A Private and Finite-Time Algorithm for Solving a Distributed System of Linear Equations

Shripad Gade   Ji Liu   Nitin H. Vaidya

Abstract— **This paper studies a system of linear equations, denoted as $Ax = b$, which is horizontally partitioned (rows in A and b) and stored over a network of m devices connected in a fixed directed graph. We design a fast distributed algorithm for solving such a partitioned system of linear equations, that additionally, protects the privacy of local data against an honest-but-curious adversary that corrupts at most τ nodes in the network. First, we present TITAN, privaTe fInite Time Average coNsensus algorithm, for solving a general average consensus problem over directed graphs, while protecting statistical privacy of private local data against an honest-but-curious adversary. Second, we propose a distributed linear system solver that involves each agent/devices computing an update based on local private data, followed by private aggregation using TITAN. Finally, we show convergence of our solver to the least squares solution in finite rounds along with statistical privacy of local linear equations against an honest-but-curious adversary provided the graph has weak vertex-connectivity of at least $\tau+1$. We perform numerical experiments to validate our claims and compare our solution to the state-of-the-art methods by comparing computation, communication and memory costs.**

I. Introduction

Consider a system of linear equations,

$$Ax = b, \tag{1}$$

where, $x \in \mathbb{R}^n$ is the $n-$dimensional solution to be learned, and $A \in \mathbb{R}^{p \times n}$, $b \in \mathbb{R}^{p \times 1}$ encode p linear equations in $n-$variables. The system of linear equations is horizontally partitioned and stored over a network of m devices. Each device $i \in \{1, \ldots, m\}$ has access to p_i linear equations denoted by,

$$A_i x = b_i, \tag{2}$$

where, $A_i \in \mathbb{R}^{p_i \times n}$ and $b_i \in \mathbb{R}^{p_i \times 1}$. For instance, in Fig. 1 we show a network of $m = 5$ nodes and horizontal partitioning of $p = 15$ linear equations in $n = 5$ variables using colored blocks. In this paper, we consider an honest-but-curious adversary that corrupts at most τ devices/nodes in the network and exploits observed information to infer private data. We are interested in designing a fast, distributed algorithm that solves problem (1), while, protecting privacy of local information (A_i, b_i) against such an honest-but-curious adversary.
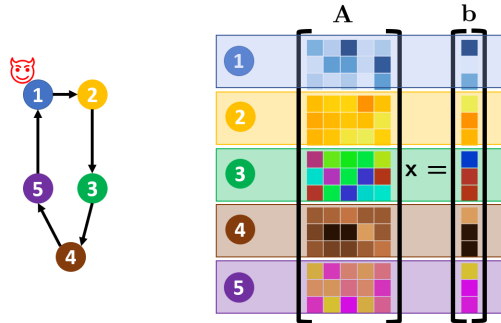
Fig. 1: Directed network with $m = 5$ nodes and 1 adversary. A and b are horizontally partitioned and stored at m nodes. Local information is private to the nodes.

Solving a system of linear algebraic equations is a fundamental problem that is central to analysis of electrical networks, sensor networking, supply chain management and filtering [1]–[3]. Several of these applications involve linear equations being stored at geographically separated devices/agents that are connected via a communication network. The geographic separation between agents along with communication constraints and unavailability of central servers necessitates design of distributed algorithms. Recently, several articles have proposed distributed algorithms for solving (1), [4]–[10] to name a few. In this work, we specifically focus on designing private methods that protect sensitive and private linear equations at each device/agent.

Literature has explored several approaches to solving a distributed system of linear equations. Authors in [2] formulated the problem as a parameter estimation task. Consensus or gossip based distributed estimation algorithms are then used to solve (1). Interleaved consensus and local projection based methods are explored in [4]–[6]. These direct methods, involving feasilble iterates that move only along the null space of local coefficient matrix A_i, converge exponentially fast. One can also view solving (1) as a constrained consensus [11] problem, where agents attempt to agree to a variable x such that local equations at each agent are satisfied. Problem (1) can also be formulated as a convex optimization problem, specifically linear regression, and solved using plethora of distributed optimization methods as explored in [7]. Authors augment their optimization based algorithms with a finite-time decentralized consensus scheme to achieve finite-time convergence of iterates to the solution. In comparison, our approach is not incremental and only needs two steps – (a) computing local updates, followed by, (b) fast aggregation and exact solution computation. Our

algorithm converges to the unique least squares solution in finite-time and additionally guarantees information-theoretic privacy of local data/equations (A_i, b_i).

Few algorithms focus on privacy of local equations (A_i, b_i). In this paper, we design algorithms with provable privacy properties. One can leverage vast private optimization literature by reformulating problem (1) as a least-squares regression problem and use privacy preserving optimization algorithms [12]–[17] on the resulting strongly convex cost function. Differential privacy is employed in [14], [17] for distributed convex optimization, however, it suffers from a fundamental privacy - accuracy trade-off [16]. Authors in [15], [18] use partially homomorphic encryption for privacy. However, these methods incur high computational costs and unsuitable for high dimensional problems. Secure Multiparty Computation (SMC) based method for privately solving system of linear equations is proposed in [19], however, this solution requires a central Crypto System Provider for generating garbled circuits. In this work, we design a purely distributed solution. Liu *et al.* propose a private linear system solver in [20], [21], however, as we discuss in Section IV-A, our algorithm is faster, requiring fewer iterations. Our prior work [12], [13] proposes non-identifiability over equivalent problems as a privacy definition and algorithms to achieve it. It admits privacy and accuracy guarantees simultaneously, however, it uses a weaker adversary model that does not know distribution of noise/perturbations used by agents.

In this paper, we consider a stronger definition of privacy viz. statistical privacy from [22], [23]. This definition of privacy allows for a stronger adversary that knows distribution of random numbers used by agents and has unbounded computational capabilities. We call this definition information-theoretic because additional observations do not lead to incremental improvement of adversarial knowledge about private data. Algorithms in [22], [23] provide algorithms for private average consensus over undirected graphs. We generalize their work to solve the problem over directed graphs and show a superior finite-time convergence guarantee.

Our Contributions:

Algorithm: We present an algorithm, TITAN (priva**T**e f**I**nite **T**ime **A**verage co**N**sensus), that solves average consensus problem over directed graphs in finite-time, while protecting statistical privacy of private inputs. It involves a distributed *Obfuscation Step* to hide private inputs, followed by a distributed recovery algorithm to collect all perturbed inputs at each node. Agents then compute exact average/aggregate using the recovered perturbed inputs. We further leverage TITAN to solve Problem (1) in finite time with strong statistical privacy guarantees.

Convergence Results: We show that TITAN converges to the exact average in finite time that depends only on the number of nodes and graph diameter. We show that graph being strongly connected is sufficient for convergence. Moreover, algorithm needs to know only an *an upper bound* on the number of nodes and graph diameter. We do not require out-degree of nodes to be known, a limitation commonly observed in push-sum type methods [24], for solving average consensus over directed graphs. TITAN based solver converges to the unique least squares solution of (1) in finite time.

