
Learning Privately over Distributed Features: An ADMM Sharing Approach

Yaochen Hu
University of Alberta
yaochen@ualberta.ca

Peng Liu
University of Kent
P.Liu@kent.ac.uk

Linglong Kong
University of Alberta
lkong@ualberta.ca

Di Niu
University of Alberta
dniu@ualberta.ca

Abstract

Distributed machine learning has been widely studied in order to handle exploding amount of data. In this paper, we study an important yet less visited distributed learning problem where features are inherently distributed or vertically partitioned among multiple parties, and sharing of raw data or model parameters among parties is prohibited due to privacy concerns. We propose an ADMM sharing framework to approach risk minimization over distributed features, where each party only needs to share a single value for each sample in the training process, thus minimizing the data leakage risk. We establish convergence and iteration complexity results for the proposed parallel ADMM algorithm under non-convex loss. We further introduce a novel differentially private ADMM sharing algorithm and bound the privacy guarantee with carefully designed noise perturbation. The experiments based on a prototype system shows that the proposed ADMM algorithms converge efficiently in a robust fashion, demonstrating advantage over gradient based methods especially for data set with high dimensional feature spaces.

1 Introduction

The effectiveness of a machine learning model does not only depend on the quantity of samples, but also the quality of data, especially the availability of high-quality features. Recently, a wide range of distributed and collaborative machine learning schemes, including gradient-based methods Li et al. (2014a,b); Hsieh et al. (2017); Ho et al. (2013) and ADMM-based methods Zhang et al. (2018); Shi et al. (2014); Zhang and Zhu (2016a); Huang et al. (2018), have been proposed to enable learning from distributed samples, since collecting data for centralized learning will incur compliance overhead, privacy concerns, or even judicial issues. Most existing schemes, however, are under the umbrella of *data parallel* schemes, where multiple parties possess different training samples, each sample with the same set of features. For example, different users hold different images to jointly train a classifier.

An equally important scenario is to collaboratively learn from distributed features, where multiple parties may possess different features about a same sample, yet do not wish to share these features with each other. Examples include a user’s behavioural data logged by multiple apps, a patient’s record stored at different hospitals and clinics, a user’s investment behavior logged by multiple financial institutions and government agencies and so forth. The question is—how can we train a joint model to make predictions about a sample leveraging the potentially rich and vast features possessed by other parties, without requiring different parties to share their data to each other?

The motivation of gleaning insights from vertically partitioned data dates back to association rule mining Vaidya and Clifton (2002, 2003). A few very recent studies Kenthapadi et al. (2013); Ying et al. (2018); Hu et al. (2019); Heinze-Deml et al. (2018); Dai et al. (2018); Stolpe et al. (2016) have reinvestigated vertically partitioned features under the setting of distributed machine learning, which is motivated by the ever-increasing data dimensionality as well as the opportunity and challenge of

cooperation between multiple parties that may hold different aspects of information about the same samples.

In this paper, we propose an ADMM algorithm to solve the empirical risk minimization (ERM) problem, a general optimization formulation of many machine learning models visited by a number of recent studies on distributed machine learning Ying et al. (2018); Chaudhuri et al. (2011). We propose an ADMM-sharing-based distributed algorithm to solve ERM, in which each participant does not need to share any raw features or local model parameters to other parties. Instead, each party only transmits a single value for each sample to other parties, thus largely preventing the local features from being disclosed. We establish theoretical convergence guarantees and iteration complexity results under the non-convex loss in a fully parallel setting, whereas previously, the convergence of ADMM sharing algorithm for non-convex losses is only known for the case of sequential (Gauss-Seidel) execution Hong et al. (2016).

To further provide privacy guarantees, we present a privacy-preserving version of the ADMM sharing algorithm, in which the transmitted value from each party is perturbed by a carefully designed Gaussian noise to achieve the notion of ϵ, δ -differential privacy Dwork (2008); Dwork et al. (2014). For distributed features, the perturbed algorithm ensures that the probability distribution of the values shared is relatively insensitive to any change to a single feature in a party’s local dataset.

Experimental results on two realistic datasets suggest that our proposed ADMM sharing algorithm can converge efficiently. Compared to the gradient-based method, our method can scale as the number of features increases and yields robust convergence. The algorithm can also converge with moderate amounts of Gaussian perturbation added, therefore enabling the utilization of features from other parties to improve the local machine learning task.

1.1 Related Work

Machine Learning Algorithms and Privacy. Chaudhuri and Monteleoni (2009) is one of the first studies combining machine learning and differential privacy (DP), focusing on logistic regression. Shokri and Shmatikov (2015) applies a variant of SGD to collaborative deep learning in a data-parallel fashion and introduces its variant with DP. Abadi et al. (2016) provides a stronger differential privacy guarantee for training deep neural networks using a momentum accountant method. Pathak et al. (2010); Rajkumar and Agarwal (2012) apply DP to collaborative machine learning, with an inherent tradeoff between the privacy cost and utility achieved by the trained model. Recently, DP has been applied to ADMM algorithms to solve multi-party machine learning problems Zhang et al. (2018); Zhang and Zhu (2016a); Zhang et al. (2019); Zhang and Zhu (2017).

However, all the work above is targeting the data-parallel scenario, where samples are distributed among nodes. The uniqueness of our work is to enable privacy-preserving machine learning among nodes with vertically partitioned features, or in other words, the feature-parallel setting, which is equally important and is yet to be explored.

Another approach to privacy-preserving machine learning is through encryption Gilad-Bachrach et al. (2016); Takabi et al. (2016); Kikuchi et al. (2018) or secret sharing Mohassel and Zhang (2017); Wan et al. (2007); Bonte and Vercauteren (2018), so that models are trained on encrypted data. However, encryption cannot be generalized to all algorithms or operations, and incurs additional computational cost.

Learning over Distributed Features. Gratton et al. (2018) applies ADMM to solve ridge regression. Ying et al. (2018) proposes a stochastic learning method via variance reduction. Zhou et al. (2016) proposes a proximal gradient method and mainly focuses on speeding up training in a model-parallel scenario. These studies do not consider the privacy issue. Hu et al. (2019) proposes a composite model structure that can jointly learn from distributed features via a SGD-based algorithm and its DP-enabled version, yet without offering theoretical privacy guarantees. Our work establishes the first (ϵ, δ) -differential privacy guarantee result for learning over distributed features. Experimental results further suggest that our ADMM sharing method converges in fewer epochs than gradient methods in the case of high dimensional features. This is critical to preserving privacy in machine learning since the privacy loss increases as the number of epochs increases Dwork et al. (2014).

Querying Vertically Partitioned Data Privately. Vaidya and Clifton (2002); Evfimievski et al. (2004); Dwork and Nissim (2004) are among the early studies that investigate the privacy issue of

querying vertically partitioned data. Kenthapadi et al. (2012) adopts a random-kernel-based method to mine vertically partitioned data privately. These studies provide privacy guarantees for simpler static queries, while we focus on machine learning jobs, where the risk comes from the shared values in the optimization algorithm. Our design simultaneously achieves minimum message passing, fast convergence, and a theoretically bounded privacy cost under the DP framework.

2 Empirical Risk Minimization over Distributed Features

Consider N samples, each with d features distributed on M parties, which do not wish to share data with each other. The entire dataset $\mathcal{D} \in \mathbb{R}^N \times \mathbb{R}^d$ can be viewed as M vertical partitions $\mathcal{D}_1, \dots, \mathcal{D}_M$, where $\mathcal{D}_m \in \mathbb{R}^N \times \mathbb{R}^{d_m}$ denotes the data possessed by the m th party and d_m is the dimension of features on party m . Clearly, $d = \sum_{m=1}^M d_m$. Let \mathcal{D}^i denote the i th row of \mathcal{D} , and \mathcal{D}_m^i be the i th row of \mathcal{D}_m ($k = 1, \dots, N$). Then, we have

$$\mathcal{D} = \begin{bmatrix} \mathcal{D}_1^1 & \mathcal{D}_2^1 & \dots & \mathcal{D}_M^1 \\ \mathcal{D}_1^2 & \mathcal{D}_2^2 & \dots & \mathcal{D}_M^2 \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{D}_1^N & \mathcal{D}_2^N & \dots & \mathcal{D}_M^N \end{bmatrix},$$

where $\mathcal{D}_m^i \in \mathcal{A}_m \subset \mathbb{R}^{d_m}$, ($i = 1, \dots, N, m = 1, \dots, M$). Let $Y_i \in \{-1, 1\}^N$ be the label of sample i .

Let $x = (x_1^\top, \dots, x_m^\top, \dots, x_M^\top)^\top$ represent the model parameters, where $x_m \in \mathbb{R}^{d_m}$ are the local parameters associated with the m th party. The objective is to find a model $f(\mathcal{D}^i; x)$ with parameters x to minimize the regularized empirical risk, i.e.,

$$\underset{x \in X}{\text{minimize}} \quad \frac{1}{N} \sum_{i=1}^N l_i(f(\mathcal{D}^i; x), Y_i) + \lambda R(x),$$

where $X \subset \mathbb{R}^d$ is a closed convex set and the regularizer $R(\cdot)$ prevents overfitting.

Similar to recent literature on distributed machine learning Ying et al. (2018); Zhou et al. (2016), ADMM Zhang and Zhu (2016a); Zhang et al. (2018), and privacy-preserving machine learning Chaudhuri et al. (2011); Hamm et al. (2016), we assume the loss has a form

$$\sum_{i=1}^N l_i(f(\mathcal{D}^i; x), Y_i) = \sum_{i=1}^N l_i(\mathcal{D}^i x, Y_i) = l\left(\sum_{m=1}^M \mathcal{D}_m^i x_m\right),$$

where we have abused the notation of l and in the second equality absorbed the label Y_i into the loss l , which is possibly a non-convex function. This framework incorporates a wide range of commonly used models including support vector machines, Lasso, logistic regression, boosting, etc.

Therefore, the risk minimization over distributed features, or vertically partitioned datasets $\mathcal{D}_1, \dots, \mathcal{D}_M$, can be written in the following compact form:

$$\underset{x}{\text{minimize}} \quad l\left(\sum_{m=1}^M \mathcal{D}_m x_m\right) + \lambda \sum_{m=1}^M R_m(x_m), \quad (1)$$

$$\text{subject to} \quad x_m \in X_m, m = 1, \dots, M, \quad (2)$$

where $X_m \subset \mathbb{R}^{d_m}$ is a closed convex set for all m .

