Distributed Newton-Type Methods with Communication Compression and Bernoulli Aggregation

Anonymous authors
Paper under double-blind review

Abstract

Despite their high computation and communication costs, Newton-type methods remain an appealing option for distributed training due to their robustness against ill-conditioned convex problems. In this work, we study *communication compression* and *aggregation mechanisms* for curvature information in order to reduce these costs while preserving theoretically superior local convergence guarantees. We prove that the recently developed class of *three point compressors (3PC)* of (Richtárik et al., 2022) for gradient communication can be generalized to Hessian communication as well. This result opens up a wide variety of communication strategies, such as *contractive compression* and *lazy aggregation*, available to our disposal to compress prohibitively costly curvature information. Moreover, we discovered several new 3PC mechanisms, such as *adaptive thresholding* and *Bernoulli aggregation*, which require reduced communication and occasional Hessian computations. Furthermore, we extend and analyze our approach to bidirectional communication compression and partial device participation setups to cater to the practical considerations of applications in federated learning. For all our methods, we derive fast *condition-number-independent* local linear and/or superlinear convergence rates. Finally, with extensive numerical evaluations on convex optimization problems, we illustrate that our designed schemes achieve state-of-the-art communication complexity compared to several key baselines using second-order information.

1 Introduction

In this work we consider the distributed optimization problem given by the form of ERM:

$$\min_{x\in\mathbb{R}^d} \left\{ f(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x) \right\}, \tag{1}$$

where d is the (potentially large) number of parameters of the model $x \in \mathbb{R}^d$ we aim to train, n is the (potentially large) number of devices in the distributed system, $f_i(x)$ is the loss/risk associated with the data stored on machine $i \in [n] := \{1, 2, \ldots, n\}$, and $f(x)$ is the empirical loss/risk.

In order to jointly train a single machine learning model using all devices' local data, *collective efforts* are necessary from all compute nodes. Informally, each entity should invest some "knowledge" from its local "wisdom" to create the global "wisdom". The classical approach in distributed training to implement the collective efforts was to literally collect all the raw data devices acquired and then perform the training in one place with traditional methods. However, the mere access to the raw data hinders the clients' *data privacy* in federated learning applications (Konečný et al., 2016b;a; McMahan et al., 2017). Besides, even if we ignore the privacy aspect, accumulating all devices' data into a single machine is often *infeasible* due to its increasingly large size (Bekkerman et al., 2011).

Because of these considerations, there has been a serious stream of works studying distributed training with decentralized data. This paradigm of training brings its own advantages and limitations. Perhaps the major advantage is that each remote device's data can be processed simultaneously using *local computational* *resources.* Thus, from another perspective, we are scaling up the traditional single-device training to a distributed training of multiple parallel devices with decentralized data and local computation. However, the cost of scaling the training over multiple devices forces *intensive communication* between nodes, which is the *key bottleneck* in distributed systems.

1.1 Related work: from first-order to second-order distributed optimization

Currently, first-order optimization methods are the default options for large-scale distributed training due to their cheap per-iteration costs. Tremendous amount of work has been devoted to extend and analyze gradient-type algorithms to conform to various practical constraints such as efficient communication through compression mechanisms (Alistarh et al., 2017; 2018b; Wen et al., 2017; Wangni et al., 2018; Sahu et al., 2021; Tyurin & Richtárik, 2022) and local methods (Gorbunov et al., 2021b; Stich, 2020; Karimireddy et al., 2020; Nadiradze et al., 2021a; Mishchenko et al., 2022), peer-to-peer communication through graphs (Koloskova et al., 2019; 2020; Kovalev et al., 2021), asynchronous communication (Feyzmahdavian & Johansson, 2021; Nadiradze et al., 2021b), partial device participation (Yang et al., 2021), Byzantine or adversarial attacks (Karimireddy et al., 2021; 2022), faster convergence through acceleration (Allen-Zhu, 2017; Li et al., 2020b; Qian et al., 2021) and variance reduction techniques (Lee et al., 2017; Mishchenko et al., 2019; Horváth et al., 2019; Cen et al., 2020; Gorbunov et al., 2021a), data privacy and heterogeneity over the nodes (Kairouz et al, 2019; Li et al., 2020a),

Nevertheless, despite their wide applicability, all first-order methods (including accelerated ones) inevitably suffer from ill-conditioning of the problem. In the past few years, several algorithmic ideas and mechanisms to tackle the above-mentioned constraints have been adapted for second-order optimization. The goal in this direction is to enhance the convergence by increasing the resistance of gradient-type methods against ill-conditioning using the knowledge of curvature information. The basic motivation that the Hessian computation will be useful in optimization is the fast *condition-number-independent* (local) convergence rate of classic Newton's method (Beck, 2014), that is beyond the reach of *all* first-order methods.

Because of the quadratic dependence of Hessian information (d^2 floats per each Hessian matrix) from the dimensionality of the problem, the primary challenge of taming second-order methods was efficient communication between the participating devices. To alleviate prohibitively costly Hessian communication, many works such as DiSCO (Zhang & Xiao, 2015; Zhuang et al., 2015; Lin et al., 2014; Roosta et al., 2019), GIANT (Wang et al., 2018; Shamir et al., 2014; Reddi et al., 2016) and DINGO (Crane & Roosta, 2019; Ghosh et al., 2020b) impart second-order information by condensing it into Hessian-vector products. Inspired from compressed first-order methods, an orthogonal line of work, including DAN-LA (Zhang et al., 2020b), Quantized Newton (Alimisis et al., 2021), NewtonLearn (Islamov et al., 2021), FedNL (Safaryan et al., 2022), Basis Learn (Qian et al., 2022) and IOS (Fabbro et al., 2022), applies lossy compression strategies directly to Hessian matrices reducing the number of encoding bits. Other techniques that have been migrated from first-order optimization literature are local methods (Gupta et al., 2021), partial device participation (Safaryan et al., 2022; Qian et al., 2022), defenses against Byzantine attacks (Ghosh et al., 2020a;c). The theoretical comparison of second order methods is presented in Table 1. We defer more detailed review of a literature of second-order methods to the Appendix.

2 Motivation and Contributions

Handling and taking advantage of the second-order information in distributed setup is rather challenging. As opposed to gradient-type methods, Hessian matrices are both harder to compute and much more expensive to communicate. To avoid directly accessing costly Hessian matrices, methods like DiSCO (Zhang & Xiao, 2015), GIANT (Wang et al., 2018) and DINGO (Crane & Roosta, 2019) exploit Hessian-vector products only, which are as cheap to compute as gradients (Pearlmutter, 1994). However, these methods typically suffer from data heterogeneity, need strong assumptions on problem structure (e.g., generalized linear models) and/or do not provide fast local convergence rates.

On the other hand, recent works (Safaryan et al., 2022; Qian et al., 2022) have shown that, with the access of Hessian matrices, fast local rates can be guaranteed for solving general finite sums (1) under compressed

Table 1: Theoretical comparison of several second order methods (including ours) in strongly convex setup with Lipschitz continuous Hessians. Advantages are written in green, while limitations are colored in red.

Method	LipC grad[1]	Comm. Cost[2]	Comp. Cost[3]	Rate	Comments
GIANT[4] (Wang et al., 2018)	No	$\mathcal{O}(d)$	Full Hessian	Local κ-dependent linear. Global $\mathcal{O}(\log {}^{\kappa}/_{\epsilon})$, quadratics	Big data regime (#data $\gg d$)
DINGO[5,6] (Crane & Roosta, 2019)	No	$\mathcal{O}(d)$	Hessian-vector products	Global linear, but no fast local	Also requires Hessian pseudo-inverse and vector products.
DAN (Zhang et al., 2020b)	No	$\mathcal{O}(nd^2)$	Full Hessian	Global quadratic rate after $\mathcal{O}({}^{L}/_{\mu^2})$ iterations.	-
DAN-LA (Zhang et al., 2020b)	Yes	$\mathcal{O}(nd)$	Full Hessian	Asymptotic and implicit global superlinear rate.	$\lim_{k\to\infty} \frac{\|x_{k+1}-x^*\|}{\|x_k-x^*\|} = 0$ Independent of κ ? Better non-asymptotic complexity over linear rate ?
NL[4] (Islamov et al., 2021)	No	$\mathcal{O}(d)$	Full Hessian	Local linear and superlinear independent of κ, but dependent on #data. global linear	reveals local data to server
Quantized Newton[6] (Alimisis et al., 2021)	Yes	$\widetilde{\mathcal{O}}(d^2)$	Full Hessian	Local (fixed) linear without global	-
FedNL (Safaryan et al., 2022)	No	$\mathcal{O}(d)$	Full Hessian	Local (fixed) linear and superlinear, independent of κ and #data Global linear	Supports contractive Hessian compression, Bidirectional compression.
BL (Qian et al., 2022)	No	$\mathcal{O}(d)$	Full Hessian	Local (fixed) linear and superlinear, independent of κ and #data Global linear	Supports contractive Hessian compression, Bidirectional compression. Exploits lower intrinsic dimensionality of data.
Fib-IOS[6] (Fabbro et al., 2022)	Yes	$\mathcal{O}(d)$	Periodic Full Hessian	implicit global linear	Only rank-type compression. Backtracking line search. SVD in each round.
FLECS (Agafonov et al., 2022b)	Yes	$\mathcal{O}(d)$	Hessian-vector products	Implicit global linear, but no fast local[7]	Backtracking line search. SVD in each round.
Newton-3PC (this work)	No	$\mathcal{O}(d)$	Periodic Full Hessian and/or Hessian-vector products	Local (fixed) linear and superlinear, independent of κ, independent of #data. Global rate[8]	Supports contractive Hessian compression, Bidirectional compression.