Privacy Results: We show that TITAN provides statistical privacy of local inputs as long as weak vertex connectivity of communication graph is at least $t + 1$. This condition is also necessary and hence tight. Our privacy guarantee implies that for any two problems (1) characterized by (A, b) and (A', b'), such that rows of (A, b) and (A', b') stored at corrupted nodes are same and least squares solution for both systems is identical, the distribution of observations by the adversary is same. This equivalently entails adversary learning very little (statistically) in addition to the solution.

II. Problem Formulation

Consider a group of m agents/nodes connected in a directed network. We model the directed communication network as a directed graph $\mathbf{G} = (\mathcal{V}, \mathcal{E})$, where, $\mathcal{V} = \{1, 2, \ldots, m\}$ denotes the set of nodes and $\mathcal{E}$ denote reliable, loss-less, synchronous and directed communication links.

Recall, we are interested in solving a system of linear equations, problem (1), that is horizontally partitioned and stored at m agents. Each agent i has access to private p_i linear equations in n variables denoted by (2). Equivalently our problem formulation states that each agent i has access to private data matrices (A_i, b_i) that characterize (2). In this work, we assume that our system of linear equations in (1) admits a unique least squares solution, i.e. $A^T A$ matrix is full rank. If an exact solution for (1) exists, then it matches the least squares solution. Let ls-sol(A, b) denote the unique least squares solution of $Ax = b$. We wish to compute $x^* \triangleq$ ls-sol(A, b) that solves the collective system $Ax = b$, while, protecting privacy of local data (A_i, b_i). Next, we discuss the adversary model, privacy definition and few preliminaries.

A. Adversary Model and Privacy Definition

We consider an honest-but-curious adversary, that follows the prescribed protocol, however, is interested in learning private information from other agents. The adversary can corrupt at most τ nodes and has access to all the information stored, processed and received by the corrupted nodes. Let us denote the corrupted nodes as $\mathcal{A}$. We assume the adversary and corrupted nodes have unbounded storage and computational capabilities.

The coefficients of linear equations encode private and sensitive information. In the context of robotic or sensor networks, the coefficients of linear equations conceal sensor observations and measurements – information private to agents. In the context of supply chain management and logistics, linear equations are used to optimize transport of raw-materials, and the coefficients of linear equations often leak business sensitive information about quantity and type of raw-materials/products being transported by a company. Mathematically, adversary seeks to learn local coefficient matrices (A_i, b_i) corresponding to any non-corrupt agent i.

Privacy requires that the observations made by the adversary do not leak significant information about the private inputs. We use the definition of information-theoretic or statistical privacy from [22], [23]. Let $\mathsf{View}_{\mathcal{A}}((A, b))$ be the random variable denoting the observations made by set of adversarial nodes $\mathcal{A}$ given private inputs (A, b). We formally define statistical privacy (from [22], [23]) as follows.

Definition 1. *A distributed protocol is $\mathcal{A}$-private if for all (A, b) and (A', b'), such that $A[i, :] = A'[i, :]$, $b[i] = b'[i]$ for all $i \in \mathcal{A}$, and $\mathsf{ls\text{-}sol}(A, b) = \mathsf{ls\text{-}sol}(A', b')$, the distributions of $\mathsf{View}_{\mathcal{A}}((A, b))$ and $\mathsf{View}_{\mathcal{A}}((A', b'))$ are identical.*

Intuitively, for all systems of equations (A', b'), such that linear equations stored at $\mathcal{A}$ are the same and $\mathsf{ls\text{-}sol}(A, b) = \mathsf{ls\text{-}sol}(A', b')$, observations made by the adversary will have the same distribution, making them statistically indistinguishable from adversary $\mathcal{A}$.

More, generally as discussed in [22], [23] for average consensus over private inputs $\{x_i\}_{i=1}^m$, an algorithm is $\mathcal{A}-$private, if for all inputs $\{x_i\}_{i=1}^m$ and $\{x_i'\}_{i=1}^m$, such that $x_i = x_i'$ for all $i \in \mathcal{A}$ and $\sum_{i=1}^m x_i = \sum_{i=1}^m x_i'$, the distributions of information observed by adversary is the same. In other words, probability density function of $\mathsf{View}_{\mathcal{A}}(\{x_i\}_{i=1}^m)$ and $\mathsf{View}_{\mathcal{A}}(\{x_i'\}_{i=1}^m)$ are same.

B. Notation and Preliminaries

For each node $i \in \mathcal{V}$, we define in-neighbor set, $\mathcal{N}_i^{in}$, as the set of all nodes that send information to node i; and out-neighbor set, $\mathcal{N}_i^{out}$, as the set of all nodes that receive information from node i. Let $\delta(\mathbf{G})$ denote the diameter of graph $\mathbf{G}$, and B denote the incidence matrix of graph $\mathbf{G}$. Let $\{\perp\}^m$ denote a $m-$dimensional empty vector. Let $\mathcal{U}(a, b)$ be uniform distribution over $[a, b)$.

Modular arithmetic, typically defined over a finite field of integers, involves numbers wrapping around when reaching a certain value. In this work, we use real numbers and define an extension of modular arithmetic over reals:

Definition 2. *Consider a real interval $[0, a) \in \mathbb{R}$. We define $\mathrm{mod}(x, a) \in [0, a)$ as the remainder obtained when x is divided by a and the quotient is an integer. That is, $\mathrm{mod}(x, a) = x - pa$ where p is the unique integer such that $\mathrm{mod}(x, a) \in [0, a)$.*

Modulo operator satisfies useful properties as detailed below. The proofs are easy and omitted for brevity.

Remark 1. *Modular arithmetic over reals satisfies following properties for all real numbers $\{y_i\}_i$ and integer q.*

1) $\mathrm{mod}(\sum_{i=1}^q y_i, a) = \mathrm{mod}(\sum_{i=1}^q \mathrm{mod}(y_i, a), a)$,
2) $\mathrm{mod}(-y_i, a) = a - \mathrm{mod}(y_i, a)$.

III. TITAN - PRIVATE AVERAGE CONSENSUS

In this section, we develop TITAN, an algorithm for solving distributed average consensus with provable statistical privacy and finite-time convergence. In Section IV, we will use TITAN to solve Problem (1) with statistical privacy of local data (A_i, b_i) and finite-time convergence to x^*.

Consider a simple average consensus problem over m agents connected using directed graph $\mathbf{G}$. Each node $i \in \mathcal{V}$ has access to private input $x_i \in [0, a)$. The objective is to compute average $(1/m) \sum_{i=1}^m x_i$, while, protecting statistical privacy of inputs x_i (see Section II-A).

TITAN involves an obfuscation step to hide private information and generate obfuscated inputs. This is followed by several rounds of Top-k consensus primitive for distributed recovery of perturbed inputs. Consequently, we exploit *modulo aggregate invariance* property of the obfuscation step and locally process perturbed inputs to arrive at desired average. The obfuscation step guarantees statistical privacy of inputs, while, the distributed recover and local computation of average are key to finite-time convergence of the algorithm. We detail each of the steps below.

