Privacy-Preserving Distributed Learning via Obfuscated Stochastic Gradients

Shripad Gade [1] and Nitin H. Vaidya

Abstract

Distributed (federated) learning has become a popular paradigm in recent years. In this scenario private data is stored among several machines (possibly cellular or mobile devices). These machines collaboratively solve a distributed optimization problem, using private data, to learn predictive models. The aggressive use of distributed learning to solve problems involving sensitive data has resulted in privacy concerns. In this paper we present a synchronous distributed stochastic gradient descent based algorithm that introduces privacy via gradient obfuscation in client-server model. We prove the correctness of our algorithm. We also show that obfuscation of gradients via additive and multiplicative perturbations provides privacy to local data against honest-but-curious adversaries.

I. Introduction

Machine learning has recently found applications in increasingly many fields ranging from personal preference predictions to finance and healthcare. For instance, banks and financial institutions can use persons' financial records and transaction history, to decide if a certain transaction is fraudulent, or to decide if they should increase the credit limits for a certain individual. Similarly hospitals and insurance providers use medical records to predict if re-hospitalization will be needed for specific patient and to model individual risk to certain disorders. Personal financial and medical records are highly sensitive and their privacy needs to be protected. As machine learning comes into common use it is becoming increasingly important to design algorithms and systems that will honor privacy considerations. In this work we present a synchronous distributed learning and optimization algorithm with privacy properties.

In a distributed learning scenario, the data is segregated among several machines, possibly mobile devices [1]. Clients compute updates based on local data and iteratively improve the model. This quest to find the best predictive model, essentially implies solving the following optimization problem,

$$\text{Find } x^* \in \arg\min \sum_{i=1}^{C} f_i(x),$$

where, $f_i(x)$ is the local objective function (loss function) of a client i, known only to client i ($1 \leq i \leq C$). The vector x parameterizes the predictive model and x^* represents the best "model parameters". Observe that often the data stored at a client is large and gradient update computation is very expensive. In such a scenario, one may replace the gradient update with an unbiased estimator of the gradient. This is usually done by dividing the dataset into several batches and then randomly picking a batch to represent the dataset for gradient computation. Such an unbiased estimate of gradient is termed stochastic gradient and we will be using it here.

Distributed optimization is often attractive due to the reduced communication requirements, since the agents (clients) communicate updates that are often much smaller in size than each agent's local dataset that characterizes its local objective (loss) function. Moreover, distributed methods provide scalability with respect to number of participants and naturally fit the fragmented and segregated data sources [2]–[4].

However, the updates transmitted by the clients often can give strong indications of the dataset (loss function) that generated the update and hence classical optimization algorithms are vulnerable to privacy violations [5].

In this paper we present POLAR-SGD (Private Optimization and Learning Algorithm - Stochastic Gradient Descent) that protects privacy by obfuscating the updates transmitted by the clients. In particular, we use correlated additive and multiplicative perturbations to obfuscate the stochastic gradient provided by clients. Introduction of perturbation helps clients improve the privacy of their information (as elaborated later in the paper) while the correlated nature of perturbations helps us maintain correctness.

Related Work: Several distributed optimization algorithms have been introduced in literature in recent years, including Sub-gradient Descent [6], [7], Dual Averaging [8], Incremental Algorithms [9], [10], Accelerated Gradient [11], ADMM [12] and EXTRA [13]. Convex distributed optimization has been studied in a variety of scenarios involving directed graphs [14], [15],

[1] (`gade3@illinois.edu`) and [2] (`nhv@illinois.edu`) are with ECE Department, and Coordinated Science Laboratory, at University of Illinois Urbana-Champaign. This research is supported in part by NSF awards 1421918 and 1610543, and Toyota InfoTechnology Center. Any opinions, findings, and conclusions or recommendations expressed here are those of the authors and do not necessarily reflect the views of the funding agencies or the U.S. government.

communication link failures and losses [16], asynchronous communication models [17], [18], and stochastic objective functions [7].

Several privacy-preserving optimization algorithms have been proposed in recent years. These methods can be broadly classified into cryptographic and non-cryptographic methods [19]. Cryptographic methods use cryptographic protocols to provide privacy [20]. Cryptographic protocols however incur high computational overheads and may not be suited for iterative algorithms. Differential privacy has emerged to be a popular solution in the past few years [5], [21]–[23]. Differential privacy involves adding special noise (Gaussian, Laplacian etc.) to the information exchanged so as to maximize the accuracy of the information exchanged while minimizing the probability of inferring the exact record that generated the information. Such methods however are bound by a fundamental trade-off between accuracy and privacy. Consequently, differentially private optimization algorithms do not converge to the optimum. Transformation is another popular non-cryptographic methods for obtaining privacy. It involves converting the problem into a new problem via algebraic transformations such that the solution of the new problem is the same as old problem [24], [25]. Time-correlated randomized perturbations are used in [26] and secure-multi party computation based correlated randomization is used in [27]–[29] to solve privacy-preserving distributed optimization.

Contributions: Our contributions are enumerated below-

- *Algorithm:* We present POLAR-SGD (Private Optimization and Learning Algorithm - Stochastic Gradient Descent). It is a synchronous protocol where clients use correlated additive and multiplicative perturbations to obfuscate the stochastic gradient. These obfuscated stochastic gradients are uploaded to multiple parameter servers, who then use consensus iteration and projected stochastic gradient descent to learn predictive models.
- *Correctness and Privacy:* We prove convergence of POLAR-SGD under a few conditions on the perturbations. While other perturbation based algorithms such as differential privacy incur accuracy loss due to privacy, we show that our algorithm solves the optimization problem correctly. We also discuss privacy characteristics of POLAR-SGD.

II. Problem Formulation

We will first review the system architecture, distributed learning problem and some preliminaries on distributed learning in client-server models.

A. System Architecture

System architecture is depicted in Figure 1. Our system consists of S parameter servers (also referred to as "servers") and C clients. As discussed before, client i represents a computer that has access to its private data and corresponding loss function $f_i(x)$. Parameter server J maintains a copy of the model, x^J.

The clients receive latest model parameters from the parameter servers and the clients in turn transmit updates (based on private data) to improve the latest model estimate. The clients can communicate with one or more parameter servers in each iteration. The parameter servers communicate with each other every few iterations. We assume that parameter servers form a fully connected component whenever they wish to exchange models (parameter vector) with each other. We also assume that each client communicates with more than one parameter servers in every iteration.

We will assume synchronous and fault-free execution of protocol at all clients and parameter servers. This architecture was analyzed by the authors in [30] and is experimentally validated by Hsieh *et al.* in [31], although [31] does not address privacy as we do.

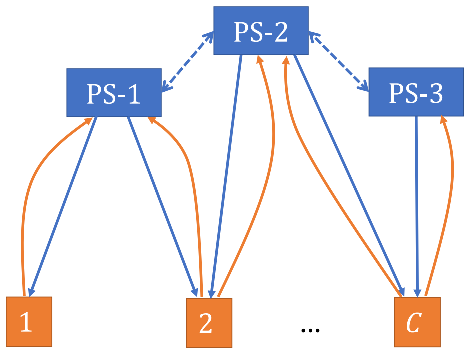

Fig. 1: System Architecture: Parameter Servers (labeled as PS-J) and Clients.

B. Learning Problem

The central problem here is to build a predictive model based on private local information. We translate this into a distributed optimization problem as discussed below.

The machine learning model is parameterized as a D-dimensional vector $x \in \mathbb{R}^D$. The set of all feasible model parameter is denoted as $\mathcal{X}$, which is a convex, compact subset of $\mathbb{R}^D$ (i.e. $x \in \mathcal{X} \subset \mathbb{R}^D$).

We assume that the objective function at client i, denoted as $f_i(x)$ is convex and the gradients denoted as $\nabla f_i(x)$ are Lipschitz. We will later show that this assumption can be relaxed for minimum-wait and client-averaged variants of POLAR-SGD which only need sub-gradient of $f_i(x)$ to be bounded. The learning problem is formally presented below.

Problem 1. *Distributed learning implies finding a minimizer to to the objective function $f(x) \triangleq \sum_{i=1}^{C} f_i(x)$.*

$$\textit{Find } x^* \in \arg\min_{x\in\mathcal{X}} f(x)$$

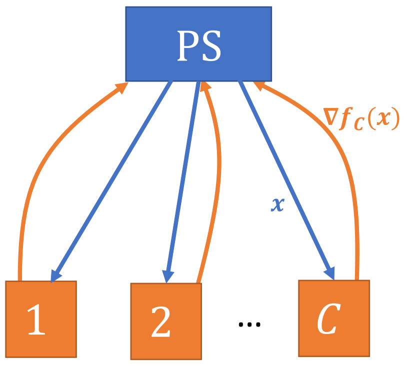

Fig. 2: System with single parameter server [32].

C. Privacy Model

Before we discuss the privacy model, we will review the basic non-privacy preserving algorithm (Figure 2). The algorithm involves iteratively running the following steps -

- Clients download latest model parameters (x) from parameter server and compute gradient of local objective function ($\nabla f_i(x)$).
- Clients upload gradients ($\nabla f_i(x)$) to the server.
- Server performs projected gradient descent using the sum of all received gradients,
$$x \leftarrow \mathcal{P}_{\mathcal{X}}\left[x - \alpha_k \sum_{i=1}^{C} \nabla f_i(x)\right],$$
where, α_k is suitably chosen step size and $\mathcal{P}_{\mathcal{X}}$ is the projection operator.

Note that the basic parameter server algorithm above directly exposes local gradients to the parameter server. An honest but curious adversary can use these gradients to infer membership of certain data points in the client's private local datasets. In fact, observing even a part of gradient can potentially leak membership information about specific data points or a class of data points.

In this work we will assume that multiple parameter servers are available and at most one of the parameter servers is an honest-but-curious adversary (this can be easily generalized to multiple honest but curious adversaries). The parameter servers are often controlled by corporations that provide machine learning as a service that want to learn the best possible machine learning model. However, they may be interested in uncovering private individual data. Our algorithm will protect private data against such honest-but-curious parameter server(s).

D. Notation

Time is indexed as as $\{i, k\}$. The first index i denotes the number of gradient based descent steps performed since last consensus and the second index refers to the number of consensus operations. We consider consensus to happen after every Δ gradient-based updates. Let $\mathcal{N}_h$ denote the servers that can communicate with client h.

III. Algorithm

We reviewed the basic distributed optimization algorithm for parameter server framework [32]. It can be described as iterative projected gradient descent algorithm where the gradients are collected at a central parameter server.

Algorithm 1 POLAR-SGD: Client h

```
1: Input: $x^J_{i,k}$, $\Delta$, NSteps
2: Result: Upload Obfuscated Gradient - $g^{J,h}_{i,k}$
3: for k = 1 to NSteps do
4:     for i = 0 to $\Delta$-1 do
5:         Download: $x^J_{i,k}$
6:             minimum-wait: $u = x^I_{i,k}$ $(I \in \mathcal{N}_h)$
7:             client-averaged: $u = \frac{1}{|\mathcal{N}_h|} \sum_{I \in \mathcal{N}_h} x^I_{i,k}$
8:             basic: $u = x^J_{i,k}$
9:         Compute: $g_h(u, \xi) = \nabla f_h(u, \xi)$
10:        Select Perturbation: $W^{J,h}_{i,k}$, $d^{J,h}_{i,k}$
11:        Compute: $g^{J,h}_{i,k} = \left(W_{i,k}[J,h] g_h(u,\xi) + d^{J,h}_{i,k}\right)$
12:        Upload Obfuscated Stochastic Gradient: $g^{J,h}_{i,k}$
13:    end for
14: end for
```

A. POLAR-SGD

Private Optimization and Learning Algorithm - Stochastic Gradient Descent (POLAR-SGD) is a distributed optimization algorithm for multiple parameter-server architecture (Figure 1).

