JMLR: Workshop and Conference Proceedings vol (2012) 1–34

Differentially Private Online Learning

Prateek Jain PRAJAIN@MICROSOFT.COM
Microsoft Research India

Pravesh Kothari KOTHARI@CS.UTEXAS.EDU
University of Texas at Austin

Abhradeep Thakurta AZG161@CSE.PSU.EDU
Pennsylvania State University

Abstract

In this paper, we consider the problem of preserving privacy in the context of online learning. Online learning involves learning from data in real-time, due to which the learned model as well as its predictions are continuously changing. This makes preserving privacy of each data point significantly more challenging as its effect on the learned model can be easily tracked by observing changes in the subsequent predictions. Furthermore, with more and more online systems (e.g. search engines like Bing, Google etc.) trying to learn their customers' behavior by leveraging their access to sensitive customer data (through cookies etc.), the problem of privacy preserving online learning has become critical.

We study the problem in the framework of online convex programming (OCP)—a popular online learning setting with several theoretical and practical implications—while using differential privacy as the formal measure of privacy. For this problem, we provide a generic framework that can be used to convert any given OCP algorithm into a private OCP algorithm with provable privacy as well as regret guarantees (utility), provided that the given OCP algorithm satisfies the following two criteria: 1) linearly decreasing sensitivity, i.e., the effect of the new data points on the learned model decreases linearly, 2) sub-linear regret. We then illustrate our approach by converting two popular OCP algorithms into corresponding differentially private algorithms while guaranteeing $\tilde{O}(\sqrt{T})$ regret for strongly convex functions. Next, we consider the practically important class of online linear regression problems, for which we generalize the approach by Dwork et al. (2010a) to provide a differentially private algorithm with just poly-log regret. Finally, we show that our online learning framework can be used to provide differentially private algorithms for the offline learning problem as well. For the offline learning problem, our approach guarantees *better* error bounds and is more practical than the existing state-of-the-art methods (Chaudhuri et al., 2011; Rubinstein et al., 2009).

1. Introduction

With the continuous increase in the amount of computational resources available, modern websites and online systems can now, in real time, process large amounts of potentially sensitive information gathered from their customers. Although in most cases, these websites intend to just leverage real-time learning using their customers' data, it might actually compromise their customers' privacy.

© 2012 P. Jain, P. Kothari & A. Thakurta.

For example, consider the following scenario in the context of sponsored search. Sponsored search advertisements (ads) are served with organic search results and form a major source of revenue for search engines. To serve these ads effectively, search engines attempt to learn the relevance of an ad for a user. For this purpose, search engines typically store users' profile information, e.g., gender of the user. Now, suppose a male user clicks an ad, through which the search engine learns the rule "males like this ad". This rule is directly observable by the corresponding advertiser whose ad was clicked. To do this, the advertiser makes two profiles, one that reports the gender as male and the other one as female. He now compares the rank of his ad presented by the search engine to each of his profile and observes that the ad is presented at the top of other ads for the male profile. Also, the advertiser can obtain the IP address of the user, as the user clicked the ad. Thus, he can make a direct association between the user and his gender, compromising the user's privacy. Similar examples can be constructed for several other online learning domains such as, *online portfolio management* (Kalai and Vempala, 2005), *online linear prediction* (Hazan et al., 2007) etc.

In this paper, we address privacy concerns in online learning scenarios similar to the examples mentioned above. Specifically, we provide a generic framework for privacy preserving online learning. We use *differential privacy* (Dwork et al., 2006b) as the formal notion of privacy, and use *online convex programming* (OCP) (Zinkevich, 2003) as the formal online learning model. Note that OCP is a popular online learning paradigm and includes several online learning problems faced by real-life systems. Examples include online logistic regression, online linear regression etc.

Differential privacy (DP) is a popular privacy notion with a sound cryptographic foundation and has recently been used in the context of several learning problems (Chaudhuri et al., 2011; Williams and McSherry, 2010; Rubinstein et al., 2009). At a high level, a differentially private learning algorithm guarantees that its output does not change "too much" because of perturbations in any individual training data point. Now, a trivial way to ensure this is by providing a random/constant output that is completely independent of the input. However, such an output does not have any "utility" or "goodness" such as bounded generalization error.

Hence, a differentially private algorithm is measured with respect to two criteria: 1) Privacy and 2) Utility. Most of the existing results in differentially private learning have focused on the offline setting only, where all the training data is available beforehand. Hence, both privacy and utility need to be argued only over one final output.

In contrast, in the online learning setting, data arrives online[1] (e.g. user queries and clicks) and the algorithm has to provide an output (e.g. relevant ads) at each step. Hence, the number of outputs produced is the same as the size of the entire dataset. To guarantee differential privacy, one has to analyze the privacy of the *complete* sequence of outputs produced, thereby making privacy preservation a significantly harder problem. For utility, we need to show that asymptotically the algorithm is at least as good as the optimal offline solution, i.e., the algorithm has sub-linear *regret*.

In this paper, we study both privacy and utility aspects of privacy preserving online learning in the online convex programming (OCP) model. The goal is to provide differ-

[1] At each time step one data entry arrives.

entially private OCP algorithms with sub-linear regret. To this end, we provide a generic framework to convert any OCP algorithm into a differentially private algorithm with sub-linear regret, provided that the algorithm satisfies two criteria: a) linearly decreasing sensitivity (see Definition 3), b) sub-linear regret.

Next, we instantiate our generic framework with two popular OCP algorithms: Implicit Gradient Descent (IGD) by Crammer et al. (2006); Kulis and Bartlett (2010) and Generalized Infinitesimal Gradient Ascent (GIGA) by Zinkevich (2003). Our algorithms guarantee differential privacy as well as $\tilde{O}(\sqrt{T})$ regret for a fairly general class of strongly convex functions with Lipschitz continuous gradients. In fact, we show that IGD can be used with our framework for non-differentiable functions as well. We also show that if the cost functions are quadratic (e.g. online linear regression), we can use another OCP algorithm called Follow The Leader (FTL) along with a generalization of a technique by Dwork et al. (2010a) to guarantee $O(\ln^{1.5} T)$ regret while preserving privacy.

Finally, our generic framework can be used to obtain privacy preserving algorithms for a large class of *offline* learning problems as well. In particular, we show that our private OCP framework can be used to obtain generalization error bounds for various offline learning problems using techniques of Kakade and Tewari (2008) (see Section 5). Our differentially private offline learning framework provide better error bounds and is more practical than the existing state-of-the-art methods (Chaudhuri et al., 2011; Rubinstein et al., 2009).

1.1. Our Contributions

Following are the main contributions of this paper:

1. We formalize the problem of differentially private OCP, and provide a generic framework for the same with provable privacy and utility (regret) guarantees. (see Section 3).
2. We instantiate our framework with two popular OCP algorithms: Implicit Gradient Descent (IGD) and Generalized Infinitesimal Gradient Ascent (GIGA). For both the algorithms we provide privacy guarantees and $\tilde{O}(\sqrt{T})$ regret. To guarantee privacy, we need to show that the effect of any data entry on the output of any of the algorithms (at time step t) *decreases linearly in* t. This stability bound is of independent interest and has implications for connections between online learning and stability (Ross and Bagnell, 2011; Poggio et al., 2011).
3. For the special case of privacy preserving online linear regression problem, we improve the regret bound to $O(\ln^{1.5} T)$ by exploiting techniques from Dwork et al. (2010a).
4. In Section 5 we show that our differentially private framework for OCP can be used to solve a large class of offline learning problems as well, for which our method provides better guarantees than the existing state-of-the-art results (Chaudhuri et al., 2011; Rubinstein et al., 2009).
5. Finally, through empirical experiments on benchmark datasets, we demonstrate practicality of our algorithms for two popular learning problems: online linear regression and online logistic regression (see Appendix G).

1.2. Related Work

As more and more personal data is being digitized, privacy has become a critical issue. Over the years, several ad-hoc privacy notions have been proposed, however, most of them stand broken now. For example, de-anonymization of the Netflix challenge dataset by Narayanan and Shmatikov (2008). In a seminal work, Dwork et al. (2006b) proposed *differential privacy*, a cryptography inspired privacy notion with solid theoretical foundation. This notion is now accepted as the standard notion of privacy, and in this work we adhere to it for our privacy guarantees.

Recently, several differentially private algorithms have been developed for learning problems (Blum et al., 2008; Chaudhuri et al., 2011; Williams and McSherry, 2010; Manas Pathak and Raj, 2010; Rubinstein et al., 2009). Among these, the works by Chaudhuri et al. (2011); Rubinstein et al. (2009); Williams and McSherry (2010) are the most related as they consider a large class of offline learning problems that can be written as regularized empirical risk minimization (ERM) problems with convex loss functions. In particular, Chaudhuri et al. (2011); Rubinstein et al. (2009) proposed a differentially private method that ensures privacy by either adding noise to the optima of the corresponding ERM or by perturbing the ERM itself. In both these cases, the privacy guarantees depend on the promise that the *exact* minimum to the underlying optimization problem is obtained, which is unlikely for several practical problems. In contrast, Williams and McSherry (2010) proposed a *noisy* gradient descent method to optimize ERM. Although their method maintains differential privacy at each gradient descent step, it fails to provide any convergence guarantees. Interestingly, our online learning techniques can be applied to this offline learning problem as well. In fact, our method provides better error bounds and is more practical than the existing methods (see Section 5).

As mentioned earlier, most of the existing work in differentially private learning has been in the offline setting. One notable exception is the work of Dwork et al. (2010a), that considers the problem of preserving privacy in the experts setting. In particular, they provide a differentially private algorithm for experts framework that has $\tilde{O}(\sqrt{T})$ regret. However, their results are restricted to the experts setting only, and it is not clear how their techniques can be generalized to the general class of OCP problems.

In a related line of work, there have been a few results that use online learning techniques to obtain differentially private algorithms (Hardt and Rothblum, 2010; Dwork et al., 2010b; Gupta et al., 2011). In particular, Hardt and Rothblum (2010); Gupta et al. (2011) used the experts framework to obtain a differentially private algorithm for answering *adaptive* counting queries on a dataset. We stress that although these methods use online learning techniques, they are designed to handle offline problems only where the dataset is fixed and is known in advance.

2. Preliminaries

2.1. Online Convex Programming

Online convex programming (OCP) is one of the most popular and powerful paradigms in the online learning setting. OCP can be thought of as a game between a player and an adversary. At each step t, the player selects a point $\boldsymbol{x}_t \in \mathbb{R}^d$ from a convex set $\mathcal{C}$. Then, the adversary selects a convex cost function $f_t : \mathbb{R}^d \to \mathbb{R}$ and the player has to pay a cost of $f_t(\boldsymbol{x}_t)$. An OCP algorithm $\mathcal{A}$ maps a function sequence $F = \langle f_1, f_2, \ldots, f_T \rangle$ to a sequence of points $X = \langle \boldsymbol{x}_2, \boldsymbol{x}_3, \ldots, \boldsymbol{x}_{T+1} \rangle \in \mathcal{C}^T$, i.e., $\mathcal{A}(F) = X$. Now, the goal of the player (or the algorithm) is to minimize *regret*, i.e., the total cost incurred when compared to the optimal offline solution $\boldsymbol{x}^*$ selected in hindsight, i.e., when all the functions have already been provided. Formally,

Definition 1 (Regret) *Let $\mathcal{A}$ be an online convex programming algorithm that selects a point $\boldsymbol{x}_t \in \mathcal{C}$ at the $t-1$-th iteration. Let $f_t : \mathbb{R}^d \to \mathbb{R}$ be a convex cost function served at the t-th iteration. Then, the regret $\mathcal{R}_\mathcal{A}$ of $\mathcal{A}$ over T iterations is given by: $\mathcal{R}_\mathcal{A}(T) = \sum_{t=1}^T f_t(\boldsymbol{x}_t) - \min_{\boldsymbol{x}^* \in \mathcal{C}} \sum_{t=1}^T f_t(\boldsymbol{x}^*)$.*

Several OCP algorithms exist in the literature that guarantee $O(\sqrt{T})$ regret for bounded Lipschitz functions f_t and $O(\ln T)$ regret for *strongly* convex functions f_t (Kulis and Bartlett, 2010; Zinkevich, 2003; Kakade and Shalev-Shwartz, 2008).

2.2. Differential Privacy

We now formally define the notion of differential privacy in the context of our problem.

Definition 2 ((ϵ, δ)-differential privacy Dwork et al. (2006b,a)) *Let $F = \langle f_1, f_2, \ldots, f_T \rangle$ be a sequence of convex functions. Let $\mathcal{A}(F) = X$, where $X = \langle \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T \rangle \in \mathcal{C}^T$ be T outputs of the OCP algorithm $\mathcal{A}$ when applied to F. A randomized OCP algorithm $\mathcal{A}$ is (ϵ, δ)-differentially private if for any two function sequences F and F' that differ in at most one function entry, and for all $\mathcal{S} \subset \mathcal{C}^T$ the following holds:*

$$\Pr[\mathcal{A}(F) \in \mathcal{S}] \leq e^\epsilon \Pr[\mathcal{A}(F') \in \mathcal{S}] + \delta$$

Intuitively, the above definition means that changing any $f_t \in F$, $t \leq T$ to some other function f'_t will not modify the output sequence X by a large amount. If we consider each f_t to be some information or data point associated with an individual, then the definition above states that the presence or absence of that individual's entry in the dataset will not affect the output by too much. Hence, the output of algorithm $\mathcal{A}$ will not reveal any extra information about the individual. Privacy parameters (ϵ, δ) decide the extent to which an individual's entry affects the output; lower values of ϵ and δ imply higher level of privacy.

2.3. Notation

$F = \langle f_1, f_2, \ldots, f_T \rangle$ denotes the function sequence given to an OCP algorithm $\mathcal{A}$ and $\mathcal{A}(F) = X$ s.t. $X = \langle \boldsymbol{x}_2, \boldsymbol{x}_3, \ldots, \boldsymbol{x}_{T+1} \rangle \in \mathcal{C}^T$ represents output sequence when $\mathcal{A}$ is applied to F. We denote the subsequence of functions F till the t-th step as $F_t = \langle f_1, \ldots, f_t \rangle$. $\mathcal{C} \subseteq \mathcal{R}^d$, denotes a convex set in d dimensions. Vectors are denoted by bold-face symbols (e.g., $\boldsymbol{x}$), matrices are represented by capital letters (e.g., M). $\boldsymbol{x}^T \boldsymbol{y}$ denotes the inner product between $\boldsymbol{x}$ and $\boldsymbol{y}$. $\|M\|_2$ denotes the spectral norm of matrix M and is the largest eigenvalue of M.

Typically, α is the minimum strong convexity parameter of any $f_t \in F$. Similarly, L is the largest Lipschitz constant of any $f_t \in F$ and L_G is the largest Lipschitz constant of the gradient of any $f_t \in F$. Recall that a function $f : \mathcal{C} \to \mathbb{R}$ is α-strongly convex, if

$\forall\ \boldsymbol{x}, \boldsymbol{y} \in \mathcal{C}$ the following holds: $f(\gamma\boldsymbol{x} + (1-\gamma)\boldsymbol{y}) \leq \gamma f(\boldsymbol{x}) + (1-\gamma)f(\boldsymbol{y}) - \frac{\alpha\gamma(1-\gamma)}{2}||\boldsymbol{x} - \boldsymbol{y}||_2^2, 0 \leq \gamma \leq 1$. Also recall that a function f is L-Lipschitz, if $\forall\ \boldsymbol{x}, \boldsymbol{y} \in \mathcal{C}$ the following holds: $|f(\boldsymbol{x}) - f(\boldsymbol{y})| \leq L||\boldsymbol{x} - \boldsymbol{y}||_2$. Function f is L_G-Lipschitz continuous gradient if $|| \bigtriangledown f(\boldsymbol{x}) - \bigtriangledown f(\boldsymbol{y})||_2 \leq L_G||\boldsymbol{x} - \boldsymbol{y}||_2, \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{C}$.

At time-step t, non-private and private versions of any OCP algorithm output $\boldsymbol{x}_{t+1}$ and $\hat{\boldsymbol{x}}_{t+1}$, respectively. $\boldsymbol{x}^*$ denotes the optimal offline solution, that is $\boldsymbol{x}^* = \text{argmin}_{\boldsymbol{x}\in\mathcal{C}} \sum_{t=1}^{T} f_t(\boldsymbol{x})$. $\mathcal{R}_{\mathcal{A}}(T)$ denotes the regret of an OCP algorithm $\mathcal{A}$ when applied for T steps.

3. Differentially Private Online Convex Programming

In this section we first present our differentially private framework for solving OCP problems, and provide privacy as well as regret guarantees for our framework. Then, in Appendix B, we instantiate our framework with the *Implicit Gradient Descent* (IGD) (Kulis and Bartlett, 2010) algorithm and provide regret, privacy guarantees for the same. We also instantiate our framework with the *Generalized Infinitesimal Gradient Ascent* (GIGA) (Zinkevich, 2003) algorithm (see Appendix C).

Recall that a differentially private OCP algorithm should not produce *significantly* different sequences of outputs ($X = \langle \boldsymbol{x}_2, \cdots, \boldsymbol{x}_{T+1}\rangle$) for input function sequences F and F', where F and F' differ in exactly one cost function. A trivial way to ensure it is by selecting output sequence X independently of the input cost functions F. However, such an "algorithm" can have $O(T)$ regret.

