The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Uplink Communication Efficient Differentially Private Sparse Optimization with Feature-Wise Distributed Data

Jian Lou, Yiu-ming Cheung*
Department of Computer Science, Hong Kong Baptist University,
Kowloon Tong, Hong Kong SAR, China
{jianlou, ymc}@comp.hkbu.edu.hk

Abstract

Preserving differential privacy during empirical risk minimization model training has been extensively studied under centralized and sample-wise distributed dataset settings. This paper considers a nearly unexplored context with features partitioned among different parties under privacy restriction. Motivated by the nearly optimal utility guarantee achieved by centralized private Frank-Wolfe algorithm (Talwar, Thakurta, and Zhang 2015), we develop a distributed variant with guaranteed privacy, utility and uplink communication complexity. To obtain these guarantees, we provide a much generalized convergence analysis for block-coordinate Frank-Wolfe under *arbitrary sampling*, which greatly extends known convergence results that are only applicable to two specific block sampling distributions. We also design an active feature sharing scheme by utilizing private Johnson-Lindenstrauss transform, which is the key to updating local partial gradients in a differentially private and communication efficient manner.

Introduction

Empirical risk minimization (ERM) is a fundamental tool for learning useful models from data that are collected from individuals, e.g. see (Cheung 2005; Cheung and Zeng 2009; Wang et al. 2017). To avoid breaching the privacy of the individuals, privacy protection mechanisms have been considered to ensure that the adversary cannot infer any individual data from the output of the learning process. Beginning with the seminal work (Chaudhuri, Monteleoni, and Sarwate 2011), which considers private ERM training under the formal statistical differential privacy notion (Dwork, Roth, and others 2014), various differentially private optimization algorithms have been developed for training the model with centralized datasets (Smith 2011; Kifer, Smith, and Thakurta 2012; Bassily, Smith, and Thakurta 2014; Talwar, Thakurta, and Zhang 2014; Jain and Thakurta 2013; Kasiviswanathan and Jin 2016) and sample-wise distributed datasets (Huang, Mitra, and Vaidya 2015; Nozari, Tallapragada, and Cortés 2016; Han, Topcu, and Pappas 2017).

Deviating from the above mentioned centralized and sample-wise distributed settings, we consider the feature-wise distributed dataset setting. Such setting appears in many real applications, where the information describing an individual is collected and held by different parties which can be different sets of sensory systems or different organizations. For example, a person's medical records are sensitive personal information that can be held by several clinics. Although privacy issue has been considered for these vertically-partitioned datasets (Yunhong, Liang, and Guoping 2009; Yu, Vaidya, and Jiang 2006; Mangasarian, Wild, and Fung 2008), it has rarely been studied with the more restrict differential privacy notion. It would be ideal to make use of all attributes kept by different parties in a distributed fashion, while still ensuring differential privacy.

During distributed training, information sourced from user nodes to the server node, referred to as uplink communication, are computed based on sensitive individual information. On the other hand, the information broadcasted by the server back to users (referred to as downlink communication), by the post-processing property of the differential privacy (Dwork, Roth, and others 2014), will maintain differential privacy even without any privacy protection mechanism as long as the uplink communication is private. As such, designing privacy mechanism to prevent adversary from inferring sensitive individual information by spying on the uplink communication is the key to private training for distributed dataset. Intuitively, minimizing the uplink communication complexity means less exposure of sensitive data and reduced potential of personal data leakage. Thus, less uplink communication would generally require less privacy protection budget for ensuring differential privacy.

To be specific, we consider the constrained ERM model in this paper as $\min_{\boldsymbol{x}\in\mathcal{M}} \sum_{i=1}^{n} f(\boldsymbol{x}; \mathbb{D}_i)$, where $\mathbb{D}_i$ represents the i-th sample of the $n \times d$ dataset $\mathbb{D}$, where n is the sample size and d is the feature dimension. With the feature-wise distributed setting, the $n \times d$ data matrix is partitioned vertically with each disjoint partition held by one of the K user nodes. $f(\boldsymbol{x})$ is a smooth convex loss function. We assume the convex compact constraint set $\mathcal{M}$ is coordinate separable as $\mathcal{M}_1 \times, ..., \times \mathcal{M}_d$. In particular, we elaborate the LASSO problem, where $f(\boldsymbol{x})$ is the least square loss and $\mathcal{M}$ is the ℓ_1-norm ball. For this problem, (Talwar, Thakurta, and Zhang 2015) proposes a centralized private Frank-Wolfe algorithm (FW) (Jaggi 2013) by adapting the Report-Noisy-Max mechanism (Dwork, Roth, and others 2014) for ensuring differential privacy. In particular, they prove that the algorithm has nearly

*corresponding author

optimal utility guarantee for the private LASSO task. That is, no private algorithm has better utility up to a $\log(n)$ factor.

With the above in mind, we propose a distributed private FW algorithm for solving ERM in an uplink communication efficient way to obtain the same nearly optimal utility as in the centralized setting. At first glance, it seems straightforward by integrating the Report-Noisy-Max for privacy protection (Talwar, Thakurta, and Zhang 2015) with the existing distributed FW design. However, such direct combinations either 1) does not have known differentially private strategy for communicating active features which are indispensable for computing local partial gradient under feature-wise distributed setting (Bellet et al. 2015); or 2) requiring each user node to have full replication of the entire feature set, which is undesired as local features need to be communicated with an extra preprocessing step (Wang et al. 2016). The seemingly different aspects of incapabilities actually attribute to the same reason: existing randomized block-coordinate FW (BCFW) algorithms have limited convergence guarantees that are only applicable under two simple block sampling distributions. In this regard, this paper makes primarily two contributions:

1. BCFW under arbitrary sampling: we develop a much general convergence analysis for BCFW under arbitrary sampling (BCFW-AS). It enjoys greater flexibility than existing analysis (Lacoste-julien et al. 2013; Wang et al. 2016) whose analysis is highly dependent on the specific samplings. Further, in contrast to existing Expected Separable Overapproximation (ESO) inequality based coordinate descent under arbitrary sampling algorithms (Richtárik and Takáč 2016b; Richtárik and Takác 2016a; Qu and Richtárik 2016a; 2016b), we develop our convergence analysis by introducing a new notion called expected curvature, which is a more fundamental quantity for FW algorithms than ESO-based counterparts.

2. Private FW for feature-wise distributed dataset: We provide an uplink communication efficient private algorithm based on the BCFW-AS. By adopting our general convergence guarantee, it has guaranteed utility which preserves the same nearly optimality for the LASSO task as that in the centralized setting (Talwar, Thakurta, and Zhang 2015). In addition, compared to the existing work (Heinze-Deml, McWilliams, and Meinshausen 2017) under the same setting, our method with $O(n^{\frac{2}{3}})$ has improved the overall uplink communication complexity in comparison with their $O(n)$.

In the next section, after introducing the notation, we provide a thorough discussion of known convergence guarantees for BCFW algorithms, as well as existing private methods for training under feature-wise distributed setting.

