Fault-Tolerance in Distributed Optimization: The Case of Redundancy

Nirupam Gupta
Department of Computer Science
Georgetown University
Washington DC, USA

Nitin H. Vaidya
Department of Computer Science
Georgetown University
Washington DC, USA

ABSTRACT

This paper considers the problem of Byzantine fault-tolerance in distributed multi-agent optimization. In this problem, each agent has a local cost function. The goal of a distributed optimization algorithm is to allow the agents to collectively compute a minimum of their aggregate cost function. We consider the case when a certain number of agents may be Byzantine faulty. Such faulty agents may not follow a prescribed algorithm, and they may send arbitrary or incorrect information regarding their local cost functions. Unless a fault-tolerance mechanism is employed, traditional distributed optimization algorithms cannot tolerate such faulty agents.

A reasonable goal in presence of faulty agents is to minimize the aggregate cost of the non-faulty agents. However, we show that this goal is *impossible* to achieve *unless* the cost functions of the non-faulty agents have a *minimal redundancy* property. We further propose a distributed optimization algorithm that allows the non-faulty agents to obtain a minimum of their aggregate cost if the *minimal redundancy* property holds. The scope of our algorithm is demonstrated through distributed sensing and learning applications, which are special cases of distributed optimization.

ACM Reference Format:
Nirupam Gupta and Nitin H. Vaidya. 2020. Fault-Tolerance in Distributed Optimization: The Case of Redundancy. In *ACM Symposium on Principles of Distributed Computing (PODC '20), August 3–7, 2020, Virtual Event, Italy.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3382734.3405748

1 INTRODUCTION

The problem of distributed optimization in multi-agent systems has gained significant attention in recent years [6, 13, 21]. In this problem, each agent has a *local* cost function and, when the agents are fault-free, the goal is to design algorithms that allow the agents to collectively minimize the aggregate of their cost functions. To be precise, suppose that there are n agents in the system and let $Q_i(w)$ denote the *local* cost function of agent i, where w is a d-dimensional vector of real numbers, i.e., $w \in \mathbb{R}^d$. A traditional distributed optimization algorithm outputs a *global* minimum w^* such that

$$w^* \in \arg\min_{w} \sum_{i=1}^{n} Q_i(w). \tag{1}$$

As a simple example, $Q_i(w)$ may denote the cost for an agent i (which may be a robot or a person) to travel to location w from their current location, and w^* is a location that minimizes the total cost of meeting for all the agents. Such multi-agent optimization is of interest in many practical applications, including distributed machine learning [6], swarm robotics [26], and distributed sensing [25]. Most of the prior work, however, assumes the agents to be fault-free, i.e., they cooperate and follow a prescribed algorithm. We consider a scenario wherein some of the agents may be faulty.

Su and Vaidya [30] introduced the problem of distributed optimization in the presence of Byzantine faulty agents. The Byzantine faulty agents may behave arbitrarily [19]. In particular, the faulty agents may send incorrect and inconsistent information in order to bias the output of a distributed optimization algorithm. For example, consider an application of multi-agent optimization in the case of distributed sensing where the agents (or *sensors*) observe a common *object* in order to collectively identify the object. However, the faulty agents may send arbitrary observations concocted to prevent the non-faulty agents from making the correct identification [10, 12, 23, 31]. Similarly, in the case of distributed learning, which is another application of distributed optimization, the faulty agents may send incorrect information based on *mislabelled* or arbitrary concocted data points to prevent the non-faulty agents from learning a *good* classifier [1, 2, 4, 8, 9, 11, 15, 34].

We consider the distributed optimization problem in the presence of up to f Byzantine faulty agents. The *ideal* fault-tolerance goal in this case is to design a distributed optimization algorithm that allows all the non-faulty agents to compute a minimum of the aggregate cost of just the non-faulty agents [30]. To be precise, in a given execution, suppose that set $\mathcal{B}$ with $|\mathcal{B}| \leq f$ denotes the set of Byzantine faulty agents, where notation $|\cdot|$ denotes the set cardinality, and $\mathcal{H} = \{1, \ldots, n\} \setminus \mathcal{B}$ denotes the set of non-faulty (i.e., honest) agents. Then, a distributed optimization algorithm achieves *ideal* fault-tolerance if it outputs a point $w^*_{\mathcal{H}}$ such that

$$w^*_{\mathcal{H}} \in \arg\min_{w} \sum_{i \in \mathcal{H}} Q_i(w). \tag{2}$$

Since the non-faulty agents do not necessarily know the identity of the faulty agents, in general, the above ideal objective is unachievable [30]. In this paper, we precisely identify the condition under which the achievability of ideal fault-tolerance is guaranteed. In

PODC '20, August 3–7, 2020, Virtual Event, Italy

ACM ISBN 978-1-4503-7582-5/20/08…$15.00
https://doi.org/10.1145/3382734.3405748

particular, we show that the $2f$-redundancy condition below is necessary and sufficient for achieving the ideal fault-tolerance goal of computing $w^*_{\mathcal{H}}$ that satisfies (2). Recall that there are n agents in the system, of which at most f may be Byzantine faulty.

Definition 1. *[$2f$-redundancy] For a given set of non-faulty agents $\mathcal{H}$, their non-faulty cost functions are said to satisfy $2f$-*redundancy *if for every subset $S \subseteq \mathcal{H}$ of size at least $n-2f$,*

$$\arg\min_{w} \sum_{i \in S} Q_i(w) = \arg\min_{w} \sum_{i \in \mathcal{H}} Q_i(w).$$

The $2f$-redundancy property implies that a minimum of the aggregate of any $n-2f$ non-faulty cost functions is also a minimum of the aggregate of all the non-faulty cost functions, and vice-versa. In addition to showing the necessity of $2f$-redundancy, we present a distributed optimization algorithm that achieves ideal fault-tolerance if the non-faulty cost functions satisfy $2f$-redundancy, and the fraction of the faulty agents is bounded by a threshold dependent on certain properties of the non-faulty cost functions. Additional discussion on $2f$-redundancy, and characterization of fault-tolerance in the *absence* of $2f$-redundancy, can be found in our technical report [16].

Realizing $2f$-redundancy: Although the $2f$-redundancy property may appear somewhat technical at this point, we note that, in many practical applications, redundancy in cost functions occurs naturally. Indeed, such redundancy is easily realized in practical applications such as distributed sensing, and distributed learning (or *federated learning* [18]). We discuss both distributed sensing and learning briefly in Section 1.2. A detailed presentation of fault-tolerance in these applications can be found in our technical report [14].

System architecture: The contributions of this paper apply to two different system architectures illustrated in Figure 1. In the server-based architecture, the server is assumed to be trustworthy, but up to f agents may be Byzantine faulty. The trusted server helps solve the distributed optimization problem in coordination with the agents. In the peer-to-peer architecture, the agents are connected to each other by a complete network, and up to f of these agents may be Byzantine faulty. Provided that $f < \frac{n}{3}$, any algorithm for the server-based architecture can be simulated in the peer-to-peer system using the well-known *Byzantine broadcast* primitive [20]. For simplicity of presentation, the rest of this paper considers the server-based architecture.

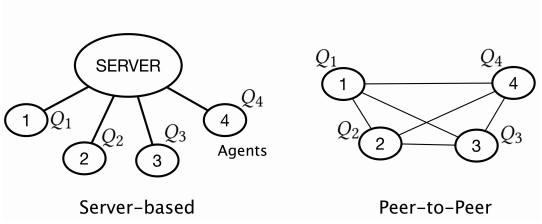

Figure 1: The system architectures.

1.1 Summary of our contributions

We make the following two major contributions.

- **An impossibility result:** We show that a minimum point of the sum of cost functions of non-faulty agents $\sum_{i \in \mathcal{H}} Q_i(w)$ cannot be computed accurately by the non-faulty agents if the non-faulty cost functions $\{Q_i(w),\ i \in \mathcal{H}\}$ violate the $2f$-*redundancy* property. This impossibility result is formally presented in Section 3.
- **A fault-tolerant distributed algorithm:** We propose a distributed optimization algorithm that allows the non-faulty agents to compute a minimum point

  $$w^*_{\mathcal{H}} \in \arg\min_{w} \sum_{i \in \mathcal{H}} Q_i(w),$$

  if the non-faulty cost functions have $2f$-redundancy.
  The key component of our algorithm is a *gradient-filter* named *comparative gradient clipping* (CGC) that *robustifies* the traditional gradient-descent method. In each iteration t, the trusted server maintains an estimate w^t of the desired minimum, and updates it iteratively using gradients sent by the agents corresponding to their local cost functions at w^t. However, instead of using the received gradients directly, the server applies the CGC filter on the gradients, and uses the *filtered* gradients. In particular, the server sorts the n gradients received from the n agents, and clips the gradients with the f largest norms such that their norm equals the $(f+1)$-th largest norm among the n gradient norms (the remaining $n-f$ gradients are not altered). The estimate w^t is then updated along the direction opposite to the aggregate of the resultant filtered gradients. The algorithm schematic is shown in Figure 2, with more details in Section 4.