To discard such bad solutions, we require a differentially private OCP algorithm to have both: a) Privacy: (ϵ, δ)-differential privacy, and b) Utility: sub-linear regret.

Our generic framework can transform any given OCP algorithm, $\mathcal{A}$, into a differentially private OCP algorithm that satisfies the above given requirements. However, we require $\mathcal{A}$ to have sub-linear regret and low sensitivity, i.e., $\mathcal{A}$ should not be very sensitive to any particular cost function in the input sequence. We now formalize this notion of sensitivity:

Definition 3 (L_2-sensitivity (Dwork et al., 2006b; Chaudhuri et al., 2011)) *Let F, F' be two function sequences differing in at most one entry, i.e., at most one function is different in the two sequences. Let $\boldsymbol{x}_{t+1} = \mathcal{A}(F)_t$ be the t-th output of $\mathcal{A}$ when supplied F, and similarly, $\boldsymbol{x}_{t+1} = \mathcal{A}(F')_t$ is the t-th output of $\mathcal{A}$ for input sequence F'. Then sensitivity of the algorithm $\mathcal{A} : F \to \mathcal{C}^T$, at the t-th time-step is given by: $\mathcal{S}(\mathcal{A}, t) = \sup_{F,F'} ||\mathcal{A}(F)_t - \mathcal{A}(F')_t||_2$.*

Using this definition of sensitivity, we now state the assumptions that $\mathcal{A}$ should satisfy:

- L_2-sensitivity: The L_2-sensitivity of the algorithm $\mathcal{A}$ should decay linearly with time, i.e.,
$$\mathcal{S}(\mathcal{A}, t) \leq \frac{\lambda_{\mathcal{A}}}{t}, \tag{1}$$
where $\lambda_{\mathcal{A}} > 0$ is a constant depending only on $\mathcal{A}$, L and α, i.e., the Lipschitz constant and the strong convexity parameter of the functions in F.
- Regret bound $\mathcal{R}_{\mathcal{A}}(T)$: Regret of $\mathcal{A}$ is assumed to be sub-linear in T, i.e.,
$$\mathcal{R}_{\mathcal{A}}(T) = \sum_{t=1}^{T} f_t(\boldsymbol{x}_t) - \min_{\boldsymbol{x}^*\in\mathcal{C}} \sum_{t=1}^{T} f_t(\boldsymbol{x}^*) = o(T). \tag{2}$$

Algorithm 1 Private OCP Method (POCP)

```
Input: OCP algorithm $\mathcal{A}$, cost function sequence $F = \langle f_1, \cdots, f_T \rangle$ and the convex set $\mathcal{C}$
Parameter: privacy parameters $(\epsilon, \delta)$
Choose $\boldsymbol{x}_1$ and $\hat{\boldsymbol{x}}_1$ randomly from $\mathcal{C}$
for $t = 1$ to $T - 1$ do
    Cost: $L_t(\hat{\boldsymbol{x}}_t) = f_t(\hat{\boldsymbol{x}}_t)$
    OCP Update: $\boldsymbol{x}_{t+1} \leftarrow \mathcal{A}(\langle f_1, \ldots, f_t \rangle, \langle \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t \rangle, \mathcal{C})$
    Noise Addition: $\tilde{\boldsymbol{x}}_{t+1} \leftarrow \boldsymbol{x}_{t+1} + \boldsymbol{b}_{t+1}$, $\boldsymbol{b}_{t+1} \sim \mathcal{N}(0^d, \frac{\beta^2}{t^2}\mathbb{I}^d)$,
        where $\beta = \lambda_{\mathcal{A}} T^{0.5+c} \sqrt{\frac{2}{\epsilon}\left(\ln\frac{T}{\delta} + \frac{\sqrt{\epsilon}}{T^{0.5+c}}\right)}$ and $c = \frac{\ln\frac{1}{2\epsilon}\ln(2/\delta)}{2\ln T}$
    Output $\hat{\boldsymbol{x}}_{t+1} = \mathrm{argmin}_{\boldsymbol{x} \in \mathcal{C}} \left( \|\boldsymbol{x} - \tilde{\boldsymbol{x}}_{t+1}\|_2^2 \right)$
end for
```

A natural question to ask is whether there exists an OCP algorithm $\mathcal{A}$ that satisfies both the conditions above. In Appendix B and Appendix C, we show that both IGD and GIGA satisfy these conditions. In fact, recent results by Ross and Bagnell (2011); Poggio et al. (2011) seem to indicate a close connection between sensitivity and regret for online learning algorithms . We leave further investigation of the interplay between sensitivity and regret as a topic for future research.

Now, given $\mathcal{A}$ that satisfies both (1) and (2), we transform it into a private algorithm by perturbing $\boldsymbol{x}_{t+1}$ (output of $\mathcal{A}$ at t-th step) by a small amount of noise, whose magnitude is fixed by the *noise parameter* β in Algorithm 1. Let $\tilde{\boldsymbol{x}}_{t+1}$ be the perturbed output which might lie outside the convex set $\mathcal{C}$. As OCP requires each output to lie in $\mathcal{C}$, we project $\tilde{\boldsymbol{x}}_{t+1}$ back to $\mathcal{C}$ and output the projection $\hat{\boldsymbol{x}}_{t+1}$. Note that our Private OCP (POCP) algorithm also stores the "uncorrupted" iterate $\boldsymbol{x}_{t+1}$, which is used in the next step. See Algorithm 1 for a pseudo-code of our method.

Now, using the above two assumptions along with concentration bounds for Gaussian noise, we obtain privacy and regret guarantees for our Private OCP algorithm. Using the sensitivity assumption, it is easy to prove differential privacy for the output of any *fixed* time step. However, we need to provide differential privacy jointly over *all* time steps, which is the main technical novelty of our work. See Sections 3.1 and 3.2 for a detailed analysis of privacy and regret guarantees, respectively.

3.1. Privacy Analysis for POCP

Under the assumption (1), changing one function in the cost function sequence F can lead to a change of at most $\lambda_{\mathcal{A}}/t$ in the t-th output of $\mathcal{A}$. Intuitively, adding a noise of the same order should make the output of Algorithm 1 at the t-th step "almost independent" of any fixed cost function and hence, differentially private. We make this idea precise in the following lemma.

Lemma 1 *Let $\mathcal{A}$ be an OCP algorithm that satisfies the sensitivity assumption (1) and let $\lambda_{\mathcal{A}}$ be the sensitivity parameter. Fix the noise parameter in Algorithm 1 as*

$\beta = \lambda_{\mathcal{A}} T^{0.5+c}\sqrt{\frac{2}{\epsilon}\left(\ln\frac{T}{\delta} + \frac{\sqrt{\epsilon}}{T^{0.5+c}}\right)}$ for $c > 0$. Then, the output at the t-th step of the Algorithm 1, $\hat{\boldsymbol{x}}_{t+1}$, is $(\frac{\sqrt{\epsilon}}{T^{0.5+c}}, \frac{\delta}{T})$-differentially private.

See appendix A.1 for a proof of the above lemma.

The above lemma shows that the output at each step of Algorithm 1 is $(\frac{\sqrt{\epsilon}}{T^{0.5+c}}, \frac{\delta}{T})$-differentially private. Hence, a simple composition argument guarantees $(T^{0.5-c}\sqrt{\epsilon}, \delta)$-differential privacy for all the steps (Dwork and Lei, 2009). Thus, setting $c = 0.5$ makes the outputs at every time step (ϵ, δ)-differentially private. However, this requires that a noise of variance $\sim O(T/t)$ be added at each step. This means that the noise added to any fixed output is much larger than the effect of incoming function f_t and thus can lead to an arbitrarily high regret.

To avoid this problem and obtain better regret bounds, we exploit the interdependence between the iterates (and outputs) of our algorithm. For this purpose, we use a lemma by Dwork et al. (2010b, Lemma III.2) that bounds the relative entropy of two random variables in terms of the L_∞ norm of the ratio of their probability densities and also a proof technique developed by Hardt and Rothblum (2010); Hardt et al. (2010) for the problem of releasing differentially private datasets.

Now we state the privacy guarantee for Algorithm 1 over *all* T iterations.

Theorem 1 (POCP Privacy) *Let $\mathcal{A}$ be an OCP algorithm that satisfies the sensitivity assumption (1) with sensitivity parameter $\lambda_{\mathcal{A}}$. Then, the POCP algorithm (Algorithm 1) with the noise parameter $\beta = \lambda_{\mathcal{A}} T^{0.5+c}\sqrt{\frac{2}{\epsilon}\left(\ln\frac{T}{\delta} + \frac{\sqrt{\epsilon}}{T^{0.5+c}}\right)}$ is $(3\epsilon, 2\delta)$-differentially private for $c = \frac{\left(\ln(\frac{1}{2\epsilon}\ln\frac{2}{\delta}\right)}{2\ln T}$.*

The proof of this theorem is included in Appendix A.2.

3.2. Utility (Regret) Analysis for POCP

In this section, we provide a generic regret bound analysis for our POCP algorithm (see Algorithm 1). The regret bound of POCP depends on the regret $\mathcal{R}_{\mathcal{A}}(T)$ of the non-private OCP algorithm $\mathcal{A}$. For typical OCP algorithms like IGD, GIGA , $\mathcal{R}_{\mathcal{A}}(T) = O(\log T)$, assuming each cost function f_t is strongly convex.

Theorem 2 (POCP Regret) *Let $\mathcal{A}$ be an OCP algorithm that satisfies sensitivity assumption (1), and let $\lambda_{\mathcal{A}}$ be the corresponding sensitivity parameter. Also, let $\mathcal{R}_{\mathcal{A}}(T)$ be the regret of $\mathcal{A}$ over T-time steps, and $L > 0$ be the maximum Lipschitz constant of any function f_t. Then, the expected regret of our POCP algorithm (Algorithm 1) satisfies:*

$$\mathbb{E}\left[\sum_{t=1}^{T} f_t(\hat{\boldsymbol{x}}_t)\right] - \min_{\boldsymbol{x}\in\mathcal{C}}\sum_{t=1}^{T} f_t(\boldsymbol{x}) \leq 2\sqrt{d}L(\lambda_{\mathcal{A}} + \|\mathcal{C}\|_2)\sqrt{T}\frac{\ln^2\frac{T}{\delta}}{\epsilon} + \mathcal{R}_{\mathcal{A}}(T),$$

where $\boldsymbol{x} \in \mathbb{R}^d$ and $\|\mathcal{C}\|_2$ is the diameter of the convex set $\mathcal{C}$.

The above theorem shows that in expectation, POCP($\mathcal{A}$) algorithm has an additional regret of $\tilde{O}(\sqrt{dT})$ compared to the regret of $\mathcal{A}$. We present a detailed proof of this theorem in

Section A.3. The main idea is to bound the total effect of noise on the regret via bounding the effect of noise on the output of individual iterations. Using Chebyshev's inequality, we can also obtain a high probability bound on the regret (see Corollary 3 in Section A.4).

4. Logarithmic regret for Quadratic Cost Functions

In Appendix B, we describe a differentially private algorithm (PIGD) with $\tilde{O}(\sqrt{T})$ regret for *any* strongly convex Lipschitz continuous cost functions. In this section we show that by restricting the input cost functions to be quadratic functions only, we can design a differentially private algorithm that incurs only logarithmic regret.

For simplicity of exposition, we consider cost functions of the form:

$$f_t(\boldsymbol{x}) = \frac{1}{2}(y_t - \boldsymbol{v}_t^T \boldsymbol{x})^2 + \frac{\alpha}{2}\|\boldsymbol{x}\|^2, \tag{3}$$

where $\alpha > 0$. For such cost functions we show that there is an algorithm that incurs just $O(\text{poly}(\log T))$ regret while providing (ϵ, δ)-differential privacy.

Our Private Quadratic Follow the Leader (PQFTL) algorithm at a high level is a noisy version of the Follow the Leader (FTL) algorithm. Now, for the specific case of quadratic cost function (3) with $\mathcal{C} = \mathbb{R}^d$, FTL updates can be re-written as:

$$\text{QFTL}: \qquad \boldsymbol{x}_{t+1} = (t\alpha\mathbb{I} + V_t)^{-1}(\boldsymbol{u}_t), \tag{4}$$

where $V_t = V_{t-1} + \boldsymbol{v}_t\boldsymbol{v}_t^T$ and $\boldsymbol{u}_t = \boldsymbol{u}_{t-1} + y_t\boldsymbol{v}_t$ with $V_0 = 0$ and $\boldsymbol{u}_0 = 0$.

Using elementary linear algebra and assuming $|y_t|, \|\boldsymbol{v}_t\|_2 \leq R$, we can show that $\|\boldsymbol{x}_{t+1}\|_2 \leq 2R/\alpha, \ \forall t \leq T$. Now, using Theorem 2 of Kakade and Shalev-Shwartz (2008) along with our bound on $\|\boldsymbol{x}_{t+1}\|_2$, we obtain the following regret bound for the QFTL algorithm:

$$\mathcal{R}_{\text{QFTL}}(T) \leq \frac{R^4(1 + 2R/\alpha)^2}{\alpha} \log T. \tag{5}$$

Furthermore, we can show that the QFTL algorithm (see Equation 4) also satisfies assumption 1. Hence, similar to Appendix B, we can obtain a differentially private variant of QFTL with $\tilde{O}(\sqrt{T})$ regret. However, we show that using the special structure of QFTL updates, we can improve the regret to $O(\text{poly}(\log T))$.

The key observation is that each QFTL update is dependent on the function sequence F through V_t and $\boldsymbol{u}_t$ only. Hence, computing V_t and $\boldsymbol{u}_t$ in a differentially private manner would imply differential privacy for our QFTL updates as well. Furthermore, each V_t and $\boldsymbol{u}_t$ themselves are just sums of individual "atoms" ($\boldsymbol{v}_\tau\boldsymbol{v}_\tau^T$ for V_t and $y_\tau\boldsymbol{v}_\tau$ for $\boldsymbol{u}_t$). This special structure of V_t and $\boldsymbol{u}_t$ facilitates usage of a generalization of the "tree-based" technique for computing privacy preserving partial sums proposed by Dwork et al. (2010a). Note that the "tree-based" technique to compute sums (Algorithm 5 in Appendix D.2) adds significantly lower amount of noise at each step than what is added by our POCP algorithm (see Algorithm 1). Hence, it leads to a significantly better regret.

Algorithm 4 (in Appendix D.1) provides a pseudo-code of our PQFTL method. At each step t, $\hat{V}_t$ and $\hat{\boldsymbol{u}}_t$ are computed by perturbing V_t and $\boldsymbol{u}_t$ (to preserve privacy) using PrivateSum algorithm (see Algorithm 5 in Appendix D.2). Next, $\hat{V}_t$ and $\hat{\boldsymbol{u}}_t$ are used in the QFTL update (4) to obtain the next iterate $\hat{x}_{t+1}$.

Theorem 3 (PQFTL **Privacy**) *Let F be a sequence of quadratic functions, where $f_t(\boldsymbol{x}; y_t, \boldsymbol{v}_t) = \frac{1}{2}(y_t - \boldsymbol{v}_t^T\boldsymbol{x})^2 + \frac{\alpha}{2}||\boldsymbol{x}||_2^2$. Then,* PQFTL *(Algorithm 4) is (ϵ, δ)-differentially private.*

In Appendix D.2, we show how one can compute partial sums privately by only adding a noise of variance $poly - log(T)$. The proof then follows by observing that the computation of the output at time step t can be done by computing appropriate partial sums privately. The complete proof is in Appendix D.1 of the appendix.

Theorem 4 (PQFTL **Regret**) *Let F be a sequence of quadratic functions, where $f_t(\boldsymbol{x}; y_t, \boldsymbol{v}_t) = \frac{1}{2}(y_t - \boldsymbol{v}_t^T\boldsymbol{x})^2 + \frac{\alpha}{2}||\boldsymbol{x}||_2^2$. Let R be the maximum L_2 norm of any $\boldsymbol{v}_t$ and $|y_t|$. Then, w.p. at least $\geq 1 - \exp(-d/2)$, the regret bound of* PQFTL *(Algorithm 4) satisfies : $\mathcal{R}_{\mathsf{PQFTL}}(T) = \tilde{O}\left(\frac{\mathcal{R}^6 \log\frac{1}{\delta}}{\epsilon\alpha^3}\sqrt{d}\log^{1.5} T\right)$.*

The regret follows by using regret bound of QFTL algorithm and by accounting for noise at each step t. See Appendix D.1 for a detailed proof.

5. Application to Offline Learning

In Section 3, we proposed a generic framework for differentially private OCP algorithms with sub-linear regret bounds. Recently, Kakade and Tewari (2008) showed that OCP algorithms with sub-linear regret bounds can be used to solve several offline learning problems as well. In this section, we exploit this connection to provide a generic differentially private framework for a large class of offline learning problems as well.

In related works, Chaudhuri et al. (2011); Rubinstein et al. (2009) also proposed methods to obtain differentially private algorithms for offline learning problems. However, as discussed later in the section, our method is more practical and obtains better error bounds for the same level of privacy. It also covers a wider range of problems than Chaudhuri et al. (2011).

