DIST-CURE: A Robust Distributed Learning Algorithm with Cubic Regularized Newton

Avishek Ghosh, IIT Bombay Raj Kumar Maity, UMass Amherst Arya Mazumdar, UC San Diego

***Abstract*—The problem of saddle-points avoidance for non-convex optimization is quite challenging in large scale distributed learning frameworks. The celebrated cubic-regularized Newton method of Nesterov and Polyak [1] is one of the most elegant algorithms to avoid saddle-points in the standard centralized (non-distributed) setup. In this paper, we analyze the cubic-regularized Newton method in the distributed framework and simultaneously address several practical challenges that naturally arises, such as communication bottleneck and Byzantine attacks. To that end, we propose DISTributed CUbic REgularized Newton's method (`DIST-CURE`), and obtain convergence guarantees under several settings. We emphasize that the issue of saddle-point avoidance becomes more crucial in the presence of Byzantine machines since rogue machines may create *fake local minima* near the saddle-points of the loss function (this is known as the saddle-point attack).**

Being a second order algorithm, the iteration complexity of `DIST-CURE` is much lower than its first order counterparts, and furthermore we can further compress to achieve communication efficiency. To address the challenge of Byzantine resilience, we employ norm based thresholding on the local solutions. We validate the performance of `DIST-CURE` with experiments using standard datasets and several types of Byzantine attacks, and obtain an improvement of 25% with respect to first order methods in total iteration complexity.

Full Paper: Available at: http://tinyurl.com/3axkfy8w

I. Introduction

In real-world machine learning applications such as recommendation systems, image recognition, and conversational AI, it has become crucial to implement learning algorithms in a distributed fashion. In many applications, like Federated Learning (FL) [2], [3], data is stored in user devices such as mobile phones and personal computers. In a standard distributed learning framework, several local machines (aka user devices) perform local computations and communicate to the center machine (a parameter server), and the center machine aggregates and broadcasts the information iteratively.

In such a distributed framework, it is well-known that one of the major challenges is to tackle the behavior of the Byzantine machines [4]. This can happen owing to software or hardware crashes, poor communication link between the local machines and the center machine, stalled computations, and even coordinated or malicious attacks by a third party. In this setup, we assume ([5], [6]) that a subset of local machines behave completely arbitrarily even in a way that depends on the algorithm used and the data on the other machines.

Another critical challenge is the communication cost between the local machines and the center machine. The gains we obtain by parellelization often get bottle-necked by this cost. In case of FL, this cost is directly linked with the (internet) bandwidth of the users and thus resource constrained.

It is well known that in-terms of the number of iterations, second order methods (like Newton and its variants) outperform their competitor; the first order gradient based methods. In this work, we simultaneously handle the Byzantine and communication cost aspects of distributed learning for non-convex functions. In particular, we focus on optimizing a non-convex loss function $f(.)$ [6]–[9]. We have m local machines, out of which α fraction may behave in a Byzantine fashion, where $\alpha < \frac{1}{2}$. Most of the current approaches either work when $f(.)$ is convex, or provide weak guarantees in the non-convex case (for example: zero gradient points, maybe a saddle point).

In order to fit complex machine learning models, one often requires to find local minima of a non-convex loss $f(.)$, instead of just critical points which may include several saddle points. Training deep neural networks and other high-capacity learning architectures [10], [11] are some of the examples where finding local minima is crucial. The stationary points of these problems are in fact saddle points and far away from any local minimum [11], [12], and hence designing efficient algorithm that escapes saddle points is of interest. Moreover, [13], [14] argue that saddle points can lead to highly sub-optimal solutions in many problems of interest. This is amplified in high dimension as shown in [15], and becomes the main bottleneck in training deep neural nets. Furthermore, a line of recent work [14], [16], [17], show that for many non-convex problems, it is sufficient to find a local minimum. In fact, in many problems of interest, all local minima are global minima (e.g., dictionary learning [17], phase retrieval [14], matrix sensing and completion [11], [16], and some of neural nets [12]). Also, in [18], it is argued that for more general neural nets, the local minima are as good as global minima.

The issue of local minima convergence becomes non-trivial in the presence of Byzantine machines. Since we do not assume anything on the behavior of the Byzantine machines, it is certainly conceivable that by appropriately modifying their messages to the center, they can create *fake local minima* that are close to the saddle point of the loss function $f(.)$, and these are far away from the true local minima. This is popularly known as the *saddle-point attack* (see [19]), and it can arbitrarily destroy the performance of any non-robust learning algorithm. Hence, our goal is to design an algorithm that escapes saddle points of $f(.)$ in an efficient manner as well as resists the saddle-point attack simultaneously. The complexity of such an algorithm emerges from the the interplay between non-convexity of the loss function and the behavior of the Byzantine machines.

The problem of saddle point avoidance in the context of non-convex optimization has received considerable attention in the past few years. In [20], a (first order) gradient descent based approach is proposed. A few papers [21], [22] following the above use various modifications to obtain saddle point avoidance guarantees. A Byzantine robust first order saddle point avoidance algorithm is proposed

in [19], and probably is the closest to this work. In [19], the authors propose a repeated check-and-escape type of first order gradient descent based algorithm. First of all, being a first order algorithm, the convergence rate is quite slow (the rate for gradient decay is $1/\sqrt{T}$, where T is the number of iterations). Moreover, implementation-wise, the algorithm presented in [19] is computation heavy, and takes potentially many iterations between the center and the local machines (as we check in Section VI and Appendix). Hence, this algorithm is not efficient in terms of the communication cost.

Our Approach: We consider a variant of the famous cubic-regularized Newton algorithm of Nesterov and Polyak [1], which efficiently escapes the saddle points of a non-convex function by appropriately choosing a regularization and thus pushing the Hessian towards a positive semi-definite matrix. The primary motivation behind this choice is the faster convergence rate compared to first order methods. Indeed, the rate of gradient decay is $\frac{1}{T^{2/3}}$.

We apply the cubic regularized Newton algorithm in the distributed setup and address several practical issues like communication efficiency and robustness. We propose a novel algorithm, namely DISTributed CUbic REgularized Newton (`DIST-CURE`). In this scheme, the center machine asks the local machines to solve an auxiliary problem and return the result. The center machine aggregates the solution of the local machines and takes a descent step. Note that, unlike gradient aggregation, the aggregation of the solutions of the local optimization problems is a highly non-linear operation. Furthermore, the local problems lack any closed form expression, making this extension to be quite non-trivial and technically challenging.

In addition to the above, `DIST-CURE` simultaneously use (i) a δ-approximate compressor (defined shortly) to compress the message send from local machines to center to gain further communication reduction and (ii) a simple norm-based thresholding on the (compressed) solution sent by the local machines to defend adversarial (Byzantine) attacks. Norm based thresholding is also a standard trick for Byzantine resilience as featured in [23], [24]. However, since the local optimization problem lacks a closed form solution, using norm-based trimming is also technical challenging in this case. We now list our contributions.

A. Our Contributions

1) Technical Novelty: We propose a novel distributed, communication efficient and robust cubic regularized Newton algorithm, namely `DIST-CURE` that escapes saddle point efficiently. We prove that the algorithm convergence at a rate of $\frac{1}{T^{2/3}}$, which is faster than the first order methods (which converge at $1/\sqrt{T}$ rate, see [19]). Also, the convergence rate matches to that of the centralized scheme of [1] and hence, we do not lose in terms of convergence rate while making the algorithm distributed. In `DIST-CURE`, the center machine aggregates the solution of the local machines. We emphasize that, unlike gradient aggregation, the aggregation of the solutions of the local optimization problems is a highly non-linear operation, as evidenced by even a much simpler second order optimization algorithm like GIANT ([25]). Hence, it is quite non-trivial to extend the centralized cubic regularized algorithm to a distributed one. The solution to the cubic regularization even lacks a closed form solution unlike the second order Hessian based update or the first order gradient based update. The analysis of `DIST-CURE` is carried out by leveraging the first order and second order stationary conditions of the auxiliary function solved in each local machines.

In [19], a *perturbed gradient based algorithm* to escape the saddle point in non-convex optimization in the presence of Byzantine local machines is provided. The Byzantine resilience is achieved using techniques such as trimmed mean, median and collaborative filtering. These methods require additional assumptions (coordinate of the gradient being sub-exponential etc.) for the purpose of analysis. In this work, we do not require such assumptions. Moreover, we perform a simple *norm based thresholding* that provides robustness. Also the perturbed gradient descent (PGD) actually requires multiple rounds of communications between the central machine and the local machines whenever the norm of the gradient is small as this is an indication of either a local minima or a saddle point. In contrast to that, our method does not require any additional communication for *escaping* the saddle points. Our method provides such ability by virtue of cubic regularization.

2) Experiments: In Section VI and in Appendix, we verify our theoretical findings via experiments. We first show that `DIST-CURE` indeed avoids saddle points via a simple example. Moreover, we use benchmark LIBSVM ([26]) datasets for logistic regression and non-convex robust regression and show convergence results for both non-Byzantine and several different Byzantine attacks. Specifically, we characterize the total iteration complexity (defined in Section VI) of our algorithm, and compare it with several baselines. We observe that the algorithm of [19] requires 25% more total iterations than ours.

B. Preliminaries:

We denote the norm $\|\cdot\|$ as ℓ_2 norm or spectral norm when the argument is a vector or a matrix respectively. A point $\mathbf{x}$ is said to satisfy the ϵ-second order stationary condition of $f(.)$ if,

$$\|\nabla f(\mathbf{x})\| \leq \epsilon, \quad \lambda_{\min}(\nabla^2 f(\mathbf{x})) \geq -\sqrt{\epsilon},$$

where $\nabla f(\mathbf{x})$ denotes the gradient of the function and $\lambda_{\min}(\nabla^2 f(\mathbf{x}))$ denotes the minimum eigenvalue of the Hessian of the function. Hence, under the assumption (which is standard in the literature, see [19], [20]) that all saddle points are strict (i.e., $\lambda_{\min}(\nabla^2 f(\mathbf{x}_s)) < 0$ for any saddle point $\mathbf{x}_s$), all second order stationary points (with $\epsilon = 0$) are local minima, and hence converging to a stationary point is equivalent to converging to a local minima.

II. Problem Formulation

We minimize a loss function of the form: $f(\mathbf{x}) = \frac{1}{m}\sum_{i=1}^{m} f_i(\mathbf{x})$, where the function $f : \mathbb{R}^d \to \mathbb{R}$ is twice differentiable and non-convex. We consider a standard learning framework with m local machines and one center machine where the local machines can only communicate to the center machine. Each local machine is associated with a local loss function f_i. We assume that the data distribution is non-iid across local machines. In addition to this, we also consider the case where α fraction of the local machines are Byzantine for some $\alpha < \frac{1}{2}$. The Byzantine machines can send arbitrary updates to the central machine which can disrupt the learning. Furthermore, the Byzantine machines can collude with each other, create *fake local minima* or attack maliciously by gaining information about the learning algorithm and other local machines.

Furthermore, we use compression for communication efficiency and consider a generic class of compressors from [23], [27]:

Definition 1 (δ-Approximate Compressor): An operator $Q(.):\mathbb{R}^d \to \mathbb{R}^d$ is defined as δ approximate compressor on a set $\mathcal{S} \subseteq \mathbb{R}^d$ if, $\forall x \in \mathcal{S}$, $\|Q(x)-x\|^2 \leq (1-\delta)\|x\|^2$, where $\delta \in (0,1]$ is the compression factor.

For randomized operator, the above holds on expectation. In this paper, for clarity, we consider the deterministic form (as in Definition 1). However, the results can be easily extended for randomized $Q(.)$. Notice that $\delta=1$ implies $Q(x)=x$ (no compression).

III. Related Work

A. Saddle Point avoidance algorithms

In the recent years, there are handful first order algorithms [28]–[30] that focus on the escaping saddle points and convergence to local minima. The critical algorithmic aspect is running gradient based algorithm and adding perturbation to the iterates when the gradient is small. ByzantinePGD [19], PGD [20], Neon+GD [21], Neon2+GD [22] are examples of such algorithms. The work of Nesterov and Polyak [1] first proposes the cubic regularized second order Newton method and provides analysis for the second order stationary condition. An algorithm called Adaptive Regularization with Cubics (ARC) was developed by [31], [32] where cubic regularized Newton method with access to inexact Hessian was studied. Cubic regularization with both the gradient and Hessian being inexact was studied in [33]. In [34], a cubic regularized Newton with sub-sampled Hessian and gradient was proposed and analyzed. Momentum based cubic regularized algorithm was studied in [35]. A variance reduced cubic regularized algorithm was proposed in [25], [36]. In terms of solving the cubic sub-problem, [37] proposes a gradient based algorithm and [38] provides a Hessian-vector product technique. [39] employs a *a negative curvature finding algorithm* based on gradient descent and accelerated gradient descent method to improve the PGD algorithm [20]. [40] proposes perturbed compressed SGD with error feedback.

B. Compression and Robustness

In the recent years, several gradient quantization or sparsification schemes have been studied in [41]–[44]. In [27], the authors introduced the idea of δ-approximate compressor. In [45], the authors use δ-approximate compressor to sparsify the second order update. In the distributed learning context, [8] proposes one shot median based robust learning. A median of mean based algorithm was proposed in [9] where the local machines are grouped in batches and the Byzantine resilience is achieved by computing the median of the grouped machines. Later [5] proposes co-ordinate wise median, trimmed mean and iterative filtering based approaches. Communication-efficient and Byzantine robust algorithms were developed in [23], [46]. A norm based thresholding approach for Byzantine resilience for distributed Newton algorithm was also developed [24]. All these works provide only first order convergence guarantee (small gradient). The work [19] is the only one that provides second order guarantee (Hessian positive semi-definite) under Byzantine attack.

