Recycled ADMM: Improve Privacy and Accuracy with Less Computation in Distributed Algorithms

Xueru Zhang, Mohammad Mahdi Khalili, Mingyan Liu

Abstract—Alternating direction method of multiplier (ADMM) is a powerful method to solve decentralized convex optimization problems. In distributed settings, each node performs computation with its local data and the local results are exchanged among neighboring nodes in an iterative fashion. During this iterative process the leakage of data privacy arises and can accumulate significantly over many iterations, making it difficult to balance the privacy-utility tradeoff. In this study we propose Recycled ADMM (R-ADMM), where a linear approximation is applied to every even iteration, its solution directly calculated using only results from the previous, odd iteration. It turns out that under such a scheme, half of the updates incur no privacy loss and require much less computation compared to the conventional ADMM. We obtain a sufficient condition for the convergence of R-ADMM and provide the privacy analysis based on objective perturbation.

I. INTRODUCTION

Distributed optimization and learning are crucial for many settings where the data is possessed by multiple parties or when the quantity of data prohibits processing at a central location [1]–[5]. Many problems can be formulated as a convex optimization of the following form: \( \min_x \sum_{i=1}^N f_i(x) \). In a distributed setting, each entity/node \( i \) has its own local objective \( f_i \), \( N \) entities/nodes collaboratively work to solve this objective through an interactive process of local computation and message passing. At the end all local results should ideally converge to the global optimum.

The information exchanged over the iterative process gives rise to privacy concerns if the local training data contains sensitive information such as medical or financial records, web search history, and so on. It is therefore highly desirable to ensure such iterative processes are privacy-preserving. We adopt the \( \varepsilon \)-differential privacy to measure such privacy guarantee; it is generally achieved by perturbing the algorithm such that the probability distribution of its output is relatively insensitive to any change to a single record in the input [6].

Existing approaches to decentralizing the above problem primarily consist of subgradient-based algorithms [7]–[10] and ADMM-based algorithms [11]–[21]. It has been shown that ADMM-based algorithms can converge at the rate of \( O(\frac{1}{k}) \) while subgradient-based algorithms typically converge at the rate of \( O(\frac{1}{k^2}) \), where \( k \) is the number of iterations [16]. In this study, we will solely focus on ADMM-based algorithms. While a number of differentially private (sub)gradient-based distributed algorithms have been proposed [22]–[25], the same is much harder for ADMM-based algorithms due to its computational complexity stemming from the fact that each node is required to solve an optimization problem in each iteration. To the best of our knowledge, only [26]–[28] apply differential privacy to ADMM. In particular, [26] proposed the dual/primal variable perturbation method to inspect the privacy loss of one node in every single iteration; this, however, is not sufficient for guaranteeing privacy as an adversary can potentially use the revealed results from all iterations to perform inference. In [27] we address this issue by inspecting the total privacy loss over the entire process and the entire network; we proposed a penalty perturbation method which improves the privacy-utility tradeoff significantly. In the recent work [28], a first-order approximation is applied to the augmented Lagrangian in all iterations; however, this method requires a central server to average all updated primal variables over the network in each iteration.

In this study we present Recycled ADMM (R-ADMM), a modified version of ADMM where the privacy leakage only happens during half of the updates. Specifically, we adopt a linearized approximated optimization in every even iteration, whose solution can actually be calculated directly from results in the previous, odd iteration, and is used for updating primal variable. We establish a sufficient condition for convergence and provide a privacy analysis using the objective perturbation method. Our numerical results show that the privacy-utility tradeoff can be improved significantly.

The remainder of the paper is organized as follows. We present problem formulation and definition of differential privacy and ADMM in Section II and the Recycled ADMM algorithm along with its convergence analysis in Section III. A private version of this ADMM algorithm is then introduced in Section IV and numerical results in Section V. Section VI concludes the paper.

II. PRELIMINARIES

A. Problem Formulation

Consider a connected network given by an undirected graph \( G(\mathcal{N}, \mathcal{E}) \), which consists of a set of nodes \( \mathcal{N} = \{1, 2, \cdots, N\} \) and a set of edges \( \mathcal{E} = \{1, 2, \cdots, E\} \). Two nodes can exchange information if and only if they are connected by an edge. Let \( \mathcal{N}_i \) denote node \( i \)’s set of neighbors, excluding itself. A node \( i \) has a dataset \( D_i = \{(x^n_i, y^n_i) | n = 1, 2, \cdots, B_i\} \), where \( x^n_i \in \mathbb{R}^d \) is the feature

\footnote{A connected network is one in which every node is reachable (via a path) from every other node.}
vector representing the $n$-th sample belonging to $i$, $y_i \in \{-1, 1\}$ the corresponding label, and $B_i$ the size of $D_i$.

Consider the regularized empirical risk minimization (ERM) problem for binary classification defined as follows:

$$\min_{f_c} O_{ERM}(f_c, D_{all}) = \sum_{i=1}^{N} \frac{C}{B_i} \sum_{a=1}^{B_i} \mathcal{L}(y_i^a f_c^a x_i^a) + \rho R(f_c)$$  \hspace{1cm} (1)

where $C \leq B_i$ and $\rho > 0$ are constant parameters of the algorithm, the loss function $\mathcal{L}(\cdot)$ measures the accuracy of the classifier, and the regularizer $R(\cdot)$ helps prevent overfitting. The goal is to train a (centralized) classifier $f_c \in \mathbb{R}^d$ over the union of all local datasets $D_{all} = \cup_{i \in \mathcal{N}} D_i$ in a distributed manner using ADMM, while providing privacy guarantee for each data sample.

B. Differential Privacy [6]

A randomized algorithm $\mathcal{A}(\cdot)$ taking a dataset as input satisfies $\varepsilon$-differential privacy if for any two datasets $D, D'$ differing in at most one data point, and for any set of possible outputs $S \subseteq \text{range} \mathcal{A}(\cdot)$, $Pr(\mathcal{A}(D) \in S) \leq e^{\varepsilon} Pr(\mathcal{A}(D') \in S)$ holds. We call two datasets differing in at most one data point as neighboring datasets. The above definition suggests that for a sufficiently small $\varepsilon$, an adversary will observe almost the same output regardless of the presence (or value change) of any one individual in the dataset; this is what provides privacy protection for that individual.

C. Conventional ADMM

To decentralize (1), let $f_i$ be the local classifier of each node $i$. To achieve consensus, i.e., $f_1 = f_2 = \cdots = f_N$, a set of auxiliary variables $\{w_{ij} | i \in \mathcal{N}, j \in \mathcal{Y}_j\}$ are introduced for every pair of connected nodes. As a result, (1) is reformulated equivalently as:

$$\min_{\{f_i, \{w_{ij}\}\}} \hat{O}_{ERM}(\{f_i\}_{i=1}^N, D_{all}) = \sum_{i=1}^{N} O(f_i, D_i)$$  \hspace{1cm} (2)

s.t. $O(f_i, D_i) = \frac{C}{B_i} \sum_{n=1}^{B_i} \mathcal{L}(y^n_{i} f^n_i x^n_i) + \frac{\rho}{N} R(f_i)$.