First, we describe the standard offline learning model that we use. Consider a domain $\mathcal{Z}$ and an arbitrary distribution $\mathcal{D}_{\mathcal{Z}}$ over $\mathcal{Z}$ from which the training data is generated. Let $D = \langle \boldsymbol{z}_1, \cdots, \boldsymbol{z}_T\rangle$ be the training dataset, where each $\boldsymbol{z}_i$ is drawn i.i.d. from the distribution $\mathcal{D}_{\mathcal{Z}}$. Typically, $\boldsymbol{z}_i$ is a tuple of a training point and its label. Also, consider a loss function $\ell : \mathcal{C} \times \mathcal{Z} \rightarrow \mathbb{R}^+$, where $\mathcal{C} \subseteq \mathbb{R}^d$ is a (potentially unbounded) convex set. Let $\ell(\cdot;\cdot)$ be a L-Lipschitz (in the first parameter) convex function. Intuitively, the loss function quantifies the goodness of a learned model $\boldsymbol{x} \in \mathcal{C}$ w.r.t. the training data. Now, the goal is to solve the following *Risk Minimization* problem:

$$\min_{\boldsymbol{x}\in\mathcal{C}} \mathbb{E}_{\boldsymbol{z}\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\boldsymbol{x}; \boldsymbol{z})]. \tag{6}$$

Let $\boldsymbol{x}^*$ be the optimal solution to (6), i.e., $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}\in\mathcal{C}} \mathbb{E}_{\boldsymbol{z}\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\boldsymbol{x}; \boldsymbol{z})]$. Recently, Kakade and Tewari (2008) provided a stochastic offline learning algorithm to obtain an additive approximation to (6) via OCP. The algorithm of Kakade and Tewari (2008) is as follows: execute any reasonable OCP algorithm $\mathcal{A}$ (like IGD or GIGA) on the function sequence F, where $f_t = \ell(\boldsymbol{x}; \boldsymbol{z}_t) + \frac{\alpha}{2}\|\boldsymbol{x}\|^2$. Note that each $\boldsymbol{z}_t$ is sampled i.i.d. from $\mathcal{D}_{\mathcal{Z}}$. Also, if the convex set $\mathcal{C}$ required in OCP is an unbounded set, then it can be set to be an L_2 ball of radius $\|\boldsymbol{x}^*\|_2$, i.e, $\mathcal{C} = \{\boldsymbol{x} : \boldsymbol{x} \in \mathbb{R}^d, \|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}^*\|_2\}$. In practice, $\|\boldsymbol{x}^*\|$ can be estimated using cross validation, which is analogous to tuning the regularization parameter in standard learning problems like SVM.

Let $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T$ be the sequence of outputs produced by $\mathcal{A}$. Then, the output of the stochastic offline learning algorithm is given by, $\tilde{\boldsymbol{x}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{x}_t$. Kakade and Tewari (2008) show that $\tilde{\boldsymbol{x}}$ is a reasonable approximation to $\boldsymbol{x}^*$ with provable approximation error (see Theorem 12).

To produce differentially private output, we add noise of an appropriate variance to the output $\tilde{\boldsymbol{x}}$ and project it back to $\mathcal{C}$. That is,

$$\mathsf{POL}: \quad \hat{\boldsymbol{x}} = \mathrm{argmin}_{\boldsymbol{x}\in\mathcal{C}} \|\boldsymbol{x} - \tilde{\boldsymbol{x}} - \boldsymbol{b}\|_2^2, \quad \boldsymbol{b} \sim \mathcal{N}(0, \beta^2\mathbb{I}^d), \quad \beta = \frac{2\sqrt{2}(L+\alpha\|\boldsymbol{x}^*\|_2)\ln T}{T\epsilon_p}\sqrt{\ln\frac{1}{\delta} + \epsilon_p}.$$

We refer to this algorithm as Private Offline Learning (POL) and provide a detailed pseudocode in Algorithm 6 (Appendix E). Next, we show that POL (Algorithm 6) is differentially private.

Theorem 5 (POL Privacy) *Private Offline Learning (POL) algorithm (see Algorithm 6) is (ϵ_p, δ)-differentially private.*

See appendix E.1 for a detailed proof. At a high level, the proof follows from the L_2-sensitivity analysis of the IGD algorithm.

Next, we provide a utility guarantee for POL, i.e., a bound on the approximation error for the Risk Minimization problem (6). See appendix E.2 for a detailed proof. The main tool in the proof is a bound on the approximation error for the stochastic offline learning by Kakade and Tewari (2008).

Theorem 6 (POL Utility (Approximation Error in Eq. 6)) *Let L be the Lipschitz bound on the loss function ℓ and T be the total number of points in the training dataset $D = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_T\}$. Let (ϵ_p, δ) be the differential privacy parameters, d the dimensionality, $C > 0$ a global constant. Then, with probability at least $1-\gamma$,*

$$\mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\hat{\boldsymbol{x}}; z)] - \min_{\boldsymbol{x}\in\mathcal{C}} \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\boldsymbol{x}; z)] \le \epsilon_g,$$

when the number of points sampled (T) satisfies,

$$T \ge C \max\left(\frac{\sqrt{d}L(L+\epsilon_g/\|\boldsymbol{x}^*\|_2)\sqrt{\ln\frac{1}{\gamma}\ln\frac{1}{\delta}}}{\epsilon_g\epsilon_p}, \frac{(L+\epsilon_g/\|\boldsymbol{x}^*\|_2)^2\|\boldsymbol{x}^*\|_2^2\ln T\ln\frac{\ln T}{\gamma}}{\epsilon_g^2}\right).$$

Comparison to existing differentially private offline learning methods: We now compare our POL algorithm for private (offline) Risk Minimization with the existing methods (Chaudhuri et al., 2011; Rubinstein et al., 2009):

- **Better error bound**: Our Theorem 6 improves the sample complexity bounds of Chaudhuri et al. (2011); Rubinstein et al. (2009) by a factor of $\sqrt{d}$. We believe the difference is primarily due to our use of Gaussian noise instead of Gamma noise added by the existing methods.
- **More practical**: Both Chaudhuri et al. (2011); Rubinstein et al. (2009) need to compute the *exact* optimal solution to the optimization problem that they consider and it not clear if their privacy guarantees hold if one can obtain only an approximate solution to their respective optimization problems. In contrast, our method uses an explicit iterative method for solving (6) and provides privacy and utility guarantees even if the algorithm stops early.

Remark: Note that (Chaudhuri et al., 2011) does not allow the loss function ℓ to be non-differentiable and the convex set $\mathcal{C}$ to be bounded. In comparison both (Rubinstein et al., 2009) and our POL method support non-differentiable loss functions and bounded convex sets. Also, note that our $\sqrt{d}$ sample complexity bound does not contradict the corresponding $\Omega(d)$ lower bound proved by Chaudhuri and Hsu (2011). Reason being, we use (ϵ, δ)-differential privacy notion which is a less strict notion of privacy than the ϵ-differential privacy notion used by Chaudhuri and Hsu (2011).

6. Discussion

6.1. Other Differentially Private Algorithms

Recall that in Appendix B, we described our Private IGD algorithm that achieves $\tilde{O}(\sqrt{T})$ regret for any sequence of strongly convex, Lipschitz continuous functions. While, this class of functions is reasonably broad, we can further drop the strong convexity condition as well, albeit with higher regret. To this end, we perturb each f_t and apply IGD over $\tilde{f}_t = f_t + \frac{\alpha}{\sqrt{t}}||\boldsymbol{x} - \boldsymbol{x}_0||_2^2$, where $\boldsymbol{x}_0$ is randomly picked point from the convex set $\mathcal{C}$. We can then show that using this perturbation "trick", we can obtain a regret of $\tilde{O}(T^{2/3})$.

We now briefly discuss another OCP algorithm, namely, Exponentially Weighted Online Optimization algorithm (Hazan et al., 2007). This algorithm does not directly fit into our POCP framework, and is not wide-spread in practice due to relatively inefficient updates (see (Hazan et al., 2007) for more discussion). However, for completeness, we note that by using techniques similar to our POCP framework and using *exponential mechanism* (see McSherry and Talwar (2007)), one can analyze this algorithm as well to guarantee differential privacy along with $\tilde{O}(\sqrt{T})$ regret.

6.2. POCP algorithm when the number of iterations (T) is not known

Algorithm 1 requires the number of iterations T to be known apriori, as the amount of noise to be added at each step depends on T. However, in many practical situations T might not be known in advance. We address this problem by using the standard *doubling trick*.

At a high level, the idea is the following: rather than adding enough noise to provide $(\frac{\sqrt{\epsilon}}{T^{0.5}}, \frac{\delta}{T})$-differentially private for the output at each time step (as is done in Algorithm 1), we provide iteration dependent guarantees. That is, at the t-th iteration, we add enough noise to guarantee $(\sqrt{\frac{\epsilon}{2^{\lfloor \log_2 t \rfloor}}}, \frac{\delta}{2^{\lfloor \log_2 t \rfloor}})$-differential privacy for the output at time step t. Hence, we make the output in first iteration $(\sqrt{\epsilon}, \delta)$-private, and of the second and the third iteration $(\sqrt{\frac{\epsilon}{2}}, \frac{\delta}{2})$-private, and so on.

Repeating the analysis of Algorithm 1 with this modification, we obtain $(3\epsilon \log T, \delta \log_2 T + 2/T^{2\epsilon})$-differential privacy while incurring the *same* regret as Theorem 2 (see Appendix F). Now, to get differential privacy guarantee with constant ϵ', δ', we need to set $\epsilon = \epsilon'/\log T$ and $\delta = \delta'/\log T$, which is still dependent on T. However, the dependence on T is significantly weaker now, and an estimate of T within a factor of $poly(T)$ will weaken the privacy by a constant only.

References

Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to non-interactive database privacy. In *STOC*, 2008.

Kamalika Chaudhuri and Daniel Hsu. Sample complexity bounds for differentially private learning. *JMLR*, 19, 2011.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D. Sarwate. Differentially private empirical risk minimization. *JMLR*, 2011.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *JMLR*, 2006.

Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *STOC*, 2009.

Cynthia Dwork, Krishnaram Kenthapadi, Frank Mcsherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, 2006a.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006b.

Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *STOC*, 2010a.

Cynthia Dwork, Guy N. Rothblum, and Salil P. Vadhan. Boosting and differential privacy. In *FOCS*, 2010b.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL http://archive.ics.uci.edu/ml.

Anupam Gupta, Aaron Roth, and Jonathan Ullman. Iterative constructions and private data release. *CoRR*, abs/1107.3731, 2011.

Moritz Hardt and Guy N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *FOCS*, 2010.

Moritz Hardt, Katrina Ligett, and Frank McSherry. A simple and practical algorithm for differentially private data release. *CoRR*, abs/1012.4763, 2010.

Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69, 2007.

Sham Kakade and Shai Shalev-Shwartz. Mind the duality gap: Logarithmic regret algorithms for online optimization. In *NIPS*, 2008.

Sham M. Kakade and Ambuj Tewari. On the generalization ability of online strongly convex programming algorithms. In *NIPS*, 2008.

Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer System and Science*, 71, 2005.

Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132, 1995.

Brian Kulis and Peter L. Bartlett. Implicit online learning. In *ICML*, 2010.

Shantanu Rane Manas Pathak and Bhiksha Raj. Multiparty differential privacy via aggregation of locally trained classifiers. In *NIPS*, 2010.

Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS*, 2007.

Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *IEEE Symposium on Security and Privacy (ISSP)*, 2008.

Tomaso Poggio, Stephen Voinea, and Lorenzo Rosasco. Online learning, stability, and stochastic gradient descent. *CoRR*, abs/1105.4701, 2011.

Stéphane Ross and J. Andrew Bagnell. Stability conditions for online learnability. *CoRR*, abs/1108.3154, 2011.

Benjamin I. P. Rubinstein, Peter L. Bartlett, Ling Huang, and Nina Taft. Learning in a large function space: Privacy-preserving mechanisms for SVM learning. *CoRR*, abs/0911.5708, 2009.

Oliver Williams and Frank McSherry. Probabilistic inference and differential privacy. In *NIPS*, 2010.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, 2003.

Appendix A. Differentially Private Online Convex Programming

A.1. Proof of Lemma 1

Proof First, note that $\hat{\boldsymbol{x}}_{t+1}$ is obtained by projecting $\tilde{\boldsymbol{x}}_{t+1}$ on $\mathcal{C}$. Thus if $\tilde{\boldsymbol{x}}_{t+1}$ is (ϵ', δ')-differentially private then $\hat{\boldsymbol{x}}_{t+1}$ is also (ϵ', δ')-differentially private . Therefore, we prove the lemma for $\tilde{\boldsymbol{x}}_{t+1}$.

Let F and F' be two input function sequences that differ in exactly one entry. Suppose $\boldsymbol{x}_{t+1}$ and $\boldsymbol{x}'_{t+1}$ are the uncorrupted outputs of the OCP algorithm $\mathcal{A}$ (before adding noise) on input sequences F and F', respectively. Similarly, let $\tilde{\boldsymbol{x}}_{t+1} = \boldsymbol{x}_{t+1} + \boldsymbol{b}_{t+1}$ and $\tilde{\boldsymbol{x}}'_{t+1} = \boldsymbol{x}'_{t+1} + \boldsymbol{b}_{t+1}$ be the perturbed t-th step outputs of the algorithm $\mathcal{A}$ on sequences F and F' (see Algorithm 1, Step 7). Now by the definition of differential privacy (see Definition 2), $\tilde{\boldsymbol{x}}_{t+1}$ is $(\epsilon_1, \frac{\delta}{T})$-differential private, if for any measurable set $\Omega \subseteq \mathbb{R}^d$:

$$\Pr[\tilde{\boldsymbol{x}}_{t+1} \in \Omega] \le e^{\epsilon_1} \Pr[\tilde{\boldsymbol{x}}'_{t+1} \in \Omega] + \delta/T.$$

Recall that $\boldsymbol{b}_{t+1} \sim \mathcal{N}(0, \frac{\beta^2}{t^2}\mathbb{I}^d)$. We have $(\tilde{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1})^T \Delta\boldsymbol{x}_{t+1} = {\boldsymbol{b}_{t+1}}^T \Delta\boldsymbol{x}_{t+1} \sim \mathcal{N}(0, \frac{\beta^2}{t^2}\|\Delta\boldsymbol{x}_{t+1}\|_2^2)$, where $\Delta\boldsymbol{x}_{t+1} = \boldsymbol{x}_{t+1} - \boldsymbol{x}'_{t+1}$.

Also, using the low sensitivity property (1) of the OCP algorithm $\mathcal{A}$, $\|\Delta\boldsymbol{x}_{t+1}\| \le \frac{\lambda_{\mathcal{A}}}{t}$. Thus,

$$\begin{aligned}\Pr\left[\left|(\tilde{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1})^T \Delta\boldsymbol{x}_{t+1}\right| \ge \frac{\beta\lambda_{\mathcal{A}}}{t^2} z\right] &= \Pr\left[\left|{\boldsymbol{b}_{t+1}}^T \Delta\boldsymbol{x}_{t+1}\right| \ge \frac{\beta\lambda_{\mathcal{A}}}{t^2} z\right] \\ &\le \Pr\left[\left|{\boldsymbol{b}_{t+1}}^T \Delta\boldsymbol{x}_{t+1}\right| \ge \frac{\beta}{t}\|\boldsymbol{x}_{t+1} - \boldsymbol{x}'_{t+1}\|_2 z\right], \\ &\le e^{-\frac{z^2}{2}},\end{aligned}$$

where $z > 0$, and the second inequality follows from Mill's inequality. Setting R.H.S. $\le \frac{\delta}{T}$, we have $z \ge \sqrt{2\ln\frac{T}{\delta}}$.

Let $\mathcal{G} \subseteq \mathbb{R}^d$ be a "good" set defined by:

$$\boldsymbol{b} \in \mathcal{G} \text{ iff } \left|\boldsymbol{b}^T \Delta\boldsymbol{x}_{t+1}\right| \le \frac{\beta\lambda_{\mathcal{A}}}{t^2}\sqrt{2\ln\frac{T}{\delta}}. \tag{7}$$

Note that,

$$\Pr[\boldsymbol{b}_{t+1} \notin \mathcal{G}] = \Pr\left[\left|{\boldsymbol{b}_{t+1}}^T \Delta\boldsymbol{x}_{t+1}\right| \ge \frac{\beta\lambda_{\mathcal{A}}}{t^2}\sqrt{2\ln\frac{T}{\delta}}\right] \le \frac{\delta}{T}. \tag{8}$$

We now bound $\Pr[\tilde{\boldsymbol{x}}_{t+1} \in \Omega]$:

$$\Pr[\tilde{\boldsymbol{x}}_{t+1} \in \Omega] \le \Pr[\tilde{\boldsymbol{x}}_{t+1} \in \Omega \wedge \boldsymbol{b}_{t+1} \in \mathcal{G}] + \Pr[\boldsymbol{b}_{t+1} \notin \mathcal{G}] \le \Pr[\tilde{\boldsymbol{x}}_{t+1} \in \Omega \wedge \boldsymbol{b}_{t+1} \in \mathcal{G}] + \frac{\delta}{T}. \tag{9}$$

For the purpose of brevity, we define the following notation (which we will be using in the later parts of the proof): for a given set $S \subseteq \mathbb{R}^d$ and a vector $\boldsymbol{x} \in \mathbb{R}^d$, the set $\{\boldsymbol{y} : \boldsymbol{y} + \boldsymbol{x} \in S\}$ is denoted as $S - \boldsymbol{x}$.