Notation and Background

We use $[d]$ to denote the set $\{1, 2, ..., d\}$ and use bold characters $\boldsymbol{A}, \boldsymbol{x}$ for matrices and vectors. $\boldsymbol{A}^\top$ denotes the transpose of matrix $\boldsymbol{A}$. The operator $\circ$ denotes the Hadamard multiplication, i.e. element-wise multiplication of vectors or matrices of the same sizes. The superscript is associated with iteration number, e.g. $\boldsymbol{x}^t$ denotes the decision variable at iteration t, while subscripts is associated with indices of the coordinates or different parties. For a random subset $\mathcal{T}$ of $[d]$, $|\mathcal{T}|$ denotes the cardinality of set $\mathcal{P}$. $\boldsymbol{x}_{(\mathcal{T})} \in \mathcal{M}_\mathcal{T}$ denotes the vector of length $|\mathcal{T}|$ which only keeps the values of $\boldsymbol{x}$ indicated by $\mathcal{T}$ and $\boldsymbol{x}_{[\mathcal{T}]}$ denotes the zero padded one. We use $\nabla f(\boldsymbol{x})$ to denote the gradient of the loss function $f(\boldsymbol{x})$ at $\boldsymbol{x}$ and $\nabla_{(\mathcal{T})} f(\boldsymbol{x})$ is the partial gradient taken with respect to subset of coordinates indexed by $\mathcal{T}$. $||\cdot||_{(\mathcal{M}_i)}$ denotes a primal norm defined on $\mathcal{M}_i$ and $||\cdot||^*_{(\mathcal{M}_i)}$ denotes the associated dual norm. We use $\boldsymbol{e}_i$ to denote the standard basis vector.

Sampling Distributions and Known Convergence Results for Block Coordinate Frank-Wolfe Algorithms

We describe related Frank-Wolfe algorithms by interpreting them as randomized block coordinate Frank-Wolfe under the corresponding sampling distributions to highlight the association of the convergence analysis with the types of distributions. Table 1 summarizes the previous and our new BCFW algorithms, samplings and convergence guarantees. We follow the naming convention used in existing arbitrarily sampled coordinate descent papers (Richtárik and Takáč 2016b; Qu and Richtárik 2016a; 2016b) for referring several common samplings.

Elementary sampling: (Jaggi 2013) is the conventional deterministic FW algorithm using the full gradient per-iteration, which can be seen as sampling the coordinates under elementary sampling with set $[d]$, i.e. sampling set $[d]$ with probability one. With the step size $\gamma^t = \frac{2}{t+2}$, it guarantees $h^t = f(\boldsymbol{x}^t) - f(\boldsymbol{x}^*) \leq \frac{2C_f}{t+2}$, where $x^* \in \mathcal{M}$ denotes an optimum and C_f is the curvature of $f(x)$ on the whole constraint set $\mathcal{M}$, which measures the non-linearity of $f(\boldsymbol{x})$ on the entire constraint set $\mathcal{M}$, reflecting the geometric property of $f(\boldsymbol{x})$ on $\mathcal{M}$:

$$C_f := \sup_{\substack{\boldsymbol{x},\boldsymbol{s}\in\mathcal{M},\\ \gamma\in[0,1]}} \frac{2}{\gamma^2}\Big(f(\boldsymbol{x}+\gamma(\boldsymbol{s}-\boldsymbol{x})) - f(\boldsymbol{x}) - \gamma\langle \boldsymbol{s}-\boldsymbol{x}, \nabla f(\boldsymbol{x})\rangle\Big). \tag{1}$$

The algorithm dFW (Bellet et al. 2015) is a distributed FW method. During one communication pass, each worker evaluates the partial linear oracle based on the local features and then sends both the partial linear oracle index and the associated local duality gap value to the sever node for comparison. Subsequently, the partial linear oracle with the maximum local duality gap is selected and sends back to all workers for the next update. However, the updates of the local partial gradient requires sharing of "active features" at each communication round. It is unknown how to communicate active features in a private and communication efficient way.

Uniform serial sampling: (Lacoste-julien et al. 2013) is a randomized block coordinate Frank-Wolfe (BCFW) method selecting the block to be updated in each iteration according to the uniform serial sampling, i.e. samples one block at each iteration with uniform probability. For analyzing the convergence, (Lacoste-julien et al. 2013) designs the step size $\gamma^t = \frac{2d}{t+2d}$ and introduces the product curvature to obtain $h^t \leq \frac{2d(C_f^\otimes + h^0)}{t+2d}$ primal gap. The product curvature $C_f^\otimes :=$

$\sum_{i=1}^{d} C_f^i$, where C_f^i is the block-wise partial curvature for measuring the non-linearity on individual $\mathcal{M}_i$,

$$C_f^i := \sup_{\substack{\boldsymbol{x},\boldsymbol{s}\in\mathcal{M},\\ \gamma\in[0,1]}} \frac{2}{\gamma^2}\Big(f(\boldsymbol{x}+\gamma(\boldsymbol{s}_{[i]}-\boldsymbol{x}_{[i]})) - f(\boldsymbol{x}) - \gamma\langle \boldsymbol{s}_{(i)}-\boldsymbol{x}_{(i)}, \nabla_{(i)} f(\boldsymbol{x})\rangle\Big). \tag{2}$$

τ-nice sampling: AP-BCFW (Wang et al. 2016) is a parallel and distributed BCFW method, provided that all user nodes have the full replication of the entire dataset. During one communication pass, each worker uniformly samples one block from *all* blocks for updating and the server node summarizes τ non-duplicate updates (discard duplicate update, e.g. two worker samples the same node). Under ideal computational facility, (Wang et al. 2016) analyzes the convergence by equalizing one communication pass as one iteration of centralized BCFW selecting blocks for updating according to τ-nice sampling, i.e. samples τ blocks with uniform probability. It requires yet another set of step size $\gamma^t = \frac{2d\tau}{\tau^2 t+2d}$ and expected set curvature $C_f^\tau := \binom{d}{\tau}^{-1} \sum_{S\subset[d],|S|=\tau} C_f^{(S)}$, where the set curvature $C_f^{(S)}$ is

$$C_f^S := \sup_{\substack{\boldsymbol{x},\boldsymbol{s}\in\mathcal{M},\\ \gamma\in[0,1],|S|=\tau}} \frac{2}{\gamma^2}\Big(f(\boldsymbol{x}+\gamma(\boldsymbol{s}_{[S]}-\boldsymbol{x}_{[S]})) - f(\boldsymbol{x}) - \gamma\langle \boldsymbol{s}_{(S)}-\boldsymbol{x}_{(S)}, \nabla_{(S)} f(\boldsymbol{x})\rangle\Big). \tag{3}$$

AP-BCFW is obviously unsuited to the distributed feature set because it would require copy-and-paste local features to other nodes before computation, which incurs high communication cost and raises privacy concern.

(K,τ)-distributed sampling: This is probability the simplest sampling scheme for the distributed optimization tasks with disjointly divided local blocks, where each user nodes uniformly sample τ blocks from their *local* blocks, which collectively constitutes $K\times\tau$ random block updates from K workers. However, existing random BCFWs do not have convergence guarantee even for this simplest sampling.

Arbitrary sampling: We consider a general BCFW that has guaranteed convergence under arbitrary sampling with two minimal assumptions: 1) the sampling is independent across iterations, i.e. the sampling distribution at the present iteration independent of the sampling of the last iteration; 2) the sampling is proper that any block has nonzero probability to be sampled.