Unlike previously introduced applications of *gradient clipping* (elaborated in Section 2) for solving other unrelated problems in the gradient-descent method [24, 28], our CGC gradient-filter employs an *adaptive* threshold for clipping. Specifically, the clipping threshold in our case is not a constant but varies depending upon the magnitude of the non-faulty gradients. Section 2 discusses the other work on gradient clipping.

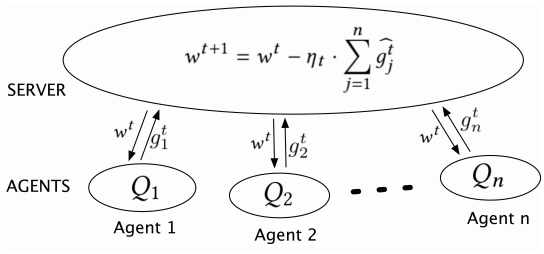

Figure 2: Schematic of the algorithm.

1.2 Realizing $2f$-Redundancy in Practice

In this section, we briefly discuss two applications where the necessary property of $2f$-redundancy can be realized quite naturally,

namely, distributed sensing and distributed learning.

Distributed Sensing: As a simple example, consider the problem of correctly identifying an aircraft in the sky using images taken from multiple cameras. Even if some number k of strategically placed cameras might suffice to identify the aircraft, redundancy can be achieved by deploying more than k cameras. Applications of distributed sensing abound in cyber-physical systems such as the power grid or unmanned mobile vehicles. In general, sensors are deployed to measure *state* of the cyber-physical system under observation, and redundancy is introduced to ensure robustness [12, 22, 23]. Commonly, in these systems each agent (sensor) makes some *partial observations* of the system's *state* ω such that the agents can collectively compute ω uniquely. Redundancy is achieved by deploying more sensors than minimally necessary to identify the system state correctly. In particular, suppose that each agent i has a set of m_i *observations*, represented by $(X_i,\ Y_i)$, where X_i is an $m_i \times d$ matrix, and Y_i is a m_i-dimensional column vector. In many systems, X_i and Y_i are linearly related through the state ω as $Y_i = X_i\omega$. The goal then is to collectively determine ω. We define the local cost function of each agent i as $Q_i(w) = \|Y_i - X_i w\|^2$, where $\|\cdot\|$ denotes the Euclidean norm. The agents' observations (i.e., $(X_i,\ Y_i)$'s) collectively need to be sufficient to determine ω uniquely – specifically, the matrix obtained by stacking X_i's of all the agents needs to be full column rank [17]. When this property holds and the agents are fault-free, then it is easy to show that ω is the minimum of the aggregate of all the agents' costs. That is, $\omega = \arg\min_w \sum_{i=1}^{n} Q_i(w)$.

When some of the agents are Byzantine faulty, they may maliciously provide incorrect information, which can make it difficult to correctly determine the system's state ω. Our results imply that the correct state ω can be computed in the presence of Byzantine faulty agents *only if* the non-faulty cost functions satisfy the $2f$-*redundancy* property. Equivalently, the state ω must be uniquely determinable using observations of any $n - 2f$ non-faulty agents. A detailed presentation of fault-tolerance in distributed sensing, including the case of noisy observations, can be found in our technical report [14].

Distributed Learning: In case of distributed learning, the goal for the agents is to collectively compute a learning parameter w^* that minimizes a *global* cost function $Q(w)$, namely, the *expected loss* incurred for a learning parameter w [5]. Specifically, suppose for each *data point* z the loss function for w is denoted by $\ell(w; z)$. If the probability distribution of the *data points* is $\mathcal{D}$ then

$$Q(w) \triangleq \mathbb{E}_{z\sim\mathcal{D}}\ \ell(w; z)$$

where notation $\mathbb{E}_{z\sim\mathcal{D}}$ denotes the expectation with respect to the random data point z. The goal is to determine a learning parameter w^* such that $w^* \in \arg\min_w\ Q(w)$. A commonly used distributed learning method relies on *stochastic* gradient descent (SGD) using the server-based architecture [5]. In each iteration t, the server maintains an estimate w^t of w^* and sends its current estimate to the agents. A non-faulty agent i samples a data point $z \sim \mathcal{D}$ and returns the stochastic gradient $g_i^t = \nabla\ell(w^t; z)$ to the server. The server updates w^t using the stochastic gradients received from all the agents. However, if an agent is faulty then it may send arbitrary vectors instead of the true stochastic gradients to prevent the server from learning w^*, or worse, learn an incorrect parameter [1, 4, 9, 11].

In the above distributed learning setting the cost functions of all the non-faulty agents are identical in expectation. Therefore, $2f$-redundancy property holds trivially when $f < n/2$. However, in this case the non-faulty agents send *stochastic* gradients of the global cost function $Q(w)$, instead of the full gradients $\nabla Q(w^t)$. Nevertheless, the distributed algorithm presented in this paper can guarantee fault-tolerance in the SGD-based distributed learning framework, with minor modifications, provided that the stochastic gradients satisfy the standard assumptions [5]. A detailed presentation of the modified algorithm and its fault-tolerance property can be found in the appendix of our technical report [14].

2 OTHER RELATED WORK

The prior work on fault-tolerance in multi-agent distributed optimization, such as Su and Vaidya, 2016 [30], and Sundaram and Gharesifard, 2018 [33], only consider *approximate* fault-tolerance where the agents compute a point that is an approximate minimum of the non-faulty aggregate cost. For instance, Su and Vaidya, 2016 [30] proposed a distributed optimization algorithm that outputs a minimum of the *non-uniformly* weighted aggregate of the non-faulty cost functions, instead of the actual uniformly weighted aggregate. Moreover, these works only consider *univariate* cost functions. In contrast to these works, we consider the more general class of *multivariate* cost functions, and present results for minimization of the actual aggregate cost of the non-faulty agents.

There is also some work on *approximate* fault-tolerance for multivariate cost functions [29, 32, 35]. However, each of these works only consider degenerate cases of the distributed optimization problem presented above. To be specific, Su and Vaidya, 2016 [32] consider a setting where the individual cost functions are assumed to be linear combinations of a common set of *basis functions*. Yang and Bajwa, 2017 [35] rely on the assumption that the individual cost functions can be split into independent univariate *strictly convex* functions. We do not make such assumptions on the cost functions, and present tight results for computing the exact minimum (not an approximation) of the non-faulty aggregate cost.

Su and Shahrampour, 2018 [29] consider the problem of fault-tolerance for the case of distributed linear sensing, which, as we have mentioned above, is a special case of distributed optimization. The fault-tolerance guarantee of their proposed distributed algorithm relies on an additional assumption, besides $2f$-redundancy, on the observations of non-faulty agents. In comparison to [29], we consider the more general case of distributed optimization. When applied to the special case of distributed linear sensing, our presented algorithm guarantees fault-tolerance if $2f$-redundancy holds, and the fraction of faulty agents f/n is smaller than a threshold which is inversely proportional to the *condition number*, i.e., the ratio between the maximum and the minimum singular values, of the non-faulty *observation matrix* [14].

Subsequent to the initial work on fault-tolerant distributed optimization by Su and Vaidya [30], the problem of fault-tolerance in distributed *learning* has gained significant attention in recent years [1, 2, 4, 8, 11, 34]. However, distributed learning is only a special case of the more general distributed optimization problem that we consider. As elaborated in our technical report [14], our algorithm (with minor modifications) can also guarantee fault-tolerance in the case of distributed learning. Importantly, the computational complexity of our proposed fault-tolerance mechanism, i.e., the CGC gradient-filter, is $O(n(d + \log n))$, which is significantly lower than that of the fault-tolerance mechanisms proposed in some of these prior works [1, 4, 8, 11]. Moreover, our algorithm applies to more general distributed learning settings that satisfy $2f$-redundancy.