Algorithm Sketch:

1) In POLAR-SGD, clients download latest model parameters from one or more servers and compute stochastic gradients at these values.
2) Clients then obfuscate the stochastic gradients using multiplicative and additive perturbations (detailed later in the section). These obfuscated stochastic gradients are uploaded to the server(s). A client may communicate with any subset of available servers.
3) Each server uses the received stochastic gradients to perform a projected gradient descent step.
4) Servers periodically perform a consensus iteration over the model parameters.

POLAR-SGD algorithm for Clients and Servers is formally presented as Algorithms 1 and 2 respectively.

POLAR-SGD Client: At each iteration $\{i, k\}$, the client first requests latest model parameters, $x^J_{i,k}$ from parameter servers J (Line 5, Algorithm 1). The model parameters that are received first are used by the client $u = x^I_{i,k}$ where $I \in \mathcal{N}_h$ (Line 6, Algorithm 1). As we select the parameters that arrive first we call this algorithm *minimum-wait* variant of POLAR-SGD. We will discuss a few more variants in next subsection.

Next, each client h then computes stochastic gradient using its private local objective function at the parameter value u (Line 9, Algorithm 1). Clients then obfuscate the stochastic gradient using additive and multiplicative perturbations. We select these perturbations in the Line 10, Algorithm 1 and use these steps to compute the obfuscated gradient in Line 11, Algorithm 1. The obfuscated gradient uploaded by client h to server J at time $\{i, k\}$, denoted as $g^{J,h}_{i,k}$, is defined as,

$$g^{J,h}_{i,k} = \left(W_{i,k}[J,h] g_h(u,\xi) + d^{J,h}_{i,k}\right) \tag{1}$$

where, $W_{i,k}[J, h]$ is the multiplicative perturbation and $d^{J,h}_{i,k}$ is the additive perturbation assigned by client h to server J at time $\{i, k\}$.

The multiplicative ($W_{i,k}[J, h]$) and additive ($d^{J,h}_{i,k}$) perturbations are arbitrary so long as they satisfy Symmetric Learning and Bounded Update Conditions elaborated below.

Symmetric Learning Condition (SLC) ensures that over a period of Δ gradient descent steps, the multiplicative perturbations assigned by clients are equal and that the additive perturbations sum to zero.

Assumption 1 (SLC)**.** *Equal multiplicative perturbations are assigned to updates from every client over a period of Δ steps. Formally, for each client h,*

$$M \triangleq \sum_{J=1}^{S} \left(\sum_{i=0}^{\Delta-1} W_{i,k}[J,h] \right). \quad (M > 0) \tag{2}$$

Also, the additive perturbations add to zero $\forall\{i,k\}$.

$$\sum_{J=1}^{S} d_{i,k}^{J,h} = 0.\ \forall h \tag{3}$$

Bounded Update Condition (BUC), ensures that the multiplicative and additive perturbations are bounded.

Assumption 2 (BUC)**.** *The sum of absolute value of multiplicative weights over* Δ *steps is upper bounded by a finite constant* ($\bar{M} < \infty$)*. Formally, for each client* h

$$\sum_{J=1}^{S} \left(\sum_{i=1}^{\Delta} |W_{i,k}[J,h]| \right) \leq \bar{M}. \quad (\bar{M} > 0) \tag{4}$$

Also, the additive perturbations are bounded.

$$\|d_{i,k}^{J,h}\| \leq Y. \tag{5}$$

In the next step, Line 11 Algorithm 1, each client h uploads the obfuscated stochastic gradient $g_{i,k}^{J,h}$ to server J.

POLAR-SGD Server: Any parameter server J performs two tasks - 1) using received gradients to perform projected stochastic gradient descent and 2) perform secure consensus over model parameters from all servers.

At every iteration $\{i,k\}$, parameter server J uses the obfuscated stochastic gradients $g_{i,k}^{J,h}$ received from clients h at iteration $\{i,k\}$ to perform projected stochastic gradient descent (Line 5, Algorithm 2),

$$x_{i+1,k}^{J} = \mathcal{P}_{\mathcal{X}} \left(x_{i,k}^{J} - \alpha_k \sum_{h=1}^{C} g_{i,k}^{J,h} \right) \tag{6}$$

where, α_k is the step size. We assume that $\{\alpha_k\}$ is monotonically non-increasing, positive sequence with $\sum_k \alpha_k = \infty$ and $\sum_k \alpha_k^2 < \infty$.

The servers perform secure consensus over the model parameters (Lines 8-11, Algorithm 2) after every Δ gradient based updates. The secure multi-party computing based private consensus algorithm is from [33]. As discussed in [33], the private consensus algorithm uses *wrap-around* feature of modulo arithmetic and uniformly distributed (U) noise to provide information theoretic privacy while computing sum/average over a fully connected graph. Observations of A^J from other servers do not leak information about $x_{\Delta,k}^{J}$.

B. POLAR-SGD Variants

We described the minimum-wait version of POLAR-SGD, where a client uses the parameter estimates that were received first (Line 6, Algorithm 1). However, there are a couple of more methods for selecting parameter estimates.

1) Client-averaged: All downloaded parameters are averaged and used for gradient computation. Use of average parameter ensures that we only compute one stochastic gradient per iteration (similar to minimum-wait), however, this requires completion of download of parameters from several servers, it has to wait for the slowest server.

$$u = \frac{1}{\mathcal{N}_h} \sum_{I \in \mathcal{N}_h} x_{i,k}^{I}. \tag{7}$$

Algorithm 2 POLAR-SGD: Parameter Server J

```
1:  Input: $x_{0,1}^J$, $\alpha_k$, $\Delta$, NSteps
2:  Result: $x^* = \arg\min_{x \in \mathcal{X}} \sum_{i=1}^{C} f_i(x)$
3:  for k = 1 to NSteps do
4:      for i = 0 to $\Delta$-1 do
5:          $x_{i+1,k}^J = \mathcal{P}_{\mathcal{X}} \left[ x_{i,k}^J - \alpha_k \sum_{h=1}^{C} g_{i,k}^{J,h} \right]$
6:      end for
7:      Secure Consensus (per coordinate):
8:      Send $R_{J,L} \sim$ U$[0, 2|\mathcal{X}|]$, Receive $R_{L,J} \sim$ U$[0, 2|\mathcal{X}|]$
9:      $A^J = x_{\Delta,k}^J + |\mathcal{X}| + \sum_L R_{L,J} - \sum_L R_{J,L} (mod\ 2|X|)$
10:     Send and Receive $A^J$
11:     Compute: $x_{0,k+1}^J = \frac{1}{S} \left[ \sum_{I=1}^{S} A^I (mod\ 2|X|) \right] - |\mathcal{X}|$
12: end for
```

2) Basic: Clients compute one stochastic gradient per downloaded model parameter. This results in larger computational cost for clients and hence is not preferred.

$$u = x^J_{i,k} \tag{8}$$

Both *minimum-wait* and *client-averaged* variants of POLAR-SGD lead to efficient and desirable implementations since stochastic gradients are computed only once per iteration. As we will see later in Section IV, since only one stochastic gradient is computed per iteration, we will be able to relax the Lipschitzness condition on gradients, thereby allowing the local objective functions to be non-differentiable.

IV. Main Results

We prove following two results for POLAR-SGD:

- Correctness: algorithm solves distributed optimization problem and asymptotically reaches optimum $x^* \in \mathcal{X}^*$
- Privacy: algorithm does not reveal information about private data via uploaded stochastic gradients

We will first show that the iterates of POLAR-SGD converge to the optimum. Theorem 1 states that any variant of POLAR-SGD, such that the perturbations satisfy SLC and BUC (Assumptions 1,2), converges to the optimum asymptotically with probability 1.

Theorem 1 (Correctness)**.** *Let $f_h(x)$ be convex functions ($\forall h$), and gradients $\nabla f_h(x)$ be bounded and Lipschitz. The iterates generated by any POLAR-SGD variants minimum-wait, client-averaged or basic satisfying SLC and BUC (Assumptions 1, 2) converge to the solution of Problem 1 in X^* with probability 1.*

In contrast to Theorem 1, the next result shows that even if the functions are not-differentiable, we can still solve the problem as long as the sub-gradients are bounded and we use either minimum-wait or client-averaged case with symmetric condition satisfied at each $\{i, k\}$. This implies that if $\sum_{J=1}^{S} W_{i,k}[J, h] = M$ and $\sum_{J=1}^{S} |W_{i,k}[J, h]| \leq \bar{M}$ for each client h at any $\{i, k\}$, then the iterates converge to the optimum with probability 1 asymptotically even when $f_h(x)$ are non-differentiable.

Theorem 2. *Let $f_h(x)$ be convex functions ($\forall h$), the sub-gradients $g_h(x)$ be bounded, then the iterates generated by POLAR-SGD variants minimum-wait, or client-averaged satisfying symmetric learning and bounded update condition for each $\{i, k\}$ converge to the solution of Problem 1 in X^* with probability 1.*

An important take-away of these results should be that we do not trade-off accuracy or correctness of the solution with privacy. The privacy and correctness are kind of independent. The trade-off that we observe is between privacy and the speed of convergence.

We will now discuss the privacy result.

Claim 1. *A honest but curious adversarial server J can only observe $g^{J,h}_{i,k}$ from any client h.*

The claim asserts that private gradients have information theoretic security against honest-but-curious adversaries. Information is observed by parameter server via two sources - 1) gradient estimate received from client and 2) information exchanged between different servers. Obfuscation via additive and multiplicative perturbations protects the gradients from curious parameter servers.

The gradients sent by client h to all other servers can be inferred by an honest-but-curious adversary by observing exchange of information during consensus. Private consensus (adapted from [33]) protects the model parameters from a curious adversary using *wrap-around* feature of modulo arithmetic. The parameters A^J exchanged by servers appear to be uniformly distributed similar to the noise added by agents $\sum_L R_{L,J} - \sum_L R_{J,L}$. The component of gradient received by servers other than J is hence protected against curious adversaries.

V. Analysis and Discussion

A. Convergence Analysis

Let $\mathcal{F}_{i,k}$ denote the $\sigma-$ algebra, $\sigma(\xi^h_{[i,k]}; \forall h)$ generated by the stochastic choice of batches until the time step $\{i, k\}$ (note, $\xi^h_{[i,k]} = (\xi^h_{0,0}, \dots, \xi^h_{\Delta,0}, \xi^h_{0,1}, \dots, \xi^h_{\Delta,1}, \dots \xi^h_{i,k})$). $\mathcal{F}_{i,k}$ denotes the history of all stochastic gradients uploaded by clients. Clearly $\mathcal{F}_{i,k} \subset \mathcal{F}_{i,k+1}$ from the construction of $\sigma-$algebra.

Observe that the randomness appears from the gradient computation step (batch selection). The multiplicative and additive weights are arbitrary (possibly random) but they *balance-out* due to symmetric learning condition.

