Bandit Online Learning with Unknown Delays

Bingcong Li
University of Minnesota

Tianyi Chen
University of Minnesota

Georgios B. Giannakis
University of Minnesota

Abstract

This paper deals with bandit online learning, where feedback of unknown delay can emerge in non-stochastic multi-armed bandit (MAB) and bandit convex optimization (BCO) settings. MAB and BCO require only values of the objective function to become available through feedback, and are used to estimate the gradient appearing in the corresponding iterative algorithms. Since the challenging case of feedback with *unknown* delays prevents one from constructing the sought gradient estimates, existing MAB and BCO algorithms become intractable. Delayed exploration, exploitation, and exponential (DEXP3) iterations, along with delayed bandit gradient descent (DBGD) iterations are developed for MAB and BCO with unknown delays, respectively. Based on a unifying analysis framework, it is established that both DEXP3 and DBGD guarantee an $\tilde{\mathcal{O}}\big(\sqrt{K(T+D)}\big)$ regret, where D denotes the delay accumulated over T slots, and K represents the number of arms in MAB or the dimension of decision variables in BCO. Numerical tests using both synthetic and real data validate DEXP3 and DBGD.

1 INTRODUCTION

Sequential decision making emerges in several learning and optimization tasks, such as online advertisement, online routing, and portfolio management (Hazan, 2016; Bubeck et al., 2012). Among popular methods for sequential decision making, non-stochastic multi-armed bandit (MAB) and bandit convex optimization (BCO) have widely-appreciated merits because they only rely on the value of loss function, and offer quantifiable performance guarantees. MAB and BCO can be viewed as a repeated game between a possibly randomized learner, and the possibly adversarial nature. In each round, the learner selects an action, and incurs the associated loss that is returned by the nature. In contrast to the full information setting, only the loss of the performed action rather than the gradient of the loss function (or even the loss function itself) is revealed to the learner. Popular approaches to bandit online learning estimate gradients using several point-wise evaluations of the loss function, and use them to run online gradient-type algorithms; see e.g., Auer et al. (2002) for MAB and Flaxman et al. (2005); Agarwal et al. (2010) for BCO.

Although widely applicable with solid performance guarantees, standard MAB and BCO approaches do not account for delayed feedback that is naturally present in various applications. For example, when carrying out machine learning tasks using distributed mobile devices (a setup referred to as federated learning) (McMahan et al., 2017), delay comes from the time it takes to compute at mobile devices and also to transmit over the wireless communication links; in online recommender systems the click-through rate could be aggregated and then periodically sent back (Li et al., 2010); in online routing over communication networks, the latency of each routing decision can be revealed only after packets arrive to their destination (Awerbuch and Kleinberg, 2004); and in parallel computing, computations are carried with outdated information because workers are not synchronized (Agarwal and Duchi, 2011; Duchi et al., 2013; McMahan and Streeter, 2014). In these cases, the delay can be *unknown* to the learner. One relevant example appears also in recommender systems, when the same item is recommended multiple times and the feedback is delayed, which makes it challenging to deduce which round of recommendation triggered the user interests.

Challenges arise naturally when dealing with bandit online learning with unknown delays, simply because unknown delays prevent existing MAB as well as BCO methods to construct reliable gradient estimates. To address this limitation, our solution is a fine-grained *biased* gradient estimator for MAB and a *deterministic* gradient estimator for BCO, where the standard unbiased loss estimator for MAB and the nearly unbiased one for BCO are no longer available. The resultant algorithms, that we abbreviate as DEXP3 and DBGD, are guaranteed to achieve an $\tilde{\mathcal{O}}\big(\sqrt{K(T+D)}\big)$ re-

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89. Copyright 2019 by the author(s).

gret over a T-slot time horizon, where D denotes the overall delay, and K represents the number of arms in MAB or the dimension of decision variables in BCO.

1.1 Related Works

Delayed online learning can be categorized depending on whether the feedback information is full or bandit. We review prior works from these two aspects.

Delayed Online Learning. This class deals with delayed but fully revealed information, namely fully known gradient or loss function. It is proved that an $\mathcal{O}\big(\sqrt{T+D}\big)$ regret is achievable for a T-slot time horizon with overall delay D. Particularly, algorithms dealing with a fixed delay have been studied by Weinberger and Ordentlich (2002). To reduce the storage and computation burden of Weinberger and Ordentlich (2002), an online gradient descent type algorithm for fixed d-slot delay was developed by Langford et al. (2009), where the lower bound $\mathcal{O}\big(\sqrt{(d+1)T}\big)$ was also provided. Adversarial delay has been tackled recently by Joulani et al. (2016); Shamir and Szlak (2017) and Quanrud and Khashabi (2015). However, the algorithms as well as the corresponding analyses are not applicable to the bandit online learning setup.

Delayed Bandit Online Learning. Stochastic MAB with delays has been reported by Chapelle and Li (2011); Desautels et al. (2014); Vernade et al. (2017); Pike-Burke et al. (2017); see also Joulani et al. (2013) for multi-instance generalizations introduced to handle adversarial delays in stochastic and non-stochastic MAB settings. For non-stochastic MAB, EXP3-based algorithms were developed to handle fixed delays in Cesa-Bianchi et al. (2016); Neu et al. (2010). Although not requiring memory for extra instances, the delay in Neu et al. (2010) and Cesa-Bianchi et al. (2016) must be known. A recent work by Cesa-Bianchi et al. (2018) considers a more general non-stochastic MAB setting, where the feedback is composite and anonymous. Although the scope in the work of Cesa-Bianchi et al. (2018) is general, we will see in Section 5.2, that the regret bound of the MAB algorithm in Cesa-Bianchi et al. (2018) can be improved in terms of delay dependence in the considered setting[1].

1.2 Contributions

Our main contributions can be summarized as follows.

c1) Based on a *biased* gradient estimator, a delayed exploration-exploitation exponentially (DEXP3) weighted algorithm is developed for delayed non-stochastic MAB with *unknown* and even *adversarially* chosen delays;

c2) Relying on a *deterministic* gradient estimator, a delayed bandit gradient descent (DBGD) algorithm is developed to handle the delayed BCO setting; and,

c3) A unified analysis framework is introduced to reveal that DEXP3 and DBGD provably achieve an $\tilde{\mathcal{O}}\big(\sqrt{K(T+D)}\big)$ regret, where D is the cumulative delay over T slots, and K denotes the number of arms in MAB or the dimension of decision variables in BCO. Numerical tests validate the proposed DEXP3 and DBGD.

Notational Conventions. Bold lowercase letters denote column vectors; $\mathbb{E}[\,\cdot\,]$ represents expectation; $\mathbb{1}(\,\cdot\,)$ denotes the indicator function; $\boldsymbol{x}^\top$ stands for vector transposition; and $\|\boldsymbol{x}\|$ denotes the ℓ_2-norm of a vector $\boldsymbol{x}$.

2 PROBLEM STATEMENT

Before introducing our setups with delays, we outline first the standard non-stochastic MAB and BCO ones.

2.1 MAB and BCO

Non-stochastic MAB. Consider the MAB setup with K arms (a.k.a. actions) (Bubeck et al., 2012; Auer et al., 2002). At the beginning of slot t, without knowing the loss corresponding to each arm, the learner selects an arm a_t in accordance to the $K \times 1$ probability mass distribution vector $\boldsymbol{p}_t \in \Delta_K$, where $\Delta_K := \{\boldsymbol{p} \in \Delta_K : p(k) \geq 0, \forall k;\ \sum_{k=1}^{K} p(k) = 1\}$ denotes the probability simplex. The loss $l_t(a_t)$ incurred when selecting a_t is an entry of the $K \times 1$ loss vector $\boldsymbol{l}_t$, and it is observed by the learner. This and previously observed losses $\{l_s(a_s)\}_{s=1}^{t}$, are used to obtain $\boldsymbol{p}_{t+1}$ of the ensuing slot; see also Fig. 1 (a1).

