MARINA MEETS MATRIX STEPSIZES: VARIANCE REDUCED DISTRIBUTED NON-CONVEX OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Matrix-stepsized gradient descent algorithms have been demonstrated to exhibit superior efficiency in non-convex optimization compared to their scalar counterparts. The `det-CGD` algorithm, as introduced by Li et al. (2023), leverages matrix stepsizes to perform compressed gradient descent for non-convex objectives and matrix-smooth problems in a federated manner. The authors establish the algorithm's convergence to a neighborhood of the weighted stationarity point under a convex condition for the symmetric and positive-definite stepsize matrix. In this paper, we propose a variance-reduced version of the `det-CGD` algorithm, incorporating the `MARINA` method. Notably, we establish theoretically and empirically, that `det-MARINA` outperforms both `MARINA` and the distributed `det-CGD` algorithms in terms of iteration and communication complexities.

1 INTRODUCTION

1.1 PROBLEM SETTING

We are focusing on optimizing the finite sum non-convex objective, as defined below:

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

In this context, each function $f_i : \mathbb{R}^d \to \mathbb{R}$ is differentiable. It is assumed that f is lower bounded by $f^\star$, while each individual function f_i is lower bounded by $f_i^\star$. This type of objective function finds extensive application in various practical machine learning algorithms, which increase not only in terms of the data size but also in the model size and overall complexity as well. As a result, most neural network architectures result in highly non-convex empirical losses, which need to be minimized. In addition, it becomes computationally infeasible to train these models on one device, often excessively large, and one needs to redistribute them amongst different devices/clients. This redistribution results in a high communication overheads, which are often become the bottleneck in this framework.

In other words, we have the following setting. The data is partitioned into n clients, where the i-th client has access to the component function f_i and its derivatives. The clients are connected to each other through a central device, called the server. In this work, we are going to study iterative gradient descent-based algorithms that operate as follows: The clients compute the local gradients in parallel. Then they compress these gradients to reduce the communication cost and send them to the server in parallel. The server then aggregates these vectors and broadcasts the iterate update back to the clients. This meta-algorithm is called federated learning. We refer the readers to Konečnỳ et al. (2016); McMahan et al. (2017); Kairouz et al. (2021) for a more thorough introduction to federated learning.

Contributions. In this paper, we introduce a federated learning algorithm named `det-MARINA`. This algorithm extends a recent method called `det-CGD` (Li et al., 2023), which aims to solve (1) using matrix stepsized gradient descent. Under the matrix smoothness assumption (Safaryan et al., 2021), they demonstrate that the matrix stepsized version of the Distributed Compressed Gradient Descent (Khirirat et al., 2018) algorithm enhances communication complexity compared to its scalar counterpart. However, in their analysis, Li et al. (2023) show stationarity only within a certain neighborhood due to stochastic compressors. Our algorithm addresses this issue by incorporating a variance reduction scheme called `MARINA` (Gorbunov et al., 2021), which is an optimal federated learning algorithm in the non-convex setting. We establish theoretically and empirically, that `det-MARINA` outperforms both `MARINA` and

the distributed det-CGD algorithms in terms of iteration and communication complexities. In addition, we describe specific matrix stepsize choices, for which algorithm beats the scalar MARINA and distributed det-CGD both in theory and in practice.

1.2 Background and motivation

Ideally, for any given $\varepsilon > 0$, the goal is to find a point $x_\varepsilon \in \mathbb{R}^d$ satisfying:

$$f(x) - f^\star \geq \varepsilon.$$

However, it's important to note that in the general case, finding such an approximately global optimum is known to be NP-hard, as discussed in Jain et al. (2017); Danilova et al. (2022).

On the contrary, convex optimization problems are well-studied, and there exists an extensive body of literature on various methods. In the context of our work, methods based on gradient descent are of particular interest. When these methods are applied to non-convex objectives, they treat the function f as locally convex and aim to converge to a local minimum. Despite this simplification, such methods have gained popularity in practice due to their superior performance compared to other approaches for non-convex optimization, such as convex relaxation-based methods (Tibshirani, 1996; Cai et al., 2010).

Stochastic Gradient Descent. Arguably, one of the most prominent meta-methods for tackling non-convex optimization problems is stochastic gradient descent (SGD). The formulation of SGD is presented as the following iterative algorithm:

$$x^{k+1} = x^k - \gamma g^k. \tag{SGD}$$

Here, $g^k \in \mathbb{R}^d$ serves as a stochastic estimator of the gradient $\nabla f(x^k)$. SGD essentially mimics the classical gradient descent algorithm, and recovers it when $g^k = \nabla f(x^k)$. In this scenario, the method approximates the objective function f using a linear function and takes a step of size γ in the direction that maximally reduces this approximation. When the stepsize is sufficiently small, and the function f is suitably smooth, it can be demonstrated that the function value decreases, as discussed in Bubeck et al. (2015); Gower et al. (2019).

However, computing the full gradient can often be computationally expensive. In such cases, stochastic approximations of the gradient come into play. Stochastic estimators of the gradient can be employed for various purposes, leading to the development of different methods. These include stochastic batch gradient descent (Nemirovski et al., 2009; Johnson & Zhang, 2013; Defazio et al., 2014), randomized coordinate descent (Nesterov, 2012; Wright, 2015), and compressed gradient descent (Alistarh et al., 2017; Khirirat et al., 2018; Mishchenko et al., 2019). The latter, compressed gradient descent, holds particular relevance to this paper, and we will delve into a more detailed discussion of it in subsequent sections.

Second order methods. The stochastic gradient descent is considered as a first-order method as it uses only the first order derivative information. Although being immensely popular, the first order methods are not always the most optimal. Not surprisingly, using higher order derivatives in deciding update direction can yield to faster algorithms. A simple instance of such algorithms is the Newton Star algorithm (Islamov et al., 2021):

$$x^{k+1} = x^k - \left(\nabla^2 f(x^\star)\right)^{-1} \nabla f(x^k), \tag{NS}$$

where $x^\star$ is the minimum point of the objective function. The authors establish that under specific conditions, the algorithm's convergence to the unique solution $x^\star$ in the convex scenario occurs at a local quadratic rate. Nonetheless, its practicality is limited since we do not have prior knowledge of the Hessian matrix at the optimal point. Despite being proposed recently, the Newton-Star algorithm gives a deeper insight on the generic Newton method (Gragg & Tapia, 1974; Miel, 1980; Yamamoto, 1987):

$$x^{k+1} = x^k - \gamma \left(\nabla^2 f(x^k)\right)^{-1} \nabla f(x^k). \tag{NM}$$

Here, the unknown Hessian of the Newton-Star algorithm, is estimated progressively along the iterations. The latter causes elevated computational costs, as the inverting a large square matrix is expensive. As an alternative, quasi-Newton methods replace the inverse of the Hessian at the iterate with a computationally cheaper estimate (Broyden, 1965; Dennis & Moré, 1977; Al-Baali & Khalfan, 2007; Al-Baali et al., 2014).