Differentially Private Optimization for Feature-wise Distributed Dataset

The formal definition of differential privacy for a randomized algorithm $\mathcal{ALG}$ with parameter ϵ and δ is as follows.

Definition 1. ((ϵ,δ)-Differential Privacy ((ϵ,δ)-DP)) A randomized algorithm $\mathcal{ALG}$ is (ϵ,δ)-differentially private if, for all neighboring data sets $\mathcal{D}$ and $\mathcal{D}'$, which differ in only one data sample, we have $Pr(\mathcal{ALG}(\mathcal{D})\in\mathcal{O}) \le e^\epsilon Pr(\mathcal{ALG}(\mathcal{D}')\in\mathcal{O}) + \delta$ for all outputs $\mathcal{O}$.

Feature-wise distributed private learning: Feature-wise distributed data is more challenging than sample-wise distributed dataset under privacy restriction. For the latter setting, each user node has enough information to take local update (e.g. user can compute the local gradient based on local data samples) and only the decision variables need to be communicated. However, for feature-wise distributed data, apart from the decision variable, additional information needs to be shared to perform local update (e.g. compute local partial block-wise gradient). In general, more information sent by the user node, more likely sensitive individual privacy is at risk, which makes the privacy protection design more challenging. As a largely unexplored setting, to the best of our knowledge, most recently work in (Heinze-Deml, McWilliams, and Meinshausen 2017) is the only existing one to take into account the same differentially private ERM learning task with disjoint features held by different parties. (Heinze-Deml, McWilliams, and Meinshausen 2017) proposes to add privacy protection during preprocessing by communicating perturbed sketched features (Kenthapadi et al. 2013). Although the uplink communication is one-shot during the preprocessing and its sketching step partially relieves the high communication complexity in terms of the feature dimension d, its complexity is still linearly dependent on the sample size n (i.e. $O(n)$). In comparison, our method only communicates active features indicated by the optimization procedure, featuring a "share-at-need" strategy. As a result, to achieve the nearly optimal utility, the overall uplink complexity of our method is $O(n^{\frac{2}{3}}\log(n^{1/3}))$, which is more uplink communication efficient.

Private Frank-Wolfe algorithm: (Talwar, Thakurta, and Zhang 2015) proposes a centralized private FW algorithm for ERM problem constrained by atomic norm. In each iteration, the FW algorithm greedily selects a linear oracle from the atomic norm set $\mathcal{A}$ (which has a finite number of atomic norm) by picking the one with the largest duality gap. (Talwar, Thakurta, and Zhang 2015) selects the iterative linear oracle by Report-Noisy-Max mechanism (Dwork, Roth, and others 2014) (a special variant of the more general exponential mechanism), which ensures the differential privacy. For the LASSO task, (Talwar, Thakurta, and Zhang 2015) is proved to provide nearly optimal utility guarantee. Since the utility guarantee is based on the convergence analysis, the adaptation of the method to distributed setting is a non-trivial task due to the missing convergence result for BCFW-AS. Furthermore, with features distributed among user nodes, apart from the linear oracle evaluation, the gradient computation also requires additional perturbation for privacy protection, whose effect on utility demands careful quantization and further analysis.

Block-Coordinate Frank-Wolfe under Arbitrary Sampling

Algorithm Description

Algorithm 1 presents the BCFW-AS algorithm under arbitrary proper sampling $\mathcal{S}$. For notational convenience, in this section, $i\in[d]$ can be either single coordinate or block of coordinates that we do not explicitly differentiate them with

additional notation, while in the next section it refers to a single coordinate. Denote the probability for sampling block i by p_i and collectively by $\boldsymbol{p} := \{p_1, ..., p_d\}$. Let p_{min} denote the smallest entry in $\boldsymbol{p}$.

In each iteration (e.g. t), line 2 samples a random set of blocks $\mathcal{T}^t$ from $\{1, 2, ..., d\}$ according to sampling distribution $\mathcal{S}$. To accommodate the injected noise for privacy protection in the next section, and also for wider applicability of BCFW-AS, line 5 allows the partial **l**inear **o**racle (LO) $\hat{\boldsymbol{s}}_{(i)}$ to be evaluated approximately with inexactness parameter ϱ_l^i, given inexact **p**artial **g**radient (PG) $\hat{\nabla}_{(i)} f(\boldsymbol{x})$ with parameter ϱ_g^i in line 4. For brevity, $\hat{\boldsymbol{s}}_{(i)}$ is referred as ϱ_l^i-LO and $\hat{\nabla}_{(i)} f(\boldsymbol{x})$ is referred as ϱ_g^i-PG. Convergence analysis shows that as long as the following assumptions hold, BCFW-AS is still guaranteed to converge:

Assumption 1. *(Inexact linear oracle and inexact gradient) Let γ^t denote the step size at iteration t, $C_f^{\mathbb{E}\mathcal{S}}$ denote expected curvature of function f with sampling $\mathcal{S}$.*

- *Let ϱ_l^i be a constant parameter. The inexact linear oracle $\hat{\boldsymbol{s}}_{(i)}$ satisfies,*

$$\langle \hat{\boldsymbol{s}}_{(i)}, \hat{\nabla}_{(i)} f(\boldsymbol{x}) \rangle \leq \min_{\boldsymbol{s}_{(i)} \in \mathcal{M}_i} \langle \boldsymbol{s}_{(i)}, \hat{\nabla}_{(i)} f(\boldsymbol{x}) \rangle + \frac{\varrho_l^i \gamma^t C_f^{\mathbb{E}\mathcal{S}}}{2}. \tag{4}$$

- *Let ϱ_g^i be the inexact gradient constant parameter, $\nabla_{(i)} f(\boldsymbol{x})$ be the exact partial gradient at $\boldsymbol{x}$. The partial gradient $\hat{\nabla}_{(i)} f(\boldsymbol{x})$ satisfies,*

$$||\hat{\nabla}_{(i)} f(\boldsymbol{x}) - \nabla_{(i)} f(\boldsymbol{x})||^*_{(\mathcal{M}_i)} \leq \frac{\varrho_g^i \gamma^t C_f^{\mathbb{E}\mathcal{S}}}{2}, \tag{5}$$

where $||\cdot||^*_{(\mathcal{M}_i)}$ is the dual norm of the norm associated with $\mathcal{M}_i$.