The objective in (2) can be solved using ADMM with the augmented Lagrangian:

$$L_{\eta}(\{f_i\}, \{w_{ij}, \lambda^k_{ij}\}) = \sum_{i=1}^{N} O(f_i, D_i)$$

$$+ \sum_{i=1}^{N} \sum_{j \in \mathcal{Y}_j} (\lambda^k_{ij})^T (f_i - w_{ij}) + \sum_{i=1}^{N} \sum_{j \in \mathcal{Y}_j} (\lambda^k_{ij})^T (w_{ij} - f_j)$$

$$+ \sum_{i=1}^{N} \sum_{j \in \mathcal{Y}_j} \frac{\eta}{2} (||f_i - w_{ij}||_2^2 + ||w_{ij} - f_j||_2^2).$$

In the $(t+1)$-th iteration, the ADMM updates consist of the following:

$$f_i(t+1) = \arg\min_{f_i} L_{\eta}(\{f_i\}, \{w_{ij}(t), \lambda^k_{ij}(t)\}) ;$$  \hspace{1cm} (4)

$$w_{ij}(t+1) = \arg\min_{w_{ij}} L_{\eta}(\{f_i(t+1)\}, \{w_{ij}, \lambda^k_{ij}(t)\}) ;$$  \hspace{1cm} (5)

$$\lambda^k_{ij}(t+1) = \lambda^k_{ij}(t) + \eta (f_i(t+1) - w_{ij}(t+1)) ;$$  \hspace{1cm} (6)

$$\lambda_0^k_{ij}(t+1) = \lambda_0^k_{ij}(t) + \eta (w_{ij}(t+1) - f_j(t+1)).$$  \hspace{1cm} (7)

Using Lemma 3 in [29], if dual variables $\lambda^k_{ij}(t)$ and $\lambda_0^k_{ij}(t)$ are initialized to zero for all node pairs $(i, j)$, $\lambda^k_{ij}(t)$ and $\lambda_0^k_{ij}(t)$ will hold for all iterations with $k \in \{a, b\}, i \in \mathcal{N}, j \in \mathcal{Y}_j$. Let $\lambda_i(t) = \sum_{j \in \mathcal{Y}_j} \lambda^k_{ij}(t)$, then the ADMM iterations (4)-(7) can be simplified as (Refer to Appendix A in [27] for proof):

$$f_i(t+1) = \arg\min_{f_i} \{O(f_i, D_i) + 2\lambda_i(t)^T f_i$$

$$+ \eta \sum_{j \in \mathcal{Y}_j} (\frac{1}{2} (f_i(t) + f_j(t)) - f_i(t))^2 \} ;$$  \hspace{1cm} (8)

$$\lambda_i(t+1) = \lambda_i(t) + \eta \sum_{j \in \mathcal{Y}_j} (f_i(t+1) - f_j(t+1)) .$$  \hspace{1cm} (9)

D. Private ADMM [26] & Private M-ADMM [27]

In private ADMM [26], the noise is added either to the updated primal variable before broadcasting to its neighbors (primal variable perturbation), or to the dual variable before updating its primal variable using (8) (dual variable perturbation). The privacy property is only evaluated for a single node and a single iteration, both methods cannot balance the privacy-utility tradeoff very well if consider the total privacy loss. In [27] the total privacy loss of the whole network over the entire iterative process is considered. A modified ADMM (M-ADMM) was proposed to improve the privacy-utility tradeoff, specifically, it explores the rule of step-size (penalty parameter) in stabilizing the algorithm. M-ADMM allows each node to independently determine its penalty parameter, by perturbing the algorithm with noise correlated to penalty parameter and at the same time increasing the penalty parameters, the privacy and accuracy can be improved simultaneously.

E. Main idea

Fundamentally, the accumulation of privacy loss over iterations stems from the fact that the raw data is used in every primal update. If the updates can be made without using the raw data, but only from computational results that already exist, then the privacy loss originating from these updates will be zero, while at the same time the computational cost be reduced significantly. Based on this idea, we start with modifying ADMM such that we can repeatedly use some computational results to make updates.

III. RECYCLED ADMM (R-ADMM)

A. Making information recyclable

ADMM can outperform gradient-based methods in terms of requiring fewer number of iterations for convergence; this
however comes at the price of high computational cost in every iteration. In particular, the primal variable is updated by performing an optimization in each iteration. In [17], [30], [31], either a linear or quadratic approximation of the objective function is used to obtain an inexact solution in each iteration in lieu of solving the original optimization problem. While this clearly lowers the computational cost, the approximate computation is performed using the local, raw data in every iteration, which means that privacy loss inevitably accumulates over the iterations.

We begin by modifying ADMM in such a way that in every even iteration, without using the raw data, the primal variable is updated solely based on the existing computational results from the previous, odd iteration. Compared with conventional ADMM, these updates incur no privacy loss and less computation. Since the computational results are repeatedly used, this method will be referred to as Recycled ADMM (R-ADMM).

Specifically, in the $2k$-th (even) iteration, we approximate $\mathcal{O}(f_i, D_i)$ (Eqn. (8), primal update optimization) by $\mathcal{O}(f_i, D_i) \approx \mathcal{O}(f_i(2k-1), D_i) + \nabla \mathcal{O}(f_i(2k-1), D_i)^T (f_i - f_i(2k-1)) + \frac{\eta}{2} \|f_i - f_i(2k-1)\|^2$ $(\gamma \geq 0)$ and update only the primal variables. Using the first-order condition, the updates in the $2k$-th iteration become:

$$f_i(2k) = f_i(2k-1) - \frac{1}{2\eta V_i + \gamma} \{ \nabla \mathcal{O}(f_i(2k-1), D_i) \\ + 2\lambda_i(2k-1) + \eta \sum_{j \in V_i} (f_i(2k-1) - f_j(2k-1)) \} ; \quad (10)$$

$$\lambda_i(2k-1) = \lambda_i(2k-1) \cdot (11)$$

In the $(2k-1)$-th (odd) iteration, the updates are kept the same as [30-31]:

$$f_i(2k-1) = \text{argmin}_{f_i} \{ \mathcal{O}(f_i, D_i) + 2\lambda_i(2k-2)^T f_i \\ + \eta \sum_{j \in V_i} \|f_i(2k-2) + f_j(2k-2) - f_i \|^2 \} ; \quad (12)$$

$$\lambda_i(2k-1) = \lambda_i(2k-2) \\ + \frac{\eta}{2} \sum_{j \in V_i} (f_i(2k-1) - f_j(2k-1)) . \quad (13)$$

Note that in the $(2k)$-th (even) iteration, we need the gradient $\nabla \mathcal{O}(f_i(2k-1), D_i)$ and primal difference $\frac{\eta}{2} \sum_{j \in V_i} (f_i(2k-1) - f_j(2k-1))$ for the updates; these are available directly from the previous, $(2k-1)$-th (odd) iteration, i.e., this information can be recycled. In this sense, R-ADMM may be viewed as alternating between conventional ADMM (odd iterations) and a variant of gradient descent (even iterations), where $\frac{2\eta V_i + \gamma}{\eta V_i + \gamma}$ is the step-size with a slightly modified gradient term. The procedure is shown in Algorithm 1.