[1] **LipC grad** = **Lip**schitz **C**ontinuous **grad**ients.
[2] **Comm. Cost** = **Comm**unication **Cost** per round. [3] **Comp. Cost** = **Comp**utation **Cost** per round.
[4] Only for Generalized Linear Models, e.g. $loss_j(x; a_j) = \phi_j(a_j^\top x) + \lambda\|x\|^2$.
[5] Uses Moral Smoothness: $\|\nabla^2 f(x)\nabla f(x) - \nabla^2 f(y)\nabla f(y)\| \leq L\|x - y\|$. [6] Strongly convex local loss functions for all clients.
[7] FLECS has a local rate under the condition that all iterates remain within some fixed neighborhood of the optimum.
[8] See Section I in the Appendix for globalization strategies.

communication and arbitrary heterogeneous data. In view of these advantages, in this work we adhere to this approach and study communication mechanisms that can further lighten communication and reduce computation costs. Below, we summarize our key contributions.

2.1 Flexible communication strategies for Newton-type methods

We prove that the recently developed class of *three point compressors (3PC)* of Richtárik et al. (2022) for gradient communication can be generalized to Hessian communication as well. In particular, we propose a new method, which we call Newton-3PC (Algorithm 1), extending FedNL (Safaryan et al., 2022) algorithm for arbitrary 3PC mechanism. This result opens up a wide variety of communication strategies, such as *contractive compression* (Stich et al., 2018; Alistarh et al., 2018a; Karimireddy et al., 2019) and *lazy aggregation* (Chen et al., 2018; Sun et al., 2019; Ghadikolaei et al., 2021), available to our disposal to compress prohibitively costly curvature information. Besides, Newton-3PC (and its local convergence theory) recovers FedNL (Safaryan et al., 2022) (when *contractive compressors* are used as 3PC) and BL (Qian et al., 2022) (when *rotation compression* is used as 3PC) in special cases.

2.2 New compression and aggregation schemes

Moreover, we discovered several new 3PC mechanisms, which require reduced communication and occasional Hessian computations. In particular, to reduce communication costs, we design an *adaptive thresholding* (Example 3.3) that can be seamlessly combined with an already adaptive *lazy aggregation* (Example 3.7). In order to reduce computation costs, we propose *Bernoulli aggregation* (Example 3.9) mechanism which allows local workers to *skip* both computation and communication of local information (e.g., Hessian and gradient) with some predefined probability. Moreover, *sketch-and-project* operator (Example 3.5) reduces the computation costs relying on Hessian-vector products.

2.3 Extensions

Furthermore, we provide several extensions to our approach to cater to the practical considerations of applications in federated learning. In the main part of the paper, we consider only bidirectional communication compression (Newton-3PC-BC) setup, where we additionally apply Bernoulli aggregation for gradients (worker to server direction) and another 3PC mechanism for the global model (server to worker direction). The extension for partial device participation (Newton-3PC-BC-PP) setup and the discussion for globalization are deferred to the Appendix.

2.4 Fast local linear/superlinear rates

All our methods are analyzed under the assumption that the global objective is strongly convex and local Hessians are Lipschitz continuous. In this setting, we derive fast *condition-number-independent* local linear and/or superlinear convergence rates.

2.5 Extensive experiments and Numerical Study

Finally, with extensive numerical evaluations on convex optimization problems, we illustrate that our designed schemes achieve state-of-the-art communication complexity compared to several key baselines using second-order information.

3 Three Point Compressors for Matrices

To properly incorporate second-order information in distributed training, we need to design an efficient strategy to synchronize locally evaluated $d \times d$ Hessian matrices. Simply transferring d^2 entries of the matrix each time it gets computed would put significant burden on communication links of the system. Recently, Richtárik et al. (2022) proposed a new class of gradient communication mechanisms under the name *three point compressors (3PC)*, which unifies contractive compression and lazy aggregation mechanisms into one class. Here we extend the definition of 3PC for matrices under the Frobenius norm $\|\cdot\|_{\mathrm{F}}$ and later apply to matrices involving Hessians.

Definition 3.1 (3PC for Matrices)**.** *We say that a (possibly randomized) map*

$$\mathcal{C}_{\mathbf{H},\mathbf{Y}}(\mathbf{X}) : \underbrace{\mathbb{R}^{d\times d}}_{\mathbf{H}\in} \times \underbrace{\mathbb{R}^{d\times d}}_{\mathbf{Y}\in} \times \underbrace{\mathbb{R}^{d\times d}}_{\mathbf{X}\in} \to \mathbb{R}^{d\times d} \tag{2}$$

is a three point compressor (3PC) if there exist constants $0 < A \leq 1$ and $B \geq 0$ such that

$$\mathbb{E}\left[\|\mathcal{C}_{\mathbf{H},\mathbf{Y}}(\mathbf{X}) - \mathbf{X}\|^2_{\mathrm{F}}\right] \quad \leq \quad (1-A)\|\mathbf{H}-\mathbf{Y}\|^2_{\mathrm{F}} + B\|\mathbf{X}-\mathbf{Y}\|^2_{\mathrm{F}}. \tag{3}$$

holds for all matrices $\mathbf{H}, \mathbf{Y}, \mathbf{X} \in \mathbb{R}^{d\times d}$.

The matrices $\mathbf{Y}$ and $\mathbf{H}$ can be treated as parameters defining the compressor that would be chosen adaptively. Once they fixed, $\mathcal{C}_{\mathbf{H},\mathbf{Y}} : \mathbb{R}^{d\times d} \to \mathbb{R}^{d\times d}$ is a map to compress a given matrix $\mathbf{X}$. Let us discuss special cases with some examples.

Example 3.2 (**Contractive compressors** (Karimireddy et al., 2019))**.** *The (possibly randomized) map $\mathcal{C} \colon \mathbb{R}^d \to \mathbb{R}^d$ is called contractive compressor with contraction parameter $\alpha \in (0,1]$, if the following holds for any matrix $\mathbf{X} \in \mathbb{R}^{d\times d}$*

$$\mathbb{E}\left[\|\mathcal{C}(\mathbf{X}) - \mathbf{X}\|^2_{\mathrm{F}}\right] \leq (1-\alpha)\|\mathbf{X}\|^2_{\mathrm{F}}. \tag{4}$$

Notice that (4) is a special case of (2) when $\mathbf{H} = \mathbf{0}$, $\mathbf{Y} = \mathbf{X}$ and $A = \alpha$, $B = 0$. Therefore, contractive compressors are already included in the 3PC class. Contractive compressors cover various well known compression schemes such as greedy sparsification, low-rank approximation and (with a suitable scaling factor) arbitrary unbiased compression operator (Beznosikov et al., 2020). There have been several recent works utilizing these compressors for compressing Hessian matrices (Zhang et al., 2020b; Alimisis et al., 2021; Islamov et al., 2021; Safaryan et al., 2022; Qian et al., 2022; Fabbro et al., 2022). Below, we introduce yet another contractive compressor based on thresholding idea which shows promising performance in our experiments.