Fixed matrix stepsizes. The det-CGD algorithm falls into this framework of the second order methods as well. Proposed by Li et al. (2023)[1], the algorithm suggests using a uniform "upper bound" on the inverse Hessian matrix. Assuming matrix smoothness of the objective (Safaryan et al., 2021), they replace the scalar stepsize with a positive definite matrix $\boldsymbol{D}$. The algorithm is given as follows:

$$x^{k+1} = x^k - \boldsymbol{D}\boldsymbol{S}^k \nabla f(x^k). \tag{det-CGD}$$

- Here, $\boldsymbol{D}$ plays the role of the stepsize. It essentially uniformly upper bounds the inverse Hessian. The standard SGD is a particular case of this method, as the scalar stepsize γ can be seen as a matrix $\gamma \boldsymbol{I}_d$, where $\boldsymbol{I}_d$ is the d-dimensional identity matrix. An advantage of using a matrix stepsize is more evident if we take the perspective of the second order methods. Indeed, the scalar stepsize $\gamma \boldsymbol{I}_d$ uniformly estimates the largest eigenvalue of the Hessian matrix, while $\boldsymbol{D}$ can capture the Hessian more accurately. The authors show both theoretical and empirical improvement that comes with matrix stepsizes.
- $\boldsymbol{S}^k$ is a positive semi-definite stochastic sketch matrix, that is unbiased: $\mathbb{E}[\boldsymbol{S}^k] = \boldsymbol{I}_d$. We notice that det-CGD can be seen as a matrix stepsize instance of SGD, with $g^k = \boldsymbol{S}^k \nabla f(x^k)$. The sketch matrix can be seen as a linear compressing operator, hence the name of the algorithm: CGD (Compressed Gradient Descent) (see Alistarh et al. (2017); Khirirat et al. (2018)). A commonly used example of such a compressor is the Rand-k compressor. This compressor randomly selects m entries from its input and scales them using a scalar multiplier to ensure an unbiased estimation. By adopting this approach, instead of using all d coordinates of the gradient, only a subset of size m is communicated. Formally, rand-τ is defined as follows:

$$\boldsymbol{S} = \frac{d}{\tau} \sum_{j=1}^{\tau} e_{i_j} e_{i_j}^\top. \tag{2}$$

  Here, e_{i_j} denotes the i_j-th standard basis vector in $\mathbb{R}^d$. For a more comprehensive understanding of compression techniques, we refer to the paper by Safaryan et al. (2022b).

The neighborhood of the distributed det-CGD1. The distributed version of det-CGD follows the standard federated learning paradigm (McMahan et al., 2017). The pseudocode of the method, as well as the convergence result of Li et al. (2023), can be found in the appendix. Informally, their convergence result can be written as

$$\min_{k=1,\ldots,K} \mathbb{E}[\left\|\nabla f(x^k)\right\|_{\boldsymbol{D}}^2] \leq \mathcal{O}\left(\frac{(1+\alpha)^K}{K}\right) + \mathcal{O}(\alpha), \tag{3}$$

where $\alpha > 0$ is a constant that can be controlled. The crucial insight from this result is that the error bound doesn't diminish as the number of iterations increases. In fact, by controlling α and considering a large K, it's impossible to make the second term smaller than ε. This implies that the algorithm converges to a certain neighborhood surrounding the (local) optimum. This phenomenon is a common occurrence in SGD and is primarily attributable to the variance associated with the stochastic gradient estimator. In the case of det-CGD the stochasticity comes from the sketch $\boldsymbol{S}^k$.

Variance reduction. To eliminate this neighborhood, various techniques for reducing variance are employed. One of the simplest techniques applicable to CGD is gradient shifting. By replacing $\boldsymbol{S}^k \nabla f(x^k)$ with $\boldsymbol{S}^k(\nabla f(x^k) - \nabla f(x^\star)) + \nabla f(x^\star)$, the neighborhood effect is removed from the general CGD. This algorithm is an instance of a more commonly known method called $\text{SGD}_\star$ (Gower et al., 2020). However, since the exact optimum $x^\star$ is typically unknown, this technique encounters similar challenges as the Newton-Star algorithm mentioned earlier. Fortunately, akin to quasi-Newton methods, one can employ methods that iteratively learn the optimal shift (Shulgin & Richtárik, 2022).

A line of research focuses on variance reduction for CGD based on this insight. To mitigate the neighborhood effect in the distributed version of CGD, denoted as det-CGD1, we apply a technique called MARINA (Gorbunov et al., 2021). MARINA cleverly combines the general shifting technique with loopless variance reduction techniques (Qian et al., 2021). This approach introduces an alternative gradient estimator specifically designed for the federated learning setting. Thanks to its structure, it allows to establish an upper bound on the stationarity error that diminishes significantly with a large number of iterations. In this paper, we construct the analog of the this algorithm called det-MARINA, using matrix stepsizes and sketch gradient compressors. For this new method, we prove a convergence guarantee similar to (3) that without a neighborhood term.

[1] In the original paper, the algorithm is referred to as det-CGD, as there is a variant of the same algorithm named det-CGD2. Since we are going to use only the first one and our framework is applicable to both, we will remove the number in the end for the sake of brevity.

1.3 PRIOR WORK

Numerous effective convex optimization techniques have been adapted for application in non-convex scenarios. Here's a selection of these techniques, although it's not an exhaustive list: adaptivity (Dvinskikh et al., 2019; Zhang et al., 2020b), variance reduction (J Reddi et al., 2016; Li et al., 2021), and acceleration (Guminov et al., 2019). Of particular relevance to our work is the paper by Khaled & Richtárik (2020), which introduces a unified approach for analyzing stochastic gradient descent for non-convex objectives. A comprehensive overview of non-convex optimization can be found in (Jain et al., 2017; Danilova et al., 2022).