Obfuscation Step

The *obfuscation step* is a distributed method to generate network correlated noise that vanishes under modulo operation over the aggregate. First, each node i sends uniform random noise $r_{ij} \sim \mathcal{U}[0, ma)$ to out-neighbors $\mathcal{N}_i^{out}$ and receives r_{li} from in-neighbors $l \in \mathcal{N}_i^{in}$ (Line 2, Algorithm 1).

Next, each agent i computes perturbation t_i using (3) (Line 3, Algorithm 1). Observe that due to the modulo operation, each perturbation t_i satisfies $t_i \in [0, ma)$.

Finally, agent i adds perturbation t_i to its private value x_i and performs a modulo operation about ma to get the perturbed input, $\tilde{x}_i$, as seen in (4). Observe that $\tilde{x}_i \in [0, ma)$.

We now show the *modulo aggregate invariance* property of the obfuscation mechanism described above. Notice that each noise r_{ij} is added by node j to get t_j (before modulo

---

Algorithm 1: TITAN($\{x_i\}_{i=1}^m, T, k, m$)

Input: $\{x_i\}_{i=1}^m$, where, $x_i \in [0, a) \; \forall i \in \mathcal{V}$, T, k
Output: $\overline{x} = (1/m) \sum_{i=1}^m x_i$

1 **Initialization** Node i initializes $r_i = \{\perp\}^m$

Obfuscation Step

2 Node i sends random numbers $r_{ij} \sim \mathcal{U}[0, ma)$ to each out-neighbor $j \in \mathcal{N}_i^{out}$
3 Node i constructs perturbation t_i,

$$t_i = \mathrm{mod}\left(\sum_{j \in \mathcal{N}_i^{in}} r_{ji} - \sum_{j \in \mathcal{N}_i^{out}} r_{ij}, ma\right) \quad (3)$$

4 Each node i perturbs private input,

$$\tilde{x}_i = \mathrm{mod}(x_i + t_i, ma). \quad (4)$$

Distributed Recovery using Top-k Primitive

5 Each node $i \in \mathcal{V}$ runs Top-k primitive $\lceil m/k \rceil$ times
6 **for** $t = 0, 1, \ldots, \lceil m/k \rceil - 1$ **do**
7 $\quad (r_i[tk + 1 : (t+1)k], I_i[tk + 1 : (t+1)k]) =$
$\quad$ Top-k$((\{x_i\}_{i=1}^m \setminus r_i[1 : tk], \{i\}_{i=1}^m \setminus I_i[1 : tk]), T)$
8 **end**
9 **Return** $\overline{x} = (1/m)\mathrm{mod}(\sum_{l=1}^m r_i[l], ma)$.

operation) and subtracted by node i to get t_i (before modulo operation). This gives us,

$$\mod\left(\sum_{i=1}^{m} t_i, ma\right)$$

$$\stackrel{(a)}{=} \mod\left(\sum_{i=1}^{m} \mod\left(\sum_{j \in \mathcal{N}_i^{in}} r_{ji} - \sum_{j \in \mathcal{N}_i^{out}} r_{ij}, ma\right), ma\right)$$

$$\stackrel{(b)}{=} \mod\left(\sum_{i=1}^{m} \left[\sum_{j \in \mathcal{N}_i^{in}} r_{ji} - \sum_{j \in \mathcal{N}_i^{out}} r_{ij}\right], ma\right)$$

$$\stackrel{(c)}{=} \mod(0, ma) = 0. \tag{5}$$

In the above expression, (a) follows from definition of t_i in (3), (b) follows from property 1 in Remark 1, and (c) is a consequence of perturbation design in (3). We call this as *modulo aggregate invariance* of the obfuscation step.

Distributed Recovery via Top-k Consensus Primitive

We perform distributed recovery of perturbed inputs, that is, we run a distributed algorithm to "gather" all perturbed inputs ($\{\tilde{x}_i\}_{i=1}^{m}$) at each node. After the completion of this step, each node i will have access to the entire set of perturbed inputs $\{\tilde{x}_i\}_{i=1}^{m}$. Top-k consensus primitive is a method to perform distributed recovery.

The Top-k consensus primitive is a distributed protocol for all nodes to agree on the largest k inputs in the network. In TITAN, we run the Top-k consensus primitive and store the resulting list of top-k perturbed inputs. We then run Top-k primitive again while excluding the perturbed inputs recovered from prior iterations. Executing Top-k consensus primitive successively $\lceil m/k \rceil$ times leads us to the list of all perturbed inputs in the network (Lines 5-8, Algorithm 1).

Top-k Consensus Primitive: Recall, each node i has access to a perturbed input $\tilde{x}_i$ and an unique identifier i. The Top-k consensus is a consensus protocol for nodes to agree over k largest input values and associated node id's with ties going to nodes with larger id. Formally, the algorithm results in each node agreeing on $\{(x_{(1)}, \ldots, x_{(k)}), (i_{(1)}, \ldots, i_{(k)})\}$, where, $x_{(1)} \geq \ldots \geq x_{(k)} \geq x_{(k+1)} \geq \ldots \geq x_{(m)}$ is the ordering of private inputs $\{x_j\}_{j=1}^{m}$ and $i_{(k)}$ denote the ids corresponding to $x_{(k)}$ for each k. Ties go to nodes with larger id, implying, if $x_{(j)} = x_{(j+1)}$ then $i_{(j)} > i_{(j+1)}$.

Each node i stores an estimate of k largest inputs and their id's denoted by L_i and ℓ_i respectively. These vectors, L_i and ℓ_i, are initialized with local private input and own agent id respectively (Line 1, Algorithm 2).

At each iteration $r = 1, \ldots, T$, agent i share their estimates L_i and ℓ_i to out-neighbors and receives L_j and ℓ_j from in-neighbors $j \in \mathcal{N}_i^{in}$ (Line 3, Algorithm 2). Agent i sets L_i^+ equal to the k largest values of available inputs, that is L_i and $\cup_{j \in \mathcal{N}_i^{in}} L_j$; and sets ℓ_i^+ as id's corresponding to the entries in L_i^+ (Lines 5,6, Algorithm 2). In case of ties, that is several agents having the same input, the tie goes to agent with larger id. Next, agents update the local estimate of top k entries as $L_i = L_i^+$ and $\ell_i = \ell_i^+$ (Line 8, Algorithm 2).

