Unbiased Quantization of the L_1 Ball for Distributed Mean Estimation

Nithish Suresh Babu **Ritesh Kumar** **Shashank Vatedka**
Department of Electrical Engineering
Indian Institute of Technology Hyderabad, Telangana, India

Abstract

We study the problem of unbiased minimum mean squared error quantization of the L_1 ball, with applications to distributed mean estimation and federated learning. Inspired by quantization of probability distributions using types, we design a novel computationally efficient unbiased quantization scheme for vectors that lie within the L_1 ball. We also derive upper bounds on the worst-case mean squared error achieved by our scheme and show that this is order optimal. We then use this to design polynomial (in the dimension of the input vectors)-time schemes for communication-efficient distributed mean estimation and distributed/federated learning, and demonstrate its effectiveness using simulations.

1 INTRODUCTION

Vector quantization Gray (1984) plays an important role in data storage, machine learning and signal processing, where large datasets need to be represented using a small number of bits. Vector quantization also allows us to reduce communication costs, which is a fundamental bottleneck in large-scale distributed systems. It has been widely used in applications such as image compression, pattern recognition, and speech coding, where preserving essential features while minimizing data redundancy is crucial.

In recent years, vector quantization has become increasingly relevant in communications Cover and Thomas (1999), distributed systems, model compression Polino et al. (2018), and federated learning Kairouz et al. (2021), and a number of novel solutions have been proposed. Vector quantization is also a fundamental component of solutions for communication-constrained distributed mean estimation (DME) and federated learning, which is the focus of this paper.

The distributed mean estimation (DME) problem (see, e.g., Suresh et al. (2017)) is one where there are n users, each having a d-dimensional vector, and a central server, who wants to compute the mean/average of the n vectors. However, the communication links from the users to the server are rate limited, in the sense that number of bits that can be sent by each user is very small (typically $O(d)$). In recent years, DME algorithms have been used as primitives in distributed optimization, e.g., Alistarh et al. (2017); Acharya et al. (2021); Mayekar et al. (2023), transmission of the gradients in each round in federated learning in Suresh et al. (2022); Kairouz et al. (2021); Vargaftik et al. (2022); Ben-Basat et al. (2024); Davies et al. (2021); Konecnỳ and Richtárik (2018), and acceleration or variance reduction mechanisms for federated learning in Richtárik et al. (2021); Caldas et al. (2018); Richtárik et al. (2022); Basu et al. (2019); Condat et al. (2022); Condat and Richtárik (2022); Tyurin and Richtárik (2024); Horváth et al. (2023); He et al. (2024), to name a few. See Ben-Basat et al. (2024) for an excellent survey and further references.

The main contributions of our work can be summarized as follows:

- We reduce the problem of distributed mean estimation to one of quantizing the L_1 ball, and further to quantizing probability vectors.
- We design a novel polynomial-time unbiased quantization scheme for probability vectors, and hence the L_1 ball.

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

[0]The work of Shashank Vatedka was supported by a Core Research Grant CRG/2022/004464 from SERB, India.

Quantization of probability vectors is a fundamental problem that has been studied previously in, e.g., Graf and Luschgy (2000); Gagie (2006); Reznik (2011). This has many applications, including image and pattern recognition, where histograms of gradients are extracted from images, quantized, and used to find the best match for the correct image or pattern. See Lowe (2004); Chandrasekhar et al. (2012); Bay et al. (2006). Although our main focus is distributed mean estimation and federated learning, our quantizer can be used in these applications as well.

- Our quantizer is very efficient and works in time $O(d)$, where d is the dimension of the input vector. For efficient communication, encoding requires us to represent the quantized vector in binary, which can be performed in poly(d) time. We propose a simple polynomial-time solution based on enumeration of types for binary representation, and believe that a more efficient algorithm can be designed.

- Our scheme achieves order-optimal worst-case mean squared error for vectors with bounded L_2 norm, and state-of-the-art normalized mean squared error (NMSE) for distributed mean estimation Vargaftik et al. (2022); Ben-Basat et al. (2024).

- The best-known algorithms for DME in Suresh et al. (2017); Vargaftik et al. (2022); Ben-Basat et al. (2024) require shared randomness between the users and the server, which is an additional resource. In comparison, we achieve nearly the same worst-case NMSE with no shared randomness required. This is one of the major contributions of this work.

- Through simulations, we demonstrate that our algorithm achieves state-of-the-art performance in distributed mean estimation and federated learning.

2 PRIOR WORKS

The distributed mean estimation (DME) problem in the setting of the present paper was first studied by Suresh et al. (2017), who analyzed various schemes for DME that achieve low mean squared error. One of their solutions was to use random rotations (either using a uniform random rotation matrix or a randomized Hadamard matrix) followed by scalar quantization, which remains a popular approach for DME even today.

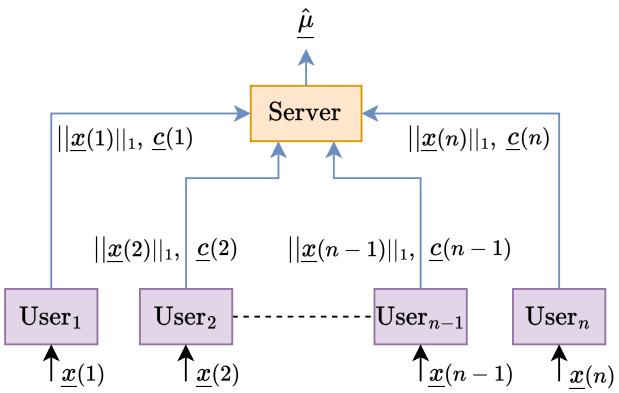


Figure 1: An illustration of our scheme for DME. Each user employs the encoder of Algorithm 1, and the server uses the corresponding decoder to compute an estimate $\hat{\underline{\mathbf{x}}}(i)$ corresponding to each user i. The final estimate $\hat{\underline{\mu}} = \frac{1}{n}\sum_{i=1}^{n} \hat{\underline{\mathbf{x}}}(i)$.

Vargaftik et al. (2022) improved on this approach, and showed that a uniform random rotation along with a carefully designed scalar Lloyd-Max quantizer and suitable scaling can yield an unbiased quantizer that achieves $O(1)$ normalized mean squared error (NMSE) for communication rates[1] that scale as $\Theta(1)$ (as a function of d). However, this requires generation of an independent uniform random rotation matrix for each user that is known to the server, and therefore requires an infinite amount of shared randomness. This is not practical. They instead suggested using a randomized Hadamard matrix as a replacement for the uniform random rotation, and derived bounds on the NMSE. More recently, Ben-Basat et al. (2024) gave an improved computationally efficient scheme using randomized Hadamard rotation matrices and a novel unbiased quantizer, showed that their scheme is unbiased and achieves $O(1)$ NMSE at $\Theta(1)$ rates. However, their solution also requires shared randomness. Safaryan et al. (2022) used a different approach to design an unbiased quantizer based on Kashin's representation, but this has a higher computational complexity. Improved algorithms (with non-identical quantizers at the users) that use shared random rotation matrices were proposed by Mayekar et al. (2023) for distributed optimization.

