AC-SGD: Adaptively Compressed SGD for Communication-Efficient Distributed Learning

Guangfeng Yan, Tan Li *Student Member, IEEE,* Shao-Lun Huang, Tian Lan *Member, IEEE* and
and Linqi Song *Senior Member, IEEE*

***Abstract*—Gradient compression (e.g., gradient quantization and gradient sparsification) is a core technique in reducing communication costs in distributed learning systems. The recent trend of gradient compression is to use a varying number of bits across iterations, however, relying on empirical observations or engineering heuristics without a systematic treatment and analysis. To the best of our knowledge, a general dynamic gradient compression that leverages both quantization and sparsification techniques is still far from understanding. This paper proposes a novel Adaptively-Compressed Stochastic Gradient Descent (AC-SGD) strategy to adjust the number of quantization bits and the sparsification size with respect to the norm of gradients, the communication budget, and the remaining number of iterations. In particular, we derive an upper bound, tight in some cases, of the convergence error for arbitrary dynamic compression strategy. Then we consider communication budget constraints and propose an optimization formulation - denoted as the *Adaptive Compression Problem (ACP)* - for minimizing the deep model's convergence error under such constraints. By solving the ACP, we obtain an enhanced compression algorithm that significantly improves model accuracy under given communication budget constraints. Finally, through extensive experiments on computer vision and natural language processing tasks on MNIST, CIFAR-10, CIFAR-100 and AG-News datasets, respectively, we demonstrate that our compression scheme significantly outperforms the state-of-the-art gradient compression methods in terms of mitigating communication costs.**

***Index Terms*—Distributed Learning, Communication-efficient, Quantization, Sparsification**

I. Introduction

Stochastic gradient descent (SGD) is a widely-used optimization technology to machine learning tasks, due to its lower computational complexity and good empirical performances. However, the traditional centralized SGD framework cannot cope with massive data nowadays. Instead, distributed SGD [2], [3] utilizes local user data to build distributed models and transfers local gradients among distributed nodes and a parameter servers until all nodes reach a global consensus on the learning model.

However, the explosion of edge devices, such as mobile phones and wearable devices, and an increase in model size make communication the bottleneck for distributed SGD training. Three typical communication reduction schemes have been proposed to improve the efficiency of distributed SGD. Quantization [4] and sparsification [5] reduce communication overhead by scaling down the uploaded model size. In particular, quantization encode the original gradient vectors to smaller bits while sparsification drops out less informative elements. Another idea is to reduce the number of communication rounds between distributed nodes and server. For example, periodic or other less-frequent model updates [6], [7]. Some work consider a combination of the above three schemes [8], [9]. In this work, we mainly focus on the first two techniques, namely quantization and sparsification.

Above works still lack exploration on two aspects: 1) Dynamic compression level adjustment. Most current work uses a fixed compression level, i.e., the fixed quantization bits or sparse size during the whole training process but ignore the fact that the statistics of gradients of models change during training. In contrast to them, [10]–[13] have shown from empirical observations that dynamically adjusting the compression level can achieve a better convergence compared to the fixed scheme. Different from them, our work gives the compression level allocation rule for each iteration from a theoretical perspective. 2) A systematic framework to characterize the trade-off between the explicit communication budget and learning performance. Most of the existing work reveals from the experimental results that the smaller communication cost (the larger compression level) leads to lower model accuracy. We take into account of an explicit communication budget constraints - which limit the total number of bits available for transferring gradients during the entire training process, and further characterizes the trade-off relationship between this budget and convergence error.

This paper develops a novel Adaptively-Compressed SGD (AC-SGD) strategy for dynamic gradient descent to achieve communication-efficient distributed SGD. We propose an optimization formulation - denoted as the *Adaptive Compression Problem (ACP)* - for finding a compressor and the optimal dynamic compression level at each iteration by minimizing the convergence error under such constraints. To solve the problem, We propose SQ Compressor (Sparsification-Quantization Compressor), that collaboratively leverages both quantization and sparsification techniques. At each iteration step, we first optimizes the dynamic compression level for each client and then determines the optimal quantization bits and sparsification size for the SQ compressor by minimizing the gradient variance under the current compression level.

G. Yan, T. Li and L. Song are with the Department of Computer Science, City University of Hong Kong, and City University of Hong Kong Shenzhen Research Institute, Shenzhen, China. S. Huang is with Data Science and Information Technology Research Center, Tsinghua-Berkeley Shenzhen Institute. T. Lan is with the Department of Electrical and Computer Engineering, George Washington University. Email: gfyan2-c@my.cityu.edu.hk,tanli6-c@my.cityu.edu.hk, twn2gold@gmail.com, tlan@gwu.edu, linqi.song@cityu.edu.hk (Corresponding author: Linqi Song).
This work was presented in part at 2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS) [1]

We use convergence error to qualify the gap between the loss of learned and the optimal model. For strongly convex problems, it is shown that the convergence error consists of two parts: an error due to the stochastic gradient descent method and an error resulted from quantization and sparsification, which diminishes to zero as the communication budget increases. The upper and lower bounds on the convergence error are derived and proven to be tight for a special case of quadratic functions with the isotropic Hessian matrix. For non-convex learning problems, different from the ordinary mean square of the gradient norms used in previous work [8], [14], [15], we derive an upper bound on the weighted mean square of gradient norms at each iteration step, termed the non-convex convergence error. We give more significant weight to the gradient norm in the later stage of training, which better characterize the convergence characteristics of non-convex problems.

This paper makes the following key contributions:

- We propose a novel framework, AC-SGD, to realize communication budget aware distributed learning by unifying dynamic gradient quantization and sparsification.
- We first propose the SQ Compressor (Sparsification-Quantization Compressor), which determines the optimal quantization bits and sparsification size by minimizing the gradient variance under the communication bit constraint. Based on this designed compressor, our proposed AC-SGD algorithm leverages an optimization problem to jointly adjust the compression level with respect to the norm of gradients, the communication budget, and the remaining number of iterations.
- Our theoretical results characterize the trade-off between communication budget and convergence error. Specifically, we derive an upper bound on the convergence error of compressed stochastic gradient descent for both strongly convex objectives and non-convex objectives. The upper bound is shown to be tight in special cases.
- We validate our theoretical analysis through extensive experiments some machine learning tasks, including image classification tasks on MNIST, CIFAR-10 and CIFAR-100 and text classification tasks on AG-News. The results demonstrate significant improvement over state-of-the-art gradient compression methods in terms of mitigating communication costs.

II. Related Work

The use of quantization and sparsification for communication-efficient gradient methods has decades rich history and its recent use in training deep neural networks has re-ignited interest [4], [5], [16]–[18].

Static Compression A number of current works focus on static compression schemes, including quantization and sparsification, where the quantization bits or sparsification size used in the training process are fixed in advance.

For quantization, Sign SGD uses 1 bit to quantize each dimension of the gradients [16]. QSGD [4] and k-level quantization [19] are stochastic quantization schemes that can quantize elements into arbitrary bits.

For sparsification, Top-k is a biased sparsifier which retains largest k elements of the gradient vector, and sets the rest elements to 0 [5]. Rand-k sparsifier randomly drops out some elements and amplifies the remaining elements appropriately to ensure the sparsified gradient unbiased [18]. TCS [20] seeks a certain correlation between the sparse representations used at consecutive iterations in FL, to reduce the overhead of encoding the non-zero positional information. PowerSGD [21] performs a low-rank linear transformation to sparsify the model (reduce the number of parameters).

Some recent work jointly uses these two techniques [8], [9] for more efficient communication. In particular, [8] combines aggressive sparsification with quantization by keeping track of the difference between the original and compressed gradients. [9] defines the gradient magnitude to indicate the importance. Then, the gradient whose magnitude is larger than a certain threshold will be quantized to a fixed number of bits and transmitted.

Adaptive Compression However, as the statistics of gradients change during training, it is unwise to consider the fixed compression level during the whole training process. Some recent studies have started to construct adaptive compression schemes to dynamically decide the compression level using empirical observations or engineering heuristics. [12] and [13] determine the compression level according to gradient's size. Anders [10] demonstrates that using few quantization bits in the early epochs and gradually increase bits in the later epochs. MQGrad [11] formulates the quantization determination problem as an online learning problem where the states record historical information from the past optimization iterations. AdaComp [22] is based on localized selection of gradient residues and automatically tunes the compression rate depending on local activity. MIPD [23] adaptively compresses the gradients based on model interpretability and probability distribution of gradients. Different from above heuristic compression scheme, recently, [24] and [25] proposed adaptive compression from the theoretical perspective. [24] adaptively adjust the quantization points to minimize the variance of vector quantization while [25] adaptively computes the scaling factors for integer rounding operators. However, none of them consider the communication budget constrained setting for adaptive compression.

Error Compensation To compensate the compression errors and accelerate the learning speed, various error compensation techniques have been introduced [5], [26]. ScaleCom [27] explores the similarity of gradient distribution across clients to provide a scalable error compensation of Top-k compressors. CSER [28] has developed an error compensation method termed error reset to help compressors speed up the learning speed.

To sum up, the above works either fix the compression level in advance or adjust the level according to engineering heuristics, which may reach contradicted conclusions, e.g., MQGrad [11] and AdaQS [10] suggest using few quantization bits in early epochs and gradually increasing the number of bits in later epochs, while Anders [12] chooses to use more bits in the early training stage and fewer bits in the later stage. Error compensation is a parallel technique to accelerate the learning speed of different compressors. Part of this work has been presented in [1], which gives the theoretical framework of adaptive quantization. This work extends the previous algorithm to jointly consider quantization and sparsification. To our best knowledge, our proposed AC-SGD is the first to propose a systematic *communication budget aware* framework to adaptively adjust the compression level with respect to the gradient norm, the number of remaining training iteration and available communication budget. In Table I, we compare the properties of AC-SGD with existing works.

TABLE I
COMPARISON OF AC-SGD AND RELATED WORKS.

	Quantization	Sparsification	Adaptive
[16], [17], [4], [19]	✓	×	×
[5], [18], [20], [21], [27]	×	✓	×
[10]–[12]	✓	×	✓ (Heuristic)
[13]	×	✓	✓ (Heuristic)
[8], [9]	✓	✓	×
[1] (Our previous conf.)	✓	×	✓ (Theoretical)
AC-SGD (Our work)	✓	✓	✓ (Theoretical)

III. SYSTEM FRAMEWORK AND PROBLEM FORMULATION

A. Distributed SGD

A distributed learning system consists of W clients and one parameter server (PS). Each client can perform local learning based on dataset D_i.The server coordinates the optimization of a set of global parameters $\mathbf{x}$ by minimizing the objective function $F : \mathbb{R}^d \rightarrow \mathbb{R}$ without sharing local datasets:

$$\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x}) = \min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{W}\sum_{i=1}^{W} L_i(\mathbf{x}, D_i) \tag{1}$$

where $L_i(\mathbf{x}; D_i)$ is the local objective, computed by

$$L_i(\mathbf{x}, D_i) = \mathbb{E}_{\xi\sim D_i}[l_i(\mathbf{x}; \xi)], \tag{2}$$

where with $l_i(\cdot,\cdot)$ is a user-specific loss function. Before introducing the proposed framework, we first explain the Distributed SGD [6], [29], which is a benchmark distributed optimization algorithm widely used to solve the problem 1. At each iteration t, the clients download the aggregated model $\mathbf{x}_{t-1}$ from the server, perform local optimization minimizing an empirical objective $L_i(\mathbf{x}, D_i)$ with learning rate η using a local optimizer SGD, and then send the local gradients $\mathbf{g}_t^{(i)}$ back to the server. The server averages the solutions obtained from the clients by: $\mathbf{x}_t = \mathbf{x}_{t-1} - \frac{\eta}{W}\sum_{i=1}^{W}\mathbf{g}_t^{(i)}$. The procedure is iterated for T iterations.

B. Communication-Efficient Distributed SGD

From the above learning process, we can see that the clients and the server must iteratively exchange their model parameters over the links. This inevitably introduces significant communication overhead, especially in scenarios with massive edge devices like IoT. To reduce the communication cost, we propose the communication-efficient SGD framework, shown in Fig. 1 (Considering that in the actual scenario, the downlink speed of the device is usually much higher than the upload speed, this work will follow the previous work settings [5], [17], only consider compressing the upload gradient, and do not deal with the downlink parameters.). Each client i has a restricted communication budget $C^{(i)}$, indicating the total communication resources (in bits) that client i can use during the whole training process. The key idea is to force each client to adaptively compress its local stochastic gradients before sending them to the server.