Example 3.3 (**Adaptive Thresholding [NEW]**). *Following Sahu et al. (2021), we design an* adaptive thresholding *operator with parameter* $\lambda \in (0, 1]$ *defined as follows; for all* $j, l \in [d]$ *and* $\mathbf{X} \in \mathbb{R}^{d\times d}$

$$[\mathcal{C}(\mathbf{X})]_{jl} := \begin{cases} \mathbf{X}_{jl} & \text{if } |\mathbf{X}_{jl}| \geq \lambda \|\mathbf{X}\|_\infty, \\ 0 & \text{otherwise}, \end{cases} \tag{5}$$

In contrast to *hard thresholding* operator of Sahu et al. (2021), (5) uses adaptive threshold $\lambda\|\mathbf{X}\|_\infty$ instead of fixed threshold λ. With this choice, we ensures that at least the Top-1 is transferred. In terms of computation, thresholding approach is more efficient than Top-K as only single pass over the values is already enough instead of partial sorting.

Lemma 3.4. *The adaptive thresholding operator* (5) *is contractive with* $\alpha = \max(1 - (d\lambda)^2, 1/d^2)$.

A popular technique to decrease computation cost of Newton-type methods is to rely on Hessian-vector products. We show that so called *sketch-and-project* mechanism (Gower & Richtárik, 2017) is a special case of contractive compressor.

Example 3.5 (**Sketch-and-Project** (Gower & Richtárik, 2017)). *Let* $\mathbf{S}$ *be a sketching matrix sampled from a fixed distribution* $\mathcal{D}$ *over matrices in* $\mathbb{R}^{d\times\tau}$ *(*$\tau \geq 1$ *can but does not need to be fixed). We define sketch-and-project operator as follows*

$$\mathcal{C}(\mathbf{X}) = \mathbf{S}(\mathbf{S}^\top\mathbf{S})^\dagger\mathbf{S}^\top\mathbf{X}. \tag{6}$$

For more details on sketch-and-project operator we refer a reader to the Appendix.

Lemma 3.6. *The sketch-and-project operator* (6) *is a contractive compressor with* $\alpha = \lambda^+_{\min}(\mathbb{E}\left[\mathbf{S}(\mathbf{S}^\top\mathbf{S})^\dagger\mathbf{S}^\top\right])$ *where the expectation is taken w.r.t. randomness of the sketching* $\mathbf{S}$*, and* $\lambda^+_{\min}(\mathbf{M})$ *indicates the smallest positive eigenvalue of a symmetric matrix* $\mathbf{M}$*.*

The next two examples are 3PC schemes which in addition to contractive compressors utilize aggregation mechanisms, which is an orthogonal approach to contractive compressors.

Example 3.7 (**Compressed Lazy AGgregation (CLAG)** (Richtárik et al., 2022)). *Let* $\mathcal{C}\colon \mathbb{R}^d \to \mathbb{R}^d$ *be a contractive compressor with contraction parameter* $\alpha \in (0, 1]$ *and* $\zeta \geq 0$ *be a trigger for the aggregation. Then CLAG mechanism is defined as*

$$\mathcal{C}_{\mathbf{H},\mathbf{Y}}(\mathbf{X}) = \begin{cases} \mathbf{H} + \mathcal{C}(\mathbf{X} - \mathbf{H}) & \text{if } \|\mathbf{X} - \mathbf{H}\|^2_{\mathrm{F}} > \zeta\|\mathbf{X} - \mathbf{Y}\|^2_{\mathrm{F}} \\ \mathbf{H} & \text{otherwise} \end{cases} \tag{7}$$

In the special case of identity compressor $\mathcal{C} = \mathrm{Id}$ (i.e., $\alpha = 1$), CLAG reduces to lazy aggregation (Chen et al., 2018). On the other extreme, if the trigger $\zeta = 0$ is trivial, CLAG recovers recent variant of error feedback for contractive compressors, i.e., EF21 mechanism (Richtárik et al., 2021).

Lemma 3.8 (see Lemma 4.3 in (Richtárik et al., 2022)). *CLAG mechanism* (7) *is a 3PC compressor with* $A = 1 - (1 - \alpha)(1 + s)$ *and* $B = \max\{(1 - \alpha)(1 + 1/s), \zeta\}$*, for any* $s \in (0, \alpha/(1-\alpha))$*.*

From the first glance, the structure of CLAG in (7) may not seem communication efficient as the the matrix $\mathbf{H}$ (appearing in both cases) can potentially by dense. However, as we will see in the next section, $\mathcal{C}_{\mathbf{H},\mathbf{Y}}$ is used to compress $\mathbf{X}$ when there is no need to communicate $\mathbf{H}$. Thus, with CLAG we either send compressed matrix $\mathcal{C}(\mathbf{X} - \mathbf{H})$ if the condition with trigger ζ activates or nothing.

Example 3.9 **(Compressed Bernoulli AGgregation (CBAG) [NEW]).** *Let $\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^d$ be a contractive compressor with contraction parameter $\alpha \in (0, 1]$ and $p \in (0, 1]$ be the probability for the aggregation. We then define CBAG mechanism is defined as*

$$\mathcal{C}_{\mathbf{H},\mathbf{Y}}(\mathbf{X}) = \begin{cases} \mathbf{H} + \mathcal{C}(\mathbf{X} - \mathbf{H}) & \textit{with prob. } p, \\ \mathbf{H} & \textit{with prob. } 1 - p. \end{cases} \tag{8}$$

The advantage of CBAG (8) over CLAG is that there is no condition to evaluate and check. This choice of probabilistic switching reduces computation costs as with probability $1 - p$ it is useless to compute $\mathbf{X}$. Note that CBAG has two independent sources of randomness: Bernoulli aggregation and possibly random operator $\mathcal{C}$.

Lemma 3.10. *CBAG mechanism* (8) *is a 3PC compressor with $A = (1 - p\alpha)(1 + s)$ and $B = (1 - p\alpha)(1 + 1/s)$, for any $s \in (0, p\alpha/(1 - p\alpha))$.*

For more examples of 3PC compressors see section C of Richtárik et al. (2022) and the Appendix.

4 Newton-3PC: Newton's Method with 3PC

In this section we present our first Newton-type method, called Newton-3PC, employing communication compression through 3PC compressors discussed in the previous section. The proposed method is an extension of FedNL (Safaryan et al., 2022) from contractive compressors to arbitrary 3PC compressors. From this perspective, our Newton-3PC (see Algorithm 1) is much more flexible, offering a wide variety of communication strategies beyond contractive compressors.

4.1 General technique for learning the Hessian

The central notion in FedNL is the technique for learning *a priori unknown* Hessian $\nabla^2 f(x^*)$ at the (unique) solution x^* in a communication efficient manner. This is achieved by maintaining and iteratively updating local Hessian estimates $\mathbf{H}_i^k$ of $\nabla^2 f_i(x^*)$ for all devices $i \in [n]$ and the global Hessian estimate $\mathbf{H}^k = \frac{1}{n}\sum_{i=1}^n \mathbf{H}_i^k$ of $\nabla^2 f(x^*)$ for the central server. We adopt the same idea of Hessian learning and aim to update local estimates in such a way that $\mathbf{H}_i^k \to \nabla^2 f_i(x^*)$ for all $i \in [n]$, and as a consequence, $\mathbf{H}^k \to \nabla^2 f(x^*)$, throughout the training process. However, in contrast to FedNL, we update local Hessian estimates via generic 3PC mechanism, namely

$$\mathbf{H}_i^{k+1} = \mathcal{C}_{\mathbf{H}_i^k, \nabla^2 f_i(x^k)}\left(\nabla^2 f_i(x^{k+1})\right),$$

which is a particular instantiation of 3PC compressor $\mathcal{C}_{\mathbf{H},\mathbf{Y}}(\mathbf{X})$ using previous local Hessian $\mathbf{Y} = \nabla^2 f_i(x^k)$ and previous estimate $\mathbf{H} = \mathbf{H}_i^k$ to compress current local Hessian $\mathbf{X} = \nabla^2 f_i(x^{k+1})$.