IV. Algorithm–DIST-CURE

We describe a communication efficient and Byzantine robust distributed cubic Newton algorithm, namely DIST-CURE that can avoid saddle point and thus converge to a local minima for non-convex loss function. Starting with initialization $\mathbf{x}_0$, the center machine broadcasts the parameter to the local machines. At k-th iteration, the i-th local machine solves a cubic-regularized auxiliary loss function based on its local data:

$$\mathbf{s}_{i,k+1} = \arg\min_{\mathbf{s}} \mathbf{g}_{i,k}^T \mathbf{s} + \frac{\gamma}{2}\mathbf{s}^T \mathbf{H}_{i,k}\mathbf{s} + \frac{M}{6}\gamma^2\|\mathbf{s}\|^3, \qquad (1)$$

where $M>0, \gamma>0$ are parameter choose suitably and $\mathbf{g}_{i,t}, \mathbf{H}_{i,t}$ are the gradient and Hessian of the local loss function f_i computed on data (S_i) stored in the local machine.

$$\mathbf{g}_{i,k} = \nabla f_i(x_k) = \frac{1}{|S_i|}\sum_{z_i \in S_i} \nabla f_i(x_k, z_i)$$

$$\mathbf{H}_{i,k} = \nabla^2 f_i(x_k) = \frac{1}{|S_i|}\sum_{z_i \in S_i} \nabla^2 f_i(x_k, z_i).$$

After solving the problem described in (1), each local machine applies compression operator Q as defined in Definition 1 on update $\mathbf{s}_{i,k+1}$. The application of the compression on the update is to minimize the communication cost.

Moreover, we also consider that $\alpha(<\frac{1}{2})$ fraction of the local machines are Byzantine in nature. We denote the set of Byzantine local machines by $\mathcal{B}$ and the set of the rest of the good machines as $\mathcal{M}$. In each iteration, the good machines send the compressed update of solution of the sub-cubic problem described in equation (1) and the Byzantine machines can send any arbitrary values or intentionally disrupt the learning algorithm with malicious updates. Lack of any robust measure towards these type of intentional and unintentional attacks can be catastrophic to the learning procedure as the learning algorithm can get stuck in such sub-optimal point. To tackle such Byzantine local machines, we employ a simple process called *norm based thresholding*.

After receiving all the updates from the local machines, the central machine outputs a set $\mathcal{U}$ which consists of the indexes of the local machines with smallest norm. DIST-CURE chooses the size of the set $\mathcal{U}$ to be $(1-\beta)m$. Hence, we 'trim' β fraction of

Algorithm 1 DIST-CURE

1: **Input:** Step size η_k, parameter $0 \leq \alpha \leq \beta, \gamma > 0, M > 0$ and δ-approximate compressor Q.
2: **Initialize:** Initial iterate $\mathbf{x}_0 \in \mathbb{R}^d$
3: **for** $k=0,1,...,T-1$ **do**
4: Central machine: broadcasts $\mathbf{x}_k$
for $i \in [m]$ **do in parallel**
5: i-th local machine:
Non-Byzantine: Compute local gradient $\mathbf{g}_{i,k}$ and Hessian $\mathbf{H}_{i,k}$; locally solve the problem equation (1). Use the compressor Q and send $Q(\mathbf{s}_{i,k+1})$ to the center,
Byzantine: Generate $\star$ (arbitrary), and send it to the center
end for
6: Center Machine:
(i) Sort local machines in a non decreasing order according to norm of updates $\{Q(\mathbf{s}_{i,k+1})\}_{i=1}^m$
(ii) Return indices of first $1-\beta$ fraction of machines, $\mathcal{U}_k$,
(iii) Update: $\mathbf{x}_{k+1} = \mathbf{x}_k + \eta_k \frac{1}{|\mathcal{U}_k|}\sum_{i \in \mathcal{U}_k} Q(\mathbf{s}_{i,k+1})$
7: **end for**

the local machine so that we can control the iterated update by not letting the local machines with large norm participate and diverge the learning process. We denote the set of trimmed machine as $\mathcal{T}$. We choose $\beta > \alpha$ so that at least one of the good machines gets trimmed. In this way, the norm of the all the updates in the set $\mathcal{U}$ is bounded by at least the largest norm of the good machines.

Remark 1 (Exact solution only for theory): We emphasize that the exact solution of the sub-problem (which the original work of [1] also needed) is only required for theoretical tractability. In practice, it is not possible to obtain such solution. For that reason, in experiments (Section VI) we run the gradient based first order algorithm of [33] to achieve this. We expand on this in the Appendix.

Remark 2: We introduce the parameter γ in the cubic regularized sub-problem, which was absent in the original formulation of [1]. The parameter γ emphasizes the effect of the second and third order terms in the sub-problem. The choice of γ plays an important role in our analysis in handling the updates from different local machines.

V. Theoretical Guarantees

We have the following standard assumptions:

Assumption 1: The non-convex loss $f(.)$ is twice continuously-differentiable and bounded below, i.e., $f^* = \inf_{\mathbf{x}\in\mathbb{R}^d} f(x) > -\infty$.

Assumption 2: The loss $f(.)$ is *L-Lipschitz continuous* ($\forall \mathbf{x}, \mathbf{y},\ |f(\mathbf{x}) - f(\mathbf{y})| \leq L\|\mathbf{x}-\mathbf{y}\|$), has L_1-*Lipschitz gradients* ($\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L_1\|\mathbf{x}-\mathbf{y}\|$) and L_2-*Lipschitz Hessian* ($\left\|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{y})\right\| \leq L_2\|\mathbf{x}-\mathbf{y}\|$).

The above assumption states that the loss and the gradient and Hessian of the loss do not drastically change in the local neighborhood. These assumptions are standard in the analysis of the saddle point escape for cubic regularization (see [33], [34], [37]) and have also appeared in the original work by Nesterov and Polyak ([1]).

We assume the data distribution across local machines to be non-iid. However, we assume that the local gradient and Hessian computed at local machines (using local data) satisfies the following gradient and Hessian dissimilarity conditions. Note that these conditions are only applicable for non-Byzantine machines only. Byzantine machines do not adhere to any assumptions.

Definition 2 (Heterogeneity): In the FL setup, the gradient and Hessian heterogeneity are defined as the following: $\epsilon_g > 0$ and $\epsilon_H > 0$, we have, for all k,i,

$$\|\nabla f(\mathbf{x}_k) - \mathbf{g}_{i,k}\| \leq \epsilon_g \quad \|\nabla^2 f(\mathbf{x}_k) - \mathbf{H}_{i,k}\| \leq \epsilon_H.$$

We emphasize that bounded gradient and hessian dissimilarity are quite common in distributed learning (specially in Federated Learning), and are one major way to characterize the *degree of heterogeneity*. For example, see [47]–[53] and the references therein. These papers use this *bounded heterogeneity* condition to analyze convergence results. Although the above condition is written in terms of all the *good* machines, with a slight modification, we can extend our analysis to the case where bounded heterogeneity is required *on an average*, that is not for all i but on average gradient and Hessian.

ϵ_g and ϵ_H in special cases: The gradient and Hessian bound have been studied under more relaxed condition. In [33]–[35], the authors consider gradient and Hessian with sub-sampled data being drawn uniformly randomly from the data set. Due to the data being drawn in iid manner, both the bound (ϵ_g, ϵ_H) parameters value diminish at the rate $\propto 1/\sqrt{|S|}$ where $|S|$ is the size of the data sample in each local machine. In [24], the authors analyze the deviation in case of *data partitioning* where each local machine sample data uniformly *without replacement* from a given data set.

Remark 3 (Two rounds of communication $\epsilon_g = 0$, $\epsilon_H = 0$): We can make $\epsilon_g = 0$ one more round of communication in each iteration. In the first iteration, all the local machines compute the gradient based on the stored data and send it to the center machine. The center machine averages them and then broadcast the global gradient $\nabla f(\mathbf{x}_k) = \frac{1}{m}\sum_{i=1}^{m} \mathbf{g}_{i,k}$ at iteration k. In this manner, the local machines solve the sub-problem (1) with the actual gradient. Note that [25] does this exactly to avoid ϵ_g. Similarly, with more communication cost, we can make $\epsilon_H = 0$ by allowing local machines to send local Hessians and the center to aggregate and broadcast the aggregated Hessian. However, in standard FL, one typically avoids this additional round of communication and deal with gradient and Hessian dissimilarities.

We now present the main results of the paper. We first present the convergence guarantees of `DIST-CURE` with simultaneous compression and Byzantine resilience. Subsequently, we relax the restrictions on communication efficiency and robustness.

Recall that `DIST-CURE` uses δ approximate compression for communication efficiency and norm based thresholding for Byzantine resilience. In the theoretical analysis, to avoid clutter and for the clarity of exposition, we substitute $\delta = 1$. However, as seen in [23], [27], the analysis can be seamlessly extended to the setting where $\delta \in (0,1)$. We have the following result.

Theorem 1 (Convergence of `DIST-CURE`): Suppose $0 \leq \alpha < \beta \leq \frac{1}{2}$ and $m \geq 2$. Furthermore, we choose the problem parameters, $M = \mathcal{O}(m(1-\beta))$, and $\eta_k = \frac{c}{Tm^\nu}$;$\gamma = \frac{c}{Tm^\nu}$, for some constant $c > 0$,$\nu > 3$. Then, after T iterations of `DIST-CURE` (Algorithm 1), the sequence $\{\mathbf{x}_i\}_{i=1}^{T}$ generated contains a point $\tilde{x}$ such that

$$\|\nabla f(\tilde{x})\| \leq \frac{\chi_1}{T^{2/3}} + \epsilon_g + \chi_G$$
$$\lambda_{\min}\left(\nabla^2 f(\tilde{x})\right) \geq -\frac{\chi_2}{T^{\frac{1}{3}}} - \epsilon_H - \chi_H,$$

where, $\chi_1 = \mathcal{O}(\frac{(1-\alpha)}{(1-\beta)} + m(1-\beta)), \chi_2 = \mathcal{O}(\frac{(1-\alpha)}{(1-\beta)} + m(1-\beta))$

$$\chi_G = \mathcal{O}([\frac{(1-\alpha)}{(1-\beta)} + m(1-\beta)](\frac{1}{m})^{(\frac{2\nu}{3})} + \frac{\alpha}{(1-\beta)m^{2\nu}} + \frac{\alpha}{(1-\beta)m^{\nu}})$$
$$\chi_H = \mathcal{O}([m(1-\beta) + \frac{(1-\alpha)}{(1-\beta)})](\frac{1}{m})^{\frac{\nu}{3}} + \frac{\alpha}{(1-\beta)m^{\nu}}).$$

Corollary 1.1 (Recovering the results of [1]): Suppose $\alpha = 0, \beta = 0$. Moreover, we choose $m = 1$ (centralized) and hence $\epsilon_G = \epsilon_H = 0$. Moreover, as in [1], we choose $\eta = \gamma = 1$. With the above-mentioned choices of problem parameters, we show in Appendix E that that $\chi_G = \chi_H = 0$. Furthermore, we get

$$\|\nabla f(\tilde{x})\| \leq \mathcal{O}(\frac{1}{T^{2/3}}) \quad \lambda_{\min}\left(\nabla^2 f(\tilde{x})\right) \geq -\mathcal{O}(\frac{1}{T^{1/3}}),$$

which matches [1].

The proof of Theorem 1 and Corollary 1.1 is in the Appendix.

Discussion: Note that both the gradient and the minimum eigenvalue of the Hessian in the Theorem 1 have three terms. The first term decreases with the number iterations T. The rate of

decay for gradient and the minimum eigenvalue of the Hessian are $O(1/T^{\frac{2}{3}})$ and $O(1/T^{\frac{1}{3}})$, respectively. We point out that both of these rates match with that of the centralized version of the cubic regularized Newton as shown in [1]. The quantities, χ_1 and χ_2 associated with these terms are independent of T and depends on problem parameters like α,β and m, as shown.

The second term of the gradient bound and the minimum eigenvalue of the Hessian depends on ϵ_g and ϵ_H. This term appears owing to the *non-iid* nature of data in the local machines. Note that the appearance of such terms (depicting the *degree of non-iid ness*) is quite common in distributed optimization literature (ex, see [47], [51]–[53]). Note that in the centralized setup of [1], this aspect of heterogeneity was absent and hence these terms were absent. Furthermore, as mentioned above, in the special cases, both the terms ϵ_g and ϵ_H decrease at the rate of $1/\sqrt{|S|}$, where $|S|$ is the number of data in each of the local machines.

The third term in the expression is an error floor that decays with the number of machines, m, and can be made arbitrarily small by choosing appropriate step-size. Note that as shown in Corollary 1.1, this term vanishes when $m=1$. This term originates from several sources. First, note that the center machine simply aggregates the solution of the local machines to obtain the next update. Unlike gradient aggregation (in first order methods), this simple averaging of local solutions yields a different solution from the global one, and hence one incurs a bias by this simple averaging strategy. This is the cost of going from centralized to a distributed setup, and this is incurred even in the absence of compression and Byzantine resilience. Second, we employ norm based thresholding, and remove the contribution of β fraction of the local machines. This naturally creates an error floor.

Remark 4: Since our algorithm is second order in nature, it requires less number of iterations compared to the first order gradient based algorithms. Our algorithm achieves a superior rate of $O(1/T^{\frac{2}{3}})$ compared to the gradient based approach of rate $O(1/\sqrt{T})$. Our algorithm dominates ByzantinePGD [19] in terms of convergence, communication rounds and simplicity.

A. Special case of Theorem 1

Here, we choose the non-Byzantine setup with $\alpha = \beta = 0$ in addition to the uncompressed update. This is just the distributed variant of the cubic regularized Newton method of [1].

Corollary 1.2 (Non Byzantine and no compression): Suppose we choose $M = \mathcal{O}(m)$, $\eta = \gamma = c/Tm^{\nu}$ for some $c > 0, \nu > 3$. Then, after T iterations of DIST-CURE for uncompressed update ($\delta = 1$), the sequence $\{\mathbf{x}_i\}_{i=1}^{T}$ generated contains a point $\tilde{x}$ such that

$$\|\nabla f(\tilde{x})\| \leq \frac{\chi_1}{T^{2/3}} + \epsilon_g + \chi_G$$
$$\lambda_{\min}\big(\nabla^2 f(\tilde{x})\big) \geq -\frac{\chi_2}{T^{\frac{1}{3}}} - \epsilon_H - \chi_H,$$

where, $\chi_1 = \chi_2 = \mathcal{O}(m)$ and

$$\chi_G = \mathcal{O}(\frac{1}{m^{\frac{2\nu}{3}-1}}), \chi_H = \mathcal{O}(\frac{1}{m^{\frac{\nu}{3}-1}}).$$

Note that the term χ_1,χ_2,χ_G and χ_H have reduced, thus improving the performance. As $\nu > 3$, the parameter χ_G,χ_H are decreasing with the number of local machines. Note that even in the simple distributed variant, the extra error terms (second and third terms) are present. As explained earlier, these are owing to the non-iid nature of data distribution and the simple (biased) aggregation of local solutions at the center respectively.