Suresh et al. (2022) showed that if all clients and the server share a common random string (as opposed to a different string per client, also known to the server), then it is possible to achieve even lower NMSE. However, this requires significantly higher amounts of shared randomness.

[1]Here, the rate of communication is the total number of bits that a user can transmit, divided by d.

In this work, we use a totally different approach and design a *polynomial-time unbiased* quantizer that achieves $O(1)$ NMSE for the DME problem at $\Theta(1)$ communication rates *without using any shared randomness*. Our quantizer runs in $O(d)$ time, but the encoding and decoding of quantized vectors in binary requires slightly higher complexity.

We do so by sequentially reducing the DME problem to one of designing an efficient unbiased quantizer for the L_1 ball, and then an efficient unbiased quantizer for probability vectors.

A comparison of our results with the state-of-the-art can be found in Table 1.

Perhaps most relevant is the work of Reznik (2011), who designed deterministic *biased* type-based quantizers for probability vectors. Types (see the next section for a definition) are fundamental objects studied in information theory and have been heavily used to derive theorems for source and channel coding. See Cover and Thomas (1999); Csiszár and Körner (2011). In comparison, we design a novel unbiased type-based quantizer using a different approach and also show how this can be used for DME.

3 NOTATION

The set of nonnegative integers is denoted $\mathbb{Z}_+$, while the set of nonnegative reals is denoted $\mathbb{R}_+$. Underlined symbols represent vectors (e.g., $\underline{u}, \underline{\mathbf{u}}$). We use boldface to denote random variables (e.g., $\mathbf{u}, \mathbf{v}$) and random vectors (e.g., $\underline{\mathbf{u}}$), and normal font to denote deterministic scalars (u, v) and vectors $(\underline{u}, \underline{v})$. Matrices are denoted in upper case, e.g., G, H. The all-ones vector is denoted $\underline{\mathbf{1}}$ and the all-zeros vector is denoted $\underline{\mathbf{0}}$. For a real number a, the fractional part of a is defined as $\{a\} = a - \lfloor a \rfloor$, where $\lfloor a \rfloor$ is the largest integer that less than or equal to a. For any positive integer n, the set $\{1, 2, \ldots, n\}$ is denoted by $[n]$.

Unless otherwise mentioned, all vectors will be assumed to lie in $\mathbb{R}^d$. The unit L_p ball in $\mathbb{R}^d$ is denoted $\mathcal{B}_{L_p}$. The set of all probability vectors in $\mathbb{R}^d$ is $\Delta^d = \{\underline{x} \in \mathbb{R}_+^d : \sum_{i=1}^d x_i = 1\}$. For a positive integer m, the set of m-types is $\mathbb{P}_m = \{\frac{1}{m}\underline{x} \in \frac{1}{m}\mathbb{Z}_+^d : \sum_{i=1}^d x_i = m\}$.

4 COMMUNICATION-CONSTRAINED DISTRIBUTED MEAN ESTIMATION

The distributed mean estimation (DME) problem can be described as follows: There are n users, each having a d-dimensional vector with real entries. Assume that user $i \in [n]$ has $\underline{x}(i)$. There is a central server who wishes to compute (or at least approximate) the empirical mean of these vectors, i.e., $\underline{\mu} \triangleq \frac{1}{n}\sum_{i=1}^n \underline{x}(i)$. However, the users cannot communicate their respective vectors directly but are instead allowed to only send a fixed number of bits to the server. We will assume that each user is allowed to send dR bits to the central server, for some $R > 0$ (which we call the communication rate). The server receives the $n \times dR$ bits, and outputs an estimate $\hat{\underline{\mu}}$. The goal is to design a protocol/scheme Π adhering to the above constraints while ensuring that the worst-case normalized mean squared error

$$\text{NMSE}(\Pi) = \sup_{\underline{x}(1),\ldots,\underline{x}(n)\in\mathcal{B}_{L_2}} \frac{\mathbb{E}[\|\hat{\underline{\mu}} - \underline{\mu}\|_2^2]}{\frac{1}{n}\sum_{i=1}^n \|\underline{x}(i)\|_2^2}$$

is as small as possible. Here, the expectation is taken over all the (shared and private) randomness in the protocol.

DME serves as a primitive in various problems of optimization (including distributed SGD Alistarh et al. (2017) and federated learning McMahan et al. (2017); Kairouz et al. (2021)) and all our results can be translated to these settings as well.

As is standard in most works Safaryan et al. (2022); Vargaftik et al. (2022); Ben-Basat et al. (2024) we assume that each user is allowed to send a single real number along with the dR bits. This additional overhead is typically small, particularly when d is very large. One of the most popular approaches towards communication-efficient DME is to normalize and then quantize the vectors. See, e.g, Safaryan et al. (2022); Vargaftik et al. (2022); Ben-Basat et al. (2024). The norm (which is a real number) is typically sent along with the quantized/encoded version of $\underline{x}(i)$. Most works also assume the existence of shared randomness between each user and the server (which is an additional resource). Each user employs an identical scheme, and the server first reconstructs/finds an estimate $\hat{\underline{\mathbf{x}}}(i)$ of each $\underline{x}(i)$, and then estimates the mean as

$$\hat{\underline{\mu}} = \frac{1}{n}\sum_{i=1}^n \hat{\underline{\mathbf{x}}}(i).$$

The quantizers could be randomized, and there are two types of quantizers of interest: unbiased and biased. A quantizer is said to be unbiased if $\mathbb{E}[\underline{\hat{\mathbf{x}}}] = \underline{x}$ for all inputs $\underline{x}$. It is biased otherwise. Typically, DME is used in iterative algorithms such as distributed optimization, distributed machine learning and federated learning, where the error can accumulate, and therefore it is desirable to use unbiased quantizers particularly when the communication budget is very limited. We are especially interested in the regime when R is close to, or less than 1.

Table 1: DME worst-case guarantees for schemes with communication rate of $\Theta(1)$ bits per dimension, in the regime of large d.

Reference	NMSE	Shared Randomness	Unbiased
QSGD Alistarh et al. (2017)	$O(\sqrt{d}/n)$	No	Yes
Hadamard Suresh et al. (2017)	$O(\log d/n)$	$\Theta(nd)$	No
Kashin Caldas et al. (2018); Safaryan et al. (2022)	$O(1/n)$	No	Yes
EDEN-UHT Vargaftik et al. (2022)	$O(1)$	$\Theta(nd)$	No
EDEN-URR Vargaftik et al. (2022)	$O(1/n)$	∞	Yes
QUIC-FLBen-Basat et al. (2024)	$O(1/n)$	$\Theta(d)$	Yes
This work	$O(1/n)$	No	Yes

Note that if each user independently quantizes their vectors and transmits them to the server using an unbiased scheme, then the MSE in the estimation of the empirical mean is equal to the average of the MSE achieved for the reconstruction of the individual vectors normalized by n. If the quantization and reconstruction of each vector can be done with a constant MSE, then the overall MSE for estimating $\underline{\mu}$ decays as $\Theta(1/n)$. If the quantization is biased, then MSE for the estimation of the empirical mean of all the vectors can be $\Theta(1)$, as the bias from the estimate of each vector adds up. See Vargaftik et al. (2022) for a discussion.

