Differentially Private Online Algorithms

Abinav Chari
Georgia Institute of Technology
achari7@gatech.edu

1 Introduction

Online learning is a powerful model for adapting to continuously arriving data. In this framework, a learning algorithm must make sequential decisions or predictions, refining its model in real-time as new information becomes available. Unlike traditional batch learning, where the algorithm has access to all data at once, online learning operates under the constraint that data arrives piece by piece, and decisions must be made immediately and potentially irreversibly. This paradigm is particularly valuable in dynamic environments such as recommendation systems, financial trading, and search engines, where real-time responses are crucial. However, as online systems increasingly rely on personal or sensitive data to optimize their performance, protecting user privacy has become a critical concern.

Differential privacy offers a rigorous mathematical framework to address these concerns. Introduced to provide strong, quantifiable privacy guarantees, differential privacy ensures that the output of a computation does not significantly change when a single individual's data is modified. In other words, whether or not a user's data is included, the algorithm's behavior remains statistically similar, limiting the risk of leaking sensitive information. While differential privacy has been extensively studied in offline settings—where all data is available beforehand—its application to online learning presents new challenges. In an online environment, decisions must be made at each step based on current and past data, and thus, protecting every output throughout the sequence, rather than a single batch result, requires more sophisticated techniques and often introduces additional noise or uncertainty into the system.

Consider the case of a major search engine like Bing or Google, which uses online learning to optimize the advertisements shown to users in real time. The engine continuously learns user preferences based on search history, demographic data, and interaction with past ads. Suppose a user searches for medical information about a sensitive condition, and the engine updates its model to start displaying related pharmaceutical ads. If these updates are not carefully protected, an advertiser—or even an external observer—could infer sensitive details about the user's health based on the sudden appearance of certain ads. Here, differential privacy becomes crucial: the learning algorithm must be designed so that the inclusion or exclusion of any single user's search history cannot substantially influence the ad selection process. This protects users from unintended information leaks, even as the system continues to learn and improve its performance in real time.

The goal of this project is to review recent work in differentially private online algorithms from a mainly theoretical standpoint. This paper will explore differentially private extensions of classical online algorithms, especially noting changes in regret guarantees

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

2 Preliminaries

There are several different online learning settings, each with different parameters such as amount of information feedback, reward schemes, and action spaces. This section will introduce the problem settings discussed throughout this paper and provide definitions for regret in each setting. The framework of differential privacy under continual observation will also be defined.

The basic online learning framework involves a T round repeated game between a learner and adversary. At the beginning of every round t, the learner picks an action $x_t \in X$ and the adversary simultaneously chooses a loss function l_t. Then, the learner incurs loss $l(x_t)$, and repeats in the next round. The regret of an online algorithm can be loosely thought of as the performance of the algorithm in comparison with the best fixed action in hindsight, but we will define this more formally in each problem setting. In all settings, however, the goal of most online algorithms is to achieve at least $o(T)$ (sublinear) regret, which indicates the algorithm approaches the best decision in hindsight over time.

2.1 Online Linear Optimization with Full Information

Online Linear Optimization (OLO) is an online learning setting in which the loss function in each round is linear with respect to x_t, i.e. $l_t(x_t) = \langle l_t, x_t \rangle$. Both l_t and x_t are vectors contstrained to the unit Euclidean sphere $\{x \in \mathbb{R}^n : \|x\|_2 \leq 1\}$. Finally, and importantly compared to the other problem settings explored in this paper, this setting has *full-information feedback*, meaning that in each round, the learner sees not only the incurred loss but the entire loss function, i.e. l_t. The regret of an online algorithm in this setting is defined to be

$$R_T = \mathbb{E}[\sum_{t=1}^{T} \langle l_t, x_t \rangle - \min_{x \in X} \sum_{t=1}^{T} \langle l_t, x \rangle]$$