The goal is to minimize the regret, which is the difference between the expected cumulative loss of the learner relative to the loss of the best fixed policy in hindsight, given by

$$\mathrm{Reg}_T^{\mathrm{MAB}} := \sum_{t=1}^{T} \mathbb{E}\big[\boldsymbol{p}_t^\top \boldsymbol{l}_t\big] - \sum_{t=1}^{T} (\boldsymbol{p}^*)^\top \boldsymbol{l}_t \tag{1}$$

where the expectation is taken w.r.t. the possible randomness of $\boldsymbol{p}_t$ induced by the selection of $\{a_s\}_{s=1}^{t-1}$, while the best fixed policy $\boldsymbol{p}^*$ is defined as

$$\boldsymbol{p}^* := \arg\min_{\boldsymbol{p} \in \Delta_K} \sum_{t=1}^{T} \boldsymbol{p}^\top \boldsymbol{l}_t.$$

If for instance $\boldsymbol{p}^* = [0, \ldots, 1, \ldots, 0]^\top$, the regret is relative to the corresponding best fixed arm in hindsight.

BCO. Consider now the BCO setup with M-point feedback (Agarwal et al., 2010). At the beginning of slot t, without knowing the loss function f_t, the learner selects the $K \times 1$ vector $\boldsymbol{x}_t \in \mathcal{X}$ of decision variables, where $\mathcal{X} \subset \mathbb{R}^K$ is a compact and convex set. Along with $\boldsymbol{x}_{t,0} := \boldsymbol{x}_t$, loss values $\{f_t(\boldsymbol{x}_{t,k})\}_{k=0}^{M-1}$ are then observed at another $M-1$

[1]However, these two regret bounds match in the worst case where all delays equal the maximal delay.

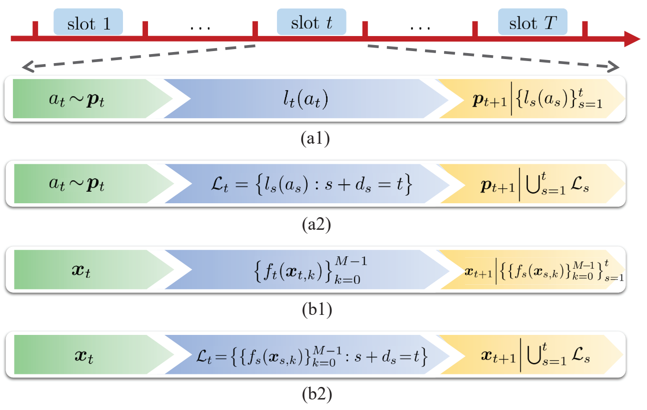
slot 1
. . .
slot t
. . .
slot T
$a_t \sim \boldsymbol{p}_t$
$l_t(a_t)$
$\boldsymbol{p}_{t+1} \big| \{l_s(a_s)\}_{s=1}^t$
(a1)
$a_t \sim \boldsymbol{p}_t$
$\mathcal{L}_t = \{l_s(a_s) : s + d_s = t\}$
$\boldsymbol{p}_{t+1} \big| \bigcup_{s=1}^t \mathcal{L}_s$
(a2)
$\boldsymbol{x}_t$
$\{f_t(\boldsymbol{x}_{t,k})\}_{k=0}^{M-1}$
$\boldsymbol{x}_{t+1} \big| \{\{f_s(\boldsymbol{x}_{s,k})\}_{k=0}^{M-1}\}_{s=1}^t$
(b1)
$\boldsymbol{x}_t$
$\mathcal{L}_t = \{\{f_s(\boldsymbol{x}_{s,k})\}_{k=0}^{M-1} : s + d_s = t\}$
$\boldsymbol{x}_{t+1} \big| \bigcup_{s=1}^t \mathcal{L}_s$
(b2)

Figure 1: A single-slot structure of: (a1) standard non-stochastic MAB, where $\boldsymbol{p}_{t+1}|\{l_s(a_s)\}_{s=1}^t$ means that $\boldsymbol{p}_{t+1}$ is updated based on past losses $\{l_s(a_s)\}_{s=1}^t$; (a2) MAB with delayed feedback; (b1) standard BCO; and (b2) BCO with delayed feedback.

points $\{\boldsymbol{x}_{t,k} \in \mathcal{X}\}_{k=1}^{M-1}$. The learner leverages the revealed losses to obtain the next decision vector (action) $\boldsymbol{x}_{t+1}$; see also Fig. 1 (b1). The sequence of $\left\{\{\boldsymbol{x}_{t,k}\}_{k=0}^{M-1}\right\}_{t=1}^T$ is chosen to minimize the regret relative to the best fixed action in hindsight[2]

$$\text{Reg}_T^{\text{BCO}} := \sum_{t=1}^T \mathbb{E}\big[f_t(\boldsymbol{x}_t)\big] - \sum_{t=1}^T f_t(\boldsymbol{x}^*) \qquad (2)$$

where the expectation is taken over the sequence of random actions $\{\boldsymbol{x}_\tau\}_{s=1}^{t-1}$. The best fixed action $\boldsymbol{x}^*$ in hindsight is

$$\boldsymbol{x}^* := \arg\min_{\boldsymbol{x}\in\mathcal{X}} \sum_{t=1}^T f_t(\boldsymbol{x}).$$

In both MAB and BCO settings, an online algorithm is desirable with regret that is sublinear w.r.t. the time horizon T, that is, $\text{Reg}_T^{\text{MAB}} = o(T)$ and $\text{Reg}_T^{\text{BCO}} = o(T)$; see e.g., (Hazan, 2016; Bubeck et al., 2012).

2.2 Delayed MAB and BCO

MAB with Unknown Delays. In *delayed* MAB, the learner still chooses an arm $a_t \sim \boldsymbol{p}_t$ at the beginning of slot t. However, the loss $l_t(a_t)$ is observed after d_t slots, namely, at the end of slot $t + d_t$, where delay $d_t \geq 0$ can vary from slot to slot. In this paper, we assume that $\{d_t\}_{t=1}^T$ can be chosen adversarially by nature. Let $l_{s|t}(a_{s|t})$ denote the loss incurred by the selected arm a_s in slot s but observed at t, i.e., the learner receives the losses collected in $\mathcal{L}_t = \left\{l_{s|t}(a_{s|t}),\ s : s+d_s = t\right\}$ at the end of slot t. Note that it is possible to have $\mathcal{L}_t = \emptyset$ in certain slots. And the order of feedback can be arbitrary, meaning it is possible to have $t_1 + d_{t_1} \geq t_2 + d_{t_2}$ when $t_1 \leq t_2$. In contrast to Joulani et al. (2013) however, we consider that the delay d_t may not be accessible, in which case the learner just observes the value of $l_{s|t}(a_{s|t})$, but not s. The learner's goal is to select $\{\boldsymbol{p}_t\}_{t=1}^T$ "on-the-fly" to minimize the regret in (1). Note that with unknown delays, the information available to decide $\boldsymbol{p}_t$ is even less compared with that in the standard MAB. Specifically, the available information to decide $\boldsymbol{p}_t$ is contained in $\mathcal{L}_{1:t-1} := \bigcup_{s=1}^{t-1} \mathcal{L}_s$; see also Fig. 1 (a2).

