Global-QSGD: Allreduce-Compatible Quantization for Distributed Learning with Theoretical Guarantees

Jihao Xin
KAUST

Marco Canini
KAUST

Peter Richtárik
KAUST

Samuel Horváth
MBZUAI

Abstract

Distributed training enables large-scale deep learning, but suffers from high communication overhead, especially as models and datasets grow. Gradient compression, particularly quantization, is a promising approach to mitigate this bottleneck. However, existing quantization schemes are often incompatible with Allreduce, the dominant communication primitive in distributed deep learning, and many prior solutions rely on heuristics without theoretical guarantees. We introduce Global-QSGD, an Allreduce-compatible gradient quantization method that leverages global norm scaling to reduce communication overhead while preserving accuracy. Global-QSGD is backed by rigorous theoretical analysis, extending standard unbiased compressor frameworks to establish formal convergence guarantees. Additionally, we develop a performance model to evaluate its impact across different hardware configurations. Extensive experiments on NVLink, PCIe, and large-scale cloud environments show that Global-QSGD accelerates distributed training by up to 3.51× over baseline quantization methods, making it a practical and efficient solution for large-scale deep learning workloads.

CCS Concepts

• **Computing methodologies** → **Machine learning**; *Distributed algorithms.*

Keywords

Distributed Training, Gradient Compression, Collective Communication

ACM Reference Format:
Jihao Xin, Marco Canini, Peter Richtárik, and Samuel Horváth. 2025. Global-QSGD: Allreduce-Compatible Quantization for Distributed Learning with Theoretical Guarantees. In *The 5th Workshop on Machine Learning and Systems (EuroMLSys '25), March 30–April 3, 2025, Rotterdam, Netherlands.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3721146.3721932

EuroMLSys '25, Rotterdam, Netherlands

ACM ISBN 979-8-4007-1538-9/25/03
https://doi.org/10.1145/3721146.3721932

1 Introduction

Distributed deep learning has become the standard approach for scaling training across multiple compute nodes, enabling faster convergence on large models and datasets [1, 9]. However, as training scales up, communication overhead increasingly dominates total runtime, particularly in large-scale deployments. For example, Sapio et al. [25] reports that communication accounts for up to more than 90% of the total training time in deep learning workloads, significantly limiting the benefits of additional computing resources.

Gradient quantization has emerged as a practical solution to alleviate this bottleneck, which reduces the gradient precision (e.g., from 32-bit to 8-bit) using random rounding, making it computationally efficient and naturally unbiased (in the sense that, in expectation, the quantized gradient is an unbiased estimator of original gradient). However, existing quantization methods such as QSGD [3] are often impractical in real-world scenarios due to their incompatibility with *Allreduce*–the dominant communication primitive for distributed AI tasks [2, 6, 7, 18, 19, 23]. The key issue arises because quantization is performed locally on each worker. For example, when reducing the precision from 32-bit to 8-bit, the gradients must be scaled using a *norm*, which varies between workers. As a result, gradients are quantized at different scales, and directly aggregating these quantized values via Allreduce leads to inconsistencies.

To address this limitation, we introduce **Global-QSGD**. Instead of using local norms for scaling, Global-QSGD leverages a *global norm* computed across all workers, ensuring consistent quantization scales and enabling Allreduce compatibility; that is, quantized gradient values can be directly aggregated in their compressed (quantized) representation without the need (and overhead) of switching numeric representation. Like other compression methods, quantization inevitably loses information. Traditional quantization applies a *linear scaling* approach, known as **standard dithering**,

Table 1: Comparision of Allreduce-Compatible Compressors

Algorithm	Seamless Integration	w/o Extra Step	w/o Heuristic	w/o EF	Tunable Compression
PowerSGD	✗	✗	✗	✗	✓
GradiVeQ	✗	✗	✗	✓	✓
IntSGD	✓	✓	✗	✓	✗
THC	✗	✗	✗	✗	✓
Global-QSGD	✓	✓	✓	✓	✓

which is suboptimal because gradients decrease in magnitude as the model converges. Smaller gradients require finer precision to maintain accuracy. To address this, we propose **exponential dithering**, which allocates higher precision to smaller gradient values, improving convergence accuracy.

Unlike previous approaches that rely on heuristics, we provide a rigorous convergence analysis in Section 4 to establish the theoretical foundations. Using its unbiased nature, we prove that Global-QSGD maintains a bounded variance, ensuring stable convergence and making it a theoretically sound choice. To study the impact across different hardware configurations, we derive a performance modeling in Section 5. We also evaluate Global-QSGD on various hardware settings in Section 7, including single-node and cloud environments. Our results show that Global-QSGD accelerates model training by up to **3.51**×, significantly reducing communication overhead while maintaining model accuracy. This paper makes the following key contributions:

- Global-QSGD is the first quantization method that integrates seamlessly with Allreduce, achieving up to $O(\sqrt{n})$ higher compression efficiency than QSGD.
- We establish a general theoretical framework for unbiased compressors, proving that Global-QSGD maintains bounded variance and ensures convergence.

2 Related Work

Gradient compression methods can be broadly categorized into three approaches: *sparsification* [2, 4, 17, 20, 27, 28, 31], *decomposition* [21, 30], and *quantization* [3, 13, 22, 26, 32]. Although these methods differ in how they reduce communication overhead, they can also be classified according to their impact on gradient updates, falling into two major theoretical categories: *unbiased* and *biased* compressors. Unbiased compressors, such as quantization schemes like QSGD [3], maintain an expectation-equal gradient estimate with bounded variance, making them easier to analyze in theoretical frameworks [11]. In contrast, biased compressors, including *Top-k* [4], *SignSGD* [5], and *PowerSGD* [30], introduce systematic distortion, requiring correction mechanisms such as error feedback (EF) [15, 24, 27] or induced compressors (IC) [12] to restore convergence guarantees. However, these correction mechanisms introduce additional memory overhead (EF) or

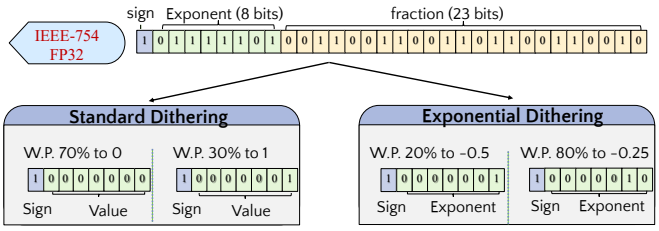


Figure 1: Quantize -0.3 from FP32 to 8 Bits

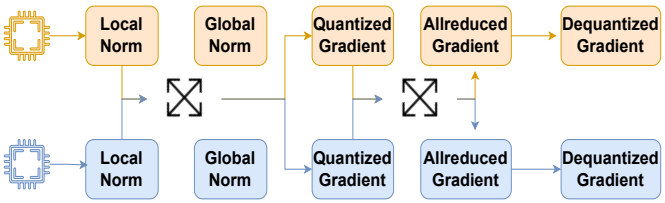


Figure 2: Global-QSGD Workflow

rely on an auxiliary unbiased compressor (IC), which limits their practical efficiency.

Unfortunately, most of the existing compressors are not naively compatible with Allreduce since the sum of two compressed vectors is not an inexpensive operation, thus it requires an expensive decompress-aggregate-compress flow. This includes greedy and random sparsification[1], quantization, and sign-based methods.

Several compressors have been proposed to ensure *Allreduce* compatibility, but all rely on some heuristics, so they all lack rigorous theoretical guarantees. In addition, some of them introduce an additional step, which blocks the seamless replacing with existing *Allreduce*. GradiVeQ assumes adjacent gradients are linearly correlated, IntSGD requires the clipping of communicated integers, and PowerSGD approximates low-rank decomposition using power iteration while depending on memory-intensive error feedback. In particular, THC also leverages the *global norm* but needs to solve an *integer linear programming (ILP)problem* to heuristically choose the quantization intervals based on the data distribution. A detailed comparison is provided in Table 1.

3 Global-QSGD

In this section, we present the design details of Global-QSGD. As a theoretically rigorous approach, we first establish a formal mathematical formulation (Section 3.1), which serves as the foundation for analyzing convergence (Section 4). We then describe the workflow and design details (Section 3.2,3.3).

[1]For random sparsification, one can share random seeds across workers to make it Allreduce compatible, e.g., see synchronized random seed [33].