B. Solution of the cubic sub-problem

The cubic regularized sub-problem (1) needs to be solved to update the parameter. As this particular problem does not have a closed form solution, a solver is usually employed which yields a satisfactory solution. In previous works, different types of solvers have been used. [31], [32] solve the sub-problem using Lanczos based method in Krylov subspace. In [38], the authors propose a solver based on Hessian-vector product and binary search. Gradient descent based solver is proposed in [33], [37].

Previous works, [25], [35], [36], consider the exact solution of the cubic sub-problem for theoretical analysis. Recently, inexact solutions to the sub-problem is also proposed in the centralized (non-distributed) framework. For instance, [34] analyzes the cubic model with sub-sampled Hessian with approximate model minimization technique developed in [31]. Moreover, [33] shows improved analysis with gradient based minimization which is a variant studied in [37]. Both exact and inexact solutions to the sub-problem yields similar theoretical guarantees.

In our framework, each local machine is tasked with solving the sub-problem. For the purpose of theoretical convergence analysis, we consider that local machines obtain the exact solution in each round. However, in experiments (Section VI), we apply the gradient based solver of [33] to solve the sub-problem. Here, we let each local machines run the gradient based solver for 10 iterations and send the update to the center machine in each iteration.

VI. Experimental results

We defer the experimental section in Appendix owing to space limitation, but provide the gist here. We first validate the saddle point avaidance capability of DIST-CURE via a simple example. Then we use standard benchmark LIBSVM ([26]) datasets for logistic regression and non-convex robust regression examples and show convergence results for both non-Byzantine and several different Byzantine attacks. We characterize the total iteration complexity (defined in Section VI) of DIST-CURE, and compare it with several baselines. We observe that DIST-CURE beats its competitor, [19] (which requires 25% more total iterations).

VII. Conclusion

We propose, analyze and experimentally validate DIST-CURE, that uses cubic regularized Newton [1] for saddle point avoidance and norm based thresholding for robustness. We compare the performance of DIST-CURE with existing state of the art algorithms. One immediate future direction is to theoretically understand DIST-CURE where the local machines *approximately* solve the local sub-problem. This is indeed non-trivial as seen in [33]. Another interesting direction is to analyze trust region based methods. We keep this as future endeavors.

VIII. Acknowledgment

The work is supported in part by NSF awards 2133484, 2112665, and 2217058.

REFERENCES

[1] Y. Nesterov and B. T. Polyak, "Cubic regularization of newton method and its global performance," *Mathematical Programming*, vol. 108, no. 1, pp. 177–205, 2006.

[2] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv:1610.02527*, 2016.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *20th AISTATS*, 2017, pp. 1273–1282.

[4] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, Jul. 1982.

[5] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *35th ICML*, 2018.

[6] P. Blanchard, E. M. E. Mhamdi, R. Guerraoui, and J. Stainer, "Byzantine-tolerant machine learning," *arXiv:1703.02757*, 2017.

[7] D. Alistarh, Z. Allen-Zhu, and J. Li, "Byzantine stochastic gradient descent," in *NeurIPS*, vol. 31, 2018, pp. 4613–4623.

[8] J. Feng, H. Xu, and S. Mannor, "Distributed robust learning," *arXiv:1409.5937*, 2014.

[9] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 2, p. 44, 2017.

[10] D. Soudry and Y. Carmon, "No bad local minima: Data independent training error guarantees for multilayer neural networks," 2016.

[11] R. Ge, C. Jin, and Y. Zheng, "No spurious local minima in nonconvex low rank problems: A unified geometric analysis," in *34th ICML*, vol. 70. PMLR, 2017, pp. 1233–1242.

[12] K. Kenji, "Deep learning without poor local minima," in *NeurIPS*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016, pp. 586–594.

[13] P. Jain, C. Jin, S. M. Kakade, and P. Netrapalli, "Global convergence of non-convex gradient descent for computing matrix squareroot," 2017.

[14] J. Sun, Q. Qu, and J. Wright, "A geometric analysis of phase retrieval," *CoRR*, vol. abs/1602.06664, 2016.

[15] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in *NeurIPS*, vol. 27, 2014, pp. 2933–2941.

[16] S. Bhojanapalli, B. Neyshabur, and N. Srebro, "Global optimality of local search for low rank matrix recovery," 2016.

[17] J. Sun, Q. Qu, and J. Wright, "Complete dictionary recovery over the sphere i: Overview and the geometric picture," *IEEE Transactions on Information Theory*, vol. 63, no. 2, p. 853–884, Feb 2017.

[18] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks," 2015.

[19] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Defending against saddle point attack in byzantine-robust distributed learning," in *ICML*, 2019.

[20] C. Jin, R. Ge, P. Netrapalli, S. M. Kakade, and M. I. Jordan, "How to escape saddle points efficiently," in *ICML*. PMLR, 2017, pp. 1724–1732.

[21] Y. Xu, R. Jin, and T. Yang, "First-order stochastic algorithms for escaping from saddle points in almost linear time," *arXiv:1711.01944*, 2017.

[22] Z. Allen-Zhu and Y. Li, "Neon2: Finding local minima via first-order oracles," *arXiv:1711.06673*, 2017.

[23] A. Ghosh, R. K. Maity, S. Kadhe, A. Mazumdar, and K. Ramchandran, "Communication-efficient and byzantine-robust distributed learning with error feedback," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 3, pp. 942–953, 2021.

[24] A. Ghosh, R. K. Maity, and A. Mazumdar, "Distributed newton can communicate less and resist byzantine workers," in *NeurIPS December 6-12, 2020, virtual*, 2020.

[25] Z. Wang, Y. Zhou, Y. Liang, and G. Lan, "Stochastic variance-reduced cubic regularization for nonconvex optimization," in *The 22nd AISTATS*. PMLR, 2019, pp. 2731–2740.

[26] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[27] S. P. Karimireddy, Q. Rebjock, S. Stich, and M. Jaggi, "Error feedback fixes signsgd and other gradient compression schemes," in *ICML*. PMLR, 2019, pp. 3252–3261.

[28] J. D. Lee, M. Simchowitz, M. I. Jordan, and B. Recht, "Gradient descent converges to minimizers," *arXiv:1602.04915*, 2016.

[29] J. D. Lee, I. Panageas, G. Piliouras, M. Simchowitz, M. I. Jordan, and B. Recht, "First-order methods almost always avoid saddle points," *arXiv:1710.07406*, 2017.

[30] S. S. Du, C. Jin, J. D. Lee, M. I. Jordan, B. Poczos, and A. Singh, "Gradient descent can take exponential time to escape saddle points," *arXiv:1705.10412*, 2017.

[31] C. Cartis, N. I. Gould, and P. L. Toint, "Adaptive cubic regularisation methods for unconstrained optimization. part i: motivation, convergence and numerical results," *Mathematical Programming*, vol. 127, no. 2, pp. 245–295, 2011.

[32] ——, "Adaptive cubic regularisation methods for unconstrained optimization. part ii: worst-case function-and derivative-evaluation complexity," *Mathematical programming*, vol. 130, no. 2, pp. 295–319, 2011.

[33] N. Tripuraneni, M. Stern, C. Jin, J. Regier, and M. Jordan, "Stochastic cubic regularization for fast nonconvex optimization," in *NeurIPS*, 2018, pp. 2899–2908.

[34] J. M. Kohler and A. Lucchi, "Sub-sampled cubic regularization for non-convex optimization," in *ICML*. PMLR, 2017, pp. 1895–1904.

[35] Z. Wang, Y. Zhou, Y. Liang, and G. Lan, "Cubic regularization with momentum for nonconvex optimization," in *Uncertainty in Artificial Intelligence*. PMLR, 2020, pp. 313–322.

[36] D. Zhou, P. Xu, and Q. Gu, "Stochastic variance-reduced cubic regularized newton methods," in *ICML*. PMLR, 2018, pp. 5990–5999.

[37] Y. Carmon and J. C. Duchi, "Gradient descent efficiently finds the cubic-regularized non-convex newton step," *arXiv:1612.00547*, 2016.

[38] N. Agarwal, Z. Allen-Zhu, B. Bullins, E. Hazan, and T. Ma, "Finding approximate local minima faster than gradient descent," in *49th Annual ACM SIGACT Symposium on Theory of Computing*, 2017, pp. 1195–1199.

[39] C. Zhang and T. Li, "Escape saddle points by a simple gradient-descent based algorithm," *In NeurIPS*, 2021.

[40] D. Avdiukhin and G. Yaroslavtsev, "Escaping saddle points with compressed sgd," *In NeurIPS*, 2021.

[41] V. Gandikota, D. Kane, R. K. Maity, and A. Mazumdar, "vqsgd: Vector quantized stochastic gradient descent," in *AISTATS*. PMLR, 2021, pp. 2197–2205.

[42] D. Alistarh, T. Hoefler, M. Johansson, N. Konstantinov, S. Khirirat, and C. Renggli, "The convergence of sparsified gradient methods," in *NeurIPS*, 2018, pp. 5973–5983.

[43] H. Wang, S. Sievert, S. Liu, Z. Charles, D. Papailiopoulos, and S. Wright, "Atomo: Communication-efficient learning via atomic sparsification," in *NeurIPS*, 2018, pp. 9850–9861.

[44] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *NeurIPS*, 2017, pp. 1709–1720.

[45] A. Ghosh, R. K. Maity, A. Mazumdar, and K. Ramchandran, "Communication efficient distributed approximate newton method," in *ISIT*. IEEE, 2020, pp. 2539–2544.

[46] J. Bernstein, J. Zhao, K. Azizzadenesheli, and A. Anandkumar, "signsgd with majority vote is communication efficient and byzantine fault tolerant," *arXiv:1810.05291*, 2018.

[47] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv:1806.00582*, 2018.

[48] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith, "On the convergence of federated optimization in heterogeneous networks," *arXiv:1812.06127*, vol. 3, 2018.

[49] L. Li, W. Xu, T. Chen, G. B. Giannakis, and Q. Ling, "Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets," *arXiv:1811.03761*, 2018.

[50] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, 2019.

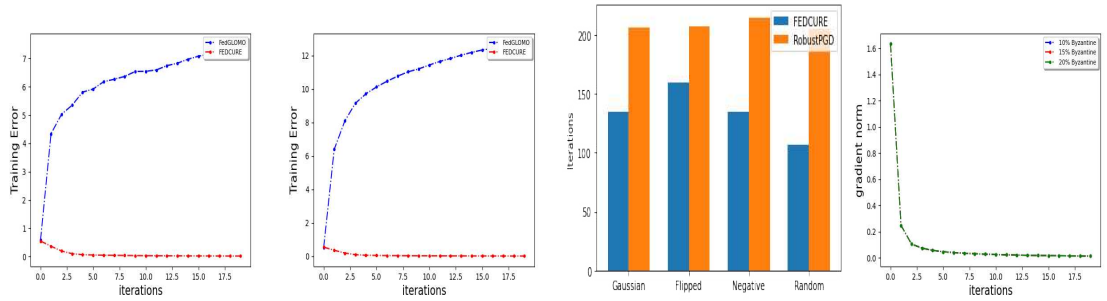
[51] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *ICML*. PMLR, 2019, pp. 4615–4625.

[52] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *ICML*. PMLR, 2020, pp. 5132–5143.

[53] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," *arXiv:2002.07948*, 2020.

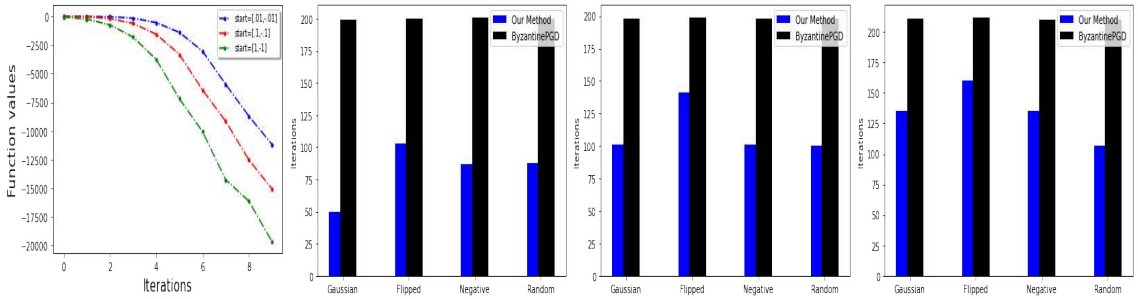
[54] C. Jin, P. Netrapalli, R. Ge, S. M. Kakade, and M. I. Jordan, "On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points," *Journal of the ACM (JACM)*, vol. 68, no. 2, pp. 1–29, 2021.

[55] R. Das, A. Acharya, A. Hashemi, S. Sanghavi, I. S. Dhillon, and U. Topcu, "Faster non-convex federated learning via global and local momentum," *arXiv preprint arXiv:2012.04061*, 2020.



(a) (b) (c) (d)

Fig. 1. Comparison of the DIST-CURE with (a) FedGLOMO and FedAvg (b) for Gaussian attack. (c) Comparison of DIST-CURE with robust PGD. (d) Plot of the gradient norm for 'a9a' data-set with Gaussian attack for robust linear regression.



(a) (b) (c) (d)

Fig. 2. (a) Plot of the function value with different initialization to show that the algorithm escapes the saddle point with functional value 0. (b,c,d) Comparison of our algorithm with ByzantinePGD [19] in terms of the total number of iterations.

APPENDIX

A. *Experimental results*

First we show that DIST-CURE indeed *escapes saddle point* with a toy example. We choose a $d=2$: $\min_{w\in\mathbb{R}^2}[f_1(w)+f_2(w)]$ where $f_1(w)=w_1^2-w_2^2$ and $f_2(w)=2w_1^2-2w_2^2$ (Here w_i denotes the i-th coordinate of w. This problem is the sum of two non-convex function and has a saddle point at $(0,0)$. In Figure 2 (a) we observe that our algorithm escapes the saddle point $(0,0)$, with random initialization.

Note that, checking whether a point is a local minima or a saddle point is an NP-hard problem for non-convex losses (see [54], Sec. 2.2). So, for a simple toy problem, we may brute-force our way through to show *saddle points escape*, but this becomes intractable for real data examples.