For simplicity, we assume that all feedback information is received at the end of slot T. This assumption comes without loss of generality since the feedback arriving at the end of slot T cannot aid the arm selection, hence the final performance of the learner will not be affected.

BCO with Unknown Delays. For delayed BCO, the learner still chooses $\boldsymbol{x}_t$ to play while querying $\{\boldsymbol{x}_{t,k}\}_{k=1}^{M-1}$ at the beginning of slot t. However, the loss as well as the querying responses $\left\{f_t(\boldsymbol{x}_{t,k})\right\}_{k=0}^{M-1}$ are observed at the end of slot $t + d_t$. With $f_{s|t}(\boldsymbol{x}_{s|t})$ denoting the loss incurred in slot s but observed at t, the feedback set at the end of slot t is $\mathcal{L}_t = \left\{\{f_{s|t}(\boldsymbol{x}_{s|t,k})\}_{k=0}^{M-1}, s : s+d_s = t\right\}$. To find $\boldsymbol{x}_t$, the learner relies on the history $\mathcal{L}_{1:t-1} := \bigcup_{s=1}^{t-1} \mathcal{L}_s$, with the goal of minimizing the regret in (2).

Why Existing Algorithms Fail with Unknown Delays? The algorithms for standard (non-delayed) MAB, such as EXP3, cannot be applied to delayed MAB with unknown delays. Recall that in settings without delay, to deal with the partially observed $\boldsymbol{l}_t$, EXP3 relies on an importance sampling type of loss estimates given by

$$\hat{l}_t(k) = \frac{l_t(a_t)\mathbb{1}(a_t = k)}{p_t(k)}, \quad k = 1, \ldots, K\,. \qquad (3)$$

The denominator as well as the indicator function in (3) ensure unbiasedness of $\hat{l}_t(k)$. Leveraging the estimated loss, the distribution vector $\boldsymbol{p}_{t+1}$ is obtained with entries

$$p_{t+1}(k) = \frac{p_t(k)\exp(-\eta\hat{l}_t(k))}{\sum_{j=1}^K p_t(j)\exp(-\eta\hat{l}_t(j))}, \quad k = 1, \ldots, K$$

where η is the learning rate. Consider now that the loss $l_{s|t}(a_{s|t})$ with delay d_s is observed at $t = s + d_s$. To recover the unbiased estimator $\hat{l}_{s|t}(k)$ in (3), $p_s(k)$ must be known. However, since d_s is not revealed, even if the learner can store past probability distribution vectors, it is not clear how to attain the loss estimator.

Knowing the delay is instrumental when it comes to the gradient estimator in BCO as well. For non-delayed single-point ($M = 1$) feedback BCO (Flaxman et al., 2005), since only one value of the loss instead of the full gradient is observed per slot, the idea is to draw $\boldsymbol{u}_t$ uniformly from the surface of a unit ball in $\mathbb{R}^K$, to form the gradient estimate

$$\boldsymbol{g}_t = \frac{K}{\delta} f_t(\boldsymbol{x}_t + \delta\boldsymbol{u}_t)\boldsymbol{u}_t \qquad (4)$$

[2]This definition is slightly different with that in Agarwal et al. (2010). However, we will show in Section 5.2 that the regret bound is not affected.

where δ is a small constant. The next action is obtained using a standard online (projected) gradient descent iteration leveraging the estimated gradient, that is

$$\boldsymbol{x}_{t+1} = \Pi_{\mathcal{X}_\delta}[\boldsymbol{x}_t - \eta \boldsymbol{g}_t]$$

where $\mathcal{X}_\delta$ is the shrunk feasibility set to ensure $\boldsymbol{x}_t + \boldsymbol{u}_t$ is feasible. While $\boldsymbol{g}_t$ serves as a nearly unbiased estimator of $\nabla f_t(\boldsymbol{x}_t)$, the unknown delay causes mismatch between the feedback $f_{s|t}(\boldsymbol{x}_{s|t} + \delta \boldsymbol{u}_{s|t})$ and $\boldsymbol{g}_{s|t}$. Specifically, given the feedback $f_{s|t}(\boldsymbol{x}_{s|t} + \delta \boldsymbol{u}_{s|t})$, since d_s is unknown, the learner does not know $\boldsymbol{u}_{s|t}$ to obtain $\boldsymbol{g}_{s|t}$ in (4). Similar arguments also hold for BCO with multi-point feedback.

Therefore, performing delayed bandit learning with unknown delays is challenging, and has not been explored.

3 DEXP3 FOR DELAYED MAB

We start with the non-stochastic MAB setup that is randomized in nature since an arm a_t is chosen randomly per slot according to a $K \times 1$ probability mass vector $\boldsymbol{p}_t$. In this section, we show that for the MAB problem, so long as the (unknown) delay is bounded, based only on a single-point feedback, the randomized algorithm that we term Delayed EXP3 (DEXP3) can cope with unknown delays in MAB through a *biased* loss estimator, which provably leads to a desirable regret.

Recall that the feedback at slot t includes losses incurred at slots $s_n, n = 1, 2, \ldots, |\mathcal{L}_t|$, where $\mathcal{L}_t := \{l_{s_n|t}(a_{s_n|t}) : \forall s_n = t - d_{s_n}\}$. Once $\mathcal{L}_t$ is revealed, the learner estimates $\boldsymbol{l}_{s_n|t}$ by scaling the observed loss according to $\boldsymbol{p}_t$ at the current slot. For each entry $l_{s_n|t}(a_{s_n|t}) \in \mathcal{L}_t$, the estimator of the loss vector $\boldsymbol{l}_{s_n|t}$ is

$$\hat{l}_{s_n|t}(k) = \frac{l_{s_n|t}(k) \mathbb{1}(a_{s_n|t} = k)}{p_t(k)}, \quad k = 1, \ldots, K. \tag{5}$$

It is worth mentioning that the index s_n in (5) is only used for analysis while during the implementation, there is no need to know s_n. In contrast to EXP3 and its variant for delayed MAB with known delays (Joulani et al., 2013; Cesa-Bianchi et al., 2016), our estimator for $l_{s_n|t}(k)$ in (5) turns out to be *biased* since $a_{s_n|t}$ is chosen according to $\boldsymbol{p}_{s_n}$ instead of $\boldsymbol{p}_t$, that is

$$\mathbb{E}_{a_{s_n|t}}\left[\hat{l}_{s_n|t}(k)\right] = \frac{l_{s_n|t}(k) p_{s_n}(k)}{p_t(k)} \neq l_{s_n|t}(k).$$

Since $\mathcal{L}_t$ may contain multiple rounds of feedback, leveraging each $\hat{\boldsymbol{l}}_{s_n|t}$, the learner must perform $|\mathcal{L}_t|$ updates to obtain $\boldsymbol{p}_{t+1}$. Intuitively, to upper bound the bias of (5), an upper bound on $p_{s_n}(k)/p_t(k)$ is required, which in turn calls for a lower bound on $p_t(k)$. On the other hand however, the lower bound of $p_t(k)$ should be small enough to avoid incurring extra regret. Different from EXP3, our DEXP3 ensures a lower bound on $p_t(k)$ by introducing an intermediate weight vector $\tilde{\boldsymbol{w}}_t$ to evaluate the historical performance of each arm. Let n denote the index of the inner-loop update at slot t starting from $\boldsymbol{p}_t^0 := \boldsymbol{p}_t$. For each $l_{s_n|t}(a_{s_n|t}) \in \mathcal{L}_t$, the learner first updates $\tilde{\boldsymbol{w}}_t^n$ by using the estimated loss $\hat{\boldsymbol{l}}_{s_n|t}$ as

$$\tilde{w}_t^n(k) = p_t^{n-1}(k) \exp\left(-\eta \min\left\{\delta_1, \hat{l}_{s_n|t}(k)\right\}\right), \ \forall k \tag{6}$$

where η is the learning rate, and δ_1 serves as an upper bound of $\hat{l}_{s_n|t}(k)$ to control the bias of $\hat{l}_{s_n|t}(k)$. However, to confine the extra regret incurred by introducing δ_1, a carefully-selected δ_1 should ensure that the probability of having $\hat{l}_{s_n|t}(k)$ larger than δ_1 is small enough. Then the learner finds $\boldsymbol{w}_t^n$ by a trimmed normalization with weights

$$w_t^n(k) = \max\left\{\frac{\tilde{w}_t^n(k)}{\sum_{j=1}^K \tilde{w}_t^n(j)}, \frac{\delta_2}{K}\right\}, \quad k = 1, \ldots, K. \tag{7}$$