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

2.2 Stochastic Multi-armed Bandit

The bandit setting differs from the previous in one key aspect: rather than seeing the entire loss function, the learner only observes the incurred loss after each round. In this sense, this problem setting is more "difficult" for the learner than the full-information setting. To make the problem feasible, more assumptions are placed on other aspects of the problem. In the stochastic multi-armed bandit setting (MAB), at each round t, the learner selects an arm a from a finite set of K arms, and receives a reward $r_{a,t}$ which is drawn i.i.d. from a distribution with support $[0, 1]$ and mean μ_a. Importantly, in this setting, the loss functions picked by the adversary are not directly "adversarial"; they follow a strict pattern. Furthermore, due to the stochastic structure of rewards, we use expected regret or pseudoregret to evaluate the performance of the algorithm, formally defined as follows: let a^* denote the arm with the highest mean reward, and μ^* be the corresponding mean. Then, the pseudoregret is defined as

$$\bar{R}_T = \sum_{t=1}^{T} (\mu^* - \mu_{a,t})$$

where $\mu_{a,t}$ is the mean of the arm pulled by the algorithm in round t. This can also be defined in an alternate way: let $N(a)$ denote the number of times the algorithm pulls arm a. Then, the pseudoregret can be equivalently defined as

$$\bar{R}_T = \sum_{a \neq a^*} N(a)(\mu^* - \mu_a)$$

Intuitively, this definition counts how many times the algorithm pulls a suboptimal arm and incurs the expected difference between pulling that arm and the optimal.

2.3 Stochastic Linear Bandit

The stochastic linear bandit problem is similar to the stochastic MAB problem, except with a well-structured action space. In each round t, the learner plays $a_t \in A \subseteq \mathbb{R}^d$ and receives a reward $r_t = \langle \theta_*, a_t \rangle + \eta_t$ for some predetermined $\theta_* \in \mathbb{R}^d$ and zero-mean, independent noise η_t. This problem setting is somewhat of a combination of the previous two, taking the limited feedback and stochastic properties of the bandit setting along with the predetermined structure of the OLO setting. Regret in this setting is defined as

$$R_T = T \max_{a \in A} \langle \theta_*, a \rangle - \sum_{t=1}^{T} \langle \theta_*, a_t \rangle$$

2.4 Differential Privacy under Continual Observation

Differential privacy is a framework for guarantees on privacy loss given participation in a dataset.

Definition 2.1 ((ϵ, δ)-Differential Privacy)**.** A randomized online learning algorithm $\mathcal{A}$ on the action set $\mathcal{X}$ and the loss set $\mathcal{Y}$ is (ε, δ)-*differentially private* if for any two sequences of loss vectors $L = (\ell_1, \ldots, \ell_T) \subseteq \mathcal{Y}^T$ and $L' = (\ell'_1, \ldots, \ell'_T) \subseteq \mathcal{Y}^T$ differing in at most one vector - that is to say, $\exists t_0 \in [T]$ such that $\forall t \in [T] \setminus \{t_0\}$, $\ell_t = \ell'_t$ - for all subsets $S \subseteq \mathcal{X}^T$, it holds that

$$\mathbb{P}(\mathcal{A}(L) \in S) \leq e^{\varepsilon} \mathbb{P}(\mathcal{A}(L') \in S) + \delta.$$

This definition is specific to online algorithms; it ensures "plausible deniability" in that a specific sequence of decisions made by the algorithm does not necessarily imply the identity of a specific loss vector. It also entails that the entire sequence of predictions is differentially private.

3 OLO with Differential Privacy

Algorithm 1 FTRL Template for OLO

Require: Noise distribution $\mathcal{D}$, Regularization $R(x)$

Initialize an empty binary tree B to compute differentially private estimates of $P_{s=1}^{t} l_s$.
Sample $n_1^0, \ldots, n_{\lceil \log T \rceil}^0$ independently from $\mathcal{D}$.
$\widetilde{L}_0 \leftarrow \sum_{i=1}^{\lceil \log T \rceil} n_i^0$
for $t = 1$ to T **do**
    Choose $x_t = \arg\min_{x \in \mathcal{X}} \left[\eta \langle x, \widetilde{L}_{t-1} \rangle + R(x) \right]$
    Observe $l_t \in \mathcal{Y}$ and suffer a loss of $\langle l_t, x_t \rangle$
    $(\widetilde{L}_t, B) \leftarrow \text{TreeBasedAgg}(l_t, B, t, \mathcal{D}, T)$
end for=0

The algorithm presented for differentially private OLO is an extension of the classical Follow-the-Regularized-Leader (FTRL) algorithm. Specifically, a tree-based aggregate noising scheme is used to noise the inputs enough to ensure differential privacy. However, to simplify the regret bound proof, it is assumed that the added noise is injected in a one-shot manner at the beginning of the algorithm: similar to the regret bound proof of Follow-the-Perturbed-Leader (FTPL), the expected loss of the original algorithm and the simplified one are the same. The main theorem is as follows: Let $\|\mathcal{X}\|_p = \max\{\|l_t\|_p : l_t \in \mathcal{X}\}$ and $\text{Lap}^N(\lambda)$ be the Laplace distribution over $\mathbb{R}^N$ with each coordinate drawn i.i.d. from the Laplace distribution with λ parameter.