The system operates over T iterations. Each iteration t consists of four steps. 1) The model distribution step follows the standard distributed SGD; 2) After local optimization, client i employs a compressor operator $\mathcal{C}_{c_t^{(i)}}[\cdot]$ to compress gradient $\mathbf{g}_t^{(i)}$ to $\hat{\mathbf{g}}_t^{(i)}$ of size $c_t^{(i)}$ bits; 3) The compressed local gradients are sent to the server and 4) the global model is optimized by:

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \frac{\eta}{W}\sum_{i=1}^{W}\hat{\mathbf{g}}_t^{(i)} = \mathbf{x}_{t-1} - \frac{\eta}{W}\sum_{i=1}^{W}\mathcal{C}_{c_t^{(i)}}[\mathbf{g}_t^{(i)}] \tag{3}$$

Our goal is to solve problem (1) with constraint $\sum_{t=0}^{T-1} c_t^{(i)} \leq C^{(i)}$ for $i = 1, ..., W$. The most challenged parts are: 1) how to determine the compression bits $c_t^{(i)}$? and 2) How to construct the compress operator $\mathcal{C}_{c_t^{(i)}}[\cdot]$? To address these two issues, we propose an algorithm, called Adaptively Compress SGD (AC-SGD) in the next section by solving some optimization problems iteratively. Before that, we first make some commonly used assumptions for stochastic gradients $\mathbf{g}_t^{(i)}$ and objective function $F(\mathbf{x})$.

Assumption 1 (Unbiasness and Bounded Variance of Stochastic Gradient [30], [31])**.** *The stochastic gradient oracle gives us an independent unbiased estimate* $\mathbf{g}$ *with a bounded variance:*

$$\mathbb{E}_{\xi\sim D_i}[\mathbf{g}_t^{(i)}] = \nabla F(\mathbf{x}_t), \tag{4}$$

$$\mathbb{E}_{\xi\sim D_i}[\|\mathbf{g}_t^{(i)} - \nabla F(\mathbf{x}_t)\|^2] \leq \sigma^2. \tag{5}$$

Assumption 2 (Smoothness [32])**.** *The objective function* $F(\mathbf{x})$ *is L-smooth, which means* $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, $\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$.

It implies that $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have

$$F(\mathbf{y}) \leq F(\mathbf{x}) + \nabla F(\mathbf{x})^{\mathrm{T}}(\mathbf{y} - \mathbf{x}) + \frac{L}{2}\|\mathbf{y} - \mathbf{x}\|^2 \tag{6}$$

$$\|\nabla F(\mathbf{x})\|^2 \leq 2L[F(\mathbf{x}) - F(\mathbf{x}^*)] \tag{7}$$

Assumption 3 (Strong convexity [30])**.** *The objective function* $F(\mathbf{x})$ *is* μ*-strongly convex, which means* $\exists\mu > 0$, $F(\mathbf{x}) - \frac{\mu}{2}\mathbf{x}^{\mathrm{T}}\mathbf{x}$ *is a convex function.*

From Assumption 3, we have: $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$,

$$F(\mathbf{y}) \geq F(\mathbf{x}) + \nabla F(\mathbf{x})^{\mathrm{T}}(\mathbf{y} - \mathbf{x}) + \frac{\mu}{2}\|\mathbf{y} - \mathbf{x}\|^2 \tag{8}$$

$$\|\nabla F(\mathbf{x})\|^2 \geq 2\mu[F(\mathbf{x}) - F(\mathbf{x}^*)] \tag{9}$$

IV. Adaptively-Compressed SGD

In this section, We first investigate the properties for the designed SQ Compressor, which leverages both quantization and sparsification operations. We then propose a dynamic compression level adjustment algorithm for this compressor, called AC-SGD algorithm, and exam its learning performance towards convergence error.

A. SQ Compressor Design

Generally, we can choose unbiased and biased compressors. However, commonly used biased compressors usually require strong statistical assumptions and may lead to divergence, as shown in [33]. Therefore, we focus on the unbiased compressor in our design[1]. We propose the compression scheme as a combination of $Rand_k$ sparsification and stochastic uniform quantization, which are unbiased with bounded variance. Specifically, we form our compression operator $\mathcal{C}_c[\mathbf{g}]$ as $\mathcal{Q}_b[\mathcal{S}_k(\mathbf{g})]$, where $\mathcal{Q}_b[\cdot]$ is a quantizer with b quantization bits; $\mathcal{S}_k(\cdot)$ is a sparsifier with k sparsification size.

Sparsification: Similar to the operation in [5], [18], we let $\mathcal{S}_k(\mathbf{g}) = \frac{d}{k} Rand_k(\mathbf{g})$. For vector $\mathbf{g}$, the sparsifier randomly selects k elements, enlarges them by $\frac{d}{k}$ times, and then sets the remaining $d-k$ elements to 0.

Quantization: There are several types of quantization operations – categorized from different perspectives, such as grid quantization, uniform and non-uniform quantization, biased and unbiased quantization. Here, we adopt a family of stochastic uniform quantization, similar to [4], to quantize the selected k elements. The j-th component of the vector $\mathcal{S}_k(\mathbf{g})$ is quantized as $\mathcal{Q}_b[\mathcal{S}_k(\mathbf{g})_j] = \|\mathcal{S}_k(\mathbf{g})\| \cdot \mathrm{sgn}(\mathcal{S}_k(\mathbf{g})_j) \cdot \zeta(\mathcal{S}_k(\mathbf{g})_j, s)$. Here $\|\mathcal{S}_k(\mathbf{g})\|$ is the l_2 norm of $\mathcal{S}_k(\mathbf{g})$; $\mathrm{sgn}(\mathcal{S}_k(\mathbf{g})_j) = \{+1, -1\}$ is the sign of $\mathcal{S}_k(\mathbf{g})_j$; $s = 2^{b-1}$ is the quantization level; and $\zeta(S_k(\mathbf{g})_j, s)$ is an unbiased stochastic function that maps scalar $|\mathcal{S}_k(\mathbf{g})_j|/\|\mathcal{S}_k(\mathbf{g})\|$ to one of the values in set $\{0, 1/s, 2/s, \ldots, s/s\}$.

The following Lemma characterize the properties of the compressed gradient $\hat{\mathbf{g}} = \mathcal{Q}_b[\mathcal{S}_k(\mathbf{g})]$.

Lemma 1 (Unbiasness and Bounded Variance of SQ-Compressor)**.** *For gradient vector $\mathbf{g} \in \mathbb{R}^d$, if the number of quantization bits is b and the sparsification size is k, then the compressed vector $\hat{\mathbf{g}} = \mathcal{Q}_b[\mathcal{S}_k(\mathbf{g})]$ satisfies:*

$$\mathbb{E}[\hat{\mathbf{g}}] = \mathbf{g} \tag{10}$$

and

$$\mathbb{E}\left[||\hat{\mathbf{g}}||^2\right] \leq \|\mathbf{g}\|^2 + \left[\frac{d-k}{k} + \frac{d}{4^b}\right]\|\mathbf{g}\|^2, \tag{11}$$

The full proof is shown in Appendix. A. Eq. (10) means that the compressed gradient $\hat{\mathbf{g}}$ is the unbiased estimate of $\mathbf{g}$. We focus on Eq. (11), that implies the noise added on the uncompressed gradient, defined as:

$$h(k,b) \triangleq \frac{d-k}{k} + \frac{d}{4^b} \tag{12}$$

Smaller $h(k,b)$ implies less information loss during compression, but that usually means more bits are needed to express $\mathbf{g}$. Thus, the optimal design of the compressor can be found by minimizing $h(k,b)$ with the bits constrain c:

$$\begin{aligned} \min_{b,k} \quad & h(k,b) \\ s.t. \quad & k(b + \log_2 d) + B_{pre} = c \end{aligned} \tag{13}$$

where c is the constricted bits of $\hat{\mathbf{g}}$; $\log_2 d$ is the additional number of bits to encode the indices of the $Rand_k$ elements and B_{pre} is the number of bits of full-precision floating point (e.g., $B_{pre} = 32$ or 64) to represent $\|\mathcal{S}_k(\mathbf{g})\|$ after sparsification operation.

By solving the above optimization problem, determine the parameters of our SQ compressor :

$$b^* = \frac{1}{2}\log_2\left[2\ln 2 * (c - B_{pre})\right] \tag{14}$$

$$k^* = \frac{c - B_{pre}}{\frac{1}{2}\log_2\left[2\ln 2 * (c - B_{pre})\right] + \log_2 d} \tag{15}$$

Substituting b^* and k^* to (12), we have

$$h(k^*, b^*) \leq \frac{3d\log_2 d}{2c} \tag{16}$$

This is the upper bound of error introduced by SQ compressor with optimal parameters b and k given allocated bit c.

B. Compression Level Allocation

From Eqs. (14) and (15), we find that values of $b_t^{(i)}$ and $k_t^{(i)}$ depend oh the given $c_t^{(i)}$. Therefore, we next discuss how to adaptively adjust the compression level the $c_t^{(i)}$ at each iteration t for SQ-Compressor.

We formulate the compression level decision problem as a performance loss minimization problem under the communication constraints, called *Adaptive Compression Problem (ACP)* in Eq. (17). Specifically, we use convergence error to measure the performance loss:

$$\begin{aligned} \text{(ACP):} \quad & \min_{\{c_t^{(i)}\}} \ \delta(F,T,W,\{c_t^{(i)}\}) \\ & s.t. \ \sum_{t=0}^{T-1} c_t^{(i)} \leq C^{(i)}, \quad i = 1,2,...,W \end{aligned} \tag{17}$$

where $\delta(F,T,W) = F(\mathbf{x}_T) - F(\mathbf{x}^*)$ is the convergence error using SQ-compressor and $\mathbf{x}^*$ is the optimal point to minimize F. Please note that here we only considered the convergence error of the strong convex problem, please refer to Section VI for the discussion for non-convex problems. The original ACP problem is not easy to solve therefore we relax the problem to minimize the *upper bound* of convergence error. According to Lemma 1, the upper bound of the errors when performing SQ compression with allocated bit $c_t^{(i)}$ is $\frac{3d\log_2 d}{2c_t^{(i)}}$. After local gradient uploading, the aggregated stochastic gradient: $\hat{\mathbf{g}}_t \triangleq \frac{1}{W}\sum_{i=1}^{W} \mathcal{C}_{c_t^{(i)}}[\mathbf{g}_t^{(i)}]$ accumulates the errors caused by W clients. Together with Assumption 1, we characterize the property of $\hat{\mathbf{g}}_t$ using the following Lemma.

[1] Note that we can also extend our design to biased compression scenarios, however, the effectiveness may need strong statistical assumptions as well.

Lemma 2 (Unbiasness and Bounded Variance of Aggregated Stochastic Gradient)**.** *For the local gradient* $\mathbf{g}_t^{(i)}$*, given allocated bits is* $c_t^{(i)}$*, after SQ-compression the aggregated gradient* $\hat{\mathbf{g}}_t$ *satisfies:*

$$\mathbb{E}[\hat{\mathbf{g}}_t] = \nabla F(\mathbf{x}_t) \tag{18}$$

and

$$\mathbb{E}\left[||\hat{\mathbf{g}}_t||^2\right] \leq \|\nabla F(\mathbf{x}_t)\|^2 + \underbrace{\frac{\sigma^2}{W}}_{\substack{\textit{Sampling}\\ \textit{Noise}}} + \underbrace{\frac{3d\log_2 d}{2W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}}_{\textit{Compression Noise}}, \tag{19}$$

Eq. (18) means that the aggregated gradient $\hat{\mathbf{g}}_t$ is the unbiased estimate of $\nabla F(\mathbf{x})$. Eq. (19) implies that the gradient noises(i.e., the difference between $||\hat{\mathbf{g}}_t||^2$ and $\|\nabla F(\mathbf{x}_t)\|^2$) consists of two parts: the first part is the sampling noise, which results from the stochastic noise of stochastic gradient; the second part is the compression noise, which is proportional to $\|\mathbf{g}_t^{(i)}\|^2$ and decays with the increase of the number bits $c_t^{(i)}$. In the following section, we consider the worst case of the gradient noise, that is, the gradient noise always reach the upper bound value:

$$\mathbb{E}\left[||\hat{\mathbf{g}}_t||^2\right] = \|\nabla F(\mathbf{x}_t)\|^2 + \frac{\sigma^2}{W} + \frac{3d\log_2 d}{2W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}, \tag{20}$$

Lemma 2 gives the noised added on the aggregated gradient $\hat{\mathbf{g}}_t$ with allocated bits $c_t^{(i)}$. Then considering totally T iterations we can obtain the convergence error bound of Eq. (1) using SQ-compressor with $c_t^{(i)}$, shown in the following Theorem.