In the above assumption, the constant $C_f^{\mathbb{E}\mathcal{S}}$ is the expected curvature to be introduced in the next subsection. Step 6 then updates the block i-th decision variable, for $i \in \mathcal{T}^t$. Concisely, we can add up $\hat{\boldsymbol{s}}_i^t$ for all $i \in \mathcal{T}^t$ to denote $\hat{\boldsymbol{s}}_{[\mathcal{T}^t]}^t = \sum_{i \in \mathcal{T}^t} \hat{\boldsymbol{s}}_{[i]}^t$. Then, the total update across all blocks being sampled can be summarized into,

$$\boldsymbol{x}_{[\mathcal{T}^t]}^{t+1} = \boldsymbol{x}^t + \gamma^t (\hat{\boldsymbol{s}}_{[\mathcal{T}^t]}^t - \boldsymbol{x}_{[\mathcal{T}^t]}^t). \tag{6}$$

Expected Curvature

Before moving to the convergence analysis of BCFW-AS, this subsection introduces a new notion called expected curvature, which will play a key role in the convergence of the BCFW-AS in the next subsection. The expected curvature compactly associates the curvature of the loss function on various directions, which are randomly sampled according to the sampling distribution. Intuitively, instead of measuring the largest deviation of the loss function from its linear approximation along some *particular sets of coordinates*, such a new quantity should be able to measure the maximum deviation "averaged" over various choices of sets of coordinates selected under the sampling distribution to manifest the interaction of the intrinsic geometric property along different directions and the distribution of the sampling. We formulate the intuition by the following expected curvature definition:

Algorithm 1 Block-Coordinate Frank-Wolfe Algorithm With Arbitrary Sampling

Require: Initial feasible variable $\boldsymbol{x}^0$, step sequence γ^t, sampling distribution $\mathcal{S}$, inexactness parameters ϱ_g^i and ϱ_l^i, estimation of expected curvature $C_f^{\mathbb{E}\mathcal{S}}$, maximum iteration T;

1: **for** $t = 0, 1, ..., T-1$ **do**
2: Generate a random set $\mathcal{T}^t \subset [d]$, following the distribution $\mathcal{S}$;
3: **for all** $i \in \mathcal{T}^t$ **do**
4: Compute approximate partial gradient $\hat{\nabla}_{(i)} f(\boldsymbol{x}^t)$ satisfies Eq.(5);
5: Compute approximate partial linear oracle $\hat{\boldsymbol{s}}_{(i)}^t$ satisfies Eq.(4);
6: Update $\boldsymbol{x}_{(i)}^{t+1} = \boldsymbol{x}_{(i)}^t + \gamma^t (\hat{\boldsymbol{s}}_{(i)}^t - \boldsymbol{x}_{(i)}^t)$;
7: **end for**
8: **end for**

Ensure: $\boldsymbol{x}^T$;

Definition 2. (Expected curvature) The expected curvature of a the loss function $f(\boldsymbol{x})$ with arbitrary proper sampling $\mathcal{S}$ is defined as,

$$C_f^{\mathbb{E}\mathcal{S}} = \sup_{\substack{\boldsymbol{x}, \boldsymbol{s} \in \mathcal{M}, \\ \gamma \in [0,1]}} \mathbb{E}_{\mathcal{T} \sim \mathcal{S}} \Big[\frac{2}{\gamma^2} f(\boldsymbol{x} + \gamma(\boldsymbol{s}_{[\mathcal{T}]} - \boldsymbol{x}_{[\mathcal{T}]})) - f(\boldsymbol{x}) - \gamma \langle \boldsymbol{s}_{[\mathcal{T}]} - \boldsymbol{x}_{[\mathcal{T}]}, \nabla f(\boldsymbol{x}) \rangle \Big]. \tag{7}$$

It first calculates the average deviation with various combinations of proper $\boldsymbol{x}, \boldsymbol{s}, \gamma$ under the sampling $\mathcal{S}$ and then chooses the largest value calculated from a certain pair of $(\boldsymbol{x}^\#, \boldsymbol{s}^\#, \gamma^\#)$ as the expected curvature. In general, $(\boldsymbol{x}^\#, \boldsymbol{s}^\#, \gamma^\#)$ may not achieve the maximum partial curvature for every possible combination of blocks under the sampling, yet it ensures the overall supremacy after taking into the probability of the appearance of the combinations of blocks. We discuss some properties of the expected curvature by comparing it with: 1) Lipschitz smoothness-based ESO quantities in (Richtárik and Takáč 2016b; Richtárik and Takác 2016a; Qu and Richtárik 2016b); 2) Various curvatures used by existing BCFWs under the samplings mentioned on Page 2 The comparisons show that the expected curvature can be much smaller than Lipschitz-based ESO constants and also refine existing curvature constants used in the specific samplings. For the ease of comparison, we make the same assumption as the compared methods given by

Assumption 2. *There is an $n \times d$ matrix $\boldsymbol{A}$ such that for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{M}$,*

$$f(\boldsymbol{y}) \leq f(\boldsymbol{x}) + \langle \boldsymbol{y} - \boldsymbol{x}, \nabla f(\boldsymbol{x}) \rangle + \frac{1}{2} (\boldsymbol{y} - \boldsymbol{x})^\top \boldsymbol{A}^\top \boldsymbol{A} (\boldsymbol{y} - \boldsymbol{x}). \tag{8}$$

Comparison with ESO quantity: Similar to conventional curvature quantities that have the Lipschitz smoothness constant times the squared diameter of the constraint set as the upper bound, the expected curvature is upper bounded by the ESO quantity times the squared diameter given by

Table 1: Comparison of BCFW convergence results

Sampling	p_{min}	Existing result	Ours
Elementary with set $[d]$	1	$\frac{2(1+\delta)C_f}{t+2}$ (Jaggi 2013; Bellet et al. 2015)	$\frac{2(h_0+(1+\delta)C_f)}{t+2}$
Uniform Serial	$\frac{1}{d}$	$\frac{2d(h^0+(1+\delta)C_f^{\otimes})}{t+2d}$ (Lacoste-julien et al. 2013)	$\frac{2d(h^0+(1+\delta)dC_f^{\mathbb{E}uni\cdot seri})}{t+2d}$
τ-nice	$\frac{\tau}{d}$	$\frac{2d(h^0+(1+\delta)dC_f^{\tau})}{\tau^2 t+2d}$ (Wang et al. 2016)	$\frac{2d(\tau h^0+(1+\delta)dC_f^{\mathbb{E}\tau nice})}{\tau^2 t+2d\tau}$
(K,τ)-distributed	$\frac{(K\tau)^2}{d^2}$	-	$\frac{2d(K\tau h^0+(1+\delta)dC_f^{\mathbb{E}(K,\tau)})}{(\tau K)^2 t+2dK\tau}$
Arbitrary	p_{min}	-	$\frac{2(h^0+\frac{(1+\varrho)}{p_{min}}C_f^{\mathbb{E}\mathcal{S}})}{p_{min}\cdot t+2}$

Proposition 1. *Under Assumption 2, we denote the pairwise probability matrix of arbitrary sampling $\mathcal{S}$ by $\mathbb{P}$, the diameter of $\mathcal{M}_i$ by $D_{\mathcal{M}_i}$. By choosing $\boldsymbol{\beta} = (\beta_1, ...\beta_d)$ with $\beta_i = \min\{\sigma'(\mathbb{P}), \sigma'(\boldsymbol{A}^\top \boldsymbol{A})\}||A_i||_2^2$, where $\sigma'(\mathbb{P}), \sigma'(\boldsymbol{A}^\top \boldsymbol{A})$ are the largest normalized eigenvalues of the matrices $\mathbb{P}$ and $\boldsymbol{A}^\top \boldsymbol{A}$, then we have $C_f^{\mathbb{E}\mathcal{S}} \leq \sum_{i=1}^d p_i \beta_i D_{M_i}^2$.*

In the above proposition, β_i is exactly one of the ESO quantity obtained in (Qu and Richtárik 2016b) under the same assumption. (Qu and Richtárik 2016b) has also provided many other estimations of β_i. We omit those comparisons because we can also show the same result in a similar fashion. The proofs can be found in the supplementary material.