Update (7) ensures that $w_t^n(k)$ is lower bounded by δ_2/K. Finally, the learner normalizes $\boldsymbol{w}_t^n$ to obtain $\boldsymbol{p}_t^n$ as

$$p_t^n(k) = \frac{w_t^n(k)}{\sum_{j=1}^K w_t^n(j)}, \quad k = 1, \ldots, K. \tag{8}$$

It can be shown that $p_t^n(k)$ is lower bounded as $p_t^n(k) \geq \frac{\delta_2}{K(1+\delta_2)}$ [cf. (28) in supplementary material]. After all the elements of $\mathcal{L}_t$ have been used, the learner finds $\boldsymbol{p}_{t+1}$ via

$$\boldsymbol{p}_{t+1} = \boldsymbol{p}_t^{|\mathcal{L}_t|}. \tag{9}$$

Furthermore, if $\mathcal{L}_t = \emptyset$, the learner directly reuses the previous distribution, meaning $\boldsymbol{p}_{t+1} = \boldsymbol{p}_t$, and chooses an arm accordingly. Our DEXP3 is summarized step-by-step

Algorithm 1 DEXP3

```
 1: Initialize: $\boldsymbol{p}_1(k) = 1/K, \forall k$.
 2: for $t = 1, 2 \ldots, T$ do
 3:     Select an arm $a_t \sim \boldsymbol{p}_t$.
 4:     Observe feedback collected in set $\mathcal{L}_t$.
 5:     for $n = 1, 2, \ldots, |\mathcal{L}_t|$ do
 6:         Estimate $\hat{\boldsymbol{l}}_{s_n|t}$ via (5) if $l_{s_n|t}(a_{s_n|t}) \in \mathcal{L}_t$.
 7:         Update $\boldsymbol{p}_t^n$ via (6) - (8).
 8:     end for
 9:     Obtain $\boldsymbol{p}_{t+1}$ via (9).
10: end for
```

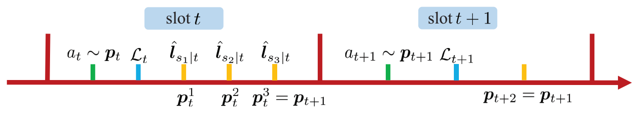
slot t
slot $t+1$
$a_t \sim \boldsymbol{p}_t$
$\mathcal{L}_t$
$\hat{\boldsymbol{l}}_{s_1|t}$
$\hat{\boldsymbol{l}}_{s_2|t}$
$\hat{\boldsymbol{l}}_{s_3|t}$
$\boldsymbol{p}_t^1$
$\boldsymbol{p}_t^2$
$\boldsymbol{p}_t^3 = \boldsymbol{p}_{t+1}$
$a_{t+1} \sim \boldsymbol{p}_{t+1}$
$\mathcal{L}_{t+1}$
$\boldsymbol{p}_{t+2} = \boldsymbol{p}_{t+1}$

Figure 2: An example of DEXP3 with $\mathcal{L}_t$ including the losses incurred in slot s_1, s_2, and s_3 and $\mathcal{L}_{t+1} = \emptyset$.

Algorithm 2 DBGD

```
1: Initialize: $\boldsymbol{x}_1 = \mathbf{0}$.
2: for $t = 1, 2 \ldots, T$ do
3:     Play $\boldsymbol{x}_t$, also query $\boldsymbol{x}_t + \delta \boldsymbol{e}_k, k = 1, \ldots, K$.
4:     Observe feedback collected in set $\mathcal{L}_t$.
5:     if $\mathcal{L}_t = \emptyset$ then set $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t$.
6:     else estimate gradient $\boldsymbol{g}_{s_n|t}$ via (11) if $s_n + d_s = t$.
7:         Update $\boldsymbol{x}_{t+1}$ via (12).
8:     end if
9: end for
```

in Alg. 1. We will show in Sec. 5.2 that if the delay d_t is bounded by a constant $\bar{d}$, DEXP3 can guarantee an $\tilde{\mathcal{O}}\big(\sqrt{K(T+D)}\big)$ regret, where $D = \sum_{t=1}^{T} d_t$ is the overall delay.

Remark 1. The recent composite loss wrapper algorithm (abbreviated as CLW) in Cesa-Bianchi et al. (2018) can be also applied to the delayed MAB problem. However, CLW is designed for a more general setting with composite and anonymous feedback, and its *efficiency* drops when the previous action $a_{s|t}$ is known. The main differences between DEXP3 and CLW are: i) the loss estimators are different; ii) DEXP3 updates $\boldsymbol{p}_t$ in every slot, while CLW updates occur every other $\mathcal{O}(2\bar{d})$ slots (thus requiring a larger learning rate); and iii) DEXP3 does not involve $\bar{d}$ in the loss estimator, leading to a tighter regret bound in terms of delay $\tilde{\mathcal{O}}(\sqrt{K(T+D)})$, in contrast to $\tilde{\mathcal{O}}(\sqrt{(\bar{d}+1)KT})$ of CLW. As it will be corroborated by simulations, DEXP3 outperforms CLW in the considered setting.

4 DBGD FOR DELAYED BCO

In this section, we develop an algorithm that we term delayed bandit gradient descent (DBGD) based on a *deterministic* gradient estimator using $M = K + 1$-point loss feedback. DBGD enjoys regret of $\mathcal{O}\big(\sqrt{K(T+D)}\big)$ for BCO problems even when the delays are unknown. In practice, $(K+1)$-point feedback can be obtained i) when it is possible to evaluate the loss function easily; and ii) when the slot duration is long enough, meaning that the algorithm has sufficient time to query multiple points from the oracle.

The intuition behind our deterministic approximation originates from the gradient definition (Agarwal et al., 2010). Consider for example $\boldsymbol{x} \in \mathbb{R}^2$, and the gradient $\nabla f(\boldsymbol{x}) = [\nabla_1, \nabla_2]^\top$, where

$$\nabla_1 = \lim_{\delta \to 0} \frac{f(\boldsymbol{x}+\delta \boldsymbol{e}_1) - f(\boldsymbol{x})}{\delta}; \nabla_2 = \lim_{\delta \to 0} \frac{f(\boldsymbol{x}+\delta \boldsymbol{e}_2) - f(\boldsymbol{x})}{\delta}.$$

Similarly, for a K-dimensional $\boldsymbol{x}$, if $(K+1)$-point feedback is available, the gradient can be approximated as

$$\boldsymbol{g}_t = \frac{1}{\delta} \sum_{k=1}^{K} \big(f_t(\boldsymbol{x}_t + \delta \boldsymbol{e}_k) - f_t(\boldsymbol{x}_t)\big) \boldsymbol{e}_k \tag{10}$$

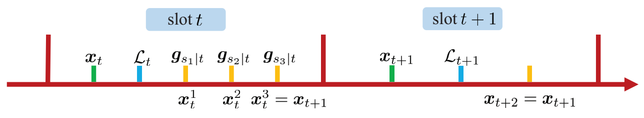

Figure 3: An example of DBGD with $\mathcal{L}_t$ including the losses incurred in slot s_1, s_2, and s_3 and $\mathcal{L}_{t+1} = \emptyset$.

where $\boldsymbol{e}_k := [0, \ldots, 1, \ldots, 0]^\top$ denotes the canonical vector with k-th entry equal to 1. Intuitively, a smaller δ improves the approximation accuracy. When $f_t(\cdot)$ is further assumed to be linear, $\boldsymbol{g}_t$ in (10) is unbiased. In this case, the gradient of $f_t(\boldsymbol{x}_t)$ can be recovered exactly, and thus the setup boils down to a delayed one with full information. However, if $f_t(\cdot)$ is generally convex, $\boldsymbol{g}_t$ in (10) is *biased*.