Recall the assumptions made in Section II. We assume that $f_i(x)$, the private objective function at client i is convex, the gradients are bounded $\|\nabla f_i(x)\| \leq L$ and Lipschitz, $\|\nabla f_i(x) - \nabla f_i(y)\| \leq N\|x - y\|$ for all i and $x \neq y \in \mathcal{X}$. Moreover we assume the stochastic gradients to be bounded, $\|g_i(u, \xi)\| \leq L$. Next we develop a couple of key lemma's to prove correctness theorems (Theorem 1,2).

We first state a result on convergence of non-negative almost supermartingales (Theorem 1, [34]).

Lemma 2. *Let $(\Omega, F, \mathcal{P})$ be a probability space and let $F_1 \subset F_2 \subset \dots$ be a sequence of sub $\sigma-$fields of F. Let u_k, v_k and w_k, $k = 0, 1, 2, \dots$ be non-negative F_k- measurable random variables and let $\{\gamma_k\}$ be a deterministic sequence. Assume that $\sum_{k=0}^{\infty} \gamma_k < \infty$ a.s., and $\sum_{k=0}^{\infty} w_k < \infty$ a.s. and*

$$E[u_{k+1}|F_k] \leq (1 + \gamma_k)u_k - v_k + w_k,$$

holds with probability 1. Then, the sequence $\{u_k\}$ converges to a non-negative random variable and $\sum_{k=0}^{\infty} v_k < \infty$ almost surely.

The well known non-expansive property (cf. [35]) of Euclidean projection onto a non-empty, closed, convex set $\mathcal{X}$, is represented by the following inequality, $\forall\, x, y \in \mathbb{R}^D$,

$$\|\mathcal{P}_{\mathcal{X}}[x] - \mathcal{P}_{\mathcal{X}}[y]\| \leq \|x - y\|. \tag{9}$$

This non-expansive property of the projection operator will be used extensively in our analysis.

Recall that we assume the parameter servers to form a fully connected topology. In order to prove this generally we will assume that at each consensus step, we use convex averaging using a doubly stochastic matrix[1] [6], [36]. At every consensus iteration we assume that the local convex averaging protocol can be written as a doubly stochastic matrix B_k. We will prove convergence when the consensus is performed over incomplete yet connected server graph as this will include the scenario when the parameter servers form a fully connected graph.

We borrow transition matrix analaysis result from [37]. We define transition matrix ($\Phi(k, s)$) as the product of doubly stochastic weight matrices, $\Phi(k, s) = B_k B_{k-1} \dots B_{s+1} B_s$, $(\forall\ k \geq s \geq 0)$. We use analysis from [37] to claim linear convergence of all transition matrix entries $\Phi(k, s)[j, i]$ to $1/S$. That is, $\forall\ k \geq s \geq 0$,

$$\left| \Phi(k, s)[i, j] - \frac{1}{S} \right| \leq \theta \beta^{k-s+1},$$

where, $\theta = (1 - \frac{\rho}{4S^2})^{-2}$ and $\beta = (1 - \frac{\rho}{4S^2})$ depend only on graph topology. Note, ρ is the smallest nonzero entry in matrix B_k for any k. [2]

Disagreement Lemma: First we construct a bound on the disagreement between the parameters at any server J and the average parameter vector. This disagreement is denoted by $\delta_k^J = x_{0,k}^J - \bar{x}_{0,k}$.

Lemma 3. *The sequence generated by POLAR-SGD following the Symmetric Learning and Bounded Update conditions,*

$$\max_J \|\delta_{k+1}^J\| \leq S\theta\beta^{k+1} \max_I \|x_{0,0}^I\| + 4\alpha_k \Delta C(\bar{M}L + Y) + 2\theta\Delta SC(\bar{M}L + Y) \sum_{l=1}^{k} \beta^{k+1-l} \alpha_{l-1}$$

We first observe that this is a deterministic bound and is an artifact of the analysis. We use bounds on the stochastic gradients to compute the bound on $\|\delta_{k+1}^J\|$. The first term shows the geometric decrease of initial disagreement among parameter servers. The other two term reflect the effect of stochastic gradient based updates. Detailed proofs can be found in appendix.

Iterate Lemma: The lemma below provides a bound on the distance between the iterates and a point $y \in \mathcal{X}$.

Lemma 4. *The sequence generated by POLAR-SGD following Symmetric Learning and Bounded Update Conditions, satisfies for all $y \in \mathcal{X}$,*

$$\mathbb{E}[\eta_{k+1}^2 | \mathcal{F}_{0,k}] = (1 + A_k)\eta_k^2 - 2M\alpha_k(f(\bar{x}_{0,k}) - f(y)) + B_k$$

$A_k = 4\frac{1}{S}\alpha_k^2 N\Delta C^2 \bar{M}(\bar{M}L + Y) + 2\frac{1}{S}\alpha_k NC\bar{M} \max_J \|\delta_k^J\|$, *and* $B_k = \alpha_k^2 \left(C_1^2 + 4N\Delta C^2 \bar{M}(\bar{M}L + Y)\right) + 2\alpha_k NC\bar{M} \max_J \|\delta_k^J\| + 2\alpha_k C(\bar{M}L + \Delta Y) \max_J \|\delta_k^J\|$.

Note that the expression has similar structure as Lemma 2 and we exploit this to prove convergence. We observe that A_k (respectively γ_k in Lemma 2) is a deterministic sequence and we show that $\sum_k A_k < \infty$. We first prove that $\sum_k \alpha_k \max_J \|\delta_k^J\| < \infty$ a.s. and use it to prove that $\sum_k B_k < \infty$. Invoking Lemma 2 we conclude that $\sum_k \alpha_k(f(\bar{x}_{0,k}) - f(y)) < \infty$ and $\lim_{k \to \infty} \eta_k$ exists. We further use Fatou's lemma to prove convergence of iterate average to the optimum with probability 1. This proves Theorem 1. The proof for Theorem 2 is very similar to the proof for Theorem 1. We exploit the fact that since only one stochastic gradient is computed by a client and the perturbations that are added and multiplied to the stochastic gradient balance-out due to SLC (at each $\{i, k\}$). The sum of these obfuscated gradients is simply the stochastic gradient. And the convergence results can be worked out following the intuition.

[1]In a fully connected topology, the doubly stochastic matrix is just a matrix with all entries being exactly equal to $1/S$.

[2]Due to fully connected graph (of parameter servers), at every consensus iteration, the parameter servers compute average of its model parameters. However, this assumption is not strictly required for correctness. The correctness results are correct even if the parameter server graph is merely connected at every consensus iteration. Hence, we will analyze the harder scenario, and assume that the parameter servers form a connected graph. Obviously if the parameter servers have a fully connected topology the correctness results hold.

B. *Privacy Analysis*

As discussed before there are two possible methods of violating privacy. The first being direct observation of received gradient from a specific client and the second being information about gradient sent to a different server leaking during consensus.

The received gradients are perturbed by multiplicative and additive arbitrary (possibly random) perturbations. The additive perturbation hides gradients especially when the gradient is close to zero (since $W[J,h]g_h(u,\xi)$ will be close to zero too). This protects gradients from direct observation from an adversary. Note that the perturbations are arbitrary and unknown to the adversary, protecting the stochastic gradients. No information leakage happens during consensus as we use secure multi-party computation based private consensus algorithm to average model parameters.

Remark 1. *It is important to note that, while private objective functions owned by individual clients are protected against honest-but-curious adversaries, the overall cost function $f(x)$ is not. Consider a scenario where $\Delta = 1$, where projected descent and consensus steps occur alternatively. As the private consensus step computes the exact average, an adversary can estimate the effect of stochastic gradients on servers other than itself. This along with stochastic gradient directly received from a client, an adversary can estimate the gradient of the whole function, violating the privacy of $f(x)$.*

Remark 1 elaborates the price we pay for exactly solving the given problem. This is common for any correctness preserving algorithm.

Note that we can have the perturbations to be different per coordinate. As long as the symmetric learning and bounded update conditions are satisfied per coordinate, we can easily extend the correctness proofs to per-coordinate weighted obfuscation strategy.

C. *Extensions and Future Work*

We assume that there is a single honest-but-curious adversary. This can be extended to multiple (τ) adversarial parameter servers scenario, however we conjecture that we will need to ensure that clients upload gradients to $\tau + 1$ parameter servers. We leave this as future work.

We make an assumption that the parameter servers form a fully connected graph. This assumption allows us to use secure consensus scheme from [33]. However, if the parameter servers are not fully connected but only connected we can claim correctness following the same proof as described above. We will probably get some privacy under this scenario as well, however, we leave it as future work.

As discussed above, it is easy to extend POLAR-SGD to have coordinate wise different multiplicative weights. Under mild additional conditions like per coordinate satisfaction of symmetric learning and bounded update condition we can extend the correctness results.

VI. Experimental Validation

In this section we empirically validate POLAR-SGD. We consider linear regression problem on a large synthetic dataset. We consider 100000 data points partitioned among $C = 100$ clients. Our system consists of $S = 5$ parameter servers. The clients upload gradients to more than one parameter server in each iteration and the servers form a fully connected graph and perform secure consensus ever Δ iterations.

If dataset D_i is stored at client i, then we can write the local objective function at client i as,

$$f_i(x) = \sum_{l \in D_i} \|x^T a_l - b_l\|^2,$$

where, (a_l, b_l) are data points belonging to D_i. And the overall objective function can be written as,

$$f(x) = \sum_i f_i(x) = \sum_{i=1}^{C} \sum_{l \in D_i} \|x^T a_l - b_l\|^2.$$

We consider multiplicative perturbations and additive perturbations that satisfy SLC and BUC with parameters $\underline{M} = 5$ and $\bar{M} = 50$. We consider three values of $\Delta = \{10, 20, 50\}$ and compare performance. We also compare performance of POLAR-SGD with respect to non-private algorithm and show the trade-off between privacy and convergence speed. Note that we use batch size of 10 samples for computing stochastic gradients in each iteration.

Figure 3 shows the sub-optimality of iterates with respect to iterations. First observe that POLAR-SGD iterates converge (in function value) to the optimum for each Δ. Observe that the non-private algorithm presented earlier converges much faster than POLAR-SGD. As expected, with higher Δ, it takes larger number if iterations for multiplicative and additive perturbations to average out and hence it take longer to converge. The smaller Δ's (like 10 and 20) the iterates follow the general trend of the non-private algorithm very closely. However with larger Δ the function sub-optimality may increase as compared to the starting sub-optimality for few iterations.

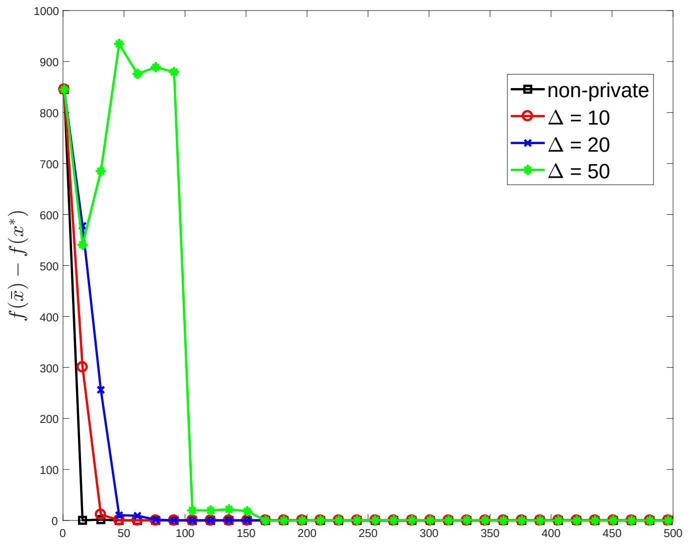

Fig. 3: Sub-optimality v/s iterations.