Table 2: Key Notation Table

Symbol	Description
d	Dimension of gradients
n	Number of workers
s	Number of divided intervals
l_i	Value of the i_{th} interval
$x, y \in \mathbb{R}^d$	Example vector with dimension d
$x_i \in \mathbb{R}^d$	Gradients of the i_{th} worker
$y_i \in \mathbb{R}^d$	Normalized Gradients of the i_{th} worker
$\mathbf{x} \in \mathbb{R}^{nd}$	Gradients of all n workers
$\mathbf{y} \in \mathbb{R}^{nd}$	Normalized Gradients of all n workers
$\|x\|_p$	ℓ_p norm of vector x
$\mathbb{U}^{n,d}(\theta)$	Unbiased compressor set from $\mathbb{R}^{nd}$ to $\mathbb{R}^d$
$\|\mathbf{x}\|_{q,p}$	(q,p)-mixed norm of vector $\mathbf{x}$
$\mathcal{G}(\mathbf{x})$	Unbiased distributed mean compressor
$\xi_i(y_i)$	Random rounding of the i_{th} worker
Global-$\mathcal{Q}_s^{q,p}$	compressor in general
$Global-\mathcal{L}_s^{q,p}$	standard dithering
$Global-\mathcal{E}_s^{q,p}$	exponential dithering
$\mathcal{Q}_s^{q,p}, \mathcal{L}_s^{q,p}, \mathcal{E}_s^{q,p}$	Global-QSGD's local compressor

3.1 Theoretical Formulation

We denote ℓ_p-norms for $y \in \mathbb{R}^d$ as $\|y\|_p \stackrel{\text{def}}{=} (\sum_{i=1}^d |y_i|^p)^{1/p}$ for $p \in (1, \infty)$. For $p = \infty$, $\|y\|_p$ denotes the maximum element of y in terms of magnitude. Let $\mathbf{x} = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{nd}$, where $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$. We denote the (q,p)-mixed norm in $\mathbb{R}^{nd}$ as $\|\mathbf{x}\|_{q,p} \stackrel{\text{def}}{=} (\sum_{i=1}^d \|x_i\|_q^p)^{1/p}$. For any vectors x, y, $x \circ y$ represents their element-wise multiplication, and, for any vector x, $|x|$, sign x stand for element-wise absolute value and signum operations, respectively. We denote $[n] \stackrel{\text{def}}{=} \{1, 2, \ldots, n\}$ for any $n \in \mathbb{N}$. We provide a notation table in Table 2. Now we can formally define Global-QSGD:

Definition 3.1 (Global-$\mathcal{Q}_s^{q,p}$)**.** The Global-QSGD operator with respect to the (q,p)-mixed norm and with s levels

$$0 = l_s < l_{s-1} < l_{s-2} < \cdots < l_1 < l_0 = 1,$$

denoted Global-$\mathcal{Q}_s^{q,p}$, is defined as follows. Let $\mathbf{x} = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{nd}$. Let $y_i \stackrel{\text{def}}{=} |x_i|/\|\mathbf{x}\|_{q,p} \in \mathbb{R}^d$ for all $i \in [1, ..., n]$. Then

$$\text{Global-}\mathcal{Q}_s^{q,p}(\mathbf{x}) \stackrel{\text{def}}{=} \|\mathbf{x}\|_{q,p} \frac{1}{n} \sum_{i=1}^n \text{sign}(x_i) \circ \xi_i(y_i)\,, \quad (1)$$

where $\xi_i(y_i)$ is an independent element-wise random rounding operator such that

$$(\xi_i(y_i))_j \stackrel{\text{def}}{=} \begin{cases} l_{u_i^j} & \text{with probability } \frac{(y_i)_j - l_{u_i^j+1}}{l_{u_i^j} - l_{u_i^j+1}} \\ l_{u_i^j+1} & \text{otherwise} \end{cases}, \quad (2)$$

Algorithm 1 Global-QSGD

1: **Input:** Update $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ distributed among n machines, *sparse* $\in$ {True, False}, Global-$\mathcal{Q}_s^{q,p}$
2: $\|\mathbf{x}\|_p = (\text{Allreduce}\Big(\text{SUM}\big\{\|x_i\|_p^p\big\}\Big))^{1/p}$
3: **for** $i \in \{1, 2, \ldots, n\}$ **do** {In parallel}
4: Compute sign(x_i), $\xi_i(y_i)$
5: **end for**
6: **if** *sparse* **then**
7: **return** $\frac{1}{n}$Allgather(SUM $\{nnz_i, \text{sign}(x_i)[nnz_i], \xi_i(y_i)[nnz_i]\}$)
8: **end if**
9: **return** $\frac{1}{n}$Allreduce(SUM{sign(x_i), $\xi_i(y_i)$})

for $j \in [d]$, where $u_i^j \in \{0, 1, 2, \ldots, s\}$ is such that $l_{u_i^j} \le (y_i)_j \le l_{u_i^j+1}$.

3.2 Algorithm Design

Algorithm 1 provides the pseudocode for Global-QSGD which can be categorized into 3 steps:

Step 1: global normalization (line 1-2) , where each worker computes its local norm and then performs an Allreduce operation to obtain the global norm. This ensures that all workers quantize their gradients consistently, preventing discrepancies due to local scaling differences.

Step 2: quantization (line 3-5) , where each worker normalizes its gradients to $[0, 1]$ and applies stochastic rounding to the nearest quantization intervals l_i for $i \in \{0, 1, \ldots, s\}$.

Step 3: aggregation (line 9) , where the workers perform an Allreduce operation on their quantized values.

Step X: sparsity (line 6-8) . Additionally, we handle sparse gradients by introducing an extra step to identify and transmit only non-zero elements. Instead of sending full quantized vectors, we use Allgather to efficiently communicate the nonzero values.

Figure 1 illustrates the example workflow of the Global-QSGD's two round Allreduce with 2 GPUs.

3.3 Quantization Interval

We formalize the two variants of Global-QSGD:

- Standard Dithering ($Global-\mathcal{L}_s^{q,p}$): A simple way is to divide the interval into equal partitions as $l_i = {}^{s-i}/_s$. However, as the gradients drop to zero during training, this method becomes less accurate since most values fall into the same intervals.

- `Exponential Dithering` ($Global-\mathcal{E}_s^{q,p}$)[2] : To provide higher precision for smaller values, we propose a non-uniform interval partitioning inspired by Horvóth et al. [13], we divide intervals exponentially as $l_s = 0$ and $l_i = 1/2^{s-i}$.

Figure 1 visualizes how both methods quantize a 32-bit floating-point number −0.3 into an 8-bit representation.

The implementation of $Global-\mathcal{L}_s^{q,p}$ is simple and inherently compatible with Allreduce, since mapping to uniform intervals is a holomorphic operation, allowing efficient aggregation by summing integers directly. However, for $Global-\mathcal{E}_s^{q,p}$, summation via Allreduce becomes more complex, as the quantized values follow an exponential form 2^k. To ensure compatibility with Allreduce, we introduce **stochastic unbiased exponential rounding**, denoted as C_{nat}, following the notation in Horvóth et al. [13]. This rounding scheme introduces a minor variance increase per step, specifically a factor of 9/8. Over multiple aggregation steps, this accounts for $(9/8)^{\#\text{ aggregation steps}}$, i.e., $(9/8)^{\log(n)} \leq n^{0.17}$ for Tree-Allreduce, resulting in a small increase in variance—for example, only 1.6× for $n = 16$ and 3.25× for $n = 1024$.

To efficiently implement $Global-\mathcal{E}_s^{q,p}$, we introduce the **exponential reduce** function (Algorithm 2), which aggregates values using integer-based arithmetic, eliminating complex branching and avoiding floating-point operations. The design principles are detailed in Appendix B.

Finally, we discuss a strong advantage of $Global-\mathcal{E}_s^{q,p}$, which is its scaling with respect to the number of nodes n. Let us look at the following example, where we use int$A$ ($A \in \mathbb{N}$) to represent integers with A bits. In the case of linear quantization levels $Global-\mathcal{L}_s^{p}$, the maximum number that we might encounter is ns , plus we need to hold one bit for the sign. Therefore, we require $1+\log(s+1)+\log(n) \leq A$, which cannot be satisfied for any s in the cases $n = 16$ and $A = 4$. On the other hand, the maximum number we can encounter with $Global-\mathcal{E}_s^{p}$ is $n2^s$, but since we only communicate the exponent, we obtain an improved scaling as the maximum communicated integer is $s+\log(n)$. Therefore, we only require $1+\log(s+1+\log(n)) \leq A$, which is satisfied for $s \leq 3$. This is because $Global-\mathcal{E}_s^{p}$ scales as $\log(\log(n))$ with n instead of just $\log(n)$.

4 Convergence Analysis

We provide a rigorous convergence analysis without heuristics. The theoretical framework extends from [8, 14, 16, 27]. In Section 4.1, we extend the concept of *Unbiased Compressors* ($\mathbb{U}^d(\omega)$) to a broader class, *Unbiased Distributed Mean Compressors* ($\mathbb{U}^{n,d}(\theta)$). We then prove that Global-$\mathcal{Q}_s^{q,p} \in \mathbb{U}^{n,d}(\theta)$, establishing its unbiasedness. In Section 4.2, using this unbiased property, we demonstrate that Global-$\mathcal{Q}_s^{q,p}$ has bounded variance, a crucial condition to ensure convergence. All derivation details are provided in Appendix C.

[2]We use base 2 for natural compatibility with floating-point representation.

