Improving the Privacy and Accuracy of ADMM-Based Distributed Algorithms

Xueru Zhang [1] **Mohammad Mahdi Khalili** [1] **Mingyan Liu** [1]

Abstract

Alternating direction method of multiplier (ADMM) is a popular method used to design distributed versions of a machine learning algorithm, whereby local computations are performed on local data with the output exchanged among neighbors in an iterative fashion. During this iterative process the leakage of data privacy arises. A differentially private ADMM was proposed in prior work (Zhang & Zhu, 2017) where only the privacy loss of a single node during one iteration was bounded, a method that makes it difficult to balance the tradeoff between the utility attained through distributed computation and privacy guarantees when considering the total privacy loss of all nodes over the entire iterative process. We propose a perturbation method for ADMM where the perturbed term is correlated with the penalty parameters; this is shown to improve the utility and privacy simultaneously. The method is based on a modified ADMM where each node independently determines its own penalty parameter in every iteration and decouples it from the dual updating step size. The condition for convergence of the modified ADMM and the lower bound on the convergence rate are also derived.

1. Introduction

Distributed machine learning is crucial for many settings where the data is possessed by multiple parties or when the quantity of data prohibits processing at a central location. It helps to reduce the computational complexity, improve both the robustness and the scalability of data processing. In a distributed setting, multiple entities/nodes collaboratively work toward a common optimization objective through an interactive process of local computation and message passing, which ideally should result in all nodes converging to a global optimum. Existing approaches to decentralizing an optimization problem primarily consist of subgradient-based algorithms (Nedic et al., 2008; Nedic & Ozdaglar, 2009; Lobel & Ozdaglar, 2011), ADMM-based algorithms (Wei & Ozdaglar, 2012; Ling & Ribeiro, 2014; Shi et al., 2014; Zhang & Kwok, 2014; Ling et al., 2016), and composite of subgradient and ADMM (Bianchi et al., 2014). 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 $O(\frac{1}{\sqrt{k}})$, where k is the number of iterations (Wei & Ozdaglar, 2012). In this study, we will solely focus on ADMM-based algorithms.

The information exchanged over the iterative process gives rise to privacy concerns if the local training data is proprietary to each node, especially when it 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.

A widely used notion of privacy is the ε-differential privacy; 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 (Dwork, 2006). Several differentially private distributed algorithms have been proposed, including (Hale & Egerstedty, 2015; Huang et al., 2015; Han et al., 2017; Zhang & Zhu, 2017; Bellet et al., 2017). While a number of such studies have been done for (sub)gradient-based algorithms, 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 (Zhang & Zhu, 2017) applies differential privacy to ADMM, where the noise is either added to the dual variable (*dual variable perturbation*) or the primal variable (*primal variable perturbation*) in ADMM updates. However, (Zhang & Zhu, 2017) could only bound the privacy loss of a single iteration. Since an attacker can potentially use all intermediate results to perform inference, the privacy loss accumulates over time through the iterative process. It turns out that the tradeoff between the utility of the algorithm and its privacy preservation over the entire computational process becomes hard using the existing method.

[1]Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, Michigan, USA. Correspondence to: Xueru Zhang <xueru@umich.edu>, Mohammad Mahdi Khalili <khalili@umich.edu>, Mingyan Liu <mingyan@umich.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

In this study we propose a perturbation method that could simultaneously improve the accuracy and privacy for ADMM. We start with a modified version of ADMM whereby each node independently decides its own penalty parameter in each iteration; it may also differ from the dual updating step size. For this modified ADMM we establish conditions for convergence and quantify the lower bound of the convergence rate. We then present a penalty perturbation method to provide differential privacy. Our numerical results show that under this method, by increasing the penalty parameter over iterations, we can achieve stronger privacy guarantee as well as better algorithmic performance, i.e., more stable convergence and higher accuracy.

The remainder of the paper is organized as follows. We present problem formulation and definition of differential privacy and ADMM in Section 2 and a modified ADMM algorithm along with its convergence analysis in Section 3. A private version of this ADMM algorithm is then introduced in Section 4 and numerical results in Section 5. Discussions are given in Section 6 and Section 7 concludes the paper.

2. Preliminaries

2.1. Problem Formulation

Consider a connected network[1] 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 $\mathcal{V}_i$ denote node i's set of neighbors, excluding itself. A node i contains a dataset $D_i = \{(x_i^n, y_i^n) | n = 1, 2, \cdots, B_i\}$, where $x_i^n \in \mathbb{R}^d$ is the feature vector representing the n-th sample belonging to i, $y_i^n \in \{-1, 1\}$ the corresponding label, and B_i the size of D_i.

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

$$\min_{f_c} O_{ERM}(f_c, D_{all}) = \sum_{i=1}^{N} \frac{C}{B_i} \sum_{n=1}^{B_i} \mathscr{L}(y_i^n f_c^T x_i^n) + \rho R(f_c) \tag{1}$$

where $C \leq B_i$ and $\rho > 0$ are constant parameters of the algorithm, the loss function $\mathscr{L}(\cdot)$ measures the accuracy of classifier, and the regularizer $R(\cdot)$ helps to 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 [2].

[1] A connected network is one in which every node is reachable (via a path) from every other node.

[2] The proposed penalty perturbation method is not limited to classification problems. It can be applied to general ADMM-based distributed algorithms since the convergence and privacy analysis in Section 3 & 4 remain valid.