We have further assumed the regularizer is separable such that $R(x) = \sum_{m=1}^M R_m(x_m)$. This assumption is consistent with our algorithm design philosophy—under vertically partitioned data, we require each party focus on training and regularizing its local model x_m , without sharing any local model parameters or raw features to other parties at all.

3 The ADMM Sharing Algorithm

We present an ADMM sharing algorithm Boyd et al. (2011); Hong et al. (2016) to solve Problem (1) and establish a convergence guarantee for the algorithm. Our algorithm requires each party only

Algorithm 1 The ADMM Sharing Algorithm

```
1: —Each party  $m$  performs in parallel:
2: for  $t$  in  $1, \dots, T$  do
3:   Pull  $\sum_k \mathcal{D}_k x_k^t - z^t$  and  $y^t$  from central node
4:   Obtain  $\sum_{k \neq m} \mathcal{D}_k x_k^t - z^t$  by subtracting the locally cached  $\mathcal{D}_m x_m^t$  from the pulled value
    $\sum_k \mathcal{D}_k x_k^t - z^t$ 
5:   Compute  $x_m^{t+1}$  according to (6)
6:   Push  $\mathcal{D}_m x_m^{t+1}$  to the central node
7: —Central node:
8: for  $t$  in  $1, \dots, T$  do
9:   Collect  $\mathcal{D}_m x_m^{t+1}$  for all  $m = 1, \dots, M$ 
10:  Compute  $z^{t+1}$  according to (7)
11:  Compute  $y^{t+1}$  according to (8)
12:  Distribute  $\sum_k \mathcal{D}_k x_k^{t+1} - z^{t+1}$  and  $y^{t+1}$  to all the parties.
```

share a single value to other parties in each iteration, thus requiring the minimum message passing. In particular, Problem (1) is equivalent to

$$\underset{x}{\text{minimize}} \quad l(z) + \lambda \sum_{m=1}^M R_m(x_m), \quad (3)$$

$$\text{s.t.} \quad \sum_{m=1}^M \mathcal{D}_m x_m - z = 0, \quad x_m \in X_M, m = 1, \dots, M, \quad (4)$$

where z is an auxiliary variable. The corresponding augmented Lagrangian is given by

$$\mathcal{L}(\{x\}, z; y) = l(z) + \lambda \sum_{m=1}^M R_m(x_m) + \langle y, \sum_{m=1}^M \mathcal{D}_m x_m - z \rangle + \frac{\rho}{2} \left\| \sum_{m=1}^M \mathcal{D}_m x_m - z \right\|^2, \quad (5)$$

where y is the dual variable and ρ is the penalty factor. In the t^{th} iteration of the algorithm, variables are updated according to

$$x_m^{t+1} := \underset{x_m \in X_m}{\text{argmin}} \quad \lambda R_m(x_m) + \langle y^t, \mathcal{D}_m x_m \rangle + \frac{\rho}{2} \left\| \sum_{k=1, k \neq m}^M \mathcal{D}_k x_k^t + \mathcal{D}_m x_m - z^t \right\|^2, \quad (6)$$
$$m = 1, \dots, M$$

$$z^{t+1} := \underset{z}{\text{argmin}} \quad l(z) - \langle y^t, z \rangle + \frac{\rho}{2} \left\| \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z \right\|^2 \quad (7)$$

$$y^{t+1} := y^t + \rho \left(\sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z^{t+1} \right). \quad (8)$$

Formally, in a distributed and fully parallel manner, the algorithm is described in Algorithm 1. Note that each party m needs the value $\sum_{k \neq m} \mathcal{D}_k x_k^t - z^t$ to complete the update, and Line 3, 4 and 12 in Algorithm 1 present a trick to reduce communication overhead.

3.1 Convergence Analysis

We follow Hong et al. (2016) to establish the convergence guarantee of the proposed algorithm under mild assumptions. Note that Hong et al. (2016) provides convergence analysis for the Gauss-Seidel version of the ADMM sharing, where x_1, \dots, x_M are updated sequentially, which is not naturally suitable to parallel implementation. In (6) of our algorithm, x_m 's can be updated by different parties in parallel in each iteration. We establish convergence as well as iteration complexity results for this parallel scenario, which is more realistic in distributed learning. We need the following set of common assumptions.

Assumption 1 1. There exists a positive constant $L > 0$ such that

$$\|\nabla l(x) - \nabla l(z)\| \leq L\|x - z\| \quad \forall x, z.$$

Moreover, for all $m \in \{1, 2, \dots, M\}$, X_m 's are closed convex sets; each \mathcal{D}_m is of full column rank so that the minimum eigenvalue $\sigma_{\min}(\mathcal{D}_m^\top \mathcal{D}_m)$ of matrix $\mathcal{D}_m^\top \mathcal{D}_m$ is positive.

2. The penalty parameter ρ is chosen large enough such that

- (a) each x_m subproblem (6) as well as the z subproblem (7) is strongly convex, with modulus $\{\gamma_m(\rho)\}_{m=1}^M$ and $\gamma(\rho)$, respectively.
- (b) $\gamma_m(\rho) \geq 2\sigma_{\max}(\mathcal{D}_m^\top \mathcal{D}_m)$, $\forall m$, where $\sigma_{\max}(\mathcal{D}_m^\top \mathcal{D}_m)$ is the maximum eigenvalue for matrix $\mathcal{D}_m^\top \mathcal{D}_m$.
- (c) $\rho\gamma(\rho) > 2L^2$ and $\rho \geq L$.

3. The objective function $l\left(\sum_{m=1}^M \mathcal{D}_m x_m\right) + \lambda \sum_{m=1}^M R_m(x_m)$ in Problem 1 is lower bounded over $\Pi_{m=1}^M X_m$ and we denote the lower bound as \underline{f} .

4. R_m is either smooth nonconvex or convex (possibly nonsmooth). For the former case, there exists $L_m > 0$ such that $\|\nabla R_m(x_m) - \nabla R_m(z_m)\| \leq L_m\|x_m - z_m\|$ for all $x_m, z_m \in X_m$.

Specifically, 1, 3 and 4 in Assumptions 1 are common settings in the literature. Assumptions 1.2 is achievable if the ρ is chosen large enough.

Denote $\mathcal{M} \subset \{1, 2, \dots, M\}$ as the index set, such that when $m \in \mathcal{M}$, R_m is convex, otherwise, R_m is nonconvex but smooth. Our convergence results show that under mild assumptions, the iteratively updated variables eventually converge to the set of primal-dual stationary solutions.

Theorem 1 Suppose Assumption 1 holds true, we have the following results:

- 1. $\lim_{t \rightarrow \infty} \|\sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z^{t+1}\| = 0$.
- 2. Any limit point $\{\{x^*\}, z^*; y^*\}$ of the sequence $\{\{x^{t+1}\}, z^{t+1}; y^{t+1}\}$ is a stationary solution of problem (1) in the sense that

$$x_m^* \in \operatorname{argmin}_{x_m \in X_m} \lambda R_m(x_m) + \langle y^*, \mathcal{D}_m x_m \rangle, m \in \mathcal{M}, \quad (9)$$

$$\langle x_m - x_m^*, \lambda \nabla l(x_m^*) - \mathcal{D}_m^\top y^* \rangle \leq 0 \quad \forall x_m \in X_m, m \notin \mathcal{M}, \quad (10)$$

$$\nabla l(z^*) - y^* = 0, \quad (11)$$

$$\sum_{m=1}^M \mathcal{D}_m x_m^* = z^*. \quad (12)$$

- 3. If \mathcal{D}_m is a compact set for all m , then $\{\{x_m^t\}, z^t; y^t\}$ converges to the set of stationary solutions of problem (1), i.e.,

$$\lim_{t \rightarrow \infty} \operatorname{dist}(\{\{x^t\}, z^t; y^t\}; Z^*) = 0,$$

where Z^* is the set of primal-dual stationary solutions for problem (1).

3.2 Iteration Complexity Analysis

We evaluate the iteration complexity over a *Lyapunov function*. More specifically, we define V^t as

$$V^t := \sum_{m=1}^M \|\tilde{\nabla}_{x_m} \mathcal{L}(\{x_m^t\}, z^t; y^t)\|^2 + \|\nabla_z \mathcal{L}(\{x_m^t\}, z^t; y^t)\|^2 + \left\| \sum_{m=1}^M \mathcal{D}_m x_m^t - z^t \right\|^2, \quad (13)$$

where

$$\tilde{\nabla}_{x_m} \mathcal{L}(\{x_m^t\}, z^t; y^t) = \nabla_{x_m} \mathcal{L}(\{x_m^t\}, z^t; y^t) \quad \text{when } m \notin \mathcal{M},$$

$$\tilde{\nabla}_{x_m} \mathcal{L}(\{x_m^t\}, z^t; y^t) = x_m^t - \operatorname{prox}_{\lambda R_m} \left[x_m^t - \nabla_{x_m} (\mathcal{L}(\{x_m^t\}, z^t; y^t) - \lambda \sum_{m=1}^M R_m(x_m^t)) \right]$$

when $m \in \mathcal{M}$,

with $\text{prox}_h[z] := \text{argmin}_x h(x) + \frac{1}{2}\|x - z\|^2$. It is easy to verify that when $V^t \rightarrow 0$, a stationary solution is achieved due to the properties. The result for the iteration complexity is stated in the following theorem, which provides a quantification of how fast our algorithm converges. Theorem 2 shows that the algorithm converges in the sense that the *Lyapunov function* V^t will be less than any $\epsilon > 0$ within $O(1/\epsilon)$ iterations.

Theorem 2 Suppose Assumption 1 holds. Let $T(\epsilon)$ denote the iteration index in which:

$$T(\epsilon) := \min\{t | V^t \leq \epsilon, t \geq 0\},$$

for any $\epsilon > 0$. Then there exists a constant $C > 0$, such that

$$T(\epsilon)\epsilon \leq C(\mathcal{L}(\{x^1\}, z^1; y^1 - \underline{f})), \quad (14)$$

where \underline{f} is the lower bound defined in Assumption 1.

4 Differentially Private ADMM Sharing

Differential privacy Dwork et al. (2014); Zhou et al. (2010) is a notion that ensures a strong guarantee for data privacy. The intuition is to keep the query results from a dataset relatively close if one of the entries in the dataset changes, by adding some well designed random noise into the query, so that little information on the raw data can be inferred from the query. Formally, the definition of differential privacy is given in Definition 1.