Gradient clipping using static thresholds has been utilized in the past for solving unrelated problems, such as controlling the *gradient explosion* and the privacy-accuracy trade-offs in *differential privacy* protocols, in the context of learning [24, 28]. We use gradient clipping for Byzantine fault-tolerance. Besides this difference in the objectives, note that unlike the past works, we use a dynamic (or adaptive) threshold for clipping the gradients. This difference is important for the performance characteristics of our algorithm.

3 NECESSITY OF $2f$-REDUNDANCY

The notion of *ideal fault-tolerance* is formalized by f*-resilience* defined as follows:

Definition 2. *[f-resilience] A distributed optimization algorithm is* f-resilient *if it outputs a minimum of the aggregate cost of all the non-faulty agents, despite the presence of up to f Byzantine faulty agents.*

Specifically, if $\mathcal{B} \subset \{1, \ldots, n\}$ with $|\mathcal{B}| \leq f$ denotes the set of faulty agents, and $\mathcal{H} = \{1, \ldots, n\} \setminus \mathcal{B}$ denotes the set of non-faulty agents, then an f*-resilient* distributed optimization algorithm outputs a point in the set

$$\arg\min_w \sum_{i \in \mathcal{H}} Q_i(w).$$

To prove the necessity of $2f$-redundancy for achievability of f-resilience, we assume that the non-faulty cost functions are differentiable and convex. The necessary condition is formally stated as follows.

Theorem 1. *Suppose that the non-faulty cost functions are differentiable and convex. There exists a deterministic* f-resilient *distributed optimization algorithm* only if *the non-faulty cost functions satisfy the* $2f$-redundancy *property.*

3.1 Proof of Theorem 1

We assume that $f > 0$, since the proof is trivial for $f = 0$. We show the necessity of $2f$*-redundancy* property for f*-resilience* by assuming the best-case scenario for the server where the agents send complete information about their local cost functions. The faulty agents may send arbitrary information. In general distributed setting, the server may only have partial information about the agents' local cost functions, and the faulty agents need not commit to well-defined cost functions. Therefore, if there cannot exist an f*-resilient* distributed optimization algorithm in this best-case scenario then the same holds true for the general distributed setting. Let the cost functions received by the server from the agents be $\{C_1, \ldots, C_n\}$, i.e., agent i sends cost function C_i. If agent i is non-faulty then $C_i = Q_i$, otherwise C_i may be an arbitrary differentiable and convex function. Note that if a faulty agent sends a cost function that is not differentiable or convex then the server can determine that the agent is faulty, and eliminate it from the computation.

We will show that if there exists a deterministic distributed optimization algorithm that is f*-resilient*, i.e., it outputs a point that minimizes the aggregate of non-faulty agents correctly, despite up to f (Byzantine) faulty cost functions amongst the received cost functions $\{C_1, \ldots, C_n\}$, then the cost functions of the non-faulty agents satisfy the $2f$*-redundancy* property.

The proof of the theorem relies on the following fundamental lemma. The proof of Lemma 1 is presented in Appendix A.

Lemma 1. *For every non-empty subset S of $\{1, \ldots, n\}$, if*

$$\bigcap_{i \in S} \arg\min_w \; Q_i(w) \neq \emptyset$$

then

$$\arg\min_w \sum_{i \in S} Q_i(w) = \bigcap_{i \in S} \arg\min_w \; Q_i(w).$$

The remainder of the proof is divided into three parts.

Part I: In this part we will show that if there exists a deterministic f*-resilient* algorithm then for every non-empty set $S \subseteq \{1, \ldots, n\}$,

$$\bigcap_{i \in S} \arg\min_w \; Q_i(w) = \arg\min_w \sum_{i \in S} Q_i(w). \tag{3}$$

Suppose that a deterministic optimization algorithm named Π is f*-resilient*. As the identity of the Byzantine faulty agents is a priori unknown to the sever, and the faulty cost functions are assumed differentiable and convex, Π cannot distinguish between the following two possibilities: (1) $S_1 = \{1, \ldots, n - f\}$ is the set of non-faulty agents, and (2) $S_2 = \{2, \ldots, n - f + 1\}$ is the set of non-faulty agents. As Π is a deterministic algorithm and the set of cost functions $\{C_1, \ldots, C_n\}$ is same in both cases, the output of Π is same for both the cases. Specifically, if w^* denotes the output of the algorithm Π then

$$\begin{aligned} w^* &\in \arg\min_w \sum_{i \in S_1} Q_i(w), \\ w^* &\in \arg\min_w \sum_{i \in S_2} Q_i(w) \end{aligned} \tag{4}$$

As Q_i's are differentiable, (4) implies that

$$\sum_{i \in S_1} \nabla Q_i(w^*) = \mathbf{0} = \sum_{i \in S_2} \nabla Q_i(w^*), \tag{5}$$

where $\mathbf{0}$ denotes the d-dimensional zero vector. As $S_1 \setminus \{1\} = S_2 \setminus \{n - f + 1\}$, (5) implies that $\nabla Q_1(w^*) = \nabla Q_{n-f+1}(w^*)$. Using similar arguments as above for other choices of sets S_1 and S_2 we can show that

$$\nabla Q_i(w^*) = \nabla Q_j(w^*), \; \forall i, j \in \{1, \ldots, n\}. \tag{6}$$

From substituting from (6) in (5) we obtain that

$$\nabla Q_i(w^*) = \mathbf{0}, \quad \forall i \in \{1, \ldots, n\}, \tag{7}$$

As Q_i's are convex functions, (7) implies that $w^* \in \arg\min_w Q_i(w)$ for all $i \in \{1, \ldots, n\}$. Equivalently,

$$w^* \in \bigcap_{i=1}^{n} \arg\min_w Q_i(w). \tag{8}$$

Consider an arbitrary non-empty subset S of $\{1, \ldots, n\}$. The above, i.e., (8), implies that the intersection $\bigcap_{i\in S} \arg\min_w Q_i(w)$ is non-empty. Therefore, due to Lemma 1,

$$\bigcap_{i\in S} \arg\min_w Q_i(w) = \arg\min_w \sum_{i\in S} Q_i(w).$$

This proves (3).

Part II: Suppose that $\mathcal{H}$ denotes the set of non-faulty agents in an execution of the algorithm Π. Recall that $|\mathcal{H}| \geq n - f$ and $n > 2f$. Let,

$$\mathcal{W}^* = \arg\min_w \sum_{i\in\mathcal{H}} Q_i(w). \tag{9}$$

In this part, we will show that for every subset $T \subset \mathcal{H}$ of size $n-2f$,

$$\mathcal{W}^* = \bigcap_{i\in T} \arg\min_w Q_i(w). \tag{10}$$

From (3), we know that

$$\mathcal{W}^* = \bigcap_{i\in\mathcal{H}} \arg\min_w Q_i(w). \tag{11}$$

Due to the non-expansion property of set intersection, for every non-empty subset $S \subseteq \mathcal{H}$,

$$\mathcal{W}^* = \bigcap_{i\in\mathcal{H}} \arg\min_w Q_i(w) \subseteq \bigcap_{i\in S} \arg\min_w Q_i(w). \tag{12}$$

Next, using contradiction, we will show the converse of (12) for an arbitrary set $\widehat{S} \subset \mathcal{H}$ of size $n - 2f$.

Let $\widehat{S}$ be an arbitrary subset of $\mathcal{H}$ of size $n - 2f$. Note that, due to (12), the set $\bigcap_{i\in\widehat{S}} \arg\min_w Q_i(w)$ is non-empty. Suppose that there exists a point

$$\widehat{w} \in \bigcap_{i\in\widehat{S}} \arg\min_w Q_i(w), \text{ such that } \widehat{w} \notin \mathcal{W}^*.$$

Suppose that $\mathcal{B}$ is the set of Byzantine faulty agents such that $|\mathcal{B}| = f$ and

$$\mathcal{B} \subset \{1, \ldots, n\} \setminus \widehat{S}.$$

Then, $|\mathcal{H}| = |\{1, \ldots, n\} \setminus \mathcal{B}| = n - f$. For each $j \in \mathcal{B}$, consider a differentiable and convex cost function C_j that has a unique minimum at $\widehat{w}$; thus, C_j does not minimize at any point in $\mathcal{W}^*$. Let,

$$T_1 = \mathcal{H}, \text{ and } T_2 = \widehat{S} \cup \mathcal{B}.$$

Note that both sets T_1 and T_2 are of size $n - f$. Due to (11) and Lemma 1, the aggregate of the cost functions of the agents in T_1 minimizes only at points in $\mathcal{W}^*$. On the other hand, the cost functions of all the agents in the set T_2 only minimize at $\widehat{w} \notin \mathcal{W}^*$. Therefore, due to Lemma 1, the aggregate of the cost functions of the agents in set $T_2 = \widehat{S} \cup \mathcal{B}$ only minimize at $\widehat{w}$, and does not minimize at any point in $\mathcal{W}^*$.