We now validate on benchmark LIBSVM ([26]) data-set in both convex and non-convex problems. We choose the following loss functions:

- Logistic loss:

$$\min_{\mathbf{w}\in\mathbb{R}^d}\frac{1}{n}\sum_i \log\big(1+\exp(-y_i\mathbf{x}_i^T\mathbf{w})\big)+\frac{\lambda}{2n}\|\mathbf{w}\|^2,$$

- Non-convex robust linear regression:

$$\min_{\mathbf{w}\in\mathbb{R}^d}\frac{1}{n}\sum_i \log\Big(\frac{(y_i-\mathbf{w}^T\mathbf{x}_i)^2}{2}+1\Big),$$

where $\mathbf{w}\in\mathbb{R}^d$ is the parameter, $\{\mathbf{x}_i\}_{i=1}^n\in\mathbb{R}^d$ are the feature vectors and $\{\mathbf{y}_i\}_{i=1}^n\in\{0,1\}$ are the corresponding labels. We choose 'a9a'($d=123, n\approx 32K$, and split the data into $70/30$ and use as training/testing purpose) and 'w8a'(training data $d=300, n\approx 50K$ and testing data $d=300, n\approx 15K$) classification datasets and partition the data in 20 machines.

We demonstrate DIST-CURE in the presence of Byzantine machines and compressed update. For compression, each local machine applies compression operator of QSGD [44]. For a given vector $\mathbf{x}\in\mathbb{R}^d$, we have

$$[Q(\mathbf{x})]_i=\|\mathbf{x}\|_2\text{sign}(\mathbf{x}_i)\times \texttt{Ber}(|\mathbf{x}_i|/\|x\|_2)$$

for all $i\in[d]$. We consider the following four Byzantine attacks:

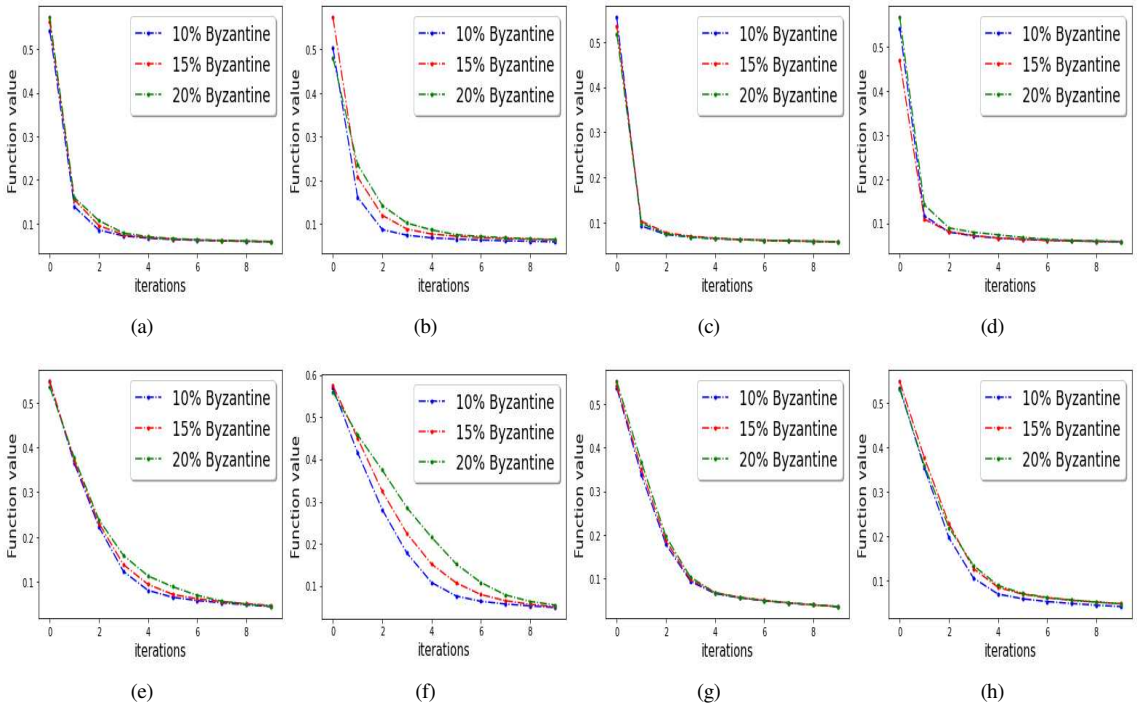


Fig. 3. Non convex robust linear regression with 'a9a' (a,b,c,d) and 'w8a' (e,f,g,h) with $10\%,15\%,20\%$ Byzantine local machines for (a,e). Flipped label attack.(b,f). Negative Update attack. (c,g) Gaussian attack. (d,h) Random attack.

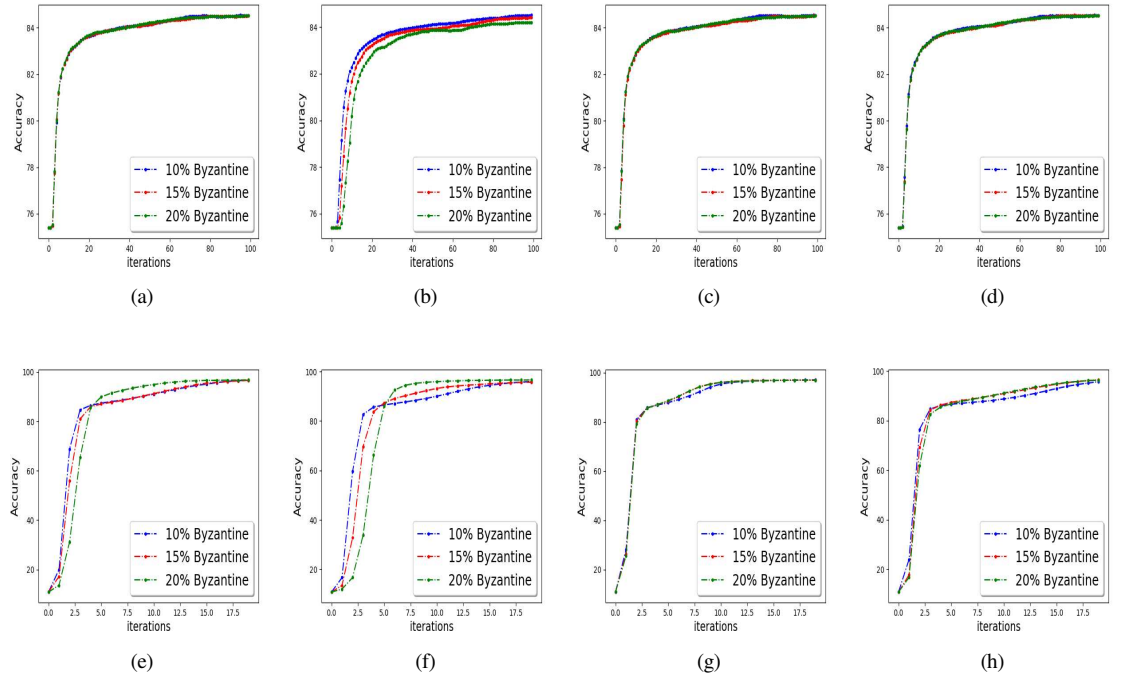


Fig. 4. Classification accuracy of the testing data 'a9a' dataset (first row) and 'w8a' dataset (second row) with $10\%,15\%,20\%$ Byzantine local machines for (a,e). Flipped label.(b,f). Negative Update (c,g). Gaussian noise and (d,h). Random label attack for logistic regression problem.

1) 'Gaussian Noise attack': where the Byzantine local machines add Gaussian noise to the update.
2) 'Random label attack': where the Byzantine local machines train and learn based on random labels instead of the proper labels.
3) 'Flipped label attack': where (for Binary classification) the Byzantine local machines flip the labels of the data and learn based

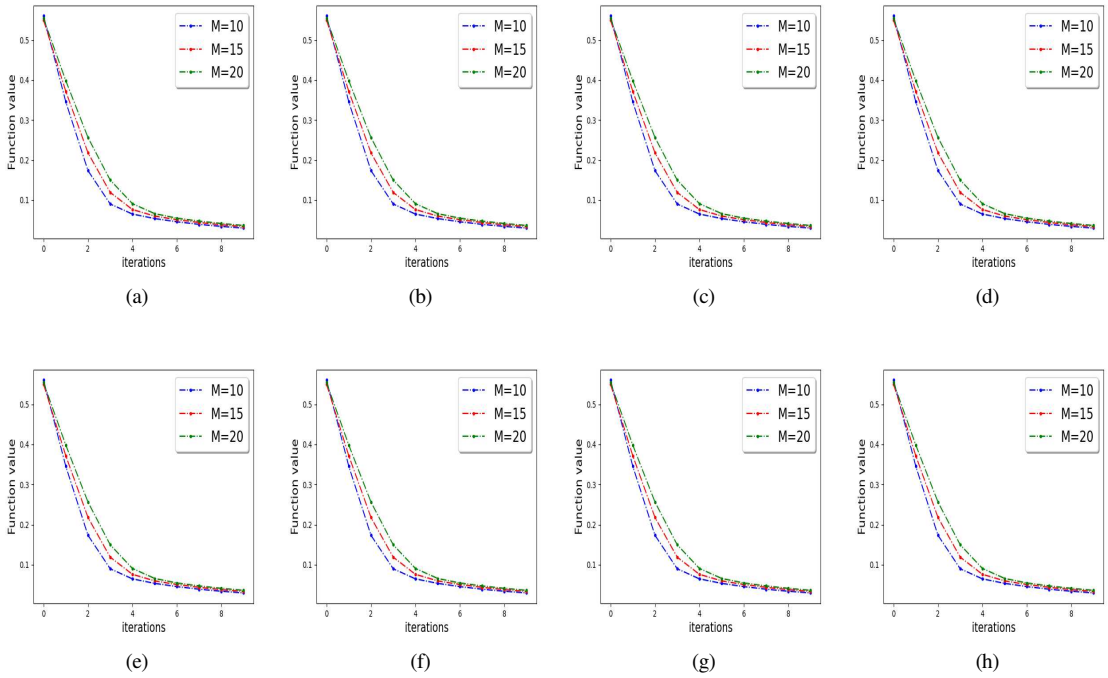


Fig. 5. Function training loss for the training data 'a9a' (first row) and 'w8a' (second row) with $10\%, 15\%, 20\%$ Byzantine local machines. for (a,e). Gaussian attack.(b,f).Random attack (c,g) Flipped Attack and (d,h) Negative update attack for non-convex robust linear regression problem.

on wrong labels.

4) 'Negative update attack': where the Byzantine local machines computes the update $\mathbf{s}$ (here solves the sub-problem in Eq. (1)) and communicates $-c*\mathbf{s}$ with $c \in (0,1)$ making the direction of the update opposite of the actual one.

a) ***Comparison with ByzantinePGD****:* We compare our uncompressed version of `DIST-CURE` ($\delta = 1$) with ByzantinePGD of [19] here. We take the *total number of iterations* as a comparison metric. One outer iteration of Algorithm 1 corresponds to one round of communication between the center and the local machines (and hence one parameter update). Note that in our algorithm the local machines use 10 steps of gradient solver (see [33]) for the local sub problem per iteration. So, the *total number of iterations* is given by 10 times the number of outer iterations. For both the algorithms, we choose ℓ_2 norm of the gradient as a stopping criteria. For ByzantinePGD, we choose $R = 10, r = 5, Q = 10, T_{th} = 10$ and 'co-ordinate wise Trimmed mean. In the Figure 2 (b-d), we plot the *total number of iterations* in all four types of attacks with different fraction of Byzantine machines. It is evident from the plot that our method requires less number of over all iterations (at least 48.4%, 29% and 25% less for 10%, 15% and 20% of Byzantine machines respectively).

Although `DIST-CURE` uses Hessian (second order) information, the sub-problem actually uses gradient based first order algorithm, and hence we compare the total iteration complexity mentioned above. To the best of our knowledge, there is no saddle point avoidance second order algorithm in FL framework, and so we adhere to the comparison with first order methods.

b) ***Comparison with standard FL algorithms****:* We have implemented and compared the performance of standard FL algorithm like FedGLOMO [55] (Federated Learning via Global and Local Momentum) and FedAvg [3] with `DIST-CURE`. The results are shown in Figure 1(a,b). Our method outperforms these standard baselines since they can tolerate Byzantine attacks (Gaussian attack in the experiment).

c) ***Training loss for compressed update****:* In Figure 3, we plot the function value of the robust linear regression problem for 'flipped labels', 'negative update', 'Gaussian' and 'Random label' attacks with compressed update for both 'w8a' and 'a9a' datasets. We choose the parameters $\lambda = 1, M = 10$, learning rate $\eta_k = 1$, $\alpha = \{.1, .15, .2\}$ and $\beta = \alpha + \frac{2}{m}$, where number of local machines $m = 20$. In Figure 1(d), we plot the gradient norm ($\|\mathbf{g}\|$) for Gaussian attack with 10%,15% and 20% of machines being Byzantine.

d) ***Classification accuracy****:* We show the classification accuracy on testing data of 'a9a' and 'w8a' dataset for logistic regression problem in Figure 4 and training function loss of 'a9a' and 'w8a' dataset for robust linear regression problem in the Figure 4. It is evident from the plots that a simple *norm based thresholding* makes the learning algorithm robust.

e) ***Training loss for uncompressed update****:* In Figure 5, we plot the function value of the robust linear regression with the similar attacks for the uncompressed update ($\delta = 1$) for both 'w8a' and 'a9a' dataset.