An illustrative example of a matrix stepsized method is Newton's method, which has been a long-standing favorite in the optimization community (Gragg & Tapia, 1974; Miel, 1980; Yamamoto, 1987). However, the computational complexity involved in computing the stepsize as the inverse of the Hessian of the current iteration is substantial. Instead, quasi-Newton methods employ a readily computable estimator to replace the inverse Hessian (Broyden, 1965; Dennis & Moré, 1977; Al-Baali & Khalfan, 2007; Al-Baali et al., 2014). An important direction of research that is relevant to our work, studies distributed second order methods. Here is a non-exhaustive list of papers in this area: Wang et al. (2018); Crane & Roosta (2019); Zhang et al. (2020a); Islamov et al. (2021); Alimisis et al. (2021); Safaryan et al. (2022a).

The Distributed Compressed Gradient Descent (`DCGD`) algorithm, initially proposed by Khirirat et al. (2018), has seen improvements in various aspects, as documented in works such as (Li et al., 2020; Horváth et al., 2022). Its variance reduced version with gradients shifts was studied by Shulgin & Richtárik (2022) in the (strongly) convex setting. Additionally, there exists a substantial body of literature on other federated learning algorithms employing unbiased compressors (Alistarh et al., 2017; Mishchenko et al., 2019; Gorbunov et al., 2021; Mishchenko et al., 2022; Maranjyan et al., 2022; Horváth et al., 2023).

Variance reduction techniques have gained significant attention in the context of stochastic batch gradient descent that is prevalent in machine learning. Numerous algorithms have been developed in this regard, including well-known ones like `SVRG` (Johnson & Zhang, 2013), `SAG` (Schmidt et al., 2017), `SDCA` (Richtárik & Takáč, 2014), `SAGA` (Defazio et al., 2014), `MISO` (Mairal, 2015), and `Katyusha` (Allen-Zhu, 2017). An overview of more advanced methods can be found in Gower et al. (2020). Notably, `SVRG` and `Katyusha` have been extended with loopless variants, namely `L-SVRG` and `L-Katyusha` (Kovalev et al., 2020; Qian et al., 2021). These loopless versions streamline the algorithms by eliminating the outer loop and introducing a biased coin-flip mechanism at each step. This simplification eases both the algorithms' structure and their analyses, while preserving their worst-case complexity bounds. `L-SVRG`, in particular, offers the advantage of setting the exit probability from the outer loop independently of the condition number, thus, enhancing both robustness and practical efficiency.

This technique of coin flipping allows to obtain variance reduction for the `CGD` algorithm. A relevant example is the `DIANA` algorithm proposed by Mishchenko et al. (2019). Its convergence was proved both in the convex and non-convex cases. Later, `MARINA` (Gorbunov et al., 2021) obtained the optimal convergence rates, improving in communication complexity compared to all previous first order methods. Finally, there is a line of work developing variance reduction in the federated setting using other methods and techniques (Chraibi et al., 2019; Hanzely & Richtárik, 2020; Dinh et al., 2020; Peng et al., 2022; Tyurin & Richtárik, 2022).

1.4 ORGANIZATION OF THE PAPER

The rest of the paper is organized as follows. Section 2 discusses the general mathematical framework. In particular, Section 2.2 lists the assumptions the we use later in the analysis and puts them in perspective with existing work. Section 3 presents the `det-MARINA` algorithm as well as the main theorem that guarantees convergence to a stationary point.. We show the superior theoretical performance of our algorithm compared to `MARINA` and `det-CGD` in Section 4. The details on the experimental setup and more plots can be found in the Appendix. Section 5 contains several plots, which confirm our theoretical findings. We conclude the main section of the paper in Section 6.

2 MATHEMATICAL FRAMEWORK

2.1 NOTATIONS

The standard Euclidean norm on $\mathbb{R}^d$ is defined as $\|\cdot\|$. We use $\mathbb{S}^d_{++}$ (resp. $\mathbb{S}^d_+$) to denote the positive definite (resp. semi-definite) cone of dimension d. $\mathbb{S}^d$ is used to denote all symmetric matrices of dimension d. We use the notation $\boldsymbol{I}_d$ to denote the identity matrix of size $d \times d$, and $\boldsymbol{O}_d$ to denote the zero matrix of size $d \times d$. Given $\boldsymbol{Q} \in \mathbb{S}^d_{++}$ and $x \in \mathbb{R}^d$, $\|x\|_{\boldsymbol{Q}} := \sqrt{x^\top \boldsymbol{Q} x} = \sqrt{\langle x, \boldsymbol{Q}x\rangle}$, where $\langle \cdot, \cdot \rangle$ is the standard Euclidean inner product on $\mathbb{R}^d$. For a matrix $\boldsymbol{A} \in \mathbb{S}^d$, we use $\lambda_{\max}(\boldsymbol{A})$ (resp. $\lambda_{\min}(\boldsymbol{A})$) to denote the largest (resp. smallest) eigenvalue of the matrix $\boldsymbol{A}$. For a function $f : \mathbb{R}^d \mapsto \mathbb{R}$, its gradient and its Hessian at a point $x \in \mathbb{R}^d$ are respectively denoted as $\nabla f(x)$ and $\nabla^2 f(x)$. For the sketch matrices $\boldsymbol{S}_i^k$ used in the algorithm, we use the superscript k to denote the iteration and subscript i to denote the client, the matrix $\boldsymbol{S}_i^k$ is thus sampled for client i in the k-th iteration from sine distribution $\mathcal{S}$. For any matrix $\boldsymbol{A} \in \mathbb{S}^d$, we use the notation $\operatorname{diag}(\boldsymbol{A}) \in \mathbb{S}^d$ to denote the diagonal of matrix $\boldsymbol{A}$.

2.2 ASSUMPTIONS AND CONDITIONS

In this section we present the assumptions we needed in order to analyze `det-MARINA`.