Now, consider two possible executions: (1) T_1 is the set of non-faulty agents, and (2) T_2 is the set of non-faulty agents. As all the cost functions in both T_1 and T_2 are differentiable and convex, and the identity of $\mathcal{B}$ is a priori unknown, Π cannot distinguish between these two executions. Being deterministic, the output of Π is identical for both the executions. Suppose that Π outputs a point in $\mathcal{W}^*$, then the aggregate of the non-faulty agents T_2 in the second possible execution is not minimized. On the other hand, if Π outputs $\widehat{w}$ then the aggregate of the non-faulty agents T_1 in the first possible execution is not minimized. Therefore, the assumption that Π is f*-resilient*, i.e., the output of Π minimizes the aggregate of the non-faulty agents, is always violated in one of the two executions.

The above argument implies that there cannot exist a point $\widehat{w}$ in the non-empty intersection $\bigcap_{i\in\widehat{S}} \arg\min_w Q_i(w)$ such that $\widehat{w} \notin \mathcal{W}^*$. Equivalently,

$$\bigcap_{i\in\widehat{S}} \arg\min_w Q_i(w) \subseteq \mathcal{W}^*.$$

This, in conjunction with (12), implies that

$$\mathcal{W}^* = \bigcap_{i\in\widehat{S}} \arg\min_w Q_i(w).$$

As $\widehat{S}$ is an arbitrary subset of $\mathcal{H}$ with $\left|\widehat{S}\right| = n - 2f$, the above proves (10).

Part III: Recall the definition of $\mathcal{W}^*$ from (9). In this last part, we will show that (10) implies $2f$*-redundancy*, i.e., for every set $S \subseteq \mathcal{H}$ of size at least $n - 2f$,

$$\arg\min_w \sum_{i\in S} Q_i(w) = \mathcal{W}^*. \tag{13}$$

Consider an arbitrary set $S \subseteq \mathcal{H}$ with $|S| \geq n - 2f > 0$. Recall that, due to (12), the set $\bigcap_{i\in S} \arg\min_w Q_i(w)$ is non-empty. Due to the non-expansion property of set intersection, for every set $T \subseteq S$ with $|T| = n - 2f$ we obtain that

$$\emptyset \neq \bigcap_{i\in S} \arg\min_w Q_i(w) \subseteq \bigcap_{i\in T} \arg\min_w Q_i(w). \tag{14}$$

Upon substituting from (10) in (14) we obtain that

$$\bigcap_{i\in S} \arg\min_w Q_i(w) \subseteq \mathcal{W}^*, \quad \forall S \subseteq \mathcal{H},\ |S| \geq n - 2f. \tag{15}$$

From (12) and (15) we obtain that

$$\bigcap_{i\in S} \arg\min_w Q_i(w) = \mathcal{W}^*, \quad \forall S \subseteq \mathcal{H},\ |S| \geq n - 2f.$$

Finally, using Lemma 1 above proves (13). ■

4 PROPOSED ALGORITHM

In this section, we present our distributed optimization algorithm and its fault-tolerance property under $2f$-redundancy. For now, we will assume the system to be synchronous. However, as elaborated in Section 5, our algorithm, upon slight modifications, can also be executed in a *partially asynchronous* system [3], without the fault-tolerance property being affected. Details on the fault-tolerance

property of our algorithm under *relaxed* $2f$-redundancy and *partial asynchronism* can be found in the technical report [14].

The algorithm is iterative. The server maintains an estimate of a minimum point defined by (2), which is updated in each iteration of the algorithm. The initial estimate, named w^0, is chosen arbitrarily by the server from $\mathbb{R}^d$. In iteration $t \in \{0, 1, \ldots\}$, the server computes estimate w^{t+1} using Steps S1 and S2 presented below. Please refer Figure 2 for an overview of the algorithm.

In Step S1 below, the server obtains from the agents the gradients of their local cost functions at w^t. A Byzantine faulty agent may send an arbitrary value as its gradient. In Step S2, to mitigate the detrimental impact of such incorrect gradients, the algorithm uses a filter to robustify the gradient aggregation step. In particular, as presented in Equation (16) in the algorithm below, the gradients with the largest f norms are clipped so that their norm equals the norm of the $(f+1)$-th largest gradient. The remaining gradients remain unchanged. The resulting gradients are then accumulated to obtain the update direction, which is then used to compute w^{t+1} (Equation (17)). We refer to the method used in Step S2 for clipping the largest f gradients as *Comparative Gradient Clipping* (CGC), since the largest f gradients are clipped to a norm that is *comparable* to the next largest gradient.

Steps performed in the t-th iteration, $t \geq 0$:

S1: The server requests from each agent the gradient of its local cost function at the current estimate w^t. Each non-faulty agent i will then send to the server the gradient $\nabla Q_i(w^t)$, whereas a faulty agent may send an incorrect arbitrary value for the gradient.
The gradient received by the server from agent i is denoted as g_i^t. If no gradient is received from some agent i, then agent i must be faulty (because the system is assumed to be synchronous) – in this case, the server assumes a default value of $\mathbf{0}$ vector for the missing gradient g_i^t.

S2: CGC gradient filter: The server sorts the n received gradients as follows:

$$\left\|g_{i_1}^t\right\| \leq \ldots \leq \left\|g_{i_{n-f}}^t\right\| \leq \left\|g_{i_{n-f+1}}^t\right\| \leq \ldots \leq \left\|g_{i_n}^t\right\|$$

where, $\|\cdot\|$ denotes the Euclidean norm. Thus, the gradient with the smallest norm, $g_{i_1}^t$, is received from agent i_1, and the gradient with the largest norm, $g_{i_n}^t$, is received from agent i_n.

If $\left\|g_{i_{n-f}}^t\right\| = 0$, the algorithm terminates[1] and outputs w^t. Otherwise, the server computes the "scaled" gradients, denoted by $\widehat{g_j^t}$, as

$$\widehat{g_j^t} = \begin{cases} \dfrac{\left\|g_{i_{n-f}}^t\right\|}{\left\|g_j^t\right\|} g_j^t & , \quad j \in \{i_{n-f+1}, \ldots, i_n\} \\ g_j^t & , \quad j \in \{i_1, \ldots, i_{n-f}\} \end{cases} \tag{16}$$

and updates its estimate,

$$w^{t+1} = w^t - \eta \sum_{j=1}^{n} \widehat{g_j^t} \tag{17}$$

Notation η denotes the (constant) step-size of positive value.

Complexity: The computation of the norm of the n gradients takes $O(nd)$ time. Sorting of these norms takes additional $O(n \log n)$ time. Hence, the time complexity of the CGC filter is $O(n(d + \log n))$.

4.1 Assumptions

To derive the fault-tolerance property of the algorithm we make Assumptions 1- 4 stated below. For a given execution of our algorithm, let the sets of faulty and non-faulty agents be denoted by $\mathcal{B}$ and $\mathcal{H}$, respectively. Recall that $|\mathcal{B}| \leq f$ and $n - 2f > 0$. Henceforth, we write '$\min_w$' simply as 'min', unless otherwise stated.