Definition 1 A randomized algorithm \mathcal{M} is (ϵ, δ) -differentially private if for all $S \subset \text{range}(\mathcal{M})$, and for all x and y , such that $|x - y|_1 \leq 1$:

$$\Pr(\mathcal{M}(x)) \leq \exp(\epsilon)\Pr(\mathcal{M}(y)) + \delta. \quad (15)$$

In our ADMM algorithm, the shared messages $\{\mathcal{D}_m x_m^{t+1}\}_{t=0,1,\dots,T-1}$ may reveal sensitive information from the data entry in D_m of Party m . We perturb the shared value $\mathcal{D}_m x_m^{t+1}$ in Algorithm 1 with a carefully designed random noise to provide differential privacy. The resulted perturbed ADMM sharing algorithm is the following updates:

$$\begin{aligned} x_m^{t+1} &:= \text{argmin}_{x_m \in X_m} \lambda R_m(x_m) + \langle y^t, \mathcal{D}_m x_m \rangle + \frac{\rho}{2} \left\| \sum_{k=1, k \neq m}^M \mathcal{D}_k \tilde{x}_k^t + \mathcal{D}_m x_m - z^t \right\|^2, \\ m &= 1, \dots, M \\ \xi_m^{t+1} &:= \mathcal{N}(0, \sigma_{m,t+1}^2 (\mathcal{D}_m^\top \mathcal{D}_m)^{-1}) \\ \tilde{x}_m^{t+1} &:= x_m^{t+1} + \xi_m^{t+1} \\ z^{t+1} &:= \text{argmin}_z l(z) - \langle y^t, z \rangle + \frac{\rho}{2} \left\| \sum_{m=1}^M \mathcal{D}_m \tilde{x}_m^{t+1} - z \right\|^2 \\ y^{t+1} &:= y^t + \rho \left(\sum_{m=1}^M \mathcal{D}_m \tilde{x}_m^{t+1} - z^{t+1} \right). \end{aligned} \quad (16)$$

In the remaining part of this section, we demonstrate that (16) guarantees (ϵ, δ) differential privacy with outputs $\{\mathcal{D}_m \tilde{x}_m^{t+1}\}_{t=0,1,\dots,T-1}$ for some carefully selected $\sigma_{m,t+1}$. Beside Assumption 1, we introduce another set of assumptions widely used by the literature.

Assumption 2 1. The feasible set $\{x, y\}$ and the dual variable z are bounded; their l_2 norms have an upper bound b_1 .

2. The regularizer $R_m(\cdot)$ is doubly differentiable with $|R_m''(\cdot)| \leq c_1$, where c_1 is a finite constant.

3. Each row of \mathcal{D}_m is normalized and has an l_2 norm of 1.

Note that Assumption 2.1 is adopted in Sarwate and Chaudhuri (2013) and Wang et al. (2019). Assumption 2.2 comes from Zhang and Zhu (2016b) and Assumption 2.3 comes from Zhang and Zhu (2016b) and Sarwate and Chaudhuri (2013). As a typical method in differential privacy analysis, we first study the l_2 sensitivity of $\mathcal{D}_m x_m^{t+1}$, which is defined by:

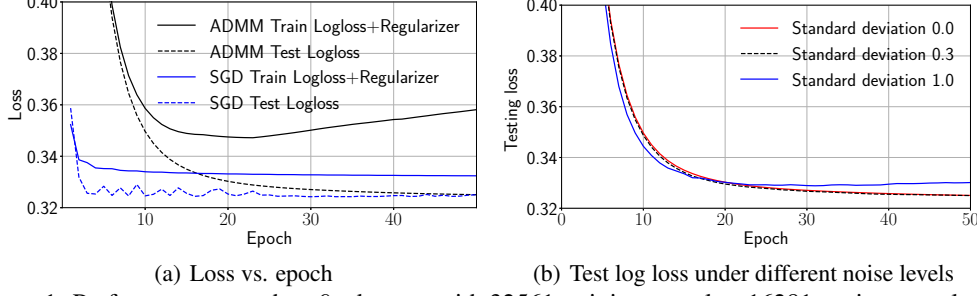


Figure 1: Performance over the *a9a* data set with 32561 training samples, 16281 testing samples and 123 features.

Definition 2 The l_2 -norm sensitivity of $\mathcal{D}_m x_m^{t+1}$ is defined by:

$$\Delta_{m,2} = \max_{\substack{\mathcal{D}_m, \mathcal{D}'_m \\ \|\mathcal{D}_m - \mathcal{D}'_m\| \leq 1}} \|\mathcal{D}_m x_{m,\mathcal{D}_m}^{t+1} - \mathcal{D}'_m x_{m,\mathcal{D}'_m}^{t+1}\|.$$

where \mathcal{D}_m and \mathcal{D}'_m are two neighbouring datasets differing in only one feature column, and $x_{m,\mathcal{D}_m}^{t+1}$ is the x_m^{t+1} derived from the first line of equation (16) under dataset \mathcal{D}_m .

We have Lemma 1 to state the upper bound of the l_2 -norm sensitivity of $\mathcal{D}_m x_m^{t+1}$.

Lemma 1 Assume that Assumption 1 and Assumption 2 hold. Then the l_2 -norm sensitivity of $\mathcal{D}_m x_{m,\mathcal{D}_m}^{t+1}$ is upper bounded by $\mathbb{C} = \frac{3}{d_m \rho} [\lambda c_1 + (1 + M\rho)b_1]$.

Theorem 3 Assume assumptions 2.1-2.3 hold and \mathbb{C} is the upper bound of $\Delta_{m,2}$. Let $\varepsilon \in (0, 1]$ be an arbitrary constant and let $\mathcal{D}_m \xi_m^{t+1}$ be sampled from zero-mean Gaussian distribution with variance $\sigma_{m,t+1}^2$, where

$$\sigma_{m,t+1} = \frac{\sqrt{2\ln(1.25/\delta)}\mathbb{C}}{\varepsilon}.$$

Then each iteration guarantees (ε, δ) -differential privacy. Specifically, for any neighboring datasets \mathcal{D}_m and \mathcal{D}'_m , for any output $\mathcal{D}_m \tilde{x}_{m,\mathcal{D}_m}^{t+1}$ and $\mathcal{D}'_m \tilde{x}_{m,\mathcal{D}'_m}^{t+1}$, the following inequality always holds:

$$P(\mathcal{D}_m \tilde{x}_{m,\mathcal{D}_m}^{t+1} | \mathcal{D}_m) \leq e^\varepsilon P(\mathcal{D}'_m \tilde{x}_{m,\mathcal{D}'_m}^{t+1} | \mathcal{D}'_m) + \delta.$$

With an application of the composition theory in Dwork et al. (2014), we come to a result stating the overall privacy guarantee for the whole training procedure.

Corollary 1 For any $\delta' > 0$, the algorithm described in (16) satisfies $(\varepsilon', T\delta + \delta')$ -differential privacy within T epochs of updates, where

$$\varepsilon' = \sqrt{2T\ln(1/\delta')}\varepsilon + T\varepsilon(e^\varepsilon - 1). \quad (17)$$

5 Experiments

We test our algorithm by training l_2 -norm regularized logistic regression on two popular public datasets, namely, *a9a* from UCI Dua and Graff (2017) and *giette* Guyon et al. (2005). We get the datasets from Lib ([n.d.]) so that we follow the same preprocessing procedure listed there. *a9a* dataset is 4 MB and contains 32561 training samples, 16281 testing samples and 123 features. We divide the dataset into two parts, with the first part containing the first 66 features and the second part remaining 57 features. The first part is regarded as the local party who wishes to improve its prediction model with the help of data from the other party. On the other hand, *gisette* dataset is 297 MB and contains 6000 training samples, 1000 testing samples and 5000 features. Similarly, we divide the features into 3 parts, the first 2000 features being the first part regarded as the local data, the next 2000 features being the second part, and the remaining 1000 as the third part. Note that *a9a* is small in terms of the number of features and *gisette* has a relatively higher dimensional feature space.

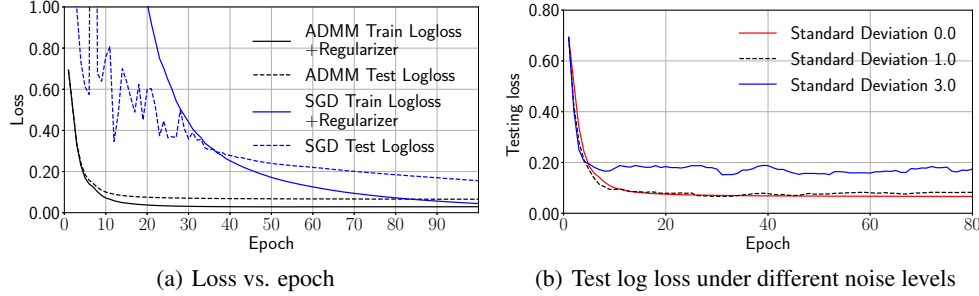


Figure 2: Performance over the *gisette* data set with 6000 training samples, 1000 testing samples and 5000 features.

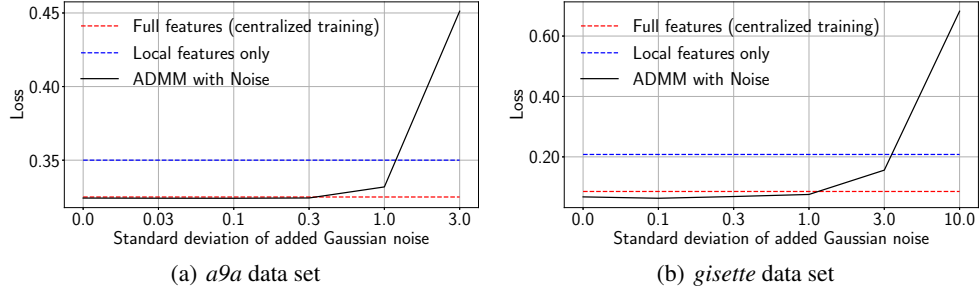


Figure 3: Test performance for ADMM under different levels of added noise.

A prototype system is implemented in *Python* to verify our proposed algorithm. Specifically, we use optimization library from *scipy* to handle the optimization subproblems. We apply L-BFGS-B algorithm to do the x update in (6) and entry-wise optimization for z in (7). We run the experiment on a machine equipped with Intel(R) Core(TM) i9-9900X CPU @ 3.50GHz and 128 GB of memory.