Algorithm 2: Top-k consensus: Top-k($\{(x_i, i)\}_{i=1}^{m}, T$)

Input: $\{(x_i, i)\}_{i=1}^{m}$, T, k
Output: $\{(x_{(1)}, \ldots, x_{(k)}), (i_{(1)}, \ldots, i_{(k)})\}$

1 **Initialization** Node i initializes two k dimensional vectors $L_i = \{x_i, \bot, \ldots, \bot\}$ & $\ell_i = \{i, \bot, \ldots, \bot\}$

2 **for** $t = 1, \ldots, T$ **do**
3   Node i sends L_i & ℓ_i to out-neighbors $\mathcal{N}_i^{out}$
    Each node $i \in \mathcal{V}$ performs:
4   **for** $r = 1, \ldots, k$ **do**
5       $L_i^+[r] = \max(L_i \cup_{j \in \mathcal{N}_i^{in}} L_j \setminus (\cup_{s=1}^{r-1} L_i^+[s]))$
6       $\ell_i^+[r] = $ id corresponding to $L_i[r]$
7       $\ldots$ Ties go to node with larger id
8       $L_i = L_i^+$ and $\ell_i = \ell_i^+$
9   **end**
10 **end**
11 **Return** $\{L_i, \ell_i\} \to \{(x_{(1)}, \ldots, x_{(k)}), (i_{(1)}, \ldots, i_{(k)})\}$

This process of selection of largest perturbed input and it's id is repeated T times. As stated in Theorem 1, each L_i and ℓ_i converge to the *Top-k* values and associated id's respectively provided that $T \geq \delta(\mathbf{G})$, the graph diameter.

Recall, that running Top-k consensus primitive $\lceil m/k \rceil$ times, successively, allows each node to recover perturbed inputs $\{\tilde{x}_i\}_{i=1}^{m}$. Agents then add the perturbed inputs recovered by Top-k consensus algorithm and exploit modulo aggregate invariance property of obfuscation step to exactly compute, $\overline{x} = (1/m)\mod(\sum_{l=1}^{m} r_i[l], ma)$.

A. Results and Discussion

Correctness Guarantee: We first begin by a correctness result for the Top-k consensus primitive. It is a consequence of convergence of max consensus over directed graphs.

Theorem 1 (Correctness of Top-k)**.** *If* $\mathbf{G}$ *is a strongly connected graph with diameter* $\delta(\mathbf{G})$ *and* $T \geq \delta(\mathbf{G})$, *then* $L_i = L_j = \{x_{(1)}, \ldots, x_{(k)}\}$, $\ell_i = \ell_j = \{I_{(1)}, \ldots, I_{(k)}\}$, *for each* $i, j \in \mathcal{V}$, *for the Top-k consensus algorithm.*

The result establishes a lower bound on parameter T for correctness of Top-k consensus primitive. If $\delta(\mathbf{G})$ is not exactly known, we can set T to be any upper bound on $\delta(\mathbf{G})$, without worrying about correctness.

The ability of Top-k protocol to recover k largest perturbed inputs leads us to the correctness guarantee for TITAN. As a consequence of Theorem 1, we can conclude that $\lceil m/k \rceil$ successive execution of Top-k primitive over perturbed inputs leads to recovery of all perturbed inputs at each node. Moreover, we use the aggregate invariant property of the obfuscation step, implying for all graphs $\mathbf{G}$,

$$\mod\left(\sum_{i=1}^{m} \tilde{x}_i, ma\right) = \sum_{i=1}^{m} x_i. \tag{6}$$

We prove this in Section VI. Consequently, perturbed inputs allows us to compute the correct aggregate and average. The following result formally states this result:

Theorem 2 (Correctness of TITAN). *If* **G** *is strongly connected and* $T \geq \delta(\mathbf{G})$, *then* TITAN *(Algorithm 1) converges to the exact average of inputs* $\overline{x} = \frac{1}{m} \sum_{i=1}^{m} x_i$ *in finite time given by* $T\lceil m/k \rceil$.

Note, for finite-time convergence, we only need **G** to be strongly connected. The time required for convergence is dependent only on the number of agents m [1], parameter T (an upper bound on graph diameter $\delta(\mathbf{G})$) and parameter k.

Privacy Guarantee: The privacy guarantee is a consequence of the obfuscation step used in TITAN. Let $\overline{\mathbf{G}} = (\overline{\mathcal{V}}, \overline{\mathcal{E}})$ be the undirected version of **G**. More specifically, $\overline{\mathbf{G}}$ has the same vertex set $\overline{\mathcal{V}} = \mathcal{V}$ but the edge set $\overline{\mathcal{E}}$ is obtained by taking all the edges in $\mathcal{E}$ and augmenting it with the reversed edges. Consequently, $\overline{\mathbf{G}}$ is undirected. We define *weak vertex-connectivity* of a directed graph **G** as the vertex-connectivity of its undirected variant $\overline{\mathbf{G}}$. Weak vertex-connectivity of **G** is $\kappa(\overline{\mathbf{G}})$, where, κ denotes the vertex-connectivity. We show that provided the weak vertex-connectivity of **G** is at least $\tau + 1$, TITAN preserves statistical privacy of input.

Theorem 3. *If weak vertex-connectivity* $\kappa(\overline{\mathbf{G}}) \geq \tau + 1$, *then* TITAN *is* $\mathcal{A}$*-private against any set of adversaries* $\mathcal{A}$ *such that* $|\mathcal{A}| \leq \tau$.

Note, we require **G** to be both strongly connected and possess weak vertex-connectivity of at least $\tau + 1$ for achieving both finite-time correctness and statistical privacy guarantees. Moreover, the weak vertex-connectivity condition is also necessary and can be showed by contradiction, similar to Proof of Theorem 2 in [25]. Consequently, the weak vertex-connectivity condition is *tight*.

Memory Costs: Top-k primitive requires each node to maintain vectors L_i and ℓ_i in addition to recovered pertrubed inputs. Overall the memory required per node is $(2k+m)d$ units, where, d is the dimension of input x_i. This is larger as compared to standard average consensus methods and ratio consensus methods that require d and $2d$ units respectively. Observe the trade-off between Memory Overhead and Convergence Time. As k increases from 1 to m, the convergence time decreases from Tm to T, while the memory overhead (per node) increases from $(2+m)d$ to $3md$.

Communication Costs: The obfuscation step requires node i to send $|\mathcal{N}_i^{out}|d$ messages. Moreover, Top-k algorithm involves exchange of L_i and ℓ_i by each node. This additionally requires $2k|\mathcal{N}_i^{out}|T\lceil m/k \rceil d$ messages in total. Together, the communication overhead for node i is $|\mathcal{N}_i^{out}|(2kT\lceil m/k \rceil + 1)d$. Total communication cost (per node) is largely independent of k, as total information exchanged over entire execution does not change with k.

---

[1]If agents know only an upper bound on $\tilde{m} \geq m$, then we can run TITAN on inputs $x_i = 1$ for each $i \in \mathcal{V}$ while using $\tilde{m}$ instead of m and $a = 2$. Finally, by computing $\text{mod}(\sum_i \tilde{x}_i, \tilde{m}a)$ we recover m exactly.

Comparison with FAIM: Oliva *et al.* propose FAIM, finite-time average-consensus by iterated max-consensus, in [26]. TITAN has several similarities with FAIM and we can recover a statistically private version of FAIM by setting $k = 1$. Our work, in addition to finite-time average consensus, is directed toward provably privacy of local information.