Algorithm 2 Reduce Function for $Global-\mathcal{E}_s^{q,p}$

Input: $(\text{sign}_1, e_1), (\text{sign}_2, e_2), m \overset{\text{def}}{=} s+1$
$k = -\left\lfloor \log\left((2^{-m} + [p-2^{-m}]_+)\right)\right\rfloor$, where $p \sim \text{Unif}[0,1]^d$ {**can be precomputed**}
e_1_is_not_zero = $\mathbf{1}(e_1 > 0)$
e_2_is_not_zero = $\mathbf{1}(e_2 > 0)$
e_2_is_zero = 1 − e_2_is_not_zero
sign_{12} = $\text{sign}_1\ \text{sign}_2$ e_1_is_not_zero e_2_is_not_zero
diff = $|e_1 - e_2| - (1 - \text{sign}_{12})//2$
leq = $(\mathbf{1}(e_1 \leq e_2) + \text{e\_2\_is\_zero})$ e_1_is_not_zero
non_zero = $1 - \mathbf{1}\big(e_1 = e_2 \text{ and } \text{sign}_{12} = -1\big)$
$\text{sign}_{\text{result}} = \text{sign}_1\, leq + \text{sign}_2(1 - leq)$
e_{result} = $\big(e_1 leq + e_2(1 - leq) - \text{sign}_{12}\,\mathbf{1}(k > \text{diff})\big)$ non_zero
Output: $\text{sign}_{\text{result}}, e_{\text{result}}$

4.1 Unbiasedness

We start by defining the Unbiased Compressor as $\mathbb{U}^d(\omega)$:

Definition 4.1 (Unbiased Compressor)**.** A randomized mapping $C\colon \mathbb{R}^d \to \mathbb{R}^d$ is an *unbiased compressor* if there exists $\omega \geq 0$ such that $\forall x \in \mathbb{R}^d$:

$$\mathbf{E}\left[C(x)\right] = x, \qquad \mathbf{E}\left[\|C(x) - x\|_2^2\right] \leq \omega\|x\|_2^2. \tag{3}$$

If this holds, for simplicity we will write $C \in \mathbb{U}^d(\omega)$.

We now generalize this notion to distributed settings as follows:

Definition 4.2 (Unbiased Distributed Mean Compressor)**.** For any $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$, let

$$\mathbf{x} \overset{\text{def}}{=} [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{nd}, \qquad \bar{\mathbf{x}} \overset{\text{def}}{=} \frac{1}{n}\sum_{i=1}^n x_i\ . \tag{4}$$

A randomized mapping $\mathcal{G}\colon \mathbb{R}^{nd} \to \mathbb{R}^d$ is an *unbiased distributed mean compressor* if there exists $\theta \geq 0$ such that $\forall \mathbf{x} \in \mathbb{R}^{nd}$:

$$\mathbf{E}\left[\mathcal{G}(\mathbf{x})\right] = \bar{\mathbf{x}}, \qquad \mathbf{E}\left[\|\mathcal{G}(\mathbf{x}) - \bar{\mathbf{x}}\|_2^2\right] \leq \frac{\theta}{n}\|\mathbf{x}\|_{2,2}^2. \tag{5}$$

If this holds, for simplicity we will write $\mathcal{G} \in \mathbb{U}^{n,d}(\theta)$.

To show that Definition 4.2 is more general than Definition 4.1, we formalize the following lemma:

Lemma 4.3 ($\mathbb{C} \subset \mathbb{U}^{n,d}$). *If $C_1, C_2, \ldots, C_n \in \mathbb{U}^n(\omega)$ and they are independent, then $\mathcal{G}\colon \mathbb{R}^{nd} \to \mathbb{R}^d$ defined as Equation 6 and belongs to $\mathbb{U}^{n,d}(\omega/n)$.*

$$\mathcal{G}(\mathbf{x}) \overset{def}{=} \frac{1}{n}\sum_{i=1}^{n} C_i(x_i). \tag{6}$$

In the next lemma, we show that Global-$Q_s^{q,p} \in \mathbb{U}^{n,d}(\theta)$ has an interesting reduction property that helps us to analyze its theoretical properties using known results for the case $n = 1$ [3, 13].

Lemma 4.4. *Let $Q_s^{q,p}(\mathbf{x}) \overset{def}{=} \|\mathbf{x}\|_{q,p}\,\mathrm{sign}(\mathbf{x}) \circ \xi(\mathbf{y})$, where $\xi(\mathbf{y}) = [\xi_1(y_1), \xi_2(y_2), \ldots, \xi_n(y_n)]$ and $\xi_i(y_i)$ is defined in (2). Then,*

$$\textit{Global-}Q_s^{q,p}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\left(Q_s^{q,p}(\mathbf{x})\right)_i, \tag{7}$$

where $\left(Q_s^{q,p}(\mathbf{x})\right)_i$ refers to coordinates $[(i-1)d+1, \ldots, id]$. Moreover, if $Q_s^{q,p} \in C(\omega)$ then Global-$Q_s^{q,p} \in \mathbb{U}^{n,d}(\theta)$ with $\theta = \omega/n$.

Note that there is a difference in the dependence on the dimension, i.e., $d \to nd$, since we work with the concatenated vector $\mathbf{x}$.

4.2 Variance Bound

Next, we derive the exact bound on the variance of both $Global - \mathcal{L}_s^{q,p}$ and $Global - \mathcal{E}_s^{q,p}$. In addition, for the special case of $p = q = 2$, we establish an upper bound on sparsity, i.e., the sum of zero norms of the communicated vectors ($\|y\|_0$ denotes the number of non-zero elements of y).

Theorem 4.5. *If $p, q \geq 2$ then Global $- \mathcal{L}_s^{q,p} \in \mathbb{U}^{n,d}\left(\frac{\sqrt{d}}{\sqrt{n}s}\right)$ for $s \leq \sqrt{nd}$, and Global $- \mathcal{E}_s^{q,p} \in \mathbb{U}^{n,d}\left(\frac{1}{8n} + \frac{\sqrt{d}}{\sqrt{n}2^{s-1}}\right)$ for $s \leq 1 + \log\left(\sqrt{nd}\right)$. Moreover, if $p = q = 2$ then for any $\mathbf{x} \in \mathbb{R}^{nd}$*

$$\sum_{i=1}^{n}\left\|\left(\mathcal{L}_s^{2,2}(\mathbf{x})\right)_i\right\|_0 \leq s^2 + \sqrt{nd},$$

$$\sum_{i=1}^{n}\left\|\left(\mathcal{E}_s^{2,2}(\mathbf{x})\right)_i\right\|_0 \leq 2^{2s-2} + \sqrt{nd}.$$

The above theorem guarantees that we can achieve $O(\sqrt{nd})$ compression ratio for $s = O(1)$. Furthermore, we note that the variance bound scales better for the exponential dithering with the number of levels s, which means that the exponential dithering exhibits a smaller relative compression error for larger s. Using these definitions, standard convergence analysis (e.g. [11]) extends naturally to our distributed setting. In particular, employing an unbiased distributed mean compressor $Q \in \mathbb{U}^{n,d}(\theta)$ increases the iteration complexity by a factor of $1 + \theta n$ (recovering $1 + \omega$ for standard compressors). For example, under exponential dithering, this factor becomes $1 + n^{1.17}\left(\frac{1}{8n} + \frac{\sqrt{d}}{\sqrt{n}2^{s-1}}\right)$. Moreover, our analysis is compatible with a variety of optimizers (e.g., SGD, Nesterov momentum, Adam), ensuring that our approach maintains convergence guarantees while reducing communication costs. Our evaluation of Transformer-XL with Adam optimizer also empirically demonstrating that our concept extends seamlessly to optimizers beyond SGD.

5 Performance Model

Gradient compression can only be beneficial if the introduced computation overhead can be compensated by communication gains. Thus, we propose a performance model to analyze when Global-QSGD can speed up the training.

In practice, there are two popular Allreduce implementations: Ring-based and Tree-based. We adopt the Tree-based Allreduce, as it has fewer reduction steps, where each quantized reduction introduces computation overhead and accuracy loss due to random rounding. The tree Allreduce algorithm first recursively aggregates the gradients, then does a recursive-doubling Allgather. The depth of the tree is $2\log(n)$ where n is the size of workers. The performance of Allreduce is well studied [29] and it is commonly modeled as: $2\log(n)\alpha + 2\frac{\log(n)S}{\beta} + \frac{\log(n)S}{\gamma}$, where α is the propagation delay in seconds, S is the size of gradients in bytes, β is the bandwidth (byte/s), and γ is the computation speed (byte/s). The first term represents the propagation delay, the second term is the bidirectional transmission delay, and the last term is the computation cost.

We denote the quantized gradient size as $\hat{S}$, and the computation cost with custom reduction as $\hat{\gamma}$. The bandwidth and propagation delay remain the same. Denote the quantization and dequantization time as δ factor the gradient size. Then the Allreduce performance after applying Global-QSGD is: $2\log(n)\alpha + 2\frac{\log(n)\hat{S}}{\beta} + \frac{\log(n)\hat{S}}{\hat{\gamma}} + \delta S$.

We denote the quantization ratio $\rho = \frac{S}{\hat{S}}$ $(= 4)$ [3]. For the computation cost, we observe that the quantization and dequantization operations are executed only once per Allreduce invocation, thus we consider it negligible ($\delta = 0$). Therefore, the cost of the reduction operation will dominate the performance as the number of workers increases. We denote the computation overhead as $\omega = \frac{\gamma}{\hat{\gamma}}$. We empirically measure[4] this overhead to be $\omega_S = 1$ and $\omega_E = 79$, for standard and exponential dithering, respectively. The reduction operation of standard dithering is the native arithmetic summation,

[3] Our evaluation by default quantizes from Float32 to 8bits . Global-QSGD also works with other bits.

[4] Measured with 25 MB (PyTorch default communication size) data in one A100 GPU.

which has a similar time for uint8 and float32 data types in our experiments. Instead, the custom reduction used for exponential dithering involves more arithmetic operations, which yield a higher computation overhead.