We compare our algorithm against an SGD based algorithm proposed in Hu et al. (2019). We keep track of the training objective value (log loss plus the l_2 regularizer), the testing log loss for each epoch for different datasets and parameter settings. We also test our algorithm with different levels of Gaussian noise added. In the training procedure, we initialize the elements in x , y and z with 0 while we initialize the parameter for the SGD-based algorithm with random numbers.

Fig. 1 and Fig. 2 show a typical trace of the training objective and testing log loss against epochs for *a9a* and *gisette*, respectively. On *a9a*, the ADMM algorithm is slightly slower than the SGD based algorithm, while they reach the same testing log loss in the end. On *gisette*, the SGD based algorithm converges slowly while the ADMM algorithm is efficient and robust. The testing log loss from the ADMM algorithm quickly converges to 0.08 after a few epochs, but the SGD based algorithm converges to only 0.1 with much more epochs. This shows that the ADMM algorithm is superior when the number of features is large. In fact, for each epoch, the x update is a trivial quadratic program and can be efficiently solved numerically. The z update contains optimization over computationally expensive functions, but for each sample, it is always an optimization over a single scalar so that it can be solved efficiently via scalar optimization and scales with the number of features.

Fig. 3 shows the testing loss for ADMM with different levels of Gaussian noise added. The other two baselines are the logistic regression model trained over all the features (in a centralized way) and that trained over only the local features in the first party. The baselines are trained with the built-in logistic regression function from *sklearn* library. We can see that there is a significant performance boost if we employ more features to help training the model on Party 1. Interestingly, in Fig. 3(b), the ADMM sharing has even better performance than the baseline trained with all features with *sklearn*. It further shows that the ADMM sharing is better at datasets with a large number of features.

Moreover, after applying moderate random perturbations, the proposed algorithm can still converge in a relatively small number of epochs, as Fig. 1(b) and Fig. 2(b) suggest, although too much noise may ruin the model. Therefore, ADMM sharing algorithm under moderate perturbation can improve the local model and the privacy cost is well contained as the algorithm converges in a few epochs.

6 Conclusion

We study learning over distributed features (vertically partitioned data) where none of the parties shall share the local data. We propose the parallel ADMM sharing algorithm to solve this challenging problem where only intermediate values are shared, without even sharing model parameters. We have shown the convergence for convex and non-convex loss functions. To further protect the data privacy, we apply the differential privacy technique in the training procedure to derive a privacy guarantee within T epochs. We implement a prototype system and evaluate the proposed algorithm on two representative datasets in risk minimization. The result shows that the ADMM sharing algorithm converges efficiently, especially on dataset with large number of features. Furthermore, the differentially private ADMM algorithm yields better prediction accuracy than model trained from only local features while ensuring a certain level of differential privacy guarantee.

References

- [n.d.]. LIBSVM Data: Classification (Binary Class). <https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html>. Accessed: 2019-05-23.
- Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 308–318.
- Charlotte Bonte and Frederik Vercauteren. 2018. *Privacy-Preserving Logistic Regression Training*. Technical Report. IACR Cryptology ePrint Archive 233.
- Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* 3, 1 (2011), 1–122.
- Kamalika Chaudhuri and Claire Monteleoni. 2009. Privacy-preserving logistic regression. In *Advances in neural information processing systems*. 289–296.
- Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. 2011. Differentially private empirical risk minimization. *Journal of Machine Learning Research* 12, Mar (2011), 1069–1109.
- Wenrui Dai, Shuang Wang, Hongkai Xiong, and Xiaoqian Jiang. 2018. Privacy preserving federated big data analysis. In *Guide to Big Data Applications*. Springer, 49–82.
- Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. <http://archive.ics.uci.edu/ml>
- Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*. Springer, 1–19.
- Cynthia Dwork and Kobbi Nissim. 2004. Privacy-preserving datamining on vertically partitioned databases. In *Annual International Cryptology Conference*. Springer, 528–544.
- Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.
- Alexandre Evfimievski, Ramakrishnan Srikant, Rakesh Agrawal, and Johannes Gehrke. 2004. Privacy preserving mining of association rules. *Information Systems* 29, 4 (2004), 343–364.
- Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. 2016. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning*. 201–210.
- Cristiano Gratton, Venkategowda Naveen KD, Reza Arablouei, and Stefan Werner. 2018. Distributed Ridge Regression with Feature Partitioning. In *2018 52nd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 1423–1427.
- Isabelle Guyon, Steve Gunn, Asa Ben-Hur, and Gideon Dror. 2005. Result analysis of the NIPS 2003 feature selection challenge. In *Advances in neural information processing systems*. 545–552.
- Jihun Hamm, Yingjun Cao, and Mikhail Belkin. 2016. Learning privately from multiparty data. In *International Conference on Machine Learning*. 555–563.
- Christina Heinze-Deml, Brian McWilliams, and Nicolai Meinshausen. 2018. Preserving Differential Privacy Between Features in Distributed Estimation. *stat* 7 (2018), e189.
- Qirong Ho, James Cipar, Henggang Cui, Seunghak Lee, Jin Kyu Kim, Phillip B Gibbons, Garth A Gibson, Greg Ganger, and Eric P Xing. 2013. More effective distributed ml via a stale synchronous parallel parameter server. In *Advances in neural information processing systems*. 1223–1231.
- Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. 2016. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization* 26, 1 (2016), 337–364.

- Kevin Hsieh, Aaron Harlap, Nandita Vijaykumar, Dimitris Konomis, Gregory R Ganger, Phillip B Gibbons, and Onur Mutlu. 2017. Gaia: Geo-Distributed Machine Learning Approaching LAN Speeds.. In *NSDI*. 629–647.
- Yaochen Hu, Di Niu, Jianming Yang, and Shengping Zhou. 2019. FDML: A Collaborative Machine Learning Framework for Distributed Features. In *Proceedings of KDD '19*. ACM.
- Zonghao Huang, Rui Hu, Yanmin Gong, and Eric Chan-Tin. 2018. DP-ADMM: ADMM-based Distributed Learning with Differential Privacy. *arXiv preprint arXiv:1808.10101* (2018).
- Krishnaram Kenthapadi, Aleksandra Korolova, Ilya Mironov, and Nina Mishra. 2012. Privacy via the johnson-lindenstrauss transform. *arXiv preprint arXiv:1204.2606* (2012).
- Krishnaram Kenthapadi, Aleksandra Korolova, Ilya Mironov, and Nina Mishra. 2013. Privacy via the Johnson-Lindenstrauss Transform. *Journal of Privacy and Confidentiality* 5 (2013).
- Hiroaki Kikuchi, Chika Hamanaga, Hideo Yasunaga, Hiroki Matsui, Hideki Hashimoto, and Chun-I Fan. 2018. Privacy-preserving multiple linear regression of vertically partitioned real medical datasets. *Journal of Information Processing* 26 (2018), 638–647.
- Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. 2014a. Scaling Distributed Machine Learning with the Parameter Server.. In *OSDI*, Vol. 14. 583–598.
- Mu Li, David G Andersen, Alexander J Smola, and Kai Yu. 2014b. Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems*. 19–27.
- Payman Mohassel and Yupeng Zhang. 2017. SecureML: A system for scalable privacy-preserving machine learning. In *2017 38th IEEE Symposium on Security and Privacy (SP)*. IEEE, 19–38.
- Manas Pathak, Shantanu Rane, and Bhiksha Raj. 2010. Multiparty differential privacy via aggregation of locally trained classifiers. In *Advances in Neural Information Processing Systems*. 1876–1884.
- Arun Rajkumar and Shivani Agarwal. 2012. A differentially private stochastic gradient descent algorithm for multiparty classification. In *Artificial Intelligence and Statistics*. 933–941.
- Anand D Sarwate and Kamalika Chaudhuri. 2013. Signal processing and machine learning with differential privacy: Algorithms and challenges for continuous data. *IEEE signal processing magazine* 30, 5 (2013), 86–94.
- Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. 2014. On the linear convergence of the ADMM in decentralized consensus optimization. *IEEE Transactions on Signal Processing* 62, 7 (2014), 1750–1761.
- Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 1310–1321.
- Marco Stolpe, Hendrik Blom, and Katharina Morik. 2016. Sustainable Industrial Processes by Embedded Real-Time Quality Prediction. In *Computational Sustainability*. Springer, 201–243.
- Hassan Takabi, Ehsan Hesamifard, and Mehdi Ghasemi. 2016. Privacy preserving multi-party machine learning with homomorphic encryption. In *29th Annual Conference on Neural Information Processing Systems (NIPS)*.
- Jaideep Vaidya and Chris Clifton. 2002. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 639–644.
- Jaideep Vaidya and Chris Clifton. 2003. Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 206–215.

- Li Wan, Wee Keong Ng, Shuguo Han, and Vincent Lee. 2007. Privacy-preservation for gradient descent methods. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 775–783.
- Yu Wang, Wotao Yin, and Jinshan Zeng. 2019. Global convergence of ADMM in nonconvex nonsmooth optimization. *Journal of Scientific Computing* 78, 1 (2019), 29–63.
- Bicheng Ying, Kun Yuan, and Ali H Sayed. 2018. Supervised Learning Under Distributed Features. *IEEE Transactions on Signal Processing* 67, 4 (2018), 977–992.
- Chunlei Zhang, Muaz Ahmad, and Yongqiang Wang. 2019. Admm based privacy-preserving decentralized optimization. *IEEE Transactions on Information Forensics and Security* 14, 3 (2019), 565–580.
- Tao Zhang and Quanyan Zhu. 2016a. A dual perturbation approach for differential private admm-based distributed empirical risk minimization. In *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*. ACM, 129–137.
- Tao Zhang and Quanyan Zhu. 2016b. Dynamic differential privacy for ADMM-based distributed classification learning. *IEEE Transactions on Information Forensics and Security* 12, 1 (2016), 172–187.
- Tao Zhang and Quanyan Zhu. 2017. Dynamic differential privacy for ADMM-based distributed classification learning. *IEEE Transactions on Information Forensics and Security* 12, 1 (2017), 172–187.
- Xueru Zhang, Mohammad Mahdi Khalili, and Mingyan Liu. 2018. Improving the Privacy and Accuracy of ADMM-Based Distributed Algorithms. In *International Conference on Machine Learning*. 5791–5800.
- Minqi Zhou, Rong Zhang, Wei Xie, Weining Qian, and Aoying Zhou. 2010. Security and privacy in cloud computing: A survey. In *2010 Sixth International Conference on Semantics, Knowledge and Grids*. IEEE, 105–112.
- Yi Zhou, Yaoliang Yu, Wei Dai, Yingbin Liang, and Eric Xing. 2016. On convergence of model parallel proximal gradient algorithm for stale synchronous parallel system. In *Artificial Intelligence and Statistics*. 713–722.