Previous approaches considered normalizing the vectors using the L_2 norm or the L_∞ norm, and then quantize the corresponding normalized vector using a computationally efficient biased/unbiased quantizer designed for unit-norm vectors. It is easy to design efficient quantizers for the L_∞ ball, as simple scalar quantization is optimal. However, this can incur a very large NMSE since the ratio between the L_∞ norm and L_2 norm can be very large. On the other hand, designing computationally efficient minimum MSE quantizers for the L_2 ball is hard, and is a long-standing open problem.

We use a totally different approach, and instead reduce the problem to one of quantizing the L_1 ball, and design computationally efficient unbiased quantizers for this.

The reduced problem (the one we solve in this paper) can be stated as follows: We are given an arbitrary vector $\underline{v}$ with unit L_1 norm, i.e., $\underline{v} \in \mathcal{B}_{L_1}$. A quantizer (denoted Q) consists of a randomized encoder $\mathrm{ENC} : \mathcal{B}_{L_1} \to \{0,1\}^k$ and a decoder $\mathrm{DEC} : \{0,1\}^k \to \mathbb{R}^d$. The set of all reconstruction points (also called the codebook) is

$$\mathcal{C} = \{\mathrm{DEC}(\underline{c}) : \underline{c} \in \{0,1\}^k\}.$$

A quantizer is said to be unbiased if

$$\mathbb{E}[\mathrm{DEC}(\mathrm{ENC}(\underline{v}))] = \underline{v}$$

for every $\underline{v} \in \mathcal{B}_{L_1}$. It is said to be biased otherwise. In the above, the expectation is taken with respect to the randomness in the encoder and decoder. In our solution, *we will assume that there is no shared randomness between the encoder and the decoder*, while the encoder is assumed to have private randomness.

We will measure the performance of a quantizer using the L_2 error, or the mean squared error, maximized over all possible inputs:

$$\mathrm{MSE}_Q = \sup_{\underline{v} \in \mathcal{B}_{L_1}} \mathrm{MSE}(\underline{v}) = \sup_{\underline{v} \in \mathcal{B}_{L_1}} \mathbb{E}\|\hat{\underline{\mathbf{v}}} - \underline{v}\|_2^2.$$

where $\hat{\underline{\mathbf{v}}} = \mathrm{DEC}(\mathrm{ENC}(\underline{v}))$. Our goal is to design an unbiased quantization scheme for the L_1 ball for which the encoder and decoder operate in time $\mathrm{poly}(d)$.

5 OUR SOLUTION AND MAIN RESULTS

Our overall quantization scheme is described in Algorithm 1. We explain the reasoning and quantify the performance in the following subsections.

The following main result is a consequence of Theorems 5.3 and 5.4.

Theorem 5.1. *If the users and server employ Algorithm 1 for DME with $m = \lfloor \beta d \rfloor$, then for every $\beta > 0$, we can achieve*

$$\mathrm{NMSE}(\Pi) \leqslant \frac{1}{4n\beta^2},$$

and asymptotic communication rate

$$\lim_{d \to \infty} R = \rho(\beta) + h_2(\rho(\beta)) + \beta h_2\left(\frac{\rho(\beta)}{\beta}\right)$$

where $\rho(\beta) = \beta + 1 - \sqrt{\beta^2 + 1}$, and $h_2(\gamma) = -\gamma \log_2 \gamma - (1-\gamma)\log_2(1-\gamma)$ is the binary entropy function.

Encoder

Input : $m \in \mathbb{Z}_+, \underline{x} \in \mathbb{R}^d$
Output: $\underline{c} \in \{0,1\}^{dR}$ and $\alpha \in \mathbb{R}$
Set $\underline{y} \leftarrow |\underline{x}|/\|\underline{x}\|_1$
Set $\alpha \leftarrow \|\underline{x}\|_1$
Sample $\underline{\mathbf{q}}$ using Algorithm 2 with $\underline{y}, m$ as input
Set $\underline{\mathbf{u}} \leftarrow \text{sgn}(\underline{x}) \odot \underline{\mathbf{q}}$
Set $\underline{c}$ to be the output of Algorithm 3 with $\underline{\mathbf{u}}, m$ as input.
Output $(\underline{c}, \alpha)$

Decoder

Input : $m \in \mathbb{Z}$, $\underline{c} \in \{0,1\}^{dR}$ and $\alpha \in \mathbb{R}$
Output: Estimate $\hat{\underline{\mathbf{x}}} \in \mathbb{R}^d$
Set $\underline{\mathbf{w}}$ to be the output of Algorithm 4 with $\underline{c}, m$ as input
Output $\hat{\underline{\mathbf{x}}} = \alpha \underline{\mathbf{w}}$

Algorithm 1: Our quantization scheme

5.0.1 Unbiased Quantization of the L_1 Ball

We now describe our unbiased quantizer for the L_1 ball.

Consider any arbitrary $\underline{v} \in \mathcal{B}_{L_1}$. We can express this as

$$\underline{v} = \underline{p} \odot \text{sgn}(\underline{v}),$$

where $\odot$ denotes componentwise product, and sgn denotes the sign. Observe that $\underline{p}$ is a valid probability vector: each entry $p_i \geqslant 0$ and $\sum_{i=1}^d p_i = 1$.

For any positive integer m, we define the set of generalized m-types

$$\mathbb{Q}_m \triangleq \left\{ \frac{1}{m}\underline{q} : \underline{q} \in \mathbb{Z}^d, \sum_{i=1}^d |q_i| = m \right\}.$$

Note that the q_i's can be negative, and $\mathbb{Q}_m \subset \mathcal{B}_{L_1}$. Every element of $\mathbb{Q}_m$ is called a generalized m-type.

For fixed m and any probability vector $\underline{p}$, we can write $m\underline{p}$ as a sum of its (elementwise) integer and fractional parts.

$$m\underline{p} = \lfloor m\underline{p} \rfloor + \{m\underline{p}\}.$$

We solve the problem of quantizing $\underline{p}$ by quantizing $\{m\underline{p}\}$ using carefully chosen $0-1$ vectors.

For this, we define

$$k_{\underline{p},m} \triangleq \sum_{i=1}^d \{mp_i\},$$

and

$$N_{\underline{p},m} \triangleq \binom{d}{k_{\underline{p},m}}.$$

Observe that $k_{\underline{p},m} = \sum_{i=1}^d (mp_i - \lfloor mp_i \rfloor) = m - \sum_{i=1}^d \lfloor mp_i \rfloor$ is always a nonnegative integer.

We define $\mathbb{S}_k$ to be the set of all vectors in $\{0,1\}^d$ with Hamming weight k. For $i \in \left[\binom{d}{k}\right]$, let $\underline{a}(i)$ be the i'th vector in $\mathbb{S}_k$ according to the lexicographic ordering. We design an unbiased quantizer which for each $\underline{p}$, randomly samples a vector $\underline{\mathbf{q}}$ from

$$\mathcal{Q}_{\underline{p}} \triangleq \left\{ \frac{1}{m}(\lfloor m\underline{p} \rfloor + \underline{a}) : \underline{a} \in \mathbb{S}_{k_{\underline{p},m}} \right\}$$

such that $\mathbb{E}[\underline{\mathbf{q}}] = \underline{p}$.