2.2. 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 $\{w_{ij} | i \in \mathcal{N}, j \in \mathcal{V}_i\}$ are introduced for every pair of connected nodes. As a result, (1) is reformulated equivalently as:

$$\begin{aligned} \min_{\{f_i\}, \{w_{ij}\}} \quad & \tilde{O}_{ERM}(\{f_i\}_{i=1}^N, D_{all}) = \sum_{i=1}^{N} O(f_i, D_i) \\ \text{s.t.} \quad & f_i = w_{ij}, w_{ij} = f_j, \quad i \in \mathcal{N}, j \in \mathcal{V}_i \end{aligned} \tag{2}$$

where $O(f_i, D_i) = \frac{C}{B_i} \sum_{n=1}^{B_i} \mathscr{L}(y_i^n f_i^T x_i^n) + \frac{\rho}{N} R(f_i)$. The objective in (2) can be solved using ADMM. Let $\{f_i\}$ be the shorthand for $\{f_i\}_{i \in \mathcal{N}}$; let $\{w_{ij}, \lambda_{ij}^k\}$ be the shorthand for $\{w_{ij}, \lambda_{ij}^k\}_{i \in \mathcal{N}, j \in \mathcal{V}_i, k \in \{a,b\}}$, where λ_{ij}^a, λ_{ij}^b are dual variables corresponding to equality constraints $f_i = w_{ij}$ and $w_{ij} = f_j$ respectively. Then the augmented Lagrangian is as follows:

$$\begin{aligned} L_\eta(\{f_i\}, \{w_{ij}, \lambda_{ij}^k\}) &= \sum_{i=1}^{N} O(f_i, D_i) \\ + \sum_{i=1}^{N} \sum_{j \in \mathcal{V}_i} (\lambda_{ij}^a)^T (f_i - w_{ij}) &+ \sum_{i=1}^{N} \sum_{j \in \mathcal{V}_i} (\lambda_{ij}^b)^T (w_{ij} - f_j) \\ + \sum_{i=1}^{N} \sum_{j \in \mathcal{V}_i} \frac{\eta}{2} (||f_i - w_{ij}||_2^2 &+ ||w_{ij} - f_j||_2^2) \,. \end{aligned} \tag{3}$$

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

$$f_i(t+1) = \underset{f_i}{\text{argmin}}\, L_\eta(\{f_i\}, \{w_{ij}(t), \lambda_{ij}^k(t)\}) \,; \tag{4}$$

$$w_{ij}(t+1) = \underset{w_{ij}}{\text{argmin}}\, L_\eta(\{f_i(t+1)\}, \{w_{ij}, \lambda_{ij}^k(t)\}) \,; \tag{5}$$

$$\lambda_{ij}^a(t+1) = \lambda_{ij}^a(t) + \eta(f_i(t+1) - w_{ij}(t+1)) \,; \tag{6}$$

$$\lambda_{ij}^b(t+1) = \lambda_{ij}^b(t) + \eta(w_{ij}(t+1) - f_j(t+1)) \,. \tag{7}$$

Using Lemma 3 in (Forero et al., 2010), if dual variables $\lambda_{ij}^a(t)$ and $\lambda_{ij}^b(t)$ are initialized to zero for all node pairs (i, j), then $\lambda_{ij}^a(t) = \lambda_{ij}^b(t)$ and $\lambda_{ij}^k(t) = -\lambda_{ji}^k(t)$ will hold for all iterations with $k \in \{a, b\}, i \in \mathcal{N}, j \in \mathcal{V}_i$.

Let $\lambda_i(t) = \sum_{j \in \mathcal{V}_i} \lambda_{ij}^a(t) = \sum_{j \in \mathcal{V}_i} \lambda_{ij}^b(t)$, then the ADMM iterations (4)-(7) can be simplified as:

$$\begin{aligned} f_i(t+1) = \underset{f_i}{\text{argmin}} \{ & O(f_i, D_i) + 2\lambda_i(t)^T f_i \\ & + \eta \sum_{j \in \mathcal{V}_i} ||\frac{1}{2}(f_i(t) + f_j(t)) - f_i||_2^2 \} \,; \end{aligned} \tag{8}$$

$$\lambda_i(t+1) = \lambda_i(t) + \frac{\eta}{2} \sum_{j \in \mathcal{V}_i} (f_i(t+1) - f_j(t+1)) \,. \tag{9}$$

2.3. Differential Privacy

Differential privacy (Dwork, 2006) can be used to measure the privacy risk of each individual sample in the dataset quantitatively. Mathematically, a randomized algorithm $\mathscr{A}(\cdot)$ taking a dataset as input satisfies ε-differential privacy if for any two datasets D, $\hat{D}$ differing in at most one data point, and for any set of possible outputs $S \subseteq \text{range}(\mathscr{A})$, $\Pr(\mathscr{A}(D) \in S) \leq \exp(\varepsilon)\Pr(\mathscr{A}(\hat{D}) \in S)$ holds. We call two datasets differing in at most one data point as neighboring datasets. The above definition suggests that for a sufficiently small ε, 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.

2.4. Private ADMM proposed in (Zhang & Zhu, 2017)

Two randomizations were proposed in (Zhang & Zhu, 2017): (i) dual variable perturbation, where each node i adds a random noise to its dual variable $\lambda_i(t)$ before updating its primal variable $f_i(t)$ using (8) in each iteration; and (ii) primal variable perturbation, where after updating primal variable $f_i(t)$, each node adds a random noise to it before broadcasting to its neighbors. Both were evaluated for a single iteration for a fixed privacy constraint. As we will see later in numerical experiments, the privacy loss accumulates significantly when inspected over multiple iterations.

In contrast, in this study we will explore the use of the penalty parameter η to provide privacy. In particular, we will allow this to be private information to every node, i.e., each decides its own η in every iteration and it is not exchanged among the nodes. Below we will begin by modifying the ADMM to accommodate private penalty terms.

3. Modified ADMM (M-ADMM)

3.1. Making η a node's private information

Conventional ADMM (Boyd et al., 2011) requires that the penalty parameter η be fixed and equal to the dual updating step size for all nodes in all iterations. Varying the penalty parameter to accelerate convergence in ADMM has been proposed in the literature. For instance, (He et al., 2002; Magnússon et al., 2014; Aybat & Iyengar, 2015; Xu et al., 2016) vary this penalty parameter in every iteration but keep it the same for different equality constraints in (2). In (Song et al., 2016; Zhang & Wang, 2017) this parameter varies in each iteration and is allowed to differ for different equality constraints. However, all of these modifications are based on the original ADMM (Eqn. (4)-(7)) and not on the simplified version (Eqn. (8)-(9)); the significance of this difference is discussed below in the context of privacy requirement. Moreover, we will decouple $\eta_i(t+1)$ from the dual updating step size, denoted as θ below. For simplicity, θ is fixed for all nodes in our analysis, but can also be private information as we show in numerical experiments.