Assumption 1 (*synchronicity*). *We assume that the system under consideration is synchronous.*

Assumption 2 (*convexity and differentiability*). *Assume that the non-faulty cost functions $\{Q_i(w),\ i \in \mathcal{H}\}$ are differentiable and convex. This implies that [7], for every i, the gradient $\nabla Q_i(w)$ exists for all $w \in \mathbb{R}^d$, and $w^* \in \arg\min Q_i(w)$ if and only if $\nabla Q_i(w^*) = 0$.*

Assumption 3 (*Lipschitz continuous gradients*). *Assume that there exists $\mu > 0$ such that for every $i \in \mathcal{H}$,*

$$\left\|\nabla Q_i(w) - \nabla Q_i(w')\right\| \leq \mu \left\|w - w'\right\|, \quad \forall w,\ w' \in \mathbb{R}^d.$$

Let, $\mathcal{W}^*$ denote the set $\arg\min_w \sum_{i \in \mathcal{H}} Q_i(w)$. Convexity of the non-faulty cost functions implies that the set $\mathcal{W}^*$ is *closed* and *convex* [7]. Therefore, the Euclidean projection of every point w onto $\mathcal{W}^*$ exists and is unique [7]. Let $[w]_{\mathcal{W}^*}$ denote the projection of w onto $\mathcal{W}^*$. Then,

$$[w]_{\mathcal{W}^*} = \arg\min_{x \in \mathcal{W}^*} \|x - w\|. \tag{18}$$

Let $\overline{Q}(w)$ denote the *average* cost of the non-faulty agents, i.e.,

$$\overline{Q}(w) = \frac{1}{|\mathcal{H}|} \sum_{i \in \mathcal{H}} Q_i(w) \quad \forall w, \tag{19}$$

where $|\cdot|$ denotes the size of a finite set.

Assumption 4 (*relaxed* strong convexity). *Assume that there exists $\lambda > 0$, such that*

$$\left\langle \nabla \overline{Q}(w),\ w - [w]_{\mathcal{W}^*} \right\rangle \geq \lambda \left\|w - [w]_{\mathcal{W}^*}\right\|^2, \quad \forall w \in \mathbb{R}^d.$$

It should be noted that when $\mathcal{W}^*$ is singleton, i.e., the minimum of the aggregate of the non-faulty agents' cost functions is unique, then Assumption 4 implies that the average non-faulty cost function $\overline{Q}(w)$ is strongly convex.

[1] An alternate, and perhaps more practical, finite-iteration stopping criterion for the algorithm can be derived using the convergence result presented in Section 4.2.

4.2 Fault-Tolerance Property

We now present the correctness and the convergence of our algorithm. For doing so, we introduce the following notation.

- Let α, defined as follows, denote the *fault-tolerance margin.*

$$\alpha = \frac{\lambda}{3\mu} - \frac{f}{n}. \tag{20}$$

  The algorithm is shown to be correct when $\alpha > 0$, or equivalently, $f/n < \lambda/3\mu$.
- Let,

$$\overline{\eta} = 6\alpha/n\mu, \text{ and} \tag{21}$$

$$\rho = 1 - (n\mu)^2\, \eta\, (\overline{\eta} - \eta)\,. \tag{22}$$

  As elaborated in Appendix B, under $2f$*-redundancy* property and Assumptions 2-4, if $\alpha > 0$ and $0 < \eta < \overline{\eta}$ then $\rho \in [0,\ 1)$.

Theorem 2 below shows the correctness of our algorithm.

Theorem 2. *Consider the algorithm presented in Section 4. Suppose that the* $2f$-redundancy *property and Assumptions 1-4 hold true. If* $\alpha > 0$, *and* $\eta \in (0,\ \overline{\eta})$ *in* (17), *then*

(i) $\rho \in [0,\ 1)$, *and*

(ii) $\left\|w^t - [w^t]_{\mathcal{W}^*}\right\|^2 \leq \rho^t \left\|w^0 - [w^0]_{\mathcal{W}^*}\right\|^2$, *for all* $t \geq 0$.

The proof of Theorem 2 is deferred to Appendix C.

Theorem 2 implies that sequence of estimates $\{w^t\}$ generated by (17) converges *linearly* to a minimum of the non-faulty aggregate cost $\sum_{i\in\mathcal{H}} Q_i(w)$. The *rate of convergence* is equal to ρ. Thus, smaller is the value of ρ, faster is the convergence. We observe below that the value of ρ reduces, i.e., the convergence speed improves, as the fault-tolerance margin α increases.

From (22), note that $\rho = 1 + (\eta n\mu)^2 - \overline{\eta}\eta(n\mu)^2$. Upon substituting the value of $\overline{\eta}$ from (21) we obtain that

$$\rho = 1 + (\eta n\mu)^2 - 6(\eta n\mu)\,\alpha.$$

Therefore, as the value of α increases the value of ρ decreases, and vice-versa.

5 PARTIAL ASYNCHRONISM

In this section, we present an extension of our algorithm presented in Section 4 for the case when the system may be *partially asynchronous* [3]. In this case, for an iteration, the server may not receive gradients from all the agents in a timely manner. However, for a partially asynchronous system it is assumed that the number of iterations elapsed between two contiguous gradients received from a non-faulty agent is bounded [3]. Our algorithm can be executed in such a partially asynchronous system, with the following modifications, without its fault-tolerance property getting affected.

In **Step S1**, upon sending the current estimate w^t to all the agents, if the server does not receive a gradient from an agent i in a timely manner then, instead of marking i as faulty and using the $\mathbf{0}$ vector as its gradient, the server uses the most recent gradient received from agent i. To be specific, for an iteration t and an agent i, let $s_i(t)$ denote the iteration in which the server received the most recent gradient from agent i. Thus, $s_i(t) = t$ if the server received a gradient from agent i in iteration t, otherwise $s_i(t) < t$. For an iteration t, if the server has not received any gradient from an agent i in iterations $\{0, \ldots,\ t\}$ then the server assigns $g_i^t = \mathbf{0}$. Else, the server assigns $g_i^t = g_i^{s_i(t)}$. **Step S2** remains unchanged, except the update rule (17) in which the constant step-size η is replaced by a varying step-size η_t such that $\eta_t > 0$ for all t.

As elaborated in our technical report [14, Section 7.2], if the sequence of step-sizes $\{\eta_t\}$ satisfies the conditions: $\sum_{t=0}^{\infty} \eta_t = \infty$ and $\sum_{t=0}^{\infty} \eta_t^2 < \infty$, then the algorithm presented in Section 4, with the above modifications, converges asymptotically to $\mathcal{W}^*$, provided that the system is only partially asynchronous, i.e., for each non-faulty agent i the difference $t - s_i(t)$ is bounded for all t. Note that step-size $\eta_t = 1/(t+1)$ satisfies the above conditions [27].

6 SUMMARY

This paper addresses the problem of fault-tolerance in distributed multi-agent optimization. It is shown that the $2f$*-redundancy* property is necessary for *ideal fault-tolerance* in the presence of up to f Byzantine faulty agents, i.e., f-resilience. An efficient distributed optimization algorithm is proposed that guarantees f-resilience, provided that the $2f$-redundancy property holds true. The scope of our algorithm is demonstrated through distributed sensing and learning applications, which are special cases of distributed optimization.

Details on fault-tolerance in the absence of $2f$-redundancy are presented in our technical report [16]. Also, our technical report [14] presents extensions on the fault-tolerance property of our algorithm under relaxed $2f$-redundancy and partial asynchronism.

ACKNOWLEDGEMENTS

Research reported in this paper was sponsored in part by the Army Research Laboratory under Cooperative Agreement W911NF- 17-2-0196, and by the National Science Foundation award 1842198.[2]

REFERENCES

[1] Dan Alistarh, Zeyuan Allen-Zhu, and Jerry Li. Byzantine stochastic gradient descent. In *Advances in Neural Information Processing Systems*, 2018.
[2] Jeremy Bernstein, Jiawei Zhao, Kamyar Azizzadenesheli, and Anima Anandkumar. signsgd with majority vote is communication efficient and Byzantine fault tolerant. *arXiv preprint arXiv:1810.05291*, 2018.
[3] Dimitri P Bertsekas and John N Tsitsiklis. *Parallel and distributed computation: numerical methods*, volume 23. Prentice hall Englewood Cliffs, NJ, 1989.
[4] Peva Blanchard, Rachid Guerraoui, Julien Stainer, et al. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, pages 119–129, 2017.
[5] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.
[6] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1), 2011.
[7] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
[8] Xinyang Cao and Lifeng Lai. Distributed gradient descent algorithm robust to an arbitrary number of byzantine attackers. *IEEE Transactions on Signal Processing*, 67(22):5850–5864, 2019.

[2] The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory, National Science Foundation or the U.S. Government.

[9] Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 47–60, 2017.

[10] Yuan Chen, Soummya Kar, and Jose MF Moura. Resilient distributed estimation through adversary detection. *IEEE Transactions on Signal Processing*, 66(9), 2018.

