Accelerated Distributed Optimization with Compression and Error Feedback

Yuan Gao[†‡*] **Anton Rodomanov**[†*] **Jeremy Rack**[†‡*] **Sebastian U. Stich**[†*]
[†] CISPA Helmholtz Center for Information Security, Germany
[‡] Universität des Saarlandes, Germany
[*] {yuan.gao, anton.rodomanov, michael.rack, stich}@cispa.de

Abstract

Modern machine learning tasks often involve massive datasets and models, necessitating distributed optimization algorithms with reduced communication overhead. Communication compression, where clients transmit compressed updates to a central server, has emerged as a key technique to mitigate communication bottlenecks. However, the theoretical understanding of stochastic distributed optimization with contractive compression remains limited, particularly in conjunction with Nesterov acceleration—a cornerstone for achieving faster convergence in optimization. In this paper, we propose a novel algorithm, ADEF (**A**ccelerated **D**istributed **E**rror **F**eedback), which integrates Nesterov acceleration, contractive compression, error feedback, and gradient difference compression. We prove that ADEF achieves the first accelerated convergence rate for stochastic distributed optimization with contractive compression in the general convex regime. Numerical experiments validate our theoretical findings and demonstrate the practical efficacy of ADEF in reducing communication costs while maintaining fast convergence.

1 INTRODUCTION

Gradient methods are foundational to training modern machine learning models. Moreover, with the rapidly growing sizes of the datasets and models, distributed training has become a necessity as accumulating the entire dataset on a single central machine is simply infeasible. Many of the recent breakthrough models are trained in such a distributed fashion, e.g. large language models (Shoeybi et al., 2019), generative models (Ramesh et al., 2021, 2022), and others (Wang et al., 2020), where data are distributed across different clients/workers and model updates are coordinated by a parameter server. Another instance of distributed optimization is the federated learning setting (Konečný et al., 2016; Kairouz et al., 2019), where clients (e.g., edge devices or hospitals) jointly train a model without sharing their local data.

One of the key challenges in distributed optimization is the *communication bottleneck*—the cost of transmitting large model updates between clients and the server (Seide et al., 2014; Strom, 2015). To address this, **communication compression** has emerged as a practical solution, enabling updates to be transmitted in a compressed format to reduce communication overhead. Contractive compression (see Definition 2.1), despite being potentially biased, has been shown to outperform unbiased compression schemes in practice (Lin et al., 2018b; Sun et al., 2019; Vogels et al., 2019) and offers favorable theoretical properties (Stich et al., 2018; Albasyoni et al., 2020; Beznosikov et al., 2023). However, naive aggregation of compressed updates can lead to divergence (Beznosikov et al., 2023). To mitigate this, **Error Feedback (EF)** (Seide et al., 2014) has been proposed as an effective remedy, compensating for compression errors (Stich et al., 2018; Karimireddy et al., 2019). EF and its variants (e.g., PowerSGD by Vogels et al. (2019)) have been widely adopted in practice, integrated into popular deep learning frameworks such as PyTorch (Paszke et al., 2019), and deployed to train state-of-the-art transformer models (Ramesh et al., 2021).

Nesterov acceleration (Nesterov, 1983) is one of the most important algorithmic tools to achieve faster convergence rates for deterministic and stochastic first-order methods in the convex regime. However, its

Proceedings of the 29[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

integration with EF and contractive compression in the general convex regime remains largely unexplored. Existing works on accelerated methods with compression primarily focus on deterministic or strongly convex settings (Li et al., 2020; Qian et al., 2021a; Bylinkin and Beznosikov, 2024), rely on unbiased compression (Li and Richtárik, 2021), or effectively function as uncompressed methods (He et al., 2023).[1] While it might be possible to apply the Catalyst scheme (Lin et al., 2018a) on top of an unaccelerated EF variant, from a practical perspective, the Catalyst scheme requires much more complicated hyperparameter tuning (Lin et al., 2018a; Mairal, 2019) while delivering weaker performance than direct acceleration methods (Mairal, 2019). From a theoretical perspective, ***direct*** acceleration for distributed stochastic optimization with contractive compression in the general convex regime is still unexplored, and in this work, we aim to fill this significant gap in the literature.

1.1 CONTRIBUTIONS

In this paper, we address the aforementioned gap by proposing ADEF (**A**ccelerated **D**istributed **E**rror **F**eedback), which integrates Nesterov acceleration, contractive compression, and bias-corrected EF in the stochastic and general convex regime. Our contributions are as follows:

Algorithm: We introduce ADEF, a single-loop algorithm for distributed optimization that combines Nesterov acceleration with contractive compression, error feedback, and gradient difference compression, designed for the practically common constant mini-batch regime.

General theoretical framework (acceleration with inexact updates): We develop a general framework for accelerated methods with inexact updates, which cleanly handles errors from stochasticity and compression. This framework is modular and may be useful for analyzing other optimization algorithms under noisy or compressed updates.

Convergence guarantees: Applying the framework, we show that ADEF achieves accelerated convergence rates under contractive compression in the general convex setting without bounded-heterogeneity assumptions. In particular, our results clarify how acceleration can be preserved in the constant-batch, single-round regime, thereby bridging a long-standing gap in the theoretical understanding of communication-efficient optimization and acceleration.

Empirical validation: We evaluate ADEF against compressed and uncompressed baselines on convex synthetic problems and image classification tasks. We report both communication rounds and transmitted bits, and observe that ADEF reduces client-to-server communication while maintaining competitive accuracy.

By bridging the gap between communication-efficient distributed optimization and acceleration, ADEF opens new avenues for scalable and efficient training of large-scale ML models.

[1] NEOLITHIC uses *repeated communication rounds within each iteration* to reduce compression error; see Section 2.1 and Appendix G for details.

2 COMMUNICATION COMPRESSION AND ACCELERATED METHODS

In this paper, we study distributed optimization algorithms equipped with communication compression. In particular, we consider the class of contractive compression operators:

Definition 2.1. We say that a (possibly randomized) mapping $\mathcal{C}\colon \mathbb{R}^d \to \mathbb{R}^d$ is a contractive compression if for some constant $0 < \delta \leq 1$ it holds

$$\mathbb{E}\left[\|\mathcal{C}(\mathbf{x}) - \mathbf{x}\|^2\right] \leq (1-\delta)\|\mathbf{x}\|^2 \quad \forall \mathbf{x} \in \mathbb{R}^d. \tag{1}$$

The classic example satisfying Definition 2.1 is Top-K (Stich et al., 2018). Top-K keeps the K largest entries in terms of absolute value and zeros out the rest and it is a biased compression. The class of contractive compressors also includes sparsification (Alistarh et al., 2018; Stich et al., 2018) and quantization operators (Wen et al., 2017; Alistarh et al., 2017; Bernstein et al., 2018; Horváth et al., 2019), and many others (Zheng et al., 2019; Beznosikov et al., 2020; Vogels et al., 2019; Safaryan et al., 2022; Islamov et al., 2023).

In Table 1 we compare our method to some key existing works most relevant to our setting; stochastic distributed optimization with contractive compression in the general convex regime. We are interested in the number of communication rounds from the clients to the server. In Table 1 we present the rates with a free parameter B the batch size used at each iteration. In all cases, the batch size B simply divides the variance σ^2 by B. We note that in the original presentation, He et al. (2023) presented the $\widetilde{\mathcal{O}}\left(\frac{R_0^2\sigma^2}{n\varepsilon^2} + \frac{\sqrt{LR_0^2}}{\delta\sqrt{\varepsilon}}\right)$ rate for NEOLITHIC when $B = \Omega(\frac{1}{\delta}\log\frac{1}{\varepsilon})$.

2.1 ACCELERATED METHODS WITH COMPRESSION

Now we survey the existing works on accelerated methods with communication compression. Compared to

Table 1: Comparison of key existing works on stochastic distributed optimization with contractive compression in the general convex regime. We present the rates when the batch size is B. $R_0^2 \coloneqq \|\mathbf{x}_0 - \mathbf{x}^\star\|^2$. For L and σ^2, see Section 3. For simplicity, we assume that $L = \ell = L_{\max}$.

Algorithm	**Acc**[(a)]	**No BDH**[(b)]	# Comm. Rounds	# Local Oracle Calls
EF (Karimireddy et al., 2019)	No	No	$\mathcal{O}\left(\frac{R_0^2\sigma^2}{nB\varepsilon^2} + \frac{\sqrt{L}R_0^2(\frac{\zeta}{\sqrt{\delta}}+\frac{\sigma}{\sqrt{B}})}{\sqrt{\delta}\varepsilon^{3/2}} + \frac{LR_0^2}{\delta\varepsilon}\right)$	$\mathcal{O}\left(\frac{R_0^2\sigma^2}{n\varepsilon^2} + \frac{\sqrt{L}R_0^2(\frac{B\zeta}{\sqrt{\delta}}+\sigma\sqrt{B})}{\sqrt{\delta}\varepsilon^{3/2}} + \frac{BLR_0^2}{\delta\varepsilon}\right)$
EControl (Gao et al., 2024)	No	Yes	$\mathcal{O}\left(\frac{R_0^2\sigma^2}{nB\varepsilon^2} + \frac{\sqrt{L}R_0^2\sigma}{\delta^2\sqrt{B}\varepsilon^{3/2}} + \frac{LR_0^2}{\delta\varepsilon}\right)$	$\mathcal{O}\left(\frac{R_0^2\sigma^2}{n\varepsilon^2} + \frac{\sqrt{L}R_0^2\sigma\sqrt{B}}{B\delta^2\varepsilon^{3/2}} + \frac{LR_0^2}{\delta\varepsilon}\right)$
NEOLITHIC[(c)] (He et al., 2023)	Yes	No	$\widetilde{\mathcal{O}}\left(\frac{R_0^2\sigma^2}{\delta nB\varepsilon^2} + \frac{\sqrt{LR_0^2}}{\delta\sqrt{\varepsilon}}\right)$[(d)]	$\mathcal{O}\left(\frac{R_0^2\sigma^2}{n\varepsilon^2} + \frac{B\sqrt{LR_0^2}}{\sqrt{\varepsilon}}\right)$
ADEF new	Yes	Yes	$\mathcal{O}\left(\frac{R_0^2\sigma^2}{nB\varepsilon^2} + \frac{\sqrt{L}R_0^2\sigma}{\delta^2\sqrt{B}\varepsilon^{3/2}} + \frac{\sqrt{LR_0^2}}{\delta^2\sqrt{\varepsilon}}\right)$	$\mathcal{O}\left(\frac{R_0^2\sigma^2}{n\varepsilon^2} + \frac{\sqrt{L}R_0^2\sigma\sqrt{B}}{\delta^2\varepsilon^{3/2}} + \frac{B\sqrt{LR_0^2}}{\delta^2\sqrt{\varepsilon}}\right)$

(a) Acc stands for accelerated methods.
(b) BDH stands for Bounded Data Heterogeneity Assumption: for EF, this is Bounded Gradient Similarity: $\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \zeta^2, \forall \mathbf{x} \in \mathbb{R}^d$. See also Assumption E.1. For NEOLITHIC, this is Bounded Local Objective Gap at Optimum: $\frac{1}{n}\sum_{i=1}^n f_i(\mathbf{x}^\star) - f^\star \leq \zeta^\star$.
(c) He et al. (2023) presented their rates when batch size is $\Omega(\frac{1}{\delta}\log\kappa)$ where κ is the total number of updates at the server. Most other works present their rates when the batch size is some constant independent of δ.
(d) $\widetilde{\mathcal{O}}$ hides logarithmic factors in $1/\varepsilon$.

the rich literature on the unaccelerated EF and its variants (see Appendix A), theory for accelerated methods (Nesterov, 1983) with communication compression is still lacking, with only a handful available (Murata and Suzuki, 2019; Li et al., 2020; Qian et al., 2021a; Li and Richtárik, 2021; Qian et al., 2023; He et al., 2023; Bylinkin and Beznosikov, 2024). This is particularly the case with contractive compression, where some work (He et al., 2023) identifies the potential biasedness in the updates as a forbidding factor, due to the negative results of (Devolder et al., 2014).

Unbiased Compression: ADIANA (Li et al., 2020) and CANITA (Li and Richtárik, 2021) are both restricted to unbiased compressions. The S-NAG-EF proposed in (Murata and Suzuki, 2019) even requires a more specific class of randomized unbiased compressors. We also note that a very recent work by Gupta et al. (2024) essentially rediscovered compressed (accelerated) GD with unbiased compressors (Li et al., 2020), under the notion of relative noise. The unbiasedness assumption is very strong, and it is known in practice that contractive compressors are superior (Vogels et al., 2019; Albasyoni et al., 2020; Beznosikov et al., 2023), rendering these theories unsatisfactory.

Lack of Theory for General Convex Case: ADIANA, ECLK (Qian et al., 2021a) and EF-OLGA (Bylinkin and Beznosikov, 2024) only work for the strongly-convex regime. While a reduction via the regularization method is possible, it typically incurs a logarithmic factor. Beyond that, seeking a direct accelerated method for the general convex setting remains an important theoretical challenge.

Requires Full Gradient: As we have seen before, mixing stochastic gradients with compression can already be problematic in the unaccelerated case. ADIANA, ECLK, CANITA and EF-OLGA all only work with full gradients. Evidence suggests that ADIANA and CANITA's gradient difference compression mechanism might fail to converge with stochastic gradients (Fatkhullin et al., 2023).

Finite Sum Stochasticity: Some other works consider the stochastic finite sum optimization problem, which is a different setting than ours. ECLK, EF-OLGA, and Catalyst accelerated EC-LSVRG (Qian et al., 2023) combine EF with SVRG-style methods. These methods require periodic accesses to *full* local gradients, or worse, both periodic accesses and communications of *full* global gradients.

Repeated Communication: NEOLITHIC (Huang et al., 2022; He et al., 2023) does not utilize any additional algorithmic tools on top of a naively compressed Nesterov accelerated method. It uses $\widetilde{\Omega}(\frac{1}{\delta})$ rounds of communications between each update to account for compression errors. Such a repeated communication procedure has been shown to perform poorly in practice (Fatkhullin et al., 2023). In many real-life applications, we do not know the precise δ value of the compression and cannot implement such a procedure. Moreover, we discuss in Appendix G that sending a vector $\mathbf{m}$ via this repeated communication procedure is ***more expensive*** than simply sending $\mathbf{m}$ uncompressed, rendering NEOLITHIC an uncompressed method in disguise.

3 PROBLEM FORMULATION AND ASSUMPTIONS

In this paper, we consider the following distributed stochastic optimization problem:

$$\mathbf{x}^\star \coloneqq \arg\min_{\mathbf{x}\in\mathbb{R}^d} \Big[f(x) \coloneqq \frac{1}{n}\sum_{i=1}^n f_i(\mathbf{x}) \Big], \tag{2}$$

where $\mathbf{x}$ are the parameters of a model that we train. The objective function $f : \mathbb{R}^d \to \mathbb{R}$ is an average of n functions $f_i : \mathbb{R}^d \to \mathbb{R}, i \in [n]$. Each function f_i is a local loss associated with a local dataset $\mathcal{D}_i$, which can only be accessed by client i.

We analyze the setting where there is a central server coordinating the training process. The server receives and aggregates the messages from the clients via a compressed communication channel, where the messages are compressed by some δ-contractive compression (see Definition 2.1), performs an update on the model parameters, and broadcasts the updated model to clients.

We measure the performance of the algorithm in terms of *the number of communication rounds from the clients to the server*. We follow the assumption made in most existing works on the theory of communication-compressed optimization that the server-to-client communication is much more efficient (as compared to client to server communication) and the broadcast cost is negligible (Mishchenko et al., 2019; Kairouz et al., 2019; Fatkhullin et al., 2023; Gao et al., 2024). We are interested in the practically relevant setting where the batch size is some constant independent of the other problem parameters. This is the standard in most existing works on stochastic distributed optimization with compression (Karimireddy et al., 2019; Stich, 2020; Fatkhullin et al., 2023; Gao et al., 2024). We particularly avoid taking target-error-dependent batch sizes, since mega-batch methods are observed empirically to be adversarial to the generalization performance (Wilson and Martinez, 2003; LeCun et al., 2012; Keskar et al., 2017), and theoretically (Sekhari et al., 2021) for certain tasks. In addition, mega-batches are often unavailable or intractable, e.g., in medical tasks (Rieke et al., 2020); federated Reinforcement Learning (Khodadadian et al., 2022; Jin et al., 2022); and multi-agent Reinforcement Learning (Doan et al., 2019). We also avoid taking δ-dependent batch sizes, since in practice we often do not know the precise δ value of the compression operator, and therefore cannot tune the batch size accordingly.