First consider replacing η with $\eta_{ij}(t+1)$ in Eqn. (4)-(5) of the original ADMM (as is done in (Song et al., 2016; Zhang & Wang, 2017)) and replacing η with θ in Eqn. (6)-(7); we obtain the following:

$$f_i(t+1) = \underset{f_i}{\text{argmin}} \{O(f_i, D_i) + 2\lambda_i(t)^T f_i + \sum_{j\in\mathscr{V}_i} \frac{\eta_{ij}(t+1)+\eta_{ji}(t+1)}{2} ||\frac{1}{2}(f_i(t)+f_j(t)) - f_i||_2^2\} ;$$

$$\lambda_i(t+1) = \lambda_i(t) + \frac{\theta}{2}\sum_{j\in\mathscr{V}_i}(f_i(t+1) - f_j(t+1)) .$$

This however violates our requirement that $\eta_{ji}(t)$ be node j's private information since this is needed by node i to perform the above computation. To resolve this, we instead start from the simplified ADMM, modifying Eqn. (8)-(9):

$$f_i(t+1) = \underset{f_i}{\text{argmin}} \{O(f_i, D_i) + 2\lambda_i(t)^T f_i + \eta_i(t+1)\sum_{j\in\mathscr{V}_i} ||f_i - \frac{1}{2}(f_i(t)+f_j(t))||_2^2 \} ; \quad (10)$$

$$\lambda_i(t+1) = \lambda_i(t) + \frac{\theta}{2}\sum_{j\in\mathscr{V}_i}(f_i(t+1) - f_j(t+1)) , \quad (11)$$

where $\eta_i(t+1)$ is now node i's private information. Indeed $\eta_i(t+1)$ is no longer purely a penalty parameter related to any equality constraint in the original sense. We will however refer to it as the private penalty parameter for simplicity. The above constitutes the M-ADMM algorithm.

3.2. Convergence Analysis

We next show that the M-ADMM (Eqn. (10)-(11)) converges to the optimal solution under a set of common technical assumptions. Our proof is based on the method given in (Ling et al., 2016).

***Assumption 1*:** Function $O(f_i, D_i)$ is convex and continuously differentiable in f_i, $\forall i$.

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

The KKT optimality condition of the primal update (10) is:

$$0 = \nabla O(f_i(t+1), D_i) + 2\lambda_i(t) + \eta_i(t+1)\sum_{j\in\mathscr{V}_i}(2f_i(t+1) - (f_i(t)+f_j(t))) . \quad (12)$$

We next rewrite (11)-(12) in matrix form. Define the adjacency matrix of the network $A \in \mathbb{R}^{N\times N}$ 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 $\nabla O(f_i(t), D_i)$ for $i \in \mathcal{N}$ into matrices, i.e.,

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

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

Let $V_i = |\mathcal{V}_i|$ be the number of neighbors of node i, and define the degree matrix $D = \mathbf{diag}([V_1; V_2; \cdots; V_N]) \in \mathbb{R}^{N\times N}$. Define for the t-th iteration a penalty-weighted matrix $W(t) = \mathbf{diag}([\eta_1(t); \eta_2(t); \cdots; \eta_N(t)]) \in \mathbb{R}^{N\times N}$. Then the matrix form of (11)-(12) are:

$$\nabla \hat{O}(\hat{f}(t+1), D_{all}) + 2\Lambda(t) + 2W(t+1)D\hat{f}(t+1) - W(t+1)(D+A)\hat{f}(t) = \mathbf{0}_{N\times d}\,; \quad (13)$$

$$2\Lambda(t+1) = 2\Lambda(t) + \theta(D-A)\hat{f}(t+1)\,. \quad (14)$$

Note that $D - A$ is the Laplacian matrix and $D + A$ is the signless Laplacian matrix of the network, with the following properties if the network is connected: (i) $D \pm A \succeq 0$ is positive semi-definite; (ii) Null$(D - A) = c\mathbf{1}$, i.e., every member in the null space of $D - A$ is a scalar multiple of $\mathbf{1}$ with $\mathbf{1}$ being the vector of all 1's (Kelner, 2007).

Let $\sqrt{X}$ denote the square root of a symmetric positive semi-definite (PSD) matrix X that is also symmetric PSD, i.e., $\sqrt{X}\sqrt{X} = X$. Define matrix $Y(t)$ such that $2\Lambda(t) = \sqrt{D-A}Y(t)$. Since $\Lambda(0) = \mathbf{zeros}(N, d)$, which is in the column space of $D - A$, this together with (14) imply that $\Lambda(t)$ is in the column space of $D - A$ and $\sqrt{D-A}$. This guarantees the existence of $Y(t)$. This allows us to rewrite (13)-(14) as:

$$\nabla \hat{O}(\hat{f}(t+1), D_{all}) + \sqrt{D-A}Y(t+1) + (W(t+1) - \theta I)(D-A)\hat{f}(t+1) + W(t+1)(D+A)(\hat{f}(t+1) - \hat{f}(t)) = \mathbf{0}_{N\times d}\,; \quad (15)$$

$$Y(t+1) = Y(t) + \theta\sqrt{D-A}\hat{f}(t+1)\,. \quad (16)$$

Lemma 3.1 *[First-order Optimality Condition (Ling et al., 2016)] Under Assumptions 1 and 2, the following two statements are equivalent:*

- $\hat{f}^* = [(f_1^*)^T; (f_2^*)^T; \cdots; (f_N^*)^T] \in \mathbb{R}^{N\times d}$ *is consensual, i.e.,* $f_1^* = f_2^* = \cdots = f_N^* = f_c^*$ *where* f_c^* *is the optimal solution to* (1).
- *There exists a pair* $(\hat{f}^*, Y^*)$ *with* $Y^* = \sqrt{D-A}X$ *for some* $X \in \mathbb{R}^{N\times d}$ *such that*

$$\nabla \hat{O}(\hat{f}^*, D_{all}) + \sqrt{D-A}Y^* = \mathbf{0}_{N\times d}\,; \quad (17)$$

$$\sqrt{D-A}\hat{f}^* = \mathbf{0}_{N\times d}\,. \quad (18)$$

Lemma 3.1 shows that a pair $(Y^*, \hat{f}^*)$ satisfying (17)(18) is equivalent to the optimal solution of our problem, hence the convergence of M-ADMM is proved by showing that $(Y(t), \hat{f}(t))$ converges to a pair $(Y^*, \hat{f}^*)$ satisfying (17)(18).