IV. PRIVATE SOLVER FOR SYSTEM OF LINEAR EQUATIONS

In this section, we develop a solver (Algorithm 3) employing TITAN to privately solve problem (1).

The least squares solution to system of linear equations (1) can be expressed in closed form as, $(A^T A)^{-1} A^T b$. If an exact solution to (1) exists then the least squares solution matches it. Moreover, as the equations are horizontally partitioned, we can rewrite,

$$A^T A = \sum_{i=1}^{m} A_i^T A_i \quad A^T b = \sum_{i=1}^{m} A_i^T b_i. \quad (7)$$

Consequently, solution to system of linear equations can be computed by privately aggregating $A_i^T A_i$ and $A_i^T b_i$ separately over the network and computing,

$$x^* = \left(\sum_{i=1}^{m} A_i^T A_i \right)^{-1} \left(\sum_{i=1}^{m} A_i^T b_i \right). \quad (8)$$

Privately computing x^* is equivalent to agents privately aggregating $A_i^T A_i$ and $A_i^T b_i$ over the directed network followed by locally computing x^* using Eq. (8).

We assume, w.l.o.g, that each entry in matrix local updates $A_i^T A_i$ and $A_i^T b_i$ lies in $[0, a)$, where, $a >$ largest entry in matrices $A_i^T A_i$ and $A_i^T b_i$. If this is not satisfied, we can add the same constant to each entry in the matrix and subtract it after computing aggregate. Next, in Algorithm 3, we run TITAN on each entry of matrices $\{A_i^T A_i\}_{i=1}^m$ to get update matrix X, run TITAN on each entry of vector $\{A_i^T b_i\}_{i=1}^m$ to get update vector Y. We select $T \geq \delta(\mathbf{G})$ and a parameter $k \leq m$ following the discussion in Section III-A. As a consequence of Theorem 2, the algorithms terminate in finite time and $X = \frac{1}{m} \sum_{i=1}^{m} A_i^T A_i$ and $Y = \frac{1}{m} \sum_{i=1}^{m} A_i^T b_i$. Finally, we use (8) to compute the least squares solution.

Algorithm 3: Private Solver for Problem (1)

Input: $\{A_i^T A_i, A_i^T b_i\}_{i=1}^m$ and
$\quad\quad$ $a >$ the largest entry of $[A_i^T A_i \; A_i^T b_i]$ over all
$\quad\quad$ $i \in \{1, 2, \ldots, m\}$

Output: x^*

Run **TITAN** over each entry of $A_i^T A_i$ and $A_i^T b_i$

1 Compute: $X = \text{TITAN}(\{A_i^T A_i\}_{i=1}^m, T, k, m)$
2 Compute: $Y = \text{TITAN}(\{A_i^T b_i\}_{i=1}^m, T, k, m)$
3 **Return** $x^* = (mX)^{-1}(mY)$.

A. Results and Discussion

Correctness and Privacy: Our solution (Algorithm 3) involves using TITAN on each entry of matrices, $A_i^T A_i$ and $A_i^T b_i$. As a consequence of Theorem 2, we know that provided $\mathbf{G}$ is strongly connected and parameter $T \geq \delta(\mathbf{G})$, we get accurate estimate of $\sum_{i=1}^m A_i^T A_i$ and $\sum_{i=1}^m A_i^T b_i$ in finite time. Using (8), we compute solution x^*, and as a result we have solved (1) accurately in finite time. From Theorem 3, provided $\kappa(\overline{\mathbf{G}}) \geq \tau + 1$, TITAN preserves the statistical privacy of local inputs $(A_i^T A_i, A_i^T b_i)$, equivalently preserves the statistical privacy of (A_i, b_i), against any adversary that corrupts $\mathcal{A}$ subject to $|\mathcal{A}| \leq \tau$.

Comparison with Relevant Literature: Liu *et al.* propose a privacy mechanism, an alternative to the obfuscation mechanism in TITAN, and augment it to gossip algorithms for privately solving average consensus. This private average consensus is used along with direct method [4] to arrive at a private linear system solver. However, it requires agents to reach complete consensus between successive direct projection based steps. This increases the number of iterations needed to solve the problem and significantly increases communication costs as the underlying method [4] is only linearly convergent. Distributed Recovery phase in TITAN also requires complete consensus, but we do it in finite time using Top-k primitive, and it is only performed once.

Authors in [7] propose a finite-time solver for solving (1). However, under this protocol an arbitrarily chosen node/agent observes states for all nodes, and the observations are used compute the exact solution. This algorithm was not designed to protect privacy of local information and consequently leads to large privacy violations by the arbitrarily chosen node. Algorithm 3 solves (1) in finite time, while additionally protecting statistical privacy of local equations. Moreover, the algorithm in [7] is computationally expensive – requiring matrix singularity checks and kernel space computation. In comparison, our algorithm is inexpensive and the most expensive step is matrix inversion in (8), that needs to be performed only once.

Direct methods [4], [6], constrained consensus applied to linear systems [11] and distributed optimization methods applied to linear regression [27]–[29] are only linearly convergent, while, we provide superior finite-time convergence guarantee. Finite-time convergence of underlying consensus is critical for statistically private mechanisms such as the one in TITAN and the algorithms from [22], [23]. These mechanisms rely on modulo arithmetic, if we add them to a linearly convergent solver which outputs in-exact average/aggregate, then performing modulo operation over the in-exact output, in final step, may arbitrarily amplify errors.

Improving Computational Efficiency: In our approach, each node needs to compute an inverse, $\left(\sum_{i=1}^m A_i^T A_i \right)^{-1}$, which takes $\mathcal{O}(n^3)$ computations. Note, this inverse is not performed on private data. We can reduce computational cost by allowing one node to perform the inversion followed by transmitting the solution to all nodes either by flooding protocol or sending it over a spanning-tree of $\mathbf{G}$.
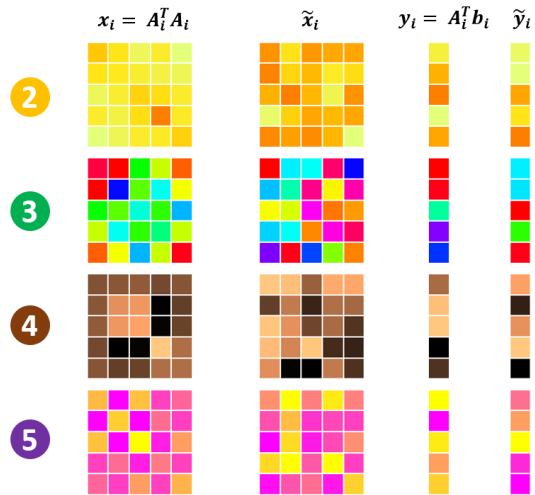

Fig. 2: Inputs $x_i = A_i^T A_i$ and $y_i = A_i^T b_i$ for $i \in \mathcal{V}$ and perturbed inputs $\tilde{x}_i$ and $\tilde{y}_i$. The perturbed inputs appear to be random and very different from the private inputs.