We first introduce the following notation, which we use throughout the paper.

Notation 3.1. We denote the Bregman divergence of f between $\mathbf{x}$ and $\mathbf{y}$ as $\beta_f(\mathbf{x},\mathbf{y})$, i.e.,

$$\beta_f(\mathbf{x},\mathbf{y}) \coloneqq f(\mathbf{y}) - f(\mathbf{x}) - \langle \nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle . \tag{3}$$

Next, we list the assumptions that we make in this paper. Notably, we do not assume that the local functions f_i's have gradient dissimilarity, which is a common limiting assumption in many of the existing works, in both the unaccelerated and accelerated cases, e.g. (Lian et al., 2017; Huang et al., 2022; Li and Li, 2023; He et al., 2023). A crucial algorithmic component of our method that enables the acceleration also achieves this bias correction phenomenon naturally.

Assumption 3.2. We assume that the objective function f is convex, closed and proper.

Assumption 3.3. We assume that the objective function f has L-Lipschitz gradients, i.e. for all $\mathbf{x},\mathbf{y}\in\mathbb{R}^d$, it holds

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \le L\|\mathbf{x}-\mathbf{y}\|. \tag{4}$$

Many existing works further assumes that each local function f_i is $L_{\max}$-smooth and convex (Li and Richtárik, 2021; He et al., 2023). We instead make the following weaker assumption:

Assumption 3.4. We assume that there exists some $\ell > 0$ such that for all $\mathbf{x},\mathbf{y}\in\mathbb{R}^d$, it holds

$$\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|^2 \le 2\ell\beta_f(\mathbf{x},\mathbf{y}). \tag{5}$$

It is easy to show that Assumption 3.4 follows from $L_{\max}$-smoothness of each local function f_i, or from the Hessian similarity assumption (Khaled and Jin, 2022; Jiang et al., 2024; Rodomanov et al., 2024; Bylinkin and Beznosikov, 2024). See Appendix B for more details.

Next we make the following standard assumption on the stochastic gradient oracles.

Assumption 3.5. We assume that each client i has access to an unbiased stochastic gradient oracle $\mathbf{g}_i(\mathbf{x},\xi^i) : \mathbb{R}^d \to \mathbb{R}^d$ for the local function f_i, such that for all $\mathbf{x}\in\mathbb{R}^d$, it holds

$$\begin{aligned} &\mathbb{E}_{\xi^i}\left[\mathbf{g}_i(\mathbf{x},\xi^i)\right] = \nabla f_i(\mathbf{x}), \\ &\mathbb{E}_{\xi^i}\left[\|\mathbf{g}_i(\mathbf{x},\xi^i) - \nabla f_i(\mathbf{x})\|^2\right] \le \sigma^2. \end{aligned} \tag{6}$$

Mini-batches are also allowed, which simply divides the variance by the batch size. Our algorithm does not impose restrictions on the minimal batch size needed during training.

4 ACCELERATED METHOD WITH INEXACT UPDATE

In this section, we present a general framework for studying accelerated methods with inexact updates.

Algorithm 1 Accelerated Method with Inexact Update

1: **Input:** $\mathbf{x}_0, \mathbf{v}_0, A_0, (a_t)_{t=1}^{\infty}$
2: **for** $t = 0, 1, \dots$ **do**
3: $\quad A_{t+1} = A_t + a_{t+1}, \;\; \mathbf{y}_t = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_t$
4: $\quad$ Compute $\hat{\mathbf{g}}_t \approx \mathbf{g}_t$
5: $\quad \mathbf{v}_{t+1} = \mathbf{v}_t - a_{t+1}\hat{\mathbf{g}}_t, \;\; \mathbf{x}_{t+1} = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_{t+1}$

Here, we do not make any assumptions about the inexactness and obtain a general framework for the analysis. Our framework is flexible and can be applied to different setups and algorithms, unlike some existing works such as (Devolder et al., 2014), which assumes specific structures on the inexactness. We consider Algorithm 1. The algorithm is a simple extension to the standard accelerated method, where in Line 5 we use some inexact $\hat{\mathbf{g}}_t$ that is "approximately" the stochastic gradient $\mathbf{g}_t$ of f at $\mathbf{y}_t$. We assume the following for the stochastic gradient $\mathbf{g}_t$:

Assumption 4.1. For all $t \geq 0, \mathbf{g}_t \coloneqq \mathbf{g}(\mathbf{y}_t, \xi_t)$ is a stochastic gradient of f at $\mathbf{y}_t$, where ξ_t are independent copies of the oracle's randomness ξ. We assume that for $\mathbf{x} \in \mathbb{R}^d, \mathbf{g}(\mathbf{x}, \xi)$ is an unbiased stochastic gradient with bound variance:

$$\mathbb{E}\left[\mathbf{g}(\mathbf{x}, \xi)\right] = \nabla f(\mathbf{x}), \quad \mathbb{E}_{\xi}\left[\|\mathbf{g}(\mathbf{x}, \xi) - \nabla f(\mathbf{x})\|^2\right] \leq \sigma_{\mathbf{g}}^2. \tag{7}$$

Furthermore, we assume that $\hat{\mathbf{g}}_t$ is a deterministic function of $\xi_{[t]} \coloneqq \{\xi_0, \dots, \xi_t\}$.

The value of $\sigma_{\mathbf{g}}^2$ can be adapted to different setups, and in particular, for distributed optimization problems, we think of $\mathbf{g}_t$ as the average of the local stochastic gradients (see Assumption 3.5) and therefore we have $\sigma_{\mathbf{g}}^2 = \frac{\sigma^2}{n}$. For now, we do not assume any specific structure on the inexact gradient $\hat{\mathbf{g}}_t$. Such an inexact gradient could be the result of absolute compression (see Appendix F); or it could be built algorithmically via compressed communication, which is the focus of this paper.

The main result in this section is a descent theorem for the final iterate of Algorithm 1, up to some cumulative error that depends on the inexactness of the gradient $\hat{\mathbf{g}}_t$. In particular, we define the following cumulative error at iteration t:

$$\mathbf{e}_t \coloneqq \sum_{j=0}^{t-1} a_{j+1}(\hat{\mathbf{g}}_j - \mathbf{g}_j), \;\; E_t \coloneqq \mathbb{E}\left[\|\mathbf{e}_t\|^2\right]. \tag{8}$$

We also introduce some additional notations:

$$F_t \coloneqq \mathbb{E}\, f(\mathbf{x}_t) - f(\mathbf{x}^\star), \;\; R_t^2 \coloneqq \mathbb{E}\left[\|\mathbf{v}_t - \mathbf{x}^\star\|^2\right], \tag{9}$$

Now, the main result of this section is the following:

Theorem 4.2. *Given Assumptions 4.1 to 3.3, for all $T \geq 1$, for $(\mathbf{x}_t, \mathbf{y}_t, \mathbf{v}_t)_{t=0}^{\infty}$ generated by Algorithm 1, it holds that:*

$$\begin{aligned} &A_T F_T + \sum_{t=0}^{T-1}\left(\frac{1}{6} - \frac{2La_{t+1}^2}{A_{t+1}}\right)a_{t+1}^2\mathbb{E}\left[\|\mathbf{g}_t\|^2\right] \\ \leq &A_0 F_0 + \frac{R_0^2}{2} + 2\sigma_g^2\sum_{t=0}^{T-1} a_{t+1}^2 + \sum_{t=0}^{T-1} w_{t+1}E_{t+1} \\ &- \sum_{t=0}^{T-1}\mathbb{E}\Big[\frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + A_t\beta_f(\mathbf{y}_t, \mathbf{x}_t) \\ &\qquad + A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\Big] \end{aligned} \tag{10}$$

where we write $w_t \coloneqq \min\{2, a_t L\} + \frac{4La_t^2}{A_t} + \frac{4La_{t+1}^2}{A_{t+1}}$.

Novel analytical framework: The missing proof is in Appendix C. We point out that while Theorem 4.2's proof employs virtual iteration techniques used in the analysis of EF and its variants (e.g. (Karimireddy et al., 2019; Stich, 2020; Gao et al., 2024)), the proof is more involved as it also carefully integrates the acceleration scheme. More importantly, Theorem 4.2 takes a different approach from the one-step descent analysis template, which most existing works applied (Karimireddy et al., 2019; Stich, 2020; Richtárik et al., 2021; Fatkhullin et al., 2023; Gao et al., 2024), in that it is an overall descent lemma for the final iterate. Instead of constructing a single Lyapunov function that attempts to capture all individual one-step descent (e.g., R_t^2 and E_t), which often involves quite some speculations for the appropriate choices of parameters, we only need to upper bound the sum of the cumulative errors now.

To address variations in the step size during training, we apply a dynamic weighting to each error term E_{t+1}, by using the weight $\min\{2M, a_{t+1}L\}$, instead of simply using either one throughout. This comes from the fact that the stepsize a_{t+1}, albeit typically growing linearly, might attain a lower value than a fixed constant throughout the algorithm due to the stepsize tuning for the stochastic terms.

Applications of the general framework: Theorem 4.2 covers a wide range of algorithms that take the form of Algorithm 1 and can be applied to study the convergence of these algorithms as long as the sum of the cumulative errors can be controlled. To demonstrate its flexibility, we apply Theorem 4.2 to analyze various algorithms besides our main algorithm in this paper:

- in Appendix F, we briefly study a different class of compressors, absolute compressor $\mathcal{C}_\Delta$ such that $\mathbb{E}\left[\|\mathcal{C}_\Delta(\mathbf{x}) - \mathbf{x}\|^2\right] \leq \Delta^2$, and analyze the convergence of Algorithm 4 with such compressors. As

Algorithm 2 ADEF Accelerated Distributed Error Feedback

Input: $\mathbf{x}_0, \mathbf{v}_0, A_0, (a_t)_{t=1}^{\infty}$ and $\mathcal{C}$.
$\mathbf{e}_0^i = \mathbf{0}$, $\tilde{\mathbf{g}}_{-1}^i = \mathbf{g}_i(\mathbf{y}_0, \xi_0^i)$, $\tilde{\mathbf{g}}_{-1} = \frac{1}{n}\sum_{i=1}^n \tilde{\mathbf{g}}_{-1}^i$
for $t = 0, 1, \ldots$ **do**
  server:
  $A_{t+1} = A_t + a_{t+1}$, send to each client $\mathbf{y}_t = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_t$
  each client i**:**
  $\mathbf{g}_t^i = \mathbf{g}_i(\mathbf{y}_t, \xi_t^i)$
  $\tilde{\delta}_t^i = \mathbf{g}_t^i - \tilde{\mathbf{g}}_{t-1}^i$, $\tilde{\Delta}_t^i = \mathcal{C}(\tilde{\delta}_t^i)$, $\tilde{\mathbf{g}}_t^i = \tilde{\mathbf{g}}_{t-1}^i + \tilde{\Delta}_t^i$
  $\delta_t^i = \mathbf{g}_t^i - \tilde{\mathbf{g}}_t^i - \frac{1}{a_{t+1}}\mathbf{e}_t^i$, $\Delta_t^i = \mathcal{C}(\delta_t^i)$
  $\mathbf{e}_{t+1}^i = a_{t+1}(\Delta_t^i - \delta_t^i)$
  send to server $\tilde{\Delta}_t^i$, Δ_t^i
  server:
  $\tilde{\mathbf{g}}_t = \tilde{\mathbf{g}}_{t-1} + \frac{1}{n}\sum_{i=1}^n \tilde{\Delta}_t^i$
  $\hat{\mathbf{g}}_t = \tilde{\mathbf{g}}_t + \frac{1}{n}\sum_{i=1}^n \Delta_t^i$
  $\mathbf{v}_{t+1} = \mathbf{v}_t - a_{t+1}\hat{\mathbf{g}}_t$, $\mathbf{x}_{t+1} = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_{t+1}$

far as we are aware, this is the first result on the accelerated method with absolute compression;

- in Appendix G, we analyze Algorithm 6, which is essentially the NEOLITHIC algorithm in He et al. (2023), where $\hat{\mathbf{g}}_t$ is built via repeated communication rounds;
- in Appendix E, as a further example, we analyze the algorithm where $\hat{\mathbf{g}}_t$ is built via the vanilla error feedback, see Algorithm 3, and demonstrate that it does not achieve the accelerated rate. This highlights the critical role of the gradient difference compression-enhanced EF mechanism we propose in the next section.

Most importantly, in the next section, we present our main algorithm for distributed optimization with communication compression, ADEF, and analyze its convergence using Theorem 4.2.

5 ACCELERATION WITH ERROR FEEDBACK

In this section, we present our main algorithm, ADEF, summarized in Algorithm 2. All missing proofs can be found in Appendix D. At its core, the algorithm is still the accelerated method with inexact updates in the form of Algorithm 1. However, we introduce an enhanced error feedback mechanism to build the inexact gradient $\hat{\mathbf{g}}_t$, and we highlight the procedures in which we build the inexact gradient with red and green color boxes in Algorithm 2. The algorithm employs a delicate combination of error feedback (green boxes, see the discussions below) and gradient difference compression (red boxes, see the discussions further below surrounding Equation (14)). The green lines highlight the backbone error feedback procedure:

$$\begin{aligned} &\delta_t^i = \mathbf{g}_t^i - \tilde{\mathbf{g}}_t^i - \frac{1}{a_{t+1}}\mathbf{e}_t^i, \quad \Delta_t^i = \mathcal{C}(\delta_t^i) \\ &\mathbf{e}_{t+1}^i = a_{t+1}(\Delta_t^i - \delta_t^i), \quad \hat{\mathbf{g}}_t = \tilde{\mathbf{g}}_t + \frac{1}{n}\sum_{i=1}^n \Delta_t^i. \end{aligned} \tag{11}$$

For any sequences $\widetilde{\mathbf{g}}_t^i$, the error feedback mechanism summarized above records the errors that occurred due to the compression. The following lemma, which is the key property of EF and most of its variants (Karimireddy et al., 2019; Stich, 2020; Gao et al., 2024), states that the cumulative error defined in Equation (8) is precisely the average of the local errors tracked by Equation (11), irrespective of the choice of $\widetilde{\mathbf{g}}_t^i$'s. Consequently, we can bound the sum of the accumulative errors $\sum_{t=0}^{T-1} w_{t+1}E_{t+1}$ with the following quantity that quantifies the distance between the local stochastic gradients and the control variates:

$$H_t \coloneqq \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\mathbf{g}_t^i - \widetilde{\mathbf{g}}_t^i\|^2\right]. \tag{12}$$

Lemma 5.1. *For an algorithm that follows Equation* (11)*, the local error terms* $(\mathbf{e}_T^i)_{t=0}^{\infty}$ *satisfies the following:* $\frac{1}{n}\sum_{i=1}^n \mathbf{e}_t^i = \sum_{j=0}^{t-1} a_{j+1}(\hat{\mathbf{g}}_j - \mathbf{g}_j) =: \mathbf{e}_t$*, where* $\mathbf{g}_t$*'s are the average local stochastic gradients* $\mathbf{g}_t = \frac{1}{n}\sum_{i=1}^n \mathbf{g}_t^i$*. Further, if for all* $t \le T-1, (1 - \frac{\delta}{2})w_{t+1} \le (1 - \frac{\delta}{4})w_t$*, then:*

$$\sum_{t=0}^{T-1} w_{t+1}E_{t+1} \le \frac{8}{\gamma}\sum_{t=0}^{T-1} w'_{t+1}H_t, \tag{13}$$

where $w'_t \coloneqq w_t a_t^2$

For the vanilla EF method, the control variate $\widetilde{\mathbf{g}}_t^i$'s are simply $\mathbf{0}$. This is however undesirable, as upper bounding $w'_{t+1}H_t$ will involve upper bounding $\frac{w'_{t+1}}{n}\sum_{i=1}^n \mathbb{E}\left[\|\nabla f_i(\mathbf{y}_t)\|^2\right]$, which requires bounded heterogeneity assumptions and induces errors that will force the stepsize to be too small and eliminates the acceleration. See Appendix E for more details. To address these issues, we build the control variates $\widetilde{\mathbf{g}}_t^i$'s to approximate $\mathbf{g}_t^i$'s and reduce the overall errors from the compression. The construction of $\widetilde{\mathbf{g}}_t^i$'s are highlighted in red in Algorithm 2 and summarized below:

$$\begin{aligned} &\mathbf{g}_t^i = \mathbf{g}_i(\mathbf{y}_t, \xi_t^i), \quad \tilde{\delta}_t^i = \mathbf{g}_t^i - \tilde{\mathbf{g}}_{t-1}^i, \\ &\tilde{\Delta}_t^i = \mathcal{C}(\tilde{\delta}_t^i), \quad \tilde{\mathbf{g}}_t^i = \tilde{\mathbf{g}}_{t-1}^i + \tilde{\Delta}_t^i. \end{aligned} \tag{14}$$

The procedure described in Equation (14) is sometimes known as a gradient difference compression mechanism,

which is mostly applied to address the data heterogeneity issue (Mishchenko et al., 2019; Fatkhullin et al., 2021; Gao et al., 2024). Here, it also enables the acceleration. We have the following lemma regarding the weighted sum of H_t:

Lemma 5.2. *Given Assumptions 3.4 and 3.5, for an algorithm that follows Equation* (14)*, if for all* $t \leq T-1$*, we have* $w'_{t+1}(1-\frac{\delta}{2}) \leq w'_t(1-\frac{\delta}{4})$*, then it holds that:*

$$\begin{aligned}&\sum_{t=0}^{T-1} w'_{t+1} H_t \\ \leq &\frac{16}{\delta^2}\sum_{t=1}^{T-1}\frac{w'_{t+1}}{n}\sum_{i=1}^{n}\mathbb{E}\left[\|\nabla f_i(\mathbf{y}_t)-\nabla f_i(\mathbf{y}_{t-1})\|^2\right] \quad (15)\\ &+\frac{16\sigma^2}{\delta^2}\sum_{t=1}^{T-1} w'_{t+1}.\end{aligned}$$

Finally, we can combine Theorem 4.2 with Lemma 5.1 and Lemma 5.2 to give the main convergence result for Algorithm 2. As we pointed out before, for the distributed optimization setting in this section, $\mathbf{g}_t$ is simply the average of the local gradients with $\sigma_{\mathbf{g}}^2 = \frac{\sigma^2}{n}$.