[11] Yudong Chen, Lili Su, and Jiaming Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2):44, 2017.

[12] Michelle S Chong, Masashi Wakaiki, and Joao P Hespanha. Observability of linear systems under adversarial attacks. In *American Control Conference*, pages 2439–2444. IEEE, 2015.

[13] John C Duchi, Alekh Agarwal, and Martin J Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic control*, 57(3):592–606, 2011.

[14] Nirupam Gupta and Nitin H Vaidya. Byzantine fault tolerant distributed linear regression. *arXiv preprint arXiv:1903.08752*, 2019.

[15] Nirupam Gupta and Nitin H Vaidya. Byzantine fault-tolerant parallelized stochastic gradient descent for linear regression. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 415–420. IEEE, 2019.

[16] Nirupam Gupta and Nitin H Vaidya. Resilience in collaborative optimization: redundant and independent cost functions. *arXiv preprint arXiv:2003.09675*, 2020.

[17] Roger A Horn, Roger A Horn, and Charles R Johnson. *Matrix analysis*. Cambridge university press, 1990.

[18] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[19] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3), 1982.

[20] Nancy A Lynch. *Distributed algorithms*. Elsevier, 1996.

[21] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1), 2009.

[22] Miroslav Pajic, Insup Lee, and George J Pappas. Attack-resilient state estimation for noisy dynamical systems. *IEEE Transactions on Control of Network Systems*, 4(1):82–92, 2017.

[23] Miroslav Pajic, James Weimer, Nicola Bezzo, Paulo Tabuada, Oleg Sokolsky, Insup Lee, and George J Pappas. Robustness of attack-resilient state estimators. In *ICCPS'14: ACM/IEEE 5th International Conference on Cyber-Physical Systems (with CPS Week 2014)*, pages 163–174. IEEE Computer Society, 2014.

[24] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. Understanding the exploding gradient problem. *CoRR, abs/1211.5063*, 2, 2012.

[25] Michael Rabbat and Robert Nowak. Distributed optimization in sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 20–27, 2004.

[26] Robin L Raffard, Claire J Tomlin, and Stephen P Boyd. Distributed optimization for cooperative agents: Application to formation flight. In *2004 43rd IEEE Conference on Decision and Control (CDC)*, volume 3, pages 2453–2459. IEEE, 2004.

[27] Walter Rudin. *Principles of mathematical analysis*, volume 3. McGraw-hill New York, 1964.

[28] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321. ACM, 2015.

[29] Lili Su and Shahin Shahrampour. Finite-time guarantees for Byzantine-resilient distributed state estimation with noisy measurements. *arXiv preprint arXiv:1810.10086*, 2018.

[30] Lili Su and Nitin H Vaidya. Fault-tolerant multi-agent optimization: optimal iterative distributed algorithms. In *Proceedings of the 2016 ACM symposium on principles of distributed computing*, pages 425–434. ACM, 2016.

[31] Lili Su and Nitin H Vaidya. Non-bayesian learning in the presence of byzantine agents. In *International symposium on distributed computing*. Springer, 2016.

[32] Lili Su and Nitin H Vaidya. Robust multi-agent optimization: coping with Byzantine agents with input redundancy. In *International Symposium on Stabilization, Safety, and Security of Distributed Systems*, pages 368–382. Springer, 2016.

[33] Shreyas Sundaram and Bahman Gharesifard. Distributed optimization under adversarial nodes. *IEEE Transactions on Automatic Control*, 2018.

[34] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Generalized Byzantine-tolerant sgd. *arXiv preprint arXiv:1802.10116*, 2018.

[35] Zhixiong Yang and Waheed U. Bajwa. Byrdie: Byzantine-resilient distributed coordinate descent for decentralized learning, 2017.

A PROOF OF LEMMA 1

Consider an arbitrary non-empty set $S \subseteq \{1, \ldots, n\}$ such that

$$\bigcap_{i \in S} \arg\min_w \ Q_i(w) \neq \emptyset.$$

Consider an arbitrary point w' in the set $\bigcap_{i \in S} \arg\min_w \ Q_i(w)$. Thus, w' is a minimum of the cost functions of all the agents in S, i.e., for every agent $i \in S$, $Q_i(w') \leq Q_i(w)$, $\forall w$. Therefore, $\sum_{i \in S} Q_i(w') \leq \sum_{i \in S} Q_i(w)$, $\forall w$. Equivalently,

$$w' \in \arg\min_w \sum_{i \in S} Q_i(w). \tag{23}$$

Recall that w' is an arbitrary point in the set $\bigcap_{i \in S} \arg\min_w \ Q_i(w)$. Thus, (23) implies that

$$\emptyset \neq \bigcap_{i \in S} \arg\min_w \ Q_i(w) \subseteq \arg\min_w \sum_{i \in S} Q_i(w). \tag{24}$$

We prove the converse of (24) using contradiction below.

Note that, due to (24), $\arg\min_w \sum_{i \in S} Q_i(w) \neq \emptyset$. Suppose that there exists a point $w^\dagger \in \arg\min_w \sum_{i \in S} Q_i(w)$ such that

$$w^\dagger \notin \bigcap_{i \in S} \arg\min_w \ Q_i(w) \neq \emptyset.$$

Thus, there exists $i^\dagger \in S$ such that $w^\dagger \notin \arg\min_w Q_{i^\dagger}(w)$. Therefore, for all $w' \in \bigcap_{i \in S} \arg\min_w \ Q_i(w) \subseteq \arg\min_w \ Q_{i^\dagger}(w)$,

$$Q_{i^\dagger}(w^\dagger) > Q_{i^\dagger}(w'). \tag{25}$$

As $Q_i(w^\dagger) \geq Q_i(w')$ for all $w' \in \bigcap_{i \in S} \arg\min_w \ Q_i(w)$ and $i \in S \setminus \{i^\dagger\}$, (25) implies that for all $w' \in \bigcap_{i \in S} \arg\min_w \ Q_i(w)$,

$$\sum_{i \in S} Q_i(w^\dagger) > \sum_{i \in S} Q_i(w').$$

This contradicts the assumption that $w^\dagger \in \arg\min_w \sum_{i \in S} Q_i(w)$. Therefore, $w^\dagger \in \bigcap_{i \in S} \arg\min_w \ Q_i(w)$ for all $w^\dagger \in \arg\min_w \sum_{i \in S} Q_i(w)$. Equivalently,

$$\arg\min_w \sum_{i \in S} Q_i(w) \subseteq \bigcap_{i \in S} \arg\min_w \ Q_i(w). \tag{26}$$

From (24) and (26),

$$\bigcap_{i \in S} \arg\min_w \ Q_i(w) = \arg\min_w \sum_{i \in S} Q_i(w) \neq \emptyset.$$

As S is an arbitrary non-empty subset of $\{1, \ldots, n\}$, the above proves the Lemma.

B BOUNDS ON THE VALUE OF ρ

In this appendix, we show that if Assumptions 2-4 and $2f$-redundancy property hold true, $\alpha > 0$ and $\eta < \overline{\eta}$ then $\rho \in [0, \ 1)$.

From Cauchy-Schwartz inequality,

$$\langle w - [w]_{\mathcal{W}^*}, \ \nabla Q_i(w) \rangle \leq \| w - [w]_{\mathcal{W}^*} \| \ \| \nabla Q_i(w) \| , \ \forall i \in \mathcal{H}, \ w.$$

Therefore,

$$\sum_{i \in \mathcal{H}} \langle w - [w]_{\mathcal{W}^*}, \ \nabla Q_i(w) \rangle \leq \sum_{i \in \mathcal{H}} \| w - [w]_{\mathcal{W}^*} \| \ \| \nabla Q_i(w) \| . \tag{27}$$

If $2f$*-redundancy* property holds true, then by Definition 1,

$$\mathcal{W}^* = \arg\min_w \sum_{i \in S} Q_i(w), \ \forall S \subseteq \mathcal{H}, \ |S| \geq n - 2f. \tag{28}$$

Convexity of Q_i's, i.e., Assumption 2, trivially implies convexity of the aggregate cost $\sum_{i \in S} Q_i(w)$ for every non-empty set S. Recall

that if $2f$-redundancy property holds true then $n > 2f$. Thus, (28) implies that

$$\sum_{i \in S} \nabla Q_i(w^*) = 0, \ \forall w^* \in \mathcal{W}^*, \ S \subseteq \mathcal{H}, \ |S| \geq n - 2f. \tag{29}$$