Leveraging the gradient in (10), we are ready to introduce our DBGD algorithm. Per slot t, the learner plays $\boldsymbol{x}_t$ and also queries $f_t(\boldsymbol{x}_t + \delta \boldsymbol{e}_k)$, for $k = 1, \ldots, K$. However, to ensure that $f_t(\boldsymbol{x}_t + \delta \boldsymbol{e}_k)$ is feasible, the $\boldsymbol{x}_t$ should be confined to the set $\mathcal{X}_\delta = \{\boldsymbol{x} : \frac{\boldsymbol{x}}{1-\delta} \in \mathcal{X}\}$. Note that if $\delta \in [0,1)$, then $\mathcal{X}_\delta$ is still convex. Let $n = 1, 2, \ldots, |\mathcal{L}_t|$ index the inner loop update at slot t. At the end of slot t, the learner receives observations $\mathcal{L}_t = \big\{\{f_{s_n|t}(\boldsymbol{x}_{s_n|t}), f_{s_n|t}(\boldsymbol{x}_{s_n|t} + \delta \boldsymbol{e}_k), k = 1, \ldots, K\}, \forall s_n = t - d_{s_n}\big\}$. Per received feedback value, the learner approximates the gradient via (10); thus, for each $\{f_{s_n|t}(\boldsymbol{x}_{s_n|t}), f_{s_n|t}(\boldsymbol{x}_{s_n|t} + \delta \boldsymbol{e}_k), k = 1, \ldots, K\}$, we have

$$\boldsymbol{g}_{s_n|t} = \frac{1}{\delta} \sum_{k=1}^{K} \big(f_{s_n|t}(\boldsymbol{x}_{s_n|t} + \delta \boldsymbol{e}_k) - f_{s_n|t}(\boldsymbol{x}_{s_n|t})\big) \boldsymbol{e}_k. \tag{11}$$

With $\boldsymbol{g}_{s_n|t}$ and $\boldsymbol{x}_t^0 := \boldsymbol{x}_t$, the learner will update $|\mathcal{L}_t|$ times to obtain $\boldsymbol{x}_{t+1}$ as

$$\boldsymbol{x}_t^n = \Pi_{\mathcal{X}_\delta}\big[\boldsymbol{x}_t^{n-1} - \eta \boldsymbol{g}_{s_n|t}\big], \quad n = 1, \ldots, |\mathcal{L}_t| \tag{12a}$$

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t^{|\mathcal{L}_t|}. \tag{12b}$$

If no feedback is received at slot t, the learner simply sets $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t$. The DBGD is summarized in Algorithm 2.

5 UNIFYING REGRET ANALYSIS

In this section, we show that both DEXP3 and DBGD can guarantee an $\tilde{\mathcal{O}}\big(\sqrt{K(T+D)}\big)$ regret. Our analysis broadens that in Joulani et al. (2016), which was originally developed for delayed online learning with full-information feedback.

5.1 Mapping from Real to Virtual Slots

To analyze the recursion involving consecutive variables ($\boldsymbol{p}_t$ and $\boldsymbol{p}_{t+1}$ in DEXP3 or $\boldsymbol{x}_t$ and $\boldsymbol{x}_{t+1}$ in DBGD) is challenging, since different from standard settings, the number

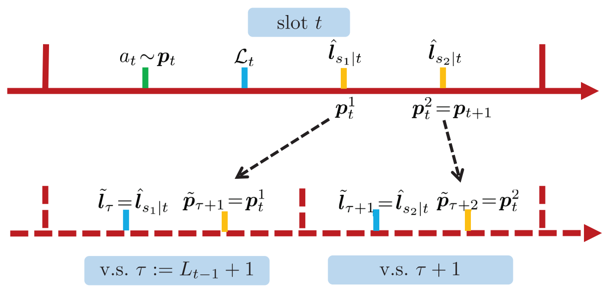
slot t
$a_t \sim \boldsymbol{p}_t$
$\mathcal{L}_t$
$\hat{\boldsymbol{l}}_{s_1|t}$
$\hat{\boldsymbol{l}}_{s_2|t}$
$\boldsymbol{p}_t^1$
$\boldsymbol{p}_t^2 = \boldsymbol{p}_{t+1}$
$\tilde{\boldsymbol{l}}_\tau = \hat{\boldsymbol{l}}_{s_1|t}$
$\tilde{\boldsymbol{p}}_{\tau+1} = \boldsymbol{p}_t^1$
$\tilde{\boldsymbol{l}}_{\tau+1} = \hat{\boldsymbol{l}}_{s_2|t}$
$\tilde{\boldsymbol{p}}_{\tau+2} = \boldsymbol{p}_t^2$
v.s. $\tau := L_{t-1} + 1$
v.s. $\tau + 1$

Figure 4: An example of mapping from real slots (solid line) to virtual slots (dotted line). At the end of slot t, the feedback is $\mathcal{L}_t = \{l_{s_1|t}(a_{s_1|t}), l_{s_2|t}(a_{s_2|t})\}$. "v.s." stands for virtual slot.

of feedback rounds varies from slot to slot. We will bypass this variable feedback using the notion of a "virtual slot."

Over the real time horizon, there are in total T virtual slots, where the τth virtual slot is associated with the τth loss value fed back. Recall that the feedback received at the end of slot t is $\mathcal{L}_t$. With the overall feedback received until the end of slot $t-1$ denoted by $L_{t-1} := \sum_{s=1}^{t-1} |\mathcal{L}_s|$, the virtual slot τ corresponding to the first feedback value received at slot t is $\tau = L_{t-1} + 1$. In what follows, we will use MAB as an example to elaborate on this mapping, but the BCO setting can be argued likewise.