Theorem 5.3. *Given Assumptions 3.2 to 3.5,* $a_t \coloneqq {}^{(t+\frac{32}{\delta})}/_{M}$ *and* $A_0 \coloneqq {}^{32^2}/_{2\delta^2 M}$*, it suffices to have:*

$$T = \mathcal{O}\left(\frac{R_0^2\sigma^2}{n\varepsilon^2} + \frac{\sqrt{L}R_0^2\sigma}{\delta^2\varepsilon^{3/2}} + \frac{\sqrt{\ell R_0^2}}{\delta^2\sqrt{\varepsilon}}\right), \qquad (16)$$

number of iterations (communication rounds) of Algorithm 2 to get $F_T \leq \varepsilon$*, where we can set* $M = \max\left\{\frac{2^{13}\ell}{\delta^4}, \left(\frac{4T(T+\frac{32}{\delta})^2\sigma^2}{R_0^2 n}\right)^{\frac{1}{2}}, 8\left(\frac{LT(T+\frac{32}{\delta})^3\sigma^2}{\delta^4 R_0^2}\right)^{\frac{1}{3}}\right\}$.

Remark 5.4. We note that for the proof of Theorem 5.3, we mainly focus on the asymptotic behavior and do not optimize the choices of constants. Most likely, one can pick much smaller constants for M in practice, see also Section 6.

Remark 5.5. Theorem 5.3 gives the first accelerated convergence rate for distributed stochastic optimization with contractive compression. We first point out that the stochastic terms $\mathcal{O}(\frac{R_0^2\sigma^2}{n\varepsilon^2} + \frac{\sqrt{L}R_0^2\sigma}{\delta^2\varepsilon^{3/2}})$ match that of EControl (Gao et al., 2024), and is the best known for the distributed setting under arbitrary data heterogeneity. In particular, the leading term $\mathcal{O}(\frac{R_0^2\sigma^2}{n\varepsilon^2})$ is optimal, δ-free, and enjoys the linear speedup in the number of clients n.

Importantly, in the deterministic regime, i.e. when $\sigma^2 = 0$, we see that ADEF achieves the accelerated $\mathcal{O}(\frac{1}{\sqrt{\varepsilon}})$ convergence rate. There is however a $\frac{1}{\delta^2}$ factor, instead of $\frac{1}{\delta}$. Nonetheless, this is faster than the unaccelerated $\mathcal{O}(\frac{LR_0^2}{\delta\varepsilon})$ when $\delta \geq \sqrt{\frac{\varepsilon}{LR_0^2}}$ (here we assume $L = \ell = L_{\max}$ for simplicity of illustration).

Remark 5.6. We note that the $\frac{1}{\delta^2}$ dependence in the deterministic term is the consequence of the constraints of the stepsizes. In particular, M needs to be at least $\mathcal{O}(\frac{\ell}{\delta^4})$ so that the errors coming from the error feedback mechanism, E_t, and the gradient difference compression mechanism, H_t, can be properly controlled at the same time. In practice, we note that the stepsizes' δ dependence might be much better than $\frac{1}{\delta^4}$, see Section 6.

We point out that this higher dependence on δ in the deterministic term seems to be a common issue in the analysis of bias-corrected EF when using two compressions. It was also observed in the analysis of an unaccelerated method D-EC-SGD with Bias Correction and Double Contractive Compression in (Gao et al., 2024). It is unknown if this is merely a deficiency in the analysis or some inherent limitation of the method. The unaccelerated EControl method resolved this issue using a delicate error-control mechanism, but it seems to be difficult to adapt it for drastically changing stepsizes, which is precisely the case with acceleration. We leave it for future work to investigate if more refined analysis that improves the δ dependence for ADEF is possible or if the EControl mechanism can be adapted to the accelerated setting.

6 EXPERIMENTS

In this section, we present numerical evaluations of our algorithm and verify its theoretical properties discussed in the previous sections. We consider parameters of the following form: $a_t = \gamma(t+\frac{1}{\delta})$ and $A_0 = \frac{\gamma}{\delta^2}$, where γ acts as the inverse of M in Theorem 5.3. For all experiments, we perform a grid search for γ, unless otherwise stated. We always report the performance in terms of the number of communications. All additional details and codes can be found in Appendix H. All of our experiment code can be accessed at this link[2].

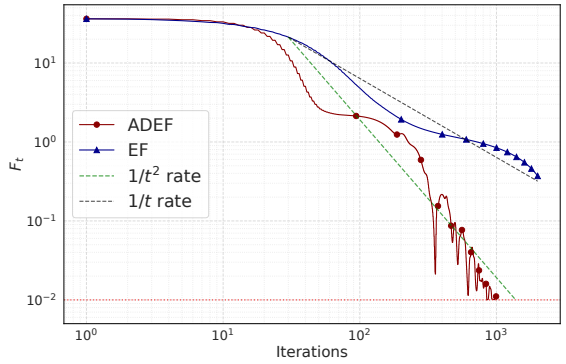
6.1 SYNTHETIC LOGISTIC REGRESSION

We first evaluate the performance of our algorithm on a synthetic logistic regression. We generate non-separable data with sklearn's built-in function (Pedregosa et al., 2011) and split the dataset into n clients. We consider the following logistic loss for each data point $(\mathbf{a}_i, b_i)$:

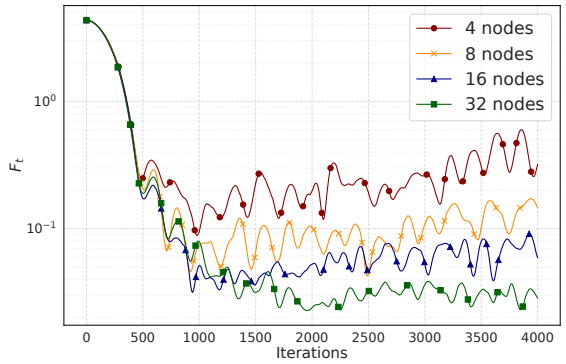
$$\ell((\mathbf{a}_i, b_i), \mathbf{x}) \coloneqq -b_i\mathbf{a}_i^\top\mathbf{x} + \log(1+\exp(\mathbf{a}_i^\top\mathbf{x})). \quad (17)$$

For the stochastic oracle, we generate the noise from a Normal distribution with variance σ^2. We use the Top-K compressor, with $\delta = 0.1$. Now we verify some of the key properties of our algorithm. **Accelerated rate in the deterministic case** We first demonstrate that Algorithm 2 indeed achieves the accelerated $\mathcal{O}\left(\frac{1}{T^2}\right)$ rate

[2] https://github.com/mlolab/adef_aistats



(a) **Achieving acceleration.** Performance of ADEF and EF with $\sigma^2 = 0$. We set a target error of 0.01. We see that ADEF achieves the accelerated $\mathcal{O}(1/T^2)$ rate.



(b) **Achieving linear speedup.** The performance of ADEF with increasing number of clients n for the synthetic logistic regression problem. We fix γ to be 0.0001. The error that the algorithm stabilizes around decreases as n increases.

Figure 1: **Synthetic logistic regression**

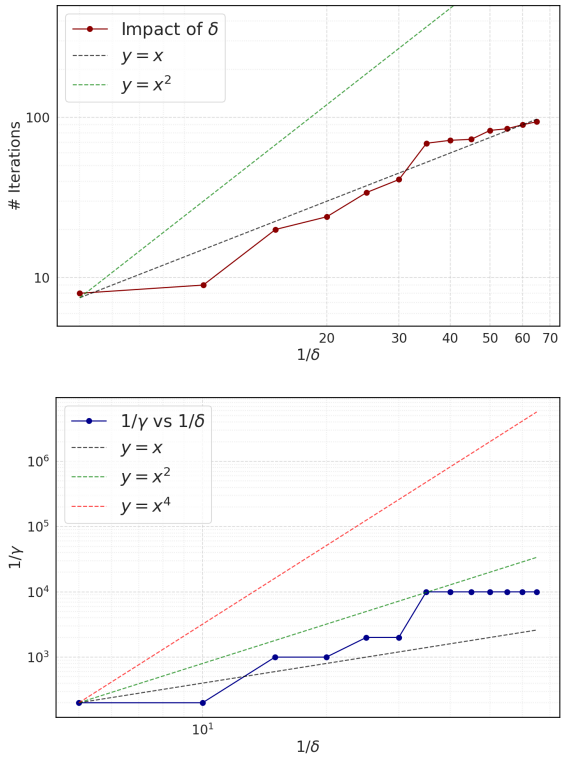


Figure 2: **Impact of δ.** The number of iterations needed to reach the target error increases *linearly* with $\frac{1}{\delta}$, and is better than the quadratic dependence in the upper bound. The stepsize parameter $\frac{1}{\gamma}$ scales *quadratically* with $\frac{1}{\delta}$, which is better than the $\frac{1}{\delta^4}$ theoretical value.

when $\sigma^2 = 0$. We contrast this against the convergence rate of the unaccelerated EF. The results are summarized in Figure 1a. We see that ADEF indeed achieves the accelerated rate $\mathcal{O}\left(\frac{1}{T^2}\right)$, while EF converges at the rate of $\mathcal{O}\left(\frac{1}{T}\right)$. **Linear speedup with the number of clients** Next we verify the linear speedup property of our algorithm with the number of clients n with the leading terms in the convergence rate. We fix $\sigma^2 = 25$ and a small enough γ to be 0.0001, and increase the number of clients n. The results are summarized in Figure 1b. We see that the algorithm gets more stable as the number of clients increases, and the error that the algorithm stabilizes around decreases as n increases, which verifies the linear speedup property of our algorithm.

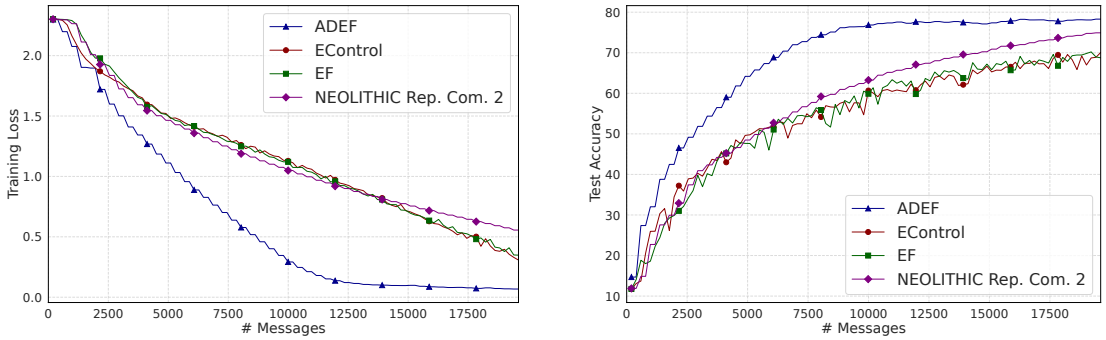
6.2 SYNTHETIC SOFTMAX LOSS AND IMPACT OF δ

Following our discussions regarding δ in Remark 5.5, we also conduct experiments to evaluate the impact of δ on the performance of ADEF in the deterministic setting. To this end, we consider the softmax objective:

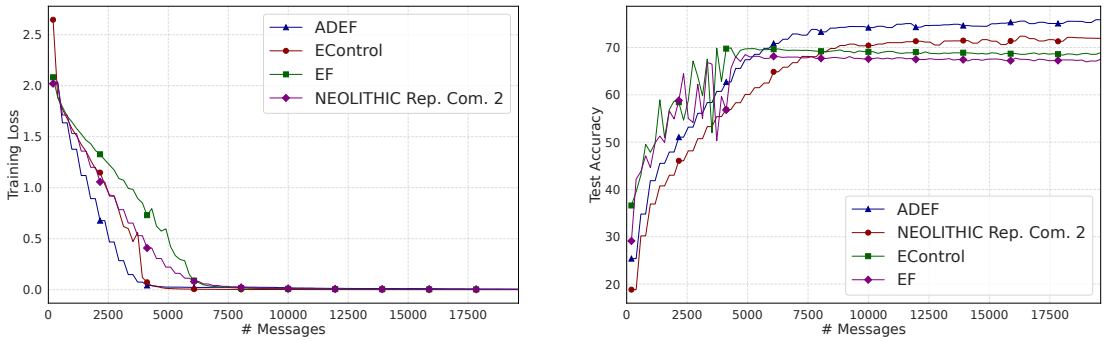
$$\min_{\mathbf{x}\in\mathbb{R}^d}\left\{f(\mathbf{x}) := \mu\log\left(\sum_{i=1}^{k}\exp\left[\frac{\langle a_i,\mathbf{x}\rangle - b_i}{\mu}\right]\right)\right\}, \tag{18}$$

where μ is a parameter that controls the smoothness of the objective, and we set it to $\mu = 0.1$. Details of the softmax loss are discussed in Appendix H.2.

We run ADEF with Top-K and $\frac{1}{\delta} \in \{\frac{1}{5}, \frac{1}{10}, \frac{1}{15}, \cdots, \frac{1}{65}\}$. We compare the number of iterations needed to reach the target error 0.05 with each δ and grid search for the best step sizes. In Figure 2 we summarize the results. In particular, we see that the number of iterations needed to reach the target error is increasing *linearly* with $\frac{1}{\delta}$, in contrast to the theoretical upper bound in $\mathcal{O}(\frac{1}{\delta^2})$ that we provide in Theorem 5.3. Moreover, the grid-searched stepsize $\frac{1}{\gamma}$ scales *quadratically* with $\frac{1}{\delta}$, which is better than the $\frac{1}{\delta^4}$ theoretical value. This suggests that there might be some room for improvement in the analysis techniques for methods that combine EF with gradient difference compression.



(a) VGG



(b) Resnet18

Figure 3: **CIFAR-10 classification** Comparison of the performance of ADEF, EConrol,EF and NEOLITHIC on the CIFAR-10 classification problem. We use Top-K compression with $\delta = 0.1$. We see that ADEF outperforms all other methods in both the training loss and the test accuracy.

6.3 CIFAR-10 CLASSIFICATION

We now consider the training of Deep Learning Models. We evaluate the performance of our algorithm ADEF against various baselines for training Resnet18 (He et al., 2016) and VGG (Simonyan and Zisserman, 2015) on the CIFAR-10 classification problem (Krizhevsky et al., 2014). We split the CIFAR-10 dataset into $n = 10$ clients, and distribute half of the dataset randomly to each client, and assign the rest of the dataset according to their labels, i.e. data with label i is distributed to client i. We compare our method against EF, EControl, and NEOLITHIC (for which the number of repeated communication rounds is tuned). We plot the figures in terms of the number of compressed messages communicated for all methods for a fair comparison. The results are summarized in Figures 3a and 3b. We see that ADEF achieves the best performance in both the training loss and the test accuracy. For both Resnet18 and VGG, ADEF achieves significantly faster convergence in the training loss. This demonstrates the practical advantages of our method in a real-world setting.