7 Supplementary Materials

7.1 Proof of Theorem 1

To help theoretical analysis, we denote the objective functions in (6) and (7) as

$$\begin{aligned} g_m(x_m) &= \lambda R_m(x_m) + \langle y^t, \mathcal{D}_m x_m \rangle + \frac{\rho}{2} \left\| \sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k x_k^t + \mathcal{D}_m x_m - z^t \right\|^2, \\ h(z) &= l(z) - \langle y^t, z \rangle + \frac{\rho}{2} \left\| \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z \right\|^2, \end{aligned} \quad (18)$$

correspondingly. We prove the following four lemmas to help prove the theorem.

Lemma 2 *Under Assumption 1, we have*

$$\nabla l(z^{t+1}) = y^{t+1},$$

and

$$\|y^{t+1} - y^t\|^2 \leq L^2 \|z^{t+1} - z^t\|^2.$$

Proof. By the optimality in (7), we have

$$\nabla l(z^{t+1}) - y^t + \rho \left(z^{t+1} - \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} \right) = 0.$$

Combined with (8), we can get

$$\nabla l(z^{t+1}) = y^{t+1}. \quad (19)$$

Combined with Assumption 1.1, we have

$$\|y^{t+1} - y^t\|^2 = \|\nabla l(z^{t+1}) - \nabla l(z^t)\|^2 \leq L^2 \|z^{t+1} - z^t\|^2. \quad (20)$$

□

Lemma 3 *We have*

$$\begin{aligned} & \left(\left\| \sum_{m=1}^M x_m^{t+1} - z \right\|^2 - \left\| \sum_{m=1}^M x_m^t - z \right\|^2 \right) - \sum_{m=1}^M \left(\left\| \sum_{\substack{k=1 \\ k \neq m}}^M x_k^t + x_m^{t+1} - z \right\|^2 - \left\| \sum_{m=1}^M x_m^t - z \right\|^2 \right) \\ & \leq \sum_{m=1}^M \|x_m^{t+1} - x_m^t\|^2. \end{aligned} \quad (21)$$

Proof.

$$\begin{aligned} \text{LHS} &= \left(\sum_{m=1}^M (x_m^{t+1} + x_m^t) - 2z \right)^\top \left(\sum_{m=1}^M x_m^{t+1} - \sum_{m=1}^M x_m^t \right) - \sum_{m=1}^M \left(\sum_{\substack{k=1 \\ k \neq m}}^M 2x_k^t + x_m^t + x_m^{t+1} - 2z \right)^\top (x_m^{t+1} - x_m^t) \\ &= - \sum_{m=1}^M \sum_{\substack{k=1 \\ k \neq m}}^M (x_k^{t+1} - x_k^t)^\top (x_m^{t+1} - x_m^t) \\ &= - \left\| \sum_{m=1}^M (x_m^{t+1} - x_m^t) \right\|^2 + \sum_{m=1}^M \|x_m^{t+1} - x_m^t\|^2 \\ &\leq \sum_{m=1}^M \|x_m^{t+1} - x_m^t\|^2. \end{aligned}$$

□

Lemma 4 Suppose Assumption 1 holds. We have

$$\begin{aligned} & \mathcal{L}(\{x_m^{t+1}\}, z^{t+1}; y^{t+1}) - \mathcal{L}(\{x_m^t\}, z^t; y^t) \\ & \leq \sum_{m=1}^M - \left(\frac{\gamma_m(\rho)}{2} - \sigma_{\max}(\mathcal{D}_m^\top \mathcal{D}_m) \right) \|x_m^{t+1} - x_m^t\|^2 - \left(\frac{\gamma(\rho)}{2} - \frac{L^2}{\rho} \right) \|z^{t+1} - z^t\|^2. \end{aligned}$$

Proof. The LFH can be decomposed into two parts as

$$\begin{aligned} & \mathcal{L}(\{x_m^{t+1}\}, z^{t+1}; y^{t+1}) - \mathcal{L}(\{x_m^t\}, z^t; y^t) \\ & = (\mathcal{L}(\{x_m^{t+1}\}, z^{t+1}; y^{t+1}) - \mathcal{L}(\{x_m^{t+1}\}, z^{t+1}; y^t)) \\ & \quad + (\mathcal{L}(\{x_m^{t+1}\}, z^{t+1}; y^t) - \mathcal{L}(\{x_m^t\}, z^t; y^t)). \end{aligned} \tag{22}$$

For the first term, we have

$$\begin{aligned} & \mathcal{L}(\{x_j^{t+1}\}, z^{t+1}; y^{t+1}) - \mathcal{L}(\{x_j^{t+1}\}, z^{t+1}; y^t) \\ & = \langle y^{t+1} - y^t, \sum_j D_j x_j^{t+1} - z^{t+1} \rangle \\ & = \frac{1}{\rho} \|y^{t+1} - y^t\|^2 \quad (\text{by (8)}) \\ & = \frac{L^2}{\rho} \|z^{t+1} - z^t\|^2 \quad (\text{by Lemma 2}). \end{aligned} \tag{23}$$

For the second term, we have

$$\begin{aligned} & \mathcal{L}(\{x_m^{t+1}\}, z^{t+1}; y^t) - \mathcal{L}(\{x_m^t\}, z^t; y^t) \\ & = \mathcal{L}(\{x_m^{t+1}\}, z^{t+1}; y^t) - \mathcal{L}(\{x_m^t\}, z^{t+1}; y^t) + \mathcal{L}(\{x_m^t\}, z^{t+1}; y^t) - \mathcal{L}(\{x_m^t\}, z^t; y^t) \\ & \leq \left(\left(\lambda \sum_{m=1}^M R_m(x_m^{t+1}) + \langle y^t, \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} \rangle + \frac{\rho}{2} \left\| \sum_{k=1}^M \mathcal{D}_k x_k^{t+1} - z^{t+1} \right\|^2 \right) \right. \\ & \quad \left. - \left(\lambda \sum_{m=1}^M R_m(x_m^t) + \langle y^t, \sum_{m=1}^M \mathcal{D}_m x_m^t \rangle + \frac{\rho}{2} \left\| \sum_{k=1}^M \mathcal{D}_k x_k^t - z^{t+1} \right\|^2 \right) \right) \\ & \quad + \left(\left(l(z^{t+1}) - \langle y^t, z^{t+1} \rangle + \frac{\rho}{2} \left\| \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z^{t+1} \right\|^2 \right) - \left(l(z^t) - \langle y^t, z^t \rangle + \frac{\rho}{2} \left\| \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z^t \right\|^2 \right) \right) \\ & \leq \sum_{m=1}^M \left(\left(\lambda R_m(x_m^{t+1}) + \langle y^t, \mathcal{D}_m x_m^{t+1} \rangle + \frac{\rho}{2} \left\| \sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k x_k^t + \mathcal{D}_m x_m^{t+1} - z^{t+1} \right\|^2 \right) \right. \\ & \quad \left. - \left(\lambda R_m(x_m^t) + \langle y^t, \mathcal{D}_m x_m^t \rangle + \frac{\rho}{2} \left\| \sum_{k=1}^M \mathcal{D}_k x_k^t - z^{t+1} \right\|^2 \right) \right) + \sum_{m=1}^M \|\mathcal{D}_m(x_m^{t+1} - x_m^t)\|^2 \\ & \quad + \left(\left(l(z^{t+1}) - \langle y^t, z^{t+1} \rangle + \frac{\rho}{2} \left\| \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z^{t+1} \right\|^2 \right) \right. \\ & \quad \left. - \left(l(z^t) - \langle y^t, z^t \rangle + \frac{\rho}{2} \left\| \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z^t \right\|^2 \right) \right) \quad (\text{by Lemma 3}) \\ & = \sum_{m=1}^M (g_m(x_m^{t+1}) - g_m(x_m^t)) + (h(z^{t+1}) - h(z^t)) + \sum_{m=1}^M \|\mathcal{D}_m(x_m^{t+1} - x_m^t)\|^2 \\ & \leq \sum_{m=1}^M (\langle \nabla g_m(x_m^{t+1}), x_m^{t+1} - x_m^t \rangle - \frac{\gamma_m(\rho)}{2} \|x_m^{t+1} - x_m^t\|^2) + \langle \nabla h(z^{t+1}), z^{t+1} - z^t \rangle - \frac{\gamma(\rho)}{2} \|z^{t+1} - z^t\|^2 \\ & \quad + \sum_{m=1}^M \|\mathcal{D}_m(x_m^{t+1} - x_m^t)\|^2 \quad (\text{by strongly convexity from Assumption 1.2}) \end{aligned}$$

$$\begin{aligned}
&\leq -\sum_{m=1}^M \frac{\gamma_m(\rho)}{2} \|x_m^{t+1} - x_m^t\|^2 - \frac{\gamma(\rho)}{2} \|z^{t+1} - z^t\|^2 + \sum_{m=1}^M \|\mathcal{D}_m(x_m^{t+1} - x_m^t)\|^2 \\
&\quad \text{(by optimality condition for subproblem in (6) and (7))} \\
&\leq \sum_{m=1}^M -\left(\frac{\gamma_m(\rho)}{2} - \sigma_{\max}(\mathcal{D}_m^\top \mathcal{D}_m)\right) \|x_m^{t+1} - x_m^t\|^2 - \frac{\gamma(\rho)}{2} \|z^{t+1} - z^t\|^2. \tag{24}
\end{aligned}$$

Note that we have abused the notation $\nabla g_m(x_m)$ and denote it as the subgradient when g is non-smooth but convex. Combining (22), (23) and (24), the lemma is proved. \square

Lemma 5 Suppose Assumption 1 holds. Then the following limit exists and is bounded from below:

$$\lim_{t \rightarrow \infty} \mathcal{L}(\{x^{t+1}\}, z^{t+1}; y^{t+1}). \tag{25}$$

Proof.

$$\begin{aligned}
&\mathcal{L}(\{x^{t+1}\}, z^{t+1}; y^{t+1}) \\
&= l(z^{t+1}) + \lambda \sum_{m=1}^M R_m(x_m^{t+1}) + \langle y^{t+1}, \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z^{t+1} \rangle + \frac{\rho}{2} \|\mathcal{D}_m x_m^{t+1} - z^{t+1}\|^2 \\
&= \lambda \sum_{m=1}^M R_m(x_m^{t+1}) + l(z^{t+1}) + \langle \nabla l(z^{t+1}), \sum_{m=1}^M \mathcal{D}_m x_m^{t+1} - z^{t+1} \rangle + \frac{\rho}{2} \|\mathcal{D}_m x_m^{t+1} - z^{t+1}\|^2 \quad \text{(by Lemma 2)} \\
&\geq \lambda \sum_{m=1}^M R_m(x_m^{t+1}) + l\left(\sum_{m=1}^M \mathcal{D}_m x_m^{t+1}\right) + \frac{\rho - L}{2} \|\mathcal{D}_m x_m^{t+1} - z^{t+1}\|^2. \tag{26}
\end{aligned}$$

Combined with Assumption 1.3, $\mathcal{L}(\{x^{t+1}\}, z^{t+1}; y^{t+1})$ is lower bounded. Furthermore, by Assumption 1.2 and Lemma 4, $\mathcal{L}(\{x^{t+1}\}, z^{t+1}; y^{t+1})$ is decreasing. These complete the proof. \square

Now we are ready for the proof.