Comparison with curvatures of existing BCFW: The next proposition shows the relationship of the expected curvature with those used in the existing BCFW algorithms.

Proposition 2. *Recall the global curvature C_f of (Jaggi 2013; Bellet et al. 2015) under elementary sampling, the product curvature $C_f^{\otimes}$ of (Lacoste-julien et al. 2013) under uniform serial sampling, and expected set curvature C_f^{τ} of (Wang et al. 2016) under τ-nice sampling as introduced in Subsection . Then, the following relationships hold,*

$$C_f^{\mathbb{E}element} = C_f,\ C_f^{\mathbb{E}uni\cdot seri} \leq \frac{1}{d} C_f^{\otimes},\ C_f^{\mathbb{E}\tau nice} \leq C_f^{\tau}, \tag{9}$$

where all the left hand side terms denote the expected curvature under the corresponding samplings.

By Proposition 2, the expected curvature is always upper bounded by the existing specific curvature constants introduced for specific samplings, which will result in refined convergence results as shown in the next subsection.

Exact form of expected curvature under specific samplings: It is possible to calculate an accurate estimation of the expected curvature under certain samplings if the probability is specified, despite its seemingly abstract definition. We illustrate this with two examples of τ-nice sampling and (K,τ)-distributed sampling. The latter one will be further used in the next section to establish the new convergence of the distributed FW algorithm with disjoint blocks and the utility of the private distributed FW algorithm.

Proposition 3. *Under Assumption 2, the expected curvature under τ-nice sampling satisfies $C_f^{\mathbb{E}\tau nice} \leq \tau\mu_1 + \tau(\tau-1)\mu_2$, where $\mu_1 = \sup_{i\in[d]} ||\boldsymbol{A}_i(\boldsymbol{s}_i - \boldsymbol{x}_i)||_2^2$, $\mu_2 = \sup_{i,j\in[d], i\neq j} (\boldsymbol{A}_i(\boldsymbol{s}_i - \boldsymbol{x}_j))^\top (\boldsymbol{A}_j(\boldsymbol{s}_i - \boldsymbol{x}_j))$.*

This matches the expected set curvature calculated in (Wang et al. 2016), based on which they further provide example curvature quantities for structured SVM and group fused LASSO.

Proposition 4. *Under Assumption 2, the expected curvature under (K,τ)-distributed sampling satisfies*

$$C_f^{\mathbb{E}(K,\tau)} \leq K\tau\mu_1 + K\tau(\tau-1)\mu_2 + K(K-1)\tau^2\mu_3, \tag{10}$$

where $\mu_1 = \sup_{i\in[d]} ||\boldsymbol{A}_i(\boldsymbol{s}_i - \boldsymbol{x}_i)||_2^2$, $\mu_2 = \sup_{i,j\in\mathcal{P}_k, i\neq j} (\boldsymbol{A}_i(\boldsymbol{s}_i - \boldsymbol{x}_j))^\top (\boldsymbol{A}_j(\boldsymbol{s}_i - \boldsymbol{x}_j))$, $\mu_3 = \sup_{i\in\mathcal{P}_{k_1}, j\in\mathcal{P}_{k_2}, k_1\neq k_2} (\boldsymbol{A}_i(\boldsymbol{s}_i - \boldsymbol{x}_j))^\top (\boldsymbol{A}_j(\boldsymbol{s}_i - \boldsymbol{x}_j))$.

We close this subsection by repeating the remarks on curvature in (Jaggi 2013) to stress that curvature is a more fundamental quantity for FW algorithm due to 1) it can be much smaller than Lipschitz smoothness based quantity; 2) it is affine invariant and the analysis applies for arbitrary choices of norms. Thus, for the analysis of BCFW-AS, it is necessary to introduce the new expected curvature quantity because it is more suitable than the existing Lipschitz smoothness related ESO quantity.

Convergence Analysis for BCFW with Arbitrary Sampling

Equipped with the expected curvature, this subsection provides the convergence result of BCFW-AS in Theorem 1, the proof can be found in the supplement.

Theorem 1. *(Convergence result of BCFW-AS) Let Assumption 1 and 2 hold. Let $D_{\mathcal{M}}$ denote the diameter of constraint set $\mathcal{M}$, $|\mathcal{T}|$ denote the maximum sampling set size among iteration $1, ...t$. Take $\varrho_g = \max_i \varrho_g^i$, $\varrho_l = \max_i \varrho_l^i$, $\varrho = (D_{\mathcal{M}} + |\mathcal{T}|)\varrho_g + \varrho_l|\mathcal{T}|)$. For each $t \geq 0$, $\boldsymbol{x}^t$ generated by Algorithm 1 with step size $\gamma^t = \frac{2}{p_{min}\cdot t+2}$ satisfies*

$$\mathbb{E}[f(\boldsymbol{x}^t)] - f(\boldsymbol{x}^*) \leq \frac{2(h^0 + \frac{(1+\varrho)}{p_{min}} C_f^{\mathbb{E}\mathcal{S}})}{p_{min}\cdot t + 2}, \tag{11}$$

where $\boldsymbol{x}^*$ denotes an optimum of the problem, $h^0 := f(\boldsymbol{x}^0) - f(\boldsymbol{x}^*)$, p_{min} denotes the minimum entry of probability vector $\boldsymbol{p}$, $C_f^{\mathbb{E}\mathcal{S}}$ is the expected curvature for arbitrary sampling $\mathcal{S}$.

According to Theorem 1, instead of designing the step size for various sampling schemes case by case, BCFW-AS introduces a universal choice that only depends on the minimum probability entry p_{min} (note that further line-search for the step size is also possible as considered in (Jaggi 2013; Lacoste-julien et al. 2013), but the convergence analysis is still based on the deterministic step size). This universal step-size alleviates design complexity, especially when applied to complex sampling schemes where one may not know full details of the distribution. In particular, in the distributed private LASSO application, participating user nodes may have their own sampling schemes and do not want to reveal, making it impossible to have full knowledge of the overall sampling distribution.

To demonstrate the flexibility of the convergence guarantee provided by Theorem 1, we recover the convergence results of the previous FW algorithms by simply substituting the corresponding p_{min} and expected curvature obtained in the previous subsection in Eq.(11), which are summarized in Table 1. In particular, we have

1. **Elementary sampling**: we obtain the same convergence rate and dependence on the curvature;
2. **Uniform serial sampling**: we obtained a slightly refined convergence result compared to (Lacoste-julien et al. 2013) since our curvature-related constant is upper bounded by theirs according to Proposition 2;
3. τ**-nice sampling**: roughly, we recover a compatible result to (Wang et al. 2016) that with same convergence rate, our dependence on the curvature-related constant is better while the dependence on the initial primal gap is inferior than theirs;
4. (K, τ)**-distributed sampling**: we obtain the convergence for this new sampling by simply substituting $p_{min} = \frac{(K\tau)^2}{d^2}$ and expected curvature $C_f^{\mathbb{E}(K,\tau)}$ in Eq.(10), which shows the generality of Theorem 1.