VII. Conclusion

In this paper we consider privacy preserving distributed optimization algorithm for a multiple parameter server architecture. Our algorithm uses additive and multiplicative weights to perform gradient obfuscation. We show correctness of our algorithm and discuss privacy. We experimentally validate our algorithm on linear regression problem.

References

[1] J. Konečnỳ, B. McMahan, and D. Ramage, "Federated optimization: Distributed optimization beyond the datacenter," *arXiv preprint arXiv:1511.03575*, 2015.

[2] T. Kraska, A. Talwalkar, J. C. Duchi, R. Griffith, M. J. Franklin, and M. I. Jordan, "Mlbase: A distributed machine-learning system.," in *CIDR*, vol. 1, pp. 2–1, 2013.

[3] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, "Communication efficient distributed machine learning with the parameter server," in *NIPS*, pp. 19–27, Curran Associates, Inc., 2014.

[4] I. Cano, M. Weimer, D. Mahajan, C. Curino, and G. M. Fumarola, "Towards geo-distributed machine learning," *arXiv preprint arXiv:1603.09035*, 2016.

[5] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 1310–1321, ACM, 2015.

[6] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *Automatic Control, IEEE Transactions on*, vol. 54, no. 1, pp. 48–61, 2009.

[7] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *Journal of optimization theory and applications*, vol. 147, no. 3, pp. 516–545, 2010.

[8] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: convergence analysis and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, 2012.

[9] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pp. 20–27, ACM, 2004.

[10] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Incremental stochastic subgradient algorithms for convex optimization," *SIAM Journal on Optimization*, vol. 20, no. 2, pp. 691–717, 2009.

[11] D. Jakovetić, J. Xavier, and J. M. Moura, "Fast distributed gradient methods," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1131–1146, 2014.

[12] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the admm in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[13] 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.

[14] A. Nedić and A. Olshevsky, "Distributed Optimization Over Time-Varying Directed Graphs," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 601–615, 2015.

[15] C. Xi and U. A. Khan, "On the linear convergence of distributed optimization over directed graphs," *arXiv:1510.02149*, 2015.

[16] 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, 2016.

[17] A. Nedić, "Asynchronous broadcast-based convex optimization over a network," *IEEE Transactions on Automatic Control*, vol. 56, no. 6, pp. 1337–1351, 2011.

[18] J. Liu and S. J. Wright, "Asynchronous stochastic coordinate descent: Parallelism and convergence properties," *SIAM Journal on Optimization*, vol. 25, no. 1, pp. 351–376, 2015.

[19] P. C. Weeraddana, G. Athanasiou, C. Fischione, and J. S. Baras, "Per-se privacy preserving solution methods based on optimization," in *Proceedings of the 52nd IEEE Conference on Decision and Control (CDC)*, pp. 206–211, 2013.

[20] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *Security and Privacy (SP), 2017 IEEE Symposium on*, pp. 19–38, IEEE, 2017.

[21] J. Hamm, Y. Cao, and M. Belkin, "Learning privately from multiparty data," in *Proceedings of The 33rd International Conference on Machine Learning*, pp. 555–563, 2016.

[22] 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.
[23] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," *arXiv preprint arXiv:1607.00133*, 2016.
[24] O. L. Mangasarian, "Privacy-preserving horizontally partitioned linear programs," *Optimization Letters*, vol. 6, no. 3, pp. 431–436, 2012.
[25] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in *INFOCOM, 2011 Proceedings IEEE*, pp. 820–828, IEEE, 2011.
[26] N. E. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," in *Control Conference (ECC), 2013 European*, pp. 760–765, IEEE, 2013.
[27] S. Gade and N. H. Vaidya, "Private optimization on networks," in *Proceedings of American Control Conference*, IEEE, 2018. (accepted).
[28] S. Gade and N. H. Vaidya, "Private learning on networks: Part ii," *arXiv preprint arXiv:1703.09185*, 2017.
[29] S. Gade and N. H. Vaidya, "Private learning on networks," *arXiv preprint arXiv:1612.05236*, 2016.
[30] S. Gade and N. H. Vaidya, "Distributed optimization for client-server architecture with negative gradient weights," *arXiv preprint arXiv:1608.03866*, 2016.
[31] K. Hsieh, A. Harlap, N. Vijaykumar, D. Konomis, G. R. Ganger, P. B. Gibbons, and O. Mutlu, "Gaia: Geo-distributed machine learning approaching lan speeds.," in *NSDI*, pp. 629–647, 2017.
[32] M. Li, D. G. Andersen, A. Smola, and K. Yu, "Communication Efficient Distributed Machine Learning with the Parameter Server," in *NIPS*, pp. 1–9, 2014.
[33] E. A. Abbe, A. E. Khandani, and A. W. Lo, "Privacy-preserving methods for sharing financial risk exposures," *The American Economic Review*, vol. 102, no. 3, pp. 65–70, 2012.
[34] H. Robbins and D. Siegmund, "A convergence theorem for non negative almost supermartingales and some applications," in *Herbert Robbins Selected Papers*, pp. 111–135, Springer, 1985.
[35] D. P. Bertsekas, A. Nedić, A. E. Ozdaglar, *et al.*, *Convex analysis and optimization*. Athena Scientific, 2003.
[36] L. Xiao and S. Boyd, "Optimal scaling of a gradient method for distributed resource allocation," *Journal of optimization theory and applications*, vol. 129, no. 3, pp. 469–488, 2006.
[37] A. Nedic, A. Olshevsky, A. Ozdaglar, and J. N. Tsitsiklis, "Distributed subgradient methods and quantization effects," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pp. 4177–4184, IEEE, 2008.

VIII. Appendix

We will begin by first restating all the assumptions used in the analysis and proofs. Next, we prove the disagreement lemma (Lemma 3), followed by iterate lemma (Lemma 4). The proofs will be slightly different for the three variants of POLAR-SGD and will be discussed specifically for each proof. We also consider distributed optimization under two scenarios - convex optimization with differentiable and non-differentiable objective functions. The differences in assumptions are discussed below and differences in proofs are elaborated later.

A. Assumptions

We will first review important assumptions required for analysis. Recall that the set of all feasible model parameters, denoted as $\mathcal{X}$, is a convex, compact subset of $\mathbb{R}^D$ (D is the dimension). The local objective functions $f_i(x)$ are convex and the gradients are norm-bounded.

Assumption 3. *Objective functions $f_i : \mathcal{X} \to \mathbb{R}$, are convex functions for all i. Thus, $f(x) \triangleq \sum_{h=1}^{C} f_h(x)$ is also a convex function.*

We consider two analysis scenarios, first where the objective functions are differentiable and we make an assumptions on gradients being Lipschitz (Assumption 4). The second scenario involves using non-differentiable convex functions with bounded sub-gradients (Assumption 5). The convergence result under first scenario (Assumption 4) is reported as Theorem 1.

Assumption 4 (Differntiable Objective Functions)**.** *The differentiable functions $f_i(x)$ satisfy the following conditions,*

1) *The gradients of objective functions are norm-bounded, i.e. $\exists L > 0$, such that, $\|\nabla f_h(x)\| \le L$, $\forall x \in \mathcal{X}$ and $\forall h$. The stochastic gradients are also norm-bounded, i.e. $\|g_h(x,\xi)\| \le L$ for all $x \in \mathcal{X}$ and for all ξ.*
2) *The gradients of objective functions are Lipschitz, i.e. $\exists N > 0$, such that, $\|\nabla f_h(x) - \nabla f_h(y)\| \le N\|x - y\|$, $\forall h$ and $\forall x \ne y \in \mathcal{X}$.*

The convergence result under non-differentiable objective functions or second scenario (Assumption 5) is reported as Theorem 2. We show that if the weights are balanced and bounded (Assumption 1, 2 satisfied) at every iteration $\{i, k\}$ then POLAR-SGD converges to the optimum with probability 1.

Assumption 5 (Non-Differntiable Objective Functions)**.**
The sub-gradients of non-differentiable objective functions are norm-bounded, i.e. $\exists L > 0$, such that, $\|g_h(x)\| \le L$, $\forall x \in \mathcal{X}$ and $\forall h$, where $g_h(x)$ is the sub-gradient of function $f_h(x)$. The stochastic sub-gradients are also norm-bounded, i.e. $\|g_h(x,\xi)\| \le L$ for all $x \in \mathcal{X}$ and for all ξ.

B. Proof of Disagreement Lemma (Lemma 3

We will first prove disagreement lemma (Lemma 3). The proof is the same for all variants of POLAR-SGD and for both differentiable and non-differentiable objective functions. Since we bound the effect of stochastic updates provided by clients, the disagreement lemma is a deterministic statement for all variants of POLAR-SGD and for both problems.

Proof. We first begin by defining an auxiliary variable, z_{k+1}^J,

$$z_{k+1}^J = x_{0,k+1}^J - \sum_{I=1}^{S} B_k[J,I]x_{0,k}^I$$

$$x_{0,k+1}^J = z_{k+1}^J + \sum_{I=1}^{S} B_k[J,I]x_{0,k}^I \tag{10}$$

We unroll the iterations to write $x_{0,k+1}^J$ as a function of $x_{0,k-1}^J$ and z_k^I,

$$x_{0,k+1}^J = z_{k+1}^J + \sum_{I=1}^{S}\left[B_k[J,I]\left(z_k^I + \sum_{l=1}^{S} B_{k-1}[I,l]x_{0,k-1}^l\right)\right] \tag{11}$$

We perform the unrolling equation successively and use the definition of transition matrix $\Phi(k,s)$,

$$x_{0,k+1}^J = z_{k+1}^J + \sum_{I=1}^{S}\Phi(k,0)[J,I]x_{0,0}^I + \sum_{l=1}^{k}\left[\sum_{I=1}^{S}\Phi(k,l)[J,I]z_l^I\right] \tag{12}$$

Next we write the expression for iterate average.

$$\begin{aligned}\bar{x}_{0,k+1} = \frac{1}{S}\sum_{I=1}^{S}x_{0,k+1}^I = \frac{1}{S}\sum_{J=1}^{S}\left[z_{k+1}^J + \sum_{I=1}^{S}B_k[J,I]x_{0,k}^I\right] = \frac{1}{S}\left[\sum_{I=1}^{S}\left(\sum_{J=1}^{S}B_k[J,I]\right)x_{0,k}^I + \sum_{J=1}^{S}z_{k+1}^J\right]\\ = \bar{x}_{0,k} + \frac{1}{S}\sum_{J=1}^{S}z_{k+1}^J = \bar{x}_{0,0} + \frac{1}{S}\sum_{l=1}^{k+1}\sum_{J=1}^{S}z_l^J\end{aligned} \tag{13}$$

We use expression for $x_{0,k+1}^J$ (Eq. 12) and $\bar{x}_{0,k+1}$ (Eq. 13). Next we use linear convergence of transition matrix to $1/S$ using transition matrix analysis from [37].