From (29), it is easy to obtain that

$$\nabla Q_i(w^*) = 0, \quad \forall w^* \in \mathcal{W}^*, \ \forall i \in \mathcal{H}.$$

Therefore, due to Assumption 3, for all $i \in \mathcal{H}$,

$$\|\nabla Q_i(w)\| \leq \mu \left\| w - w^* \right\|, \quad \forall w \in \mathbb{R}^d, \ w^* \in \mathcal{W}^* \tag{30}$$

As $[w]_{\mathcal{W}^*} \in \mathcal{W}^*$ for all w, substituting from (30) in (27) we obtain that

$$\begin{aligned} \sum_{i \in \mathcal{H}} \langle w - [w]_{\mathcal{W}^*}, \ \nabla Q_i(w) \rangle &\leq \sum_{i \in \mathcal{H}} \mu \left\| w - [w]_{\mathcal{W}^*} \right\|^2 \\ &= |\mathcal{H}| \, \mu \left\| w - [w]_{\mathcal{W}^*} \right\|^2. \end{aligned} \tag{31}$$

Now, due to Assumption 4,

$$\sum_{i \in \mathcal{H}} \langle w - [w]_{\mathcal{W}^*}, \ \nabla Q_i(w) \rangle \geq |\mathcal{H}| \, \lambda \left\| w - [w]_{\mathcal{W}^*} \right\|^2. \tag{32}$$

The inequalities (31) and (32) imply that $\lambda \leq \mu$.

As a result of the above observation, we obtain the following bound on the value of the fault-tolerance margin α (defined in (20)).

$$\alpha = \frac{\lambda}{3\mu} - \frac{f}{n} \leq \frac{\lambda}{3\mu} \leq \frac{1}{3}. \tag{33}$$

Now, from (22), $\rho = 1 - (n\mu)^2 \, \eta \, (\overline{\eta} - \eta)$ where, from (21), $\overline{\eta} = 6\alpha/n\mu$. As $0 < \eta < \overline{\eta}$, there exists $\delta \in (0, \ \overline{\eta})$ such that $\eta = \overline{\eta} - \delta$. From substituting this in the expression for ρ, we obtain that

$$\rho = 1 - (n\mu)^2 \, (\overline{\eta} - \delta) \, \delta. \tag{34}$$

Note that

$$(\overline{\eta} - \delta) \, \delta = \left(\frac{\overline{\eta}}{2} \right)^2 - \left(\delta - \frac{\overline{\eta}}{2} \right)^2.$$

Substituting the above in (34) implies that

$$\rho = 1 - (n\mu)^2 \left[\left(\frac{\overline{\eta}}{2} \right)^2 - \left(\delta - \frac{\overline{\eta}}{2} \right)^2 \right] = (n\mu)^2 \left(\delta - \frac{\overline{\eta}}{2} \right)^2 + 1 - (n\mu)^2 \left(\frac{\overline{\eta}}{2} \right)^2.$$

Therefore, the minimum value of ρ is equal to $1 - (n\mu)^2 \, (\overline{\eta}/2)^2$, which is obtained when $\delta = \overline{\eta}/2$. Also, as $\delta > 0$, $(\delta - (\overline{\eta}/2))^2 < (\overline{\eta}/2)^2$, and therefore, $\rho < 1$. Recall that $\delta \in (0, \ \overline{\eta})$. Thus,

$$1 - (n\mu)^2 \left(\frac{\overline{\eta}}{2} \right)^2 \leq \rho < 1. \tag{35}$$

Substituting $\overline{\eta} = 6\alpha/n\mu$ in (35) implies that $\rho \in [1 - 9\alpha^2, \ 1)$. Therefore, as $\alpha \leq 1/3$ (see (33)), $\rho \in [0, \ 1)$.

C PROOF OF THEOREM 2

Part (i), i.e., $\rho \in [0, \ 1)$, is proved in Appendix B. Now we prove part (ii) of the theorem. Define $\mathfrak{g}^t = \sum_{j=1}^n \widehat{g_j^t}$. Recall, from (17), that

$$w^{t+1} = w^t - \eta \cdot \mathfrak{g}^t, \quad \forall t \geq 0. \tag{36}$$

Let $[w^t]_{\mathcal{W}^*}$ be the Euclidean projection of w^t onto $\mathcal{W}^*$ for all t. From (18), note that $\|w - [w]_{\mathcal{W}^*}\| \leq \|w - w'\|, \ \forall w \in \mathbb{R}^d, \ w' \in \mathcal{W}^*$. Therefore,

$$\left\| w^{t+1} - [w^{t+1}]_{\mathcal{W}^*} \right\| \leq \left\| w^{t+1} - [w^t]_{\mathcal{W}^*} \right\|.$$

Upon substitution from (36) above we obtain that

$$\begin{aligned} \left\| w^{t+1} - [w^{t+1}]_{\mathcal{W}^*} \right\|^2 &\leq \left\| w^t - \eta \, \mathfrak{g}^t - [w^t]_{\mathcal{W}^*} \right\|^2 \\ &= \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2 - 2\eta \left\langle w^t - [w^t]_{\mathcal{W}^*}, \ \mathfrak{g}^t \right\rangle + \eta^2 \left\| \mathfrak{g}^t \right\|^2. \end{aligned} \tag{37}$$

Recall that, for each t,

$$\left\| g_{i_1}^t \right\| \leq \ldots \leq \left\| g_{i_{n-f}}^t \right\| \leq \left\| g_{i_{n-f+1}}^t \right\| \leq \ldots \leq \left\| g_{i_n}^t \right\|$$

and,

$$\widehat{g_j^t} = \begin{cases} \dfrac{\left\| g_{i_{n-f}}^t \right\|}{\left\| g_j^t \right\|} \, g_j^t & , \quad j \in \{i_{n-f+1}, \ldots, \ i_n\} \\ \\ g_j^t & , \quad j \in \{i_1, \ldots, \ i_{n-f}\} \end{cases}$$

Therefore,

$$\left\| \widehat{g_j^t} \right\| \leq \left\| g_{i_{n-f}}^t \right\|, \quad \forall j \in \{1, \ldots, \ n\}, \ t \geq 0 \tag{38}$$

Note that, as there are at most f Byzantine faulty agents, for each t there exists $\sigma_t \in \mathcal{H}$ such that

$$\left\| g_{i_{n-f}}^t \right\| \leq \left\| g_{\sigma_t}^t \right\| \tag{39}$$

As $g_i^t = \nabla Q_i(w^t), \ \forall i \in \mathcal{H}$, substituting from (39) in (38) we obtain that

$$\left\| \widehat{g_j^t} \right\| \leq \left\| \nabla Q_{\sigma_t}(w^t) \right\|, \quad \forall j \in \{1, \ldots, \ n\}, \ t \geq 0 \tag{40}$$

Substituting from (30) in (40) we obtain that

$$\left\| \widehat{g_j^t} \right\| \leq \mu \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|, \quad \forall j \in \{1, \ldots, \ n\}, \ t \geq 0 \tag{41}$$

Recall that $\mathfrak{g}^t = \sum_{j=1}^n \widehat{g_j^t}$. Due to triangle inequality, $\left\| \mathfrak{g}^t \right\| \leq \sum_{j=1}^n \left\| \widehat{g_j^t} \right\|$. This observation, in conjunction with (41), implies that

$$\left\| \mathfrak{g}^t \right\| \leq n\mu \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|. \tag{42}$$

Substituting from (42) in (37) we obtain that

$$\begin{aligned} \left\| w^{t+1} - [w^{t+1}]_{\mathcal{W}^*} \right\|^2 \leq \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2 - 2\eta \left\langle w^t - [w^t]_{\mathcal{W}^*}, \ \mathfrak{g}^t \right\rangle \\ + \eta^2 n^2 \mu^2 \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2. \end{aligned} \tag{43}$$

Now, we define

$$\phi_t = \left\langle w^t - [w^t]_{\mathcal{W}^*}, \ \mathfrak{g}^t \right\rangle. \tag{44}$$

As there are at most f Byzantine faulty agents, for each iteration t there exists a set $\mathcal{H}_1^t \subset \mathcal{H}$ such that

$$\mathcal{H}_1^t \subset \{i_1, \ldots, \ i_{n-f}\}, \ \left| \mathcal{H}_1^t \right| = n - 2f. \tag{45}$$