B. Proofs of main results

In this part, we establish some useful facts and lemmas. Next, we provide analysis of Theorems 1.

a) Proof Sketch: We now provide a proof sketch of Theorem 1. In Theorem 1, we provide the convergence guarantee of DIST-CURE in terms of the norm of the gradient and minimum eigenvalue for some point $\tilde{\mathbf{x}}$ in the iteration sequence $\{\mathbf{x}_0,...,\mathbf{x}_{T-1}\}$ satisfying

$$k_0 = \arg\min_{0\leq k\leq T-1} \|\mathbf{x}_{k+1}-\mathbf{x}_k\|.$$

First we bound the term

$$\|\mathbf{x}_{k_0+1}-\mathbf{x}_{k_0}\| = \|\eta_{k_0}\frac{1}{|\mathcal{U}_{k_0}|}\sum_{i\in\mathcal{U}_{k_0}}\mathbf{s}_{i,k_0+1}\|,$$

which is the aggregated update at iteration k_0. The sum is over $\mathcal{U}_{k_0}$ which is set of updates with smallest norm but the set may contain Byzantine update. Now, we choose $\beta>\alpha$ such that there is at least one good machine in the trimmed set. With this, in any iteration k, for Byzantine machine in the untrimmed set, $i\in\mathcal{U}_k\cap\mathcal{B}_k$, we have

$$\|\mathbf{s}_{i,k+1}\| \leq \max_{i\in\mathcal{M}_k}\|\mathbf{s}_{i,k+1}\|,$$

where $\mathcal{M}_k$ is the set of good machines. Next, we use the results of Lemma 1 and Lemma 2 (shown next) and the Assumptions 1-2 to establish the bound

$$\frac{1}{|\mathcal{M}_{k_0}|}\sum_{i\in\mathcal{M}_{k_0}}\|\eta_{k_0}\mathbf{s}_{i,k_0+1}\|^3 \leq \mathcal{O}(\frac{1}{T})+\mathcal{O}(\frac{1}{m}),$$

with the choice of $\eta_k=\frac{1}{Tm^\nu}$, for some $\nu>3$. Next, we use these facts to bound the norm of the gradient and minimum eigenvalue of the Hessian and establish the results.

C. Some useful facts

For the purpose of analysis we use the following sets of inequalities.

Fact 1: For $a_1,...,a_n$ we have the following inequality

$$\|\left(\sum_{i=1}^n a_i\right)\|^3 \leq \left(\sum_{i=1}^n\|a_i\|\right)^3 \leq n^2\sum_{i=1}^n\|a_i\|^3 \tag{2}$$

$$\|\left(\sum_{i=1}^n a_i\right)\|^2 \leq \left(\sum_{i=1}^n\|a_i\|\right)^2 \leq n\sum_{i=1}^n\|a_i\|^2 \tag{3}$$

Fact 2: For $a_1,...,a_n>0$ and $r<s$

$$\left(\frac{1}{n}\sum_{i=1}^n a_i^r\right)^{1/r} \leq \left(\frac{1}{n}\sum_{i=1}^n a_i^s\right)^{1/s} \tag{4}$$

Lemma 1 ([1]): Under Assumption 2, i.e., the Hessian of the function is L_2-Lipschitz continuous, for any $\mathbf{x},\mathbf{y}\in\mathbb{R}^d$, we have

$$\|\nabla f(\mathbf{x})-\nabla f(\mathbf{y})-\nabla^2 f(\mathbf{x})(\mathbf{y}-\mathbf{x})\| \leq \frac{L_2}{2}\|\mathbf{y}-\mathbf{x}\|^2 \tag{5}$$

$$\left|f(\mathbf{y})-f(\mathbf{x})-\nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x})-\frac{1}{2}(\mathbf{y}-\mathbf{x})^T\nabla^2 f(\mathbf{x})(\mathbf{y}-\mathbf{x})\right| \leq \frac{L_2}{6}\|\mathbf{y}-\mathbf{x}\|^2 \tag{6}$$

Next, we establish the following Lemma that provides some nice properties of the cubic sub-problem.

Lemma 2: Let $M>0,\gamma>0,\mathbf{g}\in\mathbb{R}^d,\mathbf{H}\in\mathbb{R}^{d\times d}$, and

$$\mathbf{s} = \arg\min_{\mathbf{x}} \mathbf{g}^T\mathbf{x}+\frac{\gamma}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}+\frac{M\gamma^2}{6}\|\mathbf{x}\|^3. \tag{7}$$

The following holds

$$\mathbf{g}+\gamma\mathbf{H}\mathbf{s}+\frac{M\gamma^2}{2}\|\mathbf{s}\|\mathbf{s}=0, \tag{8}$$

$$\mathbf{H}+\frac{M\gamma}{2}\|\mathbf{s}\|\mathbf{I}\succeq\mathbf{0}, \tag{9}$$

$$\mathbf{g}^T\mathbf{s}+\frac{\gamma}{2}\mathbf{s}^T\mathbf{H}\mathbf{s} \leq -\frac{M}{4}\gamma^2\|\mathbf{s}\|^3. \tag{10}$$

Proof 1: The equations (8) and (9) are from the first and second order optimal condition. We proof (10), by using the conditions of (8) and (9).

$$\mathbf{g}^T\mathbf{s}+\frac{\gamma}{2}\gamma\mathbf{s}^T\mathbf{H}\mathbf{s}=-\left(\gamma\mathbf{H}\mathbf{s}+\frac{M}{2}\gamma^2\|\mathbf{s}\|\mathbf{s}\right)^T\mathbf{s}+\frac{\gamma}{2}\gamma\mathbf{s}^T\mathbf{H}\mathbf{s} \tag{11}$$

$$=-\gamma\mathbf{s}^T\mathbf{H}\mathbf{s}-\frac{M}{2}\gamma^2\|\mathbf{s}\|^3+\frac{\gamma}{2}\gamma\mathbf{s}^T\mathbf{H}\mathbf{s}$$

$$\leq\frac{M}{4}\gamma^2\|\mathbf{s}\|^3-\frac{M}{2}\gamma^2\|\mathbf{s}\|^3 \tag{12}$$

$$=-\frac{M}{4}\gamma^2\|\mathbf{s}\|^3.$$

In (11), we substitute the expression $\mathbf{g}$ from the equation (8). In (12), we use the fact that $\mathbf{s}^T\mathbf{H}\mathbf{s}+\frac{M\gamma}{2}\|\mathbf{s}\|^3>0$ from the equation (9).

D. Proof of Theorem 1 (Main Theorem)

First we state the results of Lemma 2 for each local machine in iteration k,

$$\mathbf{g}_{i,k}+\gamma\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}+\frac{M}{2}\gamma^2\|\mathbf{s}_{i,k+1}\|\mathbf{s}_{i,k+1}=0 \tag{13}$$

$$\gamma\mathbf{H}_{i,k}+\frac{M}{2}\gamma^2\|\mathbf{s}_{i,k+1}\|\mathbf{I}\succeq\mathbf{0} \tag{14}$$

$$\mathbf{g}_{i,k}^T\mathbf{s}_{i,k+1}+\frac{\gamma}{2}\mathbf{s}_{i,k+1}^T\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}\leq-\frac{M}{4}\gamma^2\|\mathbf{s}_{i,k+1}\|^3 \tag{15}$$

We also use the following fact form the setup and trimming set

$$|\mathcal{U}|=|\mathcal{U}\cap\mathcal{M}|+|\mathcal{U}\cap\mathcal{B}| \tag{16}$$

$$|\mathcal{M}|=|\mathcal{U}\cap\mathcal{M}|+|\mathcal{T}\cap\mathcal{M}| \tag{17}$$

Combining both the equations (16) and (17), we have

$$|\mathcal{U}|=|\mathcal{M}|-|\mathcal{T}\cap\mathcal{M}|+|\mathcal{U}\cap\mathcal{B}| \tag{18}$$

Now we state the following fact from the trimming set. as mentioned in the Algorithm 1, the norm of the update from any local machines from the set $\mathcal{U}$ is less than the norm of the update from any local machines in the set $\mathcal{T}$. Now as $\beta>\alpha$, at least one good machine (the largest norm) is in the set $\mathcal{T}$. So, we can claim the following,

$$\text{For all } i\in\mathcal{U}\cap\mathcal{B},\quad \|\mathbf{s}_i\|\leq\max_{i\in\mathcal{M}}\|\mathbf{s}_i\|$$

Summing over all the Byzantine machines in the untrimmed set which is $\mathcal{U}\cap\mathcal{B}$, we get

$$\sum_{i\in\mathcal{U}\cap\mathcal{B}}\|.\|\leq\alpha m\max_{i\in\mathcal{M}}\|.\|$$

as $|\mathcal{U}\cap\mathcal{B}|\leq\alpha m$. Also,

$$\sum_{i\in\mathcal{M}\cap\mathcal{T}}\|.\|\leq\sum_{i\in\mathcal{M}}\|.\|$$

Combining the above two equations, we get

$$\sum_{i\in\mathcal{U}\cap\mathcal{B}}\|.\|+\sum_{i\in\mathcal{M}\cap\mathcal{T}}\|.\|\leq\sum_{i\in\mathcal{M}}\|.\|+\alpha m\max_{i\in\mathcal{M}}\|.\| \tag{19}$$

For the rest of the calculation, we use the following notation

$$\Gamma=\max_{i\in\mathcal{M},k}\|\mathbf{s}_{i,k}\|. \tag{20}$$

If the optimization sub-problem domain is bounded, Γ can be upper-bounded by the diameter of the parameter space. Note that in the definition of Γ, the maximum is taken over good machines only.

Characterization of Γ *::* For any good local machine $i \in \mathcal{M}$, we have the following

$$\mathbf{s}_{i,k+1} = \operatorname{argmin}_{\mathbf{s}} \mathbf{g}_{i,k}^T \mathbf{s} + \frac{\gamma}{2} \mathbf{s}^T \mathbf{H}_{i,k} \mathbf{s} + \gamma^2 \frac{M}{6} \|\mathbf{s}\|^3$$

for some $M > 0$ and $\gamma = \frac{c}{T}$. Next, we consider $\mathbf{u}_{i,k+1} = \gamma \mathbf{s}_{i,k+1}$ and we get the following expression

$$\mathbf{u}_{i,k+1} = \operatorname{argmin}_{\mathbf{u}} \mathbf{g}_{i,k}^T \mathbf{u} + \frac{1}{2} \mathbf{u}^T \mathbf{H}_{i,k} \mathbf{u} + \frac{M}{6} \|\mathbf{u}\|^3$$

Following the similar results of the 2, we have the following result from the second order condition,

$$\mathbf{g}_{i,k}^T \mathbf{u}_{i,k+1} + \frac{1}{2} \mathbf{u}_{i,k+1}^T \mathbf{H}_{i,k} \mathbf{u}_{i,k+1} \leq -\frac{M}{4} \|\mathbf{u}_{i,k+1}\|^3.$$

Therefore,

$$\begin{aligned}
&\frac{M}{4} \|\mathbf{u}_{i,k+1}\|^3 \\
\leq& \|\mathbf{g}_{i,k}\| \|\mathbf{u}_{i,k+1}\| + \frac{1}{2} \|\mathbf{H}_{i,k}\| \|\mathbf{u}_{i,k+1}\|^2 \\
=& \|\mathbf{g}_{i,k} - \nabla f(\mathbf{s}_{i,k+1}) + \nabla f(\mathbf{s}_{i,k+1})\| \|\mathbf{u}_{i,k+1}\| + \frac{1}{2} \|\mathbf{H}_{i,k} - \nabla^2 f(\mathbf{s}_{i,k+1}) + \nabla^2 f(\mathbf{s}_{i,k+1})\| \|\mathbf{u}_{i,k+1}\|^2 \\
\leq& (\|\mathbf{g}_{i,k} - \nabla f(\mathbf{s}_{i,k+1})\| + \|\nabla f(\mathbf{s}_{i,k+1})\| \|\mathbf{u}_{i,k+1}\| + \frac{1}{2} (\|\mathbf{H}_{i,k} - \nabla^2 f(\mathbf{s}_{i,k+1})\| + \|\nabla^2 f(\mathbf{s}_{i,k+1})\|) \|\mathbf{u}_{i,k+1}\|^2 \\
\leq& (\epsilon_g + L) \|\mathbf{u}_{i,k+1}\| + (\epsilon_H + L_1) \|\mathbf{u}_{i,k+1}\|^2
\end{aligned}$$

In the above expression, we have ϵ_g, ϵ_H are gradient and Hessian dissimilarity respectively and $\|\nabla f(\mathbf{s}_{i,k+1}\| \leq L, \|\nabla^2 f(\mathbf{s}_{i,k+1}\| \leq L_1$ which are constants. This shows that $\|\mathbf{u}_{i,k+1}\|$ to be bounded and hence $\max_{i \in \mathcal{M}} \|\mathbf{u}_{i,k+1}\|$ to be bounded. For $\gamma = \frac{c}{T}$, we have

$$\begin{aligned}
\|\mathbf{s}_{i,k+1}\| &= \|\mathbf{u}_{i,k+1}/\gamma\| = O(T) \\
\Rightarrow \Gamma &= O(T)
\end{aligned} \tag{21}$$

At any iteration k, we have (with Taylor's expansion)

$$\begin{aligned}
&f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k) \\
\leq& \nabla f(\mathbf{x}_k)^T (\mathbf{x}_{k+1} - \mathbf{x}_k) + \frac{1}{2} (\mathbf{x}_{k+1} - \mathbf{x}_k)^T \nabla^2 f(\mathbf{x}_k) (\mathbf{x}_{k+1} - \mathbf{x}_k) + \frac{L_2}{6} \|\mathbf{x}_{k+1} - \mathbf{x}_k\|^3 \\
=& \underbrace{\frac{\eta_k}{|\mathcal{U}|} \nabla f(\mathbf{x}_k)^T \sum_{i \in \mathcal{U}} \mathbf{s}_{i,k+1}}_{Term1} + \underbrace{\frac{\eta_k^2}{2|\mathcal{U}|^2} \left(\sum_{i \in \mathcal{U}} \mathbf{s}_{i,k+1} \right)^T \nabla^2 f(\mathbf{x}_k) \left(\sum_{i \in \mathcal{U}} \mathbf{s}_{i,k+1} \right)}_{Term2} \\
&+ \underbrace{\frac{L_2}{6} \left\| \frac{\eta_k}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \mathbf{s}_{i,k+1} \right\|^3}_{Term3}
\end{aligned} \tag{22}$$

In 22, we use the update of the parameter in the center machine $\mathbf{x}_{k+1} - \mathbf{x}_k = \frac{\eta_k}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \mathbf{s}_{i,k+1}$, as expressed in the Algorithm 1. Here $Q(\mathbf{s}_{i,k+1}) = \mathbf{s}_{i,k+1}$ as $\delta = 1$.