Equation (8) is the condition for Global-QSGD to speed up training throughput. β is the bandwidth (byte/s), γ is the computation speed (byte/s). $\omega = \frac{\gamma}{\hat{\gamma}}$ is the computation overhead, where $\hat{\gamma}$ is the computation speed after applying Global-QSGD. We put the derivation details in Appendix A.4.

$$\begin{cases} \beta > \frac{6\gamma}{(\omega-4)}, & \text{if } (\omega < 4), \\ \beta < \frac{6\gamma}{(\omega-4)}, & \text{if } (\omega > 4). \end{cases} \tag{8}$$

With $\omega_S = 1$, theoretically, standard dithering is guaranteed to speed up training (since $\beta > 0$). In the case of exponential dithering ($\omega_E = 79$), there is a training speed up if the relation $\beta < 0.08\gamma$ holds. Our evaluation is done using A100 GPU (γ = 2 TB/s) with P2P (β = 53.9 GB/s) or SHM (β = 5.4 GB/s). Therefore, both P2P and SHM network fabrics satisfy the speedup condition.

6 Implementation

For ease of use, we implement Global-QSGD with support for both standard dithering $Global - \mathcal{L}_s^{q,p}$ and exponential dithering $Global - \mathcal{E}_s^{q,p}$, which by default uses the ℓ_{inf}-norm ($p = q = \infty$), and the gradients are quantized to 8 bits ($s = 255$). Global-QSGD is theoretically compatible with any bit of precision. We choose 8 bits as a balanced example between throughput and accuracy. We plan to support additional bit configurations. The algorithm is wrapped in a custom Allreduce module integrated with the standard PyTorch DDP module as a hook. Our hook procedure is invoked by PyTorch DDP at the granularity of a gradient bucket, which by default has a size of at least 25 MB. The procedure consists of three steps: Quantization, Allreduce, and Dequantization. In the case of exponential dithering, the Allreduce step uses a custom reduction function. To achieve this, we implement a customized Allreduce algorithm using NCCL's point-to-point asynchronous communication API. We support both ring and tree Allreduce. We develop the quantization, de-quantization, and the custom reduction function in CUDA to optimize GPU performance.

We will release our code as open source. To use our approach, the user simply needs to load a Python module and register our algorithm by invoking the DDP hook API, like: `model.register_comm_hook()`.

7 Evaluation

The aim of our evaluation is to illustrate that our proposal is practical and beneficial in a range of scenarios, including with different bandwidths among GPUs. We focus on measuring the speedup of training throughput in three distinct domains of DL application. Through task-specific metrics on the test set, we show that Global-QSGD does not impair the model's generalization ability. We further illustrate that, compared to standard dithering, exponential dithering achieves better overall performance due to its higher precision and despite its additional overhead due to the custom reduction operation.

Table 3: Summary of Benchmarks Used in This Work

Model	Dataset	Parameter Size	Training Epochs
DeepLight	Tiny Criteo	607,959,381	10
Wide ResNet-101-2	MiniImageNet	126,886,696	90
TransformerXL	WikiText-103	191,950,298	20

Setup. For small-scale experiments, we use one ASUS ESC N4A-E11 server that runs Ubuntu 22.04 with CUDA 11.6, and we use PyTorch 1.13.0. The server is equipped with 4 NVIDIA A100 GPUs, each with 40 GB of RAM. GPUs are peer-to-peer connected by 4 NVlink channels (4th generation). To tease out the effects of bandwidth on training speed, we run experiments with two interconnects: P2P, which employs Nvidia GPU Direct allowing data to transmit via NVLink directly without the interference of CPU and host memory; SHM, which uses the host memory as a middle buffer; thus, the data will transmit through PCIe. The empirically measured bandwidth is 53.9 GB/s and 5.4 GB/s for P2P and SHM, respectively[5]. We also run an additional large-scale experiment with 64 servers each having 1 A100 GPU on Google Cloud Platform (GCP). The servers are connected through a network shared with other users where the bandwidth fluctuates from 200 Mbps to 1.5 Gbps.

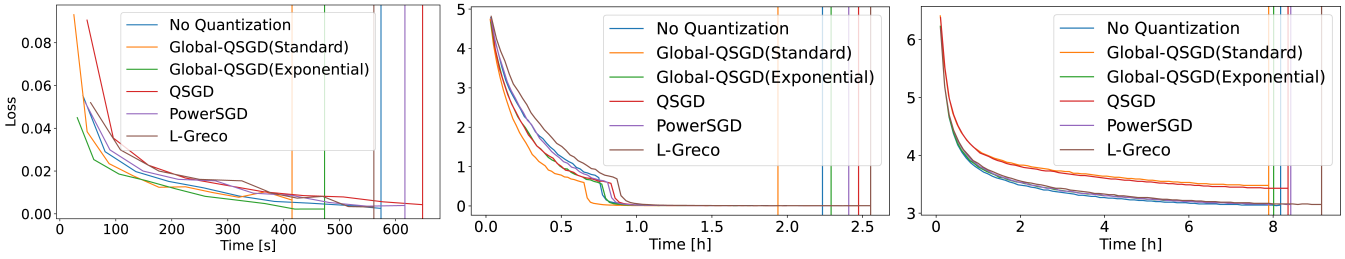
Baselines. Table 3 lists the three DNN models that we use for evaluation. On a small scale, we compare Global-QSGD against the no quantization baseline and with QSGD (baseline for quantization), PowerSGD (baseline for Allreduce compatible compression), and L-Greco (baseline for dynamic compression).[6] On a large scale, we compare Global-QSGD with the PowerSGD baseline as they are Allreduce compatible, while Allgather-based approaches, such as QSGD, will be considerably slower and expensive. **Speedup.** Fig-

Table 4: Validation Metrics After Training

	DeepLight AUC	Wide ResNet-101-2 Top 5	TransformerXL PPL
No Quantization	$6.75 * 10^{-1}$	87.06%	22.991
Global-QSGD(Standard)	$6.95 * 10^{-1}$	88.87%	32.353
Global-QSGD(Exponential)	$6.87 * 10^{-1}$	88.87%	23.678
QSGD	$6.78 * 10^{-1}$	89.39%	30.824
PowerSGD	$6.76 * 10^{-1}$	88.19%	23.364

[5]Measured by NCCL-Test with bucket size 25MB.

[6]QSGD: quantize data to 8 bits; PowerSGD: decompose matrix with rank=32; We keep the other hyper-parameters the same as each method's original repo.



(a) DeepLight **(b) Wide ResNet-101-2** **(c) TransformerXL**

Figure 3: Training Loss Collected With P2P. Vertical Lines Represent the Completion Times.

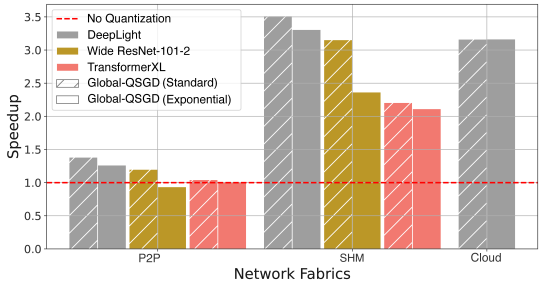


Figure 4: Training Throughput Speedup Normalized to No Quantization Baseline.

ure 4 shows the speedup obtained by Global-QSGD compared to no quantization as the baseline while training different benchmarks on a constant number of epochs. The compression method usually obtains a higher speedup with slower connections. Global-QSGD can achieve up to 3.17× speedup in Cloud, 3.51× with SHM. Even when communicating with P2P through NVLink, which has the highest bandwidth, Global-QSGD can still achieve 1.38× speedup. When applying exponential dithering on Wide ResNet-101-2 with P2P, Global-QSGD delivers robust performance in a model not dominated by communication, with less than 2% overhead to end-to-end training.

Convergence. Figure 3 shows that Global-QSGD can achieve the fastest run time while preserving convergence with P2P. Both standard and exponential dithering achieve good convergence, while striking a different trade-off between time and loss. Specifically, as expected, standard dithering achieves the quickest run time; however, exponential dithering is slightly slower but achieves a better convergence similar to no quantization. In particular, exponential dithering can preserve better convergence than other compressors. Figure 5 shows the DeepLight training in GCP, where the model experiences more communication bottlenecks, allowing compression to achieve greater time savings.

Generalization. We evaluate models with domain-specific metrics after the fixed-iteration training. Table 4 shows that Global-QSGD can maintain the same level of generalizability as no quantization.

8 Future Work

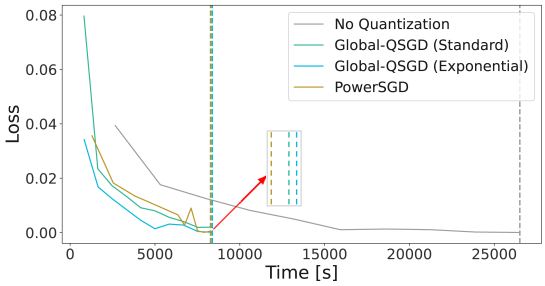


Figure 5: DeepLight Training Loss Collected with 64 Nodes on Google Cloud.