Theorem 3.1 *Consider the modified ADMM defined by* (10)-(11). *Let* $\{Y(t), \hat{f}(t)\}$ *be outputs in each iteration and* $(Y^*, \hat{f}^*)$ *a pair satisfying* (17)-(18). *Denote*

$$Z(t) = \begin{bmatrix} Y(t) \\ \hat{f}(t) \end{bmatrix} \in \mathbb{R}^{2N\times d}, \quad Z^* = \begin{bmatrix} Y^* \\ \hat{f}^* \end{bmatrix} \in \mathbb{R}^{2N\times d}$$

$$J(t) = \begin{bmatrix} \frac{I_{N\times N}}{\theta} & 0 \\ 0 & W(t)(D+A) \end{bmatrix} \in \mathbb{R}^{2N\times 2N}$$

Let $\langle\cdot,\cdot\rangle_F$ *be the Frobenius inner product of two matrices. We have*

$$\langle Z(t+1) - Z^*, J(t+1)(Z(t+1) - Z(t))\rangle_F \le 0\,. \quad (19)$$

If $\eta_i(t+1) \ge \eta_i(t) \ge \theta > 0$ *and* $\eta_i(t) < +\infty$, $\forall t, i$, *then* $(Y(t), \hat{f}(t))$ *converges to* $(Y^*, \hat{f}^*)$.

3.3. Convergence Rate Analysis

To further establish the convergence rate of modified ADMM, an additional assumption is used:

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

$$(f_i^1 - f_i^2)^T(\nabla O(f_i^1, D_i) - \nabla O(f_i^2, D_i)) \ge m_i||f_i^1 - f_i^2||_2^2$$

$$||\nabla O(f_i^1, D_i) - \nabla O(f_i^2, D_i)||_2 \le M_i||f_i^1 - f_i^2||_2 \quad (20)$$

where $m_i > 0$ is the strong convexity constant and $0 < M_i < +\infty$ is the Lipschitz constant.

Theorem 3.2 *Define* $D_m = \mathbf{diag}([m_1; m_2; \cdots; m_N]) \in \mathbb{R}^{N\times N}$ *and* $D_M = \mathbf{diag}([M_1^2; M_2^2; \cdots; M_N^2]) \in \mathbb{R}^{N\times N}$ *with* $m_i > 0$ *and* $0 < M_i < +\infty$ *as given in Assumption 3. Denote by* $||X||_J^2 = \langle X, JX\rangle_F$ *the Frobenius inner product of any matrix* X *and* JX; *denote by* $\sigma_{min}(\cdot)$ *and* $\sigma_{max}(\cdot)$ *the smallest nonzero, and the largest, singular values of a matrix, respectively.*

Let $\tilde{\sigma}_{max}(t) = \sigma_{max}(W(t)(D+A))$, $\bar{\sigma}_{max/min}(t) = \sigma_{max/min}((W(t) - \theta I)(D-A))$ *and* $\mu > 1$ *be an arbitrary constant. Consider any* $\delta(t)$ *that satisfies* (21)(22):

$$\frac{\delta(t)\mu^2\tilde{\sigma}_{max}(t)}{\theta\sigma_{min}(D-A)} \le 1 \quad (21)$$

and

$$\delta(t)(\frac{\mu\bar{\sigma}_{max}(t)^2\boldsymbol{I}_N+\mu^2 D_M}{\theta\sigma_{min}(D-A)(\mu-1)}+W(t)(D+A)) \preceq 2(W(t)-\theta I)(D-A)+2D_m\ . \tag{22}$$

If $\eta_i(t+1)\geq\eta_i(t)\geq\theta>0$ *and* $\eta_i(t)<+\infty$, $\forall t,i$, *then* $(Y(t),\hat{f}(t))$ *converges to* $(Y^*,\hat{f}^*)$ *in the following sense:*

$$(1+\delta(t))||Z(t)-Z^*||^2_{J(t)}\leq||Z(t-1)-Z^*||^2_{J(t)}\ .$$

Furthermore, a lower bound on $\delta(t)$ *is:*

$$\min\{\frac{\theta\sigma_{min}(D-A)}{\mu^2\tilde{\sigma}_{max}(t)},\frac{2m_o+2\bar{\sigma}_{min}(t)}{\frac{\mu^2M_O^2+\mu\bar{\sigma}_{max}(t)^2}{\theta\sigma_{min}(D-A)(\mu-1)}+\tilde{\sigma}_{max}(t)}\} \tag{23}$$

where $m_o=\min_{i\in\mathcal{N}}\{m_i\}$ *and* $M_O=\max_{i\in\mathcal{N}}\{M_i\}$.

Although Theorem 3.2 only gives a lower bound on the convergence rate $(1+\delta(t))$ of the M-ADMM, it reflects the impact of penalty $\{\eta_i(t)\}_{i=1}^N$ on the convergence. Since $\bar{\sigma}_{\max}(t)=\sigma_{\max}((W(t)-\theta I)(D-A))$ and $\tilde{\sigma}_{\max}(t)=\sigma_{\max}(W(t)(D+A))$, larger penalty results in larger $\bar{\sigma}_{\max}(t)$ and $\tilde{\sigma}_{\max}(t)$. By (23), the first term, $\frac{\theta\sigma_{\min}(D-A)}{\mu^2\tilde{\sigma}_{\max}(t)}$ is smaller when $\tilde{\sigma}_{\max}(t)$ is larger. The second term is bounded by $\frac{\theta\sigma_{\min}(D-A)(\mu-1)(2m_o+2\bar{\sigma}_{\min}(t))}{\mu\bar{\sigma}_{\max}(t)^2}$, which is smaller when $\bar{\sigma}_{\max}(t)$ is larger. Therefore, the convergence rate $1+\delta(t)$ decreases as $\{\eta_i(t)\}_{i=1}^N$ increase.