7 CONCLUSION

We introduced ADEF, a single-loop algorithm that combines Nesterov acceleration with error feedback and gradient-difference compression, and established accelerated convergence guarantees for distributed optimization with contractive compression in the general convex regime. Building on a general framework for accelerated methods with inexact updates, our results clarify how acceleration and compression can be reconciled without repeated communication or large batch sizes, addressing a long-standing gap in the theory of communication-efficient optimization. Future directions include refining the handling of compression errors, extending error-feedback techniques to constrained or composite problems, and exploring applications to non-convex and large-scale models.

References

Alyazeed Albasyoni, Mher Safaryan, Laurent Condat, and Peter Richtárik. Optimal gradient compression for distributed and federated learning. *arXiv preprint arXiv:2010.03246*, 2020.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient

SGD via gradient quantization and encoding. In *Proceedings of Advances in Neural Information Processing Systems*, pages 1709–1720, 2017.

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

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.

Aleksandr Beznosikov, Samuel Horváth, Peter Richtárik, and Mher Safaryan. On biased compression for distributed learning. *Journal on Machine Learning Research*, 2020.

Aleksandr Beznosikov, Samuel Horváth, Peter Richtárik, and Mher Safaryan. On biased compression for distributed learning. *Journal of Machine Learning Research*, 24(276):1–50, 2023.

Dmitry Bylinkin and Aleksandr Beznosikov. Accelerated methods with compressed communications for distributed optimization problems under data similarity. *arXiv preprint arXiv:2412.16414*, 2024.

Jean-Baptiste Cordonnier. Convex optimization using sparsified stochastic gradient descent with memory. Technical report, 2018.

Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 2012.

Olivier Devolder, François Glineur, and Yurii Nesterov. First-order methods of smooth convex optimization with inexact oracle. *Mathematical Programming*, 146: 37–75, 2014.

Thinh Doan, Siva Maguluri, and Justin Romberg. Finite-time analysis of distributed td (0) with linear function approximation on multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 1626–1635. PMLR, 2019.

Ilyas Fatkhullin, Igor Sokolov, Eduard Gorbunov, Zhize Li, and Peter Richtárik. Ef21 with bells & whistles: Practical algorithmic extensions of modern error feedback. *arXiv preprint arXiv: 2110.03294*, 2021.

Ilyas Fatkhullin, Alexander Tyurin, and Peter Richtárik. Momentum provably improves error feedback! *arXiv preprint arXiv: 2305.15155*, 2023.

Yuan Gao, Rustem Islamov, and Sebastian U Stich. EControl: Fast distributed optimization with compression and error control. In *The Twelfth International Conference on Learning Representations*, 2024.

Eduard Gorbunov, Dmitry Kovalev, Dmitry Makarenko, and Peter Richtárik. Linearly converging error compensated sgd. In *Proceedings of Advances in Neural Information Processing Systems*, 2020.

Kanan Gupta, Jonathan W. Siegel, and Stephan Wojtowytsch. Nesterov acceleration despite very noisy gradients. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *International conference on machine learning*, pages 1737–1746. PMLR, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Yutong He, Xinmeng Huang, Yiming Chen, Wotao Yin, and Kun Yuan. Lower bounds and accelerated algorithms in distributed stochastic optimization with communication compression. *arXiv preprint arXiv: 2305.07612*, 2023.

Samuel Horváth, Chen-Yu Ho, Ľudovít Horváth, Atal Narayan Sahu, Marco Canini, and Peter Richtárik. Natural compression for distributed deep learning. *arXiv preprint arXiv: 1905.10988*, 2019.

Xinmeng Huang, Yiming Chen, Wotao Yin, and Kun Yuan. Lower bounds and nearly optimal algorithms in distributed learning with communication compression. In *Proceedings of Advances in Neural Information Processing Systems*, 2022.

Rustem Islamov, Xun Qian, Slavomír Hanzely, Mher Safaryan, and Peter Richtárik. Distributed newton-type methods with communication compression and bernoulli aggregation. *Transactions on Machine Learning Research*, 2023.

Xiaowen Jiang, Anton Rodomanov, and Sebastian U Stich. Federated optimization with doubly regularized drift correction. *arXiv preprint arXiv:2404.08447*, 2024.

Hao Jin, Yang Peng, Wenhao Yang, Shusen Wang, and Zhihua Zhang. Federated reinforcement learning with environment heterogeneity. In *International Conference on Artificial Intelligence and Statistics*, pages 18–37. PMLR, 2022.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista A. Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaïd Harchaoui,

Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *arXiv preprint arXiv: 1912.04977*, 2019.

Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *Proceedings of the 36th International Conference on Machine Learning (ICML 2019)*, 2019.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *Proceedings of International Conference on Learning Representations*, 2017.

Ahmed Khaled and Chi Jin. Faster federated optimization under second-order similarity. *arXiv preprint arXiv:2209.02257*, 2022.

Sajad Khodadadian, Pranay Sharma, Gauri Joshi, and Siva Theja Maguluri. Federated reinforcement learning: Linear speedup under markovian sampling. In *International Conference on Machine Learning*, pages 10997–11057. PMLR, 2022.

Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

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

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

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 dataset. 2014.

Yann LeCun, Leon Bottou, Genevieve Orr, and Klaus-Robert Muller. Efficient backprop. *Neural Networks: Tricks of the Trade, Springer-Verlag*, 2012.

Xiaoyun Li and Ping Li. Analysis of error feedback in federated non-convex optimization with biased compression: Fast convergence and partial participation. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Zhize Li and Peter Richtárik. Canita: Faster rates for distributed convex optimization with communication compression, 2021.

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

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2017.

Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. Catalyst acceleration for first-order convex optimization: from theory to practice. *Journal of Machine Learning Research*, 18(212):1–54, 2018a.

Yujun Lin, Song Han, Huizi Mao, Yu Wang, and Bill Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *International Conference on Learning Representations*, 2018b.

Julien Mairal. Cyanure: An open-source toolbox for empirical risk minimization for python, c++, and soon more. *arXiv preprint arXiv:1912.08165*, 2019.

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

Mohammad Moshtaghifar, Anton Rodomanov, Daniil Vankov, and Sebastian U Stich. Dada: Dual averaging with distance adaptation. In *OPT 2024: Optimization for Machine Learning*.

Tomoya Murata and Taiji Suzuki. Accelerated sparsified sgd with error feedback. *arXiv preprint arXiv:1905.12224*, 2019.

Yurii Nesterov. A method for solving the convex programming problem with convergence rate o (1/k2). In *Dokl akad nauk Sssr*, volume 269, page 543, 1983.

Yurii Nesterov et al. *Lectures on convex optimization*, volume 137. Springer, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward

Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of Advances in Neural Information Processing Systems 32*, 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Xun Qian, Peter Richtárik, and Tong Zhang. Error compensated distributed sgd can be accelerated. *Advances in Neural Information Processing Systems*, 34: 30401–30413, 2021a.

Xun Qian, Peter Richtárik, and Tong Zhang. Error compensated distributed sgd can be accelerated. In *Proceedings of Advances in Neural Information Processing Systems*, 2021b.

Xun Qian, Hanze Dong, Tong Zhang, and Peter Richtárik. Catalyst acceleration of error compensated methods leads to better communication complexity. In *Proceedings of 25th International Conference on Artificial Intelligence and Statistics*, 2023.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. 2022.

Peter Richtárik, Igor Sokolov, and Ilyas Fatkhullin. Ef21: A new, simpler, theoretically better, and practically faster error feedback. In *Proceedings of Advances in Neural Information Processing Systems*, 2021.

Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger R Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N Galtier, Bennett A Landman, Klaus Maier-Hein, et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1): 119, 2020.

Anton Rodomanov, Xiaowen Jiang, and Sebastian U Stich. Universality of adagrad stepsizes for stochastic optimization: Inexact oracle, acceleration and variance reduction. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Mher Safaryan, Rustem Islamov, Xun Qian, and Peter Richtarik. FedNL: Making Newton-type methods applicable to federated learning. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

Atal Sahu, Aritra Dutta, Ahmed M Abdelmoniem, Trambak Banerjee, Marco Canini, and Panos Kalnis. Rethinking gradient sparsification as total error minimization. *Advances in Neural Information Processing Systems*, 34:8133–8146, 2021.

Amedeo Sapio, Marco Canini, Chen-Yu Ho, Jacob Nelson, Panos Kalnis, Changhoon Kim, Arvind Krishnamurthy, Masoud Moshref, Dan Ports, and Peter Richtárik. Scaling distributed machine learning with {In-Network} aggregation. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*, pages 785–808, 2021.

Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In *Proceedings of 15th annual conference of the international speech communication association*, 2014.

Ayush Sekhari, Karthik Sridharan, and Satyen Kale. Sgd: The role of implicit regularization, batch-size and multiple-epochs. *Advances In Neural Information Processing Systems*, 34:27422–27433, 2021.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick Le Gresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of International Conference on Learning Representations*, 2015.

Sebastian U. Stich. On communication compression for distributed optimization on heterogeneous data. *arXiv preprint arXiv: 2009.02388*, 2020.

Sebastian U Stich and Sai Praneeth Karimireddy. The error-feedback framework: Better rates for sgd with delayed gradients and compressed updates. *Journal of Machine Learning Research*, 2020.

Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. In *Proceedings of Advances in Neural Information Processing Systems*, 2018.

Nikko Strom. Scalable distributed DNN training using commodity GPU cloud computing. In *Proceedings of Interspeech 2015*, 2015.

Haobo Sun, Yingxia Shao, Jiawei Jiang, Bin Cui, Kai Lei, Yu Xu, and Jiang Wang. Sparse gradient compression for distributed sgd. In *International Conference on Database Systems for Advanced Applications*, pages 139–155. Springer, 2019.

Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. PowerSGD: Practical low-rank gradient compression for distributed optimization. In *Advances in Neural Information Processing Systems 32*, 2019.

Meng Wang, Weijie Fu, Xiangnan He, Shijie Hao, and Xindong Wu. A survey on large-scale machine learning. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

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. *Advances in neural information processing systems*, 30, 2017.

Randall Wilson and Tony Martinez. The general inefficiency of batch training for gradient descent learning. *Neural Networks*, 2003.

Shuai Zheng, Ziyue Huang, and James Kwok. Communication-efficient distributed blockwise momentum sgd with error-feedback. *Advances in Neural Information Processing Systems*, 32, 2019.

Heng Zhu, Avishek Ghosh, and Arya Mazumdar. Optimal compression of unit norm vectors in the high distortion regime. In *2023 IEEE International Symposium on Information Theory (ISIT)*, pages 719–724. IEEE, 2023.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. Yes. We provide a clear description of these in Section 3.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. Yes. The paper focuses precisely on the complexity of our proposed algorithm.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. Yes, we provide the link to anonymized code in Appendix H.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. Yes. All the assumptions are clearly stated with theoretical results.
   (b) Complete proofs of all theoretical results. Yes. All missing proofs can be found in the supplementary material.
   (c) Clear explanations of any assumptions. Yes. We provide clear explanations of all assumptions in Section 3.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). Yes. These can be found in Appendix H.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). Yes. These can be found in Appendix H.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). No. Our experiments are stable and easily reproducible.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). Yes. These can be found in Appendix H.4.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. Yes. We cite all the relevant works.
   (b) The license information of the assets, if applicable. Yes.
   (c) New assets either in the supplemental material or as a URL, if applicable. Yes. We provide the link to anonymized code in Appendix H.
   (d) Information about consent from data providers/curators. Not Applicable.
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. Not Applicable.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. Not Applicable.
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. Not Applicable.
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. Not Applicable.

Accelerated Distributed Optimization with Compression and Error Feedback: Supplementary Materials

Contents

A ADDITIONAL RELATED WORKS AND A BRIEF HISTROY OF EF

The EF mechanism, proposed by Seide et al. (2014), was first analyzed theoretically in (Stich et al., 2018; Alistarh et al., 2018; Karimireddy et al., 2019), but only for the single-client setting. Extensions to the distributed setting were first made under data similarity assumptions, either implicitly in the form of bounded gradient assumption (Cordonnier, 2018; Alistarh et al., 2018), or explicitly in the form of gradient similarity assumptions (Stich and Karimireddy, 2020), both are very limiting factors. Further extensions to fully decentralized settings were also considered in (Koloskova et al., 2019, 2020) under the bounded gradient assumption. The theory of distributed EF was further refined in (Beznosikov et al., 2020; Stich, 2020). A key point in the analysis of distributed EF is to obtain a convergence rate in the number of communication rounds, where the leading term (term that involves the variance of the stochastic oracle, see Assumption 3.5) is unaffected by the compression quality δ and enjoys the linear speedup in the number of clients.

Mishchenko et al. (2019) proposed the DIANA algorithm, which incorporates an additional *unbiased* compressor into EF for bias correction, alleviating the need for data similarity assumptions. It inspired a number of follow-up works (Gorbunov et al., 2020; Stich, 2020; Qian et al., 2021b), and eventually led to the EF21 algorithm, which is the first that fully supports contractive compression in the full gradient regime (Richtárik et al., 2021). However, the bias correction mechanism of EF21 does not work with stochastic gradients and leads to non-convergence up to the variance of the stochastic oracle. The challenge of bias correction with stochastic gradients was finally addressed in Fatkhullin et al. (2023) with momentum (which however, is not known to work in the general convex regime), and in Gao et al. (2024) using an error-controlled EF combined with bias correction which covers the strongly convex, general convex and non-convex cases.

B AUXILARY FACTS AND RESULTS

In this section we collect some auxilary facts and results that are useful for the analysis of our algorithms. The first one is a simple fact regarding the square of the norm of a sum of vectors.

Fact B.1. For any $\gamma_1, \ldots, \gamma_T$, we have:

$$\|\sum_{t=1}^{T} \gamma_t\|^2 \leq T \sum_{t=1}^{T} \|\gamma_t\|^2. \tag{19}$$

The next one is on upper bounding the sum of a sequence under a descent condition on the sequence:

Lemma B.2. *Given a sequence of non-negative values $\{\alpha_t\}_{t\in[T-1]}$, and some other sequences $\{\lambda_t\}_{t\in[T-1]}$. If there exists $\gamma \in (0,1)$ such that the following holds:*

$$\alpha_{t+1} \leq (1-\gamma)\alpha_t + \lambda_t, \quad \alpha_0 = 0, \tag{20}$$

then we have:

$$\sum_{t=0}^{T-1} \alpha_{t+1} \leq \frac{1}{\gamma} \sum_{t=0}^{T-1} \lambda_t \tag{21}$$

Proof. We sum Equation (20) on both sides from $t = 0$ to $T - 1$:

$$\sum_{t=0}^{T-1} \alpha_{t+1} \leq (1-\gamma) \sum_{t=0}^{T-1} \alpha_t + \sum_{t=0}^{T-1} \lambda_t$$
$$\overset{(i)}{=} (1-\gamma) \sum_{t=1}^{T-1} \alpha_t + \sum_{t=0}^{T-1} \lambda_t,$$

where in (i) we used the fact that $\alpha_0 = 0$. Now we can subtract $(1-\gamma)\sum_{t=1}^{T-1} \alpha_t$ from both sides and get:

$$\gamma \sum_{t=0}^{T-1} \alpha_{t+1} \leq \sum_{t=0}^{T-1} \lambda_t,$$

divide both sides by γ and we get the desired result. □

The next fact is on the convexity and smoothness of a function:

Fact B.3. Given Assumptions 3.2 and 3.3, for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have:

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|^2 \leq 2L\beta_f(\mathbf{x}, \mathbf{y}). \tag{22}$$

We also have the following fact.

Fact B.4. If Assumption 3.4 holds with some $\ell \geq 0$, then Assumption 3.3 holds with $L \leq \ell$.

See, e.g. Theorem 2.1.5 in Nesterov et al. (2018) for a proof for both Fact B.3 and Fact B.4.

We now state the following facts that show that Assumption 3.4 follows from either $L_{\max}$-smoothness of f_i or Hessian similarity of f_i.

Fact B.5. Given the assumption that each f_i is $L_{\max}$-smooth and convex, then Assumption 3.4 holds with $\ell \leq L_{\max}$.

Fact B.6. Given Assumptions 3.2 and 3.3, if the local functions f_i's satisfy the λ-Hessian similarity assumption in the following sense:

$$\frac{1}{n}\sum_{i=1}^{n} \|\nabla \hat{f}_i(\mathbf{x}) - \nabla \hat{f}_i(\mathbf{y})\|^2 \leq 2\lambda\beta_f(\mathbf{x}, \mathbf{y}), \tag{23}$$

where $\hat{f}_i \coloneqq f - f_i$, then $\ell \leq L + \lambda$.