Algorithm 1 Newton-3PC (Newton's method with 3PC)

Input: $x^0 \in \mathbb{R}^d,\ \mathbf{H}_1^0, \dots, \mathbf{H}_n^0 \in \mathbb{R}^{d\times d},\ \mathbf{H}^0 := \frac{1}{n}\sum_{i=1}^n \mathbf{H}_i^0, l^0 = \frac{1}{n}\sum_{i=1}^n \|\mathbf{H}_i^0 - \nabla^2 f_i(x^0)\|$.
on server
  Option 1: $x^{k+1} = x^k - [\mathbf{H}^k]_\mu^{-1}\nabla f(x^k)$
  Option 2: $x^{k+1} = x^k - [\mathbf{H}^k + l^k\mathbf{I}]^{-1}\nabla f(x^k)$
  Broadcast x^{k+1} to all nodes
for each device $i = 1, \dots, n$ in parallel **do**
  Get x^{k+1} and compute local gradient $\nabla f_i(x^{k+1})$ and local Hessian $\nabla^2 f_i(x^{k+1})$
  Apply 3PC and update local Hessian estimator to $\mathbf{H}_i^{k+1} = \mathcal{C}_{\mathbf{H}_i^k, \nabla^2 f_i(x^k)}\left(\nabla^2 f_i(x^{k+1})\right)$
  Send $\nabla f_i(x^{k+1}),\ \mathbf{H}_i^{k+1}$ to the server
  Option 2: Send $l_i^{k+1} := \|\mathbf{H}_i^{k+1} - \nabla^2 f_i(x^{k+1})\|_{\mathrm{F}}$
end for
on server
  $\mathbf{H}^{k+1} = \frac{1}{n}\sum_{i=1}^n \mathbf{H}_i^{k+1},\ l^{k+1} = \frac{1}{n}\sum_{i=1}^n l_i^{k+1}$

In the special case, when EF21 scheme $\mathcal{C}_{\mathbf{H}_i^k, \nabla^2 f_i(x^k)}\left(\nabla^2 f_i(x^{k+1})\right) = \mathbf{H}_i^k + \mathcal{C}(\nabla^2 f_i(x^{k+1}) - \mathbf{H}_i^k)$ is employed as a 3PC mechanism, we recover the Hessian learning technique of FedNL. Our Newton-3PC method also recovers recently proposed *Basis Learn* (BL) (Qian et al., 2022) algorithm if we specialize the 3PC mechanism to *rotation compression* (see Appendix).

4.2 Flexible Hessian communication and computation schemes.

The key novelty Newton-3PC brings is the flexibility of options to handle costly local Hessian matrices both in terms of computation and communication.

Due to the adaptive nature of CLAG mechanism (7), Newton-CLAG method *does not send any information* about the local Hessian $\nabla^2 f_i(x^{k+1})$ if it is sufficiently close to previous Hessian estimate $\mathbf{H}_i^k$, namely $\|\nabla^2 f_i(x^{k+1}) - \mathbf{H}_i^k\|_{\mathrm{F}}^2 \leq \zeta \|\nabla^2 f_i(x^{k+1}) - \nabla^2 f_i(x^k)\|_{\mathrm{F}}^2$ with some positive trigger $\zeta > 0$. In other words, the server *reuses* local Hessian estimate $\mathbf{H}_i^k$ while there is no essential discrepancy between locally computed Hessian $\nabla^2 f_i(x^{k+1})$. Once a sufficient change is detected by the device, only the compressed difference $\mathcal{C}(\nabla^2 f_i(x^{k+1}) - \mathbf{H}_i^k)$ is communicated since the server knows $\mathbf{H}_i^k$. By adjusting the trigger ζ, we can control the frequency of Hessian communication in an adaptive manner. Together with adaptive thresholding operator (5) as a contractive compressor, CLAG is a doubly adaptive communication strategy that makes Newton-CLAG highly efficient in terms of communication complexity.

Interestingly enough, we can design such 3PC compressors that can reduce computational costs too. To achieve this, we consider CBAG mechanism (8) which replaces the adaptive switching condition of CLAG by probabilistic switching according to Bernoulli random variable. Due to the probabilistic nature of CBAG mechanism, Newton-CBAG method requires devices to compute local Hessian $\nabla^2 f_i(x^{k+1})$ and communicate compressed difference $\mathcal{C}(\nabla^2 f_i(x^{k+1}) - \mathbf{H}_i^k)$ *only* with probability $p \in (0, 1]$. Otherwise, the whole Hessian computation and communication is *skipped*.

4.3 Options for updating the global model

We adopt the same two update rules for the global model as was design in FedNL. If the server knows the strong convexity parameter $\mu > 0$ (see Assumption 4.1), then the global Hessian estimate $\mathbf{H}^k$ is projected onto the set $\left\{\mathbf{M} \in \mathbb{R}^{d \times d} \colon \mathbf{M}^\top = \mathbf{M},\ \mu \mathbf{I} \preceq \mathbf{M}\right\}$ to get the projected estimate $[\mathbf{H}^k]_\mu$. Alternatively, all devices additionally compute and send compression errors $l_i^k := \|\mathbf{H}_i^k - \nabla^2 f_i(x^k)\|_{\mathrm{F}}$ (extra float from each device in terms of communication complexity) to the server, which then formulates the regularized estimate $\mathbf{H}^k + l^k \mathbf{I}$ by adding the average error $l^k = \frac{1}{n}\sum_{i=1}^n l_i^k$ to the global Hessian estimate $\mathbf{H}^k$.

4.4 Local convergence theory

To derive theoretical guarantees, we consider the standard assumption that the global objective is strongly convex and local Hessians are Lipschitz continuous.

Assumption 4.1. *The average loss f is μ-strongly convex, and all local losses $f_i(x)$ have Lipschitz continuous Hessians with respect to three different matrix norms: spectral, Frobenius and infinity norms, respectively. Formally, we require $\|\nabla^2 f_i(x) - \nabla^2 f_i(y)\| \leq L_* \|x - y\|$, $\|\nabla^2 f_i(x) - \nabla^2 f_i(y)\|_{\mathrm{F}} \leq L_{\mathrm{F}} \|x - y\|$, $\max_{j,l} |(\nabla^2 f_i(x) - \nabla^2 f_i(y))_{jl}| \leq L_\infty \|x - y\|$ to hold for all $i \in [n]$ and $x, y \in \mathbb{R}^d$.*

Define constants C and D depending on which option is used for global model update, namely $C = 2, D = L_*^2$ if *Option 1* is used, and $C = 8, D = (L_* + 2L_{\mathrm{F}})^2$ if *Option 2* is used. We prove three local rates for Newton-3PC: for the squared distance to the solution $\|x^k - x^*\|^2$, and for the Lyapunov function

$$\Phi^k := \mathcal{H}^k + 6(1/A + 3AB) L_{\mathrm{F}}^2 \|x^k - x^*\|^2.$$

where $\mathcal{H}^k := \frac{1}{n}\sum_{i=1}^n \|\mathbf{H}_i^k - \nabla^2 f_i(x^*)\|_{\mathrm{F}}^2$.

We present our theoretical results for local convergence with two stages. For the first stage, we derive convergence rates using specific *locality conditions* for model/Hessian estimation error. In the second stage, we prove that these locality conditions are satisfied for different situations.

Theorem 4.2. *Let Assumption 4.1 hold. Assume $\|x^0 - x^*\| \leq \frac{\mu}{\sqrt{2D}}$ and $\mathcal{H}^k \leq \frac{\mu^2}{4C}$ for all $k \geq 0$. Then,* Newton-3PC *(Algorithm 1) with any 3PC mechanism converges with the following rates:*

$$\|x^k - x^*\|^2 \leq \tfrac{1}{2^k}\|x^0 - x^*\|^2, \tag{9}$$

$$\mathbb{E}\left[\Phi^k\right] \leq (1-\rho)^k \, \Phi^0, \ \rho = \min\left\{\tfrac{A}{2}, \tfrac{1}{3}\right\}, \tag{10}$$

$$\mathbb{E}\left[\frac{\|x^{k+1} - x^*\|^2}{\|x^k - x^*\|^2}\right] \leq (1-\rho)^k \left(C + \frac{AD}{12(1+3AB)L_{\mathrm{F}}^2}\right)\frac{\Phi^0}{\mu^2}. \tag{11}$$

Clearly, these rates are independent of the condition number of the problem, and the choice of 3PC can control the parameter A. Notice that locality conditions here are upper bounds on the initial model error $\|x^0 - x^*\|$ and the errors $\mathcal{H}^k$ for all $k \geq 0$. It turns out that the latter condition may not be guaranteed in general since it depends on the structure of the 3PC mechanism. Below, we show these locality conditions under some assumptions on 3PC, covering practically all compelling cases.