Theorem 1. Algorithm 1, when run with $\mathcal{D} = \text{Lap}^N(\lambda)$ where $\lambda = \frac{\|\mathcal{X}\|_1 \log T}{\varepsilon}$, regularization $R(x)$, decision set $\mathcal{X}$, and loss vectors $l_1, \ldots, l_T$, has regret bounded by:

$$\text{Regret} \leq \sqrt{D_R \sum_{t=1}^{T} \max_{x \in \mathcal{X}} \left(\|l_t\|^*_{\nabla^2 R(x)} \right)^2} + D_{\text{Lap}},$$

where:

$$D_{\text{Lap}} = \mathbb{E}_{Z\sim\mathcal{D}'}\left[\max_{x\in\mathcal{X}}\langle Z, x\rangle - \min_{x\in\mathcal{X}}\langle Z, x\rangle\right],$$
$$D_R = \max_{x\in\mathcal{X}} R(x) - \min_{x\in\mathcal{X}} R(x),$$

and $\mathcal{D}'$ is the distribution induced by the sum of $\lceil \log T \rceil$ independent samples from $\mathcal{D}$, and $\|\cdot\|^*_{\nabla^2 R(x)}$ denotes the dual norm with respect to the Hessian of R. Moreover, the algorithm is ε-differentially private, i.e., the sequence of predictions $(x_t : t \in [T])$ is ε-differentially private.

The proof of the privacy guarantee follows from two important facts: the tree-based aggregate algorithm produces an ϵ-differentially private sum of losses, and that any post-processing done on differentially private data is still differentially private. The details aren't explained in the paper, but the second makes sense intuitively - as long as the information is private, any processing done without knowledge of the original data cannot further reduce the privacy of the data. From this, the differential privacy of the algorithm follows immediately.

For the regret guarantee, we follow a similar proof to the FTPL regret guarantee (using the Be-the-Leader lemma). Let $Z \sim D$, where Z is the one-shot noise injection added at the beginning of the algorithm, and let $\hat{x}_t$ be the sequence of iterates for the one-shot noise algorithm. Similar to the FTPL proof, we have that

$$\mathbb{E}_{Z_1,\ldots,Z_T\sim D}[\sum_{t=1}^{T}\langle l_t, x_t\rangle] = \mathbb{E}_{Z\sim D}[\sum_{t=1}^{T}\langle l_t, \hat{x}_t\rangle]$$

This follows from the fact that x_t and $\hat{x}_t$ have the same distribution. Thus, it suffices to prove the regret bound for the alternate algorithm's sequence of iterates. From here, we simply follow the FTL-BTL template proof, because, for this algorithm, the noise added at the beginning is constant and thus does not affect the stability of the algorithm. Thus, define

$$l_0(x) = \frac{1}{\eta}R(x) + \langle Z, x\rangle$$

Using the BTL lemma, we have that, for all $u \in X$,

$$\sum_{t=0}^{T} l_t(u) \geq \sum_{t=0}^{T} l_t(\hat{x}_{t+1})$$

Therefore, we can conclude that

$$\sum_{t=1}^{T}[\ell_t(\hat{x}_t) - \ell_t(u)] \leq \sum_{t=1}^{T}[\ell_t(\hat{x}_t) - \ell_t(\hat{x}_{t+1})] + \ell_0(u) - \ell_0(\hat{x}_1) \leq \sum_{t=1}^{T}[\ell_t(\hat{x}_t) - \ell_t(\hat{x}_{t+1})] + \frac{1}{\eta}D_R + D_Z$$

where

$$D_Z = \max_{x\in\mathcal{X}}\langle Z, x\rangle - \min_{x\in\mathcal{X}}\langle Z, x\rangle.$$

From here, we only need to bound the stability term, which is exactly the same as the standard FTRL proof.