4. Private M-ADMM

In this section we present a privacy preserving version of M-ADMM. To begin, a random noise $\epsilon_i(t+1)$ with probability density proportional to $\exp\{-\alpha_i(t+1)||\epsilon_i(t+1)||_2\}$ is added to penalty term in the objective function of (10):

$$L_i^{priv}(t+1)=O(f_i,D_i)+2\lambda_i(t)^Tf_i +\eta_i(t+1)\sum_{j\in\mathcal{V}_i}||f_i+\epsilon_i(t+1)-\frac{1}{2}(f_i(t)+f_j(t))||_2^2 \tag{24}$$

To generate this noisy vector, choose the norm from the gamma distribution with shape d and scale $\frac{1}{\alpha_i(t+1)}$ and the direction uniformly, where d is the dimension of the feature space. Then node i's local result is obtained by finding the optimal solution to the private objective function:

$$f_i(t+1)=\underset{f_i}{\operatorname{argmin}}\ L_i^{priv}(t+1),\ \ i\in\mathcal{N}\ . \tag{25}$$

It is equivalent to (26) below when noise $\eta_i(t+1)V_i\epsilon_i(t+1)$ is added to the dual variable $\lambda_i(t)$:

$$\underset{f_i}{\operatorname{argmin}}\ \tilde{L}_i^{priv}(t+1)=\frac{C}{B_i}\sum_{n=1}^{B_i}\mathscr{L}(y_i^nf_i^Tx_i^n)+\frac{\rho}{N}R(f_i) +2(\lambda_i(t)+\eta_i(t+1)V_i\epsilon_i(t+1))^Tf_i +\eta_i(t+1)\sum_{j\in\mathcal{V}_i}||f_i-\frac{1}{2}(f_i(t)+f_j(t))||_2^2\ .$$

Further, if $\eta_i(t+1)=\eta=\theta,\forall i,t$, then the above is reduced to the dual variable perturbation in (Zhang & Zhu, 2017)[3].

Algorithm 1 Penalty perturbation (PP) method

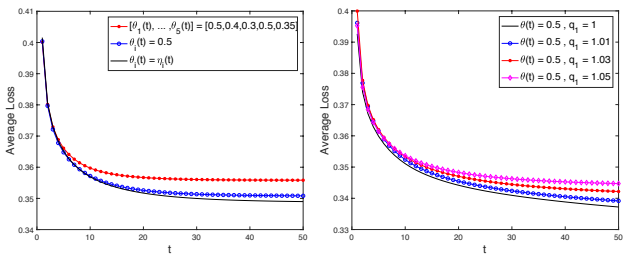
Parameter: Determine θ such that $2c_1<\frac{B_i}{C}(\frac{\rho}{N}+2\theta V_i)$ holds for all i.
Initialize: Generate $f_i(0)$ randomly and $\lambda_i(0)=\mathbf{0}_{d\times1}$ for every node $i\in\mathcal{N}$, $t=0$
Input: $\{D_i\}_{i=1}^N$, $\{\alpha_i(1),\cdots,\alpha_i(T)\}_{i=1}^N$
for $t=0$ **to** $T-1$ **do**
  for $i=1$ **to** N **do**
    Generate noise $\epsilon_i(t+1)\sim\exp(-\alpha_i(t+1)||\epsilon||_2)$
    Perturb the penalty term according to (24)
    Update primal variable via (25)
  end for
  for $i=1$ **to** N **do**
    Broadcast $f_i(t+1)$ to all neighbors $j\in\mathcal{V}_i$
  end for
  for $i=1$ **to** N **do**
    Update dual variable according to (11)
  end for
end for
Output: upper bound of the total privacy loss β

The complete procedure is shown in Algorithm 1, where the condition used to generate θ helps bound the worst-case privacy loss but is not necessary in guaranteeing convergence.

In a 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 formally state the differential privacy definition in this setting below.

Definition 4.1 *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* β*-differential privacy during* T *iterations if for any two datasets* $D_{all}=\cup_iD_i$ *and* $\hat{D}_{all}=\cup_i\hat{D}_i$, *differing in at*

[3]Only a single iteration is considered in (Zhang & Zhu, 2017) while imposing a privacy constraint. Since we consider the entire iterative process, we don't impose per-iteration privacy constraint but calculate the total privacy loss.



(a) Different $\eta_i(t)$ (b) Impact of increasing penalty

Figure 1. Convergence properties of M-ADMM.

most one data point, and for any set of possible outputs S during T iterations, the following holds:

$$\frac{Pr(\{f(t)\}_{t=0}^{T} \in S|D_{all})}{Pr(\{f(t)\}_{t=0}^{T} \in S|\hat{D}_{all})} \leq \exp(\beta)$$

We now state our main result on the privacy property of the penalty perturbation algorithm using the above definition. Additional assumptions on $\mathscr{L}(\cdot)$ and $R(\cdot)$ are used.

***Assumption 4*:** The loss function $\mathscr{L}$ is strictly convex and twice differentiable. $|\mathscr{L}'| \leq 1$ and $0 < \mathscr{L}'' \leq c_1$ with c_1 being a constant.

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

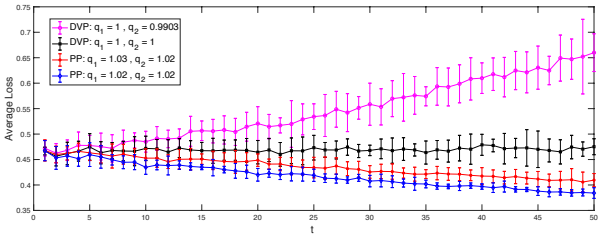
Theorem 4.1 *Normalize feature vectors in the training set such that $||x_i^n||_2 \leq 1$ for all $i \in \mathcal{N}$ and n. Then the private M-ADMM algorithm (PP) satisfies the β-differential privacy with*

$$\beta \geq \max_{i\in\mathcal{N}}\{\sum_{t=1}^{T} \frac{C(1.4c_1 + \alpha_i(t))}{\eta_i(t)V_iB_i}\}\,. \tag{26}$$

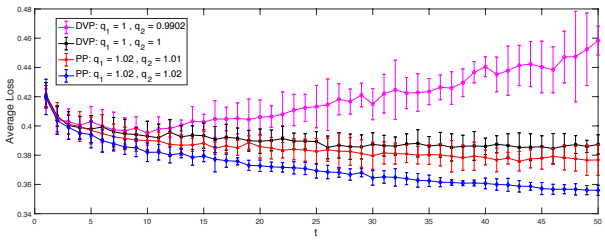
5. Numerical Experiments

We use the same dataset as (Zhang & Zhu, 2017), i.e., the *Adult* dataset from the UCI Machine Learning Repository (Lichman, 2013). 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.