Let us define $\Psi = \{\boldsymbol{x} : \left|(\boldsymbol{x} - \boldsymbol{x}_{t+1})^T \Delta \boldsymbol{x}_{t+1}\right| \leq \frac{\beta \lambda_{\mathcal{A}}}{t^2}\sqrt{2 \ln \frac{T}{\delta}}\}$. As $\boldsymbol{b}_{t+1} \sim \mathcal{N}(0, \frac{\beta^2}{t^2}\mathbb{I}^d)$,

$$\begin{aligned}
\Pr[\tilde{\boldsymbol{x}}_{t+1} \in \Omega \wedge \boldsymbol{b}_{t+1} \in \mathcal{G}] &= \int\limits_{\boldsymbol{b} \in \Omega - \boldsymbol{x}_{t+1} \cap \Psi - \boldsymbol{x}_{t+1}} \exp\left(-\frac{||\boldsymbol{b}||_2^2}{2\beta^2/t^2}\right) d\boldsymbol{b} \\
&= \int\limits_{\boldsymbol{x} \in \Omega \cap \Psi} \exp\left(-\frac{||\boldsymbol{x} - \boldsymbol{x}_{t+1}||_2^2}{2\beta^2/t^2}\right) d\boldsymbol{x} \qquad (10)
\end{aligned}$$

Now, for $\boldsymbol{x} \in \Omega \cap \Psi$:

$$\begin{aligned}
\frac{\exp\left(-\frac{t^2||\boldsymbol{x}-\boldsymbol{x}_{t+1}||_2^2}{2\beta^2}\right)}{\exp\left(-\frac{t^2||\boldsymbol{x}-\boldsymbol{x}'_{t+1}||_2^2}{2\beta^2}\right)} &= \exp\left(\frac{t^2}{2\beta^2}\Delta \boldsymbol{x}_{t+1}^T(2\boldsymbol{x} - \boldsymbol{x}_{t+1} - \boldsymbol{x}'_{t+1})\right), \\
&= \exp\left(\frac{t^2}{2\beta^2}\left(2\Delta \boldsymbol{x}_{t+1}^T(\boldsymbol{x} - \boldsymbol{x}_{t+1}) - \|\Delta \boldsymbol{x}_{t+1}\|_2^2\right)\right), \\
&\leq \exp\left(\frac{t^2}{2\beta^2}\left(2|\Delta \boldsymbol{x}_{t+1}^T(\boldsymbol{x} - \boldsymbol{x}_{t+1})| + \|\Delta \boldsymbol{x}_{t+1}\|_2^2\right)\right), \\
&\leq \exp\left(\frac{\lambda_{\mathcal{A}}}{\beta}\sqrt{2 \ln \frac{T}{\delta}} + \frac{\lambda_{\mathcal{A}}^2}{2\beta^2}\right), \\
&\leq e^{\epsilon_1}, \qquad (11)
\end{aligned}$$

where $\epsilon_1 = \frac{\sqrt{\epsilon}}{T^{0.5+c}}$ and β is as given in the Lemma statement. The second last inequality follows from the definition of $\mathcal{G}$ and the sensitivity assumption (1).

Hence, using (9), (10), and (11), we get:

$$\Pr[\tilde{\boldsymbol{x}}_{t+1} \in \Omega] \leq \int\limits_{\boldsymbol{x} \in \Omega \cap \Psi} e^{\epsilon_1} \exp\left(-\frac{t^2||\boldsymbol{x} - \boldsymbol{x}'_{t+1}||_2^2}{2\beta^2}\right) d\boldsymbol{x} + \frac{\delta}{T} \leq e^{\epsilon_1} \Pr[\tilde{\boldsymbol{x}}'_{t+1} \in \Omega] + \frac{\delta}{T}. \qquad (12)$$

This completes the proof.

■

We use the following result from Dwork et al. (2010b) in our proof of Theorem 1.

Lemma 2 (Dwork et al. (2010b)) *Suppose that random variables Y and Z satisfy $\max_x \frac{\Pr(Y=x)}{\Pr(Z=x)} \leq \epsilon$ and $\max_x \frac{\Pr(Z=x)}{\Pr(Y=x)} \leq \epsilon$. Then,*

$$\mathcal{D}(Y||Z) = \mathbb{E}_Z\left[\ln \frac{\Pr(Z=x)}{\Pr(Y=x)}\right] \leq \epsilon^2,$$

where $\mathcal{D}(Y||Z)$ is the KL-divergence between probability distribution of Y and Z.

A.2. Proof of Theorem 1

Proof Following the notation in the proof of Lemma 1, let $\mathcal{G}_{t+1}$ be the t-th step "good set" defined as:

$$\boldsymbol{b} \in \mathcal{G}_{t+1} \text{ iff } \left|\boldsymbol{b}^T \Delta \boldsymbol{x}_{t+1}\right| \leq \frac{\beta \lambda_{\mathcal{A}}}{t^2}\sqrt{2\ln\frac{T}{\delta}}, \tag{13}$$

where $\Delta \boldsymbol{x}_{t+1} = \boldsymbol{x}_{t+1} - \boldsymbol{x}'_{t+1}$ for $1 \leq t \leq T$.
Now, using (8), for each time step t,

$$\Pr[\boldsymbol{b}_{t+1} \notin \mathcal{G}_{t+1}] \leq \frac{\delta}{T}. \tag{14}$$

By union bound, the probability that every output vector $\boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}$ for $1 \leq t \leq T$, is at least $1 - T \cdot \frac{\delta}{T} = 1 - \delta$. That is,

$$\Pr[\exists t \text{ s.t. } \boldsymbol{b}_{t+1} \notin \mathcal{G}_{t+1}] \leq \delta. \tag{15}$$

For a random variable $\boldsymbol{x}$ and any point $\mathbf{a} \in \mathbb{R}^d$, let pdf$[\boldsymbol{x} = \mathbf{a}]$ denote the probability density function of the random variable $\boldsymbol{x}$ evaluated at the point $\mathbf{a}$.
Now, define the following sequence of functions with ξ being some event in the event space,

$$\mathbf{Z}_{t+1}(\mathbf{a}_{t+1};\xi) = \ln\left(\frac{\text{pdf}[\tilde{\boldsymbol{x}}_{t+1} = \mathbf{a}_{t+1} \mid \xi]}{\text{pdf}[\tilde{\boldsymbol{x}}'_{t+1} = \mathbf{a}_{t+1} \mid \xi]}\right),$$

where $\mathbf{a}_{t+1} \in \mathbb{R}^d$. Recall that $\tilde{\boldsymbol{x}}_{t+1} = \boldsymbol{x}_{t+1} + \boldsymbol{b}_{t+1}$ and $\tilde{\boldsymbol{x}}'_{t+1} = \boldsymbol{x}'_{t+1} + \boldsymbol{b}_{t+1}$. Hence, the pdfs in the above equation are associated with the random choice of the noise vectors $\boldsymbol{b}_{t+1}$ which is drawn from a multivariate Gaussian.

Using Lemma 1, we have that at each time step t, the output $\tilde{\boldsymbol{x}}_{t+1}$ of Algorithm 1 is $(\frac{\sqrt{\epsilon}}{T^{0.5+c}}, \frac{\delta}{T})$-differentially private. That is, for $1 \leq t \leq T$,

$$-\frac{\sqrt{\epsilon}}{T^{0.5+c}} \leq \mathbf{Z}_{t+1}(\mathbf{a}_{t+1};\boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}) = \ln\left(\frac{\text{pdf}[\tilde{\boldsymbol{x}}_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}]}{\text{pdf}[\tilde{\boldsymbol{x}}'_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}]}\right) \leq \frac{\sqrt{\epsilon}}{T^{0.5+c}}.$$

Using Lemma 2 along with the observation above, we obtain:

$$\mathbb{E}_{\boldsymbol{b}_{t+1}}\left[\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1};\boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1})\right] \leq \frac{2\epsilon}{T^{1+2c}}.$$

Now, let $L(\tilde{\boldsymbol{x}}_2, \cdots, \tilde{\boldsymbol{x}}_{T+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1}) = \sum_{t=1}^{T} \mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1})$. Since each $\boldsymbol{b}_{t+1}$ is sampled independently and the randomness in both $\tilde{\boldsymbol{x}}_{t+1}$ and $\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1})$ is only due to $\boldsymbol{b}_{t+1}$, therefore: i) for $1 \leq t \leq T$, $\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1}) = \mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1})$, and ii) each entry in the sequence $\tilde{\boldsymbol{x}}_{t+1}$s for $1 \leq t \leq T$ and each entry in the sequence $\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1})$s for $1 \leq t \leq T$ are independent. Now, using independence of $\tilde{\boldsymbol{x}}_{t+1}$'s and the bound given above,

$$\begin{aligned}\mathbb{E}_{\boldsymbol{b}_2,\cdots,\boldsymbol{b}_{T+1}}[L(\tilde{\boldsymbol{x}}_2, \cdots, \tilde{\boldsymbol{x}}_{T+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1})] &= \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{b}_{t+1}}\left[\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1})\right] \\ &\leq \frac{2T\epsilon}{T^{1+2c}} \leq \frac{2\epsilon}{T^{2c}} \leq 2\epsilon.\end{aligned}$$

Note also that for every $1 \le t \le T$ and $\mathbf{a}_{t+1} \in \mathbb{R}^d$, $|\mathbf{Z}_{t+1}(\mathbf{a}_{t+1}; \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1})| \le \frac{\sqrt{\epsilon}}{T^{0.5+c}}$ (from Lemma 1). Thus, using independence of $\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1})$s along with Azuma-Hoeffding inequality,

$$\Pr[L(\tilde{\boldsymbol{x}}_2, \cdots, \tilde{\boldsymbol{x}}_{T+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1}) \ge 2\epsilon + \epsilon] \le 2\exp\left(\frac{-2\epsilon^2}{T \times \frac{\epsilon}{T^{1+2c}}}\right) \le 2\exp\left(-2\epsilon T^{2c}\right). \tag{16}$$

Now, setting RHS $\le \delta$, we get: $\delta \ge 2\exp\left(-2\epsilon T^{2c}\right)$. Hence, we select $c = \frac{(\ln(\frac{1}{2\epsilon}\ln\frac{2}{\delta})}{2\ln T}$.

Using (16) along with the selected value of c, we have, with probability at least $1-\delta$ over the draws of $\mathbf{a}_{t+1}$ from $\tilde{\boldsymbol{x}}_{t+1}$,

$$\sum_{t=1}^{T} \ln\left(\frac{\mathrm{pdf}[\tilde{\boldsymbol{x}}_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}]}{\mathrm{pdf}[\tilde{\boldsymbol{x}}'_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}]}\right) \le 3\epsilon.$$

That is, with probability at least $1-\delta$, over the draw of $\forall \mathbf{a}_2, \ldots, \mathbf{a}_{T+1} \in \mathbb{R}^d$,

$$\Pi_{t=1}^{T} \mathrm{pdf}(\tilde{\boldsymbol{x}}_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}) \le e^{3\epsilon}\, \Pi_{t=1}^{T} \mathrm{pdf}(\tilde{\boldsymbol{x}}'_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}).$$

Hence, given that $\boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}$ for $1 \le t \le T$, with at least $1-\delta$ probability each $\tilde{\boldsymbol{x}}_{t+1}$ $(1 \le t \le T)$ is 3ϵ-differentially private.

Now, using (15), $\Pr[\exists t \text{ s.t. } \boldsymbol{b}_{t+1} \notin \mathcal{G}_{t+1}] \le \delta$. Hence, with probability at least $1-2\delta$ over the choice of $\boldsymbol{b}_2, \cdots, \boldsymbol{b}_{T+1}$, each $\tilde{\boldsymbol{x}}_{t+1}$ is 3ϵ-differentially private. Therefore, $(3\epsilon, 2\delta)$-differential privacy now follows using a standard argument similar to (9). ■

A.3. Proof of Theorem 2

Proof Let $\hat{\boldsymbol{x}}_1, \cdots, \hat{\boldsymbol{x}}_T$ be the output of the POCP algorithm. By the Lipschitz continuity of the cost functions f_t we have,

$$\begin{aligned}\sum_{t=1}^{T} f_t(\hat{\boldsymbol{x}}_t) - \min_{\boldsymbol{x}\in\mathcal{C}} \sum_{t=1}^{T} f_t(\boldsymbol{x}) &\le \sum_{t=1}^{T} f_t(\boldsymbol{x}_t) - \min_{\boldsymbol{x}\in\mathcal{C}} \sum_{t=1}^{T} f_t(\boldsymbol{x}) + L\sum_{t=1}^{T} ||\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t||_2, \\ &\le R_{\mathcal{A}}(T) + L\sum_{t=1}^{T} ||\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t||_2. \end{aligned} \tag{17}$$

Since at any time $t \ge 1$, $\hat{\boldsymbol{x}}_t$ is the projection of $\tilde{\boldsymbol{x}}_t$ on the convex set $\mathcal{C}$, we have

$$||\boldsymbol{x}_{t+1} - \hat{\boldsymbol{x}}_{t+1}||_2 \le ||\boldsymbol{x}_{t+1} - \tilde{\boldsymbol{x}}_{t+1}||_2 = ||\boldsymbol{b}_{t+1}||_2, \quad \forall 1 \le t \le T-1,$$

where $\boldsymbol{b}_{t+1}$ is the noise vector added in the t-th iteration of the POCP algorithm. Therefore,

$$L\sum_{t=1}^{T} ||\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t||_2 \le L\left(\|\mathcal{C}\|_2 + \sum_{t=1}^{T-1} ||\boldsymbol{b}_{t+1}||_2\right). \tag{18}$$

Now, $\boldsymbol{b}_{t+1} \sim \mathcal{N}(\mathbf{0}^d, \frac{\beta^2}{t^2}\mathbb{I}^d)$ where

$$\beta = \lambda_{\mathcal{A}} T^{0.5+c}\sqrt{\frac{2}{\epsilon}\left(\ln\frac{T}{\delta} + \frac{\sqrt{\epsilon}}{T^{0.5+c}}\right)}.$$

Therefore, $||\boldsymbol{b}_{t+1}||_2$ follows Chi-distribution with parameters $\mu = \frac{\sqrt{2}\frac{\beta}{t}\Gamma((d+1)/2)}{\Gamma(d/2)}$ and $\sigma^2 = \frac{\beta^2}{t^2}(d-\mu^2)$.

Using $c = \frac{\ln(\frac{1}{2\epsilon}\ln\frac{2}{\delta})}{2\ln T}$,

$$\begin{aligned}\mathbb{E}[\sum_{t=1}^{T-1}||\boldsymbol{b}_{t+1}||_2] &\le \frac{\sqrt{2}\beta\Gamma((d+1)/2)}{\Gamma(d/2)}\int_1^{T-1}\frac{1}{t}dt,\\ &\le \frac{\Gamma((d+1)/2)}{\Gamma(d/2)}\lambda_{\mathcal{A}}\sqrt{T}\ln T\sqrt{\frac{2}{\epsilon^2}\ln\frac{2}{\delta}\left(\ln\frac{T}{\delta} + \frac{\epsilon}{\sqrt{\frac{T}{2}\ln\frac{2}{\delta}}}\right)},\\ &\le 2\sqrt{d}\lambda_{\mathcal{A}}\sqrt{T}\frac{\ln^2\frac{T}{\delta}}{\epsilon}. \qquad (19)\end{aligned}$$

The theorem now follows by combining (17), (18), (19). ■

A.4. High-probability Utility Guarantee for Algorithm POCP

Corollary 3 *Let $L > 0$ be the maximum Lipschitz constant of any function f_t in the sequence F, $\mathcal{R}_{\mathcal{A}}(T)$, the regret of the non-private* OCP *algorithm $\mathcal{A}$ over T-time steps and $\lambda_{\mathcal{A}}$, the sensitivity parameter of $\mathcal{A}$ (see (1)). Then with probability at least $1-\gamma$, the regret of our Private* OCP *algorithm (Algorithm 1) satisfies:*

$$\sum_{t=1}^{T} f_t(\hat{\boldsymbol{x}}_t) - \min_{\boldsymbol{x}\in\mathcal{C}}\sum_{t=1}^{T} f_t(\boldsymbol{x}) \le 2\sqrt{d}L(\lambda_{\mathcal{A}} + \|\mathcal{C}\|_2)\sqrt{T}\frac{\ln^2\frac{T}{\delta}}{\epsilon\sqrt{\gamma}} + \mathcal{R}_{\mathcal{A}}(T),$$

where d is the dimensionality of the output space, $\|\mathcal{C}\|_2$ is the diameter of $\mathcal{C}$.