Algorithm 1: Recycled ADMM (R-ADMM)

Input: $\{D_i\}_{i=1}^N$
Initialize: $\forall i$, generate $f_i(0)$ randomly, $\lambda_i(0) = 0_{d \times 1}$

for $k = 1$ to $K$
do

for $i = 1$ to $N$
do

Update primal variable $f_i(2k-1)$ via [12];
Calculate the gradient $\nabla \mathcal{O}(f_i(2k-1), D_i)$;
Broadcast $f_i(2k-1)$ to all neighbors $j \in V_i$.
done

for $i = 1$ to $N$
do

Calculate $\frac{\eta}{2} \sum_{j \in V_i} (f_i(2k-1) - f_j(2k-1));$
Update dual variable $\lambda_i(2k-1)$ via [13].
done

for $i = 1$ to $N$
do

Use the stored $\nabla \mathcal{O}(f_i(2k-1), D_i)$ and $\frac{\eta}{2} \sum_{j \in V_i} (f_i(2k-1) - f_j(2k-1))$ to update
primal variable $f_i(2k)$ via [10].
Keep the dual variable $\lambda_i(2k) = \lambda_i(2k-1)$;
Broadcast $f_i(2k)$ to all neighbors $j \in V_i$.
done

Output: primal $\{f_i(2K)\}_{i=1}^N$ and dual $\{\lambda_i(2K)\}_{i=1}^N$

Assumption 2: The solution set to the original ERM problem (1) is nonempty and there exists at least one bounded element.

Assumption 3: For all $i \in V$, $\mathcal{O}(f_i, D_i)$ has Lipschitz continuous gradients, i.e., for any $f_i^1$ and $f_i^2$, we have:

$$||\nabla \mathcal{O}(f_i^1, D_i) - \nabla \mathcal{O}(f_i^2, D_i)||_2 \leq M_i ||f_i^1 - f_i^2||_2 \quad (14)$$

By the KKT condition of the primal update [12]:

$$0 = \nabla \mathcal{O}(f_i(2k-1), D_i) + 2\lambda_i(2k-2) \\ + \eta \sum_{j \in V_i} (2f_i(2k-1) - (f_i(2k-2) + f_j(2k-2))) \quad (15)$$

Define the adjacency matrix $A \in \mathbb{R}^{N \times N}$ as:

$$a_{ij} = \begin{cases} 1, & \text{if node } i \text{ and node } j \text{ are connected} \\ 0, & \text{otherwise} \end{cases}$$

Stack the variables $f_i(t)$, $\lambda_i(t)$ and $\nabla \mathcal{O}(f_i(t), D_i)$ for $i \in V$ into matrices, i.e.,

$$\hat{f}(t) = \begin{bmatrix} f_1(t)^T \\ f_2(t)^T \\ \vdots \\ f_N(t)^T \end{bmatrix} \in \mathbb{R}^{N \times d} \quad , \quad \Lambda(t) = \begin{bmatrix} \lambda_1(t)^T \\ \lambda_2(t)^T \\ \vdots \\ \lambda_N(t)^T \end{bmatrix} \in \mathbb{R}^{N \times d}$$

$$\nabla \hat{O}(\hat{f}(t), D_{all}) = \begin{bmatrix} \nabla \mathcal{O}(f_1(t), D_1)^T \\ \nabla \mathcal{O}(f_2(t), D_2)^T \\ \vdots \\ \nabla \mathcal{O}(f_N(t), D_N)^T \end{bmatrix} \in \mathbb{R}^{N \times d}$$

Let $V_i = |V_i|$ be the number of neighbors of node $i$, and define the degree matrix $D = \text{diag}([V_1; V_2; \cdots; V_N]) \in \mathbb{R}^{N \times N}$.
The convergence of R-ADMM is proved by showing that the derivative of $\hat{f}(k)$ are:

$$\dot{f}(2k) = \dot{f}(2k-1) - \Delta\dot{D}^{-1}\{\nabla\hat{O}(\dot{f}(2k-1), D_{all}) + 2\Lambda(2k-1) + \eta(D - A)\dot{f}(2k-1)\};$$

$$\dot{f}(2k) = \Delta\dot{D}^{-1}\{\nabla\hat{O}(\dot{f}(2k-1), D_{all}) + 2\Lambda(2k-1) + \eta(D - A)\dot{f}(2k-1)\};$$

where $\Delta = \Delta\dot{D}^{-1}$. First-order optimality condition [20] states that $f^*$ is equivalent to the optimal solution of our problem, hence the convergence of R-ADMM is proved by showing that $f(t), \Lambda(t)$ in (20)-(21) converges to a pair $(f^*, \Lambda^*)$ satisfying (22)-(23).

**Lemma III.1. [First-order Optimality Condition [20]]** Under Assumptions 1 and 2, the following two statements are equivalent:

- $f^* = [(f_1^*)^T; (f_2^*)^T; \cdots; (f_N^*)^T] \in \mathbb{R}^{N \times d}$ is consensual, i.e., $f_1^* = f_2^* = \cdots = f_N^* = \hat{f}_c^*$, where $\hat{f}_c^*$ is the optimal solution to (11).
- There exists a pair $(f^*, \Lambda^*)$ with $2\Lambda^* = (D - A)X$ for some $X \in \mathbb{R}^{N \times d}$ such that

$$\nabla\hat{O}(\dot{f}, D_{all}) + 2\Lambda^* = \theta_{N \times d}.$$  

**Lemma III.1** shows that a pair $(f^*, \Lambda^*)$ satisfying (22)-(23) is equivalent to the optimal solution of our problem, hence the convergence of R-ADMM is proved by showing that $(f(t), \Lambda(t))$ in (20)-(21) converges to a pair $(f^*, \Lambda^*)$ satisfying (22)-(23).

**Theorem III.1. [Sufficient Condition]** Consider the modified ADMM defined by (20)-(21). Let $\{f(t), \Lambda(t)\}$ be outputs in each iteration and $\{f^*, \Lambda^*\}$ a pair satisfying (22)-(23).

Denote $D_M = \text{diag}(M_1^2, M_2^2, \cdots, M_N^2) \in \mathbb{R}^{N \times N}$ with $0 < M_i < +\infty$ as given in Assumption 3. If the following two conditions hold for some constants $L > 0$ and $\mu > 1$:

$$I + \eta(D + A)\dot{D}^{-1} > \frac{L}{2\sigma_{\min}(D)} \eta D_M(D - A)^{\diamond};$$

$$I + \eta(D + A)\dot{D}^{-1} > \frac{2L}{\sigma_{\min}(D)} \eta D_M + \frac{2L}{\sigma_{\min}(D)} \eta D_M.$$  

where $\sigma_{\min}(D) = \min\{2\eta V_i + \gamma\}$ is the smallest singular value of $D$, then $(f(t), \Lambda(t))$ converges to $(f^*, \Lambda^*)$.