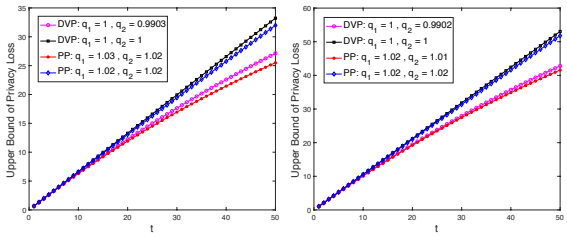
To preprocess the data, we (1) remove all individuals with missing values; (2) convert each categorical attribute (with m categories) to a binary vector of length m; (3) normalize columns (features) such that the maximum value of each column is 1; (4) normalize rows (individuals) such that its l_2 norm is at most 1; and (5) convert labels $\{\geq 50\text{k}, \leq 50\text{k}\}$ to $\{+1, -1\}$. After this preprocessing, the final data includes 45,223 individuals, each represented as a 105-dimensional vector of norm at most 1.



(a) Accuracy comparison: $\alpha(t) = 3q_2^{t-1}$



(b) Accuracy comparison: $\alpha(t) = 5q_2^{t-1}$



(c) Privacy: $\alpha(t) = 3q_2^{t-1}$ (d) Privacy: $\alpha(t) = 5q_2^{t-1}$

Figure 2. Compare accuracy and privacy, $\eta(t) = 0.5q_1^{t-1}$

We will use as loss function the logistic loss $\mathscr{L}(z) = \log(1 + \exp(-z))$, with $|\mathscr{L}'| \leq 1$ and $\mathscr{L}'' \leq c_1 = \frac{1}{4}$. 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 $L(t) := \frac{1}{N}\sum_{i=1}^{N}\frac{1}{B_i}\sum_{n=1}^{B_i}\mathscr{L}(y_i^n f_i(t)^T x_i^n)$ over the training set. We will measure the privacy of the algorithm by the upper bound $P(t) := \max_{i\in\mathcal{N}}\{\sum_{r=1}^{t}\frac{C(1.4c_1+\alpha_i(r))}{\eta_i(r)V_iB_i}\}$. The smaller $L(t)$ and $P(t)$, the higher accuracy and stronger privacy guarantee.

5.1. Convergence of M-ADMM

We consider a five-node network and assign each node the following private penalty parameters: $\eta_i(t) = \eta_i(1)q_i^{t-1}$ for node i, where $[\eta_1(1), \cdots, \eta_5(1)] = [0.55, 0.65, 0.6, 0.55, 0.6]$ and $[q_1, \cdots, q_5] = [1.01, 1.03, 1.1, 1.2, 1.02]$.

Figure 1(a) shows the convergence of M-ADMM under these parameters while using a fixed dual updating step size $\theta = 0.5$ across all nodes (blue curve). This is consistent with Theorem 3.1. As mentioned earlier, this step size can also be non-fixed (black) and different (red) for different nodes. In

Figure 1(b) we let each node use the same penalty $\eta_i(t) = \eta(t) = 0.5q_1^{t-1}$ and compare the results by increasing q_1, $q_1 \geq 1$. We see that increasing penalty slows down the convergence, and larger increase in q_1 slows it down even more, which is consistent with Theorem 3.2.

5.2. Private M-ADMM

We next inspect the accuracy and privacy of the penalty perturbation (PP) based private M-ADMM (Algorithm 1) and compare it with the dual variable perturbation (DVP) method proposed in (Zhang & Zhu, 2017). In this set of experiments, for simplicity of presentation we shall fix $\theta = 0.5$, let $\eta_i(t) = \eta(t) = \theta q_1^{t-1}$, and noise $\alpha_i(t) = \alpha(t) = \alpha(1)q_2^{t-1}$ for all nodes. We observe similar results when $\eta_i(t)$ and $\alpha_i(t)$ vary from node to node.

For each parameter setting, we perform 10 independent runs of the algorithm, and record both the mean and the range of their accuracy. Specifically, $L^l(t)$ denotes the average loss over the training dataset in the t-th iteration of the l-th experiment ($1 \leq l \leq 10$). The mean of average loss is then given by $L_{mean}(t) = \frac{1}{10}\sum_{l=1}^{10} L^l(t)$, and the range $L_{range}(t) = \max_{1\leq l\leq 10} L^l(t) - \min_{1\leq l\leq 10} L^l(t)$. The larger the range $L_{range}(t)$ 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 $L_{mean}(t)$ and $L_{range}(t)$ as vertical bars centered at $L_{mean}(t)$. 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.

Figure 2 compares PP (blue & red, with $\eta_i(t)$ increasing geometrically) with DVP (black & magenta, with $\eta_i(t) = \theta$, $\forall i, t$). We see that in both cases improved accuracy comes at the expense of higher privacy loss (from magenta to black under DVP, from red to blue under PP). However, we also see that with suitable choices of q_1, q_2, PP can outperform DVP significantly both in accuracy and in privacy (e.g., red outperforms magenta in both accuracy and privacy, and blue outperforms black in both accuracy and privacy).

We also performed experiments with the same dataset on larger networks with tens and hundreds of nodes and with samples evenly and unevenly spread across nodes. In both cases, convergence is attained and our algorithm continues to outperform (Zhang & Zhu, 2017) in a large network (see Figures 3 & 4). Since the privacy loss of the network is dominated by the node with the largest privacy loss and it increases as the number of samples in a node decreases (Theorem 4.1), the loss of privacy in a network with uneven sample size distributions is higher; note that this is a common issue with this type of analysis.