Notice that the bound only depends on the change in the cumulative loss per step, i.e., $\eta\left(\sum_t \ell_t + Z\right)$, for which the change is the loss vector $\eta\ell_{t+1}$ across time steps. Therefore, we get that

$$\ell_t(\hat{x}_t) - \ell_t(\hat{x}_{t+1}) \leq \max_{x\in\mathcal{X}} \|\eta\ell_t\|^2_{\eta\nabla^{-2}R(x)}.$$

Using this in the earlier equation, we get our result. Furthermore, with $\|\mathcal{Y}\|_1 = \sqrt{N}$, choosing $R(x) = \|x\|_2^2$ and $\lambda = \frac{N \log T}{\epsilon}$, our regret bound becomes

$$R_T \leq O(\sqrt{T} + \frac{N \log^2 T}{\epsilon})$$

This is interesting, because it shows that for any $\epsilon \geq O(\frac{1}{\sqrt{T}})$, our regret bound has the same scaling; in other words, this algorithm ensures a level of "free" privacy! This underlines a deep connection between adversarial robustness and differential privacy; the idea behind many adversarial robustness techniques in online algorithms involve decoupling the actions of the algorithm and the previous data to reduce exploitability - and as a byproduct, it becomes more difficult to infer tke values of specific loss vectors based on the actions of the algorithm. This connection is especially clear in the FTPL algorithm, for which we perturb the loss aggregate in a very similar way to DP to induce robustness.

4 Stochastic Multi-armed Bandit with Differential Privacy

The stochastic multi-armed bandit (MAB) problem with two arms can be reduced to what is known as the stopping rule problem: given a stream of i.i.d. samples drawn from a distribution over known support with unknown mean μ, the goal is to find a $(1+\alpha)$-approximation of μ with as few samples as possible. More formally, a (α, β)-stopping rule terminates at some time t^* and returns $\hat{\mu}$ such that

$$Pr[|\hat{\mu} - \mu| > \alpha|\mu|] < \beta$$

Thus, to design a differentially private MAB algorithm, we first design a differentially private stopping rule. This algorithm is an adaptation of Nonmonotonic Adaptive Sampling (NAS). For reference, the vanilla NAS algorithm uses a confidence interval $h_t = 1 - \frac{\beta}{2t^2}$ derived from the Hoeffding bound, and halts for the first t for which $|\bar{X}_t| \geq h_t(\frac{1}{\alpha} + 1)$. In order to make a differentially private version of this algorithm, we apply the sparse vector technique, which is a popular DP algorithm which operates on a series of queries and releases the identity of the first query which satisfies a condition. This algorithm fits perfectly with the stopping rule problem, since it essentially asks a series of queries. The strength of the sparse vector technique is that it incurs a fixed total privacy cost no matter how many queries are considered.

Definition 4.1 (Sparse Vector Technique (SVT))**.** The algorithm preserves differential privacy by sometimes returning the wrong index; sometimes, the index returned may be for a query whose result does not exceed the threshold, and sometimes, the index may not be the first whose query result exceeds the threshold.

Algorithm 2 DP-NAS

Set $\sigma_1 \leftarrow \frac{12R}{\epsilon}$, $\sigma_2 \leftarrow \frac{12R}{\epsilon}$, $\sigma_3 \leftarrow \frac{4R}{\epsilon}$
Sample $B \sim \text{Lap}(\sigma_1)$
Initialize $t \leftarrow 0$
repeat
  $t \leftarrow t + 1$
  $A_t \sim \text{Lap}(\sigma_2)$
  Get a new sample X_t and update the mean X_t
  $h_t \leftarrow R\sqrt{2t \log\left(\frac{16t^2}{\beta}\right)}$
  $c_t \leftarrow \sigma_1 \log\left(\frac{4}{\beta}\right) + \sigma_2 \log\left(\frac{8t^2}{\beta}\right) + \frac{\sigma_3}{\alpha} \log\left(\frac{4}{\beta}\right)$
until $|X_t| \geq \frac{h_t(1+\frac{1}{\alpha}) + c_t + B + A_t}{t}$
Sample $L \sim \text{Lap}(\sigma_3)$
return $X_t + L$
=0

Algorithm 3 DP Successive Elimination

Input: K arms, confidence β, privacy-loss ε.