**Proof.** See Appendix I.

By controlling $\gamma$, it is easy to find constants $L > 0$ and $\mu > 1$ such that conditions (24)-(25) are satisfied, and they are not unique. One example is $L = 2$ and $\mu = 2$, in which case (24)-(25) are reduced to:

$$I + \eta(D + A)\dot{D}^{-1} > \frac{2L}{\sigma_{\min}(D)} \eta D_M + \frac{2}{\sigma_{\min}(D)} D_M.$$  

(26)-(27) can be easily satisfied for sufficiently large $\gamma \geq 0$. Note that the conditions are sufficient but not necessary, so in practice convergence may be attained under weaker settings.

**IV. Private R-ADMM**

In this section we present a privacy preserving version of R-ADMM. In odd iterations, we adopt the objective perturbation [33] where a random linear term $\epsilon_i(2k-1)^T f_i$ is added to the objective function in (12) where $\epsilon_i(2k-1)$ follows the probability density proportional to $\exp(-\alpha_i(2k-1)||f_i||^2_2)$:

$$L_i^\text{priv}(2k-1) = O(f_i, D_i) + (2\alpha_i(2k-2) + \epsilon_i(2k-1))^T f_i + \eta \sum_{j \in \mathcal{N}_i} ||\frac{1}{2}(f_i(2k-2) + f_j(2k-2)) - f_i||^2_2.$$  

To generate this noisy vector, choose the norm from the gamma distribution with shape $d$ and scale $\frac{1}{\alpha_i(2k-1)}$, and the direction uniformly, where $d$ is the dimension of the feature space. Node $i$’s local result is obtained by finding the optimal solution to the private objective function:

$$f_i(2k-1) = \arg\min_{\hat{f}_i} L_i^\text{priv}(2k-1), \ i \in \mathcal{N}.$$  

(28) In the $(2k)$-th iteration, use the stored results $\epsilon_i(2k-1) + \nabla O(f_i(2k-1), D_i)$ and $\frac{\eta}{2} \sum_{j \in \mathcal{N}_i} (f_i(2k-2) - f_j(2k-1))$ to update primal variables, where the latter can be obtained from the dual update in the $(2k-1)$-th update, and the former can be obtained directly from the KKT condition in the $(2k-1)$-th iteration:

$$\epsilon_i(2k-1) + \nabla O(f_i(2k-1), D_i) = -2\alpha_i(2k-2) \eta \sum_{j \in \mathcal{N}_i} (f_i(2k-1) - f_j(2k-1)) - f_i(2k-2) - f_j(2k-2).$$  

Pure differential privacy was adopted in this work, but the weaker ($\epsilon, \delta$)-differential privacy can be applied as well.
Then the even update is given by:
\[
f_{i}(2k) = f_{i}(2k-1) - \frac{1}{2\eta_{i}} \{2\lambda_{i}(2k-1) + \epsilon_{i}(2k-1) + \nabla O(f_{i}(2k-1), D_{i})
\]
+ \eta \sum_{j \in \mathcal{V}_{i}} (f_{i}(2k-1) - f_{j}(2k-1))\] .
\[
(29)
\]

Algorithm 2 shows the complete procedure, where the condition used to generate \( \eta \) helps to bound the worst-case privacy loss but is not necessary in guaranteeing convergence.

**Algorithm 2: Private R-ADMM**

**Input:** \( \{D_{i}\}_{i=1}^{N}, \{\alpha_{i}(1), \cdots, \alpha_{i}(K)\}_{i=1}^{N} \)

**Initialize:** \( \forall i \), generate \( f_{i}(0) \) randomly, \( \lambda_{i}(0) = 0_{d \times 1} \)

**Parameter:** Select \( \eta \) s.t. \( 2c_{1} < \min_{i} \{\frac{B_{i}}{C_{i}}(\frac{2}{N} + 2\eta_{i})\} \)

**for** \( k = 1 \) to \( K \) **do**

**for** \( i = 1 \) to \( \mathcal{N} \) **do**

Generate noise \( \epsilon_{i}(2k-1) \sim \exp(-\alpha_{i}(k)||\epsilon||_{2}) \);

Update primal variable \( f_{i}(2k-1) \) via [28];

Broadcast \( f_{i}(2k-1) \) to all neighbors \( j \in \mathcal{V}_{i} \).

**for** \( i = 1 \) to \( \mathcal{N} \) **do**

Calculate \( \frac{\eta}{2} \sum_{j \in \mathcal{V}_{i}} (f_{i}(2k-1) - f_{j}(2k-1)) \);

Update dual variable \( \lambda_{i}(2k) \) via [13].

**for** \( i = 1 \) to \( \mathcal{N} \) **do**

Use the stored information \( \frac{\eta}{2} \sum_{j \in \mathcal{V}_{i}} (f_{i}(2k-1) - f_{j}(2k-1)) \) to update primal variable \( f_{i}(2k) \) via [29];

Keep the dual variable \( \lambda_{i}(2k) = \lambda_{i}(2k-1) \);

Broadcast \( f_{i}(2k) \) to all neighbors \( j \in \mathcal{V}_{i} \).

**Output:** Upper bound of the total privacy loss \( \beta \):

primal \( \{f_{i}(2K)\}_{i=1}^{N} \) and dual \( \{\lambda_{i}(2K)\}_{i=1}^{N} \)

In the distributed and iterative setting, the “output” of the algorithm is not merely the end result, but includes all intermediate results generated and exchanged during the iterative process. For this reason, we adopt the differential privacy definition proposed in [27] as follows.

**Definition IV.1.** Consider a connected network \( G(\mathcal{N}, \mathcal{E}) \) with a set of nodes \( \mathcal{N} = \{1, 2, \cdots, N\} \). Let \( f(t) = \{f_{i}(t)\}_{i=1}^{N} \) denote the information exchange of all nodes in the t-th iteration. A distributed algorithm is said to satisfy \( \beta \)-differential privacy during T iterations if for any two datasets \( D_{a} = \bigcup_{i} D_{i} \) and \( D_{b} = \bigcup_{i} D_{i} \), differing in at most one data point, for any set of possible outputs \( S \) during T iterations, the following holds:

\[
Pr([f(t)]_{t=0}^{T} \in S | D_{a}) \leq \exp(\beta)
\]

\[
Pr([f(t)]_{t=0}^{T} \in S | D_{b})
\]

We now state another result of this paper, on the privacy property of the private R-ADMM (Algorithm 2) using the above definition. Additional assumptions on \( \mathcal{L}^{e}() \) and \( R() \) are used.