We are still in the process of developing a full framework for Global-QSGD and conducting a comprehensive evaluation. The current prototype is built on NCCL's P2P API, which is less efficient. To achieve a performance comparable to that of NCCL, we will re-implement Allreduce directly in CUDA with NVIDIA GPUDirect. So far, our evaluations have been primarily conducted in a single-node setting; we aim to extend our experiments to multi-node clusters. For the Cloud setting, we do not evaluate all baselines due to budget limits. We only evaluate with 8 bits, and we do not include the models with sparse Allgather. Additionally, our current benchmarks focus on medium-scale models, but testing on large-scale models (e.g., LLMs) will provide a more rigorous evaluation of performance. Our evaluation also lacks sparse scenarios, which we will address by implementing an Allgather operation as described in Algorithm 1. Furthermore, synchronous invocations at each step currently prevent pipelining, limiting opportunities for overlapping compression and communication. We will optimize this process to further improve efficiency.

Acknowledgments

This publication is based upon work supported by the King Abdullah University of Science and Technology Research Funding (KRF) under Award No. ORA-CRG2021-4699.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation* (Savannah, GA, USA) *(OSDI'16)*. USENIX Association, USA, 265–283.

[2] Saurabh Agarwal, Hongyi Wang, Shivaram Venkataraman, and Dimitris Papailiopoulos. 2022. On the Utility of Gradient Compression in Distributed Training Systems. In *Proceedings of Machine Learning and Systems*, D. Marculescu, Y. Chi, and C. Wu (Eds.), Vol. 4. mlsys.org, Santa Clara, CA, USA, 652–672. https://proceedings.mlsys.org/paper_files/paper/2022/file/773862fcc2e29f650d68960ba5bd1101-Paper.pdf

[3] Dan Alistarh, Demjan Grubic, Jerry Z. Li, Ryota Tomioka, and Milan Vojnovic. 2017. QSGD: communication-efficient SGD via gradient quantization and encoding. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 1707–1718.

[4] Dan Alistarh, Torsten Hoefler, Mikael Johansson, Sarit Khirirat, Nikola Konstantinov, and Cédric Renggli. 2018. The convergence of sparsified gradient methods. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal, Canada) *(NIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 5977–5987.

[5] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. 2018. SIGNSGD: Compressed Optimisation for Non-Convex Problems. In *Proceedings of the 35th International Conference on Machine Learning, ICML (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer G. Dy and Andreas Krause (Eds.). PMLR, Stockholmsmässan, Stockholm, Sweden, 559–568. http://proceedings.mlr.press/v80/bernstein18a.html

[6] Adrián Castelló, Mar Catalán, Manuel F. Dolz, José I. Mestre, Enrique S. Quintana-Ortí, and José Duato. 2021. Evaluation of MPI Allreduce for Distributed Training of Convolutional Neural Networks. In *29th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, PDP*. IEEE, Valladolid, Spain, 109–116. doi:10.1109/PDP52278.2021.00025

[7] Adrián Castelló, Mar Catalán, Manuel F Dolz, Enrique S Quintana-Ortí, and José Duato. 2023. Analyzing the impact of the MPI allreduce in distributed training of convolutional neural networks. *Computing* 105, 5 (2023), 1101–1119.

[8] Jean-Baptiste Cordonnier. 2018. *Convex optimization using sparsified stochastic gradient descent with memory*. Technical Report. École Polytechnique Fédérale de Lausanne.

[9] Jeffrey Dean, Greg S. Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, and Andrew Y. Ng. 2012. Large scale distributed deep networks. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1* (Lake Tahoe, Nevada) *(NIPS'12)*. Curran Associates Inc., Red Hook, NY, USA, 1223–1231.

[10] Alexandre Défossez, Leon Bottou, Francis Bach, and Nicolas Usunier. 2022. A Simple Convergence Proof of Adam and Adagrad. *Transactions on Machine Learning Research* 2022, 1 (2022). https://openreview.net/forum?id=ZPQhzTSWA7

[11] Eduard Gorbunov, Filip Hanzely, and Peter Richtarik. 2020. A Unified Theory of SGD: Variance Reduction, Sampling, Quantization and Coordinate Descent. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 108)*, Silvia Chiappa and Roberto Calandra (Eds.). PMLR, Virtual, Sicily, Italy,, 680–690. https://proceedings.mlr.press/v108/gorbunov20a.html

[12] Samuel Horváth and Peter Richtárik. 2021. A Better Alternative to Error Feedback for Communication-Efficient Distributed Learning. In *9th International Conference on Learning Representations, ICLR*. OpenReview.net, Virtual Event, Austria. https://openreview.net/forum?id=vYVI1CHPaQg

[13] Samuel Horvóth, Chen-Yu Ho, Ludovit Horvath, Atal Narayan Sahu, Marco Canini, and Peter Richtarik. 2022. Natural Compression for Distributed Deep Learning. In *Proceedings of Mathematical and Scientific Machine Learning (Proceedings of Machine Learning Research, Vol. 190)*, Bin Dong, Qianxiao Li, Lei Wang, and Zhi-Qin John Xu (Eds.). PMLR, Beijing, China, 129–141. https://proceedings.mlr.press/v190/horvoth22a.html

[14] M. Takáč K. Mishchenko, E. Gorbunov and P. Richtárik. 2024. Distributed learning with compressed gradient differences*. *Optimization Methods and Software* 0, 0 (2024), 1–16. doi:10.1080/10556788.2024.2358790 arXiv:https://doi.org/10.1080/10556788.2024.2358790

[15] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. 2019. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*. PMLR, PMLR, Long Beach, California, USA, 3252–3261.

[16] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. 2019. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International conference on machine learning*. PMLR, PMLR, Long Beach, California, USA, 3478–3487.

[17] Jakub Konečný and Peter Richtárik. 2018. Randomized distributed mean estimation: accuracy vs communication. *Frontiers in Applied Mathematics and Statistics* 4, 62 (2018), 1–11.

[18] Matthias Langer, Zhen He, Wenny Rahayu, and Yanbo Xue. 2020. Distributed training of deep learning models: A taxonomic perspective. *IEEE Transactions on Parallel and Distributed Systems* 31, 12 (2020), 2802–2818.

[19] Mu Li, David G. Andersen, Jun Woo Park, Alexander J. Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J. Shekita, and Bor-Yiing Su. 2014. Scaling distributed machine learning with the parameter server. In *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation* (Broomfield, CO) *(OSDI'14)*. USENIX Association, USA, 583–598.

[20] Shigang Li and Torsten Hoefler. 2022. Near-optimal sparse allreduce for distributed deep learning. In *Proceedings of the 27th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming* (Seoul, Republic of Korea) *(PPoPP '22)*. Association for Computing Machinery, New York, NY, USA, 135–149. doi:10.1145/3503221.3508399

[21] Ilia Markov, Kaveh Alim, Elias Frantar, and Dan Alistarh. 2024. L-GreCo: Layerwise-adaptive Gradient Compression For Efficient Data-parallel Deep Learning. In *Proceedings of Machine Learning and Systems*, P. Gibbons, G. Pekhimenko, and C. De Sa (Eds.), Vol. 6. mlsys.org, Santa Clara, USA, 312–324. https://proceedings.mlsys.org/paper_files/paper/2024/file/9069a8976ff06f6443e7f4172990a580-Paper-Conference.pdf

[22] Taesik Na, Jong Hwan Ko, Jaeha Kung, and Saibal Mukhopadhyay. 2017. On-chip training of recurrent neural networks with limited numerical precision. In *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, Anchorage, AK, USA, 3716–3723. doi:10.1109/IJCNN.2017.7966324

[23] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. 2021. Efficient large-scale language model training on GPU clusters using megatron-LM. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (St. Louis, Missouri) *(SC '21)*. Association for Computing Machinery, New York, NY, USA, Article 58, 15 pages. doi:10.1145/3458817.3476209

[24] Peter Richtárik, Igor Sokolov, and Ilyas Fatkhullin. 2021. EF21: a new, simpler, theoretically better, and practically faster error feedback. In *Proceedings of the 35th International Conference on Neural Information Processing Systems (NIPS '21)*. Curran Associates Inc., Red Hook, NY, USA, Article 335, 13 pages.

[25] Amedeo Sapio, Marco Canini, Chen-Yu Ho, Jacob Nelson, Panos Kalnis, Changhoon Kim, Arvind Krishnamurthy, Masoud Moshref, Dan Ports, and Peter Richtarik. 2021. Scaling Distributed Machine Learning with In-Network Aggregation. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, virtual, 785–808. https://www.usenix.org/conference/nsdi21/presentation/sapio

[26] Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 2014. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs. In *15th Annual Conference of the International Speech Communication Association, INTERSPEECH*, Haizhou Li, Helen M. Meng, Bin Ma, Engsiong Chng, and Lei Xie (Eds.). ISCA, Singapore, 1058–1062. doi:10.21437/INTERSPEECH.2014-274

[27] Sebastian U. Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. 2018. Sparsified SGD with memory. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal, Canada) *(NIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 4452–4463.

[28] Ananda Theertha Suresh, Felix X. Yu, Sanjiv Kumar, and H. Brendan McMahan. 2017. Distributed mean estimation with limited communication. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70* (Sydney, NSW, Australia) *(ICML'17)*. JMLR.org, Sydney NSW Australia, 3329–3337.

[29] Rajeev Thakur, Rolf Rabenseifner, and William Gropp. 2005. Optimization of collective communication operations in MPICH. *The International Journal of High Performance Computing Applications* 19, 1 (2005), 49–66.

[30] Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. 2019. PowerSGD: practical low-rank gradient compression for distributed optimization. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Red Hook, NY, USA, Article 1278, 10 pages.