Lemma 4.3 (Deterministic 3PC)**.** *Let the 3PC compressor in* Newton-3PC *be deterministic. Assume the following initial conditions hold: $\|x^0 - x^*\| \leq e_1 := \min\{\frac{A\mu}{\sqrt{8(1+3AB)}L_{\mathrm{F}}}, \frac{\mu}{\sqrt{2D}}\}$ and $\|\mathbf{H}_i^0 - \nabla^2 f_i(x^*)\|_{\mathrm{F}} \leq \frac{\mu}{2\sqrt{C}}$. Then $\|x^k - x^*\| \leq e_1$ and $\|\mathbf{H}_i^k - \nabla^2 f_i(x^*)\|_{\mathrm{F}} \leq \frac{\mu}{2\sqrt{C}}$ for all $k \geq 0$.*

Lemma 4.4 (CBAG)**.** *Consider CBAG mechanism with only source of randomness from Bernoulli aggregation. Assume $\|x^0 - x^*\| \leq e_2 := \min\{\frac{(1-\sqrt{1-\alpha})\mu}{4\sqrt{C}L_{\mathrm{F}}}, \frac{\mu}{\sqrt{2D}}\}$ and $\|\mathbf{H}_i^0 - \nabla^2 f_i(x^*)\|_{\mathrm{F}} \leq \frac{\mu}{2\sqrt{C}}$. Then $\|x^k - x^*\| \leq e_2$ and $\|\mathbf{H}_i^k - \nabla^2 f_i(x^*)\|_{\mathrm{F}} \leq \frac{\mu}{2\sqrt{C}}$ for all $k \geq 0$.*

5 Extension to Bidirectional Compression (Newton-3PC-BC)

In this section, we consider the setup where both directions of communication between devices and the central server are bottleneck. For this setup, we propose Newton-3PC-BC (Algorithm 2) which additionally applies Bernoulli aggregation for gradients (worker to server direction) and another 3PC mechanism for the global model (server to worker direction) employed the master.

Overall, the method integrates three independent communication schemes: workers' 3PC (denoted by $\mathcal{C}^W$) for local Hessian matrices $\nabla^2 f_i(z^{k+1})$, master's 3PC (denoted by $\mathcal{C}^M$) for the global model x^{k+1} and Bernoulli aggregation with probability $p \in (0,1]$ for local gradients $\nabla f_i(z^{k+1})$. Because of these three mechanisms, the method maintains three sequences of model parameters $\{x^k, w^k, z^k\}_{k\geq 0}$. Notice that, Bernoulli aggregation for local gradients is a special case of CBAG (Example 3.9), which allows to skip the computation of local gradients with probability $(1-p)$. However, this reduction in gradient computation necessitates algorithmic modification in order to guarantee convergence. Specifically, we design gradient estimator g^{k+1} to be the full gradient $\nabla f(z^{k+1})$ if devices compute local gradients (i.e., $\xi = 1$). Otherwise, when gradient computation is skipped (i.e., $\xi = 0$), we estimate the missing gradient using Hessian estimate $\mathbf{H}^{k+1}$ and stale gradient $\nabla f(w^{k+1})$, namely we set $g^{k+1} = [\mathbf{H}^{k+1}]_\mu (z^{k+1} - w^{k+1}) + \nabla f(w^{k+1})$.

Similar to the previous result, we present convergence rates and guarantees for locality separately. Let $A_M(A_W)$, $B_M(B_W)$ be parameters of the master's (workers') 3PC mechanisms. Define constants $C_M := \frac{4}{A_M} + 1 + \frac{5B_M}{2}$, $C_W := \frac{4}{A_W} + 1 + \frac{5B_W}{2}$ and Lyapunov function $\Phi_1^k := \|z^k - x^*\|^2 + C_M\|x^k - x^*\|^2 + \frac{A_M(1-p)}{4p}\|w^k - x^*\|^2$.

Theorem 5.1. *Let Assumption 4.1 holds. Assume $\|z^k - x^*\|^2 \leq \frac{A_M\mu^2}{24C_M L_*^2}$ and $\mathcal{H}^k \leq \frac{A_M\mu^2}{96C_M}$ for all $k \geq 0$. Then,* Newton-3PC-BC *(Algorithm 2) converges with the following linear rate:*

$$\mathbb{E}[\Phi_1^k] \leq \left(1 - \min\{\tfrac{A_M}{4}, \tfrac{3p}{8}\}\right)^k \Phi_1^0. \tag{12}$$

Note that the above linear rate for Φ_1^k does not depend on the conditioning of the problem and implies linear rates for all three sequences $\{x^k, w^k, z^k\}$. Next we prove locality conditions used in the theorem for two cases:

Algorithm 2 Newton-3PC-BC (Newton's method with 3PC and Bidirectional Compression)

Parameters: Workers-side 3PC ($\mathcal{C}^W$), Master-side 3PC ($\mathcal{C}^M$), gradient probability $p \in (0,1]$
Input: $x^0 = w^0 = z^0 \in \mathbb{R}^d$; $\mathbf{H}_i^0 \in \mathbb{R}^{d\times d}$, and $\mathbf{H}^0 := \frac{1}{n}\sum_{i=1}^n \mathbf{H}_i^0$; $\xi^0 = 1$; $g^0 = \nabla f(z^0)$
on server
  Update the global model to $x^{k+1} = z^k - [\mathbf{H}^k]_\mu^{-1} g^k$
  Apply Master-side 3PC and send model estimate $z^{k+1} = \mathcal{C}^M_{z^k,x^k}(x^{k+1})$ to all devices $i \in [n]$
  Sample $\xi^{k+1} \sim \text{Bernoulli}(p)$ and send to all $i \in [n]$
for each device $i = 1,\ldots,n$ in parallel **do**
  Get $z^{k+1} = \mathcal{C}^M_{z^k,x^k}(x^{k+1})$ and ξ^{k+1} from the server
  if $\xi^{k+1} = 1$
    $w^{k+1} = z^{k+1}$, compute local gradient $\nabla f_i(z^{k+1})$ and send to the server
  if $\xi^{k+1} = 0$
    $w^{k+1} = w^k$
  Apply Worker's 3PC and update local Hessian estimator to $\mathbf{H}_i^{k+1} = \mathcal{C}^W_{\mathbf{H}_i^k,\nabla^2 f_i(z^k)}(\nabla^2 f_i(z^{k+1}))$
end for
on server
  $\nabla f(z^{k+1}) = \frac{1}{n}\sum_{i=1}^n \nabla f_i(z^{k+1})$, $\mathbf{H}^{k+1} = \frac{1}{n}\sum_{i=1}^n \mathbf{H}_i^k$
  if $\xi^{k+1} = 1$
    $w^{k+1} = z^{k+1},\ g^{k+1} = \nabla f(z^{k+1})$
  if $\xi^{k+1} = 0$
    $w^{k+1} = w^k$,
    $g^{k+1} = [\mathbf{H}^{k+1}]_\mu (z^{k+1} - w^{k+1}) + \nabla f(w^{k+1})$

for non-random 3PC schemes and for schemes that preserve certain convex combination condition. It can be seen easily that random sparsification fits into the second case.

Lemma 5.2 (Deterministic 3PC). *Let Assumption 4.1 holds. Let $\mathcal{C}^M$ and $\mathcal{C}^W$ be deterministic. Assume $\|x^0 - x^*\|^2 \le \frac{11A_M}{24C_M} e_3^2 := \frac{11A_M}{24C_M}\min\{\frac{A_M\mu^2}{24C_M L_*^2}, \frac{A_W A_M \mu^2}{384 C_W C_M L_{\rm F}^2}\}$ and $\mathcal{H}^0 \le \frac{A_M\mu^2}{96C_M}$. Then $\|x^k - x^*\|^2 \le \frac{11A_M}{24C_M} e_3^2$, $\|z^k - x^*\|^2 \le e_3^2$ and $\mathcal{H}^k \le \frac{A_M\mu^2}{96C_M}$ for all $k \ge 0$*

Lemma 5.3 (Random sparsification). *Let Assumption 4.1 holds. Assume $(z^k)_j$ is a convex combination of $\{(x^t)_j\}_{t=0}^k$, and $(\mathbf{H}_i^k)_{jl}$ is a convex combination of $\{(\nabla^2 f_i(z^k))_{jl}\}_{t=0}^k$ for all $i \in [n]$, $j,l \in [d]$, and $k \ge 0$. If $\|x^0 - x^*\|^2 \le e_4^2 := \min\{\frac{\mu^2}{d^2L_*^2}, \frac{A_M\mu^2}{24dC_ML_*^2}, \frac{A_M\mu^2}{96d^3C_ML_\infty^2}, \frac{\mu^2}{4d^4L_\infty^2}\}$, then $\|z^k - x^*\|^2 \le de_4^2$ and $\mathcal{H}^k \le \min\{\frac{A_M\mu^2}{96C_M}, \frac{\mu^2}{4d}\}$ for all $k \ge 0$.*