Theorem 1 (Convergence Error Bound for Strongly Convex Objectives)**.** *For the problem in Eq.* (1) *under Assumptions 1, 2, 3, with initial parameter* $\mathbf{x}_0$*, using SQ-Compressor to compress gradients in each iteration, we can upper bound the convergence error by*

$$\begin{aligned}&\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)]\\ &\leq \underbrace{\alpha^T[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \frac{L\eta^2\sigma^2[1-\alpha^T]}{2W(1-\alpha)}}_{\text{Error of Distributed SGD}}\\ &+ \underbrace{\frac{3L\eta^2 d\log_2 d}{4W^2}\sum_{t=0}^{T-1}\alpha^{T-1-t}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}}_{\text{Compression Error}},\end{aligned} \tag{21}$$

and lower bound it by

$$\begin{aligned}&\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)]\\ &\geq \underbrace{\beta^T[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \frac{\mu\eta^2\sigma^2(1-\beta^T)}{2W(1-\beta)}}_{\text{Error of Distributed SGD}}\\ &+ \underbrace{\frac{3\mu\eta^2 d\log_2 d}{4W^2}\sum_{t=0}^{T-1}\beta^{T-1-t}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}}_{\text{Compression Error}},\end{aligned} \tag{22}$$

where $\alpha := 1 - 2\mu\eta + L\mu\eta^2$*, and* $\beta := 1 - 2L\eta + L\mu\eta^2$*.*

The full proof is shown in Appendix. D.

We can see that the convergence error consists of two parts: the first two terms are the error of the standard distributed SGD method [34], which can be reduced by increasing the number of iterations T and also depends on the learning rate η (from the expression of α, we can see that when $\eta \leq 1/L$, with the increase of η, α decrease, and the convergence rate of the model is accelerated); The last term is **compression error**, resulted from the lossy compression of gradients, directly increases the convergence error floor. The compression error is the convolution of the compression noise and the weighting function α^{T-1-t} (or β^{T-1-t}). Note that α (or β) is less than 1. Thus we give more weight to recent gradient noises, which means that more recent gradient information is more relevant.

According to Theorem 1, we can rewrite the ACP problem as follows for strongly convex objectives:

$$\begin{aligned}\text{(ACP):}\quad &\min_{\{c_t^{(i)}\}} \sum_{t=0}^{T-1}\alpha^{T-1-t}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}\\ &s.t.\ \sum_{t=0}^{T-1}c_t^{(i)} \leq C^{(i)},\quad for\ i = 1,2,...,W,\end{aligned} \tag{23}$$

By solving the above optimization problem, we can determine the $\{c_t^{(i)}\}$ at every iteration step:

$$\boxed{c_t^{(i)} = r_C^{(i)}\alpha^{(T-1-t)/2}\|\mathbf{g}_t^{(i)}\|} \tag{24}$$

and

$$r_C^{(i)} = \frac{C^{(i)}}{\sqrt{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2\frac{1-\alpha^{T/2}}{1-\alpha^{1/2}}}} \tag{25}$$

Remark 1. *The number of compression bits is determined by three factors: (i) the communication budget* $C^{(i)}$*, more communication budget permits more bits can be allocated for compression; (ii) the iteration step t, the number of bits is increasing as the training process goes on; (iii) the local gradient norms* $\|\mathbf{g}_t^{(i)}\|$*, gradients with a larger norm should be compressed using more bits. (ii) and (iii) may affect how to adjust the number of compression bits with t: the increased weight* $\alpha^{(T-1-t)/2}$ *and the decreased local gradient norm* $\|\mathbf{g}_t^{(i)}\|$*. (*$\alpha^{(T-1-t)/2}$ *increases with the iterations t, and* $\|\mathbf{g}_t^{(i)}\|$ *usually gets smaller as the training process goes on.) Therefore,*

- **Decreasing in Communication.** *If the decreasing rate of the local gradient norm (i.e.,* $\frac{\|\mathbf{g}_{t+1}^{(i)}\|}{\|\mathbf{g}_t^{(i)}\|}$*) smaller than* $\sqrt{\alpha}$*, then* $c_{t+1}^{(i)} < c_t^{(i)}$*, which means the number of compression bits decreases with the iteration step;*
- **Increasing in Communication.** *On the contrary, if the decreasing rate* $\frac{\|\mathbf{g}_{t+1}^{(i)}\|}{\|\mathbf{g}_t^{(i)}\|}$ *is larger than* $\sqrt{\alpha}$*, then* $c_{t+1}^{(i)} > c_t^{(i)}$*, meaning that the number of compression bits increases with the iteration step.*

We summarize the above process as Alg. 1 and Fig. 1. The highlights of this work lie in Line 7-9. The other steps are the same as the standard distributed SGD, thus we do not

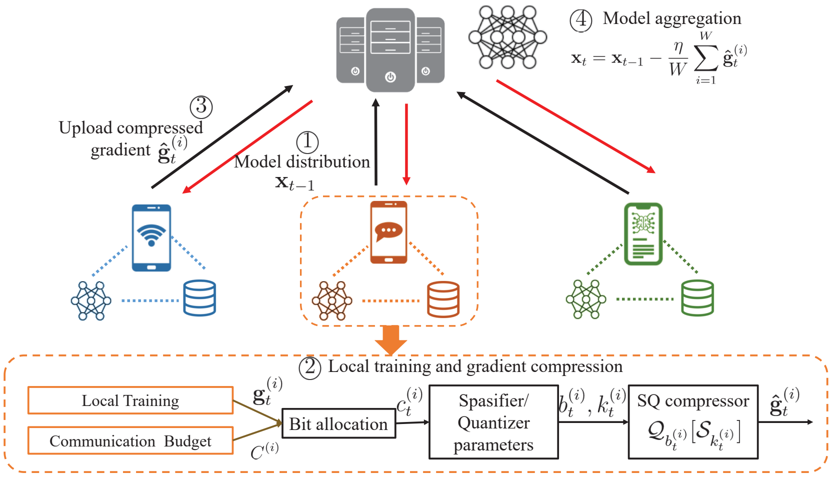

④ Model aggregation
$\mathbf{x}_t = \mathbf{x}_{t-1} - \frac{\eta}{W}\sum_{i=1}^{W}\hat{\mathbf{g}}_t^{(i)}$
③ Upload compressed gradient $\hat{\mathbf{g}}_t^{(i)}$
① Model distribution $\mathbf{x}_{t-1}$
② Local training and gradient compression
Local Training
Communication Budget
$\mathbf{g}_t^{(i)}$
$C^{(i)}$
Bit allocation
$c_t^{(i)}$
Spasifier/ Quantizer parameters
$b_t^{(i)}, k_t^{(i)}$
SQ compressor
$\mathcal{Q}_{b_t^{(i)}}[\mathcal{S}_{k_t^{(i)}}]$
$\hat{\mathbf{g}}_t^{(i)}$

Fig. 1. Adaptively Compressed SGD Framework.

go in detail here. At each iteration t, each client i first take budget $C^{(i)}$ and current gradient size (Line 7), which is the bits hat we need to express the original gradient. Then, taking $c_t^{(i)}$ as input, we can calculate $b_t^{(i)}$ and $k_t^{(i)}$ using Eqs. (14)-(15) (Line 8). Finally, the gradient can be compressed using the SQ-compressor $\mathcal{Q}_{b_t^{(i)}}[S_{k_t^{(i)}}(\mathbf{g}_t^{(i)})]$ (Line 9).

Algorithm 1 AC-SGD in Distributed Learning

```
1:  Input: Iterations number $T$, communication budget $C^{(i)}$,
    learning rate $\eta$, initial point $\mathbf{x}_0 \in \mathbb{R}^d$;
2:  Output: $\mathbf{x}_T$
3:  for each iteration $t = 1, ..., T-1$: do
4:      On each $i = 1, ..., W$:
5:      Receive $\mathbf{x}_t$ from server;
6:      Compute the local gradient $\mathbf{g}_t^{(i)}$ using SGD;
7:      Adaptively adjust the compression level
            $c_t^{(i)} = CompressLevel(C^{(i)}, \mathbf{g}_t^{(i)})$
8:      Determine the parameters for SQ Compressor
            $b_t^{(i)}, k_t^{(i)} = SQ(c_t^{(i)})$
9:      Use SQ Compressor to generate
            $\hat{\mathbf{g}}_t^{(i)} = \mathcal{Q}_{b_t^{(i)}}[S_{k_t^{(i)}}(\mathbf{g}_t^{(i)})]$
10:     Send $\hat{\mathbf{g}}_t^{(i)}$ to server;
11:     On server:
12:     Aggregates all $W$ gradients $\hat{\mathbf{g}}_t^{(i)}$ from s and updates
    the model parameter: $\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{\eta}{W}\sum_{i=1}^{W}\hat{\mathbf{g}}_t^{(i)}$ ;
13:     Send $\mathbf{x}_{t+1}$ to all clients;
14: end for
```

C. Algorithm Implementation Details

Although Eq. (24) provide valuable insights about how to adjust $c_t^{(i)}$ over time, it is still challenging to use it in practice due to the convergence rate α being known. Inspired by [35], we propose a straightforward rule where we approximate $F(\mathbf{x}^*)$ to 0 and the learning rate η is small enough. We estimate α as follows according to Theorem 1:

$$\alpha_{est} = \left[\frac{F(\mathbf{x}_t)}{F(\mathbf{x}_0)}\right]^{1/t} \tag{26}$$

Then, $\sqrt{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2}$ actually is the upper bound of $\|\mathbf{g}_0^{(i)}\|$, and we obtain a heuristic estimate of it by a simply repeat the calculation of $\|\mathbf{g}_0^{(i)}\|$. So, we obtain the number of compression bits update rule:

$$c_t^{(i)} = \frac{C^{(i)}}{\|\mathbf{g}_0^{(i)}\|_{re}\frac{1-\alpha_{est}^{T/2}}{1-\alpha_{est}^{1/2}}}\alpha_{est}^{(T-1-t)/2}\|\mathbf{g}_t^{(i)}\| \tag{27}$$

where $\|\mathbf{g}_0^{(i)}\|_{re}$ is the average value of several tries. $F(\mathbf{x}_0)$, $F(\mathbf{x}_t)$ and $\|\mathbf{g}_t^{(i)}\|$ can be easily obtained in the training.

V. Performance Analysis

In last section, we propose Alg.1 to dynamicly determine the allocated bits and parameters for the compressor. In this section, we give some theoretical performance analysis on this compression scheme. The following Theorems can be achieved by substituting specifics $c_t^{(i)}$ to Theorem 1.

As all compression operations will incur additional noise on the convergence error bound compared to non-compressed scheme due to the information loss, we first define the convergence error for the standard SGD without any compression as:

$$\delta_{DSGD} = \alpha^T[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \frac{L\eta^2\sigma^2[1-\alpha^T]}{2W(1-\alpha)} \tag{28}$$

We compare the additional error on δ_{DSGD} brought by the following two compression schemes: 1) the Adaptive SQ-Compressor proposed in this work by using the dynamic SQ-Compressor in Eq. (24) to compress the gradients; 2) fixed-bit SQ-Compressor with $c_t^{(i)} = \frac{C^{(i)}}{T}$ for all t. For simplicity, we let all clients own the same budget, i.e., $C^{(i)} = C$.

Corollary 1. *For the problem in Eq.* (1) *under Assumptions 1, 2, 3, with initial parameter* $\mathbf{x}_0$*, the upper bound of the convergence error incurred by Adaptive SQ-Compressor is:*

$$\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] \leq \delta_{DSGD} + R(C)QM_{\alpha^{t/2}}, \qquad (29)$$

Corollary 2. *For the problem in Eq.* (1) *under Assumptions 1, 2, 3, with initial parameter* $\mathbf{x}_0$*, the upper bound of the convergence error incurred by the fixed SQ-Compressor is:*

$$\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] \leq \delta_{DSGD} + R(C)AM_{\alpha^{t/2}}, \qquad (30)$$

where

$$R(C) = \frac{3LT^2\eta^2 d \log_2 d\{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2\}}{4WC} \qquad (31)$$

is the same for two kinds of compressors. $R(C) \propto \frac{1}{C}$, which means that given more communication budget, a smaller convergence error could be obtained.

The only difference lies in that the additional error term of Adaptive SQ-Compressor depends on the Arithmetic Mean $AM_{\alpha^{t/2}} = \frac{1}{T}\sum_{t=0}^{T-1}\alpha^{t/2}$, while the fixed SQ-Compressor ends up with the Quadratic Mean $QM_{\alpha^{t/2}} = \sqrt{\frac{1}{T}\sum_{t=0}^{T-1}\alpha^t}$. Note that $0 < \alpha < 1$, so $AM(\alpha) > QM(\alpha)$, which means our proposed AC-SGD can achieve lower convergence error compared with the fixed-bit algorithms.