[31] Hongyi Wang, Scott Sievert, Shengchao Liu, Zachary Charles, Dimitris Papailiopoulos, and Stephen Wright. 2018. Atomo: Communication-efficient learning via atomic sparsification. *Advances in Neural Information Processing Systems (NeurIPS)* 31 (2018), 9850–9861.

[32] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2017. TernGrad: ternary gradients to reduce communication in distributed deep learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 1508–1518.

[33] Cong Xie, Shuai Zheng, Sanmi Koyejo, Indranil Gupta, Mu Li, and Haibin Lin. 2020. Cser: Communication-efficient sgd with error reset. *Advances in Neural Information Processing Systems* 33 (2020), 12593–12603.

A Proofs

In this section, we include complete proofs of the claims made in the main paper.

A.1 Proof of Lemma 4.3

Firstly, we show unbiasedness.

$$\mathbf{E}\left[Q(\mathbf{x})\right] = \mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n} C_i(x_i)\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbf{E}\left[C_i(x_i)\right] \overset{(3)}{=} \frac{1}{n}\sum_{i=1}^{n} x_i = \bar{\mathbf{x}}.$$

For the variance,

$$\mathbf{E}\left[\|Q(\mathbf{x}) - \bar{\mathbf{x}}\|_2^2\right] = \mathbf{E}\left[\left\|\frac{1}{n}\sum_{i=1}^{n} C_i(x_i) - x_i\right\|_2^2\right]$$
$$= \frac{1}{n^2}\sum_{i=1}^{n}\mathbf{E}\left[\|C_i(x_i) - x_i\|_2^2\right] \overset{(3)}{\leq} \frac{\omega}{n}\frac{1}{n}\sum_{i=1}^{n}\|x_i\|_2^2,$$

where the second equality is due to independence and zero mean of each summand.

A.2 Proof of Lemma 4.4

It is easy to that such operator is unbiased since $\mathbf{E}\left[(\xi(y_i))\right] = (\xi(y_i))_j$ by construction. Therefore,

$$\mathbf{E}\left[\text{Global-}Q_s^{q,p}(\mathbf{x})\right] \overset{(1)}{=} \mathbf{E}\left[\|\mathbf{x}\|_{q,p}\frac{1}{n}\sum_{i=1}^{n}\text{sign}(x_i)\circ\xi_i(y_i)\right]$$
$$= \|\mathbf{x}\|_{q,p}\frac{1}{n}\sum_{i=1}^{n}\text{sign}(x_i)\circ\mathbf{E}\left[\xi_i(y_i)\right]$$
$$\overset{(2)}{=} \|\mathbf{x}\|_{q,p}\frac{1}{n}\sum_{i=1}^{n}\text{sign}(x_i)\circ y_i$$
$$= \|\mathbf{x}\|_{q,p}\frac{1}{n}\sum_{i=1}^{n}\text{sign}(x_i)\circ\frac{x_i}{\|\mathbf{x}\|_{q,p}} = \bar{\mathbf{x}}.$$

Furthermore, note that for Global-Q, the local compressors do not belong to $\mathbb{U}^n(\omega)$ for any $\omega > 0$ due to its dependence on $\mathbf{x}$.

To obtain the variance bound, we show that it is sufficient to look at $n = 1$, which corresponds to the unbiased compressor (Definition 4.1) due to the following property.

$$\mathbf{E}\left[\left\|\text{Global-}Q_s^{q,p}(\mathbf{x}) - \bar{\mathbf{x}}\right\|_2^2\right]$$
$$\overset{(1)}{=} \mathbf{E}\left[\left\|\|\mathbf{x}\|_{q,p}\frac{1}{n}\sum_{i=1}^{n}\left(\text{sign}(x_i)\circ\xi_i(y_i) - \frac{x_i}{\|\mathbf{x}\|_{q,p}}\right)\right\|_2^2\right]$$
$$= \frac{1}{n^2}\sum_{i=1}^{n}\mathbf{E}\left[\left\|\|\mathbf{x}\|_{q,p}\left(\text{sign}(x_i)\circ\xi_i(y_i) - \frac{x_i}{\|\mathbf{x}\|_{q,p}}\right)\right\|_2^2\right]$$
$$= \frac{1}{n^2}\mathbf{E}\left[\left\|\|\mathbf{x}\|_{q,p}\,\text{sign}(\mathbf{x})\circ\xi(\mathbf{y}) - \mathbf{x}\right\|_2^2\right]$$
$$= \frac{1}{n^2}\mathbf{E}\left[\left\|Q_s^{q,p}(\mathbf{x}) - \mathbf{x}\right\|_2^2\right],$$

where the second inequality is due to independence of $\{\xi_i\}$. If we can show that $Q_s^{q,p}(\mathbf{x}) \in \mathbb{U}^{1,nd}(\omega)$, then

$$\mathbf{E}\left[\left\|\text{Global-}Q_s^{q,p}(\mathbf{x}) - \bar{\mathbf{x}}\right\|_2^2\right] = \frac{1}{n^2}\mathbf{E}\left[\left\|Q_s^{q,p}(\mathbf{x}) - \mathbf{x}\right\|_2^2\right]$$
$$\leq \frac{\omega}{n^2}\|\mathbf{x}\|_{2,2}^2 = \frac{\omega}{n}\frac{1}{n}\sum_{i=1}^{n}\|x_i\|_2^2,$$

which implies $\theta = \omega/n$.

A.3 Proof of Theorem 4.5

The results provided in the theorem follow the same logic as we use in Lemma 4.4, i.e., our compression is equivalent to applying standard compression to concatenated vector $\mathbf{x} = \mathbf{x} \overset{\text{def}}{=} [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{nd}$.

The rest of the proof uses known one-node results of Alistarh et al. [3], Horvóth et al. [13].

For the variance part, we firstly use the second part of Lemma 4.4, which shows that $Q_s^{q,p} \in C(\omega) \Rightarrow \text{Global-}Q_s^{q,p} \in \mathbb{U}^{n,d}(\theta)$ with $\theta = \omega/n$. Secondly, we apply $n = 1$ results of [13, Theorem 7] and [3, Theorem 3.4] combined with the fact about norm stating that $\|\mathbf{x}\|_{q,p} \leq \|\mathbf{x}\|_{2,2}$ for $p, q \geq 2$.

The claim about sparsity with $p = q = 2$ follows from the first part of Lemma 4.4, which implies that the number of non-zero elements of Global-$Q_s^{q,p}$ before aggregation is the same as $Q_s^{q,p}$. The rest of the proof for $\mathcal{L}_s^{2,2}$ follows directly from [3, Lemma 3.1]. For $\mathcal{E}_s^{2,2}$, one can also directly apply [3, Lemma 3.1] using the fact that the length of the first segment $[l_0, l_1]$ is $1/2^{s-1}$.

A.4 Proof for Equation 8

The condition of performance gain is per-batch training time after applying Global-QSGD is less than the per-batch training time without quantization, which is:

$$2\log(N)\alpha + 2\frac{\log(N)S}{\beta} + \frac{\log(N)S}{\gamma} >$$
$$2\log(N)\alpha + 2\frac{\log(N)\hat{S}}{\beta} + \frac{\log(N)\hat{S}}{\hat{\gamma}} + \delta S$$
$$2\frac{S}{\beta} + \frac{S}{\gamma} > 2\frac{\hat{S}}{\beta} + \frac{\hat{S}}{\hat{\gamma}}$$
$$2\frac{\rho}{\beta} + \frac{\rho}{\gamma} > 2\frac{1}{\beta} + \frac{1}{\hat{\gamma}}$$
$$2\rho\gamma\hat{\gamma} + \rho\beta\hat{\gamma} > 2\gamma\hat{\gamma} + \beta\gamma$$
$$\rho\beta\hat{\gamma} - \beta\gamma > 2\gamma\hat{\gamma} - 2\rho\gamma\hat{\gamma}$$
$$\beta(\rho\hat{\gamma} - \gamma) > 2\gamma\hat{\gamma} - 2\rho\gamma\hat{\gamma}$$

Since $\hat{\gamma}$ is the computation speed (byte/s), so $\hat{\gamma} \geq 0$. We divide both sides of the inequality by $\hat{\gamma}$, and replace $\omega = \frac{\gamma}{\hat{\gamma}}$:

$$\beta(\rho - \omega) > 2\gamma - 2\rho\gamma$$
$$\beta(\rho - \omega) > 2\gamma(1 - \rho)$$

Then we have:

$$\begin{cases} \beta > \frac{2\gamma(1-\rho)}{(\rho-\omega)}, & \text{if } (\rho > \omega), \\ \beta < \frac{2\gamma(1-\rho)}{(\rho-\omega)}, & \text{if } (\rho < \omega). \end{cases}$$

B Reduce function for $Global - \mathcal{E}_s^{q,p}$

Since our reduction function acts element-wise, we only consider here a one-dimensional case. Before calling the compressor and the reduce function, we divide all values by $2n$, where n is the number of nodes, to ensure that the maximum power of two we encounter during the aggregation is -1 that corresponds to $1/2 = 2^{-1}$. Therefore, all the encountered exponents during the aggregation are guaranteed to be negative. This way, we can dedicate exponent zero to the actual zero instead of 2^0. Next, we define the representation, which we use during the aggregation when calling the reduce function after applying exponential dithering locally. Let $\text{sign} \in \{-1, +1\}$[7] be the sign and $e \in \{\mathcal{N}^+ \cup \{0\}\}$ be the communicated non-negative integer-valued exponents. Then, then real number x that corresponds to the pair (sign, e) is defined as

$$x = \begin{cases} 0, & \text{if } e = 0, \\ \text{sign}\, 2^{-e}, & \text{otherwise.} \end{cases} \tag{9}$$