C ANALYSIS OF ACCELERATED METHOD WITH INEXACT UPDATES

In this section we present the proof of Theorem 4.2.

Theorem 4.2. *Given Assumptions 4.1 to 3.3, for all $T \geq 1$, for $(\mathbf{x}_t, \mathbf{y}_t, \mathbf{v}_t)_{t=0}^{\infty}$ generated by Algorithm 1, it holds that:*

$$\begin{aligned} &A_T F_T + \sum_{t=0}^{T-1} \left(\frac{1}{6} - \frac{2La_{t+1}^2}{A_{t+1}}\right) a_{t+1}^2 \mathbb{E}\left[\|\mathbf{g}_t\|^2\right] \\ \leq &A_0 F_0 + \frac{R_0^2}{2} + 2\sigma_g^2 \sum_{t=0}^{T-1} a_{t+1}^2 + \sum_{t=0}^{T-1} w_{t+1} E_{t+1} \\ &- \sum_{t=0}^{T-1} \mathbb{E}\Big[\frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + A_t \beta_f(\mathbf{y}_t, \mathbf{x}_t) \\ &\qquad + A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\Big] \end{aligned} \tag{10}$$

where we write $w_t \coloneqq \min\{2, a_t L\} + \frac{4La_t^2}{A_t} + \frac{4La_{t+1}^2}{A_{t+1}}$.

Proof. Recall that

$$\mathbf{v}_{t+1} = \mathbf{v}_t - a_{t+1}\hat{\mathbf{g}}_t.$$

We consider $\widetilde{\mathbf{v}}_t, \widetilde{\mathbf{v}}_0 = \mathbf{v}_0$ defined the in the following way:

$$\widetilde{\mathbf{v}}_{t+1} \coloneqq \widetilde{\mathbf{v}}_t - a_{t+1}\mathbf{g}_t$$

We also consider $\widetilde{R}_t \coloneqq \mathbb{E}\left[\|\widetilde{\mathbf{v}}_t - \mathbf{x}\|^2\right]$.

We start by giving a one step descent. Given any $\mathbf{x} \in \mathbb{R}^d$ we have the following:

$$\begin{aligned}
&\mathbb{E}\left[A_t f(\mathbf{x}_t) + a_{t+1} f(\mathbf{x}) + \frac{1}{2}\|\widetilde{\mathbf{v}}_t - \mathbf{x}\|^2\right] \\
\overset{(i)}{=}&\mathbb{E}\left[A_t(f(\mathbf{y}_t) + \langle \nabla f(\mathbf{y}_t), \mathbf{x}_t - \mathbf{y}_t\rangle + \beta_f(\mathbf{y}_t, \mathbf{x}_t))\right] \\
&+ \mathbb{E}\left[a_{t+1}(f(\mathbf{y}_t) + \langle \nabla f(\mathbf{y}_t), \mathbf{x} - \mathbf{y}_t\rangle + \beta_f(\mathbf{y}_t, \mathbf{x})) + \frac{1}{2}\|\widetilde{\mathbf{v}}_t - \mathbf{x}\|^2\right] \\
\overset{(ii)}{=}&\mathbb{E}\left[A_t(f(\mathbf{y}_t) + \langle \nabla f(\mathbf{y}_t), \mathbf{x}_t - \mathbf{y}_t\rangle + \beta_f(\mathbf{y}_t, \mathbf{x}_t))\right] \\
&+ \mathbb{E}\left[a_{t+1}(f(\mathbf{y}_t) + \langle \mathbf{g}_t, \mathbf{x} - \mathbf{y}_t\rangle + \beta_f(\mathbf{y}_t, \mathbf{x})) + \frac{1}{2}\|\widetilde{\mathbf{v}}_t - \mathbf{x}\|^2\right] \\
\overset{(iii)}{=}&\mathbb{E}\left[A_t(f(\mathbf{y}_t) + \langle \nabla f(\mathbf{y}_t), \mathbf{x}_t - \mathbf{y}_t\rangle + \beta_f(\mathbf{y}_t, \mathbf{x}_t))\right] \\
&+ \mathbb{E}\left[a_{t+1}(f(\mathbf{y}_t) + \langle \mathbf{g}_t, \widetilde{\mathbf{v}}_{t+1} - \mathbf{y}_t\rangle + \beta_f(\mathbf{y}_t, \mathbf{x})) + \frac{1}{2}\|\widetilde{\mathbf{v}}_t - \widetilde{\mathbf{v}}_{t+1}\|^2 + \frac{1}{2}\|\widetilde{\mathbf{v}}_{t+1} - \mathbf{x}\|^2\right] \\
=&\mathbb{E}\left[A_t(f(\mathbf{y}_t) + \langle \nabla f(\mathbf{y}_t), \mathbf{x}_t - \mathbf{y}_t\rangle + \beta_f(\mathbf{y}_t, \mathbf{x}_t))\right] \\
&+ \mathbb{E}\left[a_{t+1}(f(\mathbf{y}_t) + \langle \nabla f(\mathbf{y}_t), \widetilde{\mathbf{v}}_{t+1} - \mathbf{y}_t\rangle + \beta_f(\mathbf{y}_t, \mathbf{x})) + \frac{1}{2}\|\widetilde{\mathbf{v}}_t - \widetilde{\mathbf{v}}_{t+1}\|^2\right] \\
&+ \mathbb{E}\left[\frac{1}{2}\|\widetilde{\mathbf{v}}_{t+1} - \mathbf{x}\|^2 + a_{t+1}\langle \mathbf{g}_t - \nabla f(\mathbf{y}_t), \widetilde{\mathbf{v}}_{t+1} - \widetilde{\mathbf{v}}_t\rangle\right] \\
\overset{(iv)}{=}&\mathbb{E}\left[A_{t+1}\left(f(\mathbf{y}_t) + \langle \nabla f(\mathbf{y}_t), \mathbf{x}_{t+1} - \mathbf{y}_t\rangle\right) + A_t\beta_f(\mathbf{y}_t, \mathbf{x}_t) + a_{t+1}\beta_f(\mathbf{y}_t, \mathbf{x}) + \frac{1}{2}\|\widetilde{\mathbf{v}}_t - \widetilde{\mathbf{v}}_{t+1}\|^2\right] \\
&+ \mathbb{E}\left[\frac{1}{2}\|\widetilde{\mathbf{v}}_{t+1} - \mathbf{x}\|^2 + a_{t+1}\langle \mathbf{g}_t - \nabla f(\mathbf{y}_t), \widetilde{\mathbf{v}}_{t+1} - \widetilde{\mathbf{v}}_t\rangle - a_{t+1}\langle \nabla f(\mathbf{y}_t), \mathbf{e}_{t+1}\rangle\right]
\end{aligned}$$

where for (i) we applied the convexity of f. For (ii) we applied the independence and unbiasedness of $\mathbf{g}_t$. For (iii) we used the fact that

$$\widetilde{\mathbf{v}}_{t+1} = \arg\min_{\mathbf{v}\in\mathbb{R}^d} \Psi_t(\mathbf{v}_t) \coloneqq a_{t+1}\langle \mathbf{g}_t, \mathbf{v}\rangle + \frac{1}{2}\|\mathbf{v} - \widetilde{\mathbf{v}}_t\|^2$$

where Ψ_t is 1-strongly convex. For (iv) we used the definition of $\mathbf{x}_{t+1}$. Next, we apply the smoothness and convexity of f:

$$\begin{aligned}
f(\mathbf{y}_t) + \langle \nabla f(\mathbf{y}_t), \mathbf{x}_{t+1} - \mathbf{y}_t\rangle &= f(\mathbf{x}_{t+1}) - \langle \nabla f(\mathbf{x}_{t+1}) - \nabla f(\mathbf{y}_t), \mathbf{x}_{t+1} - \mathbf{y}_t\rangle + \beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t) \\
&\geq f(\mathbf{x}_{t+1}) + \beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t) - L\|\mathbf{x}_{t+1} - \mathbf{y}_t\|^2 \\
&= f(\mathbf{x}_{t+1}) + \beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t) - \frac{La_{t+1}^2}{A_{t+1}^2}\|\mathbf{v}_{t+1} - \mathbf{v}_t\|^2 \\
&\geq f(\mathbf{x}_{t+1}) + \beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t) - \frac{2La_{t+1}^2}{A_{t+1}^2}\|\widetilde{\mathbf{v}}_{t+1} - \widetilde{\mathbf{v}}_t\|^2 - \frac{2La_{t+1}^2}{A_{t+1}^2}\|\mathbf{e}_{t+1} - \mathbf{e}_t\|^2
\end{aligned}$$

We plug it back in:

$$\begin{aligned}
&\mathbb{E}\left[A_t f(\mathbf{x}_t) + a_{t+1} f(\mathbf{x}^\star) + \frac{1}{2}\|\widetilde{\mathbf{v}}_t - \mathbf{x}^\star\|^2\right] \\
=&\mathbb{E}\left[A_{t+1} f(\mathbf{x}_{t+1}) + A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t) + A_t \beta_f(\mathbf{y}_t, \mathbf{x}_t) + a_{t+1}\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right] \\
&+ \mathbb{E}\left[(\frac{1}{2} - \frac{2La_{t+1}^2}{A_{t+1}})\|\widetilde{\mathbf{v}}_t - \widetilde{\mathbf{v}}_{t+1}\|^2 + \frac{1}{2}\|\widetilde{\mathbf{v}}_{t+1} - \mathbf{x}\|^2 + a_{t+1}\langle \mathbf{g}_t - \nabla f(\mathbf{y}_t), \widetilde{\mathbf{v}}_{t+1} - \widetilde{\mathbf{v}}_t\rangle\right] \\
&- \left(\mathbb{E}\left[a_{t+1}\langle \nabla f(\mathbf{y}_t), \mathbf{e}_{t+1}\rangle + \frac{2La_{t+1}^2}{A_{t+1}}\|\mathbf{e}_{t+1} - \mathbf{e}_t\|^2\right]\right) \\
\overset{(i)}{\geq}&\mathbb{E}\left[A_{t+1} f(\mathbf{x}_{t+1}) + A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t) + A_t \beta_f(\mathbf{y}_t, \mathbf{x}_t) + a_{t+1}\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right] \\
&+ \mathbb{E}\left[(\frac{1}{2} - \frac{2La_{t+1}^2}{A_{t+1}})\|\widetilde{\mathbf{v}}_t - \widetilde{\mathbf{v}}_{t+1}\|^2 + \frac{1}{2}\|\widetilde{\mathbf{v}}_{t+1} - \mathbf{x}\|^2\right] \\
&- \left(\mathbb{E}\left[\frac{3a_{t+1}^2\sigma_{\mathbf{g}}^2}{2} + \frac{1}{6}\|\widetilde{\mathbf{v}}_{t+1} - \widetilde{\mathbf{v}}_t\|^2 + a_{t+1}\langle \nabla f(\mathbf{y}_t), \mathbf{e}_{t+1}\rangle + \frac{2La_{t+1}^2}{A_{t+1}}\|\mathbf{e}_{t+1} - \mathbf{e}_t\|^2\right]\right)
\end{aligned}$$

where in (i) we applied Young's inequality and the bounded variance assumption. Now we pick $\mathbf{x} \coloneqq \mathbf{x}^\star$, and subtract $A_{t+1} f(\mathbf{x}^\star)$ from both sides, and rearrange the terms. We get:

$$\begin{aligned}
A_{t+1}F_{t+1} \leq A_t F_t + \frac{1}{2}(\widetilde{R}_t^2 - \widetilde{R}_{t+1}^2) + \frac{3a_{t+1}^2\sigma_{\mathbf{g}}^2}{2} - \left(\frac{1}{3} - \frac{2La_{t+1}^2}{A_{t+1}}\right)\mathbb{E}\left[\|\widetilde{\mathbf{v}}_t - \widetilde{\mathbf{v}}_{t+1}\|^2\right] - a_{t+1}\mathbb{E}\left[\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right] \\
- \mathbb{E}\left[A_t\beta_f(\mathbf{y}_t, \mathbf{x}_t)\right] - \mathbb{E}\left[A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\right] + \mathbb{E}\left[a_{t+1}\langle \nabla f(\mathbf{y}_t), \mathbf{e}_{t+1}\rangle + \frac{2La_{t+1}^2}{A_{t+1}}\|\mathbf{e}_{t+1} - \mathbf{e}_t\|^2\right]
\end{aligned}$$

Now we need to upper bound $a_{t+1}\mathbb{E}\left[\langle \nabla f(\mathbf{y}_t), \mathbf{e}_{t+1}\rangle\right]$. There are two options:

$$\begin{aligned}
a_{t+1}\mathbb{E}\left[\langle \nabla f(\mathbf{y}_t), \mathbf{e}_{t+1}\rangle\right] &= a_{t+1}\mathbb{E}\left[\langle \mathbf{g}_t, \mathbf{e}_{t+1}\rangle\right] + a_{t+1}\mathbb{E}\left[\langle \nabla f(\mathbf{y}_t) - \mathbf{g}_t, \mathbf{e}_{t+1}\rangle\right] \\
&\leq \frac{1}{6}\mathbb{E}\left[\|\widetilde{\mathbf{v}}_t - \widetilde{\mathbf{v}}_{t+1}\|^2\right] + \frac{a_{t+1}^2\sigma_{\mathbf{g}}^2}{2} + 2E_{t+1}
\end{aligned}$$

The other option is:

$$\begin{aligned}
a_{t+1}\mathbb{E}\left[\langle \nabla f(\mathbf{y}_t), \mathbf{e}_{t+1}\rangle\right] &\leq \frac{a_{t+1}}{4L}\mathbb{E}\left[\|\nabla f(\mathbf{y}_t)\|^2\right] + a_{t+1}LE_{t+1} \\
&\leq \frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + a_{t+1}LE_{t+1}
\end{aligned}$$

We combine these two upper bounds and get:

$$a_{t+1}\mathbb{E}\left[\langle \nabla f(\mathbf{y}_t), \mathbf{e}_{t+1}\rangle\right] \leq \frac{1}{6}\mathbb{E}\left[\|\widetilde{\mathbf{v}}_t - \widetilde{\mathbf{v}}_{t+1}\|^2\right] + \frac{a_{t+1}^2\sigma_{\mathbf{g}}^2}{2} + \frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + \min\{2, a_{t+1}L\}E_{t+1}$$

Plug it back in, we get:

$$\begin{aligned}
A_{t+1}F_{t+1} \leq A_t F_t + \frac{1}{2}(\widetilde{R}_t^2 - \widetilde{R}_{t+1}^2) + 2a_{t+1}^2\sigma_{\mathbf{g}}^2 - \left(\frac{1}{6} - \frac{2La_{t+1}^2}{A_{t+1}}\right)\mathbb{E}\left[\|\widetilde{\mathbf{v}}_t - \widetilde{\mathbf{v}}_{t+1}\|^2\right] - \frac{a_{t+1}}{2}\mathbb{E}\left[\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right] \\
- \mathbb{E}\left[A_t\beta_f(\mathbf{y}_t, \mathbf{x}_t)\right] - \mathbb{E}\left[A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\right] + \left(\min\{2, a_{t+1}L\} + \frac{4La_{t+1}^2}{A_{t+1}}\right)E_{t+1} + \frac{4La_{t+1}^2}{A_{t+1}}E_t
\end{aligned}$$

Now we sum over both sides from $t = 0$ to $T - 1$, and noticing that $E_t = 0$ and $R_0^2 = \widetilde{R}_0^2$, we get the desired result. □

D MISSING PROOFS FOR Section 5

In this section we present the missing proofs for the analysis of Algorithm 2.