Part 1. By Assumption 1.2, Lemma 4 and Lemma 5, we have

$$\begin{aligned}
&\|x_m^{t+1} - x_m^t\| \rightarrow 0 \quad \forall m = 1, 2, \dots, M, \\
&\|z^{t+1} - z^t\| \rightarrow 0.
\end{aligned}$$

Combined with Lemma 2, we have

$$\|y^{t+1} - y^t\|^2 \rightarrow 0.$$

Combined with (8), we complete the proof for Part 1.

Part 2. Due to the fact that $\|y^{t+1} - y^t\|^2 \rightarrow 0$, by taking limit in (8), we can get (12).

At each iteration $t + 1$, by the optimality of the subproblem in (7), we have

$$\nabla l(z^{t+1}) - y^t + \rho(z^{t+1} - \sum_{m=1}^M \mathcal{D}_m x_m^{t+1}) = 0. \tag{27}$$

Combined with (12) and taking the limit, we get (11).

Similarly, by the optimality of the subproblem in (6), for $\forall m \in \mathcal{M}$ there exists $\eta_m^{t+1} \in \partial R_m(x_m^{t+1})$, such that

$$\langle x_m - x_m^{t+1}, \lambda \eta_m^{t+1} + \mathcal{D}_m^\top y^t + \rho \mathcal{D}_m^\top \left(\sum_{\substack{k=1 \\ k \leq m}}^M \mathcal{D}_k x_k^{t+1} + \sum_{\substack{k=1 \\ k > j}}^M \mathcal{D}_k x_k^t - z^t \right) \rangle \geq 0 \quad \forall x_m \in X_m. \tag{28}$$

Since R_m is convex, we have

$$\lambda R_m(x_m) - \lambda R_m(x_m^{t+1}) + \langle x - x_m^{t+1}, \mathcal{D}_m^\top y^t + \rho \left(\sum_{\substack{k=1 \\ k \leq m}}^M \mathcal{D}_k x_k^{t+1} + \sum_{\substack{k=1 \\ k > j}}^M \mathcal{D}_k x_k^t - z^t \right)^\top \mathcal{D}_j \rangle \geq 0 \quad \forall x_m \in X_m. \tag{29}$$

Combined with (12) and the fact $\|x_m^{t+1} - x_m^t\| \rightarrow 0$, by taking the limit, we get

$$\lambda R_m(x_m) - \lambda R_m(x_m^*) + \langle x - x_m^*, \mathcal{D}_m^\top y^* \rangle \geq 0 \quad \forall x_m \in X_m, \forall m, \quad (30)$$

which is equivalent to

$$\lambda R_m(x) + \langle y^*, \mathcal{D}_m x \rangle - \lambda R_m(x_m^*) - \langle y^*, \mathcal{D}_m x_m^* \rangle \geq 0 \quad \forall x \in X_m, \forall m. \quad (31)$$

And we can get the result in (9).

When $m \notin \mathcal{M}$, we have

$$\langle x_m - x_m^{t+1}, \lambda \nabla R_m(x_m^{t+1}) + \mathcal{D}_m^\top y^t + \rho \left(\sum_{\substack{k=1 \\ k \leq m}}^M \mathcal{D}_k x_k^{t+1} + \sum_{\substack{k=1 \\ k > j}}^M \mathcal{D}_k x_k^t - z^t \right)^\top \mathcal{D}_m \rangle \geq 0 \quad \forall x_m \in X_m. \quad (32)$$

Taking the limit and we can get (10).

Part 3. We first show that there exists a limit point for each of the sequences $\{x_m^t\}$, $\{z^t\}$ and $\{y^t\}$. Since $X_m, \forall m$ is compact, $\{x_m^t\}$ must have a limit point. With Theorem 1.1, we can get that $\{z^t\}$ is also compact and has a limit point. Furthermore, with Lemma 2, we can get $\{y^t\}$ is also compact and has a limit point.

We prove Part 3 by contradiction. Since $\{x_m^t\}$, $\{z^t\}$ and $\{y^t\}$ lie in some compact set, there exists a subsequence $\{x_m^{t_k}\}$, $\{z^{t_k}\}$ and $\{y^{t_k}\}$, such that

$$(\{x_m^{t_k}\}, z^{t_k}; y^{t_k}) \rightarrow (\{\hat{x}_m\}, \hat{z}; \hat{y}), \quad (33)$$

where $(\{\hat{x}_m\}, \hat{z}; \hat{y})$ is some limit point and by part 2, we have $(\{\hat{x}_m\}, \hat{z}; \hat{y}) \in Z^*$. Suppose that $\{\{x_m^t\}, z^t; y^t\}$ does not converge to Z^* , since $(\{x_m^{t_k}\}, z^{t_k}; y^{t_k})$ is a subsequence of it, there exists some $\gamma > 0$, such that

$$\lim_{k \rightarrow \infty} \text{dist}((\{x_m^{t_k}\}, z^{t_k}; y^{t_k}); Z^*) = \gamma > 0. \quad (34)$$

From (33), there exists some $J(\gamma) > 0$, such that

$$\|(\{x_m^{t_k}\}, z^{t_k}; y^{t_k}) - (\{\hat{x}_m\}, \hat{z}; \hat{y})\| \leq \frac{\gamma}{2}, \quad \forall k \geq J(\gamma). \quad (35)$$

Since $(\{\hat{x}_m\}, \hat{z}; \hat{y}) \in Z^*$, we have

$$\text{dist}((\{x_m^{t_k}\}, z^{t_k}; y^{t_k}); Z^*) \leq \text{dist}((\{x_m^{t_k}\}, z^{t_k}; y^{t_k}); (\{\hat{x}_m\}, \hat{z}; \hat{y})). \quad (36)$$

From the above two inequalities, we must have

$$\text{dist}((\{x_m^{t_k}\}, z^{t_k}; y^{t_k}); Z^*) \leq \frac{\gamma}{2}, \quad \forall k \geq J(\gamma), \quad (37)$$

which contradicts to (34), completing the proof. \square

7.2 Proof of Theorem 2

We first show an upper bound for V^t .

1. Bound for $\tilde{\nabla}_{x_m} \mathcal{L}(\{x_m^t\}, z^t; y^t)$. When $m \in \mathcal{M}$, from the optimality condition in (6), we have

$$0 \in \lambda \partial_{x_m} R_j(x_m^{t+1}) + \mathcal{D}^\top y^t + \rho \mathcal{D}_j^\top \left(\sum_{\substack{k=1 \\ k \neq j}}^M \mathcal{D}_k x_k^t + \mathcal{D}_m x_m^{t+1} - z^t \right).$$

By some rearrangement, we have

$$(x_m^{t+1} - \mathcal{D}^\top y^t - \rho \mathcal{D}_m^\top \left(\sum_{\substack{k=1 \\ k \neq j}}^M \mathcal{D}_k x_k^t + \mathcal{D}_m x_m^{t+1} - z^t \right)) - x_m^{t+1} \in \lambda \partial_{x_m} R_m(x_m^{t+1}),$$

which is equivalent to

$$x_m^{t+1} = \text{prox}_{\lambda R_m} [x_m^{t+1} - \mathcal{D}^\top y^t - \rho \mathcal{D}_m^\top (\sum_{\substack{k=1 \\ k \neq j}}^M \mathcal{D}_k x_k^t + \mathcal{D}_m x_m^{t+1} - z^t)]. \quad (38)$$

Therefore,

$$\begin{aligned} & \|x_m^t - \text{prox}_{\lambda R_m} [x_m^t - \nabla_{x_m} (\mathcal{L}(\{x_m^t\}, z^t; y^t) - \lambda \sum_{m=1}^M R_m(x_m^t))]\| \\ &= \|x_m^t - x_m^{t+1} + x_m^{t+1} - \text{prox}_{\lambda R_m} [x_m^t - \mathcal{D}^\top y^t - \rho \mathcal{D}_m^\top (\sum_{k=1}^M \mathcal{D}_k x_k^t - z^t)]\| \\ &\leq \|x_m^t - x_m^{t+1}\| + \|\text{prox}_{\lambda R_m} [x_m^{t+1} - \mathcal{D}^\top y^t - \rho \mathcal{D}_m^\top (\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k x_k^t + \mathcal{D}_m x_m^{t+1} - z^t)] \\ &\quad - \text{prox}_{\lambda R_m} [x_m^t - \mathcal{D}^\top y^t - \rho \mathcal{D}_m^\top (\sum_{k=1}^M \mathcal{D}_k x_k^t - z^t)]\| \\ &\leq 2\|x_m^t - x_m^{t+1}\| + \rho \|\mathcal{D}_m^\top \mathcal{D}_m (x_m^{t+1} - x_m^t)\|. \end{aligned} \quad (39)$$

When $m \notin \mathcal{M}$, similarly, we have

$$\lambda \nabla_{x_m} R_m(x_m^{t+1}) + \mathcal{D}^\top y^t + \rho \mathcal{D}_m^\top (\sum_{\substack{k=1 \\ k \neq j}}^M \mathcal{D}_k x_k^t + \mathcal{D}_m x_m^{t+1} - z^t) = 0. \quad (40)$$