Let $S \leftarrow \{1, \ldots, K\}$
Initialize: $t \leftarrow 0$, $epoch \leftarrow 0$
repeat
  $epoch \leftarrow epoch + 1$
  $r \leftarrow 0$
  Zero all means: $\forall i \in S$ set $\bar{\mu}_i \leftarrow 0$
  $\Delta_e \leftarrow 2^{-epoch}$
  $R_e \leftarrow \max\left(\frac{32\log\left(\frac{8|S|\cdot epoch^2}{\beta}\right)}{\Delta_e^2}, \frac{8\log\left(\frac{4|S|\cdot epoch^2}{\beta}\right)}{\varepsilon\Delta_e}\right) + 1$
  while $r < R_e$ **do**
    $r \leftarrow r + 1$
    for all $i \in S$ **do**
      $t \leftarrow t + 1$
      Sample reward of arm i, update mean $\bar{\mu}_i$
    end for
  end while
  $h_e \leftarrow \sqrt{\frac{\log\left(\frac{8|S|\cdot epoch^2}{\beta}\right)}{2R_e}}$
  $c_e \leftarrow \frac{\log\left(\frac{4|S|\cdot epoch^2}{\beta}\right)}{R_e\varepsilon}$
  for all $i \in S$ **do**
    $\mu_i^e \leftarrow \bar{\mu}_i + \text{Lap}\left(\frac{1}{\varepsilon r}\right)$
  end for
  $\mu_{\max}^e \leftarrow \max_{i\in S} \mu_i^e$
  for all $j \in S$ **do**
    if $\mu_{\max}^e - \mu_j^e > 2h_e + 2c_e$ **then**
      Remove arm j from S
    end for
  end for
until $|S| = 1$
Pull the arm in S for all remaining rounds =0

Algorithm 2 is an ϵ-DP (α, β)-stopping rule. The privacy of the algorithm immediately follows from the use of the sparse vector technique: the sampling noise and query noise ensure $\frac{\epsilon}{2}$-DP while the final addition of noise releases the mean with $\frac{\epsilon}{2}$-DP. Since DP is additive, we maintain ϵ-DP. To prove the validity of the stopping rule, we apply a union bound over the Hoeffding bound for $|\bar{X}_t - \mu| > h_t$ and concentration bounds over the Laplace distribution to show that, with probability at least $1 - \beta$, $|\bar{X}_t - \mu| + \frac{|L|}{t} \leq \alpha|\mu|$.

The paper also introduces a technique to exponentially reduce the number of SVT queries by querying the average at exponentially growing intervals, which is shown to only pay a constant factor in utility. This idea is essential to solving the scaling problem of applying the stopping rule to MAB. As mentioned earlier, the stopping rule can only be used to solve MAB directly when $K = 2$; if $K > 2$, the problem becomes a successive elimination (SE) problem, where we must compare the confidence bound of each arm with every other arm. Using DP-NAS, we naively would have to instantiate $\binom{K}{2}$ stopping rules for every pair of arms. Since every pull of an arm contributes to $K - 1$ stopping rules/SVT instantiations, each SVT would have to be scaled down to $\frac{\epsilon}{K}$ privacy, thus incurring an extra K factor in the regret.

The MAB algorithm based on DP-NAS partitions arm pulls into epochs of exponentially increasing lengths. In each epoch, all arms with a certain optimality gap are removed, and the mean estimation restarts with new observations. The important detail is that, because we restart the mean estimation,

each reward only affects the mean estimation in one epoch, which allows for ϵ-DP mean estimation in each epoch, avoiding the K scaling factor.

The privacy guarantee follows easily by considering two streams which differ in only one reward. Since each reward only affects a single epoch and the difference in reward is less than 1, the difference of the means of arm a in the two streams is less than $\frac{1}{R_e}$. Thus, adding noise distributed by $Lap(\frac{1}{\epsilon R_e})$ to μ_a ensures ϵ-DP.

To argue the optimality of this algorithm, we upper bound the number of times a suboptimal arm is pulled. Specifically,

Lemma 4.1 Fix any instance of the K-armed bandit (K-MAB) problem, and denote by a^* the optimal arm with the highest mean reward. Let $\Delta_a = \mu_{a^*} - \mu_a$ be the gap between the mean reward of the optimal arm and that of any suboptimal arm $a \neq a^*$. Fix a time horizon T. Then, with probability at least $1 - \beta$, Algorithm 3 pulls each suboptimal arm $a \neq a^*$ no more than

$$\min \left\{ T, \mathcal{O}\left(\frac{\log(K/\beta) + \log\log(1/\Delta_a)}{\Delta_a^2} + \frac{1}{\epsilon \Delta_a} \right) \right\}$$

times.