Note that this is possible if and only if $\{m\underline{p}\}$ always lies in the convex hull of $\mathbb{S}_{k_{\underline{p},m}}$, and we show that this is always true. See the supplementary material for details.

Theorem 5.2. *For any $\underline{p} \in \Delta^d$ and positive integer m, the vector $\{m\underline{p}\}$ lies within the convex hull of $\mathbb{S}_{k_{\underline{p},m}}$.*

Therefore, $\underline{p}$ lies within the convex hull of $\mathcal{Q}_{\underline{p}} = \{\underline{q}_1, \ldots, \underline{q}_{N_{\underline{p},m}}\}$, and we can write

$$\underline{p} = \sum_{i=1}^{N_{\underline{p},m}} \alpha_i \underline{q}_i$$

for some $\alpha_1, \ldots, \alpha_{N_{\underline{p}}} \geqslant 0$ satisfying $\sum_{i=1}^{N_{\underline{p}}} \alpha_i = 1$. If we randomly select $\underline{\mathbf{q}}$ according to the distribution

$$P_{\underline{\mathbf{q}}}(\underline{q}_i) = \alpha_i, \quad i = 1, \ldots, N_{\underline{p},m},$$

then $\mathbb{E}[\underline{\mathbf{q}}] = \underline{p}$.

Theorem 5.2 only shows the existence of such a convex combination, and does not give a recipe to efficiently find the α_i's or otherwise sample from the distribution $P_{\underline{\mathbf{q}}}$. We propose Algorithm 2 as a means of computationally efficiently sampling according to $P_{\underline{\mathbf{q}}}$.

Lemma 5.1. *Algorithm 2 is unbiased for every $\underline{p} \in \Delta^d$, and the computational complexity is $O(d)$.*

5.0.2 Enumeration of Types and Communication Rate

A simple combinatorial argument can be used to find the number of generalized m-types, which in turn gives the number of bits required to encode an m-type in binary.

Lemma 5.2.

$$|\mathbb{Q}_m| = f(m,d) \triangleq \begin{cases} \sum_{j=1}^{\min\{d,m\}} 2^j \binom{d}{j}\binom{m-1}{j-1}, & \text{if } m > 0, d > 0 \\ 1, & \text{if } m = 0, d > 0 \\ 0, & \text{otherwise.} \end{cases}$$

Input : $m \in \mathbb{Z}_+, \underline{p} \in \Delta^d$
Output: $\underline{\mathbf{q}} \in \mathbb{P}_m$ an unbiased estimate of $\underline{p}$
Set $\underline{v} \leftarrow \{m\underline{p}\}$
Set $i \leftarrow 1$
Generate $\mathbf{x} \sim \text{Unif}[0,1]$
Set $\mathbf{s} \leftarrow -\mathbf{x}$
Set $\underline{\mathbf{a}} \leftarrow \underline{\mathbf{0}}$
while $i \leqslant d$ **do**
  if $\lfloor \mathbf{s} \rfloor + 1 = \lfloor \mathbf{s} + \{mp_i\} \rfloor$ **then**
    $\mathbf{a}_i \leftarrow 1$
  else
    $\mathbf{a}_i \leftarrow 0$
  end
  $\mathbf{s} \leftarrow \mathbf{s} + \{mp_i\}$
  $i \leftarrow i + 1$
end
Output $\underline{\mathbf{q}} = \frac{1}{m}\left(\lfloor m\underline{p} \rfloor + \underline{\mathbf{a}}\right)$

Algorithm 2: Unbiased Sampling Algorithm

Algorithm 2 outputs a random generalized type $\underline{\mathbf{q}}$, and the number of bits required to represent this is $\log_2 f(d,m)$. Algorithms 3 and 4 can be implemented in polynomial time (in m, d), and the algorithm itself is derived based on Cover (1973).

Input : Generalized m-type $\frac{1}{m}[k_1, k_2, \ldots, k_d] = \frac{1}{m}\underline{k}$
Output: Lexicographic index $i_{\underline{k}}$.
Set

$$\mathrm{w}_{\underline{k}}(j) \leftarrow \sum_{l=1}^{j} |k_l|$$

for $j = 1, 2 \ldots, d$.
The lexicographic index is,

$$i_{\underline{k}} \leftarrow \sum_{j=1}^{d-1} \sum_{l=\mathrm{w}_{\underline{k}}(j-1)-m}^{k_j - 1} \mathrm{f}\left(m - \mathrm{w}_{\underline{k}}(j-1) - |l|, d-j\right)$$

$$i_{\underline{k}} \leftarrow i_{\underline{k}} + \mathbb{1}\{k_d > 0\}$$

Algorithm 3: Encoding Algorithm

In applications such as federated learning, which involve multiple rounds of communication where m and d are kept fixed, each client (and the server) can precompute the values of $f(i,j)$ for all $0 \leqslant i \leqslant m, 0 \leqslant j \leqslant d$. Then, encoding and decoding can be performed using $O(md)$ additions and $O(md)$ memory access per user.

The rate of communication, defined as the number of bits transmitted per dimension, is equal to $\frac{1}{d}\log_2 |\mathbb{Q}_m| = \frac{1}{d}\log_2 f(m,d)$. Since we are typically interested in the regime where d is large and $m = O(d)$ (which corresponds to a constant number of bits per dimension), we can derive the following bound on the asymptotic value of $\frac{1}{d}\log_2 |\mathbb{Q}_m|$.

Input : Lexicographic index i
Output: Generalized type $\frac{1}{m}[k_1, k_2, \ldots, k_d] = \frac{1}{m}\underline{k}$
for $j = 1$ **to** $d-1$ **do**
  $l \leftarrow \mathrm{w}_{\underline{k}}(j-1) - m$
  while $f(m - w_{\underline{k}}(j-1) - |l|, d-j) \leqslant i$ **do**
    $i \leftarrow i - \mathrm{f}(m - \mathrm{w}_{\underline{k}}(j-1) - |l|, d-j)$
    $l \leftarrow l + 1$
  end
  $k_j \leftarrow l$
end
if $i = 1$ **then**
  $k_d \leftarrow m - \mathrm{w}_{\underline{k}}(d-1)$
else
  $k_d \leftarrow \mathrm{w}_{\underline{k}}(d-1) - m$
end

Algorithm 4: Decoding Algorithm

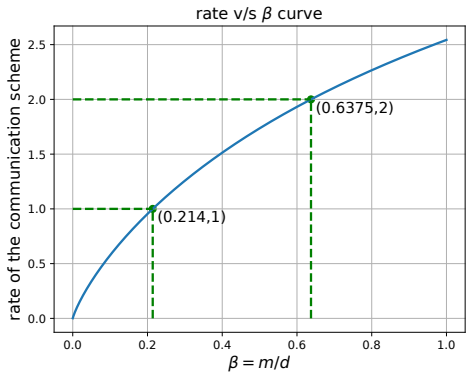


Figure 2: Asymptotic rate v/s β from Theorem 5.3.