Therefore,

$$\begin{aligned} & \|\nabla_{x_m} \mathcal{L}(\{x_m^t\}, z^t; y^t)\| \\ &= \|\lambda \nabla_{x_m} R_m(x_m^t) + \mathcal{D}^\top y^t + \rho \mathcal{D}_m^\top (\sum_{k=1}^M \mathcal{D}_k x_k^t - z^t)\| \\ &= \|\lambda \nabla_{x_m} R_m(x_m^t) + \mathcal{D}^\top y^t + \rho \mathcal{D}_m^\top (\sum_{k=1}^M \mathcal{D}_k x_k^t - z^t) \\ &\quad - (\lambda \nabla_{x_m} R_m(x_m^{t+1}) + \mathcal{D}^\top y^t + \rho \mathcal{D}_m^\top (\sum_{\substack{k=1 \\ k \neq j}}^M \mathcal{D}_k x_k^t + \mathcal{D}_m x_m^{t+1} - z^t))\| \\ &\leq \lambda \|\nabla_{x_m} R_m(x_m^t) - \nabla_{x_m} R_m(x_m^{t+1})\| + \rho \|\mathcal{D}_m^\top \mathcal{D}_m (x_m^{t+1} - x_m^t)\| \\ &\leq L_m \|x_m^{t+1} - x_m^t\| + \rho \|\mathcal{D}_m^\top \mathcal{D}_m (x_m^{t+1} - x_m^t)\|. \quad (\text{by Assumption 1.4}) \end{aligned} \quad (41)$$

2. Bound for $\|\nabla_z \mathcal{L}(\{x_m^t\}, z^t; y^t)\|$. By optimality condition in (7), we have

$$\nabla l(z^{t+1}) - y^t + \rho(z^{t+1} - \sum_{m=1}^M \mathcal{D}_m x_m^{t+1}) = 0.$$

Therefore

$$\begin{aligned} & \|\nabla_z \mathcal{L}(\{x_m^t\}, z^t; y^t)\| \\ &= \|l(z^t) - y^t + \rho(z^t - \sum_{m=1}^M \mathcal{D}_m x_m^t)\| \\ &= \|l(z^t) - y^t + \rho(z^t - \sum_{m=1}^M \mathcal{D}_m x_m^t) - (l(z^{t+1}) - y^t + \rho(z^{t+1} - \sum_{m=1}^M \mathcal{D}_m x_m^{t+1}))\| \\ &\leq (L + \rho) \|z^{t+1} - z^t\| + \rho \sum_{m=1}^M \|\mathcal{D}_m (x_m^{t+1} - x_m^t)\|. \end{aligned} \quad (42)$$

3. Bound for $\|\sum_{m=1}^M \mathcal{D}_m x_m^t - z^t\|$. According to Lemma 2, we have

$$\left\| \sum_{m=1}^M \mathcal{D}_m x_m^t - z^t \right\| = \frac{1}{\rho} \|y^{t+1} - y^t\| \leq \frac{L}{\rho} \|z^{t+1} - z^t\|. \quad (43)$$

Combining (39), (41), (42) and (43), we can conclude that there exists some $C_1 > 0$, such that

$$V^t \leq C_1 (\|z^{t+1} - z^t\|^2 + \sum_{m=1}^M \|x_m^{t+1} - x_m^t\|^2), \quad (44)$$

By Lemma 4, there exists some constant $C_2 = \min\{\sum_{m=1}^M \frac{\gamma_m(\rho)}{2}, \frac{\gamma(\rho)}{2} - \frac{L^2}{\rho}\}$, such that

$$\begin{aligned} & \mathcal{L}(\{x_m^t\}, z^t; y^t) - \mathcal{L}(\{x_m^{t+1}\}, z^{t+1}; y^{t+1}) \\ & \geq C_2 (\|z^{t+1} - z^t\|^2 + \sum_{m=1}^M \|x_m^{t+1} - x_m^t\|^2). \end{aligned} \quad (45)$$

By (44) and (45), we have

$$V^t \leq \frac{C_1}{C_2} \mathcal{L}(\{x_m^t\}, z^t; y^t) - \mathcal{L}(\{x_m^{t+1}\}, z^{t+1}; y^{t+1}). \quad (46)$$

Taking the sum over $t = 1, \dots, T$, we have

$$\begin{aligned} \sum_{t=1}^T V^t & \leq \frac{C_1}{C_2} \mathcal{L}(\{x^1\}, z^1; y^1) - \mathcal{L}(\{x^{T+1}\}, z^{T+1}; y^{T+1}) \\ & \leq \frac{C_1}{C_2} (\mathcal{L}(\{x^1\}, z^1; y^1) - \underline{f}). \end{aligned} \quad (47)$$

By the definition of $T(\epsilon)$, we have

$$T(\epsilon)\epsilon \leq \frac{C_1}{C_2} (\mathcal{L}(\{x^1\}, z^1; y^1) - \underline{f}). \quad (48)$$

By taking $C = \frac{C_1}{C_2}$, we complete the proof. \square

7.3 Proof of Lemma 1

From the optimality condition of the x update procedure in (16), we can get

$$\begin{aligned} \mathcal{D}_m x_{m, \mathcal{D}_m}^{t+1} &= -\mathcal{D}_m (\rho \mathcal{D}_m^\top \mathcal{D}_m)^{-1} \left[\lambda R'_m(x_{m, \mathcal{D}_m}^{t+1}) + \mathcal{D}_m^\top y^t + \rho \mathcal{D}_m^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right], \\ \mathcal{D}'_m x_{m, \mathcal{D}'_m}^{t+1} &= -\mathcal{D}'_m (\rho \mathcal{D}'_m{}^\top \mathcal{D}'_m)^{-1} \left[\lambda R'_m(x_{m, \mathcal{D}'_m}^{t+1}) + \mathcal{D}'_m{}^\top y^t + \rho \mathcal{D}'_m{}^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right]. \end{aligned}$$

Therefore we have

$$\begin{aligned}
& \mathcal{D}_m x_{m, \mathcal{D}_m}^{t+1} - \mathcal{D}'_m x_{m, \mathcal{D}'_m}^{t+1} \\
&= -\mathcal{D}_m (\rho \mathcal{D}_m^\top \mathcal{D}_m)^{-1} \left[\lambda R'_m(x_{m, \mathcal{D}_m}^{t+1}) + \mathcal{D}_m^\top y^t \mathcal{D}_m + \rho \mathcal{D}_m^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right] \\
&\quad + \mathcal{D}'_m (\rho \mathcal{D}'_m{}^\top \mathcal{D}'_m)^{-1} \left[\lambda R'_m(x_{m, \mathcal{D}'_m}^{t+1}) + \mathcal{D}'_m{}^\top y^t + \rho \mathcal{D}'_m{}^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right] \\
&= \mathcal{D}_m (\rho \mathcal{D}_m^\top \mathcal{D}_m)^{-1} \\
&\quad \times \left[\lambda (R'_m(x_{m, \mathcal{D}'_m}^{t+1}) - R'_m(x_{m, \mathcal{D}_m}^{t+1})) + (\mathcal{D}'_m - \mathcal{D}_m)^\top y^t + \rho (\mathcal{D}'_m - \mathcal{D}_m)^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right] \\
&\quad + [\mathcal{D}'_m (\rho \mathcal{D}'_m{}^\top \mathcal{D}'_m)^{-1} - \mathcal{D}_m (\rho \mathcal{D}_m^\top \mathcal{D}_m)^{-1}] \\
&\quad \times \left(\lambda R'_m(x_{m, \mathcal{D}'_m}^{t+1}) + \mathcal{D}'_m{}^\top y^t + \rho \mathcal{D}'_m{}^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right).
\end{aligned}$$

Denote

$$\begin{aligned}
\Phi_1 &= \mathcal{D}_m (\rho \mathcal{D}_m^\top \mathcal{D}_m)^{-1} \\
&\quad \times \left[\lambda (R'_m(x_{m, \mathcal{D}'_m}^{t+1}) - R'_m(x_{m, \mathcal{D}_m}^{t+1})) + (\mathcal{D}'_m - \mathcal{D}_m)^\top y^t + \rho (\mathcal{D}'_m - \mathcal{D}_m)^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right], \\
\Phi_2 &= [\mathcal{D}'_m (\rho \mathcal{D}'_m{}^\top \mathcal{D}'_m)^{-1} - \mathcal{D}_m (\rho \mathcal{D}_m^\top \mathcal{D}_m)^{-1}] \\
&\quad \times \left(\lambda R'_m(x_{m, \mathcal{D}'_m}^{t+1}) + \mathcal{D}'_m{}^\top y^t + \rho \mathcal{D}'_m{}^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right).
\end{aligned}$$

As a result:

$$\mathcal{D}_m x_{m, \mathcal{D}_m}^{t+1} - \mathcal{D}'_m x_{m, \mathcal{D}'_m}^{t+1} = \Phi_1 + \Phi_2. \quad (49)$$

In the following, we will analyze the components in (49) term by term. The object is to prove

$\max_{\substack{\mathcal{D}_m, \mathcal{D}'_m \\ \|\mathcal{D}_m - \mathcal{D}'_m\| \leq 1}} \|x_{m, \mathcal{D}_m}^{t+1} - x_{m, \mathcal{D}'_m}^{t+1}\|$ is bounded. To see this, notice that

$$\begin{aligned}
& \max_{\substack{\mathcal{D}_m, \mathcal{D}'_m \\ \|\mathcal{D}_m - \mathcal{D}'_m\| \leq 1}} \|\mathcal{D}_m x_{m, \mathcal{D}_m}^{t+1} - \mathcal{D}'_m x_{m, \mathcal{D}'_m}^{t+1}\| \\
&\leq \max_{\substack{\mathcal{D}_m, \mathcal{D}'_m \\ \|\mathcal{D}_m - \mathcal{D}'_m\| \leq 1}} \|\Phi_1\| + \max_{\substack{\mathcal{D}_m, \mathcal{D}'_m \\ \|\mathcal{D}_m - \mathcal{D}'_m\| \leq 1}} \|\Phi_2\|.
\end{aligned}$$

For $\max_{\substack{\mathcal{D}_m, \mathcal{D}'_m \\ \|\mathcal{D}_m - \mathcal{D}'_m\| \leq 1}} \|\Phi_2\|$, from assumption 2.3, we have

$$\begin{aligned}
& \max_{\substack{\mathcal{D}_m, \mathcal{D}'_m \\ \|\mathcal{D}_m - \mathcal{D}'_m\| \leq 1}} \|\Phi_2\| \\
&\leq \left\| \frac{2}{d_m \rho} \left(\lambda R'_m(x_{m, \mathcal{D}'_m}^{t+1}) + \mathcal{D}'_m{}^\top y^t + \rho \mathcal{D}'_m{}^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right) \right\|.
\end{aligned}$$