Therefore,

$$\phi_t = \sum_{j \in \mathcal{H}_1^t} \left\langle w^t - [w^t]_{\mathcal{W}^*}, \ \nabla Q_j(w^t) \right\rangle + \sum_{k \in \{1, \ldots, n\} \setminus \mathcal{H}_1^t} \left\langle w^t - w^*, \ \widehat{g_k^t} \right\rangle.$$

Let $\mathcal{H}_2^t = \mathcal{H} \setminus H_1^t$. Note that $\{1, \ldots, n\} \setminus \mathcal{H}_1^t = \mathcal{H}_2^t \cup \mathcal{B}$. Thus,

$$\phi_t = \sum_{j \in \mathcal{H}_1^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle + \sum_{j \in \mathcal{H}_2^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \widehat{g_j^t} \right\rangle + \sum_{k \in \mathcal{B}} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \widehat{g_k^t} \right\rangle. \quad (46)$$

Let

$$\alpha_j^t = \min \left\{ 1,\, \frac{\left\| g_{i_{n-f}}^t \right\|}{\left\| g_j^t \right\|} \right\}.$$

Recall that $g^t = \nabla Q_j(w^t)$ for all $j \in \mathcal{H}$. Now, upon substituting $\widehat{g_j^t} = \alpha_j^t g_j^t$ for all $j \in \mathcal{H}_2^t$ in (46) we obtain that

$$\phi_t = \sum_{j \in \mathcal{H}_1^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle + \sum_{k \in \mathcal{B}} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \widehat{g_k^t} \right\rangle + \sum_{j \in \mathcal{H}_2^t} \alpha_j^t \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle. \quad (47)$$

Recall that, due to Assumption 2, $Q_j(w)$ is differentiable and convex for all $j \in \mathcal{H}$. This implies that [7]

$$\left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle \geq 0,\ \forall j \in \mathcal{H}. \quad (48)$$

Recall that $\alpha_j^t \geq 0,\ \forall j$. Substituting from (48) in (47) we obtain that

$$\phi_t \geq \sum_{j \in \mathcal{H}_1^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle + \sum_{k \in \mathcal{B}} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \widehat{g_k^t} \right\rangle. \quad (49)$$

Due to Cauchy-Schwartz inequality,

$$\left\langle w^t - [w^t]_{\mathcal{W}^*},\, \widehat{g_k^t} \right\rangle \geq - \left\| w^t - [w^t]_{\mathcal{W}^*} \right\| \left\| \widehat{g_k^t} \right\|,\ \forall k \in \mathcal{B}.$$

Substituting from (41) above we obtain that

$$\left\langle w^t - [w^t]_{\mathcal{W}^*},\, \widehat{g_k^t} \right\rangle \geq -\mu \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2,\ \forall k \in \mathcal{B}. \quad (50)$$

Recall that $|\mathcal{B}| \leq f$. Substituting from (50) in (49) we obtain that

$$\phi_t \geq \sum_{j \in \mathcal{H}_1^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle - f\mu \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2. \quad (51)$$

Now, due to Assumption 4,

$$\sum_{j \in \mathcal{H}} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle \geq |\mathcal{H}|\, \lambda \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2.$$

Recall that $\mathcal{H} = \mathcal{H}_1^t \cup \mathcal{H}_2^t$. The above inequality implies that

$$\sum_{j \in \mathcal{H}_1^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle \geq |\mathcal{H}|\, \lambda \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2 - \sum_{j \in \mathcal{H}_2^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle. \quad (52)$$

Due to Cauchy-Schwartz inequality,

$$\left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle \leq \left\| w^t - [w^t]_{\mathcal{W}^*} \right\| \left\| \nabla Q_j(w^t) \right\|,\ \forall j \in \mathcal{H}.$$

Substituting from (30) above we obtain that

$$\left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle \leq \mu \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2,\ \forall j \in \mathcal{H}. \quad (53)$$

Recall that $\left|\mathcal{H}_1^t\right| = n - 2f$, and $\mathcal{H}_2^t = \mathcal{H} \setminus \mathcal{H}_1^t$. Thus, $\left|\mathcal{H}_2^t\right| = |\mathcal{H}| - (n - 2f)$. This together with (53) implies that

$$\sum_{j \in \mathcal{H}_2^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle \leq (|\mathcal{H}| - (n - 2f))\, \mu \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2. \quad (54)$$

Substituting from (54) in (52) we obtain that

$$\sum_{j \in \mathcal{H}_1^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle \geq (|\mathcal{H}|\, (\lambda - \mu) + (n - 2f)\mu) \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2. \quad (55)$$

In Appendix B, we have shown that $2f$-redundancy and Assumptions 2- 4 imply that $\lambda - \mu \leq 0$. As $|\mathcal{H}| \leq n$, this implies that $|\mathcal{H}|\, (\lambda - \mu) \geq n(\lambda - \mu)$. Substituting this in (55) implies that

$$\sum_{j \in \mathcal{H}_1^t} \left\langle w^t - [w^t]_{\mathcal{W}^*},\, \nabla Q_j(w^t) \right\rangle \geq (n\lambda - 2\mu f) \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2.$$

Upon substituting the above in (51) we obtain that

$$\phi_t \geq (n\lambda - 3\mu f) \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2. \quad (56)$$

Recall the definition of ϕ_t from (44). Thus, substituting from (56) in (43) we obtain that

$$\left\| w^{t+1} - [w^{t+1}]_{\mathcal{W}^*} \right\|^2 \leq \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2 - 2\eta\, (n\lambda - 3\mu f) \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2 + \eta^2 n^2 \mu^2 \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2$$

That is, for all t,

$$\left\| w^{t+1} - [w^{t+1}]_{\mathcal{W}^*} \right\|^2 \leq \left(1 - 2\eta\, (n\lambda - 3\mu f) + \eta^2 n^2 \mu^2\right) \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2. \quad (57)$$

Recall, from (20), that $n\lambda - 3\mu f = 3n\mu\, \alpha$. From substituting this in (57) we obtain that, for all t,

$$\left\| w^{t+1} - [w^{t+1}]_{\mathcal{W}^*} \right\|^2 \leq \left(1 - 6\eta(n\mu)\, \alpha + \eta^2 (n\mu)^2\right) \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2. \quad (58)$$

Note that

$$1 - 6\eta(n\mu)\, \alpha + \eta^2 (n\mu)^2 = 1 - (n\mu)^2 \eta \left(\frac{6\alpha}{n\mu} - \eta \right). \quad (59)$$

Upon substituting from (21), $\overline{\eta} = (6\alpha / n\mu)$, in (59) we obtain that

$$1 - 6\eta(n\mu)\, \alpha + \eta^2 (n\mu)^2 = 1 - (n\mu)^2 \eta\, (\overline{\eta} - \eta) \quad (60)$$

Substituting ρ from (22) in (60) we obtain that $1 - 6\eta(n\mu)\, \alpha + \eta^2 (n\mu)^2 = \rho$. From substituting this in (58) we obtain that

$$\left\| w^{t+1} - [w^{t+1}]_{\mathcal{W}^*} \right\|^2 \leq \rho \left\| w^t - [w^t]_{\mathcal{W}^*} \right\|^2, \quad \forall t.$$

Reasoning by induction, the above proves part (ii) of the theorem.

Note that the algorithm terminates at iteration t if $\left\| g_{i_{n-f}}^t \right\| = 0$. We now show that if $\left\| g_{i_{n-f}}^t \right\| = 0$ then $w^t \in \mathcal{W}^*$. Recall that there exists a subset $\mathcal{H}_1^t$ of $\mathcal{H}$ of size $n-2f$ such that $\mathcal{H}_1^t \subset \{i_1, \ldots, i_{n-f}\}$. If $\left\| g_{i_{n-f}}^t \right\| = 0$ then $\left\| g_j^t \right\| = 0,\ \forall j \in \mathcal{H}_1^t$. This implies that $g_j^t = \nabla Q_j(w^t) = 0,\ \forall j \in \mathcal{H}_1^t$. Therefore, $\sum_{j \in \mathcal{H}_1^t} \nabla Q_j(w^t) = 0$. Due to Assumption 2, this implies that $w^t \in \arg\min_w \sum_{j \in \mathcal{H}_1^t} \nabla Q_j(w)$. Thus, due to $2f$-*redundancy* (see Definition 1), $w^t \in \mathcal{W}^*$.