V. NUMERICAL EXPERIMENTS

In this section, we perform two numerical experiments to validate Algorithm 3. First, we conduct a simple simulation over the problem defined in Fig. 1 and show that update matrices $A_i^T A_i$ observed by the adversary appear to be random (Lemma 2). Second, we run a large scale experiment with synthetic data, with $m = 100$, $p = 10000$ and $n = 100$.

We have $m = 5$ nodes, with $p = 15$ linear equations in $n = 5$ variables being stored on the 5 nodes (3 each) as shown in Fig. 1. We generated A and b matrices by drawing their entries from a Gaussian distribution (mean = 0 and variance = 2) and verified $A^T A$ is full rank. We executed Algorithm 3 at each node. We select parameter $T = m \geq \delta(\mathbf{G}) = 4$, $k = m = 5$, and $a = 20$. The algorithm solves the problem exactly in $T = 6$ iterations. Note $\mathbf{G}$ is strongly connected and has a weak vertex-connectivity of 2. Consequently, both accuracy and statistical privacy are guaranteed by Theorems 2, 3. The perturbed update matrices generated after obfuscation step in TITAN are received by adversary node 1 and shown in Fig. 2. The color of each entry in the matrix represents its numerical value and Fig. 2 shows that the perturbed updates are starkly different from private updates and appear random.

Consider a large scale system with $m = 100$ agents and $p = 10000$ equations in $n = 100$ variables that are horizontally partitioned for each agent to have $p_i = 100$ equations $(\forall i \in \mathcal{V})$. The coefficients of the linear equations are synthetically generated via a Gaussian process and admit a unique least squares solution. The graph $\mathbf{G}$ is a directed ring with graph diameter $\delta(\mathbf{G}) = m - 1 = 99$ and $\kappa(\overline{\mathbf{G}}) = 2$. We run Algorithm 3 with parameters $T = m \geq \delta(\mathbf{G})$ and $k = 10$. We consider an honest-but-curious adversary that corrupts at most one agent and from $\kappa(\overline{\mathbf{G}}) = 2$ we guarantee statistical privacy of local data (A_i, b_i). The algorithm converges to the solution in 1000 iterations.

VI. ANALYSIS AND PROOFS

A. Convergence Analysis

We first prove correctness of Top-k consensus primitive.

Proof of Theorem 1. In the Top-k consensus algorithm, each agent/node tries to keep track of the largest k inputs observed till then. As $T \geq \delta(\mathbf{G})$, each one of the k largest inputs, i.e., $x_{(1)}, \ldots, x_{(k)}$, reaches each node in the network. Consequently, each nodes' local states L_i and ℓ_i converge to the k largest inputs in the network and associated id's. $\qquad\square$

Next, we prove correctness of TITAN in finite-time.

Proof of Theorem 2. The correctness result in Theorem 2 follows from two key statements: (1) TITAN output is exactly equal to $\overline{x} = (1/m)\sum_{i=1}^{m} x_i$, and (2) TITAN converges in finite time. We prove both the statements above.

(1) Recall that TITAN outputs $(1/m)\text{mod}(\sum_i \tilde{x}_{i=1}^{m}, ma)$. We use properties of modulo function to get,

$$\text{mod}(\sum_i \tilde{x}_{i=1}^{m}, ma) = \text{mod}(\sum_{i=1}^{m} \text{mod}(x_i + t_i, ma), ma)$$
$$\overset{(a)}{=} \text{mod}(\sum_{i=1}^{m}(x_i + t_i), ma) \overset{(b)}{=} \text{mod}(\sum_{i=1}^{m} x_i, ma) = \sum_{i=1}^{m} x_i.$$
$$(9)$$

Recall, (a) follows from Remark 1, (b) follows from $\sum_{i=1}^{m} t_i = 0$, and final equality follows from $x_i \in [0, a)$, $\sum_{i=1}^{m} x_i \in [0, ma)$ and Definition 2. We have proved TITAN computes the exact aggregate and consequently the average.

(2) The Top-k protocol involves $T = \tilde{\delta} \geq \delta(\mathbf{G})$ iterations where nodes share the largest k values that they have encountered. From Theorem 1, Top-k converges to the largest k perturbed inputs. We need to run $\lceil m/k \rceil$ iterations of Top-k consensus. Consequently, the total iterations for complete execution is $T\lceil m/k \rceil$. $\qquad\square$

B. Privacy Analysis

The privacy analysis presented here is similar in structure to [22], [23]. The key difference lies in the graph condition required for privacy. Recall $\overline{\mathbf{G}}$ is augmented $\mathbf{G}$. Specifically, $\overline{\mathbf{G}}$ has the same vertex set $\mathcal{V}$ but the edge set $\overline{\mathcal{E}}$ is obtained by taking all edges in $\mathcal{E}$ and augmenting it with reversed edges. Recall, the noise shared on edge $e = (i, j) \in \mathcal{E}$, is denoted as r_{ij} and it is uniformly distributed over $[0, ma)$. Perturbations t_i constructed using (3) can be written as $\mathbf{t} = \text{mod}(B\mathbf{r}, ma)$, where B is the incidence matrix of $\mathbf{G}$ and $\mathbf{r}$ is the vector of r_{ij} ordered according to the edge ordering in columns of B. If $\mathbf{G}$ is strongly connected then $\mathbf{G}$ is weakly connected and $\overline{\mathbf{G}}$ is connected.

Lemma 1. *If $\overline{\mathbf{G}}$ is connected then $\mathbf{t}$ is uniformly distributed over all points in $[0, ma)^m$ subject to the constraint $\text{mod}(\sum_{i=1}^{m} t_i, ma) = 0$.*

Proof. Recall that perturbation vector $\mathbf{t} = [t_i]$ can be written as $\mathbf{t} = \text{mod}(\sum_{j=1}^{|\mathcal{E}|} Br_j, ma)$, where the modulo operation is performed element-wise.

The connectivity of $\overline{\mathbf{G}}$ ensures that each t_i is a linear combination of uniform random perturbations (r_j's). We use the fact that $\text{mod}(a + b)$ is uniformly distributed if either a or b is uniformly distributed. Using above statements along with $r_j \sim \mathcal{U}[0, ma)$ we get $t_i \sim \mathcal{U}[0, ma)$. $\mathbf{t}$ is uniformly distributed over $[0, ma)^m$. Moreover, from (5), we know, $\text{mod}(1^T \mathbf{t}, ma) = 0$, is guaranteed if $\mathbf{t} = \text{mod}(B\mathbf{r}, ma)$. This completes the proof of Lemma 1. $\qquad\square$

Using the above property of perturbations, $\mathbf{t}$, we can show that perturbed inputs appear to be uniformly random, as described in the next lemma.