6 Experiments

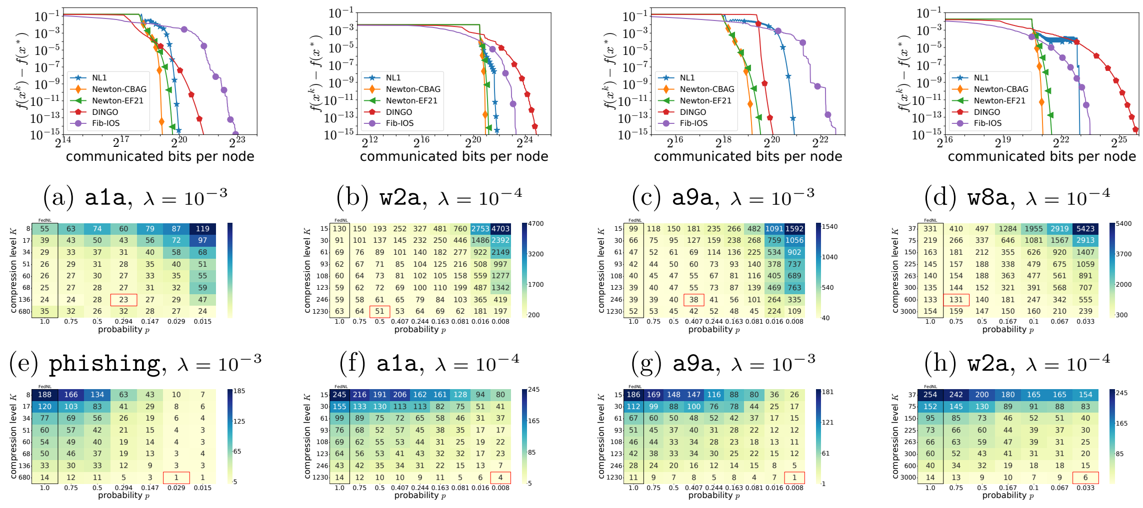
In this part, we study the empirical performance of Newton-3PC comparing its performance against other second-order methods on logistic regression problems of the form

$$\min_{x\in\mathbb{R}^d} \left\{ f(x) := \frac{1}{n}\sum_{i=1}^n f_i(x) + \frac{\lambda}{2}\|x\|^2 \right\}, \tag{13}$$

where $f_i(x) = \frac{1}{m}\sum_{j=1}^m \log\left(1 + \exp(-b_{ij}a_{ij}^\top x)\right)$ and $\{a_{ij}, b_{ij}\}_{j\in[m]}$ are data points belonging to i-th client. We use datasets from LibSVM library (Chang & Lin, 2011). Each dataset was shuffled and split into n equal parts. Detailed description of datasets and hyperparameters choice is given in the Appendix.

6.1 Comparison between Newton-3PC and other second-order methods

According to Safaryan et al. (2022), FedNL with Rank-1 compressor outperforms other second-order methods in all cases in terms of communication complexity. Thus, we compare in Figure 1 (first row) Newton-CBAG (based on Top-d compressor and probability $p = 0.75$), Newton-EF21 with Rank-1, NL1 with Rand-1, DINGO,



(a) a1a, $\lambda = 10^{-3}$ (b) w2a, $\lambda = 10^{-4}$ (c) a9a, $\lambda = 10^{-3}$ (d) w8a, $\lambda = 10^{-4}$

(e) phishing, $\lambda = 10^{-3}$ (f) a1a, $\lambda = 10^{-4}$ (g) a9a, $\lambda = 10^{-3}$ (h) w2a, $\lambda = 10^{-4}$

(i) phishing, $\lambda = 10^{-3}$ (j) a1a, $\lambda = 10^{-4}$ (k) a9a, $\lambda = 10^{-3}0$ (l) w2a, $\lambda = 10^{-4}$

Figure 1: Comparison of Newton-CBAG with Top-d compressor and probability $p = 0.75$, Newton-EF21 with Rank-1 compressor, NL1 with Rand-1 compressor, and DINGO (**first row**). The performance of Newton-CBAG with Top-d in terms of communication complexity (**second row**, in Mbytes) and the number of local Hessian computations (**third row**).

and Fib-IOS indicating how many bits are transmitted by each client in both uplink and downlink directions. We clearly see that Newton-CBAG is much more communication efficient than NL1, Fib-IOS and DINGO. Besides, it outperforms FedNL in all cases. On top of that, we achieve improvement not only in communication complexity, but also in computational cost with Newton-CBAG. Indeed, when clients do not send compressed Hessian differences to the server there is no need to compute local Hessians. Consequently, computational costs goes down. We decided not to compare Newton-3PC with first-order methods since FedNL already outperforms them in terms of communication complexity in a variety of experiments in (Safaryan et al., 2022).

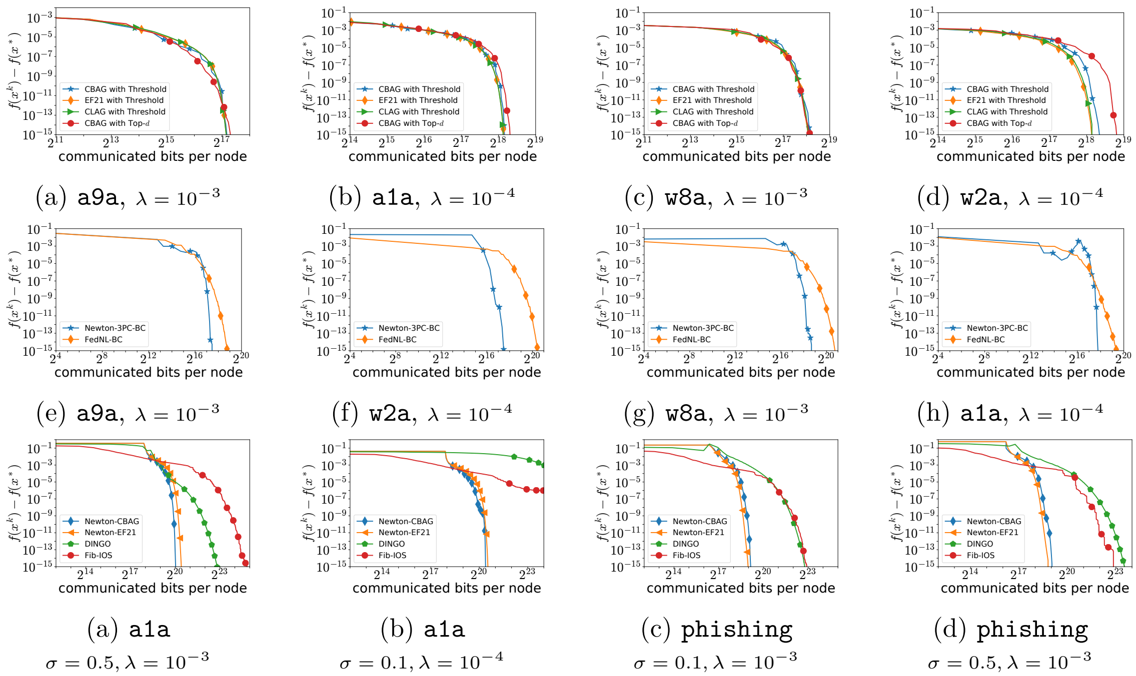
6.2 Does Bernoulli aggregation bring any advantage?

Next, we investigate the performance of Newton-CBAG based on Top-K. We report the results in heatmaps (see Figure 1, second row) where we vary probability p along rows and compression level K along columns. Notice that Newton-CBAG reduces to FedNL when $p = 1$ (left column). We observe that Bernoulli aggregation (BAG) is indeed beneficial since the communication complexity reduces when p becomes smaller than 1 (in case of a1a data set the improvement is significant). We can conclude that BAG leads to better communication complexity of Newton-3PC over FedNL.

On top of that, we claim that Newton-CBAG is also computationally more efficient than FedNL; see Figure 1 (third row) that indicates the number of Hessian computations. We observe that even if communication complexity in two regimes are close to each other, but computationally better the one with smaller p. Indeed, in the case when $p < 1$ we do not have to compute local Hessians with probability $1 - p$ that leads to acceleration in terms of computation complexity.