Finally, we would like to remark that the convergence developed for arbitrary samplings has the potential to be applied to developing new BCFW algorithms with guaranteed convergence, like a variant with importance sampling scheme.

Uplink Communication Efficient Differentially Private BCFW with Distributed Features

This section considers the private LASSO problem for feature-wise distributed datasets and proposes an uplink communication efficient algorithm based on BCFW-AS algorithm. Two major components are: 1) Private index computation subroutine for computing private linear oracle; 2) Private active feature sharing required for computing partial gradient without privacy leakage. The algorithm comes with guaranteed privacy, utility and uplink communication complexity.

Algorithm Description

K user nodes solve the LASSO problem:

$$\min_{\boldsymbol{x}} \frac{1}{2n}||\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}||_2^2,\ s.t. ||\boldsymbol{x}||_1 \leq \eta, \tag{12}$$

Algorithm 2 Differentially Private Frank-Wolfe Algorithm with Distributed Features for LASSO

····· Server Node ·····

Require: Initial feasible variable $\boldsymbol{x}_0$, step sequence γ^t, maximum iteration T, ℓ_1-norm ball size η;

1: **for** $t = 0, 1, ..., T-1$ **do**
2: Receive $i\hat{d}x_k^t$ and $\hat{\boldsymbol{a}}_k^t$ from K workers;
3: $\hat{\boldsymbol{s}}^t = \eta \sum_{k=1}^K -sign(i\hat{d}x_k^t)\boldsymbol{e}_{[i\hat{d}x_k^t]}$;
4: $\boldsymbol{x}^{t+1} = (1-\gamma^t)\boldsymbol{x}^t + \gamma^t \hat{\boldsymbol{s}}^t$;
5: $\hat{\boldsymbol{q}}^t = (1-\gamma^{t-1})\boldsymbol{q}^{t-1} + \gamma^{t-1}\sum_{k=1}^K -sign(idx_k^t)\frac{1}{n}\hat{\boldsymbol{a}}^{t-1}_{|idx_k^t|}$;
6: Broadcast $\boldsymbol{x}^{t+1}$ and $\hat{\boldsymbol{q}}^t$ to all K workers;
7: **end for**

Ensure: $\boldsymbol{x}^T$;

····· User Node ·····

Require: sampling parameter τ, JL-transform matrix $\boldsymbol{J}$

7: **for** $t = 0, 1, ..., T-1$ **do**
8: Receive $\boldsymbol{x}^t$ and $\hat{\boldsymbol{q}}^t$ from server;
9: Sample τ coordinates $\mathcal{T}_k^t \subseteq \mathcal{P}_k$ randomly;
10: For each $i \in \mathcal{T}_k^t$, $\hat{\nabla}_i f(\boldsymbol{x}^t) = \boldsymbol{a}_i^\top \hat{\boldsymbol{q}}^t - \boldsymbol{a}_i^\top \boldsymbol{y}$, $v_i = \hat{\nabla}_i f(\boldsymbol{x}^t) + pert$;
11: $i\hat{d}x_k^t = sign(v_i)\arg\max_{i\in\mathcal{T}_k^t} v_i$;
12: Send $i\hat{d}x_k^t$ and $\hat{\boldsymbol{a}}_k^t = \boldsymbol{J}\boldsymbol{a}_{|i\hat{d}x_k^t|} + \boldsymbol{\xi}$ to server;
13: **end for**

with the input matrix $\boldsymbol{A} = [\boldsymbol{a}_1, ..., \boldsymbol{a}_i, ..., \boldsymbol{a}_d]$, where $\boldsymbol{a}_i$ is an $n \times 1$ sparse feature vector with $nnz(\boldsymbol{a})$ non-zero entries at most. For simplicity, we assume the features to be randomly and evenly splitted among user nodes as $\boldsymbol{A} = [\boldsymbol{A}_1, ...\boldsymbol{A}_k, ...\boldsymbol{A}_K]$ and denote the associated coordinate indices as $\mathcal{P}_k$ for $\boldsymbol{A}_k$. To adapt the BCFW-AS to LASSO with distributed features, each user node k randomly samples τ coordinate $\mathcal{T}_k^t$ and computes the partial gradient $\nabla_{(\mathcal{T}_k^t)} f(\boldsymbol{x}^t)$ and then evaluates partial linear oracle $\arg\max_{||\boldsymbol{s}_{(\mathcal{T}_k^t)}||_1 \leq \eta} \langle \boldsymbol{s}_{(\mathcal{P}_k)}, \nabla_{(\mathcal{T}_k^t)} f(\boldsymbol{x}^t)\rangle$. Note that the linear oracle evaluation amounts to find a direction $\pm\boldsymbol{e}_i,\ i \in \mathcal{T}_k^t$ that has the maximum $\langle \pm\boldsymbol{e}_i, \nabla_i f(\boldsymbol{x}^t)\rangle$ value. The following summarizes the partial gradient computation and linear oracle evaluation for user k:

$$\text{for each } i \in \mathcal{T}_k^t, \nabla_i f(\boldsymbol{x}^t) = \boldsymbol{a}_i^\top(\frac{1}{n}\boldsymbol{A}\boldsymbol{x}^t) - \boldsymbol{a}_i^\top \boldsymbol{y}; \tag{13}$$

$$i_k^t = \arg\max_{i\in\mathcal{T}_k^t} |\nabla_i f(\boldsymbol{x}^t)|,\ idx_k^t = -sign(\nabla_{i_k^t} f(\boldsymbol{x}^t)) \cdot i_k^t. \tag{14}$$

At iteration t, to update the decision variable, the user node only needs to upload the signed index idx_k^t rather than a full partial decision variable. To compute the partial gradient $\nabla_i f(\boldsymbol{x}^t)$, $\boldsymbol{q}^t := \frac{1}{n}\boldsymbol{A}\boldsymbol{x}^t$ cannot be updated by each user independently based solely on local features, which requires additional communication (recall that local gradient can be computed only based on local samples in the sample-wise distributed setting). However, it can be iteratively updated

based on the update rule of $\boldsymbol{x}^t$, as

$$\boldsymbol{q}^t = (1-\gamma^{t-1})\boldsymbol{q}^{t-1} + \gamma^{t-1}\sum_{k=1}^{K} -sign(idx_k^t)\frac{1}{n}\boldsymbol{a}^{t-1}_{|idx_k^t|}. \tag{15}$$

Hence, in order to update $\boldsymbol{q}^t$ from $\boldsymbol{q}^{t-1}$, it suffices to sharing K local features $\boldsymbol{a}_{i\hat{dx}_k^{t-1}}$. That is, each user node should upload the local feature indicated by idx_k^t, which is referred to active feature. Apparently, at most KT nonduplicate active features need to be communicated across T iterations. Compared with (Heinze-Deml, McWilliams, and Meinshausen 2017) which randomly sketches features at preprocessing which costs $O(n \times r\log(r))$ uplink communication, the active feature communication takes a "share-at-need" strategy which costs only $O(n \times KT)$ communication complexity. Later, we will show that this strategy indeed has smaller communication complexity. Two subroutines: private singed index computing and private active feature sharing, provide privacy protection to the algorithm. Given privacy budget (ϵ, δ), we evenly assign half for each part for simplicity.