When multiple rounds of feedback are received over a real slot t, DEXP3 updates $|\mathcal{L}_t|$ times $\boldsymbol{p}_t$ to obtain $\boldsymbol{p}_{t+1}$; see (6) - (8). Using the notion of virtual slots, these $|\mathcal{L}_t|$ updates are performed over $|\mathcal{L}_t|$ consecutive virtual slots. Taking Fig. 4 as an example, when $\boldsymbol{p}_t^1$ is obtained by using an estimated loss $\hat{\boldsymbol{l}}_{s_1|t}$ [cf. (5)] and (6) - (8), this update is mapped to a virtual slot $\tau = L_{t-1} + 1$, where $\tilde{\boldsymbol{l}}_\tau := \hat{\boldsymbol{l}}_{s_1|t}$ is adopted to obtain $\tilde{\boldsymbol{p}}_{\tau+1} := \boldsymbol{p}_t^1$. Similarly, when $\boldsymbol{p}_t^2$ is obtained using $\hat{\boldsymbol{l}}_{s_2|t}$, the virtual slot yields $\tilde{\boldsymbol{p}}_{\tau+2} := \boldsymbol{p}_t^2$ via $\tilde{\boldsymbol{l}}_{\tau+1} := \hat{\boldsymbol{l}}_{s_2|t}$. That is to say, at real slot t, for $n = 1, \ldots, |\mathcal{L}_t|$, each update from $\boldsymbol{p}_t^{n-1}$ to $\boldsymbol{p}_t^n$ using the estimated loss $\hat{\boldsymbol{l}}_{s_n|t}$ is mapped to an "update" at the virtual slot $\tau + n - 1$, where $\tilde{\boldsymbol{l}}_{\tau+n-1} := \hat{\boldsymbol{l}}_{s_n|t}$ is employed to obtain $\tilde{\boldsymbol{p}}_{\tau+n} := \boldsymbol{p}_t^n$ from $\tilde{\boldsymbol{p}}_{\tau+n-1} = \boldsymbol{p}_t^{n-1}$. According to this real-to-virtual slot mapping, we have $\tilde{\boldsymbol{p}}_{\tau+|\mathcal{L}_t|} = \boldsymbol{p}_{t+1}$; see also Fig. 4 for two examples. As we will show later, it is convenient to analyze the recursion between two consecutive $\tilde{\boldsymbol{p}}_\tau$ and $\tilde{\boldsymbol{p}}_{\tau+1}$, which is instrumental for our ensuing regret analysis.

With regard to DBGD, since multiple feedback rounds are possible per real slot t, we again map the $|\mathcal{L}_t|$ updates at a real slot [cf. (12)] to $|\mathcal{L}_t|$ virtual slots. The mapping is exactly the same as that in DEXP3, that is, per virtual slot τ, vector $\tilde{\boldsymbol{g}}_\tau$ is used to obtain $\tilde{\boldsymbol{x}}_{\tau+1}$.

From this real-to-virtual mapping vantage point, DEXP3 and DBGD can be viewed as (inexact) EXP3 and BGD iterations running on the virtual time horizon with only one feedback value per virtual slot. That is to say, instead of analyzing regret on the real time horizon, which can involve multiple feedback rounds, we can alternatively turn to the virtual slot, where there is only one "update" per slot.

5.2 Regret Analysis of DEXP3

Here we analyze the regret for DEXP3. The analysis builds on the following assumptions.

Assumption 1. *The losses satisfy* $\max_{t,k} l_t(k) \leq 1$*; and*

Assumption 2. *The delay* d_t *is bounded, i.e.,* $\max_t d_t \leq \bar{d}$.

Assumption 1 requires the loss function to be upper bounded, which is common in MAB; see also Hazan (2016); Auer et al. (2002); Bubeck et al. (2012). Assumption 2 asks for the delay to be bounded that also appears in previous analyses for the delayed online learning setup (Joulani et al., 2013; Cesa-Bianchi et al., 2018, 2016). Let us first consider the changes on $\tilde{p}_\tau(k)$ after one "update" in the virtual slot.

Lemma 1. *If the parameters are properly selected such that* $1 - \delta_2 - \eta\delta_1 \geq 0$*, the following inequality holds*

$$\frac{\tilde{p}_{\tau-1}(k)}{\tilde{p}_\tau(k)} \leq \frac{1}{1 - \delta_2 - \eta\delta_1}, \quad \forall k, \tau. \tag{13}$$

Proof. See Sec. B.1 of the supplementary document. □

Lemma 2. *If the parameters are properly selected such that* $1 - \eta\delta_1 \geq 0$*, the following inequality holds*

$$\frac{\tilde{p}_\tau(k)}{\tilde{p}_{\tau-1}(k)} \leq \max\left\{1 + \delta_2, \frac{1}{1 - \eta\delta_1}\right\}, \quad \forall k, \tau. \tag{14}$$

Proof. See Sec. B.2 of the supplementary document. □

Lemmas 1 and 2 assert that both $\tilde{p}_{\tau-1}(k)/\tilde{p}_\tau(k)$ and $\tilde{p}_\tau(k)/\tilde{p}_{\tau-1}(k)$ are bounded deterministically, that is, regardless of the arm selection and observed loss. These bounds are critical for deriving the regret.

To bound the regret, the final cornerstone is the "regret" in virtual slots, specified in the following lemma.

Lemma 3. *For a given sequence of* $\{\tilde{\boldsymbol{l}}_\tau\}_{\tau=1}^T$*, the following relation holds*

$$\sum_{\tau=1}^{T} (\tilde{\boldsymbol{p}}_\tau - \boldsymbol{p})^\top \min\{\tilde{\boldsymbol{l}}_\tau, \delta_1 \cdot \mathbf{1}\} \\ \leq \frac{T\ln(1+\delta_2) + \ln K}{\eta} + \frac{\eta}{2}\sum_{\tau=1}^{T}\sum_{k=1}^{K} \tilde{p}_\tau(k)\left[\tilde{l}_\tau(k)\right]^2 \tag{15}$$

where $\mathbf{1}$ *is a* $K \times 1$ *vector of all ones, and* $\boldsymbol{p} \in \Delta_K$.

Proof. See Sec. B.3 of the supplementary document. □

Lemma 3 can be viewed as the performance guarantee of DEXP3 without delay (since if there is no delay, what happens in virtual slots is exactly the same as what happens in real slots). Using Lemma 3, the DEXP3 regret follows.

Theorem 1. *Supposing Assumptions 1 and 2 hold, defining the overall delay* $D := \sum_{t=1}^{T} d_t$, *and choosing* $\delta_2 = \frac{1}{T+D}$, $\eta = \mathcal{O}\Big(\sqrt{\frac{1+\ln K}{K(T+D)}}\Big)$, *and* $\delta_1 = \frac{1}{2\eta \bar{d}} - \frac{\delta_2}{\eta}$, *DEXP3 guarantees that the* $\mathrm{Reg}_T^{\mathrm{MAB}}$ *in* (1) *satisfies*

$$\mathrm{Reg}_T^{\mathrm{MAB}} = \mathcal{O}\big(\sqrt{(T+D)K(1+\ln K)}\big). \qquad (16)$$

Proof. See Sec. B.4 of the supplementary document. □

Theorem 1 recovers the regret bound for delayed non-stochastic MAB with fixed (yet known) delay (Neu et al., 2010; Joulani et al., 2013).

5.3 Regret Analysis of DBGD

Our analysis builds on the following assumptions.

Assumption 3. *For any t, the loss function* f_t *is convex.*

Assumption 4. *For any t,* f_t *is L-Lipschitz and* β*-smooth.*

Assumption 5. *The feasible set contains* $\epsilon\mathcal{B}$, *where* $\mathcal{B}$ *is the unit ball, and* $\epsilon > 0$ *is a predefined parameter. The diameter of* $\mathcal{X}$ *is R; that is,* $\max_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{X}} \|\boldsymbol{x}-\boldsymbol{y}\| = R$.

Assumptions 3 - 5 are common in online learning (Hazan, 2016). Assumption 4 requires that $f_t(\cdot)$ is L-Lipschitz and β-smooth, conditions needed to bound the bias of the estimator $\boldsymbol{g}_{s|t}$ (Agarwal et al., 2010). Assumption 5 is also typical in BCO (Hazan, 2016; Agarwal et al., 2010; Flaxman et al., 2005; Duchi et al., 2015). In addition, the counterpart of Assumption 1 in BCO can readily follow from Assumptions 4 and 5.