**Assumption 4:** The loss function \( \mathcal{L} \) is strictly convex and twice differentiable. \( |\nabla \mathcal{L}| \leq 1 \) and \( 0 < \mathcal{L}'' \leq c_{1} \) with \( c_{1} \) being a constant.

**Assumption 5:** The regularizer \( R \) is \( 1 \)-strongly convex and twice continuously differentiable.

**Lemma IV.1.** Consider the private R-ADMM (Algorithm 2), \( \forall k = 1, \cdots, K \), assume the total privacy loss up to the \((2k-1)\)-th iteration can be bounded by \( \beta_{2k-1} \), then the total privacy loss up to the \(2k\)-th iteration can also be bounded by \( \beta_{2k-1} \). In other words, given the private results in odd iterations, outputting private results in the even iterations does not release more information about the input data.

**Proof.** See Appendix [II]

**Theorem IV.1.** Normalize feature vectors in the training set such that \( ||x_{n}^{i}||_{2} \leq 1 \) for all \( i \in \mathcal{N} \) and \( n \). Then the private R-ADMM algorithm (Algorithm 2) satisfies the \( \beta \)-differential privacy with

\[
\beta \geq \max_{i \in \mathcal{N}} \{ \sum_{k=1}^{K} \frac{2C}{B_{i}} \left( \frac{1.4c_{1}}{(\frac{2}{N} + 2\eta_{i})} + \alpha_{i}(k) \right) \}.
\]

**Proof.** See Appendix [III]

**V. NUMERICAL EXPERIMENTS**

We use the Adult dataset from the UCI Machine Learning Repository [34]. It consists of personal information of around 48,842 individuals, including age, sex, race, education, occupation, income, etc. The goal is to predict whether the annual income of an individual is above $50,000.

Following the same pre-processing steps as in [27], the final data includes 45,223 individuals, each represented as a 105-dimensional vector of norm at most 1. We will use as loss function the logistic loss \( \mathcal{L}(z) = \log(1 + \exp(-z)) \), with \( ||\nabla \mathcal{L}|| \leq 1 \) and \( 0 < \mathcal{L}'' \leq c_{1} = \frac{1}{4} \). The regularizer is \( R(f_{i}) = \frac{1}{2}||f_{i}||_{2}^{2} \). We will measure the accuracy of the algorithm by the average loss \( L(t) := \frac{1}{N} \sum_{i=1}^{N} \frac{1}{B_{i}} \sum_{n=1}^{B_{i}} \mathcal{L}(y_{n}^{i} f_{i}(t)^{T} x_{n}^{i}) \) over the training set. We will measure the privacy of the algorithm by the upper bound \( P(t) := \max_{i \in \mathcal{N}} \sum_{k=1}^{K} \frac{2C}{B_{i}} \left( \frac{1.4c_{1}}{(\frac{2}{N} + 2\eta_{i})} + \alpha_{i}(k) \right) \). The smaller
L(t) and P(t), the higher accuracy and stronger privacy guarantee.

A. Convergence of non-private R-ADMM

Figure 1 shows the convergence of R-ADMM with different γ and fixed η = 0.5 for a small network (N = 5) and a large network (N = 20), both are randomly generated. Due to the linear approximation in even iterations, it’s possible to cause an increased average loss as shown in the plot. However, the odd iterations will always compensate this increase; if we only look at the odd iterations, R-ADMM achieves a similar convergence rate as conventional ADMM. γ can also be thought of as an extra penalty parameter for each node in even iterations to punish its update, i.e., the difference between f_i(2k) and f_i(2k−1). Larger γ can result in smaller oscillation between even and odd iterations but will also lower the convergence rate.

B. Private R-ADMM

We next inspect the accuracy and privacy of the private R-ADMM (Algorithm 2) and compare it with the private (conventional) ADMM using dual variable perturbation (DVP) [26] and the private M-ADMM using penalty perturbation (PP) [27]. In the set of experiments, we fix γ = 0.2, η = 1 in private R-ADMM and set the noise parameter α_i(k) = α, ∀i, k. The noise parameters of conventional ADMM and M-ADMM are also chosen respectively such that they have almost the same total privacy loss bounds.

For each parameter setting, we perform 10 independent runs of the algorithm, and record both the mean and the range of their accuracy. Specifically, L_i(t) denotes the average loss over the training dataset in the t-th iteration of the l-th experiment (1 ≤ l ≤ 10). The mean of average loss is then given by \( L_{\text{mean}}(t) = \frac{1}{10} \sum_{l=1}^{10} L_i(t) \), and the range \( L_{\text{range}}(t) = \max_{1 \leq l \leq 10} L_i(t) - \min_{1 \leq l \leq 10} L_i(t) \). The larger the range \( L_{\text{range}}(t) \) the less stable the algorithm, i.e., under the same parameter setting, the difference in performances (convergence curves) of every two experiments is larger. Each parameter setting also has a corresponding upper bound on the privacy loss denoted by P(t). Figures 2(a)-2(b) show both \( L_{\text{mean}}(t) \) and \( L_{\text{range}}(t) \) as vertical bars centered at \( L_{\text{mean}}(t) \). Their corresponding privacy upper bound is given in Figures 2(c)-2(d). The pair 2(a), 2(c) (resp. 2(b), 2(d)) is for the same parameter setting. We see that the private R-ADMM (red) has higher accuracy than both the private ADMM (black) and M-ADMM (blue), and the improvement is more significant with the smaller total privacy loss.

We also incorporate the idea from [27] into private R-ADMM, where we decrease the step-size, i.e., increase η and γ, over iterations to stabilize the algorithm and improve the algorithmic performance. The result is shown in Figure 3 where the privacy loss bound is controlled to be the same during the whole period. It shows that by varying the step-size, the privacy-utility tradeoff can be further improved.

VI. Conclusion

We presented Recycled ADMM (R-ADMM), a modified version of ADMM that can improve the privacy-utility tradeoff significantly with less computation. The idea is to repeatedly use the existing computational results instead of the raw data to make updates. We also established a sufficient condition for convergence and privacy analysis using objective perturbation.
\[
\langle \hat{f}(t+1) - \hat{f}^*, \eta(D + A)\hat{D}^{-1}(2\Lambda(t+1) - 2\Lambda(t)) - \eta(D + A)(\hat{f}(t+1) - \hat{f}(t)) - \eta(D + A)\hat{D}^{-1}(D - A)\hat{f}(t) \rangle_F \geq 0 .
\]

(31)

\[
\langle \hat{f}(t+1) - \hat{f}^*, \eta(D + A)\hat{D}^{-1}(2\Lambda(t+1) - 2\Lambda(t)) - \eta(D + A)\hat{D}^{-1}\eta(D - A)\hat{f}(t) \rangle_F
\]
\[
= (\hat{f}(t+1) - \hat{f}^*, \eta(D + A)\hat{D}^{-1}\eta(D - A)(\hat{f}(t+1) - \hat{f}(t)))_F
\]
\[
= \frac{1}{2} \|\hat{f}(t+1) - \hat{f}^*\|_{\hat{G}_1}^2 + \frac{1}{2} \|\hat{f}(t+1) - \hat{f}(t)\|_{\hat{G}_1}^2 - \frac{1}{2} \|\hat{f}(t) - \hat{f}^*\|_{\hat{G}_1}^2.
\]