$$\begin{aligned}\|x_{0,k+1}^J - \bar{x}_{0,k+1}\| &= \|\left(\sum_{I=1}^{S}\Phi(k,0)[J,I]x_{0,0}^I - \bar{x}_{0,0}\right) + \left(\sum_{l=1}^{k}\left[\sum_{I=1}^{S}\Phi(k,l)[J,I]z_l^I\right] - \frac{1}{S}\sum_{l=1}^{k+1}\sum_{I=1}^{S}z_l^I\right) + z_{k+1}^J\|\\ &\le \sum_{I=1}^{S}\left|\Phi(k,0)[J,I] - \frac{1}{S}\right|\|x_{0,0}^I\| + \sum_{l=1}^{k}\sum_{J=1}^{S}\left|\Phi(k,l)[J,I] - \frac{1}{S}\right|\|z_l^I\| + \|z_{k+1}^J\| + \frac{1}{S}\sum_{I=1}^{S}\|z_{k+1}^I\|\\ &\le S\theta\beta^{k+1}\max_I\|x_{0,0}^I\| + \theta\sum_{l=1}^{k}\beta^{k+1-l}\sum_{I=1}^{S}\|z_l^I\| + \|z_{k+1}^J\| + \frac{1}{S}\sum_{I=1}^{S}\|z_{k+1}^I\|\end{aligned} \tag{14}$$

Next we bound the norm of auxiliary variable $\|z_{k+1}^J\|$.

$$\begin{aligned}\|z_{k+1}^J\| &= \|x_{0,k+1}^J - \sum_{I=1}^{S}B_k[J,I]x_{0,k}^I\| = \|\sum_{I=1}^{S}B_k[J,I]x_{\Delta,k}^I - \sum_{I=1}^{S}B_k[J,I]x_{0,k}^I\|\\ &\le \max_I\|x_{\Delta,k}^I - \bar{x}_{0,k}\|\end{aligned}$$

We use projected gradient descent update equation to get a bound. Observe that the following expression would differ for variants of POLAR-SGD. However, as seen in Eq. 18 below, we use the bound on stochastic gradient to compute the bound on auxiliary variable z_k^j.

$$x_{\Delta,k}^I - x_{0,k}^I = -\alpha_k\sum_{t=0}^{\Delta-1}\left[\sum_{h=1}^{C}g_{t,k}^{I,h}\right] + \sum_{t=0}^{\Delta-1}e_t^I \tag{15}$$

where, e_t^I is the projection error. We define it as $e_t^I = x_{t+1,k}^I - \left(x_{t,k}^I - \alpha_k \sum_{h=1}^C g_{t,k}^{I,h}\right)$. Since both $x_{t+1,k}^I$ and $x_{t,k}^I$ both belong in $\mathcal{X}$, the projection error is bounded by the gradient based update.

$$\begin{aligned}
\|e_t^I\| &\le \alpha_k \| \sum_{h=1}^{C} g_{t,k}^{I,h}\| \le \alpha_k \sum_{h=1}^{C} \|g_{t,k}^{I,h}\| && (16)\\
&\le \alpha_k \sum_{h=1}^{C} \|W_{t,k}[J,h] g(u,\xi_{t,k}^h) + d_{t,k}^{I,h}\| \\
&\le \alpha_k C \max\{\|W_{t,k}[J,h] g(u,\xi_{t,k}^h) + d_{t,k}^{I,h}\|\} \\
&\le \alpha_k C \left(\bar{M}L + Y\right) && |W_{t,k}[J,h]| \le \bar{M}, \text{ and } \|d_{t,k}^{I,h}\| \le Y \text{ (BUC) }, \quad (17)\\
& && \|g(u,\xi_{t,k}^h)\| \le L \text{ (Assumption 4 or 5)}
\end{aligned}$$

Using this bound on projection error we get,

$$\begin{aligned}
\|z_{k+1}^J\| \le \max_I \|x_{\Delta,k}^I - \bar{x}_{0,k}\| &\le \| -\alpha_k \sum_{t=0}^{\Delta-1} \left[\sum_{h=1}^{C} g_{t,k}^{I,h}\right] + \sum_{t=0}^{\Delta-1} e_t^I\| \\
&\le \| -\alpha_k \sum_{t=0}^{\Delta-1} \left[\sum_{h=1}^{C} g_{t,k}^{I,h}\right] \| + \| \sum_{t=0}^{\Delta-1} e_t^I\| \\
&\le \| -\alpha_k \sum_{t=0}^{\Delta-1} \left[\sum_{h=1}^{C} g_{t,k}^{I,h}\right] \| + \sum_{t=0}^{\Delta-1} \alpha_k C \left(\bar{M}L + Y\right) && \text{Eq. 17}\\
&\le \alpha_k \sum_{t=0}^{\Delta-1} \| \left[\sum_{h=1}^{C} g_{t,k}^{I,h}\right] \| + \alpha_k \Delta C (\bar{M}L + Y) \\
&\le \alpha_k \sum_{t=0}^{\Delta-1} C(\bar{M}L + Y) + \alpha_k \Delta C (\bar{M}L + Y) && \text{Eq. 16 and 17}\\
&\le 2\alpha_k \Delta C (\bar{M}L + Y) && (18)
\end{aligned}$$

We use the bound on perturbation error to construct a bound $\|z_{k+1}^J\| \le 2\alpha_k \Delta C(\bar{M}L + Y)$ and rewrite Eq. 14.

$$\begin{aligned}
\|x_{0,k+1}^J - \bar{x}_{0,k+1}\| &\le S\theta\beta^{k+1} \max_I \|x_{0,0}^I\| + \theta \sum_{l=1}^{k} \beta^{k+1-l} \sum_{I=1}^{S} \|z_l^I\| + \|z_{k+1}^J\| + \frac{1}{S}\sum_{I=1}^{S} \|z_{k+1}^I\| \\
&\le S\theta\beta^{k+1} \max_I \|x_{0,0}^I\| + \theta \sum_{l=1}^{k} \beta^{k+1-l} \sum_{I=1}^{S} 2\alpha_{l-1}\Delta C(\bar{M}L + Y) + 2\alpha_k \Delta C(\bar{M}L + Y) + \frac{1}{S}\sum_{I=1}^{S} 2\alpha_k \Delta C(\bar{M}L + Y) \\
&\le S\theta\beta^{k+1} \max_I \|x_{0,0}^I\| + 2\theta S \Delta C(\bar{M}L + Y) \sum_{l=1}^{k} \beta^{k+1-l}\alpha_{l-1} + 4\alpha_k \Delta C(\bar{M}L + Y)
\end{aligned}$$

Note that the disagreement lemma expression appears because we assume a connected graph for servers (not fully connected) for proving convergence. If we have a fully connected graph then, we will have exact average at every consensus step and the disagreement would be exactly zero at every iteration. □

C. Proofs for Iterate Lemma - Lemma 4

The proof for Lemma 4 is differs slightly for POLAR-SGD variants. However, we will show that the same almost-supermartingale described in Lemma 4 holds for all versions of POLAR-SGD.

Proof. First we define the distance between an iterates and a point $y \in \mathcal{X}$.

$$\begin{aligned}
\eta_{k+1}^2 &\triangleq \sum_{J=1}^{S} \|x_{0,k+1}^J - y\|^2 \\
&= \sum_{J=1}^{S} \left\| \sum_{I=1}^{S} B_k[J,I] x_{\Delta,k}^I - y \right\|^2 \\
&= \sum_{J=1}^{S} \left\| \sum_{I=1}^{S} B_k[J,I] \left(x_{\Delta,k}^I - y \right) \right\|^2 && \ldots \sum_{I=1}^{S} B_k[J,I] = 1 \\
&\leq \sum_{J=1}^{S} \sum_{I=1}^{S} B_k[J,I] \| \left(x_{\Delta,k}^I - y \right) \|^2 && \ldots B_k[J,I] \geq 0 \\
&\leq \sum_{J=1}^{S} \|x_{\Delta,k}^J - y\|^2 && \ldots \sum_{I=1}^{S} B_k[J,I] = \sum_{J=1}^{S} B_k[J,I] = 1
\end{aligned}$$

Next we use the projected gradient descent update equation to rewrite the above expression in terms of iterates $x_{\Delta-1,k}^J$. This is followed by unrolling the expression further to include prior iterates.

$$\begin{aligned}
\eta_{k+1}^2 &\leq \sum_{J=1}^{S} \|x_{\Delta,k}^J - y\|^2 = \sum_{J=1}^{S} \|\mathcal{P}_{\mathcal{X}} \left[x_{\Delta-1,k}^J - \alpha_k \sum_{h=1}^{C} g_{\Delta-1,k}^{J,h} \right] - y\|^2 \\
&\leq \sum_{J=1}^{S} \|x_{\Delta-1,k}^J - \alpha_k \sum_{h=1}^{C} g_{\Delta-1,k}^{J,h} - y\|^2 && \ldots \text{Non expansive property of projection operator.} \\
&\leq \sum_{J=1}^{S} \left[\|x_{\Delta-1,k}^J - y\|^2 + \alpha_k^2 \left[\| \sum_{h=1}^{C} g_{\Delta-1,k}^{J,h} \| \right] - 2\alpha_k \left[\sum_{h=1}^{C} g_{\Delta-1,k}^{J,h} \right]^T \left[x_{\Delta-1,k}^J - y \right] \right] \\
&\leq \sum_{J=1}^{S} \left[\|x_{0,k}^J - y\|^2 + \alpha_k^2 \sum_{t=0}^{\Delta-1} \left[\| \sum_{h=1}^{C} g_{t,k}^{J,h} \|^2 \right] - 2\alpha_k \sum_{t=0}^{\Delta-1} \left[\left[\sum_{h=1}^{C} g_{t,k}^{J,h} \right]^T \left[x_{t,k}^J - y \right] \right] \right] \\
&\leq \eta_k^2 + \alpha_k^2 \sum_{J=1}^{S} \sum_{t=0}^{\Delta-1} \left[\sum_{h=1}^{C} \|g_{t,k}^{J,h}\|^2 \right] - 2\alpha_k \sum_{J=1}^{S} \sum_{t=0}^{\Delta-1} \left[\left[\sum_{h=1}^{C} g_{t,k}^{J,h} \right]^T \left[x_{t,k}^J - y \right] \right] && (19)
\end{aligned}$$

where, $g_{t,k}^{J,h} = W_{t,k}[J,h] g(u, \xi_{t,k}^h) + d_{t,k}^{J,h}$.
First we rewrite $x_{t,k}^J - y$ as $x_{t,k}^J - y = x_{0,k}^J - y - \alpha_k \sum_{i=0}^{t} \sum_{h=1}^{C} g_{i,k}^{J,h} + \sum_{i=0}^{t} e_i^J$ following the analysis from Eq. 15. We plug this expression into the above expession (Eq. 19).

$$\begin{aligned}
\eta_{k+1}^2 &\leq \eta_k^2 + \alpha_k^2 \sum_{J=1}^{S} \sum_{t=0}^{\Delta-1} \left[\sum_{h=1}^{C} \|g_{t,k}^{J,h}\|^2 \right] - 2\alpha_k \sum_{J=1}^{S} \sum_{t=0}^{\Delta-1} \left[\left[\sum_{h=1}^{C} g_{t,k}^{J,h} \right]^T \left[x_{0,k}^J - y - \alpha_k \sum_{i=0}^{t} \sum_{h=1}^{C} g_{i,k}^{J,h} + \sum_{i=0}^{t} e_i^J \right] \right] \\
&\leq \eta_k^2 + \alpha_k^2 \sum_{J=1}^{S} \sum_{t=0}^{\Delta-1} \left[\sum_{h=1}^{C} \|g_{t,k}^{J,h}\|^2 \right] - 2\alpha_k \sum_{J=1}^{S} \sum_{t=0}^{\Delta-1} \left[\left[\sum_{h=1}^{C} g_{t,k}^{J,h} \right]^T \left[x_{0,k}^J - y \right] \right] \\
&\quad \underbrace{+2\alpha_k \sum_{J=1}^{S} \sum_{t=0}^{\Delta-1} \left\| \sum_{h=1}^{C} g_{t,k}^{J,h} \right\| \left\| -\alpha_k \sum_{i=0}^{t} \sum_{h=1}^{C} g_{i,k}^{J,h} + \sum_{i=0}^{t} e_i^J \right\|}_{\Pi} && (20)
\end{aligned}$$