Assumption 1. (Lower Bound) *There exists $f^\star \in \mathbb{R}$ such that, $f(x) \geq f^\star$ for all $x \in \mathbb{R}^d$.*

This is a standard assumption in optimization, as otherwise the problem of minimizing the objective would not be correct mathematically. The same assumption is used in `MARINA`. We then need a matrix version of L-smoothness in order to proceed. Previously, Safaryan et al. (2021), Wang et al. (2022) used $\boldsymbol{L}$-matrix smoothness in the (strongly) convex setting to analyze some variants of the `DCGD`. Li et al. (2023) provided the analysis of sketched gradient descent under this assumption in the non-convex case. The assumption is formulated as follows,

Assumption 2. ($\boldsymbol{L}_i$-matrix smoothness) *Assume that each function f_i is $\boldsymbol{L}_i$-smooth for all $i \in [n] = \{1, 2, 3, ..., n\}$. That is for each function f_i, the following inequality holds:*

$$f_i(x) \leq f_i(y) + \langle \nabla f_i(y), x - y\rangle + \frac{1}{2}\|x - y\|^2_{\boldsymbol{L}_i}. \tag{4}$$

However, we do not want to use Assumption 2 in the analysis, as in the analysis of `det-MARINA` one needs to upper bound the squared difference of gradients by a multiple of squared difference of iterate. The latter is not implied from Assumption 2 for non-convex functions, as opposed to the convex ones. Instead, we introduce the matrix version of the L-Lipschitz continuous gradient assumption used in the analysis for `MARINA` of Gorbunov et al. (2021). Note that the "smoothness" the authors are referring to is indeed the Lipschitz continuous gradient assumption, instead of the standard smoothness assumption (Nesterov, 2003).

Definition 1. ($\boldsymbol{L}$-Lipschitz Gradient) *Assume that $f : \mathbb{R}^d \to \mathbb{R}$ is a continuously differentiable function and matrix $\boldsymbol{L} \in \mathbb{S}^d_{++}$. We say the gradient of f is $\boldsymbol{L}$-Lipschitz if*

$$\|\nabla f(x) - \nabla f(y)\|_{\boldsymbol{L}^{-1}} \leq \|x - y\|_{\boldsymbol{L}}, \quad \forall x, y \in \mathbb{R}^d. \tag{5}$$

This condition can be interpreted as follows. The gradient of f naturally belongs to the dual space of $\mathbb{R}^d$, as it is defined as a linear functional on $\mathbb{R}^d$. In the scalar case, ℓ_2-norm is self-dual, thus (5) reduces to the standard Lipschitz continuity of the gradient. However, with the matrix smoothness assumption, we are using the $\boldsymbol{L}$-norm for the iterates, which naturally induces the $\boldsymbol{L}^{-1}$-matrix norm for the gradients in the dual space. This insight, which is originally presented by Nemirovski & Yudin (1983), plays a key role in our analysis.

The following proposition provides us with a method to verify (5).

Proposition 1. *Given twice continuously differentiable function $f : \mathbb{R}^d \mapsto \mathbb{R}$ with bounded Hessian,*

$$\nabla^2 f(x) \preceq \boldsymbol{L}, \tag{6}$$

where $\boldsymbol{L} \in \mathbb{S}^d_{++}$ and the generalized inequality holds for any $x \in \mathbb{R}^d$. Then f satisfies (5) with the matrix $\boldsymbol{L}$.

Despite, being equivalent in the convex setting, (5) is slightly stronger compared to Assumption 2 in the non-convex case. See Appendix B for the properties of matrix smoothness. However, in practical terms, verifying Proposition 1 serves as the pipeline for confirming both conditions. Finally, we check that (5) is indeed an extension of the standard Lipschitz gradient assumption, as illustrated by the following remark.

Remark 1. *If we let $\boldsymbol{L} = L\boldsymbol{I}_d$, then (5) reduces to the standard L-Lipschitz continuous gradient assumption.*

In the following, we will assume that (5) is satisfied for component functions f_i.

Assumption 3. *Each function f_i is $\boldsymbol{L}_i$-gradient Lipschitz, while f is $\boldsymbol{L}$-gradient Lipschitz.*

In fact, the second half of the assumption is a consequence of the first one. Below, we formalize this claim.

Proposition 2. *If f_i is $\boldsymbol{L}_i$-gradient Lipschitz for every $i = 1, \ldots, n$, then function f has $\boldsymbol{L}$-Lipschitz gradient with $\boldsymbol{L} \in \mathbb{S}^d_{++}$ satisfying*

$$\frac{1}{n}\sum_{i=1}^{n} \lambda_{\max}\left(\boldsymbol{L}^{-1}\right) \cdot \lambda_{\max}\left(\boldsymbol{L}_i\right) \cdot \lambda_{\max}\left(\boldsymbol{L}_i\boldsymbol{L}^{-1}\right) = 1. \tag{7}$$

Nevertheless, the matrix $\boldsymbol{L}$ found according to Proposition 2 is only an estimate. In principle, there might exist a better $\boldsymbol{L}_f \preceq \boldsymbol{L}$ such that f has $\boldsymbol{L}_f$-Lipschitz gradient.

Remark 2. *In the scalar case, where $\boldsymbol{L} = L\boldsymbol{I}_d$, $\boldsymbol{L}_i = L_i\boldsymbol{I}_d$, the relation becomes*

$$L^2 = \frac{1}{n}\sum_{i=1}^{n} L_i^2.$$

This corresponds to the statement in Assumption 1.2 in Gorbunov et al. (2021).

3 The main result

In this section, we present our algorithm `det-MARINA` with the main convergence result. In addition, we compute both iteration and communication complexities and show that they are better than the ones of the `MARINA` algorithm, which serves as the prototype of our method. Along the iterations of the algorithms, we are constructing a sequence of vectors g^k which are stochastic estimators of $\nabla f(x^k)$. At each iteration, the server samples a Bernoulli random variable (coin flip) c_k and broadcasts it in parallel to the clients, along with the current gradient estimate g^k. Each client, then, does a `det-CGD`-type update with the stepsize $\boldsymbol{D}$ and a gradient estimate g^k. The next gradient estimate g^{k+1} is then computed. With a low probability, that is when $c_k = 1$, we take the g^{k+1} to be the full gradient $\nabla f(x^{k+1})$. Otherwise, we update it using the compressed gradient differences at each client. Below, is the pseudocode of the algorithm.

Algorithm 1 `det-MARINA`

Input: starting point x^0, stepsize matrix $\boldsymbol{D}$, probability $p \in (0, 1]$, number of iterations K
Initialize $g^0 = \nabla f(x^0)$
for $k = 0, 1, \ldots, K-1$ **do**
  Sample $c_k \sim \text{Be}(p)$
  Broadcast g^k to all workers
  for $i = 1, 2, \ldots$ in parallel **do**
    $x^{k+1} = x^k - \boldsymbol{D} \cdot g^k$
    Set $g_i^{k+1} = \begin{cases} \nabla f_i(x^{k+1}) & \text{if } c_k = 1 \\ g^k + \boldsymbol{S}_i^k\left(\nabla f_i(x^{k+1}) - \nabla f_i(x^k)\right) & \text{if } c_k = 0 \end{cases}$
  end for
  $g^{k+1} = \frac{1}{n}\sum_{i=1}^n g_i^{k+1}$
end for
Return: $\tilde{x}^K$ chosen uniformly at random from $\{x^k\}_{k=0}^{K-1}$

In the following theorem, we formulate the main result of this paper, which guarantees the convergence of Algorithm 1 under the abovementioned assumptions.