The proof of this lemma follows a similar proof to the DP-NAS utility proof in that we take a union bound over Hoeffding bounds over all arm pulls within an epoch, then take another union bound over all epochs to ensure that, for the entire duration of the algorithm, we have that $|\mu_a^c - \mu_a| \leq h_e + c_e$ with high probability. Then, assuming that this holds, it is shown that the optimal arm is never eliminated and that, in each epoch, any arm with an optimality gap above 2^{-epoch} is eliminated. With this lemma, it follows that the regret bound is

$$\bar{R}_T \leq O(\sum_{a \neq a^*} \frac{\log(T)}{\Delta_a} + \frac{K \log T}{\epsilon})$$

5 Stochastic Linear Bandit with Differential Privacy

The algorithm described in (Hanna et. al 2022) follows a similar structure to LinUCB in that it maintains a set of "good" actions which almost certainly contain the optimal action a^* through least squares estimation of θ_*, while gradually eliminating suboptimal actions. However, instead of an ellipsoid, this algorithm creates discrete "nets" over the continuous action space which approximately preserve distance.

Definition 5.1 (ζ-nets)**.** For any set $A \subseteq \{x \in \mathbb{R}^d \mid \|x\|_2 \leq 1\}$ that spans $\mathbb{R}^d$, there exists a set $N_\zeta \subseteq A$ (called a ζ-*net*) with cardinality at most $\left(\frac{3}{\zeta}\right)^d + d$ such that N_ζ spans $\mathbb{R}^d$ and for any $a \in A$, there exists some $a' \in N_\zeta$ such that $\|a' - a\|_2 \leq \zeta$.

The main idea behind this algorithm is that playing a small number of distinct actions requires less noise to be added than taking a large number of distinct actions. Because privacy entails adding noise to rewards of distinct actions, using a smaller set of actions allows less overall noise to be added to ensure privacy. Thus, for each batch, the algorithm selects a small subset of the current ζ-net as "good" actions and only plays those for the duration of the batch. Then, using these to estimate θ_*, we eliminate all underperforming arms in the original, continuous action space from consideration, and continue to the next epoch.

In order to find a suitable subset to find a good estimation of $\langle \theta_*, a \rangle \; \forall a \in A$, we employ the concept of a core set, which essentially is a small set of "representative" vectors which, along with a distribution over the core set, enables calculation of a good estimate of $a^T \theta_*$.

Lemma 2 (Core set for $\mathcal{A}$). Let $\mathcal{A} \subset \{x \in \mathbb{R}^d \mid \|x\|_2 \leq 1\}$ be a finite set of actions that spans $\mathbb{R}^d$. Then there exists a subset $\mathcal{C} \subset \mathcal{A}$ of size at most $O(d)$, and a distribution π on $\mathcal{C}$ such that for any

$a \in \mathcal{A}$,

$$a^\top \left(\sum_{\alpha \in \mathcal{C}} \pi(\alpha) \alpha \alpha^\top \right)^{-1} a \leq 2d.$$

Moreover, $\mathcal{C}$ and π can be found in polynomial time in d.

This lemma implies that the least squared estimator using only actions in the core set will not vary too much from the estimator using actions in the whole action space, which is important for the regret bound proof. Furthermore, we only add $O(d)$ variables of $Lap(\frac{1}{\epsilon})$ noise, which results in a better estimate of θ_* compared to adding noise to all rewards in the original net.