VI. Discussions

A. AC-SGD for Non-Convex Objectives

In the previous sections, we assumed that the objected functions are strongly convex. However, deep learning models (e.g., deep neural networks, recurrent neural networks and convolutional neural networks) are usually non-convex. So in this subsection, we will extend our algorithm to non-convex objectives. Notice that the design of b, k for SQ compressor is solved by minimizing the noise $h(k, b)$ (Eq. 13) given the allocate bits c. The objective function strongly convex or non-convex does not affect the results. In contrast to this, the bit allocation is determined by solving the ACP problem by minimizing the upper bound of convergence error $\delta(F, T, W)$. For the non-convex objectives,

$$\delta(F, T, W) = \frac{\sum_{t=0}^{T-1}\gamma_t\|\nabla F(\mathbf{x}_t)\|^2}{\sum_{t=0}^{T-1}\gamma_t} \qquad (32)$$

where γ_t is the weight and satisfy $0 < \gamma_0 \leq ... \leq \gamma_{t-1} \leq \gamma_t ... \leq \gamma_{T-1} \leq 1$. Analyzing the compressed SGD for non-convex objectives is more challenging than in the strongly-convex case since such functions may possess multiple local minimums. Previous works usually use an ordinary mean $\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\mathbf{x}_t)\|^2$ to character the non-convex convergence error [8], [14], [15], and when $\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(\mathbf{x}_t)\|^2 \to 0$, this condition can guarantee the algorithm converges to a stationary point. In this work, we improve it to a weighted mean $\frac{\sum_{t=0}^{T-1}\gamma_t\|\nabla F(\mathbf{x}_t)\|^2}{\sum_{t=0}^{T-1}\gamma_t}$. Note that γ_t is gradually increased with the training process, that is to say, we pay more attention to the gradient value at the later stage of training process.

Similar as Theorem 1, we first give the convergence error bound of non-convex objectives.

Theorem 2 (Convergence Error Bound of Non-Convex Objectives)**.** *For the problem in Eq.* (1) *under Assumptions 1, 2, with initial parameter* $\mathbf{x}_0$*, using compressed gradients in Eq.* (3) *for each iteration, we can upper bound the convergence error by*

$$\begin{aligned}
&\frac{1}{\sum_{t=0}^{T-1}\gamma_t}\sum_{t=0}^{T-1}\gamma_t\mathbb{E}[\|\nabla F(\mathbf{x}_t)\|^2] \\
&\leq \underbrace{\frac{2}{(2\eta - L\eta^2)\sum_{t=0}^{T-1}\gamma_t}[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \frac{L\eta\sigma^2}{(2-L\eta)W}}_{\text{Error of Distributed SGD}} \\
&\quad + \underbrace{\frac{3dL\eta\log_2 d}{(4W^2 - 2W^2L\eta)\sum_{t=0}^{T-1}\gamma_t}\sum_{t=0}^{T-1}\gamma_t\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}}_{\text{Compression Error}}
\end{aligned} \qquad (33)$$

Hence, we then rewrite the ACP problem as follows:

$$\begin{aligned}
\text{(ACP)}: \quad &\min_{\{c_t^{(i)}\}} \sum_{i=1}^{W}\sum_{t=0}^{T-1}\gamma_t\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \\
&s.t. \quad \sum_{t=0}^{T-1}c_t^{(i)} \leq C^{(i)}, \quad for\ i = 1, 2, ..., W,
\end{aligned} \qquad (34)$$

By solving the above optimization problem, we can determine the $\{c_t^{(i)}\}$ at every iteration step:

$$\boxed{c_t^{(i)} = r^{(i)}\sqrt{\gamma_t}\|\mathbf{g}_t^{(i)}\|} \qquad (35)$$

If we take γ_t as exponential growth weight, i.e. $\gamma_t = \alpha^{T-1-t}$, then

$$r^{(i)} = \frac{C^{(i)}}{\sqrt{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2}\frac{1-\alpha^{T/2}}{1-\alpha^{1/2}}} \qquad (36)$$

Eq. (35) will degrade to Eq. (24).

B. Convergence Error for Quadratic Objectives.

In previous sections, we upper bound and lower bound the convergence error of general strongly-convex objects. In this subsection, we focus on a special strongly-convex objects – quadratic functions, and give its exact convergence error.

For general quadratic functions, we can employ gradient flow [2] to calculate an exact convergence error. We have the relationship between the aggregated stochastic gradients and full gradients: $\hat{\mathbf{g}}_t = \nabla F(\mathbf{x}_t) + \boldsymbol{\epsilon}_t$. Based on the central limit theorem, it is assumed that $\boldsymbol{\epsilon}_t$ follows the Gaussian distribution, that is $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\mathbf{x}_t))$. Then using analysis within the gradient flow framework, we can get the following theorem.

[2] when the learning rate is infinitesimal, the stochastic gradient descent process can be regarded as a stochastic dynamic system.

Theorem 3 (Exact Convergence Error for Quadratic Objectives)**.** *For a quadratic optimization objective function* $F(\mathbf{x}) = 1/2\mathbf{x}^{\mathrm{T}}\mathbf{H}\mathbf{x}+\mathbf{A}^{\mathrm{T}}\mathbf{x}+B$*, consider the perturbed gradient descent dynamics*

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\nabla F(\mathbf{x}_t) - \eta\boldsymbol{\epsilon}_t, \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\mathbf{x}_t)) \tag{37}$$

We can achieve

$$\begin{aligned} &\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] \\ &= \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^*)^{'}\left(\boldsymbol{\rho}(\eta)^T\right)^{'}\mathbf{H}\boldsymbol{\rho}(\eta)^T(\mathbf{x}_0 - \mathbf{x}^*) \\ &+ \frac{\eta^2}{2}\sum_{t=0}^{T-1}\mathrm{Tr}\left[\boldsymbol{\rho}(\eta)^{T-1-t}\boldsymbol{\Sigma}(\mathbf{x}_t)\mathbf{H}\left(\boldsymbol{\rho}(\eta)^{T-1-t}\right)^{\mathrm{T}}\right] \end{aligned} \tag{38}$$

where $\boldsymbol{\rho}(\eta) := \mathbf{I} - \eta\mathbf{H}$*, and* $\mathbf{H}$ *is the Hessian matrix.*

Detailed proof is in Appendix H. We can see that the convergence error consists of two parts: the error of the gradient descent method, which is linearly convergent; the error due to gradient estimation error (sampling noise, compression noise).

Consider the case where the Hessian matrix is isotropic $\mathbf{H} = \lambda\mathbf{I}$, and let $\nu := 1 - 2\eta\lambda + \eta^2\lambda^2$, then Eq.(38) can be rewrite as

$$\begin{aligned} \mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] &= \nu^T[F(\mathbf{x}_0) - F(\mathbf{x}^*)] \\ &+ \frac{\lambda\eta^2}{2}\sum_{t=0}^{T-1}\beta(\eta)^{T-1-t}\mathrm{Tr}[\boldsymbol{\Sigma}(\mathbf{x}_t)] \end{aligned} \tag{39}$$

According to Eq. (20), we can get

$$\begin{aligned} \mathrm{Tr}[\boldsymbol{\Sigma}(\mathbf{x}_t)] &= \mathbb{E}\left[\|\hat{\mathbf{g}}_t - \nabla F(\mathbf{x}_t)\|^2\right] \\ &= \frac{\sigma^2}{W} + \frac{3d\log_2 d}{2W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \end{aligned} \tag{40}$$

Plugging Eq. (40) into Eq. (39), then we can get the same results as Theorem 1.

C. Heterogeneous Communication Resources

In Corollary 1, we assume that all clients own the same budget, i.e., $C^{(i)} = C$. In general, if each client is given a communication constraint $C^{(i)}$, we can apply Alg. 1 to solve the adaptive compression problem. In addition, if a total communication constraint C_{total} is given, each client may have a different communication budget $C^{(i)}$. Then, the convergence error of the AC-SGD is:

$$\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] \leq \delta_{DSGD} + E\sum_{i=0}^{W}\frac{1}{C^{(i)}}, \tag{41}$$

where

$$E = \frac{3LT^2\eta^2 d\log_2 d\{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2\}\sqrt{\sum_{t=0}^{T-1}\alpha^t/T}}{4W^2}.$$

Given the total communication budget C_{total}, i.e., $\sum_{i=0}^{W}C^{(i)} = C_{total}$, we try to learn the optimal communication budget allocation strategy for each client by minimizing the convergence error in Eq. (41). This problem can be formulated as:

$$\begin{aligned} &\min_{\{C^{(i)}\}} \sum_{i=0}^{W}\frac{1}{C^{(i)}} \\ &s.t. \sum_{i=0}^{W}C^{(i)} = C_{total}, \quad i = 1, 2, ..., W \end{aligned} \tag{42}$$

By solving the above optimal equation, we can obtain $C^{(i)} = \frac{C_{total}}{W}$, indicating that the optimal communication budget allocation strategy is a **uniform allocation**.

D. Further Improvement of Communication in AC-SGD

In our current design, each client transmits the positional information of sparsifier to notify the server. To further reduce this communication burden, a pseudo random generator can be used in the parameter server to know the positions of the non-zero values, however, with some extra computation cost in both the server and each client. Then the the optimal design of the compressor (Eq. (13)) can be found by minimizing $h(k, b)$ with the new bits constrain c:

$$\begin{aligned} &\min_{b,k} \quad \frac{d-k}{k} + \frac{d}{4^b}, \\ &s.t. \quad kb + B_{pre} = c. \end{aligned} \tag{43}$$

By solving the above optimization problem, we can redetermine the quantization/sparsification level for the new SQ compressor:

$$b^* = \frac{1}{2}\log_2\left[2\ln 2 * (c - B_{pre})\right], \tag{44}$$

$$k^* = \frac{c - B_{pre}}{\frac{1}{2}\log_2\left[2\ln 2 * (c - B_{pre})\right]}. \tag{45}$$

Compared with Eqs. (14) and (15), the new SQ compressor uses the same quantization bits b^* and a larger sparsity size k^* (whereas the denominator no longer contains $\log_2 d$) to compress the gradient.

VII. Experiments

In this section, we conduct experiments on CV and NLP tasks on three datasets: MNIST, CIFAR-10 [36], CIFAR-100 and AG-News [37], to validate the effectiveness of our proposed AC-SGD methods. We compare our proposed AC-SGD with the following baselines: 1) **Static compression**: QSGD [4] is a fixed-bit quantizer using Element-Wise Uniform quantization. Rand-k [5], [18] and Top-k sparsifier [5] are fixed-level sparsifiers. The former randomly drops out some elements and amplifies the remaining elements appropriately, while the latter retains largest k elements of the gradient and sets the rest elements to 0. 2) **Adaptive compression**: DQSGD [1] adaptively adjusts the quantization bits by taking into account the remaining number of iterations and the norm of gradients. ACCORDION [13] determines the compression level according to gradient's norm. 3) **Hybrid compression**: Qsparse SGD [8] uses a combination of quantization and sparsification at each round. 4) **No compression**: we use the full-precision floating-point $B_{pre} = 32$ for the standard SGD. Note that our proposed adaptive compression method is based on sparsification and quantization and is in parallel with other accelerating methods, such as momentum and error compensation.

Experimental Setting. We use logistic regression and canonical neural networks to evaluate the performance of different algorithms: logistic regression for the binary classification on MNIST [3], ResNet18 [38] for the image classification task on CIFAR-10, and fastText [39] for the text classification task on AG-News. We let all clients own the same budget, i.e., $C^{(i)} = C$, and the communication cost is the total budget of W clients (i.e., $C_{total} = WC$). We utilize the error compensation mechanism follow [26], and set $\beta = 0.98$ and $\alpha = 0.01$. We select the momentum SGD as an optimizer, where the momentum is set to 0.9, and weight decay is set to 0.0005. Other experimental settings are given in Table II.

Training performance Figures 2 and 3 show the learning loss of different algorithms on MNIST and CIFAR-10. For MNIST, the SGD can achieve a test accuracy of 0.9870 and incur the communication cost of 153KB. Then given the communication budget 9.6 KB, the QSGD with $b = 2$ fixed quantization bit, the Rand-k with $k = 0.048d$ fixed sparsification size, and our proposed AC-SGD schemes can achieve the test accuracy of 0.9742, 0.9746, and 0.9868, respectively. Our proposed AC-SGD algorithm outperforms the fixed quantization and fixed sparsification by 0.0126 and 0.0122 and only decreased by 0.0002 compared with SGD in test accuracy. Similarly, for CIFAR10, the SGD can achieve a test accuracy of 0.9031 and incur the communication cost of 1998GB. Then given the communication budget 188 GB, the QSGD with $b = 3$ fixed quantization bit, the Rand-k with $k = 0.065d$ fixed sparsification size, and our proposed AC-SGD schemes can achieve the test accuracy of 0.8048, 0.8742, and 0.8917, respectively. Our proposed AC-SGD algorithm outperforms the fixed quantization and fixed sparsification by 0.0822 and 0.0127 and only decreased by 0.0160 compared with SGD in test accuracy.