Theorem 1. *Assume that Assumptions 1 and 3 hold, and the following condition on stepsize matrix $\boldsymbol{D} \in \mathbb{S}^d_{++}$ holds,*

$$\boldsymbol{D}^{-1} \succeq \left(\frac{(1-p)\cdot R(\boldsymbol{D}, \mathcal{S})}{np} + 1\right)\boldsymbol{L}, \tag{8}$$

where

$$R(\boldsymbol{D},\mathcal{S}) := \frac{1}{n}\sum_{i=1}^{n}\lambda_{\max}\left(\mathbb{E}\left[\boldsymbol{S}_i^k\boldsymbol{D}\boldsymbol{S}_i^k\right]-\boldsymbol{D}\right)\cdot\lambda_{\max}\left(\boldsymbol{L}_i\right)\cdot\lambda_{\max}\left(\boldsymbol{L}^{-\frac{1}{2}}\boldsymbol{L}_i\boldsymbol{L}^{-\frac{1}{2}}\right). \tag{9}$$

Then, after K-iterations of det-MARINA*, we have*

$$\mathbb{E}\left[\left\|\nabla f(\tilde{x}^K)\right\|^2_{\frac{\boldsymbol{D}}{\det(\boldsymbol{D})^{1/d}}}\right] \leq \frac{2\left(f(x^0)-f^\star\right)}{\det(\boldsymbol{D})^{1/d}\cdot K}. \tag{10}$$

Here, $\tilde{x}^K$ is chosen uniformly randomly from the first K iterates of the algorithm.

Below we state several remarks regarding the interpretation of theorem.

Remark 3. *We notice that the the right-hand side of the algorithm vanishes with the number of iterations, thus solving the issue of the distributed* det-CGD*. Therefore,* det-MARINA *is indeed the variance reduced version of* det-CGD *in the distributed setting and has better convergence guarantees.*

Remark 4. *Theorem 1 implies the following iteration complexity for the algorithm. In order to get an ε^2 stationarity error, the algorithm requires K iterations, with*

$$K \geq \frac{2(f(x^0)-f^\star)}{\det(\boldsymbol{D})^{1/d}\cdot\varepsilon^2}.$$

Remark 5. *In the case where no compression is applied, that is we have $\boldsymbol{S}_i^k = \boldsymbol{I}_d$, the condition (8) reduces to*

$$\boldsymbol{D} \preceq \boldsymbol{L}^{-1}. \tag{11}$$

The latter is due to $\mathbb{E}\left[\boldsymbol{S}_i^k\boldsymbol{D}\boldsymbol{S}_i^k\right] = \boldsymbol{D}$, which results in $R(\boldsymbol{D},\mathcal{S}) = 0$. This is expected, since in the deterministic case det-MARINA *reduces to GD with matrix stepsize.*

The convergence condition and rate of matrix stepsize GD can be found in Li et al. (2023). Below we do a sanity check to verify that the convergence condition for scalar MARINA can be obtained.

Remark 6. *Let us consider the scalar case. That is*

$$\boldsymbol{L}_i = L_i\boldsymbol{I}_d, \quad \boldsymbol{L} = L\boldsymbol{I}_d, \quad \boldsymbol{D} = \gamma\boldsymbol{I}_d \quad \textit{and} \quad \omega = \lambda_{\max}\left(\mathbb{E}\left[\left(\boldsymbol{S}_i^k\right)^\top\boldsymbol{S}_i^k\right]\right)-1. \tag{12}$$

Then, the condition (8) reduces to

$$\frac{\gamma(1-p)\omega L^2}{np} - \frac{1}{\gamma} + L \leq 0. \tag{13}$$

One can check that the below bound implies (13)

$$\gamma \leq \left[L\left(1+\sqrt{\frac{(1-p)\omega}{pn}}\right)\right]^{-1}.$$

The latter coincides with the stepsize condition of the convergence result of scalar MARINA. Now let us look at the right-hand side of (10). We notice that it decreases in terms of the determinant of the stepsize matrix. Therefore, one needs to solve the following optimization problem to find the optimal stepsize:

$$\begin{array}{ll} \text{minimize} & \log\det(\boldsymbol{D}^{-1}) \\ \text{subject to} & \boldsymbol{D} \text{ satisfying (8).} \end{array}$$

The solution of this constrained minimization problem on $\mathbb{S}^d_{++}$ is not explicit. In theory, one may show that the constraint (8) is convex and attempt to solve the problem numerically. However, as stressed by Li et al. (2023), the similar stepsize condition for det-CGD is not easily computed using solvers like CVXPY (Diamond & Boyd, 2016). Instead, we may relax the problem to certain linear subspaces of $\mathbb{S}^d_{++}$. In particular, we fix a matrix $\boldsymbol{W} \in \mathbb{S}^d_{++}$, and define $\boldsymbol{D} := \gamma\boldsymbol{W}$. Then, the condition on the matrix $\boldsymbol{D}$ becomes a condition for the scalar γ, which is given in the following corollary.

Corollary 1. *Let $\boldsymbol{W} \in \mathbb{S}^d_{++}$, defining $\boldsymbol{D} := \gamma \cdot \boldsymbol{W}$, where $\gamma \in \mathbb{R}_+$. then the condition in (8) reduces to the following condition on γ*

$$\gamma \leq \frac{2\lambda_{\boldsymbol{W}}}{1+\sqrt{1+4\alpha\beta \cdot \Lambda_{\boldsymbol{W},\mathcal{S}}\lambda_{\boldsymbol{W}}}}, \tag{14}$$

where $\Lambda_{\boldsymbol{W},\mathcal{S}} = \lambda_{\max}\left(\mathbb{E}\left[\boldsymbol{S}_i^k \boldsymbol{W} \boldsymbol{S}_i^k\right] - \boldsymbol{W}\right)$, $\lambda_{\boldsymbol{W}} = \lambda_{\max}^{-1}\left(\boldsymbol{W}^{\frac{1}{2}}\boldsymbol{L}\boldsymbol{W}^{\frac{1}{2}}\right)$ and

$$\alpha = \frac{1-p}{np}; \quad \beta = \frac{1}{n}\sum_{i=1}^{n}\lambda_{\max}\left(\boldsymbol{L}_i\right) \cdot \lambda_{\max}\left(\boldsymbol{L}^{-1}\boldsymbol{L}_i\right). \tag{15}$$

This means that for every fixed $\boldsymbol{W}$, we can find the optimal scaling γ. In the next section, we will use this corollary to compute the communication complexity of our algorithm and to compare it with MARINA.