To start, the quality of the gradient estimator $\boldsymbol{g}_{s|t}$ is first evaluated. As stated, when $f_t(\cdot)$ is not linear, the estimator $\boldsymbol{g}_{s|t}$ is biased, but its bias is bounded.

Lemma 4. *If Assumption 4 holds, then for every* $f_{s|t}(\boldsymbol{x}_{s|t})$, *the corresponding estimator* (10) *satisfies*

$$\|\boldsymbol{g}_{s|t}\| \le \sqrt{K}L, \text{ and } \|\boldsymbol{g}_{s|t} - \nabla f_{s|t}(\boldsymbol{x}_{s|t})\| \le \frac{\beta\delta}{2}\sqrt{K}. \quad (17)$$

Proof. See Sec. C.1 of the supplementary document. □

Lemma 4 suggests that with δ small enough, the bias of $\boldsymbol{g}_{s|t}$ will not be too large. Then, the following lemma specifies the relation among $\tilde{\boldsymbol{g}}_\tau$, $\tilde{\boldsymbol{x}}_\tau$, and $\tilde{\boldsymbol{x}}_{\tau+1}$, in a virtual time slot.

Lemma 5. *Under Assumptions 4 and 5, the update in a virtual slot* τ *guarantees that for any* $\boldsymbol{x} \in \mathcal{X}_\delta$, *we have*

$$\tilde{\boldsymbol{g}}_\tau^\top(\tilde{\boldsymbol{x}}_\tau - \boldsymbol{x}) \le \frac{\eta}{2}KL^2 + \frac{\|\tilde{\boldsymbol{x}}_\tau - \boldsymbol{x}\|^2 - \|\tilde{\boldsymbol{x}}_{\tau+1} - \boldsymbol{x}\|^2}{2\eta}. \qquad (18)$$

Proof. See Sec. C.2 of the supplementary document. □

Lemma 5 is the counterpart of Theorem 3.1 in Hazan (2016) for the non-delayed and full-information setting, which demonstrates that DBGD is BGD running on the virtual slots. Finally, leveraging these results, the regret bound follows next.

Theorem 2. *Suppose Assumptions 3 - 5 hold. Choosing* $\delta = \mathcal{O}\big(\frac{1}{T+D}\big)$, *and* $\eta = \mathcal{O}\big(\frac{1}{\sqrt{K(T+D)}}\big)$, *the DBGD guarantees that the regret is bounded, that is*

$$\mathrm{Reg}_T^{\mathrm{BCO}} = \sum_{t=1}^{T} f_t(\boldsymbol{x}_t) - \sum_{t=1}^{T} f_t(\boldsymbol{x}^*) = \mathcal{O}\big(\sqrt{K(T+D)}\big) \qquad (19)$$

where $D := \sum_{t=1}^{T} d_t$ *is the overall delay.*

Proof. See Sec. C.3 of the supplementary document. □

For the slightly different regret definition in Agarwal et al. (2010), DBGD achieves the same regret bound.

Corollary 1. *Upon defining* $\boldsymbol{x}_{t,0} := \boldsymbol{x}_t$ *and* $\boldsymbol{x}_{t,k} := \boldsymbol{x}_t + \delta\boldsymbol{e}_k$, *choosing* $\delta = \mathcal{O}\big(\frac{1}{T+D}\big)$, *and* $\eta = \mathcal{O}\big(\frac{1}{\sqrt{K(T+D)}}\big)$, *the DBGD also guarantees that*

$$\frac{1}{K+1}\sum_{t=1}^{T}\sum_{k=0}^{K} f_t(\boldsymbol{x}_{t,k}) - \sum_{t=1}^{T} f_t(\boldsymbol{x}^*) = \mathcal{O}\big(\sqrt{K(T+D)}\big).$$

Proof. See Sec. C.4 of the supplementary document. □

The $\mathcal{O}\big(\sqrt{K(T+D)}\big)$ regret in Theorem 2 and Corollary 1 recovers the bound of delayed online learning in the full information setup (Quanrud and Khashabi, 2015; Langford et al., 2009; Joulani et al., 2016) in terms of the delay dependence, while suffering from the dimension of the decision variables due to the bandit feedback.

6 NUMERICAL TESTS

In this section, experiments are conducted to corroborate the validity of the novel DEXP3 and DBGD schemes.

In synthetic data tests, we consider $T = 2,000$ slots. Delays are periodically generated with period $1, 2, 1, 0, 3, 0, 2$, with the delay of the last few slots slightly modified to ensure that all feedback arrives at the end of $T = 2,000$, resulting in an overall delay $D = 2,569$.

DEXP3 Synthetic Tests. Consider $K = 5$ arms, and losses generated with a sudden change. Specifically, for $t \in [1, 500]$, we have $l_t(k) = 0.4k|\cos t|$ per arm k, while for the rest of the slots, $l_t(k) = 0.2k|\sin(2t)|$. To benchmark the DEXP3, we use: i) the standard EXP3 for non-delayed MAB (Auer et al., 2002); ii) the BOLD for delayed MAB with known delay (Joulani et al., 2013); and,

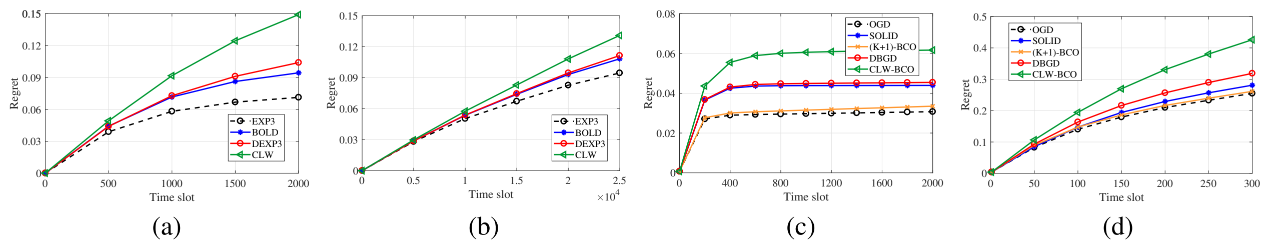

Figure 5: Tests of DEXP3 and DBGD: (a) Regret of DEXP3 using synthetic data; (b) Regret of DEXP3 using real data; (c) Regret of DBGD using synthetic data; (d) Regret of DBGD using real data.

iii) CLW to deal with the composite and anonymous feedback (Cesa-Bianchi et al., 2018). The instantaneous accumulated regret (normalized by T) versus time slots is plotted in Fig. 5 (a). The gap between BOLD and EXP3, illustrates that even with a known delay, the learner suffers from an extra regret. The small gap between DEXP3 and BOLD demonstrates that the biased estimator [cf. (5)] gives rise to larger regret, which is the price paid for the unknown delay. Compared with CLW, DEXP3 performs significantly better because it can leverage more information relative to the anonymous feedback used by CLW.

DEXP3 Real Tests. We also tested DEXP3 using the *Jester Online Joke Recommender System* dataset (Goldberg et al., 2001), where $T = 24,983$ users rate $K = 100$ different jokes from 0 (not funny) to 1 (very funny). The goal is to recommend one joke per slot t to amuse the users. The system performance is evaluated by $\boldsymbol{l}_t = (\mathbf{1} - \text{the score of this joke})$. In this test, we assign a random score in the range $[0, 1]$ for missing entries of this dataset. The delay is generated periodically as in the synthetic test, resulting in $D = 32,119$. Similar to the synthetic test, it can be observed in Fig. 5 (b) that DEXP3 incurs slightly larger regret than BOLD due to the unknown delay, but outperforms the recently developed CLW.