(32)

\[
\langle \hat{f}(t+1) - \hat{f}^*, (I + \eta(D + A)\hat{D}^{-1})(2\Lambda^* - 2\Lambda(t+1)) \rangle_F
\]
\[
= \langle \hat{f}(t+1) - \hat{f}^*, (I + \eta(D + A)\hat{D}^{-1})(2\Lambda^* - 2\Lambda(t+1)) \rangle_F
\]
\[
= \frac{1}{\eta}(D - A)^+(2\Lambda(t+1) - 2\Lambda(t))((I + \eta(D + A)\hat{D}^{-1})(2\Lambda^* - 2\Lambda(t+1)))_F
\]
\[
= \frac{1}{2}[2\Lambda^* - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2} - \frac{1}{2}[2\Lambda^* - 2\Lambda(t+1)]\|\hat{f}_2\|^2_{\hat{G}_2} - \frac{1}{2}[2\Lambda^* - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2};
\]

(33)

\[
\langle \hat{f}(t+1) - \hat{f}^*, -\eta(D + A)(\hat{f}(t+1) - \hat{f}(t)) \rangle_F
\]
\[
= \|\hat{f}(t+1) - \hat{f}^*\|_{\eta(D + A)}^2 - \frac{1}{2}\|\hat{f}(t+1) - \hat{f}^*\|_{\eta(D + A)}^2 - \frac{1}{2}\|\hat{f}(t) - \hat{f}^*\|_{\eta(D + A)}^2.
\]

(34)

\[
\langle \hat{f}(t+1) - \hat{f}^*, -\eta(D + A)\hat{D}^{-1}(\nabla\hat{O}(\hat{f}(t), D_{all}) - \nabla\hat{O}(\hat{f}, D_{all})) \rangle_F
\]
\[
= \langle \hat{f}(t+1) - \hat{f}^*, -\eta(D + A)\hat{D}^{-1}\eta(D - A)\hat{f}(t) \rangle_F
\]
\[
= \langle \hat{f}(t+1) - \hat{f}^*, \eta(D + A)\hat{D}^{-1}(2\Lambda(t+1) - 2\Lambda(t)) - \eta(D + A)(\hat{f}(t+1) - \hat{f}(t)) - \eta(D + A)\hat{D}^{-1}(D - A)\hat{f}(t) \rangle_F
\]
\[
= \langle \hat{f}(t+1) - \hat{f}^*, (I + \eta(D + A)\hat{D}^{-1})(2\Lambda^* - 2\Lambda(t+1)) \rangle_F
\]
\[
= \frac{1}{\eta}(D - A)^+(2\Lambda(t+1) - 2\Lambda(t))((I + \eta(D + A)\hat{D}^{-1})(2\Lambda^* - 2\Lambda(t+1)))_F
\]
\[
= \frac{1}{2}[2\Lambda^* - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2} - \frac{1}{2}[2\Lambda^* - 2\Lambda(t+1)]\|\hat{f}_2\|^2_{\hat{G}_2} - \frac{1}{2}[2\Lambda^* - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2};
\]

(35)

\[
\frac{1}{\eta}(D - A)^+(2\Lambda(t+1) - 2\Lambda(t))((I + \eta(D + A)\hat{D}^{-1})(2\Lambda^* - 2\Lambda(t+1)))_F
\]
\[
= \frac{1}{2}[2\Lambda(t+1) - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2} - \frac{1}{2}[2\Lambda(t+1) - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2} - \frac{1}{2}[2\Lambda^* - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2};
\]

(36)

\[
\frac{1}{2}[2\Lambda(t+1) - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2} - \frac{1}{2}[2\Lambda(t+1) - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2} - \frac{1}{2}[2\Lambda^* - 2\Lambda(t)]\|\hat{f}_2\|^2_{\hat{G}_2};
\]

(37)

\[
\langle X, JX \rangle_F \text{ and } (X)^+ \text{ be the pseudo inverse of } X. \text{ Define:}
\]
\[
G_1 = \eta(D + A)\hat{D}^{-1}\eta(D - A); \quad G_2 = \eta(D - A)^+(I + \eta(D + A)\hat{D}^{-1}).
\]

Use (21), (23) and the fact that \( \langle A, JB \rangle_F = \langle J^T A, B \rangle_F \), Eqs. (32), (33), (34) hold. Let \( \sqrt{X} \) denote the square root of a symmetric positive semi-definite (PSD) matrix \( X \) that is also symmetric PSD. Eqs. (35) holds, where the inequality uses the facts that \( O(f_i, D_i) \) is convex for all \( i \) and that the matrix \( \eta(D + A)\hat{D}^{-1} \) is positive definite.

According to (14) in Assumption 3, define the matrix \( D_M = \text{diag}(M_{T1}^2, M_{T2}^2, \ldots, M_{Tn}^2) \in \mathbb{R}^{N \times N} \), it implies \( \|\nabla\hat{O}(\hat{f}, D_{all}) - \nabla\hat{O}(\hat{f}^2, D_{all})\|_F^2 \leq \|\hat{f} - \hat{f}^2, D_M(\hat{f}^2 - \hat{f}) \|_F^2 \).
Eqn. (37) becomes:

\[ \sum_{t=0}^{\infty} \{(|\hat{f}(t) - \hat{f}(t + 1)|^2_{\hat{R}_1} + |2\Lambda(t + 1) - 2\Lambda(t)|^2_{\hat{R}_2}) \]

++ \left\{ |\hat{f}(0) - \hat{f}^*(t)|^2_{G_1} + |\hat{f}(0) - \hat{f}^*(t)|^2_{G_2} + |2\Lambda^* - 2\Lambda(0)|^2_{\hat{R}_2} + |2\Lambda^* - 2\Lambda(\infty)|^2_{\hat{R}_2} \right\} \quad (44)

The RHS of (44) is finite, implies that \( \lim_{t \to \infty} \{|\hat{f}(t) - \hat{f}(t + 1)|^2_{\hat{R}_1} + |2\Lambda(t + 1) - 2\Lambda(t)|^2_{\hat{R}_2} \} = 0 \). Since \( R_1, R_2 \) are not unique, by (41)-(42), it requires \( \lim_{t \to \infty} \{|\hat{f}(t) - \hat{f}(t + 1)|^2_{\hat{R}_1} = 0 \) and \( \lim_{t \to \infty} |2\Lambda(t + 1) - 2\Lambda(t)|^2_{\hat{R}_2} = 0 \) should hold for all possible \( R_1, R_2 \). Therefore, \( \lim_{t \to \infty} (\hat{f}(t) - \hat{f}(t + 1)) = 0 \) and \( \lim_{t \to \infty} (2\Lambda(t + 1) - 2\Lambda(t)) = 0 \) must hold.