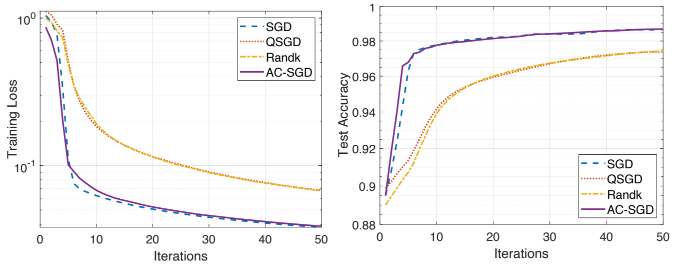


Fig. 2. Model Performance on MNIST Dataset. (QSGD with quantization bit = 2, Rand-k with $k = 0.048d$)

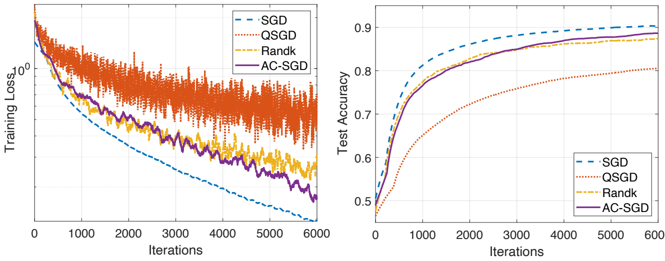
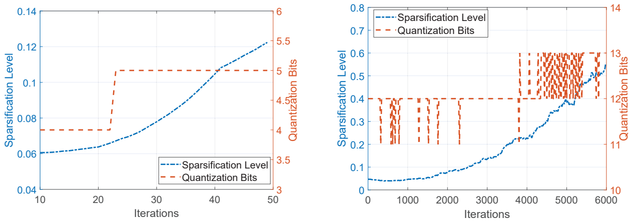


Fig. 3. Model Performance on CIFAR-10 Dataset.(QSGD with quantization bit = 3, Rand-k with $k = 0.065d$)



(a) Compression level on MNIST Dataset (b) Compression level on CIFAR-10 Dataset

Fig. 4. Compression level with iterations. (The sparsification level is $\frac{k_t}{d}$.)

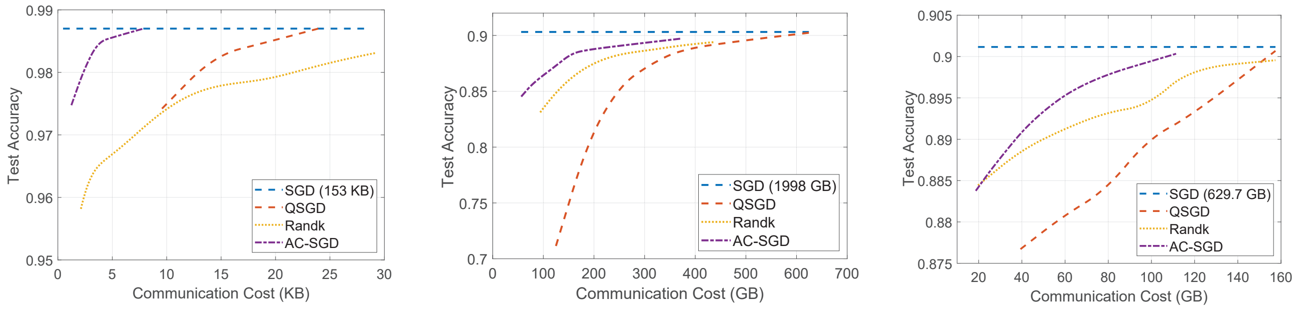
Adaptively determine the compression level. Figures 4(a) and 4(b) show the compression level of each iteration of AC-SGD. We can see that AC-SGD significantly reduces the bits assigned at the early stage of training and improves the gradient accuracy as the training goes on. The main reason is that the gradient noise in the later stage of training has a greater impact on the convergence error. We need to reduce the variance of gradient noise to ensure better convergence of the algorithm (see Remark 1 for details). This result is similar to some heuristic work [10], [11]. Specifically, the number of quantization bits increased from 4 to 5, and the sparsification level increased from 0.06 to 0.12 on MNIST. For CIFAR-10, the quantization bits increased from 11 to 13, and the sparsification level increased from 0.1 to 0.5 (The reason for the growth fluctuation of compression level is that the stochastic gradient norm fluctuates with the iteration step.). We can see that the compression level (quantization bits and sparsification level) of CIFAR-10 are larger than that of MNIST. This is because the data and model of CIFAR-10 are more complicated than that of MNIST. Higher accurate gradients are needed to guarantee the algorithm's convergence.

Testing Performance. In Table III and Table IV, we compare the test accuracy of our proposed AC-SGD with some selected algorithms on CIFAR-10 and CIFAR-100 with and without error compensation. The standard SGD without communication constraints provides a benchmark of the testing performance. For CIFAR-10, we set the same communication budget 188GB for all communication constrained algorithms, 9.5% of the communication cost incurred by the SGD. Specifically, we set fixed quantization bit $b = 3$ for QSGD, fixed sparsification size $k = 0.065d$ for Top-k and Rand-k, fixed $b = 4$ and $k = 0.67d$ for the hybrid compression case Qsparse. Note that DQSGD and ACCORDION do not take communication budget into account when adaptively calculating the compressed level. We try different sets of parameters, and shown the results when their communication costs are closest to the given budget. We can see that our proposed AC-SGD achieve the highest testing accuracy whether using error compensation or not. Compared to standard SGD, we reduced

[3] The task is to classify a given image is as number '0' or not '0'.

TABLE II
EXPERIMENT SETTING

Dataset	MINIST	CIFAR-10	AG-News	CIFAR-100
Networks	Logistic Regression	ResNet18	fastText	ResNet34
Model size	$d = 785$	$d = 1 \times 10^7$	$d = 4 \times 10^7$	$d = 3 \times 10^7$
Learning rate	1	0.01	0.001	0.01
Batch size	/	32	32	32
Workers	1	8	8	16
Iterations	50	6000	500	6000



(a) MNIST

(b) CIFAR10

(c) Agnews

Fig. 5. Communication-learning tradeoffs on different datasets.

TABLE III
TESTING ACCURACY ON CIFAR-10 DATASET (QSGD WITH $b = 3$, RAND-k AND TOP-k WITH $k = 0.065d$, QSPARSE SGD WITH $b = 4$ AND $k = 0.67d$.)

Algorithm	w/o error compensation	with error compensation
SGD	0.9031	0.9031
QSGD [4]	0.8048	0.8544
Rand-k [5], [18]	0.8742	0.8921
Top-k [5]	0.8595	0.8843
DQSGD [1]	0.8192	0.8695
ACCORDION [13]	0.8639	0.8911
Qsparse SGD [8]	0.8504	0.8799
AC-SGD (ours)	**0.8917**	**0.9008**

TABLE IV
TOP-5 TESTING ACCURACY ON CIFAR-100 DATASET (QSGD WITH $b = 3$, RAND-k AND TOP-k WITH $k = 0.065d$, QSPARSE SGD WITH $b = 4$ AND $k = 0.67d$.)

Algorithm	w/o error compensation	with error compensation
SGD	0.8807	0.8807
QSGD [3]	0.8214	0.8501
Rand-k [4]	0.8518	0.8701
Top-k [4]	0.8378	0.8633
DQSGD [24]	0.8315	0.8577
ACCORDION [12]	0.8491	0.8693
Qsparse SGD [7]	0.8639	0.8711
AC-SGD (ours)	**0.8701**	**0.8798**

the cost of communication by 90%, but only incur a performance impairment of 0.0023 with error compensation. The performance improvement brought by error compensation to other algorithms ranges from 2% to 6%. But for AC-SGD,This shows that the performance of AC-SGD is very close to the standard SGD it brought less than 1% improvement from 0.8917 to 0.9008. This shows that the performance of AC-SGD is very close to the standard SGD. Furthermore, we conduct the experiment of using a pseudo random generator for further communication reduction (in Section VI-D) and it achieves an accuracy of 0.8970 without error compensation, given the same communication budget. For CIFAR-100, we also set the 9.5% of the communication cost incurred by the SGD as communication budget for all communication constrained algorithms, then we can get the similar conclusion as CIFAR-10.

Moreover, we give the experimental results of running time in Table V. Except for the standard SGD, i.e., the full-communication case, we give the same communication budget for all other compression methods. We define the running time as: Total Time = (Compute Gradient + Compress Gradient) $*$ Num of Users + Other processing time. The time required for gradient computing is the same for all methods. Note that the rand-k sparsifier takes much less amount of time for compression since it only needs the drop-out operation. Except for rand-k sparsifier, there is no significant difference among other methods. However, our method has the highest accuracy.

Communication-Learning Tradeoff. Figure 5 shows the tradeoff between communication budget (cost) and the learning performance in terms of the test accuracy on different datasets. We compare this tradeoff between our proposed algorithm and two other baselines - fixed bit sparsification and fixed bit quantization. We also list the accuracy achieved by the SGD without communication budget constraints as a benchmark.

In Figure 5, all three algorithms show a communication-learning tradeoff, that is, the more communication budget can be used, the higher test accuracy can be achieved. However, our proposed AC-SGD can achieve a higher test accuracy than the other two under the same communication cost. The marginal utility (how much test accuracy is improved from

TABLE V
Running time of different algorithms.

	Compute Gradient (s)	Compress Gradient (s)	Other (s)	Total Time (s)
SGD	113	0	1201	2105
QSGD	113	490	1201	6025
$Rand_k$ sparsifier	113	2	1201	2121
Top_k sparsifier	113	512	1201	6201
DQSGD	113	491	1201	6033
ACCORDION	113	542	1201	6441
Qsparse SGD	113	492	1201	6041
AC-SGD (Ours)	113	493	1201	6049

the increased communication budget) is diminishing. That means when the communication budget is small, increasing the communication budget can bring significant improvement. When the communication budget is large (for example, $C > 200$ GB in CIFAR-10), the improvement of the test accuracy by increasing the communication budget is limited. This phenomenon also verifies the results of Corollary 1.

Heterogeneous Communication Budget. In addition, we investigate the case of heterogeneous communication budgets across workers. We set $C_{total} = \sum_{i=1}^{W} C^{(i)}$ as $188GB$, and consider 4 cases of communication budget distribution for a 8-worker setting in Table VI. Case 1 is the uniform allocation while Case 4 is extremely imbalance, where the budget of $C^{(7)}, C^{(8)}$ is 10 times more than that of $C^{(1)}, C^{(2)}$. We can see that the test accuracy of the above 4 cases are 0.8917, 0.8884, 0.8794, and 0.8670 respectively. From case 1 to case 4, the accuracy decreases as the degree of imbalance of the communication budget allocation (which can be evaluated, e.g., using the entropy of the budget distribution) increases. This is because when some workers have scarce communication resources, a significant gradient compression brings a significant information loss of the local gradient, leading to a poor aggregated gradient and hence resulting in a slower convergence speed.

TABLE VI
Communication budget allocation schemes.

	Communication Budget (GB)	Test Acc
case1	All workers: 23.5	0.8917
case2	$\{C^{(1)}, C^{(2)}, C^{(3)}, C^{(4)}\}$: 15.67 $\{C^{(5)}, C^{(6)}, C^{(7)}, C^{(8)}\}$: 31.33	0.8884
case3	$\{C^{(1)}, C^{(2)}\}$: 7.83 $\{C^{(3)}, C^{(4)}\}$:15.67 $\{C^{(5)}, C^{(6)}\}$:31.33 $\{C^{(7)}, C^{(8)}\}$:39.17	0.8794
case4	$\{C^{(1)}, C^{(2)}\}$: 3.92 GB $\{C^{(3)}, C^{(4)}\}$:15.67 GB $\{C^{(5)}, C^{(6)}\}$:31.33 GB $\{C^{(7)}, C^{(8)}\}$:43.08 GB	0.8670

Periodical AC-SGD. To further improve the communication efficiency of our algorithm, we propose the variant of AC-SGD, the periodical AC-SGD. Specifically, each client communicate with the server after performing τ times local SGD, therefore, the total number of communication is reduced to T/τ.

$$\mathbf{x}_t^{(i)} = \begin{cases} \mathbf{x}_{t-\tau}^{(i)} + \frac{1}{W}\sum_{j=1}^{W} \mathcal{C}_{c_t^{(i)}}[\mathbf{x}_t^{(j)} - \mathbf{x}_{t-\tau}^{(j)}], & \text{for } \tau | t \\ \mathbf{x}_{t-1}^{(i)} - \eta \mathbf{g}_{t-1}^{(i)}, & \text{otherwise} \end{cases} \tag{46}$$

Since more local iterations require fewer communication cost, we reduce the communications budget C^{τ} by a factor of τ. The experimental results of periodical-averaging AC-SGD on CIFAR-10 dataset with 8 workers are shown in Table VII. Compared to the original AC-SGD accuracy of 0.8917, the test accuracy decreases with increasing local iterations from $\tau = 10$ to 40. This fits the trade-off relationship between communication cost and learning performance.