DBGD Synthetic Tests. Consider that $K = 5$, and the feasible set $\mathcal{X} \in \mathbb{R}^5$ is the unit ball $\mathcal{X} := \big\{\|\boldsymbol{x}\| \leq 1\big\}$. The loss function at slot t is generated as $f_t(\boldsymbol{x}) = a_t\|\boldsymbol{x}\|^2 + \boldsymbol{b}_t^\top \boldsymbol{x}$, where $a_t = \cos(3t) + 3$, while $b_t(1) = 2\sin(2t) + 1$, $b_t(2) = \cos(2t) - 2$, $b_t(3) = \sin(2t)$, $b_t(4) = 2\sin(2t) - 2$, and $b_t(5) = 2$. To assess the influence of the bandit feedback and the unknown delay, we consider the following benchmarks: i) the standard OGD (Zinkevich, 2003) in the full-information and non-delayed setting; ii) the $(K+1)$-point feedback BCO (Agarwal et al., 2010) for non-delayed BCO; iii) the SOLID for delayed full-information OCO (Joulani et al., 2016); and iv) the CLW-BCO with the inner algorithm relying on $(K+1)$-point feedback BCO (Cesa-Bianchi et al., 2018). Fig. 5 (c) depicts the instantaneous accumulated regret (normalized by T) versus time slots. This test shows that DBGD performs almost as well as SOLID, and the gap between DBGD/SOLID and OGD/$(K+1)$-BCO is due to the delay. The regret of DBGD is again considrably lower than that of CLW-BCO, demonstrating the efficiency of DBGD.

DBGD Real Tests. To further illustrate the merits of DBGD, we conduct tests dealing with online regression applied to a *yacht hydrodynamics* dataset (Dheeru and Karra Taniskidou, 2017), which contains $T = 308$ data with $K = 6$ features. Per slot t, the regressor $\boldsymbol{x}_t \in \mathbb{R}^6$ predicts based on the feature $\boldsymbol{w}_t$, before the measurement y_t is revealed. The loss function for slot t is $f_t(\boldsymbol{x}_t) = \frac{1}{2}(y_t - \boldsymbol{x}_t^\top \boldsymbol{w}_t)^2$. The delay is generated periodically as before, and cumulatively it is $D = 394$. The instantaneous accumulated regret (normalized by T) versus time slots is plotted in Fig. 5 (d). Again, DBGD outperforms CLW-BCO considerably. Comparing with the regret performance of DBGD and SOLID, we can safely deduce the influence delay has on bandit feedback.

7 CONCLUSIONS

Bandit online learning with unknown delays, including non-stochastic MAB and BCO, was studied in this paper. Different from settings where the experienced delay is known in bandit online learning, the unknown delay prevents a simple gradient estimate that is needed by the iterative algorithm. To address this issue, a biased loss estimator as well as a deterministic gradient estimator were developed for non-stochastic MAB and BCO, respectively. Leveraging the proposed estimators, the so-termed DEXP3 and DBGD algorithms were developed. An $\tilde{\mathcal{O}}\big(\sqrt{K(T+D)}\big)$ regret was analytically established for both DEXP3 and DBGD. Numerical tests on synthetic and real datasets confirmed the performance gain of DEXP3 and DBGD relative to state-of-the-art approaches.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive feedback. We also gratefully acknowledge the support from NSF grants 1500713, 1508993, and 1711471.

References

Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. In *Proc. Advances in Neural Info. Process. Syst.*, pages 873–881, Granada, Spain, 2011.

Alekh Agarwal, Ofer Dekel, and Lin Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *Proc. Intl. Conf. on Learning Theory*, pages 28–40, 2010.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

Baruch Awerbuch and Robert D Kleinberg. Adaptive routing with end-to-end feedback: Distributed learning and geometric approaches. In *Proc. ACM Symp. on Theory of Computing*, pages 45–53, Chicago, IL, June 2004.

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

Nicoló Cesa-Bianchi, Claudio Gentile, Yishay Mansour, and Alberto Minora. Delay and cooperation in nonstochastic bandits. *J. Machine Learning Res.*, 49:605–622, 2016.

Nicoló Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. Nonstochastic bandits with composite anonymous feedback. In *Proc. Conf. On Learning Theory*, pages 750–773, Stockholm, Sweden, 2018.

Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Proc. Advances in Neural Info. Process. Syst.*, pages 2249–2257, Granada, Spain, 2011.

Thomas Desautels, Andreas Krause, and Joel W Burdick. Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. *J. Machine Learning Res.*, 15(1):3873–3923, 2014.

Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

John Duchi, Michael I. Jordan, and Brendan McMahan. Estimation, optimization, and parallelism when data is sparse. In *Proc. Advances in Neural Info. Process. Syst.*, pages 2832–2840, Lake Tahoe, Nevada, 2013.

John C Duchi, Michael I. Jordan, Martin J Wainwright, and Andre Wibisono. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Trans. Inform. Theory*, 61(5):2788–2806, 2015.

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

Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001. URL `http://eigentaste.berkeley.edu/dataset/`.

Elad Hazan. Introduction to online convex optimization. *Found. and Trends® in Optimization*, 2(3-4):157–325, 2016.

Pooria Joulani, Andras Gyorgy, and Csaba Szepesvári. Online learning under delayed feedback. In *Proc. Intl. Conf. Machine Learning*, pages 1453–1461, Atlanta, 2013.

Pooria Joulani, András Gyórgy, and Csaba Szepesvári. Delay-tolerant online convex optimization: Unified analysis and adaptive-gradient algorithms. In *Proc. of AAAI Conf. on Artificial Intelligence*, volume 16, pages 1744–1750, Phoenix, Arizona, 2016.

John Langford, Alexander J Smola, and Martin Zinkevich. Slow learners are fast. *Proc. Advances in Neural Info. Process. Syst.*, pages 2331–2339, 2009.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proc. of the 19th Intl. Conf. on World Wide Web*, pages 661–670, Rayleigh, NC, 2010. ACM.

Brendan McMahan and Matthew Streeter. Delay-tolerant algorithms for asynchronous distributed online learning. In *Proc. Advances in Neural Info. Process. Syst.*, pages 2915–2923, Montreal, Canada, 2014.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. In *Proc. Intl. Conf. on Artificial Intelligence and Statistics*, pages 273–1282, Fort Lauderdale, Florida, 2017.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Gergely Neu, Andras Antos, András Gyórgy, and Csaba Szepesvári. Online markov decision processes under bandit feedback. In *Proc. Advances in Neural Info. Process. Syst.*, pages 1804–1812, Vancouver, Canada, 2010.

Ciara Pike-Burke, Shipra Agrawal, Csaba Szepesvari, and Steffen Grunewalder. Bandits with delayed anonymous feedback. *arXiv preprint arXiv:1709.06853*, 2017.

Kent Quanrud and Daniel Khashabi. Online learning with adversarial delays. In *Proc. Advances in Neural Info. Process. Syst.*, pages 1270–1278, Montreal, Canada, 2015.

Ohad Shamir and Liran Szlak. Online learning with local permutations and delayed feedback. In *Proc. Intl. Conf. Machine Learning*, pages 3086–3094, Sydney, Australia, 2017.

Claire Vernade, Olivier Cappé, and Vianney Perchet. Stochastic bandit models for delayed conversions. In *Proc. Conf. on Uncertainty in Artificial Intelligence*, Sydney, Australia, 2017.

Marcelo J Weinberger and Erik Ordentlich. On delayed prediction of individual sequences. *IEEE Trans. Inform. Theory*, 48(7):1959–1976, 2002.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proc. Intl. Conf. Machine Learning*, pages 928–936, Washington D.C., 2003.