We proceed with the derivation for the reduce function. Let $x_1, x_2 \in \mathbb{R}$ represented by $(\text{sign}_1, e_1), (\text{sign}_2, e_2)$, respectively, be the values to be summed using reduce function. To facilitate efficient rounding, let us define

$$k = -\left\lfloor \log\left(2^{-m} + [p - 2^{-m}]_+\right) \right\rfloor,$$

where $p \sim \text{Unif}[0, 1]$ is a sample from the uniform distribution on the interval $[0, 1]$, $m \overset{\text{def}}{=} s + 1$ is the maximum difference $|e_1 - e_2|$ that can appear during the aggregation, and $[x]_+ \overset{\text{def}}{=} \max\{0, x\}$. Note that the support set for k is $\{0, 1, \ldots, m\}$, and for $b \in \{0, 1, \ldots, m-1\}$, it holds

$$\text{Prob}(k > b) = \text{Prob}(-k < -b) = \text{Prob}\left(p < 2^{-b}\right) = 2^{-b}.$$

Without loss of generality, we assume that $\text{sign}_1 = 1$ and $0 < e_1 \geq e_2$. We discuss how to handle the case e_1 at the end of this section. If $\text{sign}_2 = 1$, then

$$C_{\text{nat}}(2^{-e_1} + 2^{-e_2}) = \begin{cases} 2^{-e_1+1}, & \text{w.p. } 2^{e_1-e_2}, \\ 2^{-e_1}, & \text{w.p. } 1 - 2^{e_1-e_2}, \end{cases}$$

where w.p. stands for "with probability." We note that

$$p < 2^{e_1-e_2} \iff k > e_2 - e_1.$$

Therefore,

$$C_{\text{nat}}(2^{-e_1} + 2^{-e_2}) = \begin{cases} 2^{-e_1+1}, & \text{if } k > e_2 - e_1, \\ 2^{-e_1}, & \text{otherwise.} \end{cases}$$

Analogically, when $\text{sign}_2 = -1$, we write $C_{\text{nat}}(2^{-e_1} - 2^{-e_2})$ as:

$$\begin{cases} 0, & \text{if } e_1 = e_2 \\ \begin{cases} 2^{-e_1-1}, & \text{w.p. } 2^{e_2-e_1-1}, \\ 2^{-e_1}, & \text{w.p. } 1 - 2^{e_2-e_1-1}, \end{cases} & \text{otherwise.} \end{cases}$$

Equivalently, we can write $C_{\text{nat}}(2^{-e_1} - 2^{-e_2})$ as:

$$\begin{cases} 0, & \text{if } e_1 = e_2 \\ \begin{cases} 2^{-e_1-1}, & \text{w.p. } k > e_2 - e_1 - 1, \\ 2^{-e_1}, & \text{w.p. } k > e_2 - e_1 - 1, \end{cases} & \text{otherwise.} \end{cases}$$

In general case, we compare k to the following quantity

$$\text{diff} = |e_1 - e_2| - (1 - \text{sign}_{12})//2,$$

where $\text{sign}_{12} \overset{\text{def}}{=} \text{sign}_1\, \text{sign}_2 \in \{-1, 1\}$ and $//$ corresponds to the integer division. It is easy to check that the above quantity recovers all the above mentioned cases. To determine which exponent is smaller, we use $leq = \mathbf{1}(e_1 \leq e_2) \in \{0, 1\}$, where $\mathbf{1}$ is the indicator function. Finally, to filter out the case of the different signs and the same exponent, we use $\text{non\_zero} \overset{\text{def}}{=} 1 - \mathbf{1}\big(e_1 = e_2 \text{ and } \text{sign}_{12} = -1\big) \in \{0, 1\}$. Then, we obtain the resulting sign and exponent as

$$\text{sign}_{\text{result}} = \text{sign}_1\, leq + \text{sign}_2(1 - leq)$$
$$e_{\text{result}} = \big(e_1 leq + e_2(1 - leq) - \text{sign}_{12}\, \mathbf{1}(k > \text{diff})\big)\, \text{non\_zero}\,.$$

To incorporate zeros, we first define the following variables to identify zeros

$$\text{e\_1\_is\_not\_zero} = \mathbf{1}(e_1 > 0)$$
$$\text{e\_2\_is\_not\_zero} = \mathbf{1}(e_2 > 0)$$
$$\text{e\_2\_is\_not\_zero} = 1 - \text{e\_2\_is\_not\_zero}\,.$$

In the case of at least one of the exponents being zero, we do not change the exponents by adding or subtracting zero. This can be achieved by redefine $\text{sign}_{12} \in \{-1, 0, 1\}$ to

$$\text{sign}_{12} \overset{\text{def}}{=} \text{sign}_1\, \text{sign}_2\, \text{e\_1\_is\_not\_zero}\, \text{e\_2\_is\_not\_zero}\,.$$

Furthermore, we need to redefine leq to account for zeros. This can be achieved by

$$leq \overset{\text{def}}{=} (\mathbf{1}(e_1 \leq e_2) + \text{e\_2\_is\_zero})\, \text{e\_1\_is\_not\_zero}\,.$$

This concludes our construction of the reduce function for the exponential dithering.

[7] Zeros have any sign.

C Convergence Analysis

In this section, we showcase how the standard analysis for unbiased compressors can be generalized to the proposed unbiased distributed mean compressor. Since we construct compressors to be unbiased, the only extra challenge of the analysis is to bound the variance of the gradient estimator[8]. In Lemma C.1, we show that applying the unbiased distributed mean compressor $Q \in \mathbb{U}^{n,d}(\theta)$ yields the same bound on variance as the difference that ω/n is replaced with θ. The rest of the analysis is the same as the general analysis of Gorbunov et al. [11], which shows that the worst-case increase in the number of iterations to achieve the same precision as the algorithm without compression is by the factor $1+\omega$ that translates to $1+\theta n$ for the unbiased distributed mean compressors $Q \in \mathbb{U}^{n,d}(\theta)$. [11, Theorem 4.1 for Alg. 1: SGD and Alg. 13: Quantized-SGD] For our exponential dithering, this would be a factor of $1+n^{1.17}\left(1/8n+\sqrt{d}/\sqrt{n}2^{s-1}\right)$. In this estimate, we consider all the sources of variance for Allreduce, i.e., variance due to initial quantization and extra variance due to stochastic rounding during Allreduce. For example, let us consider the setup, where we use exponential dithering to decrease communication precision from 32 to 8 bits, the dimensionality of the model is $n=10^6$, and the number of machines is $n=16$. We have to reserve 1 bit to the sign and the other 7 bits to levels. Plugging these numbers into our formula yields the worst-case increment in the expected number of iterations, which is 20%, while we save 75% of communication. Therefore, if communication is a significant bottleneck, our quantization leads to guaranteed speed-up. Note that this is only the worst case, and in real-world scenarios, this can be much less, e.g., the empirical compression error in our experiments is below 0.5% for exponential dithering.

Furthermore, while our analysis primarily focuses on SGD, the concept of global quantization is intentionally designed to be broadly compatible with various optimizers, including popular ones like Nesterov momentum and Adam. This compatibility stems from its core property of being unbiased with bounded variance, aligning it with the theoretical foundations underlying these optimizers. Typically, optimizers such as SGD are predicated on the use of stochastic gradients that are unbiased and maintain variance within specific limits—conditions supported by our method. For instance, the integration with adaptive methods can be directly obtained by generalizing the results of Défossez et al. [10].

Lemma C.1 (Variance bound). *Let $x \in \mathbb{R}^d$ be deterministic and $\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\nabla f_i(x,\xi_i)\right] = \nabla f_i(x)$ for all $i \in [n]$. Then*

$$\mathbf{E}\left[\|Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)]) - \nabla f(x)\|_2^2\right] \leq \frac{\rho}{n}\sum_{i=1}\|\nabla f_i(x))\|_2^2 + \left(\rho+\frac{1}{n}\right)\frac{1}{n}\sum_{i=1}\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)-\nabla f_i(x)\|_2^2\right] \tag{10}$$

holds with $\rho = \frac{\omega}{n}$ for $Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)]) \overset{def}{=} \frac{1}{n}\sum_{i=1} C_i(\nabla f_i(x,\xi_i))$, where $C_1, C_2, \ldots, C_n \in \mathbb{U}^n(\omega)$ are independent. Furthermore, if $Q \in \mathbb{U}^{n,d}(\theta)$ then (10) holds with $\rho = \theta$.