TABLE VII
Test Accuracy of periodical AC-SGD on CIFAR-10 Dataset.

τ	10	20	30	40
Test Accuracy	0.8914	0.8909	0.8881	0.8778

The Number of Clients. We vary the number of workers from 4 to 32 and show the test accuracy of CIFAR-10 dataset in Table VIII. From Table VIII, we can see that using more workers can improve the model's performance. Specifically, when we set the number of workers as 16 and 32, the test accuracy of AC-SGD can achieve 0.9046 and 0.9094, which even out-performance the non-compressed SGD with 8 workers (0.9031). This also verifies the results of Theorem 1, which shows that increasing the number of workers can reduce the SGD convergence error and the compression error simultaneously.

TABLE VIII
Test Accuracy of AC-SGD on CIFAR-10 with Different Clients.

Number of Workers	4	8	16	32
Test Accuracy	0.8775	0.8971	0.9046	0.9094

VIII. Conclusion

This paper develops a unifying framework for dynamic gradient descent that collaboratively leverages both quantization and sparsification techniques. We then consider communication budget constraints and propose an optimization formulation - denoted as the Adaptive Compression Problem (ACP)- to minimize the deep model's convergence error under such constraints. By solving the ACP, we propose a novel Adaptively-Compressed SGD (AC-SGD) strategy to jointly adjust the number of quantization bits and the sparsification size concerning the norm of gradients, the communication budget, and the remaining number of iterations. The experimental results of image classification and text classification show that AC-SGD is superior to state-of-the-art gradient compression methods in improving the model's performance.

IX. Acknowledgment

Prof. Shao-Lun Huang is supported in part by the Shenzhen Science and Technology Program under Grant KQTD20170810150821146, National Key R&D Program of China under Grant 2021YFA0715202 and High-end Foreign Expert Talent Introduction Plan under Grant G2021032013L. Prof. Linqi Song was supported in part by the Hong Kong RGC grant ECS 21212419, the Technological Breakthrough Project of Science, Technology and Innovation Commission of Shenzhen Municipality under Grants JSGG20201102162000001, InnoHK initiative, the Government of the HKSAR, Laboratory for AI-Powered Financial Technologies, the City University of Hong Kong SRG grants 7005660, 7005561, and the Tencent AI Lab Rhino-Bird Gift Fund.

References

[1] G. Yan, S.-L. Huang, T. Lan, and L. Song, "Dq-sgd: Dynamic quantization in sgd for communication-efficient distributed learning," in *2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*. IEEE, 2021, pp. 136–144.

[2] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang *et al.*, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1223–1231.

[3] R. Bekkerman, M. Bilenko, and J. Langford, *Scaling up machine learning: Parallel and distributed approaches*. Cambridge University Press, 2011.

[4] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, 2017, pp. 1709–1720.

[5] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Advances in Neural Information Processing Systems*, 2018, pp. 4447–4458.

[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.

[7] S. U. Stich, "Local sgd converges fast and communicates little," in *International Conference on Learning Representations*, 2018.

[8] D. Basu, D. Data, C. Karakus, and S. N. Diggavi, "Qsparse-local-sgd: Distributed sgd with quantization, sparsification, and local computations," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 217–226, 2020.

[9] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," in *International Conference on Learning Representations*, 2018.

[10] J. Guo, W. Liu, W. Wang, J. Han, R. Li, Y. Lu, and S. Hu, "Accelerating distributed deep learning by adaptive gradient quantization," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 1603–1607.

[11] G. Cui, J. Xu, W. Zeng, Y. Lan, J. Guo, and X. Cheng, "MQGrad: Reinforcement learning of gradient quantization in parameter server," in *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, 2018, pp. 83–90.

[12] A. Oland and B. Raj, "Reducing communication overhead in distributed learning by an order of magnitude (almost)," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 2219–2223.

[13] S. Agarwal, H. Wang, K. Lee, S. Venkataraman, and D. Papailiopoulos, "Adaptive gradient communication via critical learning regime identification," *Proceedings of Machine Learning and Systems*, vol. 3, pp. 55–80, 2021.

[14] J. Xu, S.-L. Huang, L. Song, and T. Lan, "Live gradient compensation for evading stragglers in distributed learning," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[15] D. P. Bertsekas *et al.*, "Incremental gradient, subgradient, and proximal methods for convex optimization: A survey," *Optimization for Machine Learning*, vol. 2010, no. 1-38, p. 3, 2011.

[16] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, "1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs." *Conference of the International Speech Communication Association*, pp. 1058–1062, 2014.

[17] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li, "TernGrad: Ternary gradients to reduce communication in distributed deep learning," *Advances in Neural Information Processing Systems*, pp. 1508–1518, 2017.

[18] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," *Advances in Neural Information Processing Systems*, vol. 31, pp. 1299–1309, 2018.

[19] A. T. Suresh, F. X. Yu, S. Kumar, and H. B. Mcmahan, "Distributed mean estimation with limited communication," *International Conference on Machine Learning*, pp. 3329–3337, 2017.

[20] E. Ozfatura, K. Ozfatura, and D. Gündüz, "Time-correlated sparsification for communication-efficient federated learning," in *2021 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2021, pp. 461–466.

[21] T. Vogels, S. P. Karimireddy, and M. Jaggi, "Powersgd: Practical low-rank gradient compression for distributed optimization," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[22] C.-Y. Chen, J. Choi, D. Brand, A. Agrawal, W. Zhang, and K. Gopalakrishnan, "Adacomp: Adaptive residual gradient compression for data-parallel distributed training," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[23] Z. Zhang and C.-L. Wang, "Mipd: An adaptive gradient sparsification framework for distributed dnns training," *IEEE Transactions on Parallel and Distributed Systems*, 2022.

[24] F. Faghri, I. Tabrizian, I. Markov, D. Alistarh, D. M. Roy, and A. Ramezani-Kebrya, "Adaptive gradient quantization for data-parallel sgd," *Advances in neural information processing systems*, vol. 33, pp. 3174–3185, 2020.

[25] K. Mishchenko, B. Wang, D. Kovalev, and P. Richtárik, "Intsgd: Adaptive floatless compression of stochastic gradients," in *International Conference on Learning Representations*, 2021.

[26] J. Wu, W. Huang, J. Huang, and T. Zhang, "Error compensated quantized SGD and its applications to large-scale distributed optimization," *arXiv preprint arXiv:1806.08054*, 2018.

[27] C.-Y. Chen, J. Ni, S. Lu, X. Cui, P.-Y. Chen, X. Sun, N. Wang, S. Venkataramani, V. V. Srinivasan, W. Zhang *et al.*, "Scalecom: Scalable sparsified gradient compression for communication-efficient distributed training," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13 551–13 563, 2020.

[28] C. Xie, S. Zheng, S. Koyejo, I. Gupta, M. Li, and H. Lin, "Cser: Communication-efficient sgd with error reset," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 593–12 603, 2020.

[29] J. Konečnỳ, B. McMahan, and D. Ramage, "Federated optimization: Distributed optimization beyond the datacenter," *arXiv preprint arXiv:1511.03575*, 2015.

[30] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *Siam Review*, vol. 60, no. 2, pp. 223–311, 2018.

[31] H. Tang, C. Yu, X. Lian, T. Zhang, and J. Liu, "Doublesqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6155–6165.

[32] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[33] A. Beznosikov, S. Horváth, P. Richtárik, and M. Safaryan, "On biased compression for distributed learning," *arXiv preprint arXiv:2002.12410*, 2020.

[34] M. P. Friedlander and M. Schmidt, "Hybrid deterministic-stochastic methods for data fitting," *SIAM Journal on Scientific Computing*, vol. 34, no. 3, pp. A1380–A1405, 2012.

[35] J. Wang and G. Joshi, "Adaptive communication strategies to achieve the best error-runtime trade-off in local-update sgd," *arXiv preprint arXiv:1810.08313*, 2018.

[36] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," *University of Toronto*, 2009.

[37] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in Neural Information Processing Systems*, 2015, pp. 649–657.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[39] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "Fasttext. zip: Compressing text classification models," *arXiv preprint arXiv:1612.03651*, 2016.

Appendix

A. Proof of Lemma 1

In this EWU scheme, The j-th component of the stochastic gradient vector $\mathbf{g}$ (for any worker i) is quantized as

$$\mathcal{Q}_b[g_j] = \|\mathbf{g}\| \cdot \mathrm{sgn}(g_j) \cdot \zeta(g_j, s), \tag{47}$$

where $\|\mathbf{g}\|$ is the l_2 norm of $\mathbf{g}$; $\mathrm{sgn}(g_j) = \{+1, -1\}$ is the sign of g_j; s is the quantization level. Note that, the quantization level is roughly exponential to the number of quantized bits. If we use b bits to quantize g_j, we will use one bit to represent its sign and the other $b-1$ bits to represent $\zeta(g_j, s)$, thus resulting in a quantization level $s = 2^{b-1} - 1$. And $\zeta(g_j, s)$ is an unbiased stochastic function that maps scalar $|g_j|/\|\mathbf{g}\|$ to one of the values in set $\{0, 1/s, 2/s, \dots, s/s\}$: if $|g_j|/\|\mathbf{g}\| \in [l/s, (l+1)/s]$, we have

$$\zeta(g_j, s) = \begin{cases} l/s, & \text{with probability } 1 - p_r, \\ (l+1)/s, & \text{with probability } p_r = s\frac{|g_j|}{\|\mathbf{g}\|} - l. \end{cases} \tag{48}$$

So we have

$$\begin{aligned} \mathbb{E}[\zeta(g_i, s)] &= \frac{l}{s}[1 - s\frac{|g_i|}{\|\mathbf{g}\|} + l] + \frac{l+1}{s}[s\frac{|g_i|}{\|\mathbf{g}\|} - l] \\ &= \frac{|g_i|}{\|\mathbf{g}\|} \end{aligned}$$

Then

$$\begin{aligned} \mathbb{E}[\zeta(g_i, s)^2] &= \mathbb{E}[\zeta(g_i, s)]^2 + \mathbb{V}[\zeta(g_i, s)] \\ &= \frac{|g_i|^2}{\|\mathbf{g}\|^2} + \frac{1}{s^2}p(1-p) \\ &\leq \frac{|g_i|^2}{\|\mathbf{g}\|^2} + \frac{1}{4s^2} \end{aligned}$$

Considering that $Q_s(g_i) = \|\mathbf{g}\| \cdot \mathrm{sgn}(g_i) \cdot \zeta(g_i, s)$, we have

$$\begin{aligned} \mathbb{E}[\|Q_b[\mathbf{g}]\|^2] &= \sum_{i=0}^{d} \mathbb{E}[\|\mathbf{g}\|^2 \zeta(g_i, s)^2] \\ &\leq \sum_{i=0}^{d} \|\mathbf{g}\|^2 (\frac{|g_i|^2}{\|\mathbf{g}\|^2} + \frac{1}{4s^2}) \\ &= \|\mathbf{g}\|^2 + \frac{d}{4s^2}\|\mathbf{g}\|^2 \end{aligned}$$

So we can get

$$\mathbb{E}[Q_b[\mathbf{g}]] = \mathbf{g}$$

$$\mathbb{E}[\|Q_b[\mathbf{g}]\|^2] \leq \left[1 + \frac{d}{4^b}\right]\|\mathbf{g}\|^2$$

For the stochastic gradient vector $\mathbf{g}$, if the sparsification parameter is k, then we can get

$$\mathbb{E}[S_k(\mathbf{g})] = \mathbf{g}$$

$$\mathbb{E}[\|S_k(\mathbf{g})\|^2] \leq \mathbb{E}[\|S_k(\mathbf{g})\|^2] = \frac{d}{k}\|\mathbf{g}\|^2$$