4 Comparison of Complexities

The following corollary states the iteration complexity for det-MARINA with $\boldsymbol{W} = \boldsymbol{L}^{-1}$.

Corollary 2. *If we take $\boldsymbol{W} = \boldsymbol{L}^{-1}$, then the condition (14) on γ is given by*

$$\gamma \leq \frac{2}{1+\sqrt{1+4\alpha\beta \cdot \Lambda_{\boldsymbol{L}^{-1},\mathcal{S}}}}. \tag{16}$$

In order to satisfy, ε-stationarity, that is $\mathbb{E}\left[\left\|\nabla f(\tilde{x}^K)\right\|^2_{\boldsymbol{D}/\det(\boldsymbol{D})^{1/d}}\right] \leq \varepsilon^2$, we require

$$K \geq K_0 = \mathcal{O}\left(\frac{\Delta_0 \cdot \det(\boldsymbol{L})^{\frac{1}{d}}}{\varepsilon^2} \cdot \left(1+\sqrt{1+4\alpha\beta \cdot \Lambda_{\boldsymbol{L}^{-1},\mathcal{S}}}\right)\right), \tag{17}$$

where $\Delta_0 = f(x^0) - f(x^\star)$. Moreover, this iteration complexity is always better than the one of MARINA.

The proof can be found in the Appendix. In fact, we can show that in cases where we fix $\boldsymbol{W} = \boldsymbol{I}_d$ and $\boldsymbol{W} = \operatorname{diag}^{-1}(\boldsymbol{L})$, the same conclusion also holds, relevant details can be found in Appendix C.2. This essentially means that det-MARINA can always have a "larger" stepsize compared to MARINA, which leads to a better iteration complexity. In addition, because we are using the same compressor for those two algorithms, the communication complexity of det-MARINA is also provably better than that of MARINA.

We also give an analysis of the communication complexity of our algorithm as our main concern here is the communication complexity. We first give the following definition on the expected density, which is used to analyze the communication complexity in Gorbunov et al. (2021). The original definition is given for any unbiased compressors. However, we are focusing on sketches in this paper, so we only restrict the definition to sketches.

Definition 2. *For a given sketch matrix $\boldsymbol{S} \in \mathbb{S}^d_+$, the expected density is defined as*

$$\zeta_{\boldsymbol{S}} = \sup_{x \in \mathbb{R}^d} \mathbb{E}\left[\left\|\boldsymbol{S}x\right\|_0\right], \tag{18}$$

where $\|x\|_0$ denotes the number of non-zero components of $x \in \mathbb{R}^d$.

We can easily obtain the expected density for some commonly seen sketches, for example for rand-τ sketches, we have $\zeta_{\texttt{rand-}\tau} = \tau$. The latter means, that in average the clients communicate τ coordinates at each iteration. Below, we state the communication complexity of det-MARINA with $\boldsymbol{D}^*_{\boldsymbol{L}^{-1}}$ and the rand-τ compressor.

Corollary 3. *Assume that we are using sketch $\boldsymbol{S} \sim \mathcal{S}$ with expected density $\zeta_{\mathcal{S}}$, suppose we are running* det-MARINA *with probability p and we use the optimal stepsize matrix with respect to $\boldsymbol{W} = \boldsymbol{L}^{-1}$, then the overall communication complexity here is given by $\mathcal{O}\left((Kp+1)d + (1-p)K\zeta_{\mathcal{S}}\right)$. Specifically, if we pick $p = \zeta_{\mathcal{S}}/d$, then the communication complexity is given by*

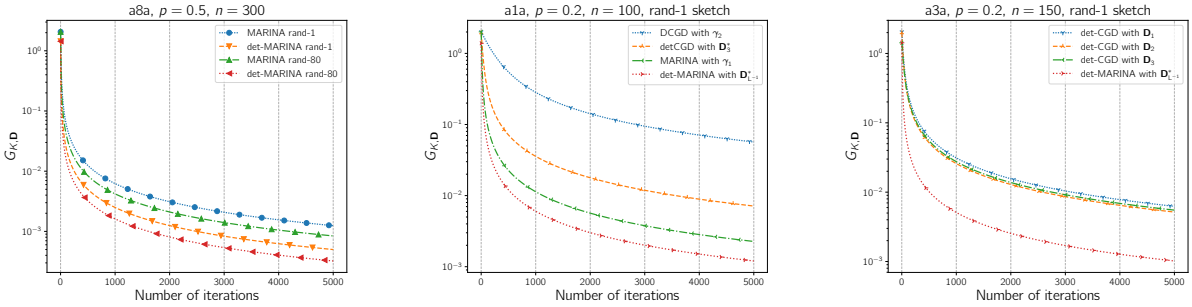
$$\mathcal{O}\left(d + \frac{\Delta_0 \cdot \det(\boldsymbol{L})^{\frac{1}{d}}}{\varepsilon^2} \cdot \left(\zeta_{\mathcal{S}} + \sqrt{\frac{\beta \cdot \Lambda_{\boldsymbol{L}^{-1},\mathcal{S}}}{n} \cdot \zeta_{\mathcal{S}}(d-\zeta_{\mathcal{S}})}\right)\right). \tag{19}$$

Notice that in case where no compression is applied, the communication complexity (resp. iteration complexity) reduces to $\mathcal{O}\left(d\Delta_0 \cdot \det(\boldsymbol{L})^{\frac{1}{d}}/\varepsilon^2\right)$ (resp. $\mathcal{O}\left(\Delta_0 \cdot \det(\boldsymbol{L})^{\frac{1}{d}}/\varepsilon^2\right)$), this coincides with the rate of matrix stepsize GD (see Li et al. (2023)). This implies that the dependence on ε is not possible to be improved further since GD is optimal in first order methods Carmon et al. (2020).

5 EXPERIMENTS

In this section, we conduct experimental comparisons between det−MARINA and the current state-of-the-art methods. The results presented in both Figure 1(a) and Figure 2 confirm that det−MARINA indeed surpasses MARINA in terms of iteration complexity and communication complexity.