Appendix B. Implicit Gradient Descent Algorithm

In this section we consider the Implicit Gradient Descent (IGD) algorithm by Kulis and Bartlett (2010) and present a differentially private version using our generic framework (see Algorithm 1). At each step t, IGD selects the output $\boldsymbol{x}_{t+1}$ using:

$$\mathsf{IGD}: \qquad \boldsymbol{x}_{t+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{x}\in\mathcal{C}} \frac{1}{2}||\boldsymbol{x}-\boldsymbol{x}_t||_2^2 + \eta_t f_t(\boldsymbol{x}), \qquad (20)$$

where $\eta_t = \frac{1}{\alpha t}$, $\alpha > 0$ is the minimum strong convexity parameter of any f_t, $t \le T$. Now, if each $f_t(x)$ is a Lipschitz continuous strongly convex function, then a simple modification

Algorithm 2 Private Implicit Gradient Descent (PIGD)

```
Input: Cost function sequence $F = \langle f_1, \cdots, f_T \rangle$ and the convex set $\mathcal{C}$
Parameter: privacy parameters $(\epsilon, \delta)$, maximum Lipschitz constant $L$ and minimum strong convexity parameter $\alpha$ of any function in $F$
Choose $\boldsymbol{x}_1$ and $\hat{\boldsymbol{x}}_1$ randomly from $\mathcal{C}$
for $t = 1$ to $T - 1$ do
    Cost: $L_t(\hat{\boldsymbol{x}}_t) = f_t(\hat{\boldsymbol{x}}_t)$
    Learning rate: $\eta_t = \frac{1}{\alpha t}$
    IGD Update: $\boldsymbol{x}_{t+1} \leftarrow \mathrm{argmin}_{\boldsymbol{x} \in \mathcal{C}} \left( \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{x}_t\|_2^2 + \eta_t f_t(\boldsymbol{x}) \right)$
    Noise Addition: $\tilde{\boldsymbol{x}}_{t+1} \leftarrow \boldsymbol{x}_{t+1} + \boldsymbol{b}_{t+1}$, $\boldsymbol{b}_{t+1} \sim \mathcal{N}(\mathbf{0}^d, \frac{\beta^2}{t^2}\mathbb{I}^d)$, where $\beta = \frac{2LT^{0.5+c}}{\alpha}\sqrt{\frac{2}{\epsilon}\left(\ln\frac{T}{\delta} + \frac{\sqrt{\epsilon}}{T^{0.5+c}}\right)}$ and $c = \frac{\ln\frac{1}{2\epsilon}\ln(2/\delta)}{2\ln T}$
    Output $\hat{\boldsymbol{x}}_{t+1} = \mathrm{argmin}_{\boldsymbol{x} \in \mathcal{C}} \left( \|\boldsymbol{x} - \tilde{\boldsymbol{x}}_{t+1}\|_2^2 \right)$
end for
```

to the proof by Kulis and Bartlett (2010) shows $O(\log T)$ regret for IGD, i.e. $\mathcal{R}_{\mathsf{IGD}}(T) = O(\log T)$.

Now, we instantiate our generic POCP framework using the IGD algorithm. See Algorithm 2 for a pseudo-code of our Private IGD (PIGD) algorithm. Similar to POCP, our PIGD algorithm also adds an appropriately calibrated noise at each step to obtain differentially private outputs $\hat{\boldsymbol{x}}_{t+1}$.

Now, to use generic privacy analysis of our POCP framework, we need to show that IGD satisfies sensitivity bound of (1). To this end, in the following lemma we bound sensitivity of IGD at each step. At a high level, our proof uses optimality of each output $\boldsymbol{x}_{t+1}$ along with strong convexity of each f_t.

Lemma 4 (IGD Sensitivity) *L_2-sensitivity (see Definition 3) of the IGD algorithm is $\frac{2L}{\alpha t}$ for the t-th iterate, where L is the maximum Lipschitz constant of any function $f_\tau, 1 \le \tau \le t$.*

Proof [Proof of Lemma 4] We prove the lemma using mathematical induction.
Base Case ($t = 1$): As $\boldsymbol{x}_1$ is chosen randomly, it's value doesn't depend on the underlying dataset.
Induction Step ($t = \tau + 1$): Consider the following function that is optimized at the $(\tau + 1)$-step of IGD:

$$\tilde{f}_\tau(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}_\tau\|_2^2 + \eta_\tau f_\tau(\boldsymbol{x}).$$

As f_τ is α strongly convex, the strong convexity coefficient of the above given function is $\frac{\tau+1}{\tau}$.

Now using strong convexity of $\tilde{f}_\tau$ and the fact that at optima $\boldsymbol{x}_{\tau+1}$, $\langle \bigtriangledown \tilde{f}_\tau(\boldsymbol{x}_{\tau+1}), \boldsymbol{x} - \boldsymbol{x}_{\tau+1} \rangle \ge 0, \forall \boldsymbol{x} \in \mathcal{C}$, we get:

$$\tilde{f}_\tau(\boldsymbol{x}'_{\tau+1}) \ge \tilde{f}_\tau(\boldsymbol{x}_{\tau+1}) + \frac{\tau+1}{2\tau}\|\boldsymbol{x}_{\tau+1} - \boldsymbol{x}'_{\tau+1}\|_2^2. \tag{21}$$

Now, we consider two cases:

- $F - F' = \{f_\tau\}$: Define $\tilde{f}'_\tau(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{x}_\tau\|^2 + \eta_\tau f'_\tau(\boldsymbol{x})$ and let $\boldsymbol{x}'_{\tau+1} = \mathrm{argmin}_{\boldsymbol{x}\in\mathcal{C}}\, \tilde{f}'_\tau(\boldsymbol{x})$. Then, similar to (21), we get:

  $$\tilde{f}'_\tau(\boldsymbol{x}_{\tau+1}) \ge \tilde{f}'_\tau(\boldsymbol{x}'_{\tau+1}) + \frac{\tau+1}{2\tau}\|\boldsymbol{x}_{\tau+1}-\boldsymbol{x}'_{\tau+1}\|_2^2. \tag{22}$$

  Adding (21) and (22), we get:

  $$\|\boldsymbol{x}_{\tau+1}-\boldsymbol{x}'_{\tau+1}\|_2^2 \le \frac{1}{\alpha(\tau+1)}|f_\tau(\boldsymbol{x}'_{\tau+1})+f'_\tau(\boldsymbol{x}_{\tau+1})-f_\tau(\boldsymbol{x}_{\tau+1})-f'_\tau(\boldsymbol{x}'_{\tau+1})| \le \frac{2L}{\alpha(\tau+1)}\|\boldsymbol{x}_{\tau+1}-\boldsymbol{x}'_{\tau+1}\|_2.$$

  Lemma now follows using simplification.

- $F - F' = \{f_i\}$, $i < \tau$: Define $\tilde{f}'_\tau(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{x}'_\tau\|^2 + \eta_\tau f_\tau(\boldsymbol{x})$ and let $\boldsymbol{x}'_{\tau+1} = \mathrm{argmin}_{\boldsymbol{x}\in\mathcal{C}}\, \tilde{f}'_\tau(\boldsymbol{x})$. Then, similar to (21), we get:

  $$\tilde{f}'_\tau(\boldsymbol{x}_{\tau+1}) \ge \tilde{f}'_\tau(\boldsymbol{x}'_{\tau+1}) + \frac{\tau+1}{2\tau}\|\boldsymbol{x}_{\tau+1}-\boldsymbol{x}'_{\tau+1}\|_2^2. \tag{23}$$

  Adding (21) and (23), we get:

  $$\|\boldsymbol{x}_{\tau+1}-\boldsymbol{x}'_{\tau+1}\|_2^2 \le \frac{\tau}{\tau+1}|(\boldsymbol{x}_{\tau+1}-\boldsymbol{x}'_{\tau+1})\cdot(\boldsymbol{x}_\tau-\boldsymbol{x}'_\tau)| \le \frac{\tau}{\tau+1}\|\boldsymbol{x}_{\tau+1}-\boldsymbol{x}'_{\tau+1}\|_2\|\boldsymbol{x}_\tau-\boldsymbol{x}'_\tau\|_2.$$

  The lemma now follows using the induction hypothesis.

■

Using the above lemma and Theorem 1, privacy guarantee for PIGD follows directly.

Theorem 7 (PIGD Privacy) PIGD *(see Algorithm 2) is* $(3\epsilon, 2\delta)$*-differentially private.*

Next, the utility (regret) analysis of our PIGD algorithm follows directly using Theorem 2 along with the regret bound of the IGD algorithm, $\mathcal{R}_{\mathsf{IGD}}(T) = O((\frac{L^2}{\alpha} + ||\mathcal{C}||_2)\log T)$.

Theorem 8 (PIGD Regret) *Let L be the maximum Lipschitz constant and let α be the minimum strong convexity parameter of any function f_t in the function sequence F. Then the expected regret of the private* IGD *algorithm over T steps is $\tilde{O}(\sqrt{dT})$. Specifically,*

$$\mathbb{E}[\sum_{t=1}^{T} f_t(\hat{\boldsymbol{x}}_t)] - \min_{\boldsymbol{x}\in\mathcal{C}}\sum_{t=1}^{T} f_t(\boldsymbol{x})) \le C\left(\frac{(L^2/\alpha + \|\mathcal{C}\|_2)\sqrt{d}\ln^2\frac{T}{\delta}}{\epsilon}\sqrt{T}\right),$$

where $C > 0$ is a constant and d is the dimensionality of the output space.

In this section and in Appendix C, we provide transformation of two standard online learning algorithms into corresponding privacy preserving algorithms with provable regret. In both these examples, we show low-sensitivity of the corresponding learning algorithms and use our analysis of POCP to obtain privacy and utility bounds. We can obtain similar low-sensitivity bounds for several other OCP algorithms such as Follow The Leader (FTL), Follow the Regularized Leader (FTRL) etc, and hence use those methods with our POCP framework as well. Our low-sensitivity proofs should be of independent interest as well, as they point to a connection between stability (sensitivity) and low-regret (online learnability)—an active topic of research in the learning community (Ross and Bagnell, 2011; Poggio et al., 2011).

Algorithm 3 Private GIGA (PGIGA)

```
1: **Input:** Cost function sequence $F = \langle f_1, \cdots, f_T \rangle$ and the convex set $\mathcal{C}$
2: **Parameter:** Privacy parameters $(\epsilon, \delta)$, Lipschitz continuity $(L)$ and strong convexity $(\alpha)$ bound on the function sequence $F$, $t_q = 2L_G^2/\alpha^2$
3: Choose $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t_q-1}$ and $\hat{\boldsymbol{x}}_1, \ldots, \hat{\boldsymbol{x}}_{t_q-1}$ randomly from $\mathcal{C}$, incurring a cost of $\sum_{t=1}^{t_q-1} f_t(\hat{\boldsymbol{x}}_t)$
4: **for** $t = t_q$ to $T-1$ **do**
5:     **Cost:** $L_t(\hat{\boldsymbol{x}}_t) = f_t(\hat{\boldsymbol{x}}_t)$
6:     **Step Size:** $\eta_t = \frac{2}{\alpha t}$
7:     GIGA **Update:** $\boldsymbol{x}_{t+1} \leftarrow \mathrm{argmin}_{x \in \mathcal{C}} \left( \|\boldsymbol{x}_t - \eta_t \bigtriangledown f_t(\boldsymbol{x}_t)\|_2^2 \right)$
8:     **Noise Addition:** $\tilde{\boldsymbol{x}}_{t+1} \leftarrow \boldsymbol{x}_{t+1} + \boldsymbol{b}_{t+1}$, $\boldsymbol{b}_{t+1} \sim \mathcal{N}(\mathbf{0}^d, \frac{\beta^2}{t^2}\mathbb{I}^d)$, where $\beta = \frac{2GT^{0.5+c}}{\alpha}\sqrt{\frac{2}{\epsilon}\left(\ln\frac{T}{\delta} + \frac{\sqrt{\epsilon}}{T^{0.5+c}}\right)}$ where $c = \frac{\ln\frac{1}{2\epsilon}\ln(2/\delta)}{2\ln T}$
9:     Output $\hat{\boldsymbol{x}}_{t+1} = \mathrm{argmin}_{\boldsymbol{x} \in \mathcal{C}} \left( \|\boldsymbol{x} - \tilde{\boldsymbol{x}}_{t+1}\|_2^2 \right)$
10: **end for**
```

Appendix C. Private GIGA Algorithm

In this section, we apply our general differential privacy framework to the Generalized Infinitesimal Gradient Ascent (GIGA) algorithm (Zinkevich, 2003), which is one of the most popular algorithms for OCP. GIGA is a simple extension of the classical projected gradient method to the OCP problem. Specifically, the iterates $\boldsymbol{x}_{t+1}$ are obtained by a projection onto the convex set $\mathcal{C}$, of the output of the gradient descent step $\boldsymbol{x}_t - \eta_t \bigtriangledown f_t(\boldsymbol{x}_t)$ where $\eta_t = 1/\alpha t$, and α is the minimum strong convexity parameter of any function f_t in F.

For the rest of this section, we assume that each of the function f_t in the input function sequence F are *differentiable*, Lipschitz continuous gradient and strongly convex. Note that this is a stricter requirement than our private IGD algorithm where we require only the Lipschitz continuity of f_t.

Proceeding similar to IGD, we obtain a privacy preserving version of the GIGA algorithm using our generic POCP framework (See Algorithm 1). Algorithm 3 details the steps involved in our Private GIGA (PGIGA) algorithm. Note that PGIGA has an additional step (Step 3) compared to POCP (Algorithm 1). This step is required to prove the sensitivity bound in Lemma 5 given below.

Furthermore, we provide the privacy and regret guarantees for our PGIGA algorithm using Theorem 1 and Theorem 2. To this end, we first show that GIGA satisfies the sensitivity assumption mentioned in (1).

Lemma 5 (GIGA Sensitivity) *Let $\alpha > 0$ be the minimum strong convexity parameter of any function f_t in the function sequence F. Also, let L_G be the maximum Lipschitz continuity parameter of the gradient of any function $f_t \in F$ and let $G = \max_{\mathcal{T}} \| \bigtriangledown f_t(x)\|_2, \forall x \in \mathcal{C}$. Then, L_2-sensitivity (see Definition 3) of the GIGA algorithm is $\frac{2G}{\alpha t}$ for the t-th iterate, where $1 \leq t \leq T$.*

Proof Let $\boldsymbol{x}_{t+1}$ and $\tilde{\boldsymbol{x}}'_{t+1}$ be the t-th iterates when GIGA is applied to F and F', respectively. Using this notation, to prove the L_2 sensitivity of GIGA, we need to show that:

$$\|\boldsymbol{x}_{t+1} - \boldsymbol{x}'_{t+1}\| \leq \frac{2G}{\alpha t}$$

We prove the above inequality using mathematical induction.
Base Case $(1 \leq t \leq t_q = 2L_G^2/\alpha^2 + 1)$: As $\boldsymbol{x}_1, \dots, \boldsymbol{x}_{t_q}$ are selected randomly, their value doesn't depend on the underlying dataset. Hence, $\boldsymbol{x}_t = \boldsymbol{x}'_t, \forall 1 \leq t \leq t_q$.
Induction Step $t = \tau > 2L_G^2/\alpha^2 + 1$: We consider two cases:

- $F - F' = \{f_\tau\}$: Since the difference between F and F' is only the τ-th function, hence $\boldsymbol{x}_\tau = \boldsymbol{x}'_\tau$. As $\mathcal{C}$ is a convex set, projection onto $\mathcal{C}$ always decreases distance, hence:

$$\begin{aligned}\|\boldsymbol{x}_{\tau+1} - \boldsymbol{x}'_{\tau+1}\|_2 &\leq \|(\boldsymbol{x}_\tau - \eta_\tau \bigtriangledown f_\tau(\boldsymbol{x}_\tau)) - (\boldsymbol{x}_\tau - \eta_\tau \bigtriangledown f'_\tau(\boldsymbol{x}_\tau))\|_2, \\ &= \eta_\tau \| \bigtriangledown f_\tau(\boldsymbol{x}_\tau) - \bigtriangledown f'_\tau(\boldsymbol{x}_\tau)\|_2, \\ &\leq \frac{2G}{\alpha\tau}.\end{aligned}$$

  Hence, lemma holds in this case.