We will first simplify Π.

$$\begin{aligned}
\Pi &= 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left\|\sum_{h=1}^{C} g_{t,k}^{J,h}\right\| \left\| -\alpha_k \sum_{i=0}^{t}\sum_{h=1}^{C} g_{i,k}^{J,h} + \sum_{i=0}^{t} e_i^J \right\| \\
&\le 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left\|\sum_{h=1}^{C} g_{t,k}^{J,h}\right\| \left(2\alpha_k \Delta C(\bar{M}L+Y)\right) \\
&\le 4\alpha_k^2 \Delta C(\bar{M}L+Y) \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left\|\sum_{h=1}^{C} g_{t,k}^{J,h}\right\| \le 4\alpha_k^2 \Delta C(\bar{M}L+Y) \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\sum_{h=1}^{C}\left\| g_{t,k}^{J,h}\right\| \\
&\le 4\alpha_k^2 \Delta C(\bar{M}L+Y) \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\sum_{h=1}^{C}\left\| W_{t,k}[J,h] g_h(u,\xi_{t,k}^h) + d_{t,k}^{J,h}\right\| \\
&\le 4\alpha_k^2 \Delta C(\bar{M}L+Y) \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\sum_{h=1}^{C}(\bar{M}L+Y) \qquad \text{BUC, Assumption 4 or 5} \\
&\le 4\alpha_k^2 S\Delta^2 C^2 (\bar{M}L+Y)^2
\end{aligned} \tag{21}$$

We plug the bound on Π found in Eq. 21 in Eq. 20,

$$\eta_{k+1}^2 \le \eta_k^2 + \alpha_k^2 \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left[\sum_{h=1}^{C}\|g_{t,k}^{J,h}\|^2\right] - 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left[\left[\sum_{h=1}^{C} g_{t,k}^{J,h}\right]^T \left[x_{0,k}^J - y\right]\right] + 4\alpha_k^2 S\Delta^2 C^2(\bar{M}L+Y)^2$$

Here, we use boundedness assumption on the stochastic gradient (sub-gradient) and both SLC and BUC (Assumptions 1, 2, 4 and 5). Hence the expression is valid for both scenarios with differentiable objective functions and non-differentiable objective functions and for all variants of POLAR-SGD.
We take expectation of both sides of the expression conditioned on $\mathcal{F}_{0,k}$.

$$\begin{aligned}
\mathbb{E}[\eta_{k+1}^2|\mathcal{F}_{0,k}] \le \eta_k^2 &+ \alpha_k^2 \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left[\sum_{h=1}^{C}\mathbb{E}[\|g_{t,k}^{J,h}\|^2|\mathcal{F}_{0,k}]\right] - 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\mathbb{E}\left[\left[\sum_{h=1}^{C} g_{t,k}^{J,h}\right]^T \left[x_{0,k}^J - y\right]|\mathcal{F}_{0,k}\right]\right) \\
&+ 4\alpha_k^2 S\Delta^2 C^2(\bar{M}L+Y)^2
\end{aligned}$$

Note that since ξ is drawn from uniform distribution,

$$\begin{aligned}
\mathbb{E}[g_{t,k}^{J,h}|\mathcal{F}_{0,k}] &= \mathbb{E}[W_{t,k}[J,h]g(u,\xi_{t,k}^h) + d_{t,k}^{J,h}|\mathcal{F}_{0,k}] \\
&= W_{t,k}[J,h]\nabla f_h(u) + d_{t,k}^{J,h}
\end{aligned}$$

where, $u = u_{t,k}^h$ is the downloaded model by client h. A client may use one of three specified methods: basic, client-averaged or minimum-weight. Hence we get,

$$\mathbb{E}[\eta_{k+1}^2|\mathcal{F}_{0,k}] \le \eta_k^2 + \alpha_k^2 C_0^2 - 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\mathbb{E}\left[\left[\sum_{h=1}^{C} g_{t,k}^{J,h}\right]^T \left[x_{0,k}^J - y\right]|\mathcal{F}_{0,k}\right]\right) + 4\alpha_k^2 S\Delta^2 C^2(\bar{M}L+Y)^2$$

where, $C_0^2 = \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left[\sum_{h=1}^{C}\mathbb{E}[\|g_{t,k}^{J,h}\|^2|\mathcal{F}_{0,k}]\right] < \infty$ follows from boundedness of stochastic gradients (sub-gradients).

$$\mathbb{E}[\eta_{k+1}^2|\mathcal{F}_{0,k}] \le \eta_k^2 + \alpha_k^2 C_1^2 - 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\mathbb{E}\left[\sum_{h=1}^{C} g_{t,k}^{J,h}|\mathcal{F}_{0,k}\right]^T \left[x_{0,k}^J - y\right]\right)$$

where, $C_1^2 = C_0^2 + 4S\Delta^2C^2(\bar{M}L+Y)^2$.

$$\mathbb{E}[\eta_{k+1}^2|\mathcal{F}_{0,k}] \le \eta_k^2 + \alpha_k^2 C_1^2 \underbrace{-2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]\nabla f_h(u) + d_{t,k}^{J,h}\right]^T \left[x_{0,k}^J - y\right]\right)}_{\Lambda} \tag{22}$$

Next we estimate a bound on Λ (Eq. 22).

$$\begin{aligned}\Lambda &= -2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]\nabla f_h(u) + d_{t,k}^{J,h}\right]^T [x_{0,k}^J - y]\right)\\
&= -2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]\nabla f_h(u) + d_{t,k}^{J,h}\right]^T [\bar{x}_{0,k} + q_{0,k}^J - y]\right)\\
&= -2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]\nabla f_h(u) + d_{t,k}^{J,h}\right]^T [\bar{x}_{0,k} - y]\right)\\
&\quad - 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]\nabla f_h(u) + d_{t,k}^{J,h}\right]^T [q_{0,k}^J]\right)\end{aligned}$$

where, $q_{0,k}^J = x_{0,k}^J - \bar{x}_{0,k}$. Here the proof differs based on the specific variant of POLAR-SGD. First let us consider the vanilla version where $u = x_{t,k}^J$.

Observe that since u is different depending upon the POLAR-SGD variant used. And hence the proof differs for different variants of the algorithm. However we will first discuss the differences and then provide a unified proof for the claims. We will first use Gradient Lipschitzness (Assumption 4) to establish the following gradient approximations and use them in the proof. Since the same quantity can be used to upper bound the gradient approximation error, $l_{t,k}^{J,h} = \nabla f_h(u) - \nabla f_h(\bar{x}_{0,k})$, the proof follows a similar structure and the same super-martingale expression is valid for all three variants of POLAR-SGD.

Case 1: Basic variant - For the basic variant where $u = x_{t,k}^J$. We assume that the gradients are Lipschitz, and hence we can write, $\nabla f_h(x_{t,k}^J) = \nabla f_h(\bar{x}_{0,k}) + l_{t,k}^{J,h}$, where,

$$\begin{aligned}\|l_{0,k}^h\| &= \max_{J,t}\|l_{t,k}^{J,h}\| = \max_{J,t}\|\nabla f_h(x_{t,k}^J) - \nabla f_h(\bar{x}_{0,k})\| \\
&\leq N\max_{J,t}\|x_{t,k}^J - \bar{x}_{0,k}\| && \nabla f_h(x) \text{ is Lipschitz}\\
&\leq N\max_{J,t}\|x_{t,k}^J - x_{0,k}^J\| + N\max_{J}\|x_{0,k}^J - \bar{x}_{0,k}\| && \text{Triangle Inequality}\\
&\leq 2\alpha_k N\Delta C(\bar{M}L + Y) + N\max_J\|q_{0,k}^J\| && \text{Eq. 18}\\
\|l_{0,k}^h\| &= \max_{J,h}\|l_{0,k}^h\| \leq 2\alpha_k N\Delta C(\bar{M}L + Y) + N\max_J\|q_{0,k}^J\| && (23)\end{aligned}$$

Case 2: Minimum-Wait variant - For the minimum-wait variant where $u = x_{t,k}^I$ for some $I \in \mathcal{N}_h$. We assume that the gradients are Lipschitz, and hence we can write, $\nabla f_h(x_{t,k}^I) = \nabla f_h(\bar{x}_{0,k}) + l_{t,k}^{I,h}$, where,

$$\begin{aligned}\|l_{0,k}^h\| &= \max_{I,t}\|l_{t,k}^{I,h}\| = \max_{I,t}\|\nabla f_h(x_{t,k}^I) - \nabla f_h(\bar{x}_{0,k})\| \\
&\leq N\max_{I,t}\|x_{t,k}^I - \bar{x}_{0,k}\| && \nabla f_h(x) \text{ is Lipschitz}\\
&\leq N\max_{I,t}\|x_{t,k}^I - x_{0,k}^I\| + N\max_{I}\|x_{0,k}^I - \bar{x}_{0,k}\| && \text{Triangle Inequality}\\
&\leq 2\alpha_k N\Delta C(\bar{M}L + Y) + N\max_{I,t}\|q_{0,k}^I\| \\
\|l_{0,k}^h\| &= \max_{I,h}\|l_{t,k}^{I,h}\| \leq 2\alpha_k N\Delta C(\bar{M}L + Y) + N\max_I\|q_{0,k}^I\| && (24)\end{aligned}$$

Case 3: Client-Averaged variant - For the client-averaged variant where $u = \frac{1}{|\mathcal{N}_h|}\sum_{I\in\mathcal{N}_h} x_{t,k}^I$. We assume that the gradients are Lipschitz, and hence we can write, $\nabla f_h(u) = \nabla f_h(\bar{x}_{0,k}) + l_{t,k}^h$, where,

$$
\begin{aligned}
\|l_{0,k}^{h}\| &= \|\nabla f_h(u) - \nabla f_h(\bar{x}_{0,k})\| \\
&\leq N\|u - \bar{x}_{0,k}\| && \nabla f_h(x) \text{ is Lipschitz} \\
&\leq N\|\frac{1}{|\mathcal{N}_h|}\sum_{I\in\mathcal{N}_h}\left(x_{t,k}^{I} - x_{0,k}^{I}\right)\| + N\|\frac{1}{|\mathcal{N}_h|}\sum_{I\in\mathcal{N}_h}\left(x_{0,k}^{I} - \bar{x}_{0,k}\right)\| \\
&\leq N\max_{I}\|x_{t,k}^{I} - x_{0,k}^{I}\| + N\max_{I}\|x_{0,k}^{I} - \bar{x}_{0,k}\| \\
&\leq 2\alpha_k N\Delta C(\bar{M}L + Y) + N\max_{I}\|q_{0,k}^{I}\|
\end{aligned}
$$

$$
\|l_{0,k}^{h}\| = \max_{J,h}\|l_{t,k}^{J,h}\| \leq 2\alpha_k N\Delta C(\bar{M}L + Y) + N\max_{I}\|q_{0,k}^{I}\| \tag{25}
$$