Proof. We assume that the noise due to sampling and compression are independent. For the first case, we have

$$\begin{aligned}
&\mathbf{E}\left[\left\|\frac{1}{n}\sum_{i=1} C_i(\nabla f_i(x,\xi_i)) - \nabla f(x)\right\|_2^2\right] \\
&= \frac{1}{n^2}\sum_{i=1}\mathbf{E}\left[\|C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x)\|_2^2\right] + \frac{1}{n^2}\sum_{i\neq j}\mathbf{E}\left[\left\langle C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x), C_j(\nabla f_j(x,\xi_i)) - \nabla f_j(x)\right\rangle\right] \\
&= \frac{1}{n^2}\sum_{i=1}\mathbf{E}\left[\|C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x)\|_2^2\right] + \frac{1}{n^2}\sum_{i\neq j}\left\langle\mathbf{E}\left[C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x)\right], \mathbf{E}\left[C_j(\nabla f_j(x,\xi_i)) - \nabla f_j(x)\right]\right\rangle.
\end{aligned}$$

Using tower property, we get

$$\begin{aligned}
\mathbf{E}\left[C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x)\right] &= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x)|\xi_i\right]\right] \\
&= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\nabla f_i(x,\xi_i) - \nabla f_i(x)\right] = 0.
\end{aligned}$$

[8]The same applies to the analysis of non-smooth functions, where gradients are replaced with subgradients.

Furthermore, using the same technique and the unbiasedness and bounded variance of the compression operator yields

$$
\begin{aligned}
&\mathbf{E}\left[\|C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x)\|_2^2\right] = \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[\|C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x)\|_2^2|\xi_i\right]\right] \\
&= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[\|C_i(\nabla f_i(x,\xi_i))\|_2^2|\xi_i\right]\right] - 2\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[\langle C_i(\nabla f_i(x,\xi_i)), \nabla f_i(x)\rangle\,|\xi_i\right]\right] + \|\nabla f_i(x)\|_2^2 \\
&= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[\|C_i(\nabla f_i(x,\xi_i))\|_2^2|\xi_i\right]\right] - \|\nabla f_i(x)\|_2^2 \\
&= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[\|C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x,\xi_i) + \nabla f_i(x,\xi_i)\|_2^2|\xi_i\right]\right] - \|\nabla f_i(x)\|_2^2 \\
&= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[\|C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x,\xi_i)\|_2^2|\xi_i\right]\right] + \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)\|_2^2\right] - \|\nabla f_i(x)\|_2^2 \\
&\quad + 2\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[\langle C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x,\xi_i), \nabla f_i(x,\xi_i)\rangle\,|\xi_i\right]\right] \\
&= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[\|C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x,\xi_i)\|_2^2|\xi_i\right]\right] + \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)\|_2^2\right] - \|\nabla f_i(x)\|_2^2 \\
&\quad + 2\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\left\langle \mathbf{E}_{C_i}\left[C_i(\nabla f_i(x,\xi_i))\right] - \nabla f_i(x,\xi_i), \nabla f_i(x,\xi_i)\right\rangle|\xi_i\right] \\
&= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\mathbf{E}_{C_i}\left[\|C_i(\nabla f_i(x,\xi_i)) - \nabla f_i(x,\xi_i)\|_2^2|\xi_i\right]\right] + \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)\|_2^2\right] - \|\nabla f_i(x)\|_2^2 \\
&\quad + 2\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\langle 0, \nabla f_i(x,\xi_i)\rangle\,|\xi_i\right] \\
&\leq (\omega+1)\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)\|_2^2\right] - \|\nabla f_i(x)\|_2^2 \\
&= (\omega+1)\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i) - \nabla f_i(x) + \nabla f_i(x)\|_2^2\right] - \|\nabla f_i(x)\|_2^2 \\
&= (\omega+1)\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i) - \nabla f_i(x)\|_2^2\right] + 2(\omega+1)\left\langle \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\nabla f_i(x,\xi_i)\right] - \nabla f_i(x), \nabla f_i(x)\right\rangle + \omega\|\nabla f_i(x)\|_2^2 \\
&= (\omega+1)\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i) - \nabla f_i(x)\|_2^2\right] + \omega\|\nabla f_i(x)\|_2^2,
\end{aligned}
$$

which concludes the first part of the proof. For the second part, we proceed analogously and have that for $Q \in \mathbb{U}^{n,d}(\theta)$

$$
\begin{aligned}
&\mathbf{E}\left[\|Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)]) - \nabla f(x)\|_2^2\right] \\
&= \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\mathbf{E}_Q\left[\left\|Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)]) - \frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i) - \nabla f(x)\right\|_2^2 |\{\xi_i\}_{i=1}^n\right]\right] \\
&= \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\mathbf{E}_Q\left[\left\|Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)]) - \frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i)\right\|_2^2 |\{\xi_i\}_{i=1}^n\right]\right] \\
&\quad + \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i) - \nabla f(x)\right\|_2^2\right] \\
&\quad + \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\left\langle \mathbf{E}_Q\left[Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)])|\{\xi_i\}_{i=1}^n\right] - \frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i), \frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i) - \nabla f(x)\right\rangle\right].
\end{aligned}
$$

The first element of the scalar product is zero since Q is unbiased. Therefore

$$
\begin{aligned}
&\mathbf{E}\left[\|Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)]) - \nabla f(x)\|_2^2\right] \\
&= \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\mathbf{E}_Q\left[\left\|Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)]) - \frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i)\right\|_2^2 |\{\xi_i\}_{i=1}^n\right]\right] \\
&\quad + \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i) - \nabla f(x)\right\|_2^2\right] \\
&\leq \frac{\theta}{n}\sum_{i=1}^n \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)\|_2^2\right] + \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i) - \nabla f(x)\right\|_2^2\right].
\end{aligned}
$$

$$\text{For} \quad \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i)-\nabla f(x)\right\|_2\right] \quad \text{we have} \quad \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i)-\nabla f(x)\right\|_2^2\right]$$

$$= \frac{1}{n^2}\sum_{i=1}^n \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)-\nabla f_i(x)\|_2^2\right] + \frac{1}{n^2}\sum_{i\neq j}\left\langle \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\nabla f_i(x,\xi_i)\right]-\nabla f_i(x), \mathbf{E}_{\xi_j\sim\mathcal{D}_j}\left[\nabla f_j(x,\xi_j)\right]-\nabla f_j(x)\right\rangle$$

$$= \frac{1}{n^2}\sum_{i=1}^n \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)-\nabla f_i(x)\|_2^2\right]$$

Finally, for the $\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)\|_2^2\right]$, we have

$$\begin{aligned}\mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)\|_2^2\right] &= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)-\nabla f_i(x)+\nabla f_i(x)\|_2^2\right]\\ &= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)-\nabla f_i(x)+\nabla f_i(x)\|_2^2\right] + \|\nabla f_i(x)\|_2^2 + 2\left\langle \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\nabla f_i(x,\xi_i)\right]-\nabla f_i(x), \nabla f_i(x)\right\rangle\\ &= \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)-\nabla f_i(x)+\nabla f_i(x)\|_2^2\right] + \|\nabla f_i(x)\|_2^2.\end{aligned}$$

Putting them all together

$$\begin{aligned}&\mathbf{E}\left[\|Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)])-\nabla f(x)\|_2^2\right]\\ &\leq \frac{\theta}{n}\sum_{i=1}^n \mathbf{E}_{\xi_i\sim\mathcal{D}_i}\left[\|\nabla f_i(x,\xi_i)\|_2^2\right] + \mathbf{E}_{\{\xi_i\sim\mathcal{D}_i\}_{i=1}^n}\left[\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(x,\xi_i)-\nabla f(x)\right\|_2^2\right]\\ &= \frac{\theta}{n}\sum_{i=1}^n \mathbf{E}\left[\|\nabla f_i(x))\|_2^2\right] + \left(\theta+\frac{1}{n}\right)\frac{1}{n}\sum_{i=1}^n \mathbf{E}\left[\|\nabla f_i(x,\xi_i)-\nabla f_i(x)\|_2^2\right]\end{aligned}$$

yields the desired bound. □

For the adaptive methods, we show that the assumptions in Section 2.3 of [10] remain applicable when combining stochastic gradients with global quantization. The only assumption related to the stochastic gradients is that the ℓ_∞ norm of the stochastic gradients is almost surely bounded.

In our case, the stochastic estimator has the following form

$$Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)]).$$

We assume that $\|\nabla f_i(x,\xi_i)\|_\infty \leq R$, for all $i\in[n]$. Furthermore, for both standard and exponential global quantization, we have

$$\begin{aligned}\|Q([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)])\|_\infty &= \left\|\frac{1}{n}\sum_{i=1}^n \left(Q_s^{q,p}([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)])\right)_i\right\|_\infty\\ &\leq \frac{1}{n}\sum_{i=1}^n \left\|\left(Q_s^{q,p}([\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)])\right)_i\right\|_\infty\\ &\leq \|[\nabla f_1(x,\xi_1),\ldots,\nabla f_n(x,\xi_n)]\|_{p,q}\\ &= \left(\sum_{i=1}^n \|\nabla f_i(x,\xi_i)\|_q^p\right)^{1/p}\\ &\leq \left(\sum_{i=1}^n (d\|\nabla f_i(x,\xi_i)\|_\infty)^{p/q}\right)^{1/p} \leq n^{1/p}d^{1/q}R.\end{aligned}$$

Therefore, global quantization preserves the bound of the ℓ_∞ norm of the stochastic gradients up to constant. This implies that all the convergence guarantees of Défossez et al. [10] also apply to our setting with global quantization.

To sum up, global quantization is compatible with standard optimization methods designed for efficient distributed learning, such as distributed variants of Adam and SGD. By maintaining unbiased gradient estimates with bounded variance, our quantization method preserves convergence guarantees comparable to those of uncompressed methods while significantly reducing communication costs.