Compared to det−CGD, which serves as the non-variance-reduced counterpart of det−MARINA, our algorithm demonstrates superior performance, as evident from Figure 1(c) and Figure 4, as predicted by our theory. An overall comparison between two non-variance-reduced methods and two variance-reduced methods is presented in Figure 1(b). This plot highlights the significance of combining variance reduction techniques with matrix stepsize and matrix smoothness for improved optimization performance.



(a) (b) (c)

Figure 1: These plots confirm that det−MARINA improves on all previous algorithms, including DCGD, det−CGD and MARINA. For more plots and the details of the experimental setup can be found in Appendix G.

6 CONCLUSION

We proposed det−MARINA, as variance reduced alternative to det−CGD, and we show that it outperforms both det−CGD and MARINA. There are numerous directions to extend the matrix-stepsized non-convex algorithms. Here we list some of them.

Extension to det−CGD2. A variant of det−CGD, called det−CGD2, was also proposed by Li et al. (2023). This algorithm, has the same structure as (det−CGD) with the sketch and stepsize interchanged. It was shown, that this algorithm has explicit stepsize condition in the single node setting. In Appendix E, we propose the variance reduced extension of the distributed det−CGD2.

Extension to DASHA. However, besides MARINA, there are other existing techniques of performing variance reduction in a non-convex setting for compressed gradient methods, such as DASHA, which offers better practicality, as it always sends compressed gradients and do not need synchronize among all the nodes, according to Tyurin & Richtárik (2022). We want to emphasize here the way we extend det−CGD to its variance-reduced counterpart is not limited to MARINA type variance reduction. The same techniques used in the proof are also applicable if we want to extend det−CGD to its DASHA type variance reduced counterpart. However, we leave this as a future work.

Other directions. i) In this paper, we have only considered (linear) sketches as the compression operator. However, there exists a variety of compressors which are useful in practice that do not fall into this category. Extending det−CGD and det−MARINA for general unbiased compressors is a promising future work direction. ii) Our motivation for using a matrix stepsize is partially inspired by second-order methods, where matrix stepsize $\boldsymbol{D}$ roughly estimates the inverse of the Hessian. Additionally, given recent successes with adaptive stepsizes (e.g., Loizou et al. (2021); Orvieto et al. (2022); Schaipp et al. (2023)), designing an adaptive matrix stepsize tailored to our case could be viable. iii) Finally, recent advances in federated learning (Philippenko & Dieuleveut, 2020; Gruntkowska et al., 2022) have shown that server-to-client compression is important. Extending our results for the bidirectional federated learning is worth human attention.

REFERENCES

Mehiddin Al-Baali and H Khalfan. An overview of some practical quasi-newton methods for unconstrained optimization. *Sultan Qaboos University Journal for Science [SQUJS]*, 12(2):199–209, 2007.

Mehiddin Al-Baali, Emilio Spedicato, and Francesca Maggioni. Broyden's quasi-Newton methods for a nonlinear system of equations and unconstrained optimization: a review and open problems. *Optimization Methods and Software*, 29(5):937–954, 2014.

Foivos Alimisis, Peter Davies, and Dan Alistarh. Communication-efficient distributed optimization with quantized preconditioners. In *International Conference on Machine Learning*, pp. 196–206. PMLR, 2021.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. *Advances in neural information processing systems*, 30, 2017.

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, 2017.

Rajendra Bhatia. *Positive definite matrices*. Princeton university press, 2009.

Charles G Broyden. A class of methods for solving nonlinear simultaneous equations. *Mathematics of computation*, 19 (92):577–593, 1965.

Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on optimization*, 20(4):1956–1982, 2010.

Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points i. *Mathematical Programming*, 184(1-2):71–120, 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.

Sélim Chraibi, Ahmed Khaled, Dmitry Kovalev, Peter Richtárik, Adil Salim, and Martin Takáč. Distributed fixed point methods with compressed iterates. *arXiv preprint arXiv:1912.09925*, 2019.

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

Marina Danilova, Pavel Dvurechensky, Alexander Gasnikov, Eduard Gorbunov, Sergey Guminov, Dmitry Kamzolov, and Innokentiy Shibaev. Recent theoretical advances in non-convex optimization. In *High-Dimensional Optimization and Probability: With a View Towards Data Science*, pp. 79–163. Springer, 2022.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

John E Dennis, Jr and Jorge J Moré. Quasi-Newton methods, motivation and theory. *SIAM review*, 19(1):46–89, 1977.

Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

Canh T Dinh, Nguyen H Tran, Tuan Dung Nguyen, Wei Bao, Albert Y Zomaya, and Bing B Zhou. Federated learning with proximal stochastic variance reduced gradient algorithms. In *Proceedings of the 49th International Conference on Parallel Processing*, pp. 1–11, 2020.

Darina Dvinskikh, Aleksandr Ogaltsov, Alexander Gasnikov, Pavel Dvurechensky, Alexander Tyurin, and Vladimir Spokoiny. Adaptive gradient descent for convex and non-convex stochastic optimization. *arXiv preprint arXiv:1911.08380*, 2019.

Eduard Gorbunov, Konstantin P Burlachenko, Zhize Li, and Peter Richtárik. Marina: Faster non-convex distributed learning with compression. In *International Conference on Machine Learning*, pp. 3788–3798. PMLR, 2021.

Robert M Gower, Mark Schmidt, Francis Bach, and Peter Richtárik. Variance-reduced methods for machine learning. *Proceedings of the IEEE*, 108(11):1968–1983, 2020.

Robert Mansel Gower, Nicolas Loizou, Xun Qian, Alibek Sailanbayev, Egor Shulgin, and Peter Richtárik. Sgd: General analysis and improved rates. In *International Conference on Machine Learning*, pp. 5200–5209. PMLR, 2019.

William B Gragg and Richard A Tapia. Optimal error bounds for the Newton–Kantorovich theorem. *SIAM Journal on Numerical Analysis*, 11(1):10–13, 1974.

Kaja Gruntkowska, Alexander Tyurin, and Peter Richtárik. EF21-P and Friends: Improved Theoretical Communication Complexity for Distributed Optimization with Bidirectional Compression, 2022. URL https://arxiv.org/abs/2209.15218.

SV Guminov, Yu E Nesterov, PE Dvurechensky, and AV Gasnikov. Accelerated primal-dual gradient descent with linesearch for convex, nonconvex, and nonsmooth optimization problems. In *Doklady Mathematics*, volume 99, pp. 125–128. Springer, 2019.

Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*, 2020.

Samuel Horváth, Chen-Yu Ho, Ludovit Horvath, Atal Narayan Sahu, Marco Canini, and Peter Richtárik. Natural compression for distributed deep learning. In *Mathematical and Scientific Machine Learning*, pp. 129–141. PMLR, 2022.

Samuel Horváth, Dmitry Kovalev, Konstantin Mishchenko, Peter Richtárik, and Sebastian Stich. Stochastic distributed learning with gradient quantization and double-variance reduction. *Optimization Methods and Software*, 38(1): 91–106, 2023.

Rustem Islamov, Xun Qian, and Peter Richtárik. Distributed second order methods with fast rates and compressed communication. In *International conference on machine learning*, pp. 4617–4628. PMLR, 2021.

Sashank J Reddi, Suvrit Sra, Barnabas Poczos, and Alexander J Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. *Advances in neural information processing systems*, 29, 2016.

Prateek Jain, Purushottam Kar, et al. Non-convex optimization for machine learning. *Foundations and Trends® in Machine Learning*, 10(3-4):142–363, 2017.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

Ahmed Khaled and Peter Richtárik. Better theory for sgd in the nonconvex world. *arXiv preprint arXiv:2002.03329*, 2020.

Sarit Khirirat, Hamid Reza Feyzmahdavian, and Mikael Johansson. Distributed learning with compressed gradients. *arXiv preprint arXiv:1806.06573*, 2018.

Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Dmitry Kovalev, Samuel Horváth, and Peter Richtárik. Don't jump through hoops and remove those loops: Svrg and katyusha are better without the outer loop. In *Algorithmic Learning Theory*, pp. 451–467. PMLR, 2020.

Hanmin Li, Avetik Karagulyan, and Peter Richtárik. Det-cgd: Compressed gradient descent with matrix stepsizes for non-convex optimization. *arXiv preprint arXiv:2305.12568*, 2023.

Zhize Li, Dmitry Kovalev, Xun Qian, and Peter Richtárik. Acceleration for compressed gradient descent in distributed and federated optimization. *arXiv preprint arXiv:2002.11364*, 2020.

Zhize Li, Hongyan Bao, Xiangliang Zhang, and Peter Richtárik. PAGE: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In *International conference on machine learning*, pp. 6286–6295. PMLR, 2021.

Nicolas Loizou, Sharan Vaswani, Issam Hadj Laradji, and Simon Lacoste-Julien. Stochastic polyak step-size for sgd: An adaptive learning rate for fast convergence. In *International Conference on Artificial Intelligence and Statistics*, pp. 1306–1314. PMLR, 2021.

Julien Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. *SIAM Journal on Optimization*, 25(2):829–855, 2015.

Artavazd Maranjyan, Mher Safaryan, and Peter Richtárik. GradSkip: Communication-Accelerated Local Gradient Methods with Better Computational Complexity. *arXiv preprint arXiv:2210.16402*, 2022.

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera 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. URL http://arxiv.org/abs/1602.05629.

George J Miel. Majorizing sequences and error bounds for iterative methods. *Mathematics of Computation*, 34(149): 185–202, 1980.

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 Richtarik. ProxSkip: Yes! Local gradient steps provably lead to communication acceleration! Finally! In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15750–15769. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/mishchenko22b.html.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.

Arkadi Semenovič Nemirovski and David Borisovich Yudin. Problem complexity and method efficiency in optimization. *Wiley-Interscience, ISSN 0277-2698*, 1983.

Yu Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

Antonio Orvieto, Simon Lacoste-Julien, and Nicolas Loizou. Dynamics of sgd with stochastic polyak stepsizes: Truly adaptive variants and convergence to exact solution. *Advances in Neural Information Processing Systems*, 35: 26943–26954, 2022.

Jie Peng, Zhaoxian Wu, Qing Ling, and Tianyi Chen. Byzantine-robust variance-reduced federated learning over distributed non-iid data. *Information Sciences*, 616:367–391, 2022.

Constantin Philippenko and Aymeric Dieuleveut. Bidirectional compression in heterogeneous settings for distributed or federated learning with partial participation: tight convergence guarantees. *arXiv preprint arXiv:2006.14591*, 2020.

Xun Qian, Zheng Qu, and Peter Richtárik. L-svrg and l-katyusha with arbitrary sampling. *The Journal of Machine Learning Research*, 22(1):4991–5039, 2021.

Peter Richtárik and Martin Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1-2):1–38, 2014.

Mher Safaryan, Filip Hanzely, and Peter Richtárik. Smoothness matrices beat smoothness constants: Better communication compression techniques for distributed optimization. *Advances in Neural Information Processing Systems*, 34: 25688–25702, 2021.

Mher Safaryan, Rustem Islamov, Xun Qian, and Peter Richtarik. Fednl: Making newton-type methods applicable to federated learning. In *International Conference on Machine Learning*, pp. 18959–19010. PMLR, 2022a.

Mher Safaryan, Egor Shulgin, and Peter Richtárik. Uncertainty principle for communication compression in distributed and federated learning and the search for an optimal compressor. *Information and Inference: A Journal of the IMA*, 11(2):557–580, 2022b.

Fabian Schaipp, Robert M Gower, and Michael Ulbrich. A stochastic proximal polyak step size. *arXiv preprint arXiv:2301.04935*, 2023.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162:83–112, 2017.

Egor Shulgin and Peter Richtárik. Shifted compression framework: Generalizations and improvements. In *Uncertainty in Artificial Intelligence*, pp. 1813–1823. PMLR, 2022.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

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

Bokun Wang, Mher Safaryan, and Peter Richtárik. Theoretically better and numerically faster distributed optimization with smoothness-aware quantization techniques. *Advances in Neural Information Processing Systems*, 35:9841–9852, 2022.

Shusen Wang, Fred Roosta, Peng Xu, and Michael W Mahoney. Giant: Globally improved approximate newton method for distributed optimization. *Advances in Neural Information Processing Systems*, 31, 2018.

Stephen J Wright. Coordinate descent algorithms. *Mathematical programming*, 151(1):3–34, 2015.

Tetsuro Yamamoto. A convergence theorem for newton-like methods in banach spaces. *Numerische Mathematik*, 51: 545–557, 1987.

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. IEEE, 2020a.

Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? *Advances in Neural Information Processing Systems*, 33: 15383–15393, 2020b.