Observe that the bound on $\|l_{t,k}^{J,h}|$ is the same for all the algorithmic variants.
Next we will use the bound on $\|l_{0,k}^{h}\|$ developed in Eq. 23, 24 and 25 to rewrite Λ and construct a bound,

$$
\begin{aligned}
\Lambda &= 2\alpha_k\sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C}W_{t,k}[J,h]\left(\nabla f_h(\bar{x}_{0,k}) + l_{0,k}^{h}\right) + d_{t,k}^{J,h}\right]^{T}[y - \bar{x}_{0,k}]\right) \\
&\quad - 2\alpha_k\sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C}W_{t,k}[J,h](\nabla f_h(u_{t,k}^{h})) + d_{t,k}^{J,h}\right]^{T}[q_{0,k}^{J}]\right) \\
&= \underbrace{2\alpha_k\sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C}W_{t,k}[J,h]\nabla f_h(\bar{x}_{0,k}) + d_{t,k}^{J,h}\right]^{T}[y - \bar{x}_{0,k}]\right)}_{T_1} \\
&\quad + \underbrace{2\alpha_k\sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C}W_{t,k}[J,h]\ l_{0,k}^{h}\right]^{T}[y - \bar{x}_{0,k}]\right)}_{T_2} \\
&\quad \underbrace{-2\alpha_k\sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C}W_{t,k}[J,h](\nabla f_h(u_{t,k}^{h})) + d_{t,k}^{J,h}\right]^{T}[q_{0,k}^{J}]\right)}_{T_3} \\
&= T_1 + T_2 + T_3
\end{aligned}
$$

We will next construct each of the individual bounds on T_1, T_2 and T_3.

$$
\begin{aligned}
T_1 &= 2\alpha_k\sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C}W_{t,k}[J,h]\nabla f_h(\bar{x}_{0,k}) + d_{t,k}^{J,h}\right]^{T}[y - \bar{x}_{0,k}]\right) \\
&= 2\alpha_k\left(\sum_{h=1}^{C}\left[\sum_{J=1}^{S}\left(\sum_{t=0}^{\Delta-1}W_{t,k}[J,h]\right)\nabla f_h(\bar{x}_{0,k}) + \sum_{t=0}^{\Delta-1}\left(\sum_{J=1}^{S}d_{t,k}^{J,h}\right)\right]^{T}\right)[y - \bar{x}_{0,k}] \\
&= 2\alpha_k\left(\sum_{h=1}^{C}\left[\sum_{J=1}^{S}\left(\sum_{t=0}^{\Delta-1}W_{t,k}[J,h]\right)\nabla f_h(\bar{x}_{0,k})\right]^{T}\right)[y - \bar{x}_{0,k}] && \sum_{J=1}^{S}d_{t,k}^{J,h} = 0 \\
&= 2\alpha_k\left(\sum_{h=1}^{C}[M\nabla f_h(\bar{x}_{0,k})]^{T}\right)[y - \bar{x}_{0,k}] && \sum_{J=1}^{S}\left(\sum_{t=0}^{\Delta-1}W_{t,k}[J,h]\right) = M \\
&\leq -2\alpha_k M\left(f(\bar{x}_{0,k}) - f(y)\right) && \sum_{h=1}^{C}\nabla f_h(\bar{x}_{0,k}) = \nabla f(\bar{x}_{0,k})
\end{aligned} \tag{26}
$$

$$
\begin{aligned}
\|T_2\| &= \left\| 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]\ l_{0,k}^h\right]^T [y-\bar{x}_{0,k}]\right)\right\| \\
&\le \left\| 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]\ l_{0,k}^h\right]\right)\right\| \|y-\bar{x}_{0,k}\| \\
&\le 2\alpha_k \left(\left[\sum_{h=1}^{C}\left(\sum_{J=1}^{S}\left(\sum_{t=0}^{\Delta-1}|W_{t,k}[J,h]|\right)\right)\ \max_{J,h}\|l_{0,k}^h\|\right]\right)\|y-\bar{x}_{0,k}\| \\
&\le 2\alpha_k \left(\left[\sum_{h=1}^{C}\bar{M}\ \max_{J,h}\|l_{0,k}^h\|\right]\right)\|y-\bar{x}_{0,k}\| && \sum_{J=1}^{S}\left(\sum_{t=0}^{\Delta-1}|W_{t,k}[J,h]|\right)\le\bar{M} \\
&\le 2\alpha_k\left(C\bar{M}\left[2\alpha_k N\Delta C(\bar{M}L+Y)+N\max_J\|q_{0,k}^J\|\right]\right)\|y-\bar{x}_{0,k}\| && \text{Eq. 23, 23, or 25} \\
&\le 4\alpha_k^2 N\Delta C^2\bar{M}(\bar{M}L+Y)\|\bar{x}_{0,k}-y\|+2\alpha_k NC\bar{M}\max_J\|q_{0,k}^J\|\|\bar{x}_{0,k}-y\| && (27)
\end{aligned}
$$

$$
\begin{aligned}
\|T_3\| &= \left\| -2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h](\nabla f_h(u_{t,k}^h))+d_{t,k}^{J,h}\right]^T \left[q_{0,k}^J\right]\right)\right\| \\
&\le 2\alpha_k\left\|\sum_{h=1}^{C}\left[\left(\sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}|W_{t,k}[J,h]|\right)\max_{h,t}\|\nabla f_h(u_{t,k}^h)\|+\sum_{t=0}^{\Delta-1}\sum_{J=1}^{S}|d_{t,k}^{J,h}|\right]^T\left[q_{0,k}^J\right]\right\| \\
&\le 2\alpha_k\sum_{h=1}^{C}\left\|\left[\bar{M}\max_{h,t}\|\nabla f_h(u_{t,k}^h)\|+\sum_{t=0}^{\Delta-1}\sum_{J=1}^{S}|d_{t,k}^{J,h}|\right]\right\|\left\|\left[q_{0,k}^J\right]\right\| && \sum_{J=1}^{S}\left(\sum_{t=0}^{\Delta-1}|W_{t,k}[J,h]|\right)\le\bar{M} \\
&\le 2\alpha_k C(\bar{M}L+\Delta Y)\max_J\|q_{0,k}^J\| && \sum_{J=1}^{S}|d_{t,k}^{J,h}|\le Y \qquad (28)
\end{aligned}
$$

First we observe that for any positive a, $a^2+1\ge 2a$. We use this to modify the bound on T_2. And we use the fact that $\|\bar{x}_{0,k}-y\|^2=\|\frac{1}{S}\sum_{J=1}^{S}(x_{0,k}^J-y)\|^2\le\frac{1}{S}\sum_{J=1}^{S}\|x_{0,k}^J-y\|^2=\frac{1}{S}\eta_k^2$.

$$
\begin{aligned}
\|T_2\| &\le 4\alpha_k^2 N\Delta C^2\bar{M}(\bar{M}L+Y)\|\bar{x}_{0,k}-y\|+2\alpha_k NC\bar{M}\max_J\|q_{0,k}^J\|\|\bar{x}_{0,k}-y\| \\
&\le 4\alpha_k^2 N\Delta C^2\bar{M}(\bar{M}L+Y)(1+\|\bar{x}_{0,k}-y\|^2)+2\alpha_k NC\bar{M}\max_J\|q_{0,k}^J\|(1+\|\bar{x}_{0,k}-y\|^2) \\
&\le 4\alpha_k^2 N\Delta C^2\bar{M}(\bar{M}L+Y)(1+\frac{1}{S}\eta_k^2)+2\alpha_k NC\bar{M}\max_J\|q_{0,k}^J\|(1+\frac{1}{S}\eta_k^2)
\end{aligned}
$$

We combine all the bounds to get a bound on Λ.

$$
\begin{aligned}
\Lambda \le &-2\alpha_k M\left(f(\bar{x}_{0,k})-f(y)\right) \\
&+4\alpha_k^2 N\Delta C^2\bar{M}(\bar{M}L+Y)(1+\frac{1}{S}\eta_k^2)+2\alpha_k NC\bar{M}\max_J\|q_{0,k}^J\|(1+\frac{1}{S}\eta_k^2) \\
&+2\alpha_k C(\bar{M}L+\Delta Y)\max_J\|q_{0,k}^J\| \qquad (29)
\end{aligned}
$$

We use this bound in our iterate relation:

$$
\begin{aligned}
\mathbb{E}[\eta_{k+1}^2|\mathcal{F}_{0,k}] &\le \eta_k^2+\alpha_k^2C_1^2+\Lambda \\
&\le \eta_k^2+\alpha_k^2C_1^2-2\alpha_k M\left(f(\bar{x}_{0,k})-f(y)\right)+4\alpha_k^2 N\Delta C^2\bar{M}(\bar{M}L+Y)(1+\frac{1}{S}\eta_k^2) \\
&\qquad +2\alpha_k NC\bar{M}\max_J\|q_{0,k}^J\|(1+\frac{1}{S}\eta_k^2)+2\alpha_k C(\bar{M}L+\Delta Y)\max_J\|q_{0,k}^J\| \\
&\le (1+A_k)\,\eta_k^2-2\alpha_k M\left(f(\bar{x}_{0,k})-f(y)\right)+B_k \qquad (30)
\end{aligned}
$$

where, $A_k=4\frac{1}{S}\alpha_k^2N\Delta C^2\bar{M}(\bar{M}L+Y)+2\frac{1}{S}\alpha_k NC\bar{M}\max_J\|\delta_k^J\|$, and $B_k=\alpha_k^2C_1^2+4\alpha_k^2N\Delta C^2\bar{M}(\bar{M}L+Y)+2\alpha_k NC\bar{M}\max_J\|\delta_k^J\|+2\alpha_k C(\bar{M}L+\Delta Y)\max_J\|\delta_k^J\|$.

This proves the iterate lemma. □

D. *Proof for Theorem 1*

Next we use the iterate lemma to prove the correctness theorem (Theorem 1).

Proof. We use almost supermartingale convergence result (Lemma 2) from [34]. First observe that the expression in iterate lemma (Lemma 4) has the desired structure as seen in Lemma 2.

$$\mathbb{E}[\eta_{k+1}^2|\mathcal{F}_{0,k}] = (1 + A_k)\eta_k^2 - 2\alpha_k M(f(\bar{x}_{0,k}) - f(y)) + B_k. \tag{31}$$

Next we observe that we need the $\sum_k A_k < \infty$ and $\sum_k B_k < \infty$ to be true for using Robbins-Siegmund lemma (Lemma 2). Also recall that A_k and B_k are non-negative, real sequences.