First we choose the Term 1 in (22) and expand it using (18)

$$\begin{aligned}
&\frac{\eta_k}{|\mathcal{U}|} \nabla f(\mathbf{x}_k)^T \sum_{i \in \mathcal{U}} \mathbf{s}_{i,k+1} \\
=& \frac{\eta_k}{(1-\beta)m} \nabla f(\mathbf{x}_k)^T \left[\sum_{i \in \mathcal{M}} \mathbf{s}_{i,k+1} - \sum_{i \in \mathcal{M} \cap \mathcal{T}} \mathbf{s}_{i,k+1} + \sum_{i \in \mathcal{U} \cap \mathcal{B}} \mathbf{s}_{i,k+1} \right] \\
=& \underbrace{\frac{\eta_k}{(1-\beta)m} \sum_{i \in \mathcal{M}} \left[\mathbf{g}_{i,k}^T \mathbf{s}_{i,k+1} + \nabla f(\mathbf{x}_k)^T \mathbf{s}_{i,k+1} - \mathbf{g}_{i,k}^T \mathbf{s}_{i,k+1} \right]}_{Term1.1} \\
&+ \underbrace{\frac{\eta_k}{(1-\beta)m} \nabla f(\mathbf{x}_k)^T \left[- \sum_{i \in \mathcal{M} \cap \mathcal{T}} \mathbf{s}_{i,k+1} + \sum_{i \in \mathcal{U} \cap \mathcal{B}} \mathbf{s}_{i,k+1} \right]}_{Term1.2}
\end{aligned} \tag{23}$$

First we consider Term 1.1 in (23) (notice that the sum is over only good machines),

$$\begin{aligned}
&\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\left[\mathbf{g}_{i,k}^T\mathbf{s}_{i,k+1}+\nabla f(\mathbf{x}_k)^T\mathbf{s}_{i,k+1}-\mathbf{g}_{i,k}^T\mathbf{s}_{i,k+1}\right]\\
=&\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\mathbf{g}_{i,k}^T\mathbf{s}_{i,k+1}+\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\left[(\nabla f(\mathbf{x}_k)-\mathbf{g}_{i,k})^T\mathbf{s}_{i,k+1}\right]\\
\leq&\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\mathbf{g}_{i,k}^T\mathbf{s}_{i,k+1}+\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}[\|\nabla f(\mathbf{x}_k)-\mathbf{g}_{i,k}\|\|\mathbf{s}_{i,k+1}\|]\\
\leq&\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\mathbf{g}_{i,k}^T\mathbf{s}_{i,k+1}+\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}[\epsilon_g\|\mathbf{s}_{i,k+1}\|] &(24)\\
\leq&\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\mathbf{g}_{i,k}^T\mathbf{s}_{i,k+1}+\frac{\eta_k(1-\alpha)}{(1-\beta)}\epsilon_g\Gamma &(25)
\end{aligned}$$

In (24), we use the following facts: 1. $\|\nabla f(\mathbf{x}_k)\|\leq L$ as the function f is L- Lipschitz. 2. $\|\nabla f(\mathbf{x}_k)-\mathbf{g}_{i,k}\|\leq\epsilon_g$ (gradient dissimilarity). In (25), we use the bound stated in (20).

Next we consider Term1.2 in (23),

$$\begin{aligned}
&\frac{\eta_k}{(1-\beta)m}\nabla f(\mathbf{x}_k)^T\left[-\sum_{i\in\mathcal{M}\cap\mathcal{T}}\mathbf{s}_{i,k+1}+\sum_{i\in\mathcal{U}\cap\mathcal{B}}\mathbf{s}_{i,k+1}\right]\\
\leq&\frac{\eta_k}{(1-\beta)m}\left[\sum_{i\in\mathcal{M}\cap\mathcal{T}}\|\nabla f(\mathbf{x}_k)\|\|\mathbf{s}_{i,k+1}\|+\sum_{i\in\mathcal{U}\cap\mathcal{B}}\|\nabla f(\mathbf{x}_k)\|\|\mathbf{s}_{i,k+1}\|\right]\\
\leq&\frac{\eta_k L}{(1-\beta)m}\left[\sum_{i\in\mathcal{M}\cap\mathcal{T}}\|\mathbf{s}_{i,k+1}\|+\sum_{i\in\mathcal{U}\cap\mathcal{B}}\|\mathbf{s}_{i,k+1}\|\right] &(26)\\
\leq&\frac{\eta_k L}{(1-\beta)m}\left[\sum_{i\in\mathcal{T}}\max_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|+\sum_{i\in\mathcal{B}}\max_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|\right] &(27)\\
\leq&\frac{\eta_k L}{(1-\beta)m}\left[\beta m\max_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|+\alpha m\max_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|\right] &(28)\\
\leq&\frac{\eta_k(\alpha+\beta)L}{(1-\beta)}\left[\max_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|)\right] &(29)\\
\leq&\frac{\eta_k(\alpha+\beta)L}{\Gamma} &(30)
\end{aligned}$$

We use the fact $\|\nabla f(\mathbf{x}_k)\|\leq L$ in (26), ,the fact stated in (19), in (27). We use the bound of update as described in (20) in (30).

We apply the bound derived for Term1.1 in (25) and for Term1.2 in (30) in the bound for Term1 in (23) and derive the following,

$$\begin{aligned}
&Term1\\
\leq&\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\mathbf{g}_{i,k}^T\mathbf{s}_{i,k+1}+\frac{\eta_k}{(1-\beta)}\epsilon_g\Gamma+\frac{\eta_k(\alpha+\beta)L}{(1-\beta)}\Gamma\\
=&\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\left[\mathbf{g}_{i,k}^T\mathbf{s}_{i,k+1}+\frac{\gamma}{2}\mathbf{s}_{i,k+1}^T\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}\right]-\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\frac{\gamma}{2}\mathbf{s}_{i,k+1}^T\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}+\frac{\eta_k(1-\alpha)}{(1-\beta)}\epsilon_g\Gamma+\frac{\eta_k(\alpha+\beta)L}{(1-\beta)}\Gamma\\
\leq&-\frac{\gamma^2M\eta_k}{4(1-\beta)m}\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^3-\frac{\eta_k}{(1-\beta)m}\sum_{i\in\mathcal{M}}\frac{\gamma}{2}\mathbf{s}_{i,k+1}^T\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}+\frac{\eta_k(1-\alpha)}{(1-\beta)}\epsilon_g\Gamma+\frac{\eta_k(\alpha+\beta)L}{(1-\beta)}\Gamma &(31)
\end{aligned}$$

In line (31), we use the bound stated in (15).

Now we consider the Term 3 in equation (22),

$$\begin{aligned}
&\frac{L_2}{6}\left\|\frac{\eta_k}{|\mathcal{U}|}\sum_{i\in\mathcal{U}}\mathbf{s}_{i,k+1}\right\|^3\\
\leq&\frac{L_2\eta_k^3}{6|\mathcal{U}|}\sum_{i\in\mathcal{U}}\|\mathbf{s}_{i,k+1}\|^3 &(32)
\end{aligned}$$

$$\leq \frac{L_2\eta_k^3}{6|\mathcal{U}|}\left[\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^3 - \sum_{i\in\mathcal{M}\cap\mathcal{T}}\|\mathbf{s}_{i,k+1}\|^3 + \sum_{i\in\mathcal{U}\cap\mathcal{B}}\|\mathbf{s}_{i,k+1}\|^3\right] \tag{33}$$

$$\leq \frac{L_2\eta_k^3}{6|\mathcal{U}|}\left[\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^3 + \sum_{i\in\mathcal{U}\cap\mathcal{B}}\|\mathbf{s}_{i,k+1}\|^3\right] \tag{34}$$

$$\leq \frac{L_2\eta_k^3}{6(1-\beta)m}\left[\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^3 + \alpha m \max_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^3\right] \tag{35}$$

$$\leq \frac{L_2\eta_k^3}{6(1-\beta)m}\left[\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^3 + \alpha m \max_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^3\right] \tag{36}$$

$$\leq \frac{L_2\eta_k^3}{6(1-\beta)m}\left[\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^3 + \alpha m \Gamma^3\right] \tag{37}$$

In (32), we use the fact stated in (2). Next in (33), we expand the trimmed set $\mathcal{U}$ using (18) and in (35), we use the bound of (19). Finally, in (36), we use the definition of the δ-compressor and the bound stated in (20) in (37).

Now we consider the Term 2 in (22)

$$\begin{aligned}
&\frac{\eta_k^2}{2|\mathcal{U}|^2}\left(\sum_{i\in\mathcal{U}}\mathbf{s}_{i,k+1}\right)^T\nabla^2 f(\mathbf{x}_k)\left(\sum_{i\in\mathcal{U}}\mathbf{s}_{i,k+1}\right)\\
&= \frac{\eta_k^2}{2(1-\beta)^2m^2}\underbrace{\sum_{i\in\mathcal{U}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}}_{Term2.1}\\
&+ \frac{\eta_k^2}{2(1-\beta)^2m^2}\underbrace{\sum_{i\neq j\in\mathcal{U}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)Q(\mathbf{s}_{j,k+1})}_{Term2.2}
\end{aligned} \tag{38}$$

Now we consider Term2.1 in (38) and expand it using (18)

$$\begin{aligned}
&\sum_{i\in\mathcal{U}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}\\
&= \underbrace{\sum_{i\in\mathcal{M}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}}_{Term2.1.1} - \underbrace{\sum_{i\in\mathcal{M}\cap\mathcal{T}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1} + \sum_{i\in\mathcal{B}\cap\mathcal{U}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}}_{Term2.1.2}
\end{aligned}$$

We consider Term2.1.1

$$\begin{aligned}
&\sum_{i\in\mathcal{M}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}\\
&= \sum_{i\in\mathcal{M}}\mathbf{s}_{i,k+1}^T\mathbf{H}_{i,k}\mathbf{s}_{i,k+1} - \sum_{i\in\mathcal{M}}\mathbf{s}_{i,k+1}^T\mathbf{H}_{i,k}\mathbf{s}_{i,k+1} + \sum_{i\in\mathcal{M}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}\\
&= \sum_{i\in\mathcal{M}}\mathbf{s}_{i,k+1}^T\mathbf{H}_{i,k}\mathbf{s}_{i,k+1} - \sum_{i\in\mathcal{M}}\mathbf{s}_{i,k+1}^T(\mathbf{H}_{i,k} - \nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}\\
&\leq \sum_{i\in\mathcal{M}}\mathbf{s}_{i,k+1}^T\mathbf{H}_{i,k}\mathbf{s}_{i,k+1} + (1-\alpha)m\epsilon_H\Gamma^2
\end{aligned} \tag{39}$$

In 39, we use the Hessian dissimilarity bound of $\|(\nabla^2 f(\mathbf{x}_k) - \mathbf{H}_{i,k})\| \leq \epsilon_H$.

Next, we consider the Term2.1.2,

$$\begin{aligned}
&\sum_{i\in\mathcal{M}\cap\mathcal{T}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1} + \sum_{i\in\mathcal{B}\cap\mathcal{U}}\mathbf{s}_{i,k+1}^T\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}\\
&\leq \sum_{i\in\mathcal{M}\cap\mathcal{T}}L_1\|\mathbf{s}_{i,k+1}\|^2 + \sum_{i\in\mathcal{B}\cap\mathcal{U}}L_1\|\mathbf{s}_{i,k+1}\|^2 \qquad (40)\\
&\leq \sum_{i\in\mathcal{B}}\max_{i\in\mathcal{T}}L_1\|\mathbf{s}_{i,k+1}\|^2 + \sum_{i\in\mathcal{B}}\max_{i\in\mathcal{M}}L_1\|\mathbf{s}_{i,k+1}\|^2 \qquad (41)
\end{aligned}$$

$$\leq \beta m L_1 \Gamma^2 + \alpha m L_1 \Gamma^2$$
$$= (\alpha+\beta) m L_1 \Gamma^2 \tag{42}$$

Combining (39) and (42), we bound the Term2.1,

$$\text{Term2.1}$$
$$\leq \sum_{i\in\mathcal{M}} \mathbf{s}_{i,k+1}^T \mathbf{H}_{i,k} \mathbf{s}_{i,k+1} + (1-\alpha) m \epsilon_H \Gamma^2 + (\alpha+\beta) m L_1 \Gamma^2 \tag{43}$$

Now we consider the Term 2.2 in equation (38)

$$\sum_{i\neq j\in\mathcal{U}} \mathbf{s}_{i,k+1}^T \nabla^2 f(\mathbf{x}_k) \mathbf{s}_{j,k+1}$$
$$\leq \sum_{i\neq j\in\mathcal{U}} L_1 \|\mathbf{s}_{i,k+1}\| \|\mathbf{s}_{j,k+1}\| \tag{44}$$
$$= L_1 \left[\|\sum_{i\in\mathcal{U}} \mathbf{s}_{i,k+1}\|^2 - \sum_{i\in\mathcal{U}} \|\mathbf{s}_{i,k+1}\|^2 \right]$$
$$\leq L_1 \left[|\mathcal{U}| \sum_{i\in\mathcal{U}} \|\mathbf{s}_{i,k+1}\|^2 - \sum_{i\in\mathcal{U}} \|\mathbf{s}_{i,k+1}\|^2 \right]$$
$$= L_1((1-\beta)m-1) \left[\sum_{i\in\mathcal{M}} \|\mathbf{s}_{i,k+1}\|^2 - \sum_{i\in\mathcal{M}\cap\mathcal{T}} \|\mathbf{s}_{i,k+1}\|^2 + \sum_{i\in\mathcal{B}\cap\mathcal{U}} \|\mathbf{s}_{i,k+1}\|^2 \right] \tag{45}$$
$$\leq L_1((1-\beta)m-1) \left[\sum_{i\in\mathcal{U}} \|\mathbf{s}_{i,k+1}\|^2 \right] \tag{46}$$
$$= L_1((1-\beta)m-1)(1-\beta)m\Gamma^2 \tag{47}$$

We use the expansion described in (18) in (45).

