independent of d. (c)-(d): Algorithm 2: Scaled cumulative regret $\mathcal{R}_t/\sqrt{t}$ vs. t for the squared loss (c) and the absolute deviation loss (d). In accordance with Lemma 5, the (scaled) regret converges to a value proportional to $\sqrt{d}$. (e)-(f): OGD algorithm of (Flaxman et al., 2005): Scaled cumulative regret $\mathcal{R}_t/t^{3/4}$ vs. t for the squared loss (e) and the absolute deviation loss (f). Compared to the corresponding pots in (a) and (b), it is evident that the regret is much higher here; in particular, in accordance with the result in (Flaxman et al., 2005), the (scaled) regret converges to a value proportional to $\sqrt{d}$.

References

Abernethy, J. D., Hazan, E., and Rakhlin, A. Competing in the dark: An efficient algorithm for bandit linear optimization. 2009.

Agarwal, A., Foster, D. P., Hsu, D. J., Kakade, S. M., and Rakhlin, A. Stochastic convex optimization with bandit feedback. In *Advances in Neural Information Processing Systems*, pp. 1035–1043, 2011.

Agarwal, A., Bird, S., Cozowicz, M., Hoang, L., Langford, J., Lee, S., Li, J., Melamed, D., Oshri, G., Ribas, O., et al. Making contextual decisions with low technical debt. *arXiv preprint arXiv:1606.03966*, 2016.

Bietti, A., Agarwal, A., and Langford, J. A contextual bandit bake-off. *arXiv preprint arXiv:1802.04064*, 2018.

Bubeck, S. and Eldan, R. Multi-scale exploration of convex functions and bandit convex optimization. In *Conference on Learning Theory*, pp. 583–589, 2016.

Bubeck, S., Cesa-Bianchi, N., et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5 (1):1–122, 2012.

Bubeck, S., Lee, Y. T., and Eldan, R. Kernel-based methods for bandit convex optimization. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 72–85. ACM, 2017.

Chen, L., Zhang, M., and Karbasi, A. Projection-free bandit convex optimization. *arXiv preprint arXiv:1805.07474*, 2018.

Flaxman, A. D., Kalai, A. T., and McMahan, H. B. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 385–394. Society for Industrial and Applied Mathematics, 2005.

Ghadimi, S. and Lan, G. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Hazan, E. and Li, Y. An optimal algorithm for bandit convex optimization. *arXiv preprint arXiv:1603.04350*, 2016.

Hazan, E., Agarwal, A., and Kale, S. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

Krishnamurthy, A., Langford, J., Slivkins, A., and Zhang, C. Contextual bandits with continuous actions: Smoothing, zooming, and adapting. In *Conference on Learning Theory*, pp. 2025–2027, 2019.

Majzoubi, M., Zhang, C., Chari, R., Krishnamurthy, A., Langford, J., and Slivkins, A. Efficient contextual bandits with continuous actions. *Advances in Neural Information Processing Systems*, 33, 2020.

Natarajan, N., Karthikeyan, A., Jain, P., Radicek, I., Rajamani, S., Gulwani, S., and Gehrke, J. Programming by rewards. *arXiv preprint arXiv:2007.06835*, 2020.

Saha, A. and Tewari, A. Improved regret guarantees for online smooth convex optimization with bandit feedback. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 636–642, 2011.

Sahu, A. K., Zaheer, M., and Kar, S. Towards gradient free and projection free stochastic optimization. *arXiv preprint arXiv:1810.03233*, 2018.

Shamir, O. On the complexity of bandit and derivative-free stochastic convex optimization. In *Conference on Learning Theory*, pp. 3–24, 2013.

Shamir, O. On the complexity of bandit linear optimization. In *Conference on Learning Theory*, pp. 1523–1551, 2015.

Shamir, O. An optimal algorithm for bandit and zero-order convex optimization with two-point feedback. *Journal of Machine Learning Research*, 18(52):1–11, 2017.

Wang, M., Fang, E. X., and Liu, H. Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. *Mathematical Programming*, 161(1-2):419–449, 2017.

Y. Abbasi-Yadkori, D. P. and Szepesvari, C. Improved algorithms for linear stochastic bandits. In *Neural Information Processing Systems*, 2011.

Yang, S. and Mohri, M. Optimistic bandit convex optimization. In *Advances in Neural Information Processing Systems*, pp. 2297–2305, 2016.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 928–936, 2003.