We will first show that $\sum_k \alpha_k \max_J \|\delta_k^J\| < \infty$. We use the disagreement lemma (Lemma 3) to prove this.

$$\begin{aligned}\sum_k \alpha_k \max_J \|\delta_k^J\| &= \alpha_1\|\delta_1^J\| + \sum_{k=1}^{\infty} \alpha_{k+1} \max_J \|\delta_{k+1}^J\| \\ &\leq \alpha_1\|\delta_1^J\| + \sum_{k=1}^{\infty}\left(S\alpha_{k+1}\theta\beta^{k+1}\max_I \|x_{0,0}^I\| + 4\alpha_{k+1}\alpha_k\Delta C(\bar{M}L + Y) + 2\alpha_{k+1}\theta\Delta SC(\bar{M}L+Y)\sum_{l=1}^{k}\beta^{k+1-l}\alpha_{l-1}\right) \\ &\leq \underbrace{\alpha_1\|\delta_1^J\|}_{U_0} + \underbrace{S\theta\max_I\|x_{0,0}^I\|\sum_{k=1}^{\infty}\alpha_{k+1}\beta^{k+1}}_{U_1} + \underbrace{4\Delta C(\bar{M}L+Y)\sum_{k=1}^{\infty}\alpha_k^2}_{U_2} \qquad\qquad (\alpha_k \geq \alpha_{k+1}) \\ &\quad \underbrace{+\, 2\theta\Delta SC(\bar{M}L+Y)\sum_{k=1}^{\infty}\alpha_{k+1}\sum_{l=1}^{k}\beta^{k+1-l}\alpha_{l-1}}_{U_3}\end{aligned} \tag{32}$$

First observe that $U_0 < \infty$ due to compact feasible set and $U_2 < \infty$ since $\sum_k \alpha_k^2 < \infty$ from assumptions on step size. Moreover, it is easy to see that $U_1 < \infty$. This follows from the ratio test for convergence of series.

$$\lim_{k\to\infty}\sup\frac{\alpha_{k+2}\beta^{k+2}}{\alpha_k\beta^{k+1}} = \lim_{k\to\infty}\sup\frac{\alpha_{k+2}\beta}{\alpha_k} < 1 \implies \sum_{k=1}^{\infty}\alpha_{k+1}\beta^k < \infty.$$

We observe the fact that $\sum_{k=1}^{\infty}\alpha_{k+1}\sum_{l=1}^{k}\beta^{k+1-l}\alpha_{l-1} \leq \sum_{k=1}^{\infty}\sum_{l=1}^{k}\beta^{k+1-l}\alpha_{l-1}^2$ from the fact that $\alpha_{k+1} \leq \alpha_{l-1}$. And use Lemma 3 from [7], to conclude $U_3 < \infty$. Thus $\sum_k \alpha_k \max_J \|\delta_k^J\| < \infty$.

We use the fact that $\sum_k \alpha_k \max_J \|\delta_k^J\| < \infty$ and $\sum_k \alpha_k^2 < \infty$ to conclude $\sum_k A_k < \infty$ and $\sum_k B_k < \infty$.

$$\begin{aligned}\sum_k A_k &= \sum_k\left(4\frac{1}{S}\alpha_k^2 N\Delta C^2\bar{M}(\bar{M}L+Y) + 2\frac{1}{S}\alpha_k NC\bar{M}\max_J\|\delta_k^J\|\right) \\ &= 4\frac{1}{S}N\Delta C^2\bar{M}(\bar{M}L+Y)\sum_k\alpha_k^2 + 2\frac{1}{S}NC\bar{M}\sum_k\alpha_k\max_J\|\delta_k^J\| < \infty \\ \sum_k B_k &= \sum_k\left(\alpha_k^2\left(C_1^2 + 4N\Delta C^2\bar{M}(\bar{M}L+Y)\right) + 2\alpha_k C(\bar{M}L+\Delta Y)\max_J\|\delta_k^J\| + 2\alpha_k NC\bar{M}\max_J\|q_{0,k}^J\|\right) \\ &= \left(C_1^2 + 4N\Delta C^2\bar{M}(\bar{M}L+Y)\right)\sum_k\alpha_k^2 + (2NC\bar{M} + 2C(\bar{M}L+\Delta Y))\sum_k\alpha_k\max_J\|\delta_{0,k}^J\|\end{aligned}$$

We use an argument similar to observed in [7]. Using Lemma 2 we conclude that the sequence η_k^2 converges to a non-negative random variable and $\sum_k \alpha_k(f(\bar{x}_{0,k}) - f^*) < \infty$ with probability 1. From the disagreement lemma we know that the iterates converge towards each other. Using the fact that $\sum_k \alpha_k(f(\bar{x}_{0,k}) - f^*) < \infty$, $\sum_k \alpha_k = \infty$, and the continuity of objective function $f(x)$ we can conclude that the iterate average $\bar{x}_{0,k}$ must converge to $\mathcal{X}^*$ with probability 1 and hence the iterates must also converge to $\mathcal{X}^*$ with probability 1. □

E. *Non-Differentiable Objective Functions (Theorem 2)*

The proof for Theorem 2 is similar to proof for Theorem 1. There are, however, a few critical differences in the proof for Lemma 4 for non-differentiable objective functions. We will discuss the differences first.

The proof for Lemma 4 for non-differentiable functions is is as described in Appendix C, with a few differences after Eq. 22. We need to construct a bound on Λ using additional assumptions on SLC and BUC being satisfied at every iteration $\{i, k\}$.

Note that we use $g_h(u)$ to denote the stochastic sub-gradient and the fact that we use either minimum-wait or client-averaged variants of POLAR-SGD (not basic). From Eq. 22, we have,

$$\begin{aligned}
\Lambda &= -2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]g_h(u) + d_{t,k}^{J,h}\right]^T [x_{0,k}^J - y]\right) \\
&= -2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]g_h(u) + d_{t,k}^{J,h}\right]^T [\bar{x}_{0,k} - y + x_{0,k}^J - \bar{x}_{0,k}]\right) \\
&\leq -2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]g_h(u) + d_{t,k}^{J,h}\right]^T [\bar{x}_{0,k} - y]\right) \\
&\qquad + 2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left\| \left[\sum_{h=1}^{C} W_{t,k}[J,h]g_h(u) + d_{t,k}^{J,h}\right]^T \right\| \max_J \|\delta_k^J\|\right) \\
&\leq -2\alpha_k \sum_{J=1}^{S}\sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} W_{t,k}[J,h]g_h(u) + d_{t,k}^{J,h}\right]^T [\bar{x}_{0,k} - y]\right) + 2\alpha_k \Delta SC(\bar{M}L + Y)\max_J \|\delta_k^J\| \\
&\qquad \ldots \text{Bounded stochastic sub-gradients and BUC} \\
&\leq -2\alpha_k \sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C}\sum_{J=1}^{S}\left(W_{t,k}[J,h]g_h(u) + d_{t,k}^{J,h}\right)\right]^T [\bar{x}_{0,k} - y]\right) + 2\alpha_k \Delta SC(\bar{M}L + Y)\max_J \|\delta_k^J\| \\
&\qquad \ldots \text{Note, we require SLC and BUC to be satisfied per iteration } \{i,k\} \\
&\leq -2\alpha_k \sum_{t=0}^{\Delta-1}\left(\left[\sum_{h=1}^{C} (Mg_h(u))\right]^T [\bar{x}_{0,k} - y]\right) + 2\alpha_k \Delta SC(\bar{M}L + Y)\max_J \|\delta_k^J\| \\
&\leq -2\alpha_k \Delta\left(\left[\sum_{h=1}^{C} (Mg_h(u))\right]^T [\bar{x}_{0,k} - y]\right) + 2\alpha_k \Delta SC(\bar{M}L + Y)\max_J \|\delta_k^J\| \\
&\leq -2\alpha_k \Delta M\left(\sum_{h=1}^{C} f_h(\bar{x}_{0,k}) - f_h(y)\right) + 2\alpha_k \Delta SC(\bar{M}L + Y)\max_J \|\delta_k^J\| \\
&\qquad \ldots \text{Convexity of function } f_h(x) \text{ and sub-gradient property for convex functions} \\
&\leq -2\alpha_k \Delta M\left(f(\bar{x}_{0,k}) - f(y)\right) + 2\alpha_k \Delta SC(\bar{M}L + Y)\max_J \|\delta_k^J\| \qquad (33)
\end{aligned}$$

Observe that the bound obtained for non-differentiable objective functions in Eq. 33 has similar structure to the bound for scenario when the objective functions are differentiable (see Eq. 29). Next, we use this bound to construct an almost supermartingale and use the Robbins-Siegmund Lemma (Lemma 2) to prove convergence. The remaining process is the same for differentiable and non-differentiable functions.

F. Privacy Result Claim 1

Proof. The fundamental reason for privacy in POLAR-SGD is the fact that clients use arbitrary multiplicative and additive perturbations when a stochastic (sub)gradient estimate is uploaded to the parameter server. While a balancing (sub)gradient estimate is uploaded to other parameter servers (due to SLC, Assumption 1), the secure consensus protocol used by parameter servers (POLAR-SGD Server Algorithm, Algo. 2) ensures that no honest-but-curious parameter server can estimate the balancing (sub)gradient estimate during the consensus iteration. The consensus protocol in Algorithm 2 is information theoretically secure due to the wrap-around feature of modulo arithmetic [33]. Hence, the most that an honest-but-curious parameter server can learn about a client is through directly received gradients (Claim 1). □

We further assert that this is an useful notion of privacy. A few discussion points elaborate our assertion.

- **Dataset with rare samples** In most machine learning problems the large size of datasets often compels users to use small batches for learning. The gradient computed over such small batches and supplied to parameter servers can be very informative of the small batch. Consider a machine learning system with a single parameter server performing learning over a medical dataset. The medical dataset has patient information such as residence information, lifestyle choices, symptoms and diagnoses. Assume that there are few samples (patient files) of Cancer patients. If a specific received gradient has non-zero component corresponding to parameters (features) relating to cancer, one can estimate details about the cancer patient based on the current parameters and gradients received. Data samples that are rare are especially vulnerable under such attacks. Under POLAR-SGD algorithm, info about such rare samples is especially well hidden. An honest-but-curious parameter server cannot know if non-zero gradient corresponding to parameters (features) relating to cancer is due to actual data sample or due to additive perturbation.

- **Comparison of POLAR-SGD to Differential Privacy based Learning**
  - As discussed in introduction (Sec I), differential privacy involves adding noise drawn from special distributions (Laplacian, Gaussian with specifically computed parameters) to the gradients. It guarantees that an adversary can uncover data samples that generated the gradient with a small probability characterized by ϵ, however, correctness cannot be guaranteed and the error is $\mathcal{O}(1/\epsilon^2)$ (a fundamental trade-off) [22].
  - We use a different notion of privacy. We guarantee information theoretically that an adversary can only learn $g_{i,k}^{J,h}$, which is a part of the gradient computed by a client. Since the multiplicative and additive perturbations are generated arbitrarily (and not from a specific distribution), estimating them is a hard problem. Moreover, POLAR-SGD converges to the correct optimum and does not suffer from the fundamental privacy-correctness trade-off.
- **Cost of Privacy** Our algorithm POLAR-SGD (minimum-wait) incurs the following overheads as compared to distributed learning with one parameter server (see Figure 2). Unlike differential privacy where accuracy has to be traded-off with privacy, POLAR-SGD has a trade-off between communication (and a small computation) overhead and privacy.
  - Computational Overhead: The gradients are multiplied with arbitrary multiplicative and additive perturbations. This incurs $\mathcal{O}(\tau D)$ cost, where D is the dimension of parameter vector and τ is the minimum number of servers that receives updates from any client. There is extra cost due to secure consensus which involves averaging and modulo operations. However these functions are easy to compute and do not incur any significant overhead.
  - Communication Overhead: POLAR-SGD incurs a small communication overhead. Every clients uploads gradients to τ parameter severs instead of one parameter server (as in Figure 2). We also require $\mathcal{O}(S^2)$ messages for the parameter servers to perform secure consensus.