A. Private signed index computation: The computation of idx can be regarded as selecting from $2|\mathcal{T}_k^t|$ indices whose associated value $sign(idx)\nabla_{|idx|}f(\boldsymbol{x})$ reaches maximum. This equivalence suggests the usage of "Report-noisy-max", a differential privacy building block, to provide privacy protection for the computation of idx. In brief, instead of selecting the index based on clean associated value, "Report-noisy-max" selects based on noise-injected associated value. For our problem, it has the updates as follows:

1. For each $i \in \mathcal{T}_k^t$, compute noise injected partial gradient by $v_i = sign(i)\nabla_i f(\boldsymbol{x}) + pert$, where $pert$ is a Laplacian noise: $pert \sim Lap(\frac{(2G\eta/Kn)\cdot\sqrt{2T\log(1/(\delta/2))}}{(\frac{\epsilon}{2k})/2})$.
2. Pick $\hat{idx}_k^t = sign(v_{i^*})i^*$, where $v_{i^*} = \max_{i\in\mathcal{T}_k^t} v_i$.

(Talwar, Thakurta, and Zhang 2015) also uses "Report-noisy-max" in the centralized FW algorithm for linear oracle selection, and our private signed index computation is an adaptation to the distributed case. We emphasize that, to develop private FW under feature-wise distributed setting, a more important design is how to share the active features in a private and communication efficient way without deteriorating the optimal utility guarantee, which is presented below.

B. Private active feature sharing: We propose to communicate perturbed sketch active features to jointly provide privacy protection as well as reduce communication cost. For $\boldsymbol{a}^t_{|i\hat{dx}|} \in \boldsymbol{A}_{trans}$, the user node takes $\hat{\boldsymbol{a}}^t_{|i\hat{dx}|} = \boldsymbol{J}\boldsymbol{a}^t_{|i\hat{dx}|} + \boldsymbol{\xi}$, where $\boldsymbol{J}$ is an $m \times n$ Gaussian sketch matrix (Woodruff and others 2014) (a type of Johnson-Lindenstrauss transformation matrix) and $\boldsymbol{\xi}$ is an $m \times 1$ noise vector with each entry sampled according to $pert \sim \mathcal{N}(0, \pi^2)$, $\pi = \frac{\sigma(\boldsymbol{J})\sqrt{KT}\sqrt{2(\ln(\frac{1}{\delta})+\epsilon/2)}}{n\epsilon/2}$, where $\sigma(\boldsymbol{J})$ is the leading singular value of $\boldsymbol{J}$. Analysis shows that m can be as small as $m = \Omega(\frac{1}{\iota^2}\log\frac{T}{\delta/2})$ with $T = O((n\epsilon)^{2/3})$ and ι being a small constant parameter (referred to as JL-parameter), which greatly reduces the communication cost for sharing the active features.

We stress that the sketched active feature sharing for reducing communication cost itself is new to distributed FW type algorithms, even without privacy protection design. Also, we provide an analysis to show that, with the provided m, the algorithm is guaranteed to converge with same $O(\frac{1}{T})$ rate. In addition, this sketching is also crucial for ensuring optimal utility guarantee. Note that the gradient inexact parameter is the much smaller $O(\log(n)/n)$ with sketch (see Table 2 and Theorem 3), compared to that of $O(\sqrt{n}/n)$ without sketch. Hence, the sketched features require much less noise injection. With inexactness parameter as large as $O(\sqrt{n}/n)$, the optimal utility $O((n\epsilon)^{\frac{2}{3}})$ would be no longer achievable.

The above is main steps taken by the user nodes. As for the server node, it simplicity updates the decision variable by adding up partial linear oracles and broadcasts the new decision variable as well as private active features. The algorithm is summarized in Algorithm 2.

Analysis

Algorithm 2 comes with guaranteed privacy, utility and uplink communication complexity.

A. Differential privacy: The sensitive communication sourced from user nodes is the indices and active features, both of which are $(\epsilon/2, \delta/2)$-differentially private by Lemma 7 and Lemma 8 in the supplement.

Then, the algorithm is guaranteed to be (ϵ, δ)-differential privacy by simple composition property of DP, as summarized by:

Theorem 2. *Algorithm 2 is (ϵ, δ)-differentially private.*

B. Utility: The utility is based on the convergence result of BCFW-AS developed in the previous section with the sampling being (K, τ)-distributed sampling. As perturbation and sketching are introduced, the gradient and linear oracle are inexact. The key step is to show Algorithm 2 satisfies Assumption 1 and show that $\hat{\boldsymbol{s}}^t$ and $\hat{\nabla}f(\boldsymbol{x}^t)$ are ϱ_l-LO and ϱ_g-PG correspondingly. The following theorem presents the utility guarantee and the parameters appeared are summarized in Table 2, where $nnz(\boldsymbol{a})$ denotes the largest count of nonzero entries of all features, $p_{min} = p = \frac{K\tau}{d}$ is the sampling rate, ι is the JL-transform parameter and $G = \frac{1}{n}||\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y})||_\infty = O(1)$.

Theorem 3. *Let Assumption 1 and 2 hold. Set $T = \left(\frac{C_f^{\mathbb{E}(K,\tau)}n\epsilon}{C_g + C_l}\right)^{\frac{2}{3}}$. Algorithm 2 ensures the following expected excess empirical risk under (ϵ, δ)-differential privacy,*

$$\begin{aligned}&\mathbb{E}[f(\boldsymbol{x}^T)] - \min_{x\in\mathcal{M}} f(\boldsymbol{x}) \\ &= O\Big(\frac{(C_g + C_l)^{\frac{2}{3}}(C_f^{\mathbb{E}(K,\tau)})^{\frac{1}{3}}\log(2G\eta K\tau n)}{p^2(n\epsilon)^{\frac{2}{3}}}\Big).\end{aligned} \tag{16}$$

Remark 1. According to Theorem 3, the utility is of order $O(\frac{1}{p^2(n\epsilon)^{\frac{2}{3}}})$. Compared with the utility result $O\Big(\frac{1}{(n\epsilon)^{\frac{2}{3}}}\Big)$ of the centralized private FW method (Talwar, Thakurta, and Zhang 2015), ours has the same dependence on n and ϵ, which is nearly optimal for any private algorithm achievable. Our utility is discounted by the sampling rate-related term,

Table 2: Summarization of Parameters in Theorem 3

Parameter	Formulation
ϱ_g	$O\Big(\frac{((K\iota nnz(\boldsymbol{a})+K^{3/2}\sqrt{nnz(\boldsymbol{a})m}\sigma(\boldsymbol{J})^2)/p_{min}(2\eta+K\tau))\sqrt{T}\sqrt{(\log(1/\delta)+\epsilon)}}{n\epsilon C_f^{\mathbb{E}\mathcal{S}}\gamma^T}\Big)$
ϱ_l	$O\Big(\frac{2G\eta\sqrt{32T\log(1/(\delta/2))}\log(2G\eta K\tau T)/(K\tau)}{n\epsilon C_f^{\mathbb{E}(K,\tau)}\gamma^T}\Big)$
C_g	$((K\iota nnz(\boldsymbol{a})+K^{3/2}\sqrt{nnz(\boldsymbol{a})}\sigma(\boldsymbol{J})^2)/p_{min})\sqrt{(\log(1/\delta)+\epsilon)}$
C_l	$2G\eta\sqrt{32\log(1/(\delta/2))}$
m	$\Omega(\frac{1}{\iota^2}\log\frac{T}{\delta/2})$

which, however, can be regarded as the trade-off between computational scalability and utility.