Theorem 5.3. *Let us fix $\beta > 0$. If $m = d\beta$ then*

$$\lim_{d\to\infty} \frac{\log_2 f(d\beta, d)}{d} = \rho(\beta) + h_2(\rho(\beta)) + \beta h_2\left(\frac{\rho(\beta)}{\beta}\right)$$

where $\rho(\beta) = \beta + 1 - \sqrt{\beta^2+1}$, and $h_2(\alpha) = -\alpha\log_2\alpha - (1-\alpha)\log_2(1-\alpha)$ is the binary entropy function.

From Figure 2, we can see that to have an asymptotic rate of 1 (as $d \to \infty$), we should set $\beta = 0.214$ or equivalently $m = 0.214d$. Likewise, for the communication scheme to have an asymptotic rate of 2, we should set $\beta = 0.6375$.

5.0.3 MSE of Algorithm 1

Lemma 5.3. *For a given parameter $k = \sum_{i=1}^{d}\{mp_i\}$, the MSE of Algorithm 2 is maximized for $\{m\underline{p}\} = \frac{k}{d}\underline{\mathbf{1}}$*

Proof. Fix any probability vector $\underline{p}$ and define $k = \sum_{i=1}^{d}\{mp_i\}$. Let $\underline{\mathbf{a}}$ be the unbiased estimate of $\{m\underline{p}\}$

given by our algorithm. We know that $\underline{\mathbf{a}}$ has exactly k ones, and $\mathbf{E}[\underline{\mathbf{a}}] = \{m\underline{p}\}$. Then the MSE is,

$$\begin{aligned}\text{MSE}\left(\{m\underline{p}\}\right) &= \mathbb{E}\left[\|\underline{\mathbf{a}} - \{m\underline{p}\}\|_2^2\right] \\ &= \mathbb{E}\left[\|\underline{\mathbf{a}}\|_2^2\right] - 2\mathbb{E}\left[\underline{\mathbf{a}}\right]^\intercal \{m\underline{p}\} + \|\{m\underline{p}\}\|_2^2 \\ &= k - \|\{m\underline{p}\}\|_2^2\end{aligned}$$

To find the maximum MSE we need to minimize $\|\{m\underline{p}\}\|_2^2$ over $\{m\underline{p}\} \in k\Delta^d \cap [0,1]^d$. It is straightforward to see that the L_2 norm is minimized by $\{m\underline{p}\} = \frac{k}{d}\underline{\mathbf{1}}$. Hence the MSE is maximized by $\{m\underline{p}\} = \frac{k}{d}\underline{\mathbf{1}}$ and,

$$\text{MSE}\left(\{m\underline{p}\}\right) \leqslant \frac{k(d-k)}{d}$$

. □

We use the above to show that

Theorem 5.4. *For any $\underline{x} \in \mathbb{R}^d$, let $\hat{\underline{\mathbf{x}}}$ be the output of the decoder in Algorithm 1. Then, $\mathbb{E}[\hat{\underline{\mathbf{x}}}] = \underline{x}$ and*

$$\frac{\mathbb{E}\left[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2\right]}{\|\underline{x}\|_2^2} \leqslant \frac{d^2}{4m^2} = \frac{1}{4\beta^2},$$

where $\beta = m/d$. Moreover,

$$\sup_{\underline{x} \in \mathcal{B}_{L_1}} \mathbb{E}\left[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2\right] \leqslant \frac{1}{4d\beta^2}$$

Proof. From Lemma 5.3 we have an upper bound on the MSE in quantizing $\{m\underline{p}\}$ with parameter $k = \sum_{i=1}^d \{mp_i\}$:

$$\text{MSE}\left(\{m\underline{p}\}\right) \leqslant \frac{k\,(d-k)}{d}$$

Hence, for any $\underline{p}$, we have

$$\begin{aligned}\mathbb{E}\left[\|\underline{\mathbf{q}} - \underline{p}\|^2\right] &= \mathbb{E}\left[\left\|\frac{\lfloor m\underline{p}\rfloor + \underline{\mathbf{a}}}{m} - \frac{\lfloor m\underline{p}\rfloor + \{m\underline{p}\}}{m}\right\|^2\right] \\ &= \frac{\mathbb{E}\left[\|\underline{\mathbf{a}} - \{m\underline{p}\}\|^2\right]}{m^2} \\ &\leqslant \frac{k\,(d-k)}{dm^2}.\end{aligned}$$

Maximizing the above bound over k gives,

$$\mathbb{E}\left[\|\underline{\mathbf{q}} - \underline{p}\|^2\right] \leqslant \frac{d}{4m^2}. \tag{1}$$

We have,

$$\hat{\underline{\mathbf{x}}} = \|\underline{x}\|_1 \operatorname{sgn}(\underline{x}) \odot \underline{\mathbf{q}} \tag{2}$$

which implies that

$$\mathbb{E}\left[\hat{\underline{\mathbf{x}}}\right] = \|\underline{x}\|_1 \operatorname{sgn}(\underline{x}) \odot \mathbb{E}[\underline{\mathbf{q}}]$$

Since $\mathbb{E}[\underline{\mathbf{q}}] = \underline{p}$, we can write,

$$\mathbb{E}[\hat{\underline{\mathbf{x}}}] = \|\underline{x}\|_1 \operatorname{sgn}(\underline{x}) \odot \underline{p} = \underline{x}. \tag{3}$$

Hence it is unbiased.

Now,

$$\mathbb{E}\left[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2\right] = \|\underline{x}\|_1^2\, \mathbb{E}\left[\|\underline{\mathbf{q}} - \underline{p}\|_2^2\right] \tag{4}$$

From (1), we can write,

$$\mathbb{E}\left[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2\right] \leqslant \|\underline{x}\|_1^2\, \frac{d}{4m^2} \tag{5}$$

Normalizing the above,

$$\frac{\mathbb{E}\left[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2\right]}{\|\underline{x}\|_2^2} \leqslant \frac{d}{4m^2} \cdot \left(\frac{\|\underline{x}\|_1}{\|\underline{x}\|_2}\right)^2 \tag{6}$$

However,

$$1 \leqslant \frac{\|\underline{x}\|_1}{\|\underline{x}\|_2} \leqslant \sqrt{d} \tag{7}$$

where the rightmost inequality follows from Cauchy-Schwartz, and we can write,

$$\begin{aligned}\frac{\mathbb{E}\left[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2\right]}{\|\underline{x}\|_2^2} &\leqslant \frac{d}{4m^2}\frac{\|\underline{x}\|_1^2}{\|\underline{x}\|_2^2} \\ &\leqslant \frac{d^2}{4m^2}\end{aligned}$$

which completes the proof. The second part is a direct consequence of (5). □

5.0.4 Lower bounds

We can compute simple lower bounds for independent quantizers and show that our scheme is order-optimal.

Lemma 5.4. *Let $\mathcal{S}$ be an arbitrary compact subset of $\mathbb{R}^d$. For any quantizer with codebook $\mathcal{C}$ having $|\mathcal{C}| = 2^{dR}$ elements, we have*

$$\sup_{\underline{x} \in \mathcal{S}} \mathbb{E}[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|^2] \geqslant \left(\frac{\text{vol}(\mathcal{S})}{2^{dR}\text{vol}(\mathcal{B}_{L_2})}\right)^{2/d}.$$

The above follows from a simple covering argument.