6. Discussion

Our numerical results show that increasing the penalty $\{\eta_i(t)\}_{i=1}^N$ over iterations can improve the algorithm's accuracy and privacy simultaneously. Below we provide some insight on why this is the case and discuss possible generalizations of our method.

6.1. Higher accuracy

When the algorithm is perturbed by random noise, which is necessary to achieve privacy, increasing the penalty parameters over iterations makes the algorithm more noise resistant. In particular, for the minimization in (25), larger $\eta_i(t+1)$ results in smaller updates of variables, i.e., smaller distance between $f_i(t+1)$ and $f_i(t)$. In the non-private case, since $f_i(t)$ always moves toward the optimum, smaller update slows down the process. In the private case, on the other hand, since a random noise is added to each update, $f_i(t)$ does not always move toward the optimum in each step. When the overall perturbation has a larger variance, it is more likely that $f_i(t)$ could move further away from the optimum in some iterations. Because larger $\eta_i(t)$ leads to smaller update, it helps prevent $f_i(t)$ from moving too far away from the optimum, thus stabilizing the algorithm (smaller $L_{range}(t)$).

6.2. Stronger privacy

First of all, more added noise means stronger privacy guarantee. Increasing $\eta_i(t)$ and $\alpha_i(t)$ in such a way that the overall perturbation $2\eta_i(t)V_i\epsilon_i(t)^T f_i(t)$ in (26) is increasing leads to less privacy loss, as shown in Figure 2. The noise resistance provided by an increasing $\eta_i(t)$ indeed allows larger noises to be added under PP without jeopardizing convergence as observed in Section 6.1.

More interestingly, keeping $\eta_i(t)$ private further strengthens privacy protection. Consider the following threat model: An attacker knows $\{(x_i^n, y_i^n)\}_{n=2}^{B_i}$ and $\{f_j(t)\}_{j\in\mathcal{V}_i\cup i}$ for all t, i.e., all data points except for the first data point of node i, as well as all intermediate results of node i and its neighbors. If the attacker also knows the dual updating step size θ and penalty parameter $\{\eta_i(t)\}_{t=1}^T$ of node i, it can then infer the unknown data point (x_i^1, y_i^1) with high confidence by combining the KKT optimality conditions from all iterations (see supplementary material for details). However, if the penalty parameters $\{\eta_i(t)\}_{t=1}^T$ are private to each node, then it is impossible for the attacker to infer the unknown data. Even if the attacker knows the participation of an individual, it remains hard to infer its features.

6.3. Generalization & comparison

The main contribution of this paper is the finding that increasing $\{\eta_i\}_{i=1}^N$ improves the algorithm's ability to resist

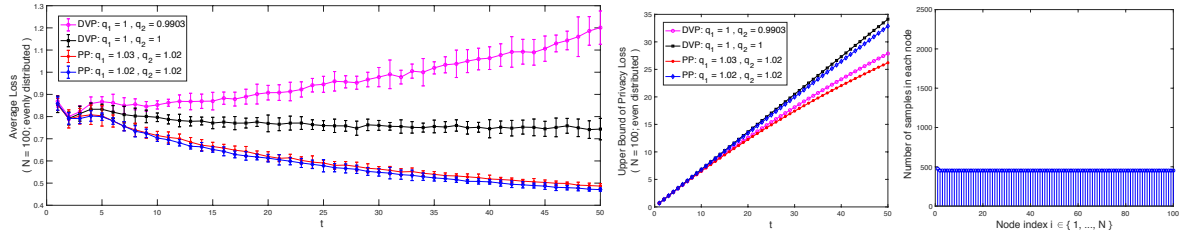


Figure 3. Compare accuracy and privacy for large network with data **evenly** spread across 100 nodes, $\eta(t) = 0.5q_1^{t-1}$, $\alpha(t) = 3q_2^{t-1}$

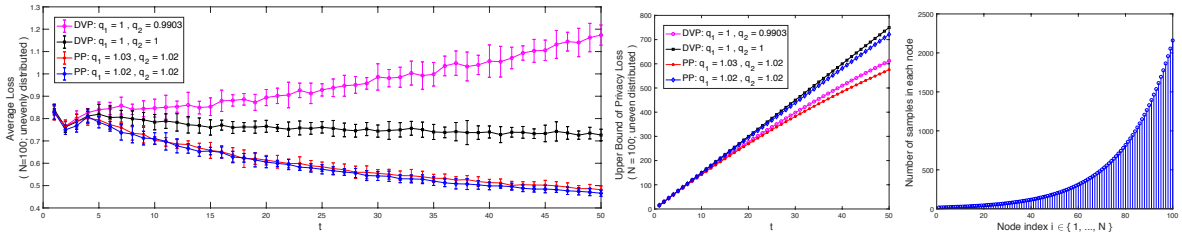


Figure 4. Compare accuracy and privacy for large network with data **unevenly** spread across 100 nodes, $\eta(t) = 0.5q_1^{t-1}$, $\alpha(t) = 3q_2^{t-1}$

noise: even though we increase noise in each iteration to improve privacy, the accuracy does not degrade significantly due to this increasing robustness, which improves the privacy-utility tradeoff. This property holds regardless of the noise distribution. While the present privacy analysis uses a similar framework as in (Chaudhuri et al., 2011; Zhang & Zhu, 2017) (objective perturbation with added Gamma noise), we can also use methods from other existing (centralized) ERM differentially private algorithms to every iteration in ADMM. For example, if we allow some probability ($\delta > 0$) of violating ϵ-differential privacy and adopt a weaker variant (ϵ, δ)-differential privacy, we can adopt methods from works such as (Kifer et al., 2012; Jain & Thakurta, 2014; Bassily et al., 2014), by adding Gaussian noise to achieve tighter bounds on privacy loss. However, as noted above, the robustness is improved as $\{\eta_i\}_{i=1}^N$ increases; thus the same conclusion can be reached that both privacy and accuracy can be improved.