Lemma 2. *If $\overline{\mathbf{G}}$ is connected then the effective inputs $\tilde{x}_i$ are uniformly distributed over $[0, ma)$ subject to the constraint $\text{mod}(\sum_{i=1}^{m} \tilde{x}_i, ma) = \sum_i x_i$.*

Proof. Let $\tilde{X}, X$ and T represent the random vectors of agents obfuscated inputs, private inputs and perturbations respectively. Let $f_{\tilde{X}}, f_X$ and f_T denote the probability distribution of the respective random variables. Recall $\tilde{x}_i = \text{mod}(x_i + t_i)$ and the fact that t_i and x_i are independent. We have,

$$f_{\tilde{X}}(\tilde{x}|X = x) = f_T(T = \text{mod}(\tilde{x} - x))$$

As $\overline{\mathbf{G}}$ is connected and $\text{mod}(\sum_{i=1}^{m} \tilde{x}_i) = \sum_{i=1}^{m} x_i$ from (9), we know that $\text{mod}(\tilde{x} - x)$ is uniformly distributed over $[0, ma)^m$. And we have $f_T(T = \text{mod}(\tilde{x} - x)) = $ constant for any $\tilde{x} \in [0, ma)^m$, given $\text{mod}(1^T \tilde{x}) = 1^T x$. We have that $f_{\tilde{X}}(\tilde{x}|X = x) = $ constant for all $\tilde{x} \in [0, ma)^m$ implying that the perturbed input appears to be uniformly distributed over $[0, ma)^m$ subject to $\text{mod}(1^T \tilde{x}) = 1^T x$. $\qquad\square$

Recall, $\mathcal{A}$ is the set of honest-but-curious adversaries with $|\mathcal{A}| \leq \tau$. Also recall, adversarial nodes observe/store all information directly received and transmitted.

Proof of Theorem 3. Let $\mathcal{H} = \mathcal{V} \setminus \mathcal{A}$ denote the set of honest nodes. Let $\mathbf{G}_{\mathcal{H}} = (\mathcal{H}, \mathcal{E}_{\mathcal{H}})$ denote the subgraph induced by honest nodes, implying, $\mathcal{E}_{\mathcal{H}} \subseteq \mathcal{E}$ is the set of all edges from $\mathbf{G}$ that are incident on & from two honest nodes. Let $B_{\mathcal{H}}$ denote the oriented incidence matrix of graph $\mathbf{G}_{\mathcal{H}}$.

We know that vertex connectivity $\kappa(\overline{\mathbf{G}}) \geq \tau + 1$, implying that deleting any τ nodes from $\overline{\mathbf{G}}$ does not disconnect it. Implying, $\overline{\mathbf{G}}_{\mathcal{H}}$ is connected.

The information accessible to an adversary, defined as $\text{View}_{\mathcal{A}}$, consists of the private inputs of corrupted agents, perturbed inputs of honest agents and the random numbers transmitted or received by corrupted nodes.

$$\text{View}_{\mathcal{A}}(x) = \{\{\tilde{x}_i | i \in \mathcal{H}\}, \{x_i | i \in \mathcal{A}\}, \{r_{ij} | i \in \mathcal{A} \text{ or } j \in \mathcal{A}\}\}$$

For privacy, we prove that the probability distributions of $\text{View}_{\mathcal{A}}(x) = \text{View}_{\mathcal{A}}(x')$ for any two inputs x, x' such that $x_i = x_i'$ for all $i \in \mathcal{A}$ and $\sum_{i \in \mathcal{V}} x_i = \sum_{i \in \mathcal{V}} x_i'$.

Let the incidence matrix be partitioned as $B = [B_{\mathcal{H}} \ B_{\mathcal{A}}]$, where $B_{\mathcal{H}}$ are columns of B corresponding to edges in $\mathcal{E}_{\mathcal{H}}$ and $B_{\mathcal{A}}$ are columns of B corresponding to edges in $\mathcal{E}_{\mathcal{A}} = \mathcal{E} \setminus \mathcal{E}_{\mathcal{H}}$. Let B_e represent the eth column of B and $B_{i,e}$ represent the $[i, e]^{th}$ entry of matrix B. $\mathbf{r}_e$ denote the eth entry

of vector $\mathbf{r}$. Note that the perturbations can be expressed as, $t_i = \mathrm{mod}\left(\sum_{e \in \mathcal{E}_\mathcal{H} \cup \mathcal{E}_\mathcal{A}} B_{i,e} \mathbf{r}_e, ma\right)$ and equivalently as $t_i = \mathrm{mod}(t_i^\mathcal{H} + t_i^\mathcal{A}, ma)$, where, $t_i^\mathcal{H} = \mathrm{mod}\left(\sum_{e \in \mathcal{E}_\mathcal{H}} B_{i,e} \mathbf{r}_e, ma\right)$ and $t_i^\mathcal{A} = \mathrm{mod}\left(\sum_{e \in \mathcal{E}_\mathcal{A}} B_{i,e} \mathbf{r}_e, ma\right)$, following Remark 1.

Using Lemma 1 we can state the following. The values $\{t_i^\mathcal{H}\}_{i \in \mathcal{H}} = \{\mathrm{mod}(\sum_{e \in \mathcal{E}_\mathcal{H}} B_{i,e} \mathbf{r}_e, ma)\}_{i \in \mathcal{H}}$ lies in the span of columns of $B_\mathcal{H}$ and is uniformly distributed over $[0, ma)^{|\mathcal{H}|}$ subject to $\sum_{i \in \mathcal{H}} \mathrm{mod}(\sum_{e \in \mathcal{E}_\mathcal{H}} B_{i,e} \mathbf{r}_e, ma) = 0$ given that $\overline{\mathbf{G}}_\mathcal{H}$ is connected. Consequently, the masks $\{t_i\}_{i \in \mathcal{H}}$ are uniformly distributed over $[0, ma)^{|\mathcal{H}|}$ subject to $\mathrm{mod}(\sum_{i \in \mathcal{H}} t_i, ma) = -\mathrm{mod}(\sum_{i \in \mathcal{A}} t_i, ma)$, given $\overline{\mathbf{G}}_\mathcal{H}$ is connected, and given perturbations $\{\mathbf{r}_e\}_{e \in \mathcal{E}_\mathcal{A}}$.

Recall, $\mathbf{r}_{e \in \mathcal{E}_\mathcal{H}}$ are uniformly and independently distributed in $[0, ma)$ given values of $\{\mathbf{r}_e\}_{e \in \mathcal{E}_\mathcal{A}}$ and t_i for each $i \in \mathcal{A}$.