If $R = \Theta(1)$, then the above lemma says that for vectors with bounded L_2 norm,

$$\sup_{\underline{x}} \frac{\mathbb{E}[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2]}{\|\underline{x}\|_2^2} = \sup_{\underline{x} \in \mathcal{B}_{L_2}} \mathbb{E}[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2] \geqslant \frac{1}{2^{2R}},$$

which is $\Omega(1)$. Hence, our scheme is optimal.

Similarly,

$$\sup_{\underline{x} \in \mathcal{B}_{L_1}} \mathbb{E}[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2] \geqslant \left(\frac{\text{vol}(\mathcal{B}_{L_1})}{2^{dR}\text{vol}(\mathcal{B}_{L_2})}\right)^{2/d}$$

Using Stirling's approximation and simplifying, we get

$$\sup_{\underline{x}\in\mathcal{B}_{L_1}} \mathbb{E}[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2] \geqslant \frac{2e}{2^{2R}\pi d}(1 + o(1)),$$

and once again, our scheme is order-optimal if $R = \Theta(1)$.

5.1 Biased Type-Based Quantization

We also propose a biased algorithm for DME based on the quantizer for probability vectors in Reznik (2011). Reznik (2011) proposed a biased algorithm that maps a probability vector to the closest point in $\mathbb{P}_m$.

The biased type-based quantization scheme is identical to Algorithm 1, except for the following: Instead of sampling $\underline{\mathbf{q}}$ using Algorithm 2, we compute $\underline{\mathbf{q}}$ to be the closest point in $\mathbb{P}_m$ using the algorithm in Reznik (2011).

The computational complexity of quantization, encoding and decoding is the same as that of our unbiased algorithm, up to a multiplicative (universal) constant.

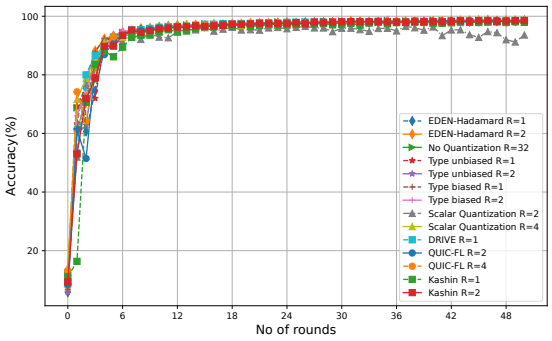
6 SIMULATION RESULTS

We describe some empirical results in this section. More simulation results can be found in the supplementary material. Simulations were performed on a workstation with dual AMD Epyc 7452 processors, 128GB RAM and an NVIDIA A6000 GPU. For the algorithms in Vargaftik et al. (2021, 2022); Safaryan et al. (2022); Ben-Basat et al. (2024), we used the implementation provided in Ben-Basat et al. (2024).

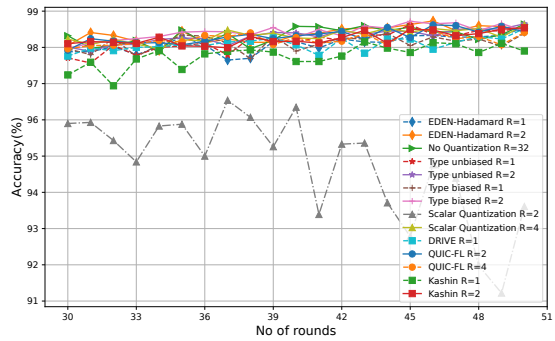
We compare our results with the following algorithms: DRIVE Vargaftik et al. (2021), EDEN Vargaftik et al. (2022), Kashin Safaryan et al. (2022), QUIC-FL Ben-Basat et al. (2024). Our algorithm is labeled "Type unbiased." We also implement "Type biased," which is described in Section 5.1. For federated learning simulations, we used the Flower library Beutel et al. (2020). Our implementation code is publicly available on GitHub and can be accessed at Suresh Babu et al. (2025).

Figures 3a and 3b show the performance of various algorithms when used for federated learning over the MNIST dataset LeCun et al. (2010). In our setup, we partition the MNIST dataset in an i.i.d. manner among 100 clients, with 10 clients selected uniformly at random in each round. Each client trains a simple convolutional neural network (as defined in our code's `Net` class) using cross entropy loss and a batch size of 2048. The network employs two main convolutional layers with max pooling and dropout for regularization, ensuring robust feature extraction and preventing overfitting. After local training, the computed gradients are quantized using various algorithms before being sent to the server for aggregation under the FedAvg strategy. We see that good accuracy can be obtained even at $R = 1$. In most of our simulations, we observe that Type biased, Type unbiased, and QUIC-FL perform the best, with their performances being very close to each other.

We also empirically simulate DME with $d = 2048$ and $n = 100$. Fig. 4 shows the worst-case and average NMSE achieved by various algorithms when the vectors are chosen to have i.i.d. Gaussian entries. In this case, our scheme and the type biased quantizer seem to perform the best, followed closely by QUIC-FL and EDEN.

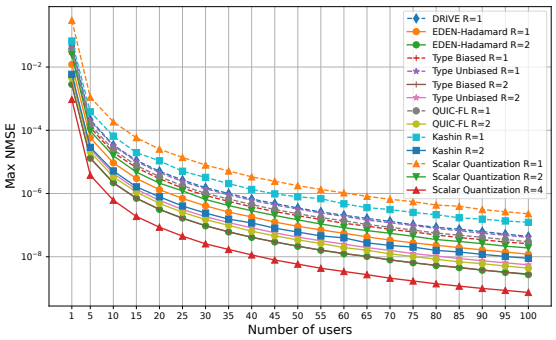


(a) Accuracy for federated learning on MNIST.

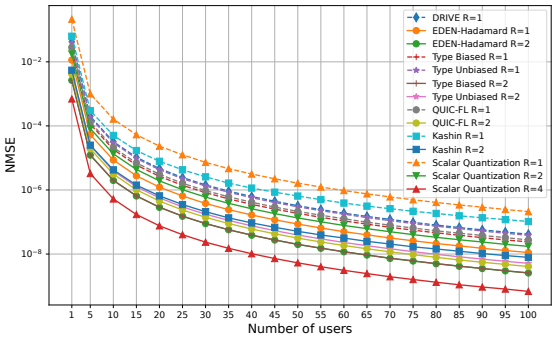


(b) Zoomed-in view of (a).

Figure 3: Federated learning accuracy on the MNIST database. The first figure shows the overall performance while the second figure provides a detailed zoomed-in view.



(a) Maximum NMSE.



(b) Average NMSE.

Figure 4: NMSE vs. number of users with $d = 2^{11}$. The vector components are i.i.d. from a $\mathcal{N}(0, 1)$ distribution.

References

Acharya, J., Canonne, C., Mayekar, P., and Tyagi, H. (2021). Information-constrained optimization: can adaptive processing of gradients help? *Advances in Neural Information Processing Systems*, 34:7126–7138.