Now combining the results in (47) and (38) we get,

$$Term2$$
$$\leq \frac{\eta_k^2}{2(1-\beta)^2 m^2} \left[\sum_{i\in\mathcal{M}} \mathbf{s}_{i,k+1}^T \mathbf{H}_{i,k} \mathbf{s}_{i,k+1} + (1-\alpha) m \epsilon_H \Gamma^2 + (\alpha+\beta) m L_1 \Gamma^2 \right] + \frac{\eta_k^2}{2(1-\beta)^2 m^2} L_1((1-\beta)m-1)(1-\beta)m\Gamma^2$$

Now we combine all the upper bound of the Term 1, Term 2 and Term 3

$$f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)$$
$$\leq -\frac{\gamma^2 M \eta_k}{4(1-\beta)m} \sum_{i\in\mathcal{M}} \|\mathbf{s}_{i,k+1}\|^3 - \frac{\eta_k}{(1-\beta)m} \sum_{i\in\mathcal{M}} \frac{\gamma}{2} \mathbf{s}_{i,k+1}^T \mathbf{H}_{i,k} \mathbf{s}_{i,k+1} + \frac{\eta_k(1-\alpha)}{(1-\beta)} \epsilon_g \Gamma + \frac{\eta_k(\alpha+\beta)L}{(1-\beta)} \Gamma$$
$$+ \frac{\eta_k^2}{2(1-\beta)^2 m^2} \left[\sum_{i\in\mathcal{M}} \mathbf{s}_{i,k+1}^T \mathbf{H}_{i,k} \mathbf{s}_{i,k+1} + (1-\alpha) m \epsilon_H \Gamma^2 + (\alpha+\beta) m L_1 \Gamma^2 \right]$$
$$+ \frac{\eta_k^2}{2} L_1 \left(\frac{1+\delta}{\delta}\right) \Gamma^2 + \frac{L_2 \eta_k^3}{6(1-\beta)m} \left[\sum_{i\in\mathcal{M}} \|\mathbf{s}_{i,k+1}\|^3 + \alpha m \Gamma^3 \right]$$
$$= \left(-\frac{\gamma^2 M \eta_k}{4(1-\beta)m} + \frac{L_2 \eta_k^3}{6(1-\beta)m} \right) \sum_{i\in\mathcal{M}} \|\mathbf{s}_{i,k+1}\|^3 - \frac{\eta_k}{2(1-\beta)m} \left(\gamma - \frac{\eta_k}{(1-\beta)m} \right) \mathbf{s}_{i,k+1}^T \mathbf{H}_{i,k} \mathbf{s}_{i,k+1}$$
$$+ \left(\frac{\eta_k(1-\alpha)}{(1-\beta)} \epsilon_g + \frac{\eta_k(\alpha+\beta)L}{(1-\beta)} \right) \Gamma + \frac{L_2 \eta_k^3}{6(1-\beta)} \alpha \Gamma^3$$
$$+ \frac{\eta_k^2}{2(1-\beta)^2 m} ((1-\alpha)\epsilon_H + (\alpha+\beta)L_1 + L_1((1-\beta)m-1)(1-\beta)m) \Gamma^2$$

Also we assume that $\gamma \geq \frac{\eta_k}{(1-\beta)m}$ and use the fact $-\mathbf{s}_{i,k+1} \mathbf{H}_{i,k} \mathbf{s}_{i,k+1} \leq \frac{M\gamma}{2} \|\mathbf{s}_{i,k+1}\|^3$. We also choose that

$$\lambda_\Gamma = \left(\frac{\eta_k(1-\alpha)}{(1-\beta)} \epsilon_g + \frac{\eta_k(\alpha+\beta)L}{(1-\beta)} \right) \Gamma + \frac{L_2 \eta_k^3}{6(1-\beta)} \alpha \Gamma^3 + \frac{\eta_k^2}{2(1-\beta)^2 m} ((1-\alpha)\epsilon_H + (\alpha+\beta)L_1 + L_1((1-\beta)m-1)(1-\beta)m) \Gamma^2 \tag{48}$$

Using the fact step-size $\eta_k = \frac{c}{m^\nu T}$ for some $\nu \geq 3$ and the bound of Γ as described in (21), we have λ_Γ to be upper bounded by $\mathcal{O}(\frac{1}{m^\nu})$

Using the fact step-size $\eta_k = \frac{c}{m^\nu T}$ for some $\nu \geq 3$ and the bound of Γ as described in (21), we have λ_Γ to be upper bounded by $\mathcal{O}(\frac{1}{m^\nu})$
Now we have,

$$
\begin{aligned}
& f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k) && (49)\\
\leq & \left(-\frac{\gamma^2 M \eta_k}{4(1-\beta)m} + \frac{L_2 \eta_k^3}{6(1-\beta)m} \right) \sum_{i \in \mathcal{M}} \|\mathbf{s}_{i,k+1}\|^3 + \frac{\eta_k}{2(1-\beta)m} \left(\gamma - \frac{\eta_k}{(1-\beta)m} \right) \sum_{i \in \mathcal{M}} \frac{M\gamma}{2} \|\mathbf{s}_{i,k+1}\|^3 + \lambda_\Gamma \\
= & \left(-\frac{\gamma M \eta_k^2}{4(1-\beta)^2 m^2} + \frac{L_2 \eta_k^3}{6(1-\beta)m} \right) \sum_{i \in \mathcal{M}} \|\mathbf{s}_{i,k+1}\|^3 + \lambda_\Gamma \\
= & -\lambda_{comp} \frac{1}{(1-\alpha)m} \sum_{i \in \mathcal{M}} \|\eta_k \mathbf{s}_{i,k+1}\|^3 + \lambda_\Gamma && (50)
\end{aligned}
$$

where

$$
\lambda_{comp} = [\frac{\gamma M}{4(1-\beta)^2 \eta_k m^2} - \frac{L_2}{6(1-\beta)m}](1-\alpha)m
$$

To ensure $\lambda_{comp} > 0$, we need

$$
M > \frac{4\eta_k m(1-\beta)}{\gamma} \frac{L_2}{6} \tag{51}
$$

Now for the choice of $\eta_k = \frac{c}{Tm^\nu}$ and $\gamma = \frac{c_1}{Tm^\nu}$ for some constant $c_1 > 0$. We have $M = \mathcal{O}(m(1-\beta)(\frac{1+\delta}{\delta})^{3/2})$. Thus we have $\lambda_{comp} = \mathcal{O}(1)$ and $\lambda_\Gamma = \mathcal{O}(\frac{1}{m^\nu})$. Now we have

$$
\frac{1}{(1-\alpha)m} \sum_{i \in \mathcal{M}} \|\eta_k \mathbf{s}_{i,k+1}\|^3 \leq \frac{1}{\lambda_{comp}} [f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) + \lambda_\Gamma]
$$

At any iteration k, we have

$$
\begin{aligned}
\|\mathbf{x}_{k+1} - \mathbf{x}_k\|^3 &= \|\eta_k \ s_{k+1}\|^3 \\
&\leq \frac{1}{(1-\beta)m} \sum_{i \in \mathcal{U}} \|\eta_{k_0} \mathbf{s}_{i,k+1}\|^3 \\
&\leq \frac{1}{(1-\beta)m} \sum_{i \in \mathcal{U}} \|\eta_k \mathbf{s}_{i,k+1}\|^3 \\
&= \frac{1}{(1-\beta)m} \left[\sum_{i \in \mathcal{M}} \|\eta_k \mathbf{s}_{i,k+1}\|^3 - \sum_{i \in \mathcal{M} \cap \mathcal{T}} \|\eta_k \mathbf{s}_{i,k+1}\|^3 + \sum_{i \in \mathcal{U} \cap \mathcal{B}} \|\eta_k \mathbf{s}_{i,k+1}\|^3 \right] \\
&\leq \frac{1}{(1-\beta)m} \left[\sum_{i \in \mathcal{M}} \|\eta_k \mathbf{s}_{i,k+1}\|^3 + \alpha m \eta_k^3 \Gamma^3 \right]
\end{aligned}
$$

Now we consider the step k_0, where $k_0 = \text{argmin}_{0 \leq k \leq T-1} \|\mathbf{x}_{k+1} - \mathbf{x}_k\|$.

$$
\begin{aligned}
& \min_{0 \leq k \leq T} \|\mathbf{x}_{k+1} - \mathbf{x}_k\|^3 \\
& \leq \min_{0 \leq k \leq T} \frac{1}{(1-\beta)m} \left[\sum_{i \in \mathcal{M}} \|\eta_k \mathbf{s}_{i,k+1}\|^3 + \alpha m \eta_k^3 \Gamma^3 \right] \\
& \leq \frac{1}{T} \sum_{k=0}^{T-1} \frac{(1-\alpha)}{(1-\beta)} \left[\frac{1}{(1-\alpha)m} \sum_{i \in \mathcal{M}} \|\eta_k \mathbf{s}_{i,k+1}\|^3 + \frac{\alpha}{1-\alpha} \eta_{k_0}^3 \Gamma^3 \right] \\
& \leq \frac{1}{T} \sum_{k=0}^{T-1} \frac{(1-\alpha)}{(1-\beta)} \left[\frac{f(\mathbf{x}_k) - f(\mathbf{x}_{k+1})}{\lambda_{comp}} + \frac{\lambda_\Gamma}{\lambda_{comp}} + \frac{\alpha}{1-\alpha} \frac{\eta_{k_0}^3 \Gamma^3}{\lambda_{comp}} \right] \\
& \leq \frac{1}{T} \frac{(1-\alpha)}{(1-\beta)} \left[\frac{f(\mathbf{x}_0) - f^*}{\lambda_{comp}} + \sum_{k=0}^{T-1} \frac{\lambda_\Gamma}{\lambda_{comp}} + \sum_{k=0}^{T-1} \frac{\alpha}{1-\alpha} \frac{\eta_{k_0}^3 \Gamma^3}{\lambda_{comp}} \right] \\
& \leq \frac{(1-\alpha)}{(1-\beta)} \left[\frac{f(\mathbf{x}_0) - f^*}{T\lambda_{comp}} + \frac{\lambda_\Gamma}{\lambda_{comp}} + \frac{\alpha}{1-\alpha} \frac{\eta_{k_0}^3 \Gamma^3}{\lambda_{comp}} \right]
\end{aligned}
$$

With the choice of η_k,γ we have the terms $\frac{\lambda_\Gamma}{\lambda_{comp}}$ and $\frac{\alpha}{1-\alpha}\frac{\eta_{k_0}^3\Gamma^3}{\lambda_{comp}}$ are upper bounded by $\mathcal{O}(\frac{1}{m^\nu})$ and higher order of $\mathcal{O}(\frac{1}{m^\nu})$.

We have

$$\frac{1}{(1-\beta)m}\left[\sum_{i\in\mathcal{M}}\|\eta_{k_0}\mathbf{s}_{i,k_0+1}\|^3+\alpha m\eta_{k_0}^3\Gamma^3\right]\leq\frac{(1-\alpha)}{(1-\beta)}\left[\frac{f(\mathbf{x}_0)-f^*}{T\lambda_{comp}}+\frac{\lambda_\Gamma}{\lambda_{comp}}+\frac{\alpha}{1-\alpha}\frac{\eta_{k_0}^3\Gamma^3}{\lambda_{comp}}\right]$$
$$\Rightarrow\frac{1}{(1-\alpha)m}\left[\sum_{i\in\mathcal{M}}\|\eta_{k_0}\mathbf{s}_{i,k_0+1}\|^3+\alpha m\eta_{k_0}^3\Gamma^3\right]\leq\left[\frac{f(\mathbf{x}_0)-f^*}{T\lambda_{comp}}+\frac{\lambda_\Gamma}{\lambda_{comp}}+\frac{\alpha}{1-\alpha}\frac{\eta_{k_0}^3\Gamma^3}{\lambda_{comp}}\right]$$
$$\Rightarrow\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_{k_0}\mathbf{s}_{i,k_0+1}\|^3\leq\left[\frac{f(\mathbf{x}_0)-f^*}{T\lambda_{comp}}\right]=\frac{\psi_{comp}}{T}+C_\Gamma$$

where $\psi_{comp}=\frac{f(\mathbf{x}_0)-f^*}{\lambda_{comp}}$ where C_Γ is $\mathcal{O}(1/m)$.

So, we have the term ψ_{comp} is of the order $\mathcal{O}(1)$.

The gradient condition is (using (13))

$$\begin{aligned}
&\|\nabla f(\mathbf{x}_{k+1})\|\\
=&\left\|\nabla f(\mathbf{x}_{k+1})-\frac{1}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\mathbf{g}_{i,k}-\frac{1}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\gamma\mathbf{H}_{i,k+1}\mathbf{s}_{i,k+1}-\frac{1}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\frac{M\gamma^2}{2}\|\mathbf{s}_{i,k+1}\|\mathbf{s}_{i,k+1}\right\|\\
\leq&\left\|\nabla f(\mathbf{x}_{k+1})-\nabla f(\mathbf{x}_k)-\nabla^2 f(\mathbf{x}_k)(x_{k+1}-x_k)\right\|+\left\|\frac{1}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}(\mathbf{g}_{i,k}-\nabla f(\mathbf{x}_k))\right\|\\
&+\left\|\nabla^2 f(\mathbf{x}_k)(x_{k+1}-x_k)-\gamma\frac{1}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}\right\|+\left\|\frac{1}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\frac{M\gamma^2}{2}\|\mathbf{s}_{i,k+1}\|\mathbf{s}_{i,k+1}\right\|\\
\leq&\frac{L_2\eta_k^2}{2}\left\|\frac{1}{|\mathcal{U}|}\sum_{i\in\mathcal{U}}\mathbf{s}_{i,k+1}\right\|^2+\epsilon_g+\frac{M\gamma^2}{2}\frac{1}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^2+\left\|\frac{\eta_k}{|\mathcal{U}|}\sum_{i\in\mathcal{U}}\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}-\frac{\gamma}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}\right\|
\end{aligned}\tag{52}$$

Now consider the term in (52)

$$\begin{aligned}
&\left\|\frac{\eta_k}{|\mathcal{U}|}\sum_{i\in\mathcal{U}}\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}-\frac{\gamma}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}\right\|\\
\leq&\left\|\frac{\eta_k}{|\mathcal{U}|}\left[\sum_{i\in\mathcal{M}}\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}-\sum_{i\in\mathcal{M}\cap\mathcal{T}}\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}+\sum_{i\in\mathcal{B}\cap\mathcal{U}}\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}\right]-\frac{\gamma}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}\right\|\\
\leq&\left\|\frac{\eta_k}{|\mathcal{U}|}\sum_{i\in\mathcal{M}}\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}-\frac{\gamma}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}\right\|+\frac{\eta_k}{|\mathcal{U}|}\sum_{i\in\mathcal{M}\cap\mathcal{T}}\|\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}\|\\
&+\frac{\eta_k}{|\mathcal{U}|}\sum_{i\in\mathcal{B}\cap\mathcal{U}}\|\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}\|\\
\leq&\left\|\frac{\eta_k}{|\mathcal{U}|}\sum_{i\in\mathcal{M}}\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}-\frac{\gamma}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}\right\|\\
&+\left\|\frac{\gamma}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\nabla^2 f(\mathbf{x}_k)\mathbf{s}_{i,k+1}-\frac{\gamma}{|\mathcal{M}|}\sum_{i\in\mathcal{M}}\mathbf{H}_{i,k}\mathbf{s}_{i,k+1}\right\|+\frac{\eta_k}{(1-\beta)m}L_1[\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|+\alpha m\Gamma]\\
\leq&(\frac{\eta_k}{(1-\beta)m}-\frac{\gamma}{(1-\alpha)m})L_1\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|\\
&+\frac{\gamma\epsilon_H}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|+\frac{\eta_k}{(1-\beta)m}L_1[\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|+\alpha m\Gamma]\\
\leq&\left(\frac{\eta_k}{(1-\beta)m}L_1(2+\alpha m)-\frac{\gamma L_1}{(1-\alpha)m}\right)\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|+\frac{\gamma\epsilon_H}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|\\
=&\left(\frac{(1-\alpha)}{(1-\beta)}2L_1-\frac{\gamma}{\eta_k}(L_1-\epsilon_H)\right)\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|+\frac{\eta_k\alpha}{(1-\beta)}\Gamma
\end{aligned}$$