We use the fact that $\overline{\mathbf{G}}_\mathcal{H}$ is connected and Lemma 2, we get that, $\tilde{x}_i$ are uniformly distributed over $[0, ma)$ subject to $\mathrm{mod}(\sum_i \tilde{x}_i, ma) = \sum_i x_i$. Thus, if $X_\mathcal{H}$, $\tilde{X}_\mathcal{H}$ are random variables representing $\{x_i\}_{i \in \mathcal{H}}$ and $\{\tilde{x}_i\}_{i \in \mathcal{H}}$, then,

$$f_{\tilde{X}_\mathcal{H}}(\tilde{x}_\mathcal{H} | x_\mathcal{H}, \{\mathbf{r}_e\}_{e \in \mathcal{E}_\mathcal{A}}) = \mathrm{constant}$$

for all $\tilde{x}_\mathcal{H}$ in $[0, ma)^{|\mathcal{H}|}$ that satisfy, $\mathrm{mod}(\sum_{i \in \mathcal{H}} \tilde{x}_i, ma) = \mathrm{mod}(\sum_{i \in \mathcal{V}} x_i - \sum_{i \in \mathcal{A}} \tilde{x}_i, ma)$.

We combine this with the fact that the perturbations $\mathbf{r}_e$ are independent of inputs x, to get,

$$f_{\mathsf{View}_\mathcal{A}(x)}(\{\tilde{x}_\mathcal{H}, \{\mathbf{r}_e\}_{e \in \mathcal{E}_\mathcal{A}}\}) = f_{\mathsf{View}_\mathcal{A}(x')}(\{\tilde{x}_\mathcal{H}, \{\mathbf{r}_e\}_{e \in \mathcal{E}_\mathcal{A}}\})$$

$\forall x, x'$ such that, $x_i = x_i'$ $\forall i \in \mathcal{A}$ and $\sum_i x_i = \sum_i x_i'$. $\qquad \square$

VII. Conclusion

We presented TITAN, a finite-time, private algorithm for solving distributed average consensus. We show that TITAN converges to the average in finite-time that is dependent only on graph diameter and number of agents/nodes. It also protects statistical privacy of inputs against an honest-but-curious adversary that corrupts at most τ nodes in the network, provided weak vertex-connectivity of graph is at least $\tau + 1$. We use TITAN to solve a horizontally partitioned system of linear equations in finite-time, while, protecting statistical privacy of local equations against an honest-but-curious adversary.

References

[1] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.* IEEE, 2005, pp. 63–70.

[2] S. Kar, J. M. Moura, and K. Ramanan, "Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication," *IEEE Transactions on Information Theory*, vol. 58, no. 6, pp. 3575–3605, 2012.

[3] G. Williams, *Linear algebra with applications.* Jones & Bartlett Learning, 2017.

[4] S. Mou, J. Liu, and A. S. Morse, "A distributed algorithm for solving a linear algebraic equation," *IEEE Transactions on Automatic Control*, vol. 60, no. 11, pp. 2863–2878, 2015.

[5] J. Liu, S. Mou, and A. S. Morse, "Asynchronous distributed algorithms for solving linear algebraic equations," *IEEE Transactions on Automatic Control*, vol. 63, no. 2, pp. 372–385, 2017.

[6] J. Liu, A. S. Morse, A. Nedić, and T. Başar, "Exponential convergence of a distributed algorithm for solving linear algebraic equations," *Automatica*, vol. 83, pp. 37–46, 2017.

[7] T. Yang, J. George, J. Qin, X. Yi, and J. Wu, "Distributed least squares solver for network linear equations," *Automatica*, vol. 113, pp. 1087–98, 2020.

[8] D. Jakovetic, N. Krejic, N. K. Jerinkic, G. Malaspina, and A. Micheletti, "Distributed fixed point method for solving systems of linear algebraic equations," *arXiv preprint arXiv:2001.03968*, 2020.

[9] P. Wang, S. Mou, J. Lian, and W. Ren, "Solving a system of linear equations: From centralized to distributed algorithms," *Annual Reviews in Control*, 2019.

[10] S. S. Alaviani and N. Elia, "A distributed algorithm for solving linear algebraic equations over random networks," in *2018 IEEE Conference on Decision and Control (CDC).* IEEE, 2018, pp. 83–88.

[11] A. Nedić, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.

[12] S. Gade and N. H. Vaidya, "Private optimization on networks," in *2018 Annual American Control Conference (ACC).* IEEE, 2018, pp. 1402–1409.

[13] ——, "Privacy-preserving distributed learning via obfuscated stochastic gradients," in *2018 IEEE Conference on Decision and Control (CDC).* IEEE, 2018, pp. 184–191.

[14] Z. Huang, S. Mitra, and N. Vaidya, "Differentially private distributed optimization," in *Proceedings of the 2015 International Conference on Distributed Computing and Networking.* ACM, 2015, p. 4.

[15] C. Zhang, M. Ahmad, and Y. Wang, "Admm based privacy-preserving decentralized optimization," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 3, pp. 565–580, 2019.

[16] S. Han, U. Topcu, and G. J. Pappas, "Differentially private distributed constrained optimization," *IEEE Transactions on Automatic Control*, vol. PP, no. 99, pp. 1–1, 2016.

[17] M. T. Hale and M. Egerstedt, "Cloud-enabled differentially private multi-agent optimization with constraints," *IEEE Transactions on Control of Network Systems*, 2017.

[18] A. B. Alexandru, K. Gatsis, and G. J. Pappas, "Privacy preserving cloud-based quadratic optimization," in *Communication, Control, and Computing (Allerton), 2017 55th Annual Allerton Conference on.* IEEE, 2017, pp. 1168–1175.

[19] A. Gascón, P. Schoppmann, B. Balle, M. Raykova, J. Doerner, S. Zahur, and D. Evans, "Privacy-preserving distributed linear regression on high-dimensional data," *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 4, pp. 345–364, 2017.

[20] Y. Liu, J. Wu, I. R. Manchester, and G. Shi, "Gossip algorithms that preserve privacy for distributed computation part i: The algorithms and convergence conditions," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 4499–4504.

[21] ——, "Gossip algorithms that preserve privacy for distributed computation part ii: Performance against eavesdroppers," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 5346–5351.

[22] N. Gupta, J. Kat, and N. Chopra, "Statistical privacy in distributed average consensus on bounded real inputs," in *2019 American Control Conference (ACC).* IEEE, 2019, pp. 1836–1841.

[23] N. Gupta, J. Katz, and N. Chopra, "Privacy in distributed average consensus," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 9515–9520, 2017.

[24] C. N. Hadjicostis, N. H. Vaidya, and A. D. Domínguez-García, "Robust distributed average consensus via exchange of running sums," *IEEE Transactions on Automatic Control*, vol. 61, no. 6, pp. 1492–1507, 2015.

[25] S. Gade and N. H. Vaidya, "Private learning on networks: Part ii," *arXiv preprint arXiv:1703.09185*, 2017.

[26] G. Oliva, R. Setola, and C. N. Hadjicostis, "Distributed finite-time average-consensus with limited computational and storage capability," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 2, pp. 380–391, 2016.

[27] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[28] A. Nedić, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2597–2633, 2017.

[29] Y. Sun, A. Daneshmand, and G. Scutari, "Convergence rate of distributed optimization algorithms based on gradient tracking," *arXiv preprint arXiv:1905.02637*, 2019.