Lemma 5.1. *For an algorithm that follows Equation* (11)*, the local error terms* $(\mathbf{e}_T^i)_{t=0}^{\infty}$ *satisfies the following:* $\frac{1}{n}\sum_{i=1}^n \mathbf{e}_t^i = \sum_{j=0}^{t-1} a_{j+1}(\hat{\mathbf{g}}_j - \mathbf{g}_j) =: \mathbf{e}_t$*, where* $\mathbf{g}_t$*'s are the average local stochastic gradients* $\mathbf{g}_t = \frac{1}{n}\sum_{i=1}^n \mathbf{g}_t^i$*. Further, if for all* $t \le T-1, (1-\frac{\delta}{2})w_{t+1} \le (1-\frac{\delta}{4})w_t$*, then:*

$$\sum_{t=0}^{T-1} w_{t+1}E_{t+1} \le \frac{8}{\gamma}\sum_{t=0}^{T-1} w'_{t+1}H_t, \tag{13}$$

where $w'_t \coloneqq w_t a_t^2$

Proof. We first prove Equation (8) by induction. For the base case $t=0$, we have $\frac{1}{n}\sum_{i=1}^n \mathbf{e}_0^i = \mathbf{e}_0 = \mathbf{0}$. Now assume that the lemma holds for some $t \ge 0$, we have:

$$\begin{aligned}
\sum_{j=0}^{t} a_{j+1}(\hat{\mathbf{g}}_j - \mathbf{g}_j) &= \sum_{j=0}^{t-1} a_{j+1}(\hat{\mathbf{g}}_j - \mathbf{g}_j) + a_{t+1}(\hat{\mathbf{g}}_t - \mathbf{g}_t) \\
&\overset{(i)}{=} \frac{1}{n}\sum_{i=1}^n \mathbf{e}_t^i + a_{t+1}(\hat{\mathbf{g}}_t - \mathbf{g}_t) \\
&= a_{t+1}\frac{1}{n}\sum_{i=1}^n (\Delta_t^i - \mathbf{g}_t^i + \frac{1}{a_{t+1}}\mathbf{e}_t^i + \widetilde{\mathbf{g}}_t^i) \\
&= \frac{1}{n}\sum_{i=1}^n \mathbf{e}_{t+1}^i,
\end{aligned}$$

where in (i) we used the induction hypothesis. This completes the proof for Equation (8). This identity allows us to loosen the upper bound on E_t, and we consider the following quantity:

$$\bar{E}_t \coloneqq \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\mathbf{e}_t^i\|^2\right] \tag{24}$$

We note that $\bar{E}_t$ is an upper bound on E_t, we have:

$$E_t = \mathbb{E}\left[\|\frac{1}{n}\sum_{i=1}^n \mathbf{e}_t^i\|^2\right] \le \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\mathbf{e}_t^i\|^2\right] = \bar{E}_t$$

Therefore, instead of upper bounding E_t directly, we will consider upper bounding $\bar{E}_t$. To utilize Lemma B.2 to upper bound the weighted some of $\bar{E}_t$, we first give an individual descent on $\bar{E}_t$:

$$\begin{aligned}
\bar{E}_{t+1} &= a_{t+1}^2 \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\Delta_t^i - \delta_t^i\|^2\right] \\
&\overset{(ii)}{\le} (1-\delta)a_{t+1}^2 \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\mathbf{g}_t^i - \widetilde{\mathbf{g}}_t^i - \frac{1}{a_{t+1}}\mathbf{e}_t^i\|^2\right] \\
&\overset{(iii)}{\le} (1-\frac{\delta}{2})\bar{E}_t + \frac{2a_{t+1}^2}{\delta}\frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\mathbf{g}_t^i - \widetilde{\mathbf{g}}_t^i\|^2\right]
\end{aligned}$$

where in (ii) we used the definition of $\mathcal{C}$. In (iii) we used the Young's inequality. Apply the definition of H_t and we get:

$$\bar{E}_{t+1} \le (1-\frac{\delta}{2})\bar{E}_t + \frac{2a_{t+1}^2}{\delta}H_t \tag{25}$$

For $t \in [T-1]$, we have:

$$\begin{aligned}
w_{t+1}\bar{E}_t &\le (1-\frac{\delta}{2})w_{t+1}\bar{E}_t + \frac{2w'_{t+1}}{\delta}H_t \\
&\overset{(iv)}{\le} (1-\frac{\delta}{4})w_t\bar{E}_t + \frac{2w'_{t+1}}{\delta}H_t
\end{aligned}$$

where in (iv) we used the assumption that $w'_{t+1}(1-\frac{\delta}{2}) \leq w'_t(1-\frac{\delta}{4})$. Now we note that $\bar{E}_0 = E_0 = 0$, write $\alpha_t = w_t\bar{E}_t, \lambda_t = w'_{t+1}H_t, \gamma = \frac{\delta}{4}$ and apply Lemma B.2 and get the desired result. □

Lemma 5.2. *Given Assumptions 3.4 and 3.5, for an algorithm that follows Equation* (14)*, if for all $t \leq T-1$, we have $w'_{t+1}(1-\frac{\delta}{2}) \leq w'_t(1-\frac{\delta}{4})$, then it holds that:*

$$\begin{aligned}
&\sum_{t=0}^{T-1} w'_{t+1}H_t \\
\leq &\frac{16}{\delta^2}\sum_{t=1}^{T-1}\frac{w'_{t+1}}{n}\sum_{i=1}^{n}\mathbb{E}\left[\|\nabla f_i(\mathbf{y}_t) - \nabla f_i(\mathbf{y}_{t-1})\|^2\right] \\
&+ \frac{16\sigma^2}{\delta^2}\sum_{t=1}^{T-1} w'_{t+1}.
\end{aligned} \tag{15}$$

Proof. We first give a one step descent on H_t:

$$\begin{aligned}
H_{t+1} &= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\|\mathbf{g}_{t+1}^i - \widetilde{\mathbf{g}}_{t+1}^i\|^2\right] \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\|\mathbf{g}_{t+1}^i - \widetilde{\mathbf{g}}_t^i - \widetilde{\Delta}_t^i\|^2\right] \\
&\overset{(i)}{\leq} \frac{(1+\alpha)}{n}\sum_{i=1}^{n}\mathbb{E}\left[\|\widetilde{\delta}_t^i - \widetilde{\Delta}_t^i\|^2\right] + \frac{(1+\alpha^{-1})}{n}\sum_{i=1}^{n}\mathbb{E}\left[\|\mathbf{g}_{t+1}^i - \mathbf{g}_t^i\|^2\right],
\end{aligned}$$

where in (i) we used Young's inequality. Pick $\alpha = \frac{\delta}{2(1-\delta)}$, we have:

$$\begin{aligned}
H_{t+1} &\overset{(ii)}{\leq} (1-\frac{\delta}{2})H_t + \frac{2}{\delta n}\sum_{i=1}^{n}\mathbb{E}\left[\|\mathbf{g}_{t+1}^i - \mathbf{g}_t^i\|^2\right] \\
&\overset{(iii)}{\leq} (1-\frac{\delta}{2})H_t + \frac{4}{\delta n}\sum_{i=1}^{n}\mathbb{E}\left[\|\nabla f_i(\mathbf{y}_{t+1}) - \nabla f_i(\mathbf{y}_t)\|^2\right] + \frac{4\sigma^2}{\delta},
\end{aligned}$$

where in (ii) we used the definition of $\mathcal{C}$. In (iii) we used the Young's Inequality and Assumption 3.5.

For all $t \in [T-2]$ we have:

$$\begin{aligned}
w'_{t+2}H_{t+1} &\leq w'_{t+2}(1-\frac{\delta}{2})H_t + \frac{4w'_{t+2}}{\delta n}\sum_{i=1}^{n}\mathbb{E}\left[\|\nabla f_i(\mathbf{y}_{t+1}) - \nabla f_i(\mathbf{y}_t)\|^2\right] + \frac{4w'_{t+2}\sigma^2}{\delta} \\
&\overset{(iv)}{\leq} w'_{t+1}(1-\frac{\delta}{4})H_t + \frac{4w'_{t+2}}{\delta n}\sum_{i=1}^{n}\mathbb{E}\left[\|\nabla f_i(\mathbf{y}_{t+1}) - \nabla f_i(\mathbf{y}_t)\|^2\right] + \frac{4w'_{t+2}\sigma^2}{\delta},
\end{aligned}$$

where (iv) uses the assumption. Note that by the algorithm we have $H_0 = 0$. Now we take $w'_{t+1}H_t$ to be α_t, $\frac{\delta}{4}$ to be γ, and $\frac{4w'_{t+2}}{\delta n}\sum_{i=1}^{n}\mathbb{E}\left[\|\nabla f_i(\mathbf{y}_{t+1}) - \nabla f_i(\mathbf{y}_t)\|^2\right] + \frac{4w'_{t+2}\sigma^2}{\delta}$ to be λ_t in Lemma B.2, we can then apply Lemma B.2 to get the desired result. □

Theorem 5.3. *Given Assumptions 3.2 to 3.5, $a_t \coloneqq {}^{(t+\frac{32}{\delta})}/_{M}$ and $A_0 \coloneqq {}^{32^2}/_{2\delta^2 M}$, it suffices to have:*

$$T = \mathcal{O}\left(\frac{R_0^2\sigma^2}{n\varepsilon^2} + \frac{\sqrt{L}R_0^2\sigma}{\delta^2\varepsilon^{3/2}} + \frac{\sqrt{\ell R_0^2}}{\delta^2\sqrt{\varepsilon}}\right), \tag{16}$$

number of iterations (communication rounds) of Algorithm 2 to get $F_T \leq \varepsilon$, where we can set $M = \max\left\{\frac{2^{13}\ell}{\delta^4}, \left(\frac{4T(T+\frac{32}{\delta})^2\sigma^2}{R_0^2 n}\right)^{\frac{1}{2}}, 8\left(\frac{LT(T+\frac{32}{\delta})^3\sigma^2}{\delta^4 R_0^2}\right)^{\frac{1}{3}}\right\}$.

Proof. We first further upper bound the weighted sum of $\bar{E}_t$ by noticing the following;

$$\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\|\nabla f_i(\mathbf{y}_{t+1})-\nabla f_i(\mathbf{y}_t)\|^2\right] \overset{(i)}{\leq} \frac{2}{n}\sum_{i=1}^{n}\left(\mathbb{E}\left[\|\nabla f_i(\mathbf{y}_{t+1})-\nabla f_i(\mathbf{x}_t)\|^2\right]+\mathbb{E}\left[\|\nabla f_i(\mathbf{x}_t)-\nabla f_i(\mathbf{y}_t)\|^2\right]\right)$$
$$\overset{(ii)}{\leq} \frac{4\ell}{n}\left(\beta_f(\mathbf{y}_{t+1},\mathbf{x}_t)+\beta_f(\mathbf{x}_t,\mathbf{y}_t)\right),$$

where in (i) we used the Young's inequality and in (ii) we used Assumption 3.4. We now apply this to Equation (13) from Lemma 5.1 and get:

$$\sum_{t=0}^{T-1} w_{t+1}\bar{E}_{t+1} \leq \frac{512\ell}{\delta^4}\sum_{t=1}^{T-1} w'_{t+1}(\beta_f(\mathbf{y}_t,\mathbf{x}_t)+\beta_f(\mathbf{x}_t,\mathbf{y}_{t-1}))+\frac{128\sigma^2}{\delta^4}\sum_{t=1}^{T-1} w'_{t+1}$$

Now consider $a_t \coloneqq \frac{t+\frac{32}{\delta}}{M}$ and $A_0 = \frac{32^2}{2\delta M}$. It's easy to verify that $w'_{t+1}(1-\frac{\delta}{2}) \leq w'_t(1-\frac{\delta}{4})$ and $w_{t+1}(1-\frac{\delta}{2}) \leq w_t(1-\frac{\delta}{4})$. We combine the above with Equation (10) from Theorem 4.2 and get the following:

$$A_T F_T \leq A_0 F_0 + \frac{R_0^2}{2} + \frac{2a_T^2 T\sigma^2}{n} + \frac{512\ell}{\delta^4}\sum_{t=1}^{T-1} w'_{t+1}(\beta_f(\mathbf{y}_t,\mathbf{x}_t)+\beta_f(\mathbf{x}_t,\mathbf{y}_{t-1}))+\frac{128\sigma^2}{\delta^4}\sum_{t=1}^{T-1} w'_{t+1}$$
$$-\sum_{t=0}^{T-1}\left(\frac{1}{6}-\frac{2La_{t+1}^2}{A_{t+1}}\right)a_{t+1}^2\mathbb{E}\left[\|\mathbf{g}_t\|^2\right]-\sum_{t=0}^{T-1}\left(\mathbb{E}\left[\frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t,\mathbf{x}^\star)+A_t\beta_f(\mathbf{y}_t,\mathbf{x}_t)+A_{t+1}\beta_f(\mathbf{x}_{t+1},\mathbf{y}_t)\right]\right)$$

We need:

$$\frac{2La_{t+1}^2}{A_{t+1}} \leq \frac{1}{6}, \quad \wedge \quad \frac{512\ell w'_{t+1}}{\delta^4} \leq A_t, \quad \wedge \quad \frac{512\ell w'_{t+2}}{\delta^4} \leq A_{t+1}$$

When $M \geq 24L$, we have $\frac{2La_{t+1}^2}{A_{t+1}} \leq \frac{1}{6}$. For such M, we have $w_t \leq \frac{8}{3}$. It now suffices to have $M \geq \frac{2^{13}\ell}{\delta^4}$ for all the above inequalities to be satisfied. Therefore:

$$A_T F_T \leq A_0 F_0 + \frac{R_0^2}{2} + \frac{2a_T^2 T\sigma^2}{n} + \frac{128\sigma^2}{\delta^4}\sum_{t=1}^{T-1} w'_{t+1}$$

Therefore:

$$F_T \leq \frac{2^{10}F_0}{\delta^2\left(T+\frac{32}{\delta}\right)^2}+\frac{MR_0^2}{\left(T+\frac{32}{\delta}\right)^2}+\frac{4T\sigma^2}{Mn}+\frac{2^9 LT\left(T+\frac{32}{\delta}\right)\sigma^2}{\delta^4 M^2}$$

Now we pick $M = \max\left\{24L, \frac{2^{13}\ell}{\delta^4}, \left(\frac{4T(T+\frac{32}{\delta})^2\sigma^2}{R_0^2 n}\right)^{\frac{1}{2}}, 8\left(\frac{LT(T+\frac{32}{\delta})^3\sigma^2}{\delta^4 R_0^2}\right)^{\frac{1}{3}}\right\}$, we get:

$$F_T \leq \frac{24LR_0^2}{(T+\frac{32}{\delta})^2}+\frac{2^{10}F_0}{\delta^2\left(T+\frac{32}{\delta}\right)^2}+\frac{2^{13}\ell R_0^2}{\delta^4\left(T+\frac{32}{\delta}\right)^2}+\frac{8L^{\frac{1}{3}}\sigma^{\frac{2}{3}}R_0^{\frac{4}{3}}}{\delta^{\frac{4}{3}}T^{\frac{2}{3}}}+\frac{2R_0\sigma}{\sqrt{Tn}}$$

It implies that, to reach ε-suboptimality, i.e. $F_T \leq \varepsilon$, we need:

$$T = \mathcal{O}\left(\frac{R_0^2\sigma^2}{n\varepsilon^2}+\frac{\sqrt{L}R_0^2\sigma}{\delta^2\varepsilon^{3/2}}+\frac{\sqrt{\ell R_0^2}}{\delta^2\sqrt{\varepsilon}}+\frac{\sqrt{LR_0^2}}{\delta\sqrt{\varepsilon}}\right),$$

Note that $L \leq \ell$ by Fact B.4 and we get the desired result. □

Algorithm 3 Accelerated Distributed Vanilla Error Feedback

Input: $\mathbf{x}_0, \mathbf{y}_0, \mathbf{v}_0, \mathbf{e}_0^i = \mathbf{0}, A_0, (a_t)_{t=1}^{\infty}$, and $\mathcal{C}$
for $t = 0, 1, \ldots$ **do**
  server side:
  $A_{t+1} = A_t + a_{t+1}$
  $\mathbf{y}_t = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_t$
  server send $\mathbf{x}_t, \mathbf{y}_t$ to the clients
  each client i**:**
  $\hat{\mathbf{g}}_t^i = \mathcal{C}(\frac{1}{a_{t+1}}\mathbf{e}_t^i + \mathbf{g}_t^i)$
  $\mathbf{e}_{t+1}^i = \mathbf{e}_t^i + a_{t+1}(\mathbf{g}_t^i - \hat{\mathbf{g}}_t^i)$
  send to server $\hat{\mathbf{g}}_t^i$
  server side:
  $\hat{\mathbf{g}}_t^i = \frac{1}{n}\sum_{i=1}^n \hat{\mathbf{g}}_t^i$
  $\mathbf{v}_{t+1} = \mathbf{v}_t - a_{t+1}\mathbf{g}_t^i$
  $\mathbf{x}_{t+1} = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_{t+1}$

E CONVERGENCE OF ACCELERATION WITH VANILLA EF

In this section, we consider an algorithm that combines acceleration with vanilla error feedback, summarized in Algorithm 3. We apply Theorem 4.2 and give a tailored upper bound on the weighted sum of E_t (in fact, as before, we upper bound the sum of $\bar{E}_t$). We need the following gradient similarity assumption:

Assumption E.1. We assume that there exists ζ^2 such that for all $\mathbf{x} \in \mathbb{R}^d$, it holds:

$$\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \zeta^2 \tag{26}$$

Since $(\mathbf{e}_t^i)_{t=0}^{\infty}$ also follows Lemma 5.1, it suffices that we give the following simple upper bound on H_t where $\widetilde{\mathbf{g}}_t^i = \mathbf{0}$:

Lemma E.2. *Given Assumptions 3.2, 3.3, 3.5 and E.1, for all $t \geq 0$, it holds that:*

$$H_t \leq 2\zeta^2 + 4L\mathbb{E}\left[\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right] + \sigma^2 \tag{27}$$

Proof.

$$\begin{aligned}
H_t &= \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\mathbf{g}_t^i\|^2\right] \\
&\overset{(i)}{\leq} \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\nabla f_i(\mathbf{y}_t)\|^2\right] + \sigma^2 \\
&\overset{(ii)}{\leq} \frac{2}{n}\sum_{i=1} n\mathbb{E}\left[\|\nabla f_i(\mathbf{y}_t) - \nabla f(\mathbf{y}_t)\|^2\right] + 2\mathbb{E}\left[\|\nabla f(\mathbf{y}_t)\|^2\right] + \sigma^2 \\
&\overset{(iii)}{\leq} 2\zeta^2 + 4L\mathbb{E}\left[\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right] + \sigma^2,
\end{aligned}$$

where in (i) we applied Assumption 3.5, in (ii) we used the Young's inequality, and in (iii) we used Assumptions 3.2, 3.3 and E.1. □

Now to give an upper bound on $\sum_{t=0}^{T-1} w_{t+1}E_{t+1}$:

Lemma E.3. *Given Assumptions 3.2, 3.3, 3.5 and E.1, for all $T \geq 1$, if $w_{t+1}(1 - \frac{\delta}{2}) \leq w_t(1 - \frac{\delta}{4})$, it holds that:*

$$\sum_{t=0}^{T-1} w_{t+1}E_{t+1} \leq \frac{32L}{\delta^2}\sum_{t=0}^{T-1} w'_{t+1}\mathbb{E}\left[\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right] + \frac{4(4\zeta^2 + \delta\sigma^2)}{\delta^2}\sum_{t=0}^{T-1} w'_{t+1} \tag{28}$$

Proof. We simply combine Equation (13) from Lemma 5.1 and Equation (27) from Lemma E.2 and get the desired result.

□

Now we can combine Equation (28) from Lemma E.3 with Equation (10) from Theorem 4.2 to get the following convergence rate:

Theorem E.4. *Given Assumptions 3.2, 3.3, 3.5 and E.1, and let* $a_t \coloneqq \frac{t+\frac{4}{\delta}}{M}$ *and* $A_0 \coloneqq \frac{8}{\delta^2 M}$*, it suffices to have:*

$$T = \mathcal{O}\left(\frac{\sigma^2 R_0^2}{\varepsilon^2} + \frac{\sqrt{L}R_0^2(\zeta/\sqrt{\delta}+\sigma)}{\sqrt{\delta}\varepsilon^{3/2}} + \frac{LR_0^2}{\delta\varepsilon}\right) \tag{29}$$

number of iterations of Algorithm 3 to get $F_T \leq \varepsilon$*, where we can set* $M = \max\{\frac{40L(T+\frac{4}{\delta})}{\delta}, \sqrt{\frac{4T(T+\frac{4}{\delta})^2\sigma^2}{R_0^2 n}}, \left(\frac{544L(4\zeta^2+\delta\sigma^2)T(T+\frac{4}{\delta})^3}{\delta^2 R_0^2}\right)^{\frac{1}{3}}\}$.

Proof. It's easy to check that for this choice of a_t and A_0, we have that $w_{t+1}(1-\frac{\delta}{2}) \leq w_t(1-\frac{\delta}{4})$.

In addition, we have that $\frac{La_{t+1}^2}{A_{t+1}} \leq \frac{2L}{M}$.

We combine Equation (10) from Theorem 4.2 and Equation (28) from Lemma E.3, pick $M \geq 24L$ and get the following:

$$\begin{aligned}
A_T F_T &\leq A_0 F_0 + \frac{R_0^2}{2} + \frac{2a_T^2 T\sigma^2}{n} \\
&\quad + \frac{32L}{\delta^2}\sum_{t=0}^{T-1} w'_{t+1}\mathbb{E}\left[\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right] + \frac{8(2\zeta^2+\sigma^2)}{\delta^2}\sum_{t=0}^{T-1} w'_{t+1} \\
&\quad - \sum_{t=0}^{T-1}\left(\mathbb{E}\left[\frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + A_t\beta_f(\mathbf{y}_t, \mathbf{x}_t) + A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\right]\right) \\
&\leq A_0 F_0 + \frac{R_0^2}{2} + \frac{2a_T^2 T\sigma^2}{n} \\
&\quad + \frac{32L^2}{\delta^2 M^3}\sum_{t=0}^{T-1}(t+17+\frac{4}{\delta})(t+1+\frac{4}{\delta})^2\mathbb{E}\left[\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right] \\
&\quad + \frac{4L(4\zeta^2+\delta\sigma^2)}{\delta^2 M^3}\sum_{t=0}^{T-1}(t+17+\frac{4}{\delta})(t+1+\frac{4}{\delta})^2 \\
&\quad - \sum_{t=0}^{T-1}\left(\mathbb{E}\left[\frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + A_t\beta_f(\mathbf{y}_t, \mathbf{x}_t) + A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\right]\right)
\end{aligned}$$

We need M such that:

$$\frac{32L^2(t+17+\frac{4}{\delta})(t+1+\frac{4}{\delta})^2}{\delta^2 M^2} \leq \frac{t+1+\frac{4}{\delta}}{2}, \forall 0 \leq t \leq T-1$$

It suffices that $M \geq \frac{40L(T+\frac{4}{\delta})}{\delta}$, and we have the following:

$$F_T \leq \frac{16F_0}{\delta^2(T+\frac{4}{\delta})^2} + \frac{MR_0^2}{(T+\frac{4}{\delta})^2} + \frac{4T\sigma^2}{Mn} + \frac{544L(4\zeta^2+\delta\sigma^2)T(T+\frac{4}{\delta})}{\delta^2 M^2}$$

Now pick $M = \max\{\frac{40L(T+\frac{4}{\delta})}{\delta}, \sqrt{\frac{4T(T+\frac{4}{\delta})^2\sigma^2}{R_0^2 n}}, \left(\frac{544L(4\zeta^2+\delta\sigma^2)T(T+\frac{4}{\delta})^3}{\delta^2 R_0^2}\right)^{\frac{1}{3}}\}$, we would have that after

$$T = \mathcal{O}\left(\frac{\sigma^2 R_0^2}{\varepsilon^2} + \frac{\sqrt{L}R_0^2(\zeta/\sqrt{\delta}+\sigma)}{\sqrt{\delta}\varepsilon^{3/2}} + \frac{LR_0^2}{\delta\varepsilon}\right)$$

iterations of Algorithm 3, we have $F_T \leq \varepsilon$. □

Algorithm 4 Accelerated Distributed Optimization with Absolute Compression

Input: $\mathbf{x}_0, \mathbf{y}_0, \mathbf{v}_0, \mathbf{e}_0^i = \mathbf{0}, A_0, (a_t)_{t=1}^{\infty}$, and $\mathcal{C}_\Delta$
for $t = 0, 1, \ldots$ **do**
  server side:
  $A_{t+1} = A_t + a_{t+1}$
  $\mathbf{y}_t = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_t$
  server send $\mathbf{x}_t, \mathbf{y}_t$ to the clients
  each client i**:**
  $\hat{\mathbf{g}}_t^i = \mathcal{C}_\Delta(\mathbf{g}_t^i)$
  send to server $\hat{\mathbf{g}}_t^i$
  server side:
  $\hat{\mathbf{g}}_t^i = \frac{1}{n}\sum_{i=1}^n \hat{\mathbf{g}}_t^i$
  $\mathbf{v}_{t+1} = \mathbf{v}_t - a_{t+1}\mathbf{g}_t^i$
  $\mathbf{x}_{t+1} = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_{t+1}$

Remark E.5. We notice that Algorithm 3 fail to achieve accelerated rate in the determinstic term and obtain the same rate as the unaccelerated error feedback. This is due to the lower bound where we need $M \geq \frac{40L(T+\frac{4}{\delta})}{\delta}$, which canceled out the acceleration effect. Such a requirement on the minimal M comes the fact that in Equation (10), we only have $\frac{a_{t+1}}{2}\mathbb{E}\left[\beta_f(\mathbf{y}_t, \mathbf{x}^\star)\right]$ to cancel out the error terms involving $\beta_f(\mathbf{y}_t, \mathbf{x}^\star)$. In contrast, in Algorithm 2, we applied the gradient difference compression, which enables us to the cancel out the errors with $A_t\mathbb{E}\left[\beta_f(\mathbf{y}_t, \mathbf{x}_t)\right]$ and $A_{t+1}\mathbb{E}\left[\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\right]$ and therefore achieve the accelerated $\mathcal{O}(\frac{1}{\sqrt{\varepsilon}})$ rate.

F CONVERGENCE OF ACCELERATED DISTRIBUTED OPTIMIZATION WITH ABSOLUTE COMPRESSION

In this section we consider a different kind of compressor, absolute compression:

Definition F.1. We say that a (possibly randomized) mapping $\mathcal{C}_\Delta \colon \mathbb{R}^d \to \mathbb{R}^d$ is an Δ-absolute compression operator if for some constant $\Delta \geq 0$ it holds

$$\mathbb{E}\left[\|\mathcal{C}(\mathbf{x}) - \mathbf{x}\|^2\right] \leq \Delta^2 \quad \forall \mathbf{x} \in \mathbb{R}^d. \tag{30}$$

This class of compressor also include some popular examples of compressions, including hard-thresholding (Sahu et al., 2021), rounding with bounded errors (Gupta et al., 2015), and scaled integer rounding (Sapio et al., 2021).

We summarize the accelerated distributed optimization with absolute compression method in Algorithm 4. Here the algorithm simply aggregates the compressed gradients and perform the update. We demonstrate our framework with Theorem 4.2 is very flexible and can also be applied to analyze Algorithm 4.

We pick the standard choice of a_t and A_0: $a_t \coloneqq \frac{t}{M}$ and $A_0 = 0$. We can apply Theorem 4.2 to get the following convergence rate:

Theorem F.2. *Given Assumptions 3.2, 3.3 and 3.5, and let $a_t \coloneqq \frac{t}{M}$ and $A_0 = 0$, then we have*

$$F_T \leq \frac{24LR_0^2}{T^2} + \sqrt{\frac{4R_0^2\sigma^2}{nT}} + \left(32LR_0^4\Delta^2\right)^{\frac{1}{3}}, \tag{31}$$

where we set $M = \max\{24L, \sqrt{\frac{4T^3\sigma^2}{nR_0^2}}, \left(\frac{2L\Delta^2(T+16)T^5}{R_0^2}\right)^{\frac{1}{3}}\}$.

Proof. With the choice of a_t, we can give an upper bound on E_t:

$$\begin{aligned}
E_t &= \mathbb{E}\left[\|\sum_{j=0}^{t-1} a_{j+1}(\hat{\mathbf{g}}_j - \mathbf{g}_j)\|^2\right] \\
&= \mathbb{E}\left[\|\sum_{j=0}^{t-1} a_{j+1}(\mathcal{C}_\Delta(\mathbf{g}_j) - \mathbf{g}_j)\|^2\right] \\
&\leq a_{t+1}^2 \mathbb{E}\left[\|\sum_{j=0}^{t-1} (\mathcal{C}_\Delta(\mathbf{g}_j) - \mathbf{g}_j)\|^2\right] \\
&\leq a_{t+1}^2 t \sum_{j=0}^{t-1} \mathbb{E}\left[\|\mathcal{C}_\Delta(\mathbf{g}_j) - \mathbf{g}_j\|^2\right] \\
&\leq a_{t+1}^2 t^2 \Delta^2
\end{aligned}$$

Therefore:

$$\begin{aligned}
\sum_{i=0}^{T-1} w_{t+1} E_{t+1} &\leq \frac{L\Delta^2}{M^3} \sum_{i=0}^{T-1} (t+17)(t+1)^2 t^2 \\
&\leq \frac{L\Delta^2 (T+16) T^5}{M^3}
\end{aligned}$$

Now we plug it into Equation (10) from Theorem 4.2, assume that $M \geq 24L$, we get the following:

$$\begin{aligned}
A_T F_T &\leq A_0 F_0 + \frac{R_0^2}{2} + \frac{2a_T^2 T\sigma^2}{n} + \frac{L\Delta^2(T+16)T^5}{M^3} \\
&= \frac{R_0^2}{2} + \frac{2T^3\sigma^2}{M^2 n} + \frac{L\Delta^2(T+16)T^5}{M^3}
\end{aligned}$$

Dividing both sides by A_T, we get:

$$F_T \leq \frac{MR_0^2}{T^2} + \frac{4T\sigma^2}{Mn} + \frac{2L\Delta^2(T+16)T^3}{M^2}$$

Now pick $M = \max\{24L, \sqrt{\frac{4T^3\sigma^2}{nR_0^2}}, \left(\frac{2L\Delta^2(T+16)T^5}{R_0^2}\right)^{\frac{1}{3}}\}$, we would have that:

$$F_T \leq \frac{24LR_0^2}{T^2} + \sqrt{\frac{4R_0^2\sigma^2}{nT}} + \left(32LR_0^4\Delta^2\right)^{\frac{1}{3}}.$$

□

Remark F.3. We see that Algorithm 4 converges up to some $\mathcal{O}(LR_0^4\Delta^2)$ neighborhood of the optimal solution. There exists some works that scales the compressed message by the inverse of the stepsize, see, e.g. Fatkhullin et al. (2023). We do not pursue this direction as here we are interested in the applicability of our framework Theorem 4.2, instead of the best possible rate.

G METHOD OF REPEATED COMMUNICATION

In this section we further discuss NEOLITHIC and the method of repeated communication, proposed by Huang et al. (2022) initially for unaccelerated methods, and later extended to accelerated methods by He et al. (2023). The method at its core is the same as compressed SGD and compressed accelerated SGD, where the server naively aggregates the compressed gradients and make an update. The only difference is that the compressor used here is a repeated compressor $\mathcal{C}_R$ that compresses the gradient by applying the basic contractive compressor $\mathcal{C}$ for R rounds (and therefore communicating with the server for R rounds). For completeness, we provide the precise definition of $\mathcal{C}_R$ in Algorithm 5, which is taken (with some simple reformulation for clarity) from He et al. (2023).