6.3 3PC based on adaptive thresholding

Next we test the performance of Newton-3PC using adaptive thresholding operator (5). We compare Newton-EF21 (equivalent to FedNL), Newton-CBAG, and Newton-CLAG with adaptive thresholding against Newton-CBAG with Top-d compressor. We fix the probability $p = 0.5$ for CBAG, the trigger $\zeta = 2$ for CLAG, and thresholding parameter $\lambda = 0.5$. According to the results presented in Figure 2 (first row), adaptive



(a) a9a, $\lambda = 10^{-3}$ (b) a1a, $\lambda = 10^{-4}$ (c) w8a, $\lambda = 10^{-3}$ (d) w2a, $\lambda = 10^{-4}$

(e) a9a, $\lambda = 10^{-3}$ (f) w2a, $\lambda = 10^{-4}$ (g) w8a, $\lambda = 10^{-3}$ (h) a1a, $\lambda = 10^{-4}$

(a) a1a $\sigma = 0.5, \lambda = 10^{-3}$ (b) a1a $\sigma = 0.1, \lambda = 10^{-4}$ (c) phishing $\sigma = 0.1, \lambda = 10^{-3}$ (d) phishing $\sigma = 0.5, \lambda = 10^{-3}$

Figure 2: Comparison of Newton-CBAG with thresholding and Top-d compressors and Newton-EF21 with thresholding compressor in terms of communication complexity (**first row**). Comparison of Newton-3PC-BC against FedNL-BC in terms of communication complexity (**second row**). The performance of Newton-CBAG combined with Top-d compressor and probability $p = 0.75$, Newton-EF21 with Rank-1 compressor, DINGO, and Fib-ISO in terms of communication complexity on Softmax problem (**third row**).

thresholding can be beneficial since it improves the performance of Newton-3PC in some cases. Moreover, it is computationally cheaper than Top-K as we do not sort entries of a matrix as it is for Top-K.

6.4 Newton-3PC-BC **against** FedNL-BC

In our next experiment we study bidirectional compression. We compare Newton-3PC-BC against FedNL-BC. For Newton-3PC-BC we fix CBAG with $p = 0.75$ combined with Top-d compressor applied on Hessians, BAG with $p = 0.75$ applied on gradients, and 3PCv4 (Richtárik et al., 2022) combined with (Top-K_1, Top-K_2) compressors on iterates. For FedNL-BC we use Top-d compressor on Hessians and BAG with $p = 0.75$ on gradients, and Top-K compressor on iterates. We choose different values for K_1 and K_2 such that it $K_1 + K_2 = K$ always hold. Such choice of parameters allows to make the iteration cost of both methods to be equal. Based on the results, we argue that the superposition of CBAG and 3PCv4 applied on Hessians and iterates respectively is more communication efficient than the combination of EF21 and EF21.

6.5 **Performance of** Newton-3PC **on Softmax problem**

Finally, we also consider L2 regularized Softmax problem where all f_i's of the form

$$f_i(x) = \sigma \log \left(\sum_{j=1}^{m} \exp \left(\frac{a_{ij}^\top x - b_{ij}}{\sigma} \right) \right).$$

Here $\sigma > 0$ is a smoothing parameter. One can show that this function has both Lipschitz continuous gradient and Lipschitz continuous Hessian. We perform the same data shift as it was done in (Hanzely et al., 2020) (section 8.2). Note that in this case we do not compare Newton-3PC against NL1 as this problem does not belong to the class of *generalized linear models*.

We compare Newton-CBAG combined with Top-d compressor and probability $p = 0.75$, Newton-EF21 with Rank-1 compressor, DINGO (Crane & Roosta, 2019), and Fib-IOS (Fabbro et al., 2022). As we can see in Figure 2 (third row), Newton-CBAG and Newton-EF21 demonstrate almost equivalent performance: in some cases slightly better the first one (a1a dataset), in some cases — the second (phishing dataset). Furthermore, DINGO and Fib-IOS are significantly slower than Newton-3PC methods in terms of communication complexity.

References

Artem Agafonov, Brahim Erraji, and Martin Takáč. Flecs-cgd: A federated learning second-order framework via compression and sketching with compressed gradient differences. *arXiv preprint: arXiv 2210.09626*, 2022a.

Artem Agafonov, Dmitry Kamzolov, Rachael Tappenden, Alexander Gasnikov, and Martin Takáč. Flecs: A federated learning second-order framework via compression and sketching. *arXiv preprint: arXiv 2206.02009*, 2022b.

Foivos Alimisis, Peter Davies, and Dan Alistarh. Communication-efficient distributed optimization with quantized preconditioners. In *International Conference on Machine Learning (ICML)*, 2021.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pp. 1709–1720, 2017.

Dan Alistarh, Torsten Hoefler, Mikael Johansson, Sarit Khirirat, Nikola Konstantinov, and Cédric Renggli. The convergence of sparsified gradient methods. In *32nd Conference on Neural Information Processing Systems*, 2018a.

Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cédric Renggli. The convergence of sparsified gradient methods. In *Advances in Neural Information Processing Systems*, pp. 5977–5987, 2018b.

Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1200–1205. ACM, 2017.

Amir Beck. *Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB*. Society for Industrial and Applied Mathematics, USA, 2014. ISBN 1611973643.

Ron Bekkerman, Mikhail Bilenko, and John Langford. *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

Aleksandr Beznosikov, Samuel Horváth, Peter Richtárik, and Mher Safaryan. On biased compression for distributed learning. *arXiv preprint arXiv:2002.12410*, 2020.

Shicong Cen, Huishuai Zhang, Yuejie Chi, Wei Chen, and Tie-Yan Liu. Convergence of distributed stochastic variance reduced methods without sampling extra data. *IEEE TRANSACTIONS ON SIGNAL PROCESSING, volume 68*, 2020.

Chih-Chung Chang and Chih-Jen Lin. LibSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–27, 2011.

T. Chen, G. Giannakis, T. Sun, and W. Yin. LAG: Lazily aggregated gradient for communication-efficient distributed learning. *Advances in Neural Information Processing Systems*, 2018.

Rixon Crane and Fred Roosta. Dingo: Distributed newton-type method for gradient-norm optimization. In *Advances in Neural Information Processing Systems*, volume 32, pp. 9498–9508, 2019.

Nicolò Dal Fabbro, Subhrakanti Dey, Michele Rossi, and Luca Schenato. A newton-type algorithm for federated learning based on incremental hessian eigenvector sharing. *arXiv preprint arXiv: 2202.05800*, 2022.

Hamid Reza Feyzmahdavian and Mikael Johansson. Asynchronous iterations in optimization: New sequence results and sharper algorithmic guarantees. *arXiv preprint arXiv:2109.04522*, 2021.

H. S. Ghadikolaei, S. Stich, and M. Jaggi. LENA: Communication-efficient distributed learning with self-triggered gradient uploads. *International Conference on Artificial Intelligence and Statistics, pp. 3943–3951. PMLR*, 2021.

Avishek Ghosh, Raj Kumar Maity, and Arya Mazumdar. Distributed Newton Can Communicate Less and Resist Byzantine Workers. *Advances in Neural Information Processing Systems*, 2020a.

Avishek Ghosh, Raj Kumar Maity, Arya Mazumdar, and Kannan Ramchandran. Communication efficient distributed approximate newton method. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 2539–2544, 2020b. doi: 10.1109/ISIT44484.2020.9174216.

Avishek Ghosh, Raj Kumar Maity, Arya Mazumdar, and Kannan Ramchandran. Escaping Saddle Points in Distributed Newton's Method with Communication Efficiency and Byzantine Resilience. *arXiv preprint arXiv:2103.09424*, 2020c.

Eduard Gorbunov, Konstantin Burlachenko, Zhize Li, and Peter Richtárik. MARINA: Faster non-convex distributed learning with compression. *arXiv preprint arXiv:2102.07845*, 2021a.

Eduard Gorbunov, Filip Hanzely, and Peter Richtárik. Local SGD: Unified theory and new efficient methods. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021b.

Robert Gower and Peter Richtárik. Stochastic dual ascent for solving linear systems. *arXiv preprint: arXiv:1512.06890*, 2015.

Robert M. Gower and Peter Richtárik. Randomized quasi-newton updates are linearly convergent matrix inversion algorithms. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1380–1409, 2017.

Vipul Gupta, Avishek Ghosh, Michal Derezinski, Rajiv Khanna, Kannan Ramchandran, and Michael Mahoney. LocalNewton: Reducing Communication Bottleneck for Distributed Learning. In *37th Conference on Uncertainty in Artificial Intelligence (UAI 2021)*, 2021.