- $F - F' = \{f_i\},\ i < \tau$: Again using convexity of $\mathcal{C}$, we get:

$$\begin{aligned}\|\boldsymbol{x}_{\tau+1} - \boldsymbol{x}'_{\tau+1}\|_2^2 &\leq \|(\boldsymbol{x}_\tau - \eta_\tau \bigtriangledown f_\tau(\boldsymbol{x}_\tau)) - (\boldsymbol{x}'_\tau - \eta_\tau \bigtriangledown f_\tau(\boldsymbol{x}'_\tau))\|_2^2, \\ &= \|\boldsymbol{x}_\tau - \boldsymbol{x}'_\tau\|_2^2 + \eta_\tau^2 \| \bigtriangledown f_\tau(\boldsymbol{x}_\tau) - \bigtriangledown f_\tau(\boldsymbol{x}'_\tau)\|_2^2 \\ &\qquad - 2\eta_\tau(\boldsymbol{x}_\tau - \boldsymbol{x}'_\tau)^T(\bigtriangledown f_\tau(\boldsymbol{x}_\tau) - \bigtriangledown f_\tau(\boldsymbol{x}'_\tau)), \\ &\leq (1 + \eta_\tau^2 L_G^2)\|\boldsymbol{x}_\tau - \boldsymbol{x}'_\tau\|_2^2 - 2\eta_\tau(\boldsymbol{x}_\tau - \boldsymbol{x}'_\tau)^T(\bigtriangledown f_\tau(\boldsymbol{x}_\tau) - \bigtriangledown f_\tau(\boldsymbol{x}'_\tau)),\end{aligned} \tag{24}$$

  where the last equation follows using Lipschitz continuity of $\bigtriangledown f_t$. Now, using strong convexity:

$$(\boldsymbol{x}_\tau - \boldsymbol{x}'_\tau)^T(\bigtriangledown f_\tau(\boldsymbol{x}_\tau) - \bigtriangledown f_\tau(\boldsymbol{x}'_\tau)) \geq \alpha \|\boldsymbol{x}_\tau - \boldsymbol{x}'_\tau\|_2^2.$$

  Combining the above observation and the induction hypothesis with (24):

$$\|\boldsymbol{x}_{\tau+1} - \boldsymbol{x}'_{\tau+1}\|_2^2 \leq \left(1 + L_G^2\eta_\tau^2 - 2\alpha\eta_\tau\right) \cdot \frac{4G^2}{(\tau-1)^2}. \tag{25}$$

  Lemma now follows by setting $\eta_\tau = \frac{2}{\alpha\tau}$ and $\tau > \frac{2L_G^2}{\alpha^2}$.

■

Using the lemma above with the privacy analysis of POCP (Theorem 1), the privacy guarantee for PGIGAfollows immediately.

Theorem 9 (PGIGA Privacy) PGIGA *(see Algorithm 3) is* $(3\epsilon, 2\delta)$*-differentially private.*

Next, using the regret bound analysis for GIGA from Hazan et al. (2007) (Theorem 1) along with Theorem 2, we get the following utility (regret bound) analysis for our PGIGA algorithm. Here again, ignoring constants, the regret simplifies to $\tilde{O}(\sqrt{dT})$.

Algorithm 4 Private Follow the Leader for Quadratic Cost (PQFTL)

```
Input: cost function sequence $F = \langle f_1, \cdots, f_T \rangle$, where each $f_t(x; y_t, \boldsymbol{v}_t) = (y_t - \boldsymbol{v}_t^T \boldsymbol{x})^2 + \frac{\alpha}{2}||\boldsymbol{x}||_2^2$
Parameter: privacy parameters $(\epsilon, \delta)$, $R = \max(\max_t ||v_t||_2, \max_t |y_t|)$
Initialize $\hat{\boldsymbol{x}}_1 = 0^d$
Initialize empty binary trees $B^V$ and $B^{\boldsymbol{u}}$, a data structure to compute $\hat{V}_t$ and $\hat{\boldsymbol{u}}_t$—differentially private versions of $V_t$ and $\boldsymbol{u}_t$
for $t = 1$ to $T - 1$ do
    Cost: $L_t(\hat{\boldsymbol{x}}_t) = f_t(\hat{\boldsymbol{x}}_t) = (y_t - \boldsymbol{v}_t^T \hat{\boldsymbol{x}}_t)^2 + \frac{\alpha}{2}||\hat{\boldsymbol{x}}_t||_2^2$
    $(\hat{V}_t, B^V) \leftarrow \text{PrivateSum}(\boldsymbol{v}_t \boldsymbol{v}_t^T, B^V, t, R^2, \frac{\epsilon}{2}, \frac{\delta}{2}, T)$ (see Algorithm 5)
    $(\hat{\boldsymbol{u}}_t, B^{\boldsymbol{u}}) \leftarrow \text{PrivateSum}(y_t \boldsymbol{v}_t, B^{\boldsymbol{u}}, t, R, \frac{\epsilon}{2}, \frac{\delta}{2}, T)$ (see Algorithm 5)
    QFTL Update: $\hat{\boldsymbol{x}}_{t+1} \leftarrow (t\alpha\mathbb{I} + \hat{V}_t)^{-1}(\hat{\boldsymbol{u}}_t)$
    Output $\hat{\boldsymbol{x}}_{t+1}$
end for
```

Theorem 10 (PGIGA Regret) *Let $\alpha > 0$ be the minimum strong convexity parameter of any function f_t in the function sequence F. Also, let L_G be the maximum Lipschitz continuity parameter of the gradient of any function $f_t \in F$ and let $G = \max_\tau \| \triangledown f_t(x)\|_2, \forall x \in \mathcal{C}$. Then, the expected regret of PGIGA satisfies*

$$\mathbb{E}[\mathcal{R}_{\mathsf{PGIGA}}(T)] \leq \frac{4\sqrt{d}(G/\alpha + \|\mathcal{C}\|_2)G \ln^2 \frac{T}{\delta}}{\epsilon}\sqrt{T} + \frac{2G^2}{\alpha}(1 + \log T) + \frac{2L_G^2 G||\mathcal{C}||_2}{\alpha^2}$$

where $||\mathcal{C}||_2$ is the diameter of the convex set $\mathcal{C}$ and d is the dimensionality of the output space.

Proof Observe that for the first $t_q = \frac{2L_G^2}{\alpha^2}$ iterations PGIGA outputs random samples from $\mathcal{C}$. The additional regret incurred during this time is bounded by a constant (w.r.t. T) that appears as the last term in the regret bound given above. For iterations $t \geq t_q$, the proof follows directly by using Theorem 2 and regret bound of GIGA. Note that we use a slightly modified step-size $\eta_t = 2/\alpha t$, instead of the standard $\eta_t = 1/\alpha t$. This difference in the step size increases the regret of GIGA as given by Hazan et al. (2007) by a factor of 2. ■

Appendix D. Logarithmic regret for Quadratic Cost Functions: Appendix

D.1. Privacy and Utility Analysis of PQFTL for Quadratic Cost Functions

Proof [Proof of Theorem 3] Using Theorem 11 (stated in Section D.2), both $\hat{V}_t$ and $\hat{\boldsymbol{u}}_t$ are each $(\frac{\epsilon}{2}, \frac{\delta}{2})$-differentially private w.r.t. v_t and y_t, $\forall t$ and hence w.r.t. the function sequence F. Now, $\hat{\boldsymbol{x}}_{t+1}$ depends on F only through $[\hat{V}_t, \hat{\boldsymbol{u}}_t]$. Hence, the theorem follows using a standard composition argument (Dwork et al., 2006b; Dwork and Lei, 2009). ■

Proof [Proof of Theorem 4] Using definition of regret,

$$\begin{aligned}\mathcal{R}_{\mathsf{PQFTL}} = \sum_{t=1}^{T} f_t(\hat{\boldsymbol{x}}_t) - \operatorname*{argmin}_{\boldsymbol{x}^*} \sum_{t=1}^{T} f_t(\boldsymbol{x}^*) &= \sum_{t=1}^{T} f_t(\boldsymbol{x}_t) - \operatorname*{argmin}_{\boldsymbol{x}^*} \sum_{t=1}^{T} f_t(\boldsymbol{x}^*) + \sum_{t=1}^{T}(f_t(\hat{\boldsymbol{x}}_t) - f_t(\boldsymbol{x}_t)),\\ &\le \mathcal{R}_{\mathsf{QFTL}}(T) + \sum_{t=1}^{T}(f_t(\hat{\boldsymbol{x}}_t) - f_t(\boldsymbol{x}_t)),\\ &\le \frac{R^4(1+2R/\alpha)^2}{\alpha}\log T + \sum_{t=1}^{T}(f_t(\hat{\boldsymbol{x}}_t) - f_t(\boldsymbol{x}_t)),\end{aligned} \tag{26}$$

where last inequality follows using (5).

Now, as $f_t(\boldsymbol{x})$ is a $(R+\alpha)$-Lipschitz continuous gradient function,

$$\begin{aligned}f_t(\hat{\boldsymbol{x}}_t) - f_t(\boldsymbol{x}_t) &\le ((\boldsymbol{v}_t^T\boldsymbol{x}_t - y_t)\boldsymbol{v}_t + \alpha\boldsymbol{x}_t)^T(\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t) + \frac{R+\alpha}{2}\|\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t\|^2,\\ &\le R(2R^2/\alpha + R + 2)\|\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t\| + \frac{R+\alpha}{2}\|\hat{\boldsymbol{x}}_t - \boldsymbol{x}_t\|^2,\end{aligned} \tag{27}$$

where last inequality follows using Cauchy-Schwarz inequality and the fact that $\|\boldsymbol{x}_t\|_2 \le 2R/\alpha$.

We now bound $||\hat{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1}||_2$. Let $\hat{V}_t = V_t + A_t$ and $\hat{\boldsymbol{u}}_t = \boldsymbol{u}_t + \beta_t$ where A_t and β_t are the noise additions introduced by the Private Sum algorithm (Algorithm 5).

Now, from the step 9 of PQFTL (Algorithm 4) we have,

$$(\hat{V}_t + t\alpha\mathbb{I})\hat{\boldsymbol{x}}_{t+1} = \hat{\boldsymbol{u}}_t \quad \Leftrightarrow \quad (\frac{1}{t}\hat{V}_t + \alpha\mathbb{I})\hat{\boldsymbol{x}}_{t+1} = \frac{1}{t}\hat{\boldsymbol{u}}_t. \tag{28}$$

Similarly, using QFTL update (see (4)) we have,

$$(\frac{1}{t}V_t + \alpha\mathbb{I})\boldsymbol{x}_{t+1} = \frac{1}{t}\boldsymbol{u}_t. \tag{29}$$

Using (28) and (29):

$$(\frac{1}{t}\hat{V}_t + \alpha\mathbb{I})(\hat{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1}) = \frac{1}{t}\beta_t - \frac{1}{t}A_t\boldsymbol{x}_{t+1}. \tag{30}$$

Now, using $\hat{V}_t = V_t + A_t$ and the triangle inequality we have,

$$||(\frac{1}{t}\hat{V}_t + \alpha\mathbb{I})(\hat{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1})||_2 \ge ||(\frac{1}{t}V_t + \alpha\mathbb{I})(\hat{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1})||_2 - ||\frac{1}{t}A_t(\hat{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1})||_2 \tag{31}$$

Furthermore,

$$||\frac{1}{t}A_t(\hat{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1})||_2 \le \frac{1}{t}||A_t||_2||\hat{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1}||_2 \tag{32}$$

Thus by combining (30), (31), (32) and using the fact that the smallest eigenvalue of $(\frac{1}{t}V_t + \alpha\mathbb{I})$ is lower-bounded by α,

$$\frac{1}{t}||\beta_t||_2 + \frac{1}{t}\|A_t\|_2\|\boldsymbol{x}_{t+1}\|_2 \ge |\alpha - \frac{||A_t||_2}{t}||\hat{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1}||_2 \tag{33}$$

Now using Theorem 11 each entry of the matrix A_t is drawn from $\mathcal{N}(0, \sigma^2 \log T)$ for $\sigma^2 = \frac{R^2}{\epsilon^2} \log^2 T \log \frac{\log T}{\delta}$. Thus the spectral norm of A_t, $||A_t||_2$ is bounded by $3\sigma\sqrt{d}$ with probability at least $1 - \exp(-d/2)$. Similarly, $||\beta_t||_2 \leq 3\sigma\sqrt{d}$, with probability at least $1 - \exp(-d/2)$. Also, $||\boldsymbol{x}_t||_2 \leq 2R/\alpha$. Using the above observation with (33),

$$||\hat{\boldsymbol{x}}_{t+1} - \boldsymbol{x}_{t+1}||_2 \leq \frac{\sigma\sqrt{d}}{t} \cdot \frac{3 + 6R/\alpha}{|\alpha - \frac{6\sigma\sqrt{d}R}{\alpha t}|}. \tag{34}$$

Using (26), (27), and (34), we get (with probability at least $1 - \exp(-d/2)$):

$$\begin{aligned}\mathcal{R}_{\mathsf{PQFTL}}(T) \leq & \frac{R^4(1 + 2R/\alpha)^2}{\alpha} \log T \\ & + 3\sqrt{d}R(2R^2/\alpha + R + 2)(1 + 2R/\alpha)(1 + \log T)\frac{1}{\epsilon}\sqrt{\log T} \log \sqrt{\frac{\log T}{\delta}}. \end{aligned} \tag{35}$$

Hence w.h.p.,

$$\mathcal{R}_{\mathsf{PQFTL}}(T) = \tilde{O}\left(\frac{\mathcal{R}^6 \log \frac{1}{\delta}}{\epsilon\alpha^3} \sqrt{d} \log^{1.5} T\right).$$

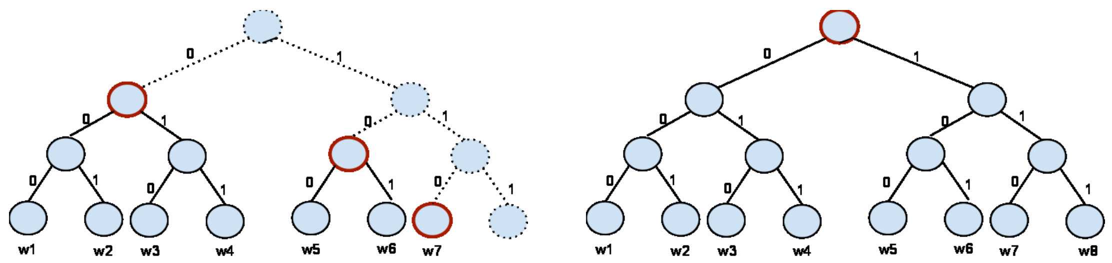
■

D.2. Computing Partial Sums Privately

In this section, we consider the problem of computing partial sums while preserving differential privacy. Formally, let $D = \langle \boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_T \rangle$ be a sequence of vectors, where at each time step t, a vector $\boldsymbol{w}_t \in \mathbb{R}^d$ is provided. Now, the goal is to output partial sums $W_t = \sum_{\tau=1}^{t} \boldsymbol{w}_\tau$ at each time step t, without compromising the privacy of the data vectors in D. Note that by treating a matrix as a long vector obtained by concatenation of its rows, we can use the same approach to compute partial sums of matrices as well.

Now, notice that the L_2-sensitivity of each partial sum is $O(R)$ ($R = \max_t \|\boldsymbol{w}_t\|_2$), as changing one w_τ can change any partial sum by an additive factor of $2R$. Hence, a naïve method is to add $O(R\frac{\sqrt{\log \frac{1}{\delta}}}{\epsilon})$ noise at t-th to obtain (ϵ, δ)-privacy for the output at a fixed step t. Using standard composition argument, the overall privacy of such a scheme over T iterations would be $(T\epsilon, T\delta)$. Hence, to get a constant (ϵ', δ') privacy, we would need to add $O(RT\frac{\sqrt{\log \frac{T}{\delta'}}}{\epsilon'})$ noise. In contrast, our method, which is based on a generalization of the technique in Dwork et al. (2010a), is able to provide the same level of privacy by adding only $O(R \log T \frac{\sqrt{\log \frac{\log T}{\delta'}}}{\epsilon'})$ noise. We first provide a high level description of the algorithm and then provide a detailed privacy and utility analysis.

Following Dwork et al. (2010a), we first create a binary tree $\boldsymbol{B}$ where each leaf node corresponds to an input vector in D. We denote a node at level i (root being at level 0) with strings in $\{0,1\}^i$ in the following way: For a given node in level i with label $s \in \{0,1\}^i$, the left child of s is denoted with the label $s \circ 0$ and the right child is denoted with $s \circ 1$. Here



(a)

(b)

Figure 1: Binary Tree for $T = 8$. Each node in the tree has noise drawn according to $\mathcal{N}(0, \sigma^2 \mathbb{I}^d)$ including the leaves. The label of any node is obtained by concatenating the labels of the edges on the unique path joining the root to the node. **(a)**: $\boldsymbol{w}_1, \boldsymbol{w}_2, .., \boldsymbol{w}_7$ are the input vectors that have arrived till time step $t = 7$. Each internal node is obtained by adding noise from $\mathcal{N}(0, \sigma^2 \mathbb{I}^d)$ to the sum of input vectors in the sub-tree rooted at the node. To return the partial sum at $t = 7$, return the sum of the nodes in thick red. The dotted nodes are unpopulated. **(b)**: figure depicts the change in the data structure after the arrival of $\boldsymbol{w}_8$. Now the partial sum at $t = 8$ is obtained by using just one node denoted in thick red.

the operator $\circ$ denotes concatenation of strings. Also, the root is labeled with the empty string .

Now, each node s in the tree $\boldsymbol{B}$ contains two values: $\boldsymbol{B}_s$ and $\hat{\boldsymbol{B}}_s$, where $\boldsymbol{B}_s$ is obtained by the summation of vectors in each of the leaves of the sub-tree rooted at s, i.e., $\boldsymbol{B}_s = \sum_{\substack{j: j = s \circ r \\ r \in \{0,1\}^{k-i}}} \boldsymbol{w}_j$. Also, $\hat{\boldsymbol{B}}_s = \boldsymbol{B}_s + \boldsymbol{b}_s$ is a perturbation of $\boldsymbol{B}_s$, $\boldsymbol{b}_s \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^d)$, and σ is as given in Lemma 6.