Alistarh, D., Grubic, D., Li, J., Tomioka, R., and Vojnovic, M. (2017). Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in neural information processing systems*, 30.

Barnes, L., Cameron, S., Chow, T., Cohen, E., Frankston, K., Howard, B., Kochman, F., Scheinerman, D., and VanderKam, J. (2024). Efficient Unbiased Sparsification. *arXiv e-prints*, page arXiv:2402.14925.

Basu, D., Data, D., Karakus, C., and Diggavi, S. (2019). Qsparse-local-sgd: Distributed sgd with quantization, sparsification and local computations. *Advances in Neural Information Processing Systems*, 32.

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up Robust Features. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*, pages 404–417. Springer.

Ben-Basat, R., Portnoy, A., Einziger, G., Ben-Itzhak, Y., Mitzenmacher, M., et al. (2024). Accelerating federated learning with quick distributed mean estimation. In *Forty-first International Conference on Machine Learning*.

Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Fernandez-Marques, J., Gao, Y., Sani, L., Kwing, H. L., Parcollet, T., Gusmão, P. P. d., and Lane, N. D. (2020). Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*.

Caldas, S., Konecny, J., McMahan, H. B., and Talwalkar, A. (2018). Expanding the reach of federated learning by reducing client resource requirements. *arXiv preprint arXiv:1812.07210*.

Chandrasekhar, V., Takacs, G., Chen, D. M., Tsai, S. S., Reznik, Y., Grzeszczuk, R., and Girod, B. (2012). Compressed Histogram of Gradients: A Low-Bitrate Descriptor. *International journal of computer vision*, 96:384–399.

Condat, L. and Richtárik, P. (2022). MURANA: A generic framework for stochastic variance-reduced optimization. In *Mathematical and Scientific Machine Learning*, pages 81–96. PMLR.

Condat, L., Yi, K., and Richtárik, P. (2022). Ef-bv: A unified theory of error feedback and variance reduction mechanisms for biased and unbiased compression in distributed optimization. *Advances in Neural Information Processing Systems*, 35:17501–17514.

Cover, T. (1973). Enumerative Source Encoding. *IEEE Transactions on Information Theory*, 19(1):73–77.

Cover, T. M. and Thomas, J. (1999). *Elements of information theory*. John Wiley & Sons.

Csiszár, I. and Körner, J. (2011). *Information theory: coding theorems for discrete memoryless systems*. Cambridge University Press.

Davies, P., Gurunanthan, V., Moshrefi, N., Ashkboos, S., and Alistarh, D.-A. (2021). New bounds for distributed mean estimation and variance reduction. In *9th International Conference on Learning Representations*.

Gagie, T. (2006). Compressing probability distributions. *Information Processing Letters*, 97(4):133–137.

Graf, S. and Luschgy, H. (2000). *Foundations of Quantization for Probability Distributions*. Springer Science & Business Media.

Gray, R. (1984). Vector quantization. *IEEE Assp Magazine*, 1(2):4–29.

He, Y., Huang, X., and Yuan, K. (2024). Unbiased compression saves communication in distributed optimization: when and how much? *Advances in Neural Information Processing Systems*, 36.

Horváth, S., Kovalev, D., Mishchenko, K., Richtárik, P., and Stich, S. (2023). Stochastic distributed learning with gradient quantization and double-variance reduction. *Optimization Methods and Software*, 38(1):91–106.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2021). Advances and open problems in federated learning. *Foundations and trends® in machine learning*, 14(1–2):1–210.

Konecnỳ, J. and Richtárik, P. (2018). Randomized distributed mean estimation: Accuracy vs. communication. *Frontiers in Applied Mathematics and Statistics*, 4:62.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical report.

LeCun, Y., Cortes, C., and Burges, C. (2010). Mnist hand-written digit database.

Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International journal of computer vision*, 60:91–110.

Mayekar, P., Jha, S., Suresh, A. T., and Tyagi, H. (2023). Wyner-ziv estimators for distributed mean estimation with side information and optimization. *IEEE Transactions on Information Theory*.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

Polino, A., Pascanu, R., and Alistarh, D. (2018). Model compression via distillation and quantization. In *International Conference on Learning Representations*.

Reznik, Y. A. (2011). An Algorithm for Quantization of Discrete Probability Distributions. In *2011 Data Compression Conference*, pages 333–342.

Richtárik, P., Sokolov, I., and Fatkhullin, I. (2021). Ef21: A new, simpler, theoretically better, and practically faster error feedback. *Advances in Neural Information Processing Systems*, 34:4384–4396.

Richtárik, P., Sokolov, I., Gasanov, E., Fatkhullin, I., Li, Z., and Gorbunov, E. (2022). 3pc: Three point compressors for communication-efficient distributed training and a better theory for lazy aggregation. In *International Conference on Machine Learning*, pages 18596–18648. PMLR.

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

Suresh, A. T., Felix, X. Y., Kumar, S., and McMahan, H. B. (2017). Distributed mean estimation with limited communication. In *International conference on machine learning*, pages 3329–3337. PMLR.

Suresh, A. T., Sun, Z., Ro, J., and Yu, F. (2022). Correlated quantization for distributed mean estimation and optimization. In *International Conference on Machine Learning*, pages 20856–20876. PMLR.

Suresh Babu, N., Kumar, R., and Vatedka, S. (2025). Github repository. `https://github.com/Ritesh622/Unbiased-Quantization-Distributed-Mean-Estimation`.

Tyurin, A. and Richtárik, P. (2024). 2direction: Theoretically faster distributed training with bidirectional communication compression. *Advances in Neural Information Processing Systems*, 36.

Vargaftik, S., Basat, R. B., Portnoy, A., Mendelson, G., Itzhak, Y. B., and Mitzenmacher, M. (2022). Eden: Communication-efficient and robust distributed mean estimation for federated learning. In *International Conference on Machine Learning*, pages 21984–22014. PMLR.

Vargaftik, S., Ben-Basat, R., Portnoy, A., Mendelson, G., Ben-Itzhak, Y., and Mitzenmacher, M. (2021). DRIVE: One-Bit Distributed Mean Estimation. *Advances in Neural Information Processing Systems*, 34:362–377.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv e-prints*, page arXiv:1708.07747.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including

external libraries. [Yes]
Provided along with the supplementary material.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   Some proofs are provided in the supplementary material.
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

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

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

7 SUPPLEMENTARY MATERIAL

7.1 Note on encoding and decoding algorithms

The algorithms for encoding and decoding generalized types into binary are based on a recursive algorithm for enumerating types introduced by Cover (1973). This can be viewed as an extension of the approach discussed in Reznik (2011) to the case of generalized types. If $\frac{1}{m}\underline{k} = \frac{1}{m}[k_1, k_2, \ldots, k_d]$ is a generalized type, then its corresponding lexicographic index is given by

$$i_{\underline{k}} = \sum_{j=1}^{d-1} \sum_{l=\mathrm{w}_{\underline{k}}(j-1)-m}^{k_j-1} \mathrm{f}\left(m - \mathrm{w}_{\underline{k}}(j-1) - |l|, d-j\right).$$

For each j in the outer summation, the inner summation counts the number of generalized types that has the same value as $\frac{1}{m}\underline{k}$ in all coordinates less than j and has a value that is less than v_j in coordinate j. We can equivalently say that for each j in the outer summation, the inner summation over l computes the cardinality of the set $\{\underline{v} \in \mathbb{Q}_m : v_i = k_i,\ \forall 1 \leqslant i \leqslant j-1, v_j < k_j\}$. Summing these values over all $j \in \{1, 2, \ldots, d-1\}$ and adding $\mathbb{1}_{\{k_d > 0\}}$ gives the lexicographic index of the generalized type $\frac{1}{m}\underline{k}$.