C. Uplink communication complexity: The uplink communication comes from: 1) KT *integers* for sending the private index (rather than the entire $d \times T$ *float* local decision variables x); 2) $m \times KT$ for sending the private active features. Based on T in Theorem 3 and m in Table 2, we have

Corollary 1. *Algorithm 2 has uplink communication complexity* $O(\frac{1}{\iota}^2 \log(n\epsilon)^{1/3}K(n\epsilon)^{2/3})$.

Remark 2. Compared to the one-shot communication at the preprocessing proposed by method (Heinze-Deml, McWilliams, and Meinshausen 2017), our uplink communication cost has better dependence on the sample size with $O(n^{2/3}\log(n)^{1/3})$ than theirs with $O(nr\log r)$, which shows our "share-at-need" feature sharing is more uplink communication efficient than random sketching at preprocessing.

Conclusion

We have considered private training of ERM model with the features distributed among user nodes and developed an uplink communication efficient, utility nearly optimal algorithm based on a new general analysis of BCFW algorithm under arbitrary sampling. Under the same privacy budget, our distributed variant features: 1) the same order of nearly optimal utility guarantee for the LASSO task as centralized counterpart, and 2) improved overall uplink communication complexity than the existing methods for the same feature-distributed setting. To derive the convergence analysis, we have introduced a universal step size as a new expected curvature notion, which comes with the detailed comparison with the existing quantities. We have demonstrated the flexibility of the convergence analysis by recovering exact, refined or matchable result of existing BCFW method under the specific samplings.

Acknowledgment

This work was supported by National Natural Science Foundation of China under Grants: 61672444 and 61272366, and the Faculty Research Grant of Hong Kong Baptist University under Project: FRG2/16-17/051.

References

Bassily, R.; Smith, A.; and Thakurta, A. 2014. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Foundations of Computer Science (FOCS-14), 2014 IEEE 55th Annual Symposium on*, 464–473. IEEE.

Bellet, A.; Liang, Y.; Garakani, A. B.; Balcan, M.-F.; and Sha, F. 2015. A distributed frank-wolfe algorithm for communication-efficient sparse learning. In *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM-15)*, 478–486. SIAM.

Chaudhuri, K.; Monteleoni, C.; and Sarwate, A. D. 2011. Differentially private empirical risk minimization. *Journal of Machine Learning Research* 12(Mar):1069–1109.

Cheung, Y.-m., and Zeng, H. 2009. Local kernel regression score for selecting features of high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering* 21(12):1798–1802.

Cheung, Y.-m. 2005. Maximum weighted likelihood via rival penalized em for density mixture clustering with automatic model selection. *IEEE Transactions on Knowledge and Data Engineering* 17(6):750–761.

Dwork, C.; Roth, A.; et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9(3–4):211–407.

Han, S.; Topcu, U.; and Pappas, G. J. 2017. Differentially private distributed constrained optimization. *IEEE Transactions on Automatic Control* 62(1):50–64.

Heinze-Deml, C.; McWilliams, B.; and Meinshausen, N. 2017. Preserving differential privacy between features in distributed estimation. *arXiv preprint arXiv:1703.00403*.

Huang, Z.; Mitra, S.; and Vaidya, N. 2015. Differentially private distributed optimization. In *Proceedings of the 2015 International Conference on Distributed Computing and Networking*, 4. ACM.

Jaggi, M. 2013. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 427–435.

Jain, P., and Thakurta, A. 2013. Differentially private learning with kernels. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 118–126.

Kasiviswanathan, S. P., and Jin, H. 2016. Efficient private empirical risk minimization for high-dimensional learning. In

International Conference on Machine Learning (ICML-16), 488–497.

Kenthapadi, K.; Korolova, A.; Mironov, I.; and Mishra, N. 2013. Privacy via the johnson-lindenstrauss transform. *Journal of Privacy and Confidentiality* 5(1):39–71.

Kifer, D.; Smith, A.; and Thakurta, A. 2012. Private convex empirical risk minimization and high-dimensional regression. *Journal of Machine Learning Research* 1(41):3–1.

Lacoste-julien, S.; Jaggi, M.; Schmidt, M.; and Pletscher, P. 2013. Block-coordinate frank-wolfe optimization for structural svms. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 53–61.

Mangasarian, O. L.; Wild, E. W.; and Fung, G. M. 2008. Privacy-preserving classification of vertically partitioned data via random kernels. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 2(3):12.

Nozari, E.; Tallapragada, P.; and Cortés, J. 2016. Differentially private distributed convex optimization via objective perturbation. In *American Control Conference (ACC-16)*. IEEE.

Qu, Z., and Richtárik, P. 2016a. Coordinate descent with arbitrary sampling i: Algorithms and complexity. *Optimization Methods and Software* 31(5):829–857.

Qu, Z., and Richtárik, P. 2016b. Coordinate descent with arbitrary sampling ii: Expected separable overapproximation. *Optimization Methods and Software* 31(5):858–884.

Richtárik, P., and Takác, M. 2016a. Distributed coordinate descent method for learning with big data. *Journal of Machine Learning Research* 17(75):1–25.

Richtárik, P., and Takáč, M. 2016b. Parallel coordinate descent methods for big data optimization. *Mathematical Programming* 156(1-2):433–484.

Smith, A. 2011. Privacy-preserving statistical estimation with optimal convergence rates. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, 813–822. ACM.

Talwar, K.; Thakurta, A.; and Zhang, L. 2014. Private empirical risk minimization beyond the worst case: The effect of the constraint set geometry. *arXiv preprint arXiv:1411.5417*.

Talwar, K.; Thakurta, A.; and Zhang, L. 2015. (nearly) optimal private lasso. In *Advances in Neural Information Processing Systems (NIPS-15)*, 3025–3033.

Wang, Y.-X.; Sadhanala, V.; Dai, W.; Neiswanger, W.; Sra, S.; and Xing, E. 2016. Parallel and distributed block-coordinate frank-wolfe algorithms. In *International Conference on Machine Learning (ICML-16)*, 1548–1557.

Wang, Y.; Wan, J.; Guo, J.; Cheung, Y.-M.; and Yuen, P. C. 2017. Inference-based similarity search in randomized montgomery domains for privacy-preserving biometric identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Woodruff, D. P., et al. 2014. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science* 10(1–2):1–157.

Yu, H.; Vaidya, J.; and Jiang, X. 2006. Privacy-preserving svm classification on vertically partitioned data. In *Proceedings of the 10th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining (PAKDD-06)*, 647–656. Springer-Verlag.

Yunhong, H.; Liang, F.; and Guoping, H. 2009. Privacy-preserving svm classification on vertically partitioned data without secure multi-party computation. In *Natural Computation, 2009. ICNC'09. Fifth International Conference on*, volume 1, 543–546. IEEE.