A node in the tree is populated only when all the vectors that form the leaves of the sub-tree rooted at the node have arrived. Hence, at time instant t we receive vector $\boldsymbol{w}_t$ and populate the nodes in the tree $\boldsymbol{B}$ for which all the leaves in the sub-tree rooted at them have arrived. To populate a node labeled s, we compute $\boldsymbol{B}_s = \boldsymbol{B}_{s\circ 0} + \boldsymbol{B}_{s \circ 1}$, the sum of the corresponding values at its two children in the tree and also $\hat{\boldsymbol{B}}_s = \boldsymbol{B}_s + \boldsymbol{b}_s$, $\boldsymbol{b}_s \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^d)$.

As we prove below in Lemma 6, for a i-th level node which is populated and has label $s \in \{0, 1\}^i$, $\hat{\boldsymbol{B}}_s$ contains an (ϵ, δ)-private sum of the 2^{k-i} vectors that correspond to the leaves of the sub-tree rooted at s. Now, to output a differentially private partial sum at time step t, we add up the perturbed values at the highest possible nodes that can be used to compute the sum. Note, that such a summation would have at most one node at each level. See Figure 1 for an illustration. We provide a pseudo-code of our method in Algorithm 5.

Theorem 11 states privacy as well as utility guarantees of our partial sums method (Algorithm 5). We first provide a technical lemma which we later use in our proof of Theorem 11.

Let $\hat{\boldsymbol{B}}(D)$ denote the set of all perturbed node values $\hat{\boldsymbol{B}}_s, \forall s$ obtained by applying Algorithm 5 on dataset D. Also, D and D' be two datasets that differ in at most one entry, say $\boldsymbol{w}_t$.

Algorithm 5 Private Sum$(\boldsymbol{w}_t, \boldsymbol{B}, t, R, \epsilon, \delta, T)$

```
Require: Data vector $\boldsymbol{w}_t$, current binary tree $\boldsymbol{B}$, current vector number $t$, $R$ a bound on $||\boldsymbol{w}_t||_2$, privacy parameters $\epsilon$ and $\delta$, total number of vectors $T$, dimensionality of vectors $d$
if $t = 1$ then
    Initialize the binary tree $\boldsymbol{B}$ over $T$ leaves with all nodes
    $\sigma^2 \leftarrow \frac{R^2}{\epsilon^2} \log^2 T \log \frac{\log T}{\delta}$
end if
$s_t \leftarrow$ the string representation of $t$ in binary
$\boldsymbol{B}_{s_t} \leftarrow \boldsymbol{w}_t$          //Populate the $s_t$-th entry of $\boldsymbol{B}$
$\hat{\boldsymbol{B}}_{s_t} \leftarrow \boldsymbol{B}_{s_t} + \boldsymbol{b}_{s_t}$, where $\boldsymbol{b}_{s_t} \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^d)$
Let $S_t$ is the set of all ancestors $s$ of $s_t$ in the tree $\boldsymbol{B}$, such that all the leaves in the sub-tree rooted at $s$ are already populated
for all $s \in S_t$ do
    $\boldsymbol{B}_s \leftarrow \boldsymbol{B}_{s\circ 0} + \boldsymbol{B}_{s\circ 1}$          // $B_s$ is the value at node with label $s$ (without noise)
    $\hat{\boldsymbol{B}}_s \leftarrow \boldsymbol{B}_s + \boldsymbol{b}_s$, where $\boldsymbol{b}_s \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^d)$          // $\hat{B}_s$ is the noisy value at node with label $s$
end for
Find the minimum set of *already* populated nodes in $\boldsymbol{B}$ that can compute $\sum_{\tau=1}^{t} \boldsymbol{w}_\tau$. Formally, starting from the left, for each bit position $i$ in $s_t$ such that $s_t(i) = 1$, form strings $s^q = s_t(1) \circ \ldots \circ s_t(i-1) \circ 0$ of length $i$. Let $s^1, s^2, \ldots, s^Q$ be all such strings, where $Q \le \log T$. For example, if $s_t = 110$ then the strings obtained this way are: 0 and 10
Output: $(\hat{W}_t = \sum_{q=1}^{Q} \hat{\boldsymbol{B}}_{s^q}, \boldsymbol{B})$
```

Lemma 6 *Let $\hat{\boldsymbol{B}}_s(D) = \boldsymbol{B}_s(D) + \boldsymbol{b}_s$, where $\boldsymbol{b}_s \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^d)$ for $\sigma^2 = \frac{R^2}{\epsilon^2} \log^2 T \log \frac{\log T}{\delta}$. Then, for any t and any $\Theta_s \in \mathbb{R}^d$,*

$$\text{pdf}[\hat{\boldsymbol{B}}_s(D) = \Theta_s] \le e^{\frac{\epsilon}{\log T}} \text{pdf}[\hat{\boldsymbol{B}}_s(D') = \Theta_s] + \frac{\delta}{\log T}$$

where D and D' are two datasets differing in exactly one entry.

Proof Let $\Delta = \boldsymbol{B}_s(D) - \boldsymbol{B}_s(D')$. Note that $\|\Delta\|_2 \le R$. Now, consider the following ratio:

$$\begin{aligned} \frac{\text{pdf}[\hat{\boldsymbol{B}}_s(D) = \Theta_s]}{\text{pdf}[\hat{\boldsymbol{B}}_s(D') = \Theta_s]} &= \frac{\exp \frac{||\Theta_s - \boldsymbol{B}_s(D)||_2^2}{2\sigma^2}}{\exp \frac{||\Theta_s - \boldsymbol{B}_s(D')||_2^2}{2\sigma^2}} = \exp \frac{||\Delta||_2^2 - 2\Delta^T(\boldsymbol{B}_s(D') - \Theta_s)}{2\sigma^2}, \\ &\le \exp \frac{R^2 + 2|\Delta^T(\boldsymbol{B}_s(D') - \Theta_s)|}{2\sigma^2}. \end{aligned} \tag{36}$$

Now, $\Delta^T(\boldsymbol{B}_s(D') - \Theta_s)$ follows $\mathcal{N}(0, ||\Delta||_2^2 \sigma^2)$. For a random variable $V \sim \mathcal{N}(0,1)$, and for all $\gamma > 1$, $\text{pdf}[|V| > \gamma] \le e^{-\gamma^2/2}$ (Mill's inequality). Thus,

$$\text{pdf}[|\Delta^T(\boldsymbol{B}_s(D') - \Theta_s)| \ge R\sigma\gamma] \le \text{pdf}[|\Delta^T(\boldsymbol{B}_s(D') - \Theta_s)| \ge ||\Delta||_2 \sigma\gamma] \le \exp(\frac{-\gamma^2}{2})$$

Lemma follows by setting $\gamma = 2\sqrt{\ln \frac{\log T}{\delta}}$ in the equation above and combining it with (36). ■

Next, we provide formal privacy and utility guarantees for Algorithm 5. Our proof is inspired by a technique developed by Dwork et al. (2010a).

Theorem 11 (Algorithm 5: Privacy and Utility) *Let $D = \langle \boldsymbol{w}_1, \cdots, \boldsymbol{w}_T \rangle$ be a dataset of vectors with $\boldsymbol{w}_t \in \mathbb{R}^d$ being provided online at each time step t. Let $R = \max_{i \leq T} ||\boldsymbol{w}_i||_2$ and $\sigma^2 = \frac{R^2}{\epsilon^2} \log^2 T \log \frac{\log T}{\delta}$. Let $W_t = \sum_{\tau=1}^{t} \boldsymbol{w}_\tau$ be the partial sum of the entries in the dataset D till the t-th entry. Then, $\forall t \in [T]$, following are true for the output of Algorithm 5 with parameters $(t, \epsilon, \delta, R, T)$.*

- ***Privacy:*** *The output $\hat{W}_t$ is (ϵ, δ)-differentially private.*
- ***Utility:*** *The output $\hat{W}_t$ has the following distribution: $\hat{W}_t \sim \mathcal{N}(W_t, k\sigma^2 \mathbb{I}_d)$, where $k \leq \lceil \log T \rceil$.*

Proof **Utility**: Note that Line 14 of the Algorithm 5 adds at most $\lceil \log T \rceil$ vectors $\hat{\boldsymbol{B}}_s$ (corresponding to the chosen nodes of the binary tree $\boldsymbol{B}$). Now each of the selected vectors $\hat{\boldsymbol{B}}_s$ is generated by adding a noise $\boldsymbol{b}_s \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^d)$. Furthermore, each $\boldsymbol{b}_s$ is generated independent of other noise vectors. Hence, the total noise in the output partial sum $\hat{W}_t$ has the following distribution: $\mathcal{N}(0, k\sigma^2 \mathbb{I}_d)$, where $k \leq \lceil \log T \rceil$.

Privacy: First, we prove that $\hat{\boldsymbol{B}}(D)$ is (ϵ, δ)-differentially private. As defined above, let D and D' be the two datasets (sequences of input vectors) that differ in exactly one entry. Let $S \subset \mathbb{R}^{2T-1}$. Now,

$$\frac{\Pr[\hat{\boldsymbol{B}}(D) \in S]}{\Pr[\hat{\boldsymbol{B}}(D') \in S]} = \frac{\int_{\Theta \in S} \mathrm{pdf}[\hat{\boldsymbol{B}}(D) = \Theta]}{\int_{\Theta \in S} \mathrm{pdf}[\hat{\boldsymbol{B}}(D') = \Theta]}.$$

Note that noise $(\boldsymbol{b}_s)$ at each node s is generated independently of all the other nodes. Hence,

$$\frac{\mathrm{pdf}[\hat{\boldsymbol{B}}(D) = \Theta]}{\mathrm{pdf}[\hat{\boldsymbol{B}}(D') = \Theta]} = \frac{\Pi_s \, \mathrm{pdf}[\hat{\boldsymbol{B}}_s(D) = \Theta_s]}{\Pi_s \, \mathrm{pdf}[\hat{\boldsymbol{B}}_s(D') = \Theta_s]}.$$

Since D and D' differ in exactly one entry, $B(D)$ and $B(D')$ can differ in at most $\log T$ nodes. Thus at most $\log T$ ratios in the above product can be different from one. Now, by using Lemma 6 to bound each of these ratios and then using composability argument Dwork et al. (2006b); Dwork and Lei (2009) over the $\log T$ nodes which have differing values in $\boldsymbol{B}(D)$ and $\boldsymbol{B}(D')$,

$$\Pr[\hat{\boldsymbol{B}}(D) = \Theta] \leq e^{\epsilon} \Pr[\hat{\boldsymbol{B}}(D') \in \Theta] + \delta,$$

i.e., $\hat{\boldsymbol{B}}(D)$ is (ϵ, δ)-differentially private.

Now, each partial sum is just a deterministic function of $\hat{\boldsymbol{B}}(D)$. Hence, (ϵ, δ)-differential privacy of each partial sum follows directly by (ϵ, δ)-differential privacy of $\hat{\boldsymbol{B}}(D)$. ■

Appendix E. Offline Learning

Algorithm 6 Private Offline Learning (POL)

```
Input: Input dataset $D = \langle \boldsymbol{z}_1, \cdots, \boldsymbol{z}_T \rangle$ and the convex set $\mathcal{C}$
Parameter: Privacy parameters $(\epsilon_p, \delta)$, generalization error parameter $\epsilon_g$, Lipschitz bound $L$ on the loss function $\ell$, bound on $\|\boldsymbol{x}^*\|_2$
If $\mathcal{C} = \mathcal{R}^d$ then set $\mathcal{C} = \{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{R}^d, \|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}^*\|_2\}$.
Choose $\boldsymbol{x}_1$ randomly from $\mathcal{C}$
Set $\alpha \leftarrow \frac{\epsilon_g}{\|\boldsymbol{x}^*\|_2^2}$
Initialize $\boldsymbol{s} = \boldsymbol{x}_1$
for $t = 1$ to $T - 1$ do
    Learning rate: $\eta_t = \frac{1}{\alpha t}$
    IGD Update: $\boldsymbol{x}_{t+1} \leftarrow \text{argmin}_{\boldsymbol{x} \in \mathcal{C}} \left(\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}_t\|_2^2 + \eta_t(\ell(\boldsymbol{x}; \boldsymbol{z}_t) + \frac{\alpha}{2}\|\boldsymbol{x}\|_2^2)\right)$
    Store sum: $\boldsymbol{s} \leftarrow \boldsymbol{s} + \boldsymbol{x}_{t+1}$
end for
Average: $\tilde{\boldsymbol{x}} \leftarrow \frac{\boldsymbol{s}}{T}$
Noise Addition: $\bar{\boldsymbol{x}} \leftarrow \tilde{\boldsymbol{x}} + \boldsymbol{b}$, where $\boldsymbol{b} \sim \mathcal{N}(0^d, \beta^2 \mathbb{I}^d)$ and $\beta = \frac{2\sqrt{2}(L+\alpha\|\boldsymbol{x}^*\|_2)\ln T}{T\epsilon_p}\sqrt{\ln\frac{1}{\delta} + \epsilon_p}$
Output $\hat{\boldsymbol{x}} = \text{argmin}_{\boldsymbol{x} \in \mathcal{C}} \left(\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_2^2\right)$
```

E.1. Proof of Theorem 5

Proof Recall that to prove differential privacy, one needs to show that changing one training point from the dataset D doesn't lead to significant change in the algorithm's output $\hat{\boldsymbol{x}}$ which is a perturbation of $\tilde{\boldsymbol{x}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{x}_t$. Hence, we need to show that the L_2-sensitivity (see Definition 3) of $\tilde{\boldsymbol{x}}$ is low.

Now let $\boldsymbol{x}'_1, \cdots, \boldsymbol{x}'_T$ be the sequence of outputs produced by the IGD algorithm used in Algorithm 6 when executed on a dataset D' which differs in exactly one entry from D. To estimate the sensitivity of $\tilde{\boldsymbol{x}}$, we need to bound $||\frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{x}_t - \boldsymbol{x}'_t)||_2$. Now, using triangle inequality and Lemma 4, we get:

$$||\frac{1}{T}\sum_{t=1}^{T}(\boldsymbol{x}_t - \boldsymbol{x}'_t)||_2 \leq \frac{1}{T}\sum_{t=1}^{T}\|\boldsymbol{x}_t - \boldsymbol{x}'_t\|_2 \leq \frac{1}{T}\sum_{t=2}^{T}\frac{2L'}{t-1} \leq \frac{2L'\ln T}{T}, \tag{37}$$

where L' is the maximum Lipschitz continuity coefficient of $\ell(\boldsymbol{x}, \boldsymbol{z}_t) + \frac{\alpha}{2}\|\boldsymbol{x}\|_2^2, \forall t$ over the set $\mathcal{C}$. Using the fact that $\|\mathcal{C}\|_2 = \|\boldsymbol{x}^*\|_2$, we obtain $L' = L + \alpha\|\boldsymbol{x}^*\|_2$.

The theorem now follows using L_2-sensitivity of $\tilde{\boldsymbol{x}}$ (see (37)) and an argument similar to that of the proof for Lemma 1. ■

E.2. Proof of Theorem 6

Before proving the utility guarantee, we first rewrite the approximation error incurred by $\tilde{\boldsymbol{x}} = \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{x}_t$, as derived by Kakade and Tewari (2008).

Theorem 12 (Approximation Error in Risk Minimization (Eq. 6) Kakade and Tewari (2008)) *Let $\mathcal{R}_{\mathcal{A}}(T)$ be the regret for the online algorithm $\mathcal{A}$. Then with probability at least $1-\gamma$,*

$$\mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\tilde{\boldsymbol{x}};z)] - \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\boldsymbol{x}^*;z)] \le \frac{\alpha}{2}\|\boldsymbol{x}^*\|^2 + \frac{\mathcal{R}_{\mathcal{A}}(T)}{T} + \frac{4}{T}\sqrt{\frac{L'^2\mathcal{R}_{\mathcal{A}}(T)\ln(\frac{4\ln T}{\gamma})}{\alpha}} + \frac{\max\{\frac{16L'^2}{\alpha},6\}\ln(\frac{4\ln T}{\gamma})}{T}$$

where $L' = L + \alpha\|\boldsymbol{x}^*\|_2$, L is the Lipschitz continuity bound on the loss function ℓ and α is the strong convexity parameter of the function sequence F.

With this result in place, we now proceed to the proof for Theorem 6.

Proof To prove the result, we upper bound $\mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\hat{\boldsymbol{x}};z)] - \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\boldsymbol{x}^*;z)]$ as:

$$\begin{aligned}\mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\hat{\boldsymbol{x}};z)] - \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\boldsymbol{x}^*;z)] &= \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\hat{\boldsymbol{x}};z)] - \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\tilde{\boldsymbol{x}};z)] \\ &\quad + \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\tilde{\boldsymbol{x}};z)] - \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\boldsymbol{x}^*;z)], \\ &\le L||\hat{\boldsymbol{x}} - \tilde{\boldsymbol{x}}||_2 + \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\tilde{\boldsymbol{x}};z) - \ell(\boldsymbol{x}^*;z)], \\ &= L||\boldsymbol{b}||_2 + \mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\tilde{\boldsymbol{x}};z) - \ell(\boldsymbol{x}^*;z)], \end{aligned} \tag{38}$$

where the second inequality follows using Lipschitz continuity of ℓ and the last equality follows by the noise addition step (Step 13) of Algorithm 6.