Therefore,

$$\mathbb{E}[Q_b[S_k(\mathbf{g})]] = \mathbb{E}[S_k(\mathbf{g})] = \mathbf{g}$$

$$\begin{aligned} \mathbb{E}[\|Q_b[S_k(\mathbf{g})]\|^2] &\leq \left[1 + \frac{k}{4^b}\right]\|S_k(\mathbf{g})\|^2 \\ &= \left[1 + \frac{k}{4^b}\right] * \frac{d}{k}\|\mathbf{g}\|^2 \\ &= \left[\frac{d}{k} + \frac{d}{4^b}\right]\|\mathbf{g}\|^2 \end{aligned}$$

B. Proof of Eq. (14)-(16)

Firstly, we have the follow optimization problem:

$$\min_{b,k} \frac{d}{4^b} + \frac{d-k}{k}$$

$$k(b + \log_2 d) + B_{pre} = c$$

From $k(b + \log_2 d) + B_{pre} = c$, we can get

$$k = \frac{c - B_{pre}}{b + \log_2 d}$$

So we need to minimize $h(b) = \frac{d}{4^b} + \frac{db + d\log_2 d}{c - B_{pre}} - 1$. From $\frac{\partial h(b)}{\partial b} = 0$, we can get

$$b^* = \frac{1}{2}\log_2\left[2\ln 2 * (c - B_{pre})\right]$$

Hence,

$$k^* = \frac{c - B_{pre}}{\frac{1}{2}\log_2\left[2\ln 2 * (c - B_{pre})\right] + \log_2 d}$$

Then we have

$$\begin{aligned} h(b^*, k^*) &= \frac{d - k^*}{k^*} + \frac{d}{4^{b^*}} \\ &= \frac{d(\frac{1}{2}\log_2\left[2\ln 2 * (c - B_{pre})\right] + \log_2 d)}{c - B_{pre}} \\ &\quad + \frac{d}{2\ln 2 * (c - B_{pre})} - 1 \\ &\overset{(a)}{\approx} \frac{d(\frac{1}{2}\log_2\left[2\ln 2 * c\right] + \log_2 d)}{c} + \frac{d}{2\ln 2 * c} - 1 \\ &= \frac{d}{c}\left[\frac{3}{2}\log_2 d + \frac{1}{2}\log_2\frac{2cln2}{d} + \frac{1}{2\ln 2} - \frac{c}{d}\right] \\ &\overset{(b)}{\leq} \frac{3d\log_2 d}{2c} \end{aligned}$$

where (a) consider $c >> B_{pre}$, and where (b) consider that $f(x) = \frac{1}{2}\log_2(2xln2) + \frac{1}{2\ln 2} - x \leq 0$ for $x > 0$.

C. Proof of Eq. (24)-(25)

The optimization problem Eq.(23) can be solved separately for each client

$$\begin{aligned} \min_{\{c_t^{(i)}\}} \quad & \sum_{t=0}^{T-1} \alpha^{T-1-t}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \\ s.t. \quad & \sum_{t=0}^{T-1} c_t^{(i)} \leq C^{(i)} \end{aligned}$$

For the objective function, we have

$$\frac{\partial^2[\sum_{t=0}^{T-1}\alpha^{T-1-t}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}]}{\partial(c_t^{(i)})^2} = \sum_{t=0}^{T-1}\frac{2\alpha^{T-1-t}\|\mathbf{g}_t^{(i)}\|^2}{(c_t^{(i)})^3} > 0$$

Hence, this optimization problem is a convex optimization problem, then we have the Lagrange function

$$\mathcal{L}(c_t^{(i)}, \lambda) = \sum_{t=0}^{T-1}\alpha^{T-1-t}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} + \lambda[\sum_{t=0}^{T-1}c_t^{(i)} - C^{(i)}]$$

where λ is Lagrange multiplier. Then we can get

$$\frac{\partial\mathcal{L}(c_t^{(i)}, \lambda)}{\partial c_t^{(i)}} = -\alpha^{T-1-t}\frac{\|\mathbf{g}_t^{(i)}\|^2}{(c_t^{(i)})^2} + \lambda = 0$$

By solve this equation, we can get

$$c_t^{(i)} = r_{\mathcal{C}}^{(i)}\alpha^{(T-1-t)/2}\|\mathbf{g}_t^{(i)}\|$$

Hence, the communication cost is

$$\begin{aligned} C^{(i)} &= \sum_{t=0}^{T-1} r_{\mathcal{C}}^{(i)}\alpha^{(T-1-t)/2}\|\mathbf{g}_t^{(i)}\| \\ &\leq r_{\mathcal{C}}^{(i)}\sqrt{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2}\sum_{t=0}^{T-1}\alpha^{(T-1-t)/2} \\ &= r_{\mathcal{C}}^{(i)}\sqrt{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2}\frac{1-\alpha^{T/2}}{1-\alpha^{1/2}} \end{aligned}$$

So,

$$r_{\mathcal{C}}^{(i)} \geq \frac{C^{(i)}}{\sqrt{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2}\frac{1-\alpha^{T/2}}{1-\alpha^{1/2}}}$$

D. Proof of Theorem 1

Firstly, we consider function F is L-smooth, and use Eq. (6), we have:

$$F(\mathbf{x}_{t+1}) \leq F(\mathbf{x}_t) + \nabla F(\mathbf{x}_t)^{\mathrm{T}}(\mathbf{x}_{t+1} - \mathbf{x}_t) + \frac{L}{2}\|\mathbf{x}_{t+1} - \mathbf{x}_t\|^2$$

For the Compressed SGD, $\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{\eta}{W}\sum_{i=1}^{W}\mathcal{C}_{c_t}[\mathbf{g}_t^{(i)}]$, so:

$$\begin{aligned} F(\mathbf{x}_{t+1}) \leq F(\mathbf{x}_t) + \nabla F(\mathbf{x}_t)^{\mathrm{T}}(-\frac{\eta}{W}\sum_{i=1}^{W}\mathcal{C}_{c_t}[\mathbf{g}_t^{(i)}]) \\ + \frac{L}{2}\| - \frac{\eta}{W}\sum_{i=1}^{W}\mathcal{C}_{c_t}[\mathbf{g}_t^{(i)}]\|^2 \end{aligned}$$

Taking total expectations, and using Eq. (20), this yields:

$$\begin{aligned} \mathbb{E}[F(\mathbf{x}_{t+1})] \leq F(\mathbf{x}_t) + (-\eta + \frac{L\eta^2}{2})\|\nabla F(\mathbf{x}_t)\|^2 \\ + \frac{L\eta^2\sigma^2}{2W} + \frac{3dL\eta^2\log_2 d}{4W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \end{aligned}$$

Then we consider that function F is μ-strongly convex, and use Eq. (9), we can get :

$$\begin{aligned} \mathbb{E}[F(\mathbf{x}_{t+1})] \leq F(\mathbf{x}_t) - (2\mu\eta - L\mu\eta^2)[F(\mathbf{x}_t) - F(\mathbf{x}^*)] \\ + \frac{L\eta^2\sigma^2}{2W} + \frac{3dL\eta^2\log_2 d}{4W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \end{aligned}$$

Subtracting $F(\mathbf{x}^*)$ from both sides, and let $\alpha(\eta) := 1 - 2\mu\eta + L\mu\eta^2$, so:

$$\begin{aligned} \mathbb{E}[F(\mathbf{x}_{t+1}) - F(\mathbf{x}^*)] \leq \alpha[F(\mathbf{x}_t) - F(\mathbf{x}^*)] + \frac{L\eta^2\sigma^2}{2W} \\ + \frac{3dL\eta^2\log_2 d}{4W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \end{aligned}$$

Applying this recursively:

$$\begin{aligned} \mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] \leq \alpha^T[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \frac{L\eta^2\sigma^2[1-\alpha^T]}{2W(1-\alpha)} \\ + \frac{3dL\eta^2\log_2 d}{4W^2}\sum_{t=0}^{T-1}\alpha^{T-1-t}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \end{aligned}$$

Similarly, we firstly consider function F is μ-strongly convex, and use Eq. (8), we have:

$$F(\mathbf{x}_{t+1}) \geq F(\mathbf{x}_t) + \nabla F(\mathbf{x}_t)^{\mathrm{T}}(\mathbf{x}_{t+1} - \mathbf{x}_t) + \frac{\mu}{2}\|\mathbf{x}_{t+1} - \mathbf{x}_t\|^2$$

For the Compressed SGD, $\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{\eta}{W}\sum_{i=1}^{W}\mathcal{C}_{c_t}[\mathbf{g}_t^{(i)}]$, so:

$$\begin{aligned} F(\mathbf{x}_{t+1}) \geq F(\mathbf{x}_t) + \nabla F(\mathbf{x}_t)^{\mathrm{T}}(-\frac{\eta}{W}\sum_{i=1}^{W}\mathcal{C}_{c_t}[\mathbf{g}_t^{(i)}]) \\ + \frac{\mu}{2}\| - \frac{\eta}{W}\sum_{i=1}^{W}\mathcal{C}_{c_t}[\mathbf{g}_t^{(i)}]\|^2 \end{aligned}$$

Taking total expectations, and using Eq. (20) (the worst case of gradient noise), this yields:

$$\begin{aligned} \mathbb{E}[F(\mathbf{x}_{t+1})] \geq F(\mathbf{x}_t) + (-\eta + \frac{\mu\eta^2}{2})\|\nabla F(\mathbf{x}_t)\|^2 \\ + \frac{\mu\eta^2\sigma^2}{2W} + \frac{3d\mu\eta^2\log_2 d}{4W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \end{aligned}$$

Then we consider that function F is L-smooth, and use Eq. (7), we can get :

$$\begin{aligned} \mathbb{E}[F(\mathbf{x}_{t+1})] \geq F(\mathbf{x}_t) - (2L\eta - L\mu\eta^2)[F(\mathbf{x}_t) - F(\mathbf{x}^*)] \\ + \frac{\mu\eta^2\sigma^2}{2W} + \frac{3d\mu\eta^2\log_2 d}{4W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \end{aligned}$$

Subtracting $F(\mathbf{x}^*)$ from both sides, and let $\beta(\eta) := 1 - 2L\eta + L\mu\eta^2$, so:

$$\begin{aligned} \mathbb{E}[F(\mathbf{x}_{t+1}) - F(\mathbf{x}^*)] \geq \beta[F(\mathbf{x}_t) - F(\mathbf{x}^*)] + \frac{\mu\eta^2\sigma^2}{2W} \\ + \frac{3d\mu\eta^2\log_2 d}{4W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \end{aligned}$$

Applying this recursively:

$$
\begin{aligned}
&\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] \\
&\geq \beta^T[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \frac{\mu\eta^2\sigma^2[1-\beta^T]}{2W(1-\beta)} \\
&\quad + \frac{3d\mu\eta^2\log_2 d}{4W^2}\sum_{t=0}^{T-1}\beta^{T-1-t}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}
\end{aligned}
$$

E. Proof of Corollary 1

If the compression bits for client i at iteration t is

$$
c_t^{(i)} = r_{\mathcal{C}}^{(i)}\alpha^{(T-1-t)/2}\|\mathbf{g}_t^{(i)}\|
$$

where $r_{\mathcal{C}}^{(i)} = \frac{C^{(i)}}{\sqrt{2L[F(\mathbf{x}_0)-F(\mathbf{x}^*)]+\sigma^2}\frac{1-\alpha^{T/2}}{1-\alpha^{1/2}}}$. So the convergence error due to compression is

$$
\begin{aligned}
&\sum_{t=0}^{T-1}\alpha^{T-1-t}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \\
&= \sum_{t=0}^{T-1}\alpha^{T-1-t}\frac{\|\mathbf{g}_t^{(i)}\|^2}{r_{\mathcal{C}}^{(i)}\alpha^{(T-1-t)/2}\|\mathbf{g}_t^{(i)}\|} \\
&\leq \frac{\{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2\}}{C^{(i)}}\left[\frac{1-\alpha^{T/2}}{1-\alpha^{1/2}}\right]^2 \\
&= \frac{T^2\{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2\}}{C^{(i)}}AM^2_{\alpha^{t/2}}
\end{aligned}
$$

where Arithmetic Mean $AM_{\alpha^{t/2}} = \frac{1}{T}\sum_{t=0}^{T-1}\alpha^{t/2}$.

F. Proof of Corollary 2

If we fixed the compression bits, that is $c_t^{(i)} = \frac{C^{(i)}}{T}$, so the convergence error due to compression is

$$
\begin{aligned}
&\sum_{t=0}^{T-1}\alpha^{T-1-t}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \\
&= \sum_{t=0}^{T-1}\alpha^{T-1-t}\frac{T\|\mathbf{g}_t^{(i)}\|^2}{C^{(i)}} \\
&\leq \frac{T\{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2\}}{C^{(i)}}\sum_{t=0}^{T-1}\alpha^{T-1-t} \\
&\leq \frac{T^2\{2L[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \sigma^2\}}{C^{(i)}}QM^2_{\alpha^{t/2}}
\end{aligned}
$$

where Quadratic Mean $QM_{\alpha^{t/2}} = \sqrt{\frac{1}{T}\sum_{t=0}^{T-1}\alpha^t}$.