This idea can also be generalized to other differentially private iterative algorithms. A key observation of our algorithm is that the overall perturbation ($2\eta_i(t)V_i\epsilon_i(t)^T f_i(t)$) is related to the parameter that controls the updating step size ($\eta_i(t)$). In general, if the algorithm is perturbed in each iteration with a quantity $\phi(\epsilon, \xi)$, which is a function of added noise ϵ and some parameter ξ that controls the step size, such that the resulting step size and $\phi(\epsilon, \xi)$ move in opposite directions (i.e., decreasing step size increases the $\phi(\epsilon, \xi)$), then it is possible to simultaneously improve both accuracy and privacy by varying ξ to decrease the step size over time.

Interestingly, in a differentially private (sub)gradient-based distributed algorithm (Huang et al., 2015), the step size and the overall perturbation move in the same direction (i.e., decreasing step size decreases perturbation). The reason for this difference is that under this subgradient-based algorithm, the sensitivity of the algorithm decreases with decreasing step size, which in turn leads to privacy constraint being satisfied with smaller perturbation. In contrast, for ADMM the sensitivity of the algorithm is independent of the step size, and the perturbation actually needs to increase to improve privacy guarantee; the decreasing step size acts to compensate for this increase in noise to maintain accuracy, as discussed in Section 6.1.

This issue of step size never arises in the study of (Zhang & Zhu, 2017) because the analysis is only for a single iteration; however, as we have seen doing so leads to significant total privacy loss over many iterations.

7. Conclusions

This paper presents a penalty-perturbation idea to introduce privacy preservation in iterative algorithms. We showed how to modify an ADMM-based distributed algorithm to improve privacy without compromising accuracy. The key idea is to add a perturbation correlated to the step size so that they change in opposite directions. Applying this idea to other iterative algorithms can be part of the future work.

Acknowledgements

This work is supported by the NSF under grants CNS-1422211, CNS-1646019, CNS-1739517.

References

Aybat, N. S. and Iyengar, G. An alternating direction method with increasing penalty for stable principal component pursuit. *Computational Optimization and Applications*, 61(3):635–668, 2015.

Bassily, R., Smith, A., and Thakurta, A. Differentially private empirical risk minimization: Efficient algorithms and tight error bounds. *arXiv preprint arXiv:1405.7085*, 2014.

Bellet, A., Guerraoui, R., Taziki, M., and Tommasi, M. *Fast and Differentially Private Algorithms for Decentralized Collaborative Machine Learning*. PhD thesis, INRIA Lille, 2017.

Bianchi, P., Hachem, W., and Iutzeler, F. A stochastic primal-dual algorithm for distributed asynchronous composite optimization. In *Signal and Information Processing (GlobalSIP), 2014 IEEE Global Conference on*, pp. 732–736. IEEE, 2014.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

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

Dwork, C. Differential privacy. In *Proceedings of the 33rd International Conference on Automata, Languages and Programming - Volume Part II*, ICALP'06, pp. 1–12, Berlin, Heidelberg, 2006. Springer-Verlag.

Forero, P. A., Cano, A., and Giannakis, G. B. Consensus-based distributed support vector machines. *Journal of Machine Learning Research*, 11(May):1663–1707, 2010.

Hale, M. and Egerstedty, M. Differentially private cloud-based multi-agent optimization with constraints. In *American Control Conference (ACC), 2015*, pp. 1235–1240. IEEE, 2015.

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

He, B., Liao, L.-Z., Han, D., and Yang, H. A new inexact alternating directions method for monotone variational inequalities. *Mathematical Programming*, 92(1):103–118, 2002.

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

Jain, P. and Thakurta, A. G. (near) dimension independent risk bounds for differentially private learning. In *International Conference on Machine Learning*, pp. 476–484, 2014.

Kelner, J. An algorithmist's toolkit, 2007. URL http://bit.ly/2C4yRCX.

Kifer, D., Smith, A., and Thakurta, A. Private convex empirical risk minimization and high-dimensional regression. In *Conference on Learning Theory*, pp. 25–1, 2012.

Lichman, M. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Ling, Q. and Ribeiro, A. Decentralized linearized alternating direction method of multipliers. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 5447–5451. IEEE, 2014.

Ling, Q., Liu, Y., Shi, W., and Tian, Z. Weighted admm for fast decentralized network optimization. *IEEE Transactions on Signal Processing*, 64(22):5930–5942, 2016.

Lobel, I. and Ozdaglar, A. Distributed subgradient methods for convex optimization over random networks. *IEEE Transactions on Automatic Control*, 56(6):1291–1306, 2011.

Magnússon, S., Weeraddana, P. C., Rabbat, M. G., and Fischione, C. On the convergence of an alternating direction penalty method for nonconvex problems. In *Signals, Systems and Computers, 2014 48th Asilomar Conference on*, pp. 793–797. IEEE, 2014.

Nedic, A. and Ozdaglar, A. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

Nedic, A., Olshevsky, A., Ozdaglar, A., and Tsitsiklis, J. N. Distributed subgradient methods and quantization effects. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pp. 4177–4184. IEEE, 2008.

Shi, W., Ling, Q., Yuan, K., Wu, G., and Yin, W. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Trans. Signal Processing*, 62(7): 1750–1761, 2014.

Song, C., Yoon, S., and Pavlovic, V. Fast admm algorithm for distributed optimization with adaptive penalty. In *AAAI*, pp. 753–759, 2016.

Wei, E. and Ozdaglar, A. Distributed alternating direction method of multipliers. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pp. 5445–5450. IEEE, 2012.

Xu, Z., Figueiredo, M. A., and Goldstein, T. Adaptive admm with spectral penalty parameter selection. *arXiv preprint arXiv:1605.07246*, 2016.

Zhang, C. and Wang, Y. Privacy-preserving decentralized optimization based on admm. *arXiv preprint arXiv:1707.04338*, 2017.

Zhang, R. and Kwok, J. Asynchronous distributed admm for consensus optimization. In *International Conference on Machine Learning*, pp. 1701–1709, 2014.

Zhang, T. and Zhu, Q. Dynamic differential privacy for admm-based distributed classification learning. *IEEE Transactions on Information Forensics and Security*, 12 (1):172–187, 2017.