Next we consider the term

$$\begin{aligned}
&\frac{L_2\eta_k^2}{2}\left\|\frac{1}{|\mathcal{U}|}\sum_{i\in\mathcal{U}}\mathbf{s}_{i,k+1}\right\|^2\\
\leq&\frac{L_2\eta_k^2}{2(1-\beta)m}\sum_{i\in\mathcal{U}}\|\mathbf{s}_{i,k+1}\|^2\\
\leq&\frac{L_2\eta_k^2}{2(1-\beta)m}\left[\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^2+\sum_{i\in\mathcal{U}\cap\mathcal{B}}\|\mathbf{s}_{i,k+1}\|^2\right]\\
=&\frac{L_2\eta_k^2}{2(1-\beta)m}\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^2+\frac{L_2\alpha\eta_k^2}{2(1-\beta)}\Gamma^2
\end{aligned}\tag{53}$$

So finally we have

$$\begin{aligned}
&\|\nabla f(\mathbf{x}_{k+1})\|\\
\leq&\frac{L_2\eta_k^2}{2(1-\beta)m}\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^2+\epsilon_g+\frac{M\gamma^2}{2(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\mathbf{s}_{i,k+1}\|^2+\left(\frac{(1-\alpha)}{(1-\beta)}2L_1-\frac{\gamma}{\eta_k}(L_1-\epsilon_H)\right)\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|\\
&+\frac{L_2\alpha\eta_k^2}{2(1-\beta)}\Gamma^2+\frac{\eta_k\alpha}{(1-\beta)}\Gamma
\end{aligned}$$

Now we choose $\gamma>\frac{(1-\alpha)}{(1-\beta)}2L_1\frac{\eta_k}{L_1-\epsilon_H}$.

$$\begin{aligned}
&\|\nabla f(\mathbf{x}_{k+1})\|\\
\leq&\left[\frac{L_2(1-\alpha)}{2(1-\beta)}+\frac{M\gamma^2}{2\eta_k^2}\right]\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|^2+\frac{L_2\alpha\eta_k^2}{2(1-\beta)}\Gamma^2+\frac{\eta_k\alpha}{(1-\beta)}\Gamma+\epsilon_g\\
\leq&\left[\frac{L_2(1-\alpha)}{2(1-\beta)}+\frac{M\gamma^2}{2\eta_k^2}\right]\left[\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|^3\right]^{2/3}+\frac{L_2\alpha\eta_k^2}{2(1-\beta)}\Gamma^2+\frac{\eta_k\alpha}{(1-\beta)}\Gamma+\epsilon_g
\end{aligned}$$

At step $k=k_0$,

$$\begin{aligned}
&\|\nabla f(\mathbf{x}_{k_0+1})\|\\
&\leq\left[\frac{L_2(1-\alpha)}{2(1-\beta)}+\frac{M\gamma^2}{2\eta_k^2}\right]\left(\frac{\psi_{comp}}{T}+C_\Gamma\right)^{2/3}+\epsilon_g+\frac{L_2\alpha\eta_k^2}{2(1-\beta)}\Gamma^2+\frac{\eta_k\alpha}{(1-\beta)}\Gamma\\
&\leq\left[\frac{L_2(1-\alpha)}{2(1-\beta)}+\frac{M\gamma^2}{2\eta_k^2}\right]\left(\frac{\psi_{comp}}{T}\right)^{2/3}+\epsilon_g\\
\leq&\left[\frac{L_2(1-\alpha)}{2(1-\beta)}+\frac{M\gamma^2}{2\eta_k^2}\right]C_\Gamma^{2/3}+\frac{L_2\alpha\eta_k^2}{2(1-\beta)}\Gamma^2+\frac{\eta_k\alpha}{(1-\beta)}\Gamma\\
&\leq\frac{\chi_1}{T^{2/3}}+\epsilon_g+\chi_G
\end{aligned}$$

where $\chi_1=\left[\frac{L_2(1-\alpha)}{2(1-\beta)}+\frac{M\gamma^2}{2\eta_k^2}\right](\psi_{comp})^{2/3}$. And as $C_\Gamma=\mathcal{O}(\frac{1}{m^\nu})$, we have $\chi_G=\mathcal{O}(\frac{1}{m^{2\nu/3-1}})+\mathcal{O}(\frac{\alpha}{m^\nu})$. As $\nu\geq 3$, χ_G is always decreasing with m.

The Hessian bound is

$$\begin{aligned}
&\lambda_{\min}(\nabla^2 f(\mathbf{x}_{k+1}))\\
=&\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\lambda_{\min}\left[\nabla^2 f(\mathbf{x}_{k+1})\right]\\
=&\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\lambda_{\min}\left[\mathbf{H}_{i,k}-(\mathbf{H}_{i,k}-\nabla^2 f(\mathbf{x}_{k+1}))\right]\\
\geq&\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\left[\lambda_{\min}(\mathbf{H}_{i,k})-\|\mathbf{H}_{i,k}-\nabla^2 f(\mathbf{x}_{k+1})\|\right]
\end{aligned}$$

$$
\begin{aligned}
&\geq \frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\lambda_{\min}(\mathbf{H}_{i,k})-\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\mathbf{H}_{i,k}-\nabla^2 f(\mathbf{x}_{k+1})\| \\
&\geq \frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}-\frac{M\gamma}{2}\|\mathbf{s}_{i,k+1}\|-\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\mathbf{H}_{i,k}-\nabla^2 f(\mathbf{x}_k)\| \\
&\quad -\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\nabla^2 f(\mathbf{x}_k)-\nabla^2 f(\mathbf{x}_{k+1})\| \\
&\geq \frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}-\frac{M\gamma}{2\eta_k}\|\eta_k\mathbf{s}_{i,k+1}\|-\epsilon_H-\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}L_2\|\mathbf{x}_k-\mathbf{x}_{k+1}\| \\
&\geq -\frac{M\gamma}{2\eta_k}\left[\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|^3\right]^{1/3}-L_2\|\mathbf{x}_k-\mathbf{x}_{k+1}\|-\epsilon_H \\
&\geq -\frac{M\gamma}{2\eta_k}\left[\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|^3\right]^{1/3}-L_2\left[\frac{1}{(1-\beta)m}\sum_{i\in\mathcal{U}}\|\eta_k\mathbf{s}_{i,k+1}\|\right]-\epsilon_H \\
&\geq -\frac{M\gamma}{2\eta_k}\left[\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|^3\right]^{1/3}-L_2\frac{1}{(1-\beta)m}\left[\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|+\sum_{i\in\mathcal{B}\cap\mathcal{U}}\|\eta_k\mathbf{s}_{i,k+1}\|\right]-\epsilon_H \\
&\geq -\frac{M\gamma}{2\eta_k}\left[\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|^3\right]^{1/3}-L_2\frac{(1-\alpha)}{(1-\beta)}\left[\frac{1}{(1-\alpha)m}\sum_{i\in\mathcal{M}}\|\eta_k\mathbf{s}_{i,k+1}\|\right]-\epsilon_H-L_2\frac{\alpha}{(1-\beta)}\eta_k\Gamma \qquad (54)
\end{aligned}
$$

At $k=k_0$ we have

$$
\begin{aligned}
&\lambda_{\min}(\nabla^2 f(\mathbf{x}_{k_0+1})) \qquad (55)\\
&\geq -\frac{M\gamma}{2\eta_k}\left[\frac{\psi_{comp}}{T}+C_\Gamma\right]^{1/3}-L_2\frac{(1-\alpha)}{(1-\beta)}\left[\frac{\psi_{comp}}{T}+C_\Gamma\right]^{1/3}-\epsilon_H-L_2\frac{\alpha}{(1-\beta)}\eta_k\Gamma \\
&\geq -\left[\frac{M\gamma}{2\eta_k}+L_2\frac{(1-\alpha)}{(1-\beta)}\right]\psi_{comp}^{1/3}\left(\frac{1}{T}\right)^{1/3}-\epsilon_H-\left(\frac{M\gamma}{2\eta_k}C_\Gamma^{1/3}+L_2\frac{(1-\alpha)}{(1-\beta)}C_\Gamma^{1/3}\right)-L_2\frac{\alpha}{(1-\beta)}\eta_k\Gamma \\
&\geq -\frac{\chi_2}{T^{1/3}}-\epsilon_H-\chi_H \qquad (56)
\end{aligned}
$$

where $\chi_2=\left[\frac{M\gamma}{2\eta_k}+L_2\frac{(1-\alpha)}{(1-\beta)}\right]\psi_{comp}^{1/3}$. And , we have $\chi_H=\mathcal{O}(\frac{1}{m^{\nu/3-1}})+\mathcal{O}(\frac{1}{m^\nu})$. As $\nu\geq 3$, we χ_H to be strictly decreasing with m.

Finally, we restate the Theorem 1,

Theorem 1 (Convergence of `DIST-CURE`*):* Suppose $0\leq\alpha<\beta\leq\frac{1}{2}$. Furthermore, we choose the problem parameters, $M=\mathcal{O}(m(1-\beta)$, and $\eta=\gamma=\frac{c}{Tm^\nu}$ for some constant $c>0,\nu>3$. Then, after T iterations of `DIST-CURE` (Algorithm 1), the sequence $\{\mathbf{x}_i\}_{i=1}^T$ generated contains a point $\tilde{x}$ such that

$$
\|\nabla f(\tilde{x})\|\leq\frac{\chi_1}{T^{2/3}}+\epsilon_g+\chi_G,\quad \lambda_{\min}\big(\nabla^2 f(\tilde{x})\big)\geq-\frac{\chi_2}{T^{\frac{1}{3}}}-\epsilon_H-\chi_H,\quad \text{where,}
$$

$$
\begin{aligned}
\chi_1&=\left[\frac{L_2(1-\alpha)}{2(1-\beta)}+\frac{M\gamma^2}{2\eta_k^2}\right](\psi_{comp})^{2/3}\\
\chi_2&=\left[\frac{M\gamma}{2\eta_k}+L_2\frac{1(1-\alpha)}{(1-\beta)}\right]\psi_{comp}^{1/3}\\
\chi_G&=\left[\frac{L_2(1-\alpha)}{2(1-\beta)}+\frac{M\gamma^2}{2\eta_k^2}\right]C_\Gamma^{2/3}+\frac{L_2\eta_k^2}{2(1-\beta)}\Gamma^2+\frac{\eta_k\alpha}{(1-\beta)}\Gamma\\
\chi_H&=\left(\frac{M\gamma}{2\eta_k}+L_2\frac{(1-\alpha)}{(1-\beta)}\right)C_\Gamma^{1/3}+L_2\frac{\alpha}{(1-\beta)}\eta_k\Gamma\\
\psi_{comp}&=\frac{f(\mathbf{x}_0)-f^*}{\lambda_{comp}}\text{and }C_\Gamma=\frac{\lambda_\Gamma}{\lambda_{comp}}
\end{aligned}
$$

$$\lambda_{comp}=[\frac{\gamma M}{4(1-\beta)\eta_k m^2}-\frac{L_2}{6(1-\beta)m}](1-\alpha)m$$

$$\lambda_\Gamma=\left(\frac{\eta_k(1-\alpha)}{(1-\beta)}\epsilon_g+\frac{\eta_k(\alpha+\beta)L}{(1-\beta)}\right)\Gamma+\frac{L_2\eta_k^3}{6(1-\beta)}\alpha\Gamma^3+\frac{\eta_k^2}{2(1-\beta)^2m}((1-\alpha)\epsilon_H+(\alpha+\beta)L_1+L_1((1-\beta)m-1)(1-\beta)m)\Gamma^2.$$

For the choice of $\eta=\frac{c}{Tm^\nu}$ and $\gamma=\frac{c}{Tm^\nu}$ and $M=\mathcal{O}(m(1-\beta)$, we have $\lambda_\Gamma=\mathcal{O}(\frac{1}{m^\nu})$ and λ_{comp} to be $\mathcal{O}(1)$.

E. Proof of Corollary 1.1

In this Corollary statement we consider centralized $(m=1)$, uncompressed $(\delta=1)$ and non-Byzantine setup $(\alpha=\beta=0)$. With these parameters, we have the value of λ_Γ from equation (48) to be 0. Consequently, we have $C_\Gamma=0$. With $\gamma=1$, we have

$$\lambda_{comp}=\frac{M}{4\eta_k}-\frac{L_2}{6}$$

So in order for $\lambda_{comp}>0$, for constant step-size $(\eta_k=1)$, we need $M>\frac{2L_2}{3}$. With $C_\Gamma=0,\alpha=0$, we have $\chi_G=\chi_H=0$. Moreover we have $\chi_1=\left[\frac{L_2+M}{2}\right](\psi_{comp})^{2/3}$ and $\chi_2=\left[\frac{2M+L_2}{2}\right](\psi_{comp})^{1/3}$. As it is a centralized setup, there are no gradient and Hessian dissimilarities $\epsilon_g=\epsilon_H=0$. So we have

$$\|\nabla f(\tilde{x})\|\leq\left[\frac{L_2+M}{2}\right](\psi_{comp})^{2/3}\frac{1}{T^{2/3}},\quad \lambda_{\min}\big(\nabla^2 f(\tilde{x})\big)\geq-\left[\frac{2M+L_2}{2}\right](\psi_{comp})^{1/3}\frac{1}{T^{1/3}},$$

where $M>\frac{2L_2}{3}$. Thus, the convergence rate of `DIST-CURE` reduces to that of [1].