The key point of the repeated compressor $\mathcal{C}_R$ is that it reduces the error of the compression, we state the following simple fact from He et al. (2023):

Algorithm 5 Repeated Compressor $\mathcal{C}_R$

Input: $\mathbf{x}, \mathcal{C}$, and R
Client side:
$\mathbf{c}_0 = \mathbf{0}$
for $q = 1, 2, \cdots, R$ **do**
  $\Delta_q = \mathcal{C}(\mathbf{x} - \mathbf{c}_{q-1})$
  $\mathbf{c}_q = \mathbf{c}_{q-1} + \Delta_q$
Client sends $\{\Delta_q\}_{q\in[R]}$ to server
Server side:
$\Delta = \sum_{q=1}^{R} \Delta_q$

Algorithm 6 NEOLITHIC

Input: $\mathbf{x}_0, \mathbf{y}_0, \mathbf{v}_0, A_0, (a_t)_{t=1}^{\infty}, \mathcal{C}_R$ and M
for $t = 0, 1, \ldots$ **do**
  server side:
  $A_{t+1} = A_t + a_{t+1}$
  $\mathbf{y}_t = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_t$
  each client i**:**
  $\mathbf{g}_t^i = \mathbf{g}_i(\mathbf{y}_t, \xi_{\mathbf{y}_t}^i)$
  $\hat{\mathbf{g}}_t^i = \mathcal{C}_R(\mathbf{g}_t^i)$
  send to server Δ_t^i
  server side:
  $\hat{\mathbf{g}}_t = \frac{1}{n}\sum_{i=1}^{n} \mathbf{g}_t^i$
  $\mathbf{v}_{t+1} = \mathbf{v}_t - a_{t+1}\hat{\mathbf{g}}_t$
  $\mathbf{x}_{t+1} = \frac{A_t}{A_{t+1}}\mathbf{x}_t + \frac{a_{t+1}}{A_{t+1}}\mathbf{v}_{t+1}$

Lemma G.1. *Given a δ-contractive compressor $\mathcal{C}$ and R, the repeated compressor $\mathcal{C}_R$ satisfies the following:*

$$\mathbb{E}\left[\|\mathcal{C}_R(\mathbf{x}) - \mathbf{x}\|^2\right] \leq (1-\delta)^R \|\mathbf{x}\|^2, \forall \mathbf{x} \in \mathbb{R}^d \tag{32}$$

However, we point out that when R is some constant value independent of δ and δ is small enough (so that $R \leq \frac{1}{2\delta}$)

$$1 - R\delta \leq (1-\delta)^R \leq (1 - R\delta + R^2\delta^2) \leq (1 - \frac{R\delta}{2}),$$

implying that in such a case $\mathcal{C}_R$ is essentially an $R\delta$-contractive compressor. Therefore, in such a case when R is some fixed constant, the NEOLITHIC algorithm simply reduces to the compressed SGD with $R\delta$-contractive compressor. It is well known that compressed SGD diverges (Beznosikov et al., 2023) and therefore so will NEOLITHIC. This also explains why in the practical situation where R is some fixed constant, NEOLITHIC's performance is very poor, as observed in (Fatkhullin et al., 2023). See also Appendix H.1

He et al. (2023) suggested choosing $R = \Omega(\frac{1}{\delta}(\log\frac{1}{\delta} + \log\frac{1}{\varepsilon}))$ (where κ denotes the total number of updates at the server), i.e. in between each updates, the algorithm sends $\Omega(\frac{1}{\delta}(\log\frac{1}{\delta} + \log\frac{1}{\varepsilon}))$ messages compressed by $\mathcal{C}$ to approximate the original gradient. Consider the classical example of Top-K compressor, which is $\frac{K}{d}$-contractive compression, which sends K entries out of d entries of the uncompressed vector. In this case, $\Omega(\frac{1}{\delta}(\log\frac{1}{\delta} + \log\frac{1}{\varepsilon}))$ Top-K compressed messages will contain $\Omega(d\log\frac{1}{\varepsilon})$ entries in total, much more than the original uncompressed message. Note also that running Algorithm 5 with Top-K for exactly $\frac{d}{K}$ rounds would send the uncompressed vector *exactly*. More generally, if a δ-contractive compression $\mathcal{C}$ sends a τ fraction of bits of the uncompresed vector, then as shown in Zhu et al. (2023), in the worst case,

$$\tau = \Omega(\delta)$$

In other words, in the worst case, for any δ-contractive compression, NEOLITHIC sends a $\Omega(\log(\frac{1}{\delta}) + \log(\frac{1}{\varepsilon}))$ factor of the size of the uncompressed local gradient. For reasonable δ and ε, this factor is larger than one, implying, again, there is no compression at all.

Finally, with such high accuracy compression (with the relative error $(1-\delta)^R \leq \frac{1}{\kappa}$), He et al. (2023) showed that the compressed accelerated (and the unaccelerated) SGD's convergence is essentialy unaffected by the compression error. Therefore, NEOLITHIC performs, up to some constant, the same number of server side updates as the uncompressed method, that is, $\kappa = \mathcal{O}(\frac{R_0^2\sigma^2}{nB\varepsilon^2} + \frac{\sqrt{LR_0^2}}{\sqrt{\varepsilon}})$ with batch size B. Therefore, the total number of communication rounds becomes

$$R\mathcal{O}\left(\frac{R_0^2\sigma^2}{nB\varepsilon^2} + \frac{\sqrt{LR_0^2}}{\sqrt{\varepsilon}}\right) = \widetilde{\mathcal{O}}\left(\frac{R_0^2\sigma^2}{\delta nB\varepsilon^2} + \frac{\sqrt{LR_0^2}}{\delta\sqrt{\varepsilon}}\right),$$

where, importantly, the leading term has a $\widetilde{\mathcal{O}}(\frac{1}{\delta})$ factor.

G.1 CONVERGENCE ANALYSIS OF NEOLITHIC

In this section we provide an alternative analysis of NEOLITHIC based on Theorem 4.2, further demonstrating the flexibility of our proposed framework. NEOLITHIC is summarized in Algorithm 6. We point out that the presentation of the algorithm here is slightly different from the original presentation in He et al. (2023), where we made some modifications to their underlying acceleration method to aligh with ours. We analyze the convergence of Algorithm 6 with Assumption E.1, while He et al. (2023) analyzed it with a Bounded Local Objective Gap assumption (at optimum). For simplicity we keep Assumption E.1 and note that typically in convex setting, it is fairly simple to extend the bounded data heterogeneity assumptions from global to only at optimum. Note that we also give slightly different recomendation on R, the number of rounds of communications between each server update.

Now we upper bound the weighted sum of E_t:

Lemma G.2. *Given Assumptions 3.2, 3.3, 3.5 and E.1, we have:*

$$\begin{aligned}\sum_{t=0}^{T-1} w_{t+1}E_{t+1} \leq (1-\delta)^R\zeta^2 \sum_{t=0}^{T-1} w_{t+1}t\sum_{j=0}^{t-1} a_{j+1}^2 + 2L(1-\delta)^R \sum_{t=0}^{T-1} w_{t+1}t\sum_{j=0}^{t-1} a_{j+1}^2\beta_f(\mathbf{y}_j,\mathbf{x}^\star) \\ + (1-\delta)^R\sigma^2 \sum_{t=0}^{T-1} w_{t+1}t\sum_{j=0}^{t-1} a_{j+1}^2\end{aligned} \tag{33}$$

Proof.

$$\begin{aligned}E_{t+1} &= \mathbb{E}\left[\|\sum_{j=0}^{t-1} a_{j+1}\mathbf{g}_j - a_{j+1}\frac{1}{n}\sum_{i=1}^n \hat{\mathbf{g}}_j^i\|^2\right] \\ &\leq t\sum_{j=0}^{t-1} a_{j+1}^2\mathbb{E}\left[\|\mathbf{g}_j - \frac{1}{n}\sum_{i=1}^n \hat{\mathbf{g}}_j^i\|^2\right] \\ &\leq \frac{t}{n}\sum_{j=0}^{t-1} a_{j+1}^2\sum_{i=1}^n \mathbb{E}\left[\|\mathbf{g}_j - \hat{\mathbf{g}}_t^i\|^2\right] \\ &\leq \frac{(1-\delta)^R t}{n}\sum_{j=0}^{t-1} a_{j+1}^2\sum_{i=1}^n \mathbb{E}\left[\|\mathbf{g}_j\|^2\right] \\ &\leq \frac{(1-\delta)^R t}{n}\sum_{j=0}^{t-1} a_{j+1}^2\sum_{i=1}^n \mathbb{E}\left[\|\nabla f_i(\mathbf{y}_j)\|^2\right] + (1-\delta)^R t\sigma^2\sum_{j=0}^{t-1} a_{j+1}^2\end{aligned}$$

We note the following:

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\nabla f_i(\mathbf{y}_j)\|^2\right] &= \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\|\nabla f_i(\mathbf{y}_j) - \nabla f(\mathbf{y}_j)\|^2\right] + \mathbb{E}\left[\|\nabla f(\mathbf{y}_j)\|^2\right] \\ &\leq \zeta^2 + 2L\beta_f(\mathbf{y}_j,\mathbf{x}^\star)\end{aligned}$$

Plugging it back in, we get:

$$E_{t+1} \leq (1-\delta)^R\zeta^2 t\sum_{j=0}^{t-1} a_{j+1}^2 + 2L(1-\delta)^R t\sum_{j=0}^{t-1} a_{j+1}^2\beta_f(\mathbf{y}_j,\mathbf{x}^\star) + (1-\delta)^R t\sigma^2\sum_{j=0}^{t-1} a_{j+1}^2$$

Summing this up, we get the desired result. □

Now we can combine the Theorem 4.2 and Lemma G.2 to get the convergence rate of NEOLITHIC:

Theorem G.3. *Given Assumptions 3.2, 3.3, 3.5 and E.1, and let* $a_t \coloneqq \frac{t}{M}, A_0 = 0$, *and* $R \coloneqq \max\{\frac{4}{\delta}\ln(T), \frac{1}{\delta}\ln(\frac{4nT^2}{3}), \frac{1}{\delta}\ln(\frac{4n\zeta^2T^2}{3\sigma^2})\}$, *it suffices to take*

$$RT = \mathcal{O}\left(\left(\frac{R_0^2\sigma^2}{\delta n\varepsilon^2} + \frac{1}{\delta}\sqrt{\frac{LR_0^2}{\varepsilon}}\right)\left(\ln(nT) + \ln(\frac{n\zeta^2T^2}{\sigma^2})\right)\right) \tag{34}$$

total number of communication rounds to get $F_T \leq \varepsilon$, *where we can set* $M = \max\{24L, \sqrt{\frac{12T^3\sigma^2}{nR_0^2}}\}$.

Proof. As before, we assume that $M \geq 24L$. Now we have $w_t \leq \frac{8}{3}$.

$$\begin{aligned}
A_TF_T &\leq A_0F_0 + \frac{R_0^2}{2} + \frac{2a_T^2T\sigma^2}{n} \\
&\quad + (1-\delta)^R\zeta^2\sum_{t=0}^{T-1} w_{t+1}t\sum_{j=0}^{t-1} a_{j+1}^2 + 2L(1-\delta)^R\sum_{t=0}^{T-1} w_{t+1}t\sum_{j=0}^{t-1} a_{j+1}^2\beta_f(\mathbf{y}_j, \mathbf{x}^\star) \\
&\quad + (1-\delta)^R\sigma^2\sum_{t=0}^{T-1} w_{t+1}t\sum_{j=0}^{t-1} a_{j+1}^2 \\
&\quad - \sum_{t=0}^{T-1}\left(\mathbb{E}\left[\frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + A_t\beta_f(\mathbf{y}_t, \mathbf{x}_t) + A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\right]\right) \\
&\leq A_0F_0 + \frac{R_0^2}{2} + \frac{2a_T^2T\sigma^2}{n} \\
&\quad + \frac{8(1-\delta)^R\zeta^2}{3M^2}\sum_{t=0}^{T-1} t^4 + \frac{16L(1-\delta)^RT^4}{3M^2}\sum_{t=0}^{T-1}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + \frac{8(1-\delta)^R\sigma^2}{3M^2}\sum_{t=0}^{T-1} t^4 \\
&\quad - \sum_{t=0}^{T-1}\left(\mathbb{E}\left[\frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + A_t\beta_f(\mathbf{y}_t, \mathbf{x}_t) + A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\right]\right) \\
&\leq A_0F_0 + \frac{R_0^2}{2} + \frac{2a_T^2T\sigma^2}{n} \\
&\quad + \frac{8(1-\delta)^R\zeta^2T^5}{3M^2} + \frac{16L(1-\delta)^RT^4}{3M^2}\sum_{t=0}^{T-1}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + \frac{8(1-\delta)^R\sigma^2T^5}{3M^2} \\
&\quad - \sum_{t=0}^{T-1}\left(\mathbb{E}\left[\frac{a_{t+1}}{2}\beta_f(\mathbf{y}_t, \mathbf{x}^\star) + A_t\beta_f(\mathbf{y}_t, \mathbf{x}_t) + A_{t+1}\beta_f(\mathbf{x}_{t+1}, \mathbf{y}_t)\right]\right)
\end{aligned}$$

Now we need the following:

$$\frac{16L(1-\delta)^RT^4}{3M} \leq \frac{t+1}{2} \quad \wedge \quad \frac{8(1-\delta)^RT^5}{3} \leq \frac{2T^3}{n} \quad \wedge \quad \frac{8(1-\delta)^R\zeta^2T^5}{3} \leq \frac{2T^3\sigma^2}{n}$$

it suffices if $R \coloneqq \max\{\frac{4}{\delta}\ln(T), \frac{1}{\delta}\ln(\frac{4nT^2}{3}), \frac{1}{\delta}\ln(\frac{4n\zeta^2T^2}{3\sigma^2})\}$. With this R, we have:

$$A_TF_T \leq \frac{R_0^2}{2} + \frac{6T^3\sigma^2}{M^2n}$$

Therefore, with $M = \max\{24L, \sqrt{\frac{12T^3\sigma^2}{nR_0^2}}\}$, we have that it suffices to take

$$T = \mathcal{O}(\frac{R_0^2\sigma^2}{n\varepsilon^2} + \sqrt{\frac{LR_0^2}{\varepsilon}})$$

number of server updates to achieve ε-accuracy, as desired. □

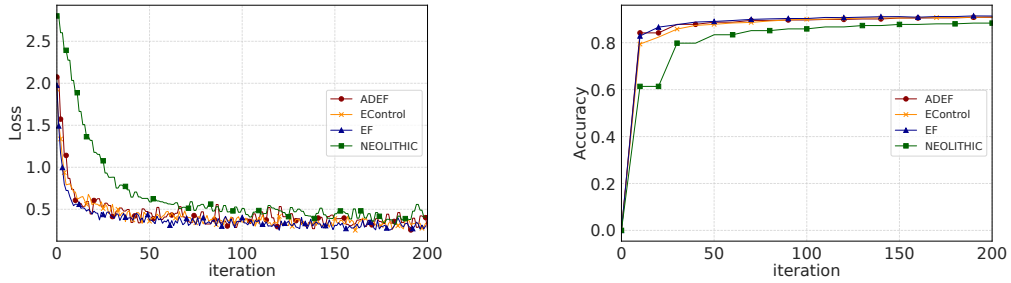


Figure 4: **Competitive performance.** Comparison of the performance of ADEF, EConrol,EF and NEOLITHIC on the MNIST classification problem. We use Top-K compression with $\delta = 0.1$. We see that ADEF performs competitively in both the loss and the accuracy, while NEOLITHIC performs the worst.

H ADDITIONAL EXPERIMENTS AND DETAILS

In this section, we discuss some additional experiments and details that are not included in the main paper. All of our experiment code can be accessed at `https://github.com/mlolab/adef_aistats.git`

H.1 MNIST CLASSIFICATION

We also evaluate our algorithm on the MNIST classification problem (Deng, 2012) when using Top-K compression with $\delta = 0.1$, and compare it against the state of the art distributed optimization algorithms with communication compression, in particular, the classical EF algorithm, and the recent EControl algorithm. We also compare against the NEOLITHIC method, where we set the number of repeated communication rounds to be 2. The results are summarized in Figure 4. We see that ADEF (with $\gamma = 0.05$ as selected by grid search) performs competetively in both the loss and the accuracy, while NEOLITHIC performs the worst.

H.2 ADDITIONAL DETAILS OF THE SOFT MAX EXPERIMENTS

We set the dimension d to be $d = 200$, total number of samples $k = 2048$, and the number of clients to be $n = 4$. We randomly generate the data $\{\mathbf{a}_i, b_i\}$ following the approach of Moshtaghifar et al.: we generate i.i.d. vectors $\hat{\mathbf{a}}_i$ whose entries are sampled from $[-1, 1]$ uniformly at random. Each b_i is generated the same way. This leads to a preliminary objective $\hat{f}$. We then set $\mathbf{a}_i := \hat{\mathbf{a}}_i - \nabla \hat{f}(\mathbf{0})$. The resulting $\{\mathbf{a}_i, b_i\}$ gives us the desired objective f with $\mathbf{0}$ being the minimizer.

We perform a grid search for the stepsizes over $\gamma \in \{0.0001, 0.0005, 0.001, 0.005, 0.01\}$.

H.3 DETAILS OF THE CIFAR-10 EXPERIMENTS SETUP

We use the batch size 256, and we tune the stepsizes ranging from 0.5 to 0.005 for all methods. We use the Top-K compressor with $\delta = 0.1$.

For NEOLITHIC, we also fine-tune the number of repeated communication rounds over $\{1, 2, 4, 6, 8, 10\}$ and the selected number is 2. Note that if the number of repeated communication round is 10, then NEOLITHIC reduces to the usual uncompressed accelerated gradient method, but with a 10 times higher per-iteration cost than the basic compressed method. We also point out that our method ADEF sends 2 compressed messages per iteration. For EControl we set the η parameter to be 0.1 as suggested by Gao et al. (2024).

H.4 COMPUTATIONAL RESOURCES

The experiments are run on a platform with 4 virtual Intel(R) Xeon(R) CPU @ 2.30GHz CPUs and 4 NVIDIA T4 GPUs each with 15GB vRAM.