G. Proof of Theorem 2

According to D, if F is L-smooth, we have:

$$
\begin{aligned}
\mathbb{E}[F(\mathbf{x}_{t+1})] \leq F(\mathbf{x}_t) + (-\eta + \frac{L\eta^2}{2})\|\nabla F(\mathbf{x}_t)\|^2 \\
+\frac{L\eta^2\sigma^2}{2W} + \frac{3dL\eta^2\log_2 d}{4W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}
\end{aligned}
$$

Subtracting $F(\mathbf{x}_t)$ from both sides, then

$$
\begin{aligned}
&\mathbb{E}[F(\mathbf{x}_{t+1}) - F(\mathbf{x}_t)] \\
&\leq (-\eta + \frac{L\eta^2}{2})\|\nabla F(\mathbf{x}_t)\|^2 + \frac{L\eta^2\sigma^2}{2W} + \frac{3dL\eta^2\log_2 d}{4W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \\
&\overset{(a)}{\leq} (-\eta + \frac{L\eta^2}{2})\gamma_t\|\nabla F(\mathbf{x}_t)\|^2 + \frac{L\eta^2\sigma^2\gamma_t}{2W} \\
&\quad + \frac{3dL\eta^2\log_2 d}{4W^2}\gamma_t\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}
\end{aligned}
$$

where $0 < \gamma_t < 1$ and (a) considers $(-\eta + \frac{L\eta^2}{2})\|\nabla F(\mathbf{x}_t)\|^2 + \frac{L\eta^2\sigma^2}{2W} + \frac{3dL\eta^2\log_2 d}{4W^2}\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}} \leq 0$. Applying it recursively, this yields:

$$
\begin{aligned}
&\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}_0)] \\
&\leq (-\eta + \frac{L\eta^2}{2})\sum_{t=0}^{T-1}\gamma_t\mathbb{E}[\|\nabla F(\mathbf{x}_t)\|_2^2] + \frac{L\eta^2\sigma^2\sum_{t=0}^{T-1}\gamma_t}{2W} \\
&\quad + \frac{3dL\eta^2\log_2 d}{4W^2}\sum_{t=0}^{T-1}\gamma_t\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}
\end{aligned}
$$

Considering that $F(\mathbf{x}_T) \geq F(\mathbf{x}^*)$, so:

$$
\begin{aligned}
&\frac{1}{\sum_{t=0}^{T-1}\gamma_t}\sum_{t=0}^{T-1}\gamma_t\mathbb{E}[\|\nabla F(\mathbf{x}_t)\|_2^2] \\
&\leq \frac{2}{(2\eta - L\eta^2)\sum_{t=0}^{T-1}\gamma_t}[F(\mathbf{x}_0) - F(\mathbf{x}^*)] + \frac{L\eta\sigma^2}{(2-L\eta)W} \\
&\quad + \frac{3dL\eta\log_2 d}{(4W^2 - 2W^2L\eta)\sum_{t=0}^{T-1}\gamma_t}\sum_{t=0}^{T-1}\gamma_t\sum_{i=1}^{W}\frac{\|\mathbf{g}_t^{(i)}\|^2}{c_t^{(i)}}
\end{aligned}
$$

H. Proof of Theorem 3

For a quadratic optimization problem $F(\mathbf{x}) = 1/2\mathbf{x}^{\mathrm{T}}\mathbf{H}\mathbf{x} + \mathbf{A}^{\mathrm{T}}\mathbf{x} + B$, we consider a Gaussian noise case

$$
\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\nabla F(\mathbf{x}_t) - \eta\boldsymbol{\epsilon}_t, \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\mathbf{x}_t))
$$

Then we have

$$
\begin{aligned}
\mathbf{x}_{t+1} &= \mathbf{x}_t - \eta\nabla F(\mathbf{x}_t) - \eta\boldsymbol{\epsilon}_t \\
&= \mathbf{x}_t - \eta[\mathbf{H}\mathbf{x}_t + \mathbf{A}] - \eta\boldsymbol{\epsilon}_t \\
&= (\mathbf{I} - \eta\mathbf{H})\mathbf{x}_t - \eta\mathbf{A} - \eta\boldsymbol{\epsilon}_t
\end{aligned}
$$

Considering $\nabla F(\mathbf{x}^*) = \eta\mathbf{A} + \eta\mathbf{H}\mathbf{x}^* = 0$, subtracting $\mathbf{x}^*$ from both sides, and rearranging, this yields:

$$
\begin{aligned}
\mathbf{x}_{t+1} - \mathbf{x}^* &= (\mathbf{I} - \eta\mathbf{H})\mathbf{x}_t - \eta\mathbf{A} - \mathbf{x}^* - \eta\boldsymbol{\epsilon}_t \\
&= (\mathbf{I} - \eta\mathbf{H})(\mathbf{x}_t - \mathbf{x}^*) - \eta\mathbf{A} - \eta\mathbf{H}\mathbf{x}^* - \eta\boldsymbol{\epsilon}_t \\
&= (\mathbf{I} - \eta\mathbf{H})(\mathbf{x}_t - \mathbf{x}^*) - \eta\boldsymbol{\epsilon}_t
\end{aligned}
$$

Applying this recursively, let $\boldsymbol{\rho} = \mathbf{I} - \eta\mathbf{H}$, we have:

$$
\mathbf{x}_T - \mathbf{x}^* = \boldsymbol{\rho}^T(\mathbf{x}_0 - \mathbf{x}^*) - \sum_{t=0}^{T-1}[\eta\boldsymbol{\rho}^{T-1-t}\boldsymbol{\epsilon}_t]
$$

Considering that $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}(\mathbf{x}_t))$, then:

$$\begin{aligned}
&\sum_{t=0}^{T-1}[\eta\boldsymbol{\rho}^{T-1-t}\boldsymbol{\epsilon}_t] \\
&= \sum_{t=0}^{T-1}[\eta\boldsymbol{\rho}^{T-1-t}\boldsymbol{\Sigma}(\mathbf{x}_t)^{\frac{1}{2}}\mathcal{N}(\mathbf{0}, \mathbf{I})] \\
&= \sum_{t=0}^{T-1}[\eta\boldsymbol{\rho}^{T-1-t}\boldsymbol{\Sigma}(\mathbf{x}_t)^{\frac{1}{2}}[\mathbf{W}(t+1) - \mathbf{W}(t)]\} \\
&\equiv I(T)
\end{aligned}$$

where, $\mathbf{W}$ is a standard d-dimensional Wiener process, and $I(T)$ is an Ito integral. Hence $\mathbf{x}_T = \mathbf{x}^* + \boldsymbol{\rho}^T(\mathbf{x}_0 - \mathbf{x}^*) - I(T)$, then:

$$\begin{aligned}
F(\mathbf{x}_T) &= \frac{1}{2}\mathbf{x}_T{}^{\mathrm{T}}\mathbf{H}\mathbf{x}_T + \mathbf{A}^{\mathrm{T}}\mathbf{x}_T + B \\
&= \frac{1}{2}(\mathbf{x}^{(0)} - \mathbf{x}^*)^{\mathrm{T}}(\boldsymbol{\rho}^T)^{\mathrm{T}}\mathbf{H}\boldsymbol{\rho}^T(\mathbf{x}_0 - \mathbf{x}^*) \\
&\quad + \frac{1}{2}I(T)^{\mathrm{T}}\mathbf{H}I(T) + F(\mathbf{x}^*) \\
&\quad - [\boldsymbol{\rho}^T(\mathbf{x}_0 - \mathbf{x}^*) + \mathbf{x}^* + \mathbf{A}]^{\mathrm{T}}\mathbf{H}I(T)
\end{aligned}$$

Subtracting $F(\mathbf{x}^*)$ from both sides, taking total expectations, and rearranging, this yields:

$$\begin{aligned}
&\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] \\
&= \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^*)^{\mathrm{T}}(\boldsymbol{\rho}^T)^{\mathrm{T}}\mathbf{H}\boldsymbol{\rho}^T(\mathbf{x}_0 - \mathbf{x}^*) + \frac{1}{2}\mathbb{E}[I(T)^{\mathrm{T}}\mathbf{H}I(T)] \\
&\quad - [\boldsymbol{\rho}^T(\mathbf{x}_0 - \mathbf{x}^*) + \mathbf{x}^* + \mathbf{A}]^{\mathrm{T}}\mathbf{H}\mathbb{E}[I(T)]
\end{aligned}$$

The property of Ito integral $I(T)$ is:

$$\mathbb{E}[I(T)] = 0$$

$$\mathbb{E}[I(T)^{\mathrm{T}}\mathbf{H}I(T)] = \sum_{t=0}^{T-1}\eta^2\mathrm{Tr}[\boldsymbol{\rho}^{T-1-t}\boldsymbol{\Sigma}(\mathbf{x}_t)\mathbf{H}(\boldsymbol{\rho}^{T-1-t})^{\mathrm{T}}]$$

Using this property, we have:

$$\begin{aligned}
\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] &= \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^*)^{\mathrm{T}}(\boldsymbol{\rho}^T)^{\mathrm{T}}\mathbf{H}\boldsymbol{\rho}^T(\mathbf{x}_0 - \mathbf{x}^*) \\
&\quad + \frac{\eta^2}{2}\sum_{t=0}^{T-1}\mathrm{Tr}[\boldsymbol{\rho}^{T-1-t}\boldsymbol{\Sigma}(\mathbf{x}_t)\mathbf{H}(\boldsymbol{\rho}^{T-1-t})^{\mathrm{T}}]
\end{aligned}$$

If we consider a simple example: the Hessian matrix is isotropic $\mathbf{H} = \lambda\mathbf{I}$, let $\alpha(\eta) := 1 - 2\eta\lambda + \eta^2\lambda^2$, so

$$\begin{aligned}
first &= \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^*)^{\mathrm{T}}(\boldsymbol{\rho}^T)^{\mathrm{T}}\mathbf{H}\boldsymbol{\rho}^T(\mathbf{x}_0 - \mathbf{x}^*) \\
&= \alpha(\eta)^T\frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^*)^{\mathrm{T}}\mathbf{H}(\mathbf{x}_0 - \mathbf{x}^*) \\
&= \alpha(\eta)^T[\frac{1}{2}\mathbf{x}_0^{\mathrm{T}}\mathbf{H}\mathbf{x}_0 + \frac{1}{2}\mathbf{x}^{*\mathrm{T}}\mathbf{H}\mathbf{x}^* - \mathbf{x}_0^{\mathrm{T}}\mathbf{H}\mathbf{x}^*] \\
&= \alpha(\eta)^T[\frac{1}{2}\mathbf{x}_0^{\mathrm{T}}\mathbf{H}\mathbf{x}_0 + \frac{1}{2}\mathbf{x}^{*\mathrm{T}}\mathbf{H}\mathbf{x}^* - \mathbf{x}_0^{\mathrm{T}}\mathbf{H}\mathbf{x}^* \\
&\quad + \mathbf{x}_0^{\mathrm{T}}(\mathbf{H}\mathbf{x}^* + A) - \mathbf{x}^{*\mathrm{T}}(\mathbf{H}\mathbf{x}^* + A)] \\
&= \alpha(\eta)^T[\frac{1}{2}\mathbf{x}_0^{\mathrm{T}}\mathbf{H}\mathbf{x}_0 - \frac{1}{2}\mathbf{x}^{*\mathrm{T}}\mathbf{H}\mathbf{x}^* + \mathbf{x}_0^{\mathrm{T}}A - \mathbf{x}^{*\mathrm{T}}A] \\
&= \alpha(\eta)^T[F(\mathbf{x}_0) - F(\mathbf{x}^*)]
\end{aligned}$$

$$second = \frac{\lambda\eta^2}{2}\sum_{t=0}^{T-1}\alpha(\eta)^{T-1-t}\mathrm{Tr}[\boldsymbol{\Sigma}(\mathbf{x}_t)]$$

Thus,

$$\begin{aligned}
\mathbb{E}[F(\mathbf{x}_T) - F(\mathbf{x}^*)] &= \alpha(\eta)^T[F(\mathbf{x}_0) - F(\mathbf{x}^*)] \\
&\quad + \frac{\lambda\eta^2}{2}\sum_{t=0}^{T-1}\alpha(\eta)^{T-1-t}\mathrm{Tr}[\boldsymbol{\Sigma}(\mathbf{x}_t)]
\end{aligned}$$