Filip Hanzely, Konstantin Mishchenko, and Peter Richtarik. Sega: Variance reduction via gradient sketching. In *Advances in Neural Information Processing Systems*, 2018.

Filip Hanzely, Nikita Doikov, Peter Richtárik, and Yurii Nesterov. Stochastic subspace cubic newton method. *arXiv preprint: arXiv 2002.09526*, 2020.

Samuel Horváth, Dmitry Kovalev, Konstantin Mishchenko, Sebastian Stich, and Peter Richtárik. Stochastic distributed learning with gradient quantization and variance reduction. *arXiv preprint arXiv:1904.05115*, 2019.

Rustem Islamov, Xun Qian, and Peter Richtárik. Distributed second order methods with fast rates and compressed communication. *International Conference on Machine Learning (ICML)*, 2021.

Peter Kairouz et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

S. P. Karimireddy, Q. Rebjock, S. Stich, and M. Jaggi. Error feedback fixes SignSGD and other gradient compression schemes. *36th International Conference on Machine Learning (ICML)*, 2019.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for on-device federated learning. In *International Conference on Machine Learning (ICML)*, 2020.

Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Learning from History for Byzantine Robust Optimization. *International Conference on Machine Learning (ICML)*, 2021.

Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Byzantine-Robust Learning on Heterogeneous Datasets via Bucketing. *International Conference on Learning Representations (ICLR)*, 2022.

Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. *Proceedings of the 36th International Conference on Machine Learning, volume 97, pp. 3478–3487. PMLR*, 2019.

Anastasia Koloskova, Tao Lin, Sebastian U Stich, and Martin Jaggi. Decentralized deep learning with arbitrary communication compression. *International Conference on Learning Representations*, 2020.

Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016a.

Jakub Konečný, H. Brendan McMahan, Felix Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: strategies for improving communication efficiency. In *NIPS Private Multi-Party Machine Learning Workshop*, 2016b.

Dmitry Kovalev, Konstanting Mishchenko, and Peter Richtárik. Stochastic Newton and cubic Newton methods with simple local linear-quadratic rates. In *NeurIPS Beyond First Order Methods Workshop*, 2019.

Dmitry Kovalev, Anastasia Koloskova, Martin Jaggi, Peter Richtárik, and Sebastian U. Stich. A linearly convergent algorithm for decentralized optimization: sending less bits for free! *The 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

Jason D. Lee, Qihang Lin, Tengyu Ma, and Tianbao Yang. Distributed Stochastic Variance Reduced Gradient Methods by Sampling Extra Data with Replacement. *Journal of Machine Learning Research, vilume 18, pages 1-43*, 2017.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020a. doi: 10.1109/MSP.2020.2975749.

Zhize Li, Dmitry Kovalev, Xun Qian, and Peter Richtárik. Acceleration for compressed gradient descent in distributed and federated optimization. In *International Conference on Machine Learning*, 2020b.

Chieh-Yen Lin, Cheng-Hao Tsai, Ching pei Lee, and Chih-Jen Lin. Large-scale logistic regression and linear support vector machines using spark. *2014 IEEE International Conference on Big Data (Big Data)*, pp. 519–528, 2014.

Alessio Maritan, Ganesh Sharma, Luca Schenato, and Subhrakanti Dey. Network-giant: Fully distributed newton-type optimization via harmonic hessian consensus. *arXiv preprint arXiv: 2305.07898*, 2023.

H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč, and Peter Richtárik. Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*, 2019.

Konstantin Mishchenko, Grigory Malinovsky, Sebastian Stich, and Peter Richtárik. ProxSkip: Yes! Local gradient steps provably lead to communication acceleration! Finally! *39th International Conference on Machine Learning (ICML)*, 2022.

Giorgi Nadiradze, Amirmojtaba Sabour, Peter Davies, Shigang Li, and Dan Alistarh. Asynchronous decentralized SGD with quantized and local updates. *35th Conference on Neural Information Processing Systems*, 2021a.

Giorgi Nadiradze, Amirmojtaba Sabour, Peter Davies, Shigang Li, and Dan Alistarh. Asynchronous decentralized SGD with quantized and local updates. *Advances in Neural Information Processing Systems*, 2021b.

Barak A. Pearlmutter. Fast exact multiplication by the hessian. *Neural Computation*, 1994.

Xun Qian, Peter Richtárik, and Tong Zhang. Error compensated distributed SGD can be accelerated. *35th Conference on Neural Information Processing Systems*, 2021.

Xun Qian, Rustem Islamov, Mher Safaryan, and Peter Richtárik. Basis matters: Better communication-efficient second order methods for federated learning. *In Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022.

Sashank J. Reddi, Jakub Konečný, Peter Richtárik, Barnabás Póczos, and Alexander J. Smola. AIDE: Fast and communication efficient distributed optimization. *CoRR*, abs/1608.06879, 2016.

Peter Richtárik and Martin Takac. Stochastic reformulations of linear systems: algorithms and convergence theory. *arXiv preprint: arXiv:1706.01108*, 2017.

Peter Richtárik, Igor Sokolov, and Ilyas Fatkhullin. Ef21: A new, simpler, theoretically better, and practically faster error feedback. *35th Conference on Neural Information Processing Systems*, 2021.

Peter Richtárik, Igor Sokolov, Ilyas Fatkhullin, Elnur Gasanov, Zhize Li, and Eduard Gorbunov. 3pc: Three point compressors for communication-efficient distributed training and a better theory for lazy aggregation. *39th International Conference on Machine Learning (ICML)*, 2022.

Fred Roosta, Yang Liu, Peng Xu, and Michael W. Mahoney. Newton-MR: Newton's Method Without Smoothness or Convexity. *arXiv preprint arXiv:1810.00303*, 2019.

Mher Safaryan, Rustem Islamov, Xun Qian, and Peter Richtárik. FedNL: Making Newton-Type Methods Applicable to Federated Learning. *39th International Conference on Machine Learning (ICML)*, 2022.

Atal Narayan Sahu, Aritra Dutta, Ahmed M. Abdelmoniem, Trambak Banerjee, Marco Canini, and Panos Kalnis. Rethinking gradient sparsification as total error minimization. *35th Conference on Neural Information Processing Systems*, 2021.

Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-effcient distributed optimization using an approximate newton-type method. In *Proceedings of the 31th International Conference on Machine Learning*, volume 32, pp. 1000–1008, 2014.

S. U. Stich, J.-B. Cordonnier, and M. Jaggi. Sparsified SGD with memory. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Sebastian U. Stich. Local SGD converges fast and communicates little. In *International Conference on Learning Representations (ICLR)*, 2020.

J. Sun, T. Chen, G. Giannakis, and Z. Yang. Communication-efficient distributed learning via lazily aggregated quantized gradients. *Advances in Neural Information Processing Systems, 32:3370–3380*, 2019.

Alexander Tyurin and Peter Richtárik. Distributed nonconvex optimization with communication compression, optimal oracle complexity, and no client synchronization. *arXiv preprint arXiv:2202.01268*, 2022.

Shusen Wang, Fred Roosta abd Peng Xu, and Michael W Mahoney. GIANT: Globally improved approximate Newton method for distributed optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. Gradient sparsification for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems*, pp. 1306–1316, 2018.

Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. In *Advances in Neural Information Processing Systems*, pp. 1509–1519, 2017.

Haibo Yang, Minghong Fang, and Jia Liu. Achieving Linear Speedup with Partial Worker Participation in Non-IID Federated Learning. *International Conference on Learning Representations (ICLR)*, 2021.

Jiaqi Zhang, Keyou You, and Tamer Basar. Distributed adaptive Newton methods with globally superlinear convergence. *arXiv preprint arXiv:2002.07378*, 2020a.

Jiaqi Zhang, Keyou You, and Tamer Başar. Achieving globally superlinear convergence for distributed optimization with adaptive newton method. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 2329–2334, 2020b. doi: 10.1109/CDC42340.2020.9304321.

Yuchen Zhang and Lin Xiao. DiSCO: Distributed optimization for self-concordant empirical loss. In *In Proceedings of the 32nd International Conference on Machine Learning, PMLR, volume 37, pages 362–370*, 2015.

Yong Zhuang, Wei-Sheng Chin, Yu-Chin Juan, and Chih-Jen Lin. Distributed newton methods for regularized logistic regression. In Tru Cao, Ee-Peng Lim, Zhi-Hua Zhou, Tu-Bao Ho, David Cheung, and Hiroshi Motoda (eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 690–703, Cham, 2015. Springer International Publishing. ISBN 978-3-319-18032-8.