Algorithm 4 DP algorithm for stochastic linear bandits

Input: set of actions $\mathcal{A}$, time horizon T, and privacy parameter ε.
Let $\mathcal{A}_1$ be a ζ-net for $\mathcal{A}$ as in Lemma 1, with $\zeta = \frac{1}{T}$.
$q \leftarrow (2T)^{1/\log T}$
for $i = 1$ to $\log(T) - 1$ **do**
  $\gamma_i \leftarrow \frac{q^4 d\sqrt{q^i \log(4|\mathcal{A}_i|T^2)} + 2Bd^2 + 2d\log(4|\mathcal{A}_i|T^2)}{\varepsilon q^i}$
  For $\mathcal{A}_i \subseteq \mathbb{R}^m$, $m \leq d$, let C_i be a core set of size at most Bm as in Lemma 2 and π_i the associated distribution.
  for each action $a \in C_i$ **do**
    Pull a, $n_{ia} = \lceil \pi_i(a) q^i \rceil$ times to get rewards $r_{ia}^{(1)}, \ldots, r_{ia}^{(n_{ia})}$.
    $\bar{r}_{ia} \leftarrow \sum_{k=1}^{n_{ia}} r_{ia}^{(k)}, \quad \hat{r}_{ia} \leftarrow \bar{r}_{ia} + z_{ia}$, where $z_{ia} \sim \text{Lap}(\frac{1}{\varepsilon})$
  end for
  $V \leftarrow \sum_{a \in C_i} n_{ia} a a^\top, \quad \hat{\theta}_i \leftarrow V^{-1} \sum_{a \in C_i} \hat{r}_{ia} a$
  $\mathcal{A}_{i+1} \leftarrow \{a \in \mathcal{A}_i \mid \langle a, \hat{\theta}_i \rangle \geq \max_{\alpha \in \mathcal{A}_i} \langle \alpha, \hat{\theta}_i \rangle - 2\gamma_i\}$
end for
Play action $\arg\max_{\alpha \in \mathcal{A}_{\log(T)-1}} \langle \alpha, \hat{\theta}_{\log(T)-1} \rangle$ for the remaining time. =0

At a high level, this algorithm first initializes A_1 to be a ζ-net for A. Then, for batches of exponentially growing length, we construct a core set for A_i and find the associated distribution $\pi_i A_i$. Each action in the core set is pulled according to the associated distribution, and the rewards for each action are summed and noised according to the Laplace distribution. The algorithm uses these observations to create a least squared estimate for θ_*, and updates the current action set by removing the actions which are not within some confidence interval of the best action with respect to the current θ_* estimate. This continues until $\log T$ iterations, at which time the algorithm only plays the best action according to the current estimate for the remainder of rounds.

Showing that this algorithm is ϵ-DP is trivial - by the Laplace mechanism, each observation receives enough noise to ensure the reward aggregates are sufficiently private, and since any post-processing has the same level of privacy, the total privacy follows. As for the regret bound, the general proof outline is as follows: first, it is shown that, with probability $1 - \frac{1}{T}$, $\forall i, a \in A_i$, we have that $|a, \hat{\theta}_i - \theta_*| \leq \gamma_i$, where $\hat{\theta}_i$ is the estimate for θ in the i'th epoch and γ_i is the confidence interval. Then, assuming this event occurs, any action with a gap greater than $2\gamma_i$ are eliminated in each batch, which means that each action still present in the i'th batch have gap at most $4\gamma_i$. The regret bound for this algorithm is

$$R_T \leq O(\sqrt{T \log T} + \frac{\log^2 T}{\epsilon})$$

6 Conclusion

There are a couple of common trends occurring throughout differentially private online algorithm literature. Firstly, as expected, the majority of algorithms are simple extensions of classical online algorithms in each setting, such as FTRL, Successive Elimination, and LinUCB. However, even

though the algorithm design may be intuitive, the optimality proofs tend to be quite complex. Furthermore, in every setting, there is some notion of "free privacy", where having some level of privacy does not increase the regret bound. This suggests not only a connection between adversarial robustness and differential privacy, but also some connection between nondeterministic algorithms and differentially private algorithms. To illustrate this, note how a fully deterministic algorithm like FTL has very coupled rewards and actions - since the sequence of rewards fully determine the next action, it becomes easier to guess what a specific reward is based on the actions of the algorithm. By introducing randomness into the algorithm - which in classical online learning is to induce robustness to adversaries and oblivious noise - some innate privacy is established.

References

[1] Naman Agarwal, Karan Singh, "The Price of Differential Privacy For Online Learning", ICML 2017

[2] Touqir Sajed, Or Sheffet, "An Optimal Private Stochastic-MAB Algorithm Based on an Optimal Private Stopping Rule", ICML 2025

[3] Osama Hanna et al, "Differentially Private Stochastic Linear Bandits: (Almost) for Free", IEEE 2024