Plug into (38) and use (21)-(23) gives Eqn. (36).

Combine (32)-(33),(34),(35), (31) becomes Eqn. (37). Suppose the following two conditions hold for some constants \( L > 0 \) and \( \mu > 1 \):

\[ (I + \eta(D + A)\hat{D}^{-1}) \succ \frac{L\mu}{2\sigma_{\min}(D)} D_M(D - A)^+ \quad (39) \]

\[ \eta(D + A) \succ \frac{L\mu}{2\sigma_{\min}(D)}(\mu - 1)D_M \quad (40) \]

Substitute \( G_1 = \eta(D + A)\hat{D}^{-1}\eta(D - A) \) and \( G_2 = \frac{L\mu}{\eta}(D - A)^+(I + \eta(D + A)\hat{D}^{-1}) \), define \( R_1 \) and \( R_2 \) below gives:

\[ R_1 = \eta(D + A) - G_1 - \frac{L\mu}{2\sigma_{\min}(D)}D_M \]

\[ R_2 = G_2 - \frac{L\mu}{2\sigma_{\min}(D)} (\frac{1}{\eta}(D - A)^+)D_M \quad (41) \]

Eqn. (37) becomes:

\[ \frac{1}{2}||\hat{f}(t) - \hat{f}(t + 1)||^2_{R_1} + \frac{1}{2}||2\Lambda(t + 1) - 2\Lambda(t)||^2_{R_2} \]

\[ \frac{1}{2}||\hat{f}(t + 1) - \hat{f}^*||^2_{Z_1} - \frac{1}{2}||\hat{f}(t) - \hat{f}^*||^2_{Z_1} \]

\[ \frac{1}{2}||2\Lambda^* - 2\Lambda(t)||^2_{Z_2} - \frac{1}{2}||2\Lambda^* - 2\Lambda(t + 1)||^2_{Z_2} \]

\[ \frac{1}{2}||\hat{f}(t) - \hat{f}^*||^2_{\eta(D + A)} - \frac{1}{2}||\hat{f}(t + 1) - \hat{f}^*||^2_{\eta(D + A)} \quad (43) \]

\[ \sum_{j \in \mathcal{V}_i} \eta \{ f_i(2k - 1) - f_i(2k - 2) - f_j(2k - 2) \} \quad (47) \]

which is a deterministic mapping taking the outputs from (2k-1)-th iteration as input. Because the differential privacy is immune to post-processing [35], releasing \( \{f_i(2k)\}_{i=1}^{N} \) doesn't increase the privacy loss, i.e., the total privacy loss up to (2k)-th iteration can still be bounded by \( \beta_{2k-1} \).
APPENDIX III
PROOF OF THEOREM IV.1

Use the uppercase letters X and lowercase letters x to denote random variables and the corresponding realizations, and use $\mathcal{F}_X(\cdot)$ to denote its probability distribution.

For two neighboring datasets $D_{all}$ and $\hat{D}_{all}$ of the network, by Lemma IV.1, the total privacy loss is only contributed by odd iterations. Thus, the ratio of joint probabilities (privacy loss) is given by:

\[
\frac{\mathcal{F}_{F_2}(0:2K)}{(f(0))^{2K}_r = 0} K[D_{all}] = \frac{\mathcal{F}_{F_2}(0)(f(0))_{r = 0} K[D_{all}]}{\mathcal{F}_{F_2}(0)(f(0))_{r = 0} K[D_{all}]}
\]

\[
\prod_{k=1}^{K} \frac{\mathcal{F}_{E_{i}(2k-1)}(g_k^{-1}(f_i(2k-1), D_i))}{\mathcal{F}_{E_{i}(2k-1)}(g_k^{-1}(f_i(2k-1), D_i))}
\]

Since $f_i(0)$ is randomly selected for all $i$, which is independent of dataset, there is $\mathcal{F}_{F_2}(0)(f(0))_{r = 0} K[D_{all}] = \mathcal{F}_{F_2}(0)(f(0))_{r = 0} K[D_{all}]$. First only consider $(2k-1)$-th iteration, since the primal variable is updated according to (38), by KKT optimality condition:

\[
\epsilon_i(2k - 1) = -\nabla O(f_i(2k - 1), D_i) - 2\lambda_i (2k - 2)
\]

\[
-\eta \sum_{j \in J_i} (2f_i(2k - 1) - f_i(2k - 2) - f_j(2k - 2))
\]

Given $\{f(r)\}_{r = 0}^{2k-2}$, $F_i(2k - 1)$ and $E_i(2k - 1)$ will be bijective $\forall i$, there is:

\[
\frac{\mathcal{F}_{E_{i}(2k-1)}(g_k^{-1}(f_i(2k-1), D_i))}{\mathcal{F}_{E_{i}(2k-1)}(g_k^{-1}(f_i(2k-1), D_i))}
\]

\[
\prod_{k=1}^{K} \frac{\mathcal{F}_{E_{i}(2k-1)}(g_k^{-1}(f_i(2k-1), D_i))}{\mathcal{F}_{E_{i}(2k-1)}(g_k^{-1}(f_i(2k-1), D_i))}
\]

Consider the first part, $E_i(2k - 1) \sim \exp\{-\alpha_i(k)|\epsilon_i(2k - 1)|\}$, let $\hat{\epsilon}_i(2k - 1) = g_k^{-1}(f_i(2k - 1), D_i)$ and $\epsilon_i(2k - 1) = g_k^{-1}(f_i(2k - 1), D_i)$

\[
\prod_{k=1}^{K} \frac{\mathcal{F}_{E_{i}(2k-1)}(g_k^{-1}(f_i(2k-1), D_i))}{\mathcal{F}_{E_{i}(2k-1)}(g_k^{-1}(f_i(2k-1), D_i))}
\]

\[
\prod_{k=1}^{K} \frac{\exp(\alpha_k(k)|\hat{\epsilon}_i(2k - 1) - |\epsilon_i(2k - 1)|)}{\exp(\sum_{k=1}^{K} \alpha_k(k)|\epsilon_i(2k - 1) - \epsilon_i(2k - 1)|)}
\]

Without loss of generality, let $D_i$ and $\hat{D}_i$ be only different in the first data point, say $(x_i^1, y_i^1)$ and $(\hat{x}_i^1, \hat{y}_i^1)$ respectively. By (49), Assumptions 4 and the facts that $\|x_i^n\|_2 \leq 1$ (pre-normalization), $y_i^n \in \{+1, -1\}$.

\[
||\hat{\epsilon}_i(2k - 1) - \epsilon_i(2k - 1)||
\]

\[
||\hat{\epsilon}_i(2k - 1) - \epsilon_i(2k - 1)|| = ||\nabla O(f_i(2k - 1), D_i) - \nabla O(f_i(2k - 1), D_i)|| \leq \frac{2C}{B_i}
\]

(52)

(53)

(54)