By mean value theorem, we have

$$\begin{aligned} & \left\| \frac{2}{d_m \rho} \left(\lambda \mathcal{D}_m'^\top R_m''(x_*) + \mathcal{D}_m'^\top y^t + \rho \mathcal{D}_m'^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right) \right\| \\ & \leq \frac{2}{d_m \rho} \left[\lambda \|R_m''(\cdot)\| + \|y^t\| + \rho \left\| \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right\| \right]. \end{aligned}$$

For $\max_{\substack{\mathcal{D}_m, \mathcal{D}_m' \\ \|\mathcal{D}_m - \mathcal{D}_m'\| \leq 1}} \|\Phi_1\|$, we have

$$\begin{aligned} & \max_{\substack{\mathcal{D}_m, \mathcal{D}_m' \\ \|\mathcal{D}_m - \mathcal{D}_m'\| \leq 1}} \|\Phi_1\| \leq \left\| \mathcal{D}_m (\rho \mathcal{D}_m^\top \mathcal{D}_m)^{-1} \right. \\ & \quad \times \left[\lambda (R_m'(x_{m, \mathcal{D}_m'}^{t+1}) - R_m'(x_{m, \mathcal{D}_m}^{t+1})) + (\mathcal{D}_m' - \mathcal{D}_m)^\top y^t + \rho (\mathcal{D}_m' - \mathcal{D}_m)^\top \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right] \left. \right\| \\ & \leq \rho^{-1} \|(\mathcal{D}_m^\top \mathcal{D}_m)^{-1}\| \left[\lambda \|R_m''(\cdot)\| + \|y^t\| + \rho \left\| \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right)^\top \right\| \right] \\ & = \frac{1}{d_m \rho} \left[\lambda \|R_m''(\cdot)\| + \|y^t\| + \rho \left\| \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right) \right\| \right]. \end{aligned}$$

Thus by assumption 2.1-2.2

$$\begin{aligned} & \max_{\substack{\mathcal{D}_m, \mathcal{D}_m' \\ \|\mathcal{D}_m - \mathcal{D}_m'\| \leq 1}} \|\mathcal{D}_m x_{m, \mathcal{D}_m}^{t+1} - \mathcal{D}_m' x_{m, \mathcal{D}_m'}^{t+1}\| \\ & \leq \frac{3}{d_m \rho} \left[\lambda c_1 + \|y^t\| + \rho \left\| \left(\sum_{\substack{k=1 \\ k \neq m}}^M \mathcal{D}_k \tilde{x}_k - z \right)^\top \right\| \right] \\ & \leq \frac{3}{d_m \rho} \left[\lambda c_1 + \|y^t\| + \rho \|z\| + \rho \sum_{\substack{k=1 \\ k \neq m}}^M \|\tilde{x}_k\| \right] \\ & \leq \frac{3}{d_m \rho} [\lambda c_1 + (1 + M\rho)b_1] \end{aligned}$$

is bounded. \square

7.4 Proof of Theorem 3

Proof: The privacy loss from $\mathcal{D}_m \tilde{x}_m^{t+1}$ is calculated by:

$$\left| \ln \frac{P(\mathcal{D}_m \tilde{x}_m^{t+1} | \mathcal{D}_m)}{P(\mathcal{D}_m' \tilde{x}_m^{t+1} | \mathcal{D}_m')} \right| = \left| \ln \frac{P(\mathcal{D}_m \tilde{x}_m^{t+1} + \mathcal{D}_m \xi_m^{t+1})}{P(\mathcal{D}_m' \tilde{x}_m^{t+1} + \mathcal{D}_m' \xi_m^{t+1})} \right| = \left| \ln \frac{P(\mathcal{D}_m \xi_m^{t+1})}{P(\mathcal{D}_m' \xi_m^{t+1})} \right|.$$

Since $\mathcal{D}_m \xi_m^{t+1}$ and $\mathcal{D}'_m \xi_m^{t+1}$ are sampled from $\mathcal{N}(0, \sigma_{m,t+1}^2)$, combine with lemma 1, we have

$$\begin{aligned}
& \left| \ln \frac{P(\mathcal{D}_m \xi_m^{t+1})}{P(\mathcal{D}'_m \xi_m^{t+1})} \right| \\
&= \left| \frac{2\xi_m^{t+1} \|\mathcal{D}_m x_{m,\mathcal{D}_m}^{t+1} - \mathcal{D}'_m x_{m,\mathcal{D}'_m}^{t+1}\| + \|\mathcal{D}_m x_{m,\mathcal{D}_m}^{t+1} - \mathcal{D}'_m x_{m,\mathcal{D}'_m}^{t+1}\|^2}{2\sigma_{m,t+1}^2} \right| \\
&\leq \left| \frac{2\mathcal{D}_m \xi_m^{t+1} \mathbb{C} + \mathbb{C}^2}{2 \frac{\mathbb{C}^2 \cdot 2\ln(1.25/\sigma)}{\varepsilon^2}} \right| \\
&= \left| \frac{(2\mathcal{D}_m \xi_m^{t+1} + \mathbb{C})\varepsilon^2}{4\mathbb{C}\ln(1.25/\sigma)} \right|.
\end{aligned}$$

In order to make $\left| \frac{(2\mathcal{D}_m \xi_m^{t+1} + \mathbb{C})\varepsilon^2}{4\mathbb{C}\ln(1.25/\sigma)} \right| \leq \varepsilon$, we need to make sure

$$|\mathcal{D}_m \xi_m^{t+1}| \leq \frac{2\mathbb{C}\ln(1.25/\sigma)}{\varepsilon} - \frac{\mathbb{C}}{2}.$$

In the following, we need to proof

$$P(|\mathcal{D}_m \xi_m^{t+1}| \geq \frac{2\mathbb{C}\ln(1.25/\sigma)}{\varepsilon} - \frac{\mathbb{C}}{2}) \leq \delta \quad (50)$$

holds. However, we will proof a stronger result that lead to (50). Which is

$$P(\mathcal{D}_m \xi_m^{t+1} \geq \frac{2\mathbb{C}\ln(1.25/\sigma)}{\varepsilon} - \frac{\mathbb{C}}{2}) \leq \frac{\delta}{2}.$$

Since the tail bound of normal distribution $\mathcal{N}(0, \sigma_{m,t+1}^2)$ is:

$$P(\mathcal{D}_m \xi_m^{t+1} > r) \leq \frac{\sigma_{m,t+1}}{r\sqrt{2\pi}} e^{-\frac{r^2}{2\sigma_{m,t+1}^2}}.$$

Let $r = \frac{2\mathbb{C}\ln(1.25/\sigma)}{\varepsilon} - \frac{\mathbb{C}}{2}$, we then have

$$\begin{aligned}
& P(\mathcal{D}_m \xi_m^{t+1} \geq \frac{2\mathbb{C}\ln(1.25/\sigma)}{\varepsilon} - \frac{\mathbb{C}}{2}) \\
&\leq \frac{\mathbb{C}\sqrt{2\ln(1.25/\sigma)}}{r\sqrt{2\pi}\varepsilon} \exp\left[-\frac{[4\ln(1.25/\sigma) - \varepsilon]^2}{8\ln(1.25/\sigma)}\right].
\end{aligned}$$

When δ is small and let $\varepsilon \leq 1$, we then have

$$\frac{\sqrt{2\ln(1.25/\sigma)}2}{(4\ln(1.25/\sigma) - \varepsilon)\sqrt{2\pi}} \leq \frac{\sqrt{2\ln(1.25/\sigma)}2}{(4\ln(1.25/\sigma) - 1)\sqrt{2\pi}} < \frac{1}{\sqrt{2\pi}}. \quad (51)$$

As a result, we can proof that

$$-\frac{[4\ln(1.25/\sigma) - \varepsilon]^2}{8\ln(1.25/\sigma)} < \ln(\sqrt{2\pi}\frac{\delta}{2})$$

by equation (51). Thus we have

$$P(\mathcal{D}_m \xi_m^{t+1} \geq \frac{2\mathbb{C}\ln(1.25/\sigma)}{\varepsilon} - \frac{\mathbb{C}}{2}) < \frac{1}{\sqrt{2\pi}} \exp(\ln(\sqrt{2\pi}\frac{\delta}{2})) = \frac{\delta}{2}.$$

Thus we proved (50) holds. Define

$$\begin{aligned}
\mathbb{A}_1 &= \{\mathcal{D}_m \xi_m^{t+1} : |\mathcal{D}_m \xi_m^{t+1}| \leq \frac{1}{\sqrt{2\pi}} \exp(\ln(\sqrt{2\pi}\frac{\delta}{2}))\}, \\
\mathbb{A}_2 &= \{\mathcal{D}_m \xi_m^{t+1} : |\mathcal{D}_m \xi_m^{t+1}| > \frac{1}{\sqrt{2\pi}} \exp(\ln(\sqrt{2\pi}\frac{\delta}{2}))\}.
\end{aligned}$$

Thus we obtain the desired result:

$$\begin{aligned}
& P(\mathcal{D}'_m \tilde{x}_m^{t+1} | \mathcal{D}_m) \\
&= P(\mathcal{D}_m x_{m, \mathcal{D}_m}^{t+1} + \mathcal{D}_m \xi_m^{t+1} : \mathcal{D}_m \xi_m^{t+1} \in \mathbb{A}_1) \\
&+ P(\mathcal{D}_m x_{m, \mathcal{D}_m}^{t+1} + \mathcal{D}_m \xi_m^{t+1} : \mathcal{D}_m \xi_m^{t+1} \in \mathbb{A}_2) \\
&< e^\varepsilon P(\mathcal{D}_m x_{m, \mathcal{D}'_m}^{t+1} + \mathcal{D}_m \xi_m^{t, t+1}) + \delta = e^\varepsilon P(\mathcal{D}_m \tilde{x}_m^{t+1} | \mathcal{D}'_m) + \delta.
\end{aligned}$$

□

7.5 Implementation

The implementation of the prototype system can be found at https://www.dropbox.com/sh/imfz3k2fhf3zkrc/AACcbREvj_PBSjEUdZmBfoFUa?dl=0.