From the tail bound on the norm of a Gaussian random vector, it follows that with probability at least $1-\frac{\gamma}{2}$,

$$||\boldsymbol{b}||_2 \le 3\sqrt{d}\beta\sqrt{\ln\frac{1}{\gamma}} \le 12\sqrt{d}L'\frac{\ln T}{T\epsilon_p}\sqrt{\ln\frac{1}{\gamma}\ln\frac{1}{\delta}}, \tag{39}$$

where $L' = L + \epsilon_g/\|\boldsymbol{x}^*\|_2$, L is the Lipschitz continuity parameter of ℓ. Note that in Line 5 of Algorithm 6 we set the strong convexity parameter $\alpha = \frac{\epsilon_g}{||\boldsymbol{x}^*||_2^2}$.

Now, regret bound of IGD is given by:

$$R_{\mathsf{IGD}}(T) = O(||\boldsymbol{x}^*||_2 + \frac{L'^2}{\alpha}\ln T), \tag{40}$$

Thus, by combining (38), (39), (40), and Theorem 12, with probability at least $1-\gamma$,

$$\mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\hat{\boldsymbol{x}};z)] - \min_{\boldsymbol{x}\in\mathcal{C}}\mathbb{E}_{z\sim\mathcal{D}_{\mathcal{Z}}}[\ell(\boldsymbol{x};z)] \le \frac{\epsilon_g}{2} + C\frac{\sqrt{d}L(L+\frac{\epsilon_g}{\|\boldsymbol{x}^*\|_2})\ln T\sqrt{\ln\frac{1}{\gamma}\ln\frac{1}{\delta}}}{\epsilon_p T} + C\frac{(L+\frac{\epsilon_g}{\|\boldsymbol{x}^*\|_2})^2\|\boldsymbol{x}^*\|_2^2\ln T\ln\frac{\ln T}{\gamma}}{\epsilon_g T},$$

where $C > 0$ is a global constant.

The result now follows by bounding the RHS above by ϵ_g. ■

Algorithm 7 Private OCP Algorithm with weak dependence on T

```
Input: OCP algorithm $\mathcal{A}$, cost function sequence $F = \langle f_1, \cdots, f_T \rangle$ and the convex set $\mathcal{C}$
Parameter: privacy parameters $(\epsilon, \delta)$
Choose $\boldsymbol{x}_1$ and $\hat{\boldsymbol{x}}_1$ randomly from $\mathcal{C}$
for $t = 1$ to $T - 1$ do
    Cost: $L_t(\hat{\boldsymbol{x}}_t) = f_t(\hat{\boldsymbol{x}}_t)$
    OCP Update: $\boldsymbol{x}_{t+1} \leftarrow \mathcal{A}(\langle f_1, \ldots, f_t \rangle, \langle \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t \rangle, \mathcal{C})$
    Noise Addition: $\tilde{\boldsymbol{x}}_{t+1} \leftarrow \boldsymbol{x}_{t+1} + \boldsymbol{b}_{t+1}$, $\boldsymbol{b}_{t+1} \sim \mathcal{N}(0^d, \frac{\beta^2}{t^2}\mathbb{I}^d)$, where $\beta = \lambda_{\mathcal{A}}\sqrt{\frac{2^{\lfloor \log_2 t \rfloor + 1}}{\epsilon}\left(\ln \frac{2^{\lfloor \log_2 t \rfloor}}{\delta} + \sqrt{\frac{\epsilon}{2^{\lfloor \log_2 t \rfloor}}}\right)}$
    Output $\hat{\boldsymbol{x}}_{t+1} = \mathrm{argmin}_{\boldsymbol{x} \in \mathcal{C}} \left(\|\boldsymbol{x} - \tilde{\boldsymbol{x}}_{t+1}\|_2^2\right)$
end for
```

Appendix F. POCP with Weak Dependence on T

In this section, we propose our modified POCP method with weaker dependence on T. As mentioned in Section 6.2, we use a doubling trick to reduce dependence on T. At each step t, we add noise with variance $\beta = \lambda_{\mathcal{A}}\sqrt{\frac{2^{\lfloor \log_2 t \rfloor + 1}}{\epsilon}\left(\ln \frac{2^{\lfloor \log_2 t \rfloor}}{\delta} + \sqrt{\frac{\epsilon}{2^{\lfloor \log_2 t \rfloor}}}\right)}$. This noise is enough to guarantee $(\sqrt{\frac{\epsilon}{2^{\lfloor \log_2 t \rfloor}}}, \frac{\delta}{2^{\lfloor \log_2 t \rfloor}})$-differential privacy for each iterate $\boldsymbol{x}_{t+1}$. That is in the first step, we guarantee $(\sqrt{\epsilon}, \delta)$ differential privacy and in the next two steps, $(\sqrt{\epsilon}/2, \delta/2)$ differential privacy, and so on. Now, using arguments similar to Theorem 1, we can obtain privacy guarantee for our modified OCP algorithm (see Algorithm 7) as well. We formalize our privacy guarantee for Algorithm 7 in the theorem below.

Theorem 13 *Algorithm 7 is $(3\epsilon \log_2 T, \delta \log_2 T + \frac{2}{T^{2\epsilon}})$-differentially private.*

Proof The proof follows the general outline of the proof of Theorem 1. However, the major difference in this case is that the output in t-th iteration is guaranteed to be $(\sqrt{\frac{\epsilon}{2^{\lfloor \log_2 t \rfloor}}}, \frac{\delta}{2^{\lfloor \log_2 t \rfloor}})$-differentially private. Proceeding as in the proof of Theorem 1, define $\mathcal{G}_{t+1}$ as:

$$\boldsymbol{b} \in \mathcal{G}_{t+1} \text{ iff } \left|\boldsymbol{b}^T \Delta \boldsymbol{x}_{t+1}\right| \leq \frac{\beta \lambda_{\mathcal{A}}}{t^2}\sqrt{2 \ln \frac{2^{\lfloor \log_2 t \rfloor}}{\delta}}, \tag{41}$$

where $\Delta \boldsymbol{x}_{t+1} = \boldsymbol{x}_{t+1} - \boldsymbol{x}'_{t+1}$ for $1 \leq t \leq T$. Just as in the proof of Theorem 1, for each time step $1 \leq t \leq T$,

$$\Pr[\boldsymbol{b}_{t+1} \notin \mathcal{G}_{t+1}] \leq \frac{\delta}{2^{\lfloor \log_2 t \rfloor}}. \tag{42}$$

Now, by a union bound, the probability that the output vectors $\boldsymbol{b}_{t+1} \in \mathcal{G}_t$ for every $1 \leq t \leq T$ is at least $1 - \sum_{t=1}^{T} \frac{\delta}{2^{\lfloor \log_2 t \rfloor}} = 1 - \delta \log_2 T$. Now, define the sequence of functions with ξ being any event in the event space,

$$\mathbf{Z}_{t+1}(\mathbf{a}_{t+1}; \xi) = \ln\left(\frac{\mathrm{pdf}[\tilde{\boldsymbol{x}}_{t+1} = \mathbf{a}_{t+1} \mid \xi]}{\mathrm{pdf}[\tilde{\boldsymbol{x}}'_{t+1} = \mathbf{a}_{t+1} \mid \xi]}\right),$$

where $\mathbf{a}_{t+1} \in \mathbb{R}^d$. Recall that $\tilde{\boldsymbol{x}}_{t+1} = \boldsymbol{x}_{t+1} + \boldsymbol{b}_{t+1}$ and $\tilde{\boldsymbol{x}}'_{t+1} = \boldsymbol{x}'_{t+1} + \boldsymbol{b}_{t+1}$. Hence, the pdfs in the above equation are associated with the random choice of the noise vectors $\boldsymbol{b}_{t+1}$ which is drawn from a multivariate Gaussian.

Using Lemma 1, we have that at each time step t, the output $\tilde{\boldsymbol{x}}_{t+1}$ of Algorithm 7 is $(\frac{\sqrt{\epsilon}}{T^{0.5+c}}, \frac{\delta}{T})$-differentially private. That is, for $1 \leq t \leq T$,

$$-\sqrt{\frac{\epsilon}{2^{\lfloor \log_2 t \rfloor}}} \leq \mathbf{Z}_{t+1}(\mathbf{a}_{t+1}; \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}) = \ln\left(\frac{\mathrm{pdf}[\tilde{\boldsymbol{x}}_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}]}{\mathrm{pdf}[\tilde{\boldsymbol{x}}'_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}]}\right) \leq \sqrt{\frac{\epsilon}{2^{\lfloor \log_2 t \rfloor}}}.$$

Let $L(\tilde{\boldsymbol{x}}_2, \cdots, \tilde{\boldsymbol{x}}_{T+1}; \xi) = \sum_{t=1}^{T} \mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1})$. Since each $\boldsymbol{b}_{t+1}$ is sampled independently and the randomness in both $\tilde{\boldsymbol{x}}_{t+1}$ and $\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1})$ is only due to $\boldsymbol{b}_{t+1}$, therefore: i) for $1 \leq t \leq T$, $\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1}) = \mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1})$, and ii)each entry in the sequence $\tilde{\boldsymbol{x}}_{t+1}$s for $1 \leq t \leq T$ and each entry in the sequence $\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1})$s for $1 \leq t \leq T$ are independent. Now, using independence of $\tilde{\boldsymbol{x}}_{t+1}$'s and the bound given above,

$$\begin{aligned}\mathbb{E}_{\tilde{\boldsymbol{x}}_2, \cdots, \tilde{\boldsymbol{x}}_{T+1}}[L(\tilde{\boldsymbol{x}}_2, \cdots, \tilde{\boldsymbol{x}}_{T+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1})] &= \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{b}_{t+1}}[\mathbf{Z}_{t+1}(\tilde{\boldsymbol{x}}_{t+1}; \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1})] \\ &\leq \sum_{t=1}^{T} \frac{2\epsilon}{2^{\lfloor \log_2 t \rfloor}} \leq 2\epsilon \log_2 T.\end{aligned}$$

Note also that for every $1 \leq t \leq T$ and $\mathbf{a}_{t+1} \in \mathbb{R}^d$, $|\mathbf{Z}_{t+1}(\mathbf{a}_{t+1}; \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1})| \leq \sqrt{\frac{\epsilon}{2^{\lfloor \log_2 t \rfloor}}}$. By the Azuma-Hoeffding inequality,

$$\Pr[L(\tilde{\boldsymbol{x}}_2, \cdots, \tilde{\boldsymbol{x}}_{T+1}; \boldsymbol{b}_2 \in \mathcal{G}_2 \cdots, \boldsymbol{b}_{T+1} \in \mathcal{G}_{T+1}) \geq 3\epsilon \log_2 T] \leq 2\exp\left(-\frac{2\epsilon^2 \log_2^2 T}{\epsilon \log_2 T}\right) \leq 2\exp(-2\epsilon \log_2 T).$$

Hence, with probability at least $1 - 2T^{-2\epsilon}$ over the draws of $\mathbf{a}_{t+1} \in \mathbb{R}^d$ from $\tilde{\boldsymbol{x}}_{t+1}$, we have:

$$\Pi_{t=1}^{T} \mathrm{pdf}(\tilde{\boldsymbol{x}}_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}) \leq e^{3\epsilon} \Pi_{t=1}^{T} \mathrm{pdf}(\tilde{\boldsymbol{x}}'_{t+1} = \mathbf{a}_{t+1} \mid \boldsymbol{b}_{t+1} \in \mathcal{G}_{t+1}).$$

Now, $\Pr[\exists t \text{ s.t. } \boldsymbol{b}_{t+1} \notin \mathcal{G}_{t+1}] \leq \delta \log_2 T$. Thus, again using a union bound, we have the required result. ■

Regret of the modified POCP algorithm: Note that, in the modified POCP algorithm, the variance of the noise added to $\boldsymbol{x}_{t+1}$, at time-step t, is always less than the variance of the corresponding noise added in the POCP algorithm. Here, the regret incurred by our modified POCP algorithm (see Algorithm 7) is bounded by the regret incurred by the original POCP algorithm.

Appendix G. Empirical Results

In this section we study the privacy and utility (regret) trade-offs for two of our private OCP approaches under different practical settings. Specifically, we consider the practically important problem of online linear regression and online logistic regression. For online linear regression we apply our PQFTL approach (see Algorithm 4) and for online logistic regression we apply our PIGD method (see Algorithm 2). For both the problems, we compare our method against the offline optimal and the non-private online version and show the regret/accuracy trade-off with privacy parameters. We show that our methods learn a meaningful hypothesis (a hyperplane for both the problems) while privacy of the data is provably preserved due to our differential privacy guarantees.

G.1. Online Linear Regression (OLR)

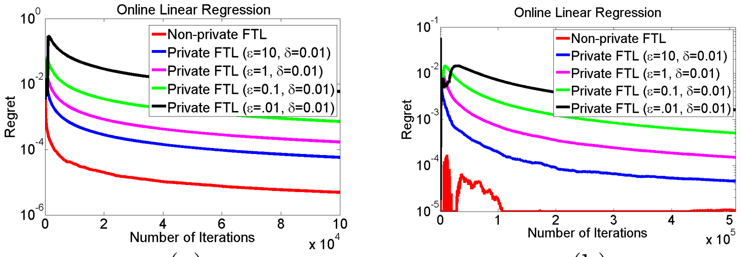
Online linear regression (OLR) requires solving for $\boldsymbol{x}_t$ at each step so that squared error in the prediction is minimized. Specifically, we need to find $\boldsymbol{x}_t$ in an online fashion such that $\sum_t (y_t - g_t^T \boldsymbol{x}_t)^2 + \alpha \|\boldsymbol{x}_t\|^2$ is minimized. OLR is a practically important learning problem and have a variety of practical applications in domains such as finance (Kivinen and Warmuth, 1995).

Now, note that we can directly apply our PQFTL approach (see Section 4) to this problem to obtain differentially private iterates $\boldsymbol{x}_t$ with the regret guaranteed to be logarithmic. Here, we apply our PQFTL algorithm for the OLR problem on a synthetic dataset as well as a benchmark real-world dataset, namely "Year Prediction" (Frank and Asuncion, 2010). For the synthetic dataset, we fix $\boldsymbol{x}^*$, generate data points g_t of dimensionality $d = 10$ by sampling a multivariate Gaussian distribution and obtain the target $y_t = g_t^T \boldsymbol{x}^* + \eta$, where η is random Gaussian noise with standard variance 0.01. We generate $T = 100,000$ such input points and targets. The Year Prediction dataset is 90-dimensional and contains around $500,000$ data points. For both the datasets, we set $\alpha = 1$ and at each step apply our PQFTL algorithm. We measure the optimal offline solution using standard ridge regression and also compute regret obtained by the non-private FTL algorithm.

Figure 2 (a) and (b) shows the average regret (i.e., regret normalized by the number of entries T) incurred by PQFTL for different privacy level ϵ on synthetic and Year Prediction data. Note that the y-axis is on the log-scale. Clearly, our PQFTL algorithm obtains low-regret even for reasonable high privacy levels ($\epsilon = 0.01$). Furthermore, the regret gets closer to the regret obtained by the non-private algorithm as privacy requirements are made weaker.

G.2. Online Logistic Regression

Online logistic regression is a variant of the online linear regression where the cost function is logistic loss rather than squared error. Logistic regression is a popular method to learn classifiers, and has been shown to be successful for many practical problems. In this experiment, we apply our private IGDalgorithm to the online logistic regression problem. To this end, we use the standard Forest cover-type dataset, a dataset with two classes, 54-dimensional feature vectors and $581,012$ data points. We select 10% data points for testing purpose and run our Private IGD algorithm on the remaining data points. Figure 2 (c) shows classification accuracy (averaged over 10 runs) obtained by IGD and our PIGD algorithm for different privacy levels. Clearly, our algorithm is able to learn a reasonable classifier from the dataset in a private manner. Note that our regret bound for PIGD method is $O(\sqrt{T})$, hence, it would require more data points to reduce regret to very small values, which is reflected by a drop in classification accuracy as ϵ decreases.



(a) (b)

Method	Accuracy
Non-private IGD	68.1%
PIGD ($\epsilon = 20, \delta = 0.01$)	66.3%
PIGD ($\epsilon = 10, \delta = 0.01$)	62.7%
PIGD ($\epsilon = 1, \delta = 0.01$)	59.4%
PIGD ($\epsilon = 0.1, \delta = 0.01$)	58.3%

(c)

Figure 2: Privacy vs Regret. **(a), (b)**: Average regret (normalized by the number of iterations) incurred by FTL and PQFTL with different levels of privacy ϵ on the synthetic 10-dimensional data and Year Prediction Data. Note that the regret is plotted on a log-scale. PQFTL obtained regret of the order of $1e - 2$ even with high privacy level of $\epsilon = 0.01$. **(c)**: Classification accuracy obtained by IGD and PIGD algorithm on Forest-covertype dataset. PIGD learns a meaningful classifier while providing privacy guarantees, especially for low privacy levels, i.e., high ϵ.