Consider the second part, the Jacobian matrix $J(g_k^{-1}(f_i(2k-1), D_i))$ is:

\[
J(g_k^{-1}(f_i(2k-1), D_i)) = -\frac{C}{B_i} \sum_{n=1}^{B_i} \lambda''(y_i^n f_i(2k-1)^T x_i^n \hat{x}_i^n) x_i^n (x_i^n)^T
\]

\[
-\frac{\lambda''(y_i^n f_i(2k-1)^T x_i^n \hat{x}_i^n) x_i^n (x_i^n)^T}{\lambda''(y_i^n f_i(2k-1)^T x_i^n \hat{x}_i^n) x_i^n (x_i^n)^T} - \nu^2 R(f_i(2k - 1)) - 2\eta V_i x_i^n
\]

Define

\[
G(k) = \frac{C}{B_i} \lambda''(y_i^n f_i(2k-1)^T x_i^n \hat{x}_i^n) x_i^n (x_i^n)^T
\]

\[
H(k) = -J(g_k^{-1}(f_i(2k-1), D_i))
\]

There is:

\[
\frac{|\det(J(g_k^{-1}(f_i(2k-1), D_i)))||}{|\det(H(k)))|} = \frac{1}{\prod_{j=1}^{J}(1 + \lambda_j(H(k)^{-1}G(k)))}
\]

\[
\lambda_j(H(k)^{-1}G(k))
\]

(55)

where $\lambda_j(H(k)^{-1}G(k))$ denotes the $j$-th largest eigenvalue of $H(k)^{-1}G(k)$. Since $G(k)$ has rank at most 2, $H(k)^{-1}G(k)$ also has rank at most 2. By Assumptions 4 and 5, the eigenvalue of $H(k)$ and $G(k)$ satisfy

\[
\lambda_j(H(k)) \geq \frac{\rho}{N} + 2\eta V_i > 0;
\]

\[
-\frac{C c_1}{B_i} \leq \lambda_j(G(k)) \leq \frac{C c_1}{B_i}.
\]
Therefore, the total privacy loss during $T$ iterations can be bounded with a simpler expression:

$$\frac{c_1}{\beta} \leq \lambda_j(H(k)^{-1}G(k)) \leq \frac{c_1}{\beta} \left( \frac{\rho}{N} + 2\eta V_i \right).$$

Since $2c_1 < \frac{B_i}{\beta} \left( \frac{\rho}{N} + 2\eta V_i \right)$, there is

$$-1 \leq \lambda_j(H(k)^{-1}G(k)) \leq \frac{1}{2}.$$

Since $\lambda_{\min}(H(k)^{-1}G(k)) > -1$, there is

$$\frac{1}{1 + \lambda_{\max}(H(k)^{-1}G(k))^2} \leq \frac{1}{\det(I + H(k)^{-1}G(k))} \leq \frac{1}{1 + \lambda_{\min}(H(k)^{-1}G(k))^2}.$$

Therefore,

$$\prod_{k=1}^{K} \left| \frac{\det(J(g_k^{-1}(f_i(2k-1), D_i)))}{\det(J(g_k^{-1}(f_i(2k-1), D_i)))} \right| \leq \exp\left( \sum_{k=1}^{K} 2 \ln \left( 1 - \frac{c_1}{B_i} \left( \frac{\rho}{N} + 2\eta V_i \right) \right) \right).$$

Since for any real number $x \in [0, 0.5]$, $-\ln(1-x) < 1.4x$, (56) can be bounded with a simper expression:

$$\prod_{k=1}^{K} \left| \frac{\det(J(g_k^{-1}(f_i(2k-1), D_i)))}{\det(J(g_k^{-1}(f_i(2k-1), D_i)))} \right| \leq \exp\left( \sum_{k=1}^{K} \frac{2.8c_1}{B_i} \left( \frac{\rho}{N} + 2\eta V_i \right) \right).$$

Combine (54)–(57), (51) can be bounded:

$$\mathcal{F}_{F(0:2K)}\left( \{f(r)\}_{r=0}^{2K} | D_{alt} \right) \leq \exp\left( \sum_{k=1}^{K} 2C \frac{1.4c_1}{B_i} \left( \frac{\rho}{N} + 2\eta V_i \right) + \alpha_i(k) \right).$$

Therefore, the total privacy loss during $T$ iterations can be bounded by any $\beta$:

$$\beta \geq \max_{i \in \mathcal{A}} \left\{ \sum_{k=1}^{K} 2C \frac{1.4c_1}{B_i} \left( \frac{\rho}{N} + 2\eta V_i \right) + \alpha_i(k) \right\}.$$
[24] S. Han, U. Topcu, and G. J. Pappas, “Differentially private distributed
constrained optimization,” IEEE Transactions on Automatic Control,
vol. 62, no. 1, pp. 50–64, 2017.
[25] A. Bellet, R. Guerraoui, M. Taziki, and M. Tommasi, “Fast and dif-
ferentially private algorithms for decentralized collaborative machine
learning,” Ph.D. dissertation, INRIA Lille, 2017.
[26] T. Zhang and Q. Zhu, “Dynamic differential privacy for admm-based
distributed classification learning,” IEEE Transactions on Information
Forensics and Security, vol. 12, no. 1, pp. 172–187, 2017.
[27] X. Zhang, M. M. Khalili, and M. Liu, “Improving the privacy and
accuracy of ADMM-based distributed algorithms,” in Proceedings of the
35th International Conference on Machine Learning, vol. 80, 2018,
pp. 5796–5805.
[28] Z. Huang, R. Hu, Y. Gong, and E. Chan-Tin, “Dp-admm: Admm-
based distributed learning with differential privacy,” arXiv preprint
arXiv:1808.10101, 2018.
[29] P. A. Forero, A. Cano, and G. B. Giannakis, “Consensus-based
distributed support vector machines,” Journal of Machine Learning
Research, vol. 11, no. May, pp. 1663–1707, 2010.
[30] A. Mokhtari, W. Shi, Q. Ling, and A. Ribeiro, “Decentralized quadrat-
ically approximated alternating direction method of multipliers,” in Signal and Information Processing (GlobalSIP), 2015 IEEE Global
Conference on. IEEE, 2015, pp. 795–799.
[31] Q. Ling, W. Shi, G. Wu, and A. Ribeiro, “Dlm: Decentralized
linearized alternating direction method of multipliers,” IEEE Trans-
actions on Signal Processing, vol. 63, no. 15, pp. 4051–4064, 2015.
[32] J. Kelner, “An algorithmists toolkit,” 2007. [Online]. Available:
http://bit.ly/2C4yRCX
[33] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, “Differentially
private empirical risk minimization,” Journal of Machine Learning
Research, vol. 12, no. Mar, pp. 1069–1109, 2011.
[34] M. Lichman, “UCI machine learning repository,” 2013. [Online].
Available: http://archive.ics.uci.edu/ml
[35] C. Dwork, A. Roth et al., “The algorithmic foundations of differential
privacy,” Foundations and Trends® in Theoretical Computer Science,
vol. 9, no. 3–4, pp. 211–407, 2014.