Similarly the decoding algorithm gives the correct result because for each j, the smallest l that satisfies the condition $f(m - w_{\underline{k}}(j-1) - |l|, d-j) > i$, is the actual value of k_j.

7.2 Proof of Theorem 5.2

The main ingredient of the proof is Farkas lemma. For notational convenience, we set $k = k_{\underline{p},m}$.

Lemma 7.1 (Farkas Lemma, Boyd (2004))**.** *Let $A \in \mathbb{R}^{m\times n}$ and $\underline{b} \in \mathbb{R}^m$. Then exactly one of the two assertions holds true*[2]*.*

1. *There exists $\underline{x} \in \mathbb{R}^n$ such that $A\underline{x} = \underline{b}$ and $\underline{x} \geqslant \underline{\mathbf{0}}$*
2. *There exists $\underline{y} \in \mathbb{R}^m$ such that $A^\intercal \underline{y} \geqslant 0$ and $\underline{b}^\intercal \underline{y} \leqslant 0$.*

Let us define

$$A = \left[\underline{a}_1, \underline{a}_2, \ldots, \underline{a}_{\binom{d}{k}}\right] \in \mathbb{R}^{d\times\binom{d}{k}}$$

and

$$\underline{b} = \left\{m\underline{p}\right\} \in \mathbb{R}^d.$$

To prove the theorem, we need to show the existence of $\underline{x} \geqslant \underline{\mathbf{0}}$ satisfying $A\underline{x} = \left\{m\underline{p}\right\}$ and $\sum_{i=1}^d x_i = 1$.

We first show that the last condition $\sum_{i=1}^d x_i = 1$ is in fact redundant. If $A\underline{x} = \{mp\}$, then

$$\begin{aligned}
\underline{\mathbf{1}}^\intercal A\underline{x} = \underline{\mathbf{1}}^\intercal \left\{m\underline{p}\right\} &= \sum_{i=1}^d \{mp_i\} \\
k\underline{\mathbf{1}}^\intercal \underline{x} &= k \\
\sum_{i=1}^d x_i &= 1
\end{aligned}$$

Since every column of A has exactly k ones and $d-k$ zeros, the left hand side is simply $k\underline{\mathbf{1}}^\intercal\underline{x} = k\sum_{i=1}^d x_i$. On the other hand, we have assumed that $\left\{m\underline{p}\right\}$ sums to k. Therefore, if $A\underline{x} = \left\{m\underline{p}\right\}$, then $\sum_{i=1}^d x_i = 1$, and it suffices to show the existence of $\underline{x} \geqslant \underline{\mathbf{0}}$ satisfying $A\underline{x} = \left\{m\underline{p}\right\}$

We show that $\forall \underline{y} \in \mathbb{R}^d$, either $A^\intercal\underline{y} \geqslant 0$ and $\underline{b}^\intercal\underline{y} \geqslant 0$, or $A^\intercal\underline{y} \leqslant 0$ and $\underline{b}^\intercal\underline{y} \leqslant 0$. Using Farkas lemma would then complete the proof.

[2]In what follows, inequalities for vectors hold componentwise.

Define the set $\mathcal{P}_A = \left\{\underline{y} \mid A^\intercal \underline{y} \geqslant \underline{\mathbf{0}}\right\}$. Since A contains as column vectors all the possible vectors with k ones and $d-k$ zeros, $A^\intercal \underline{y}$ is the vector containing the sum of all the k element subsets of $\underline{y}$. Hence, $A^\intercal \underline{y} \geqslant \underline{\mathbf{0}}$ implies that the number of negative entries in $\underline{y}$ should be less than k. Let us suppose that the number of negative entries in $\underline{y}$ is l and $1 \leqslant l \leqslant k-1$ (If $l = 0$, then the second condition of Farkas lemma cannot be true anyway). Also, without loss of generality, we will assume that $y_1 \leqslant y_2 \leqslant \ldots \leqslant y_d$ and $y_1, y_2, \ldots, y_l < 0$ and $y_{l+1}, \ldots, y_d \geqslant 0$. Now consider $\underline{b}^\intercal \underline{y}$:

$$\begin{aligned}\sum_{i=1}^{d} \{mp_i\}\, y_i &= \sum_{i=1}^{k-1} \{mp_i\}\, y_i + \sum_{i=k}^{d} \{mp_i\}\, y_i \\ &\geqslant \sum_{i=1}^{k-1} \{mp_i\}\, y_i + \left(\sum_{i=k}^{d} \{mp_i\}\right) y_k\end{aligned}$$

Using the fact that $\sum_{i=1}^{d} \{mp_i\} = k$,

$$\begin{aligned}\sum_{i=1}^{d} \{mp_i\}\, y_i &= \sum_{i=1}^{k-1} \{mp_i\}\, y_i + \left(k - \sum_{i=1}^{k-1} \{mp_i\}\right) y_k \\ &= \sum_{i=1}^{k-1} \{mp_i\}\, y_i + \left(\sum_{i=1}^{k-1} (1 - \{mp_i\})\right) y_k + y_k \\ &\geqslant \sum_{i=1}^{k-1} \{mp_i\}\, y_i + \sum_{i=1}^{k-1} (1 - \{mp_i\})\, y_i + y_k \\ &= \sum_{i=1}^{k} y_i \geqslant 0\end{aligned}$$

where the last step holds because $A^\intercal \underline{y} \geqslant \underline{\mathbf{0}}$ and the sum of any k entries of $\underline{y}$ is nonnegative. Therefore, the second statement of Farkas lemma cannot be true.

This completes the proof. □

7.3 Correctness of the Sampling Algorithm

7.3.1 Sampling with Specified Marginals

Sampling with specified marginals is a well known problem. See, e.g., Barnes et al. (2024) and the references therein. In fact, the above paper studies a more general problem of unbiased sparsification.

Consider a set $\mathcal{I}$ containing d elements. WLOG, assume $\mathcal{I} = \{1, 2, \ldots, d\}$. We define the marginal probability of inclusion for each element i as r_i, i.e., let

$$\underline{r} \in [0,1]^d, \text{ such that } \sum_{i=1}^{d} r_i = k.$$

The problem is as follows: Given $\underline{r}$, draw a random subset $\mathcal{T} \subset \mathcal{I}$ satisfying:

- $|\mathcal{T}| = k$ with probability 1, and
- $\mathbb{P}[i \in \mathcal{T}] = r_i$ for $i = 1, 2, \ldots, d$.

There are multiple solutions to this problem, and one possible solution (that is proved to minimize permutation-invariant divergences between $\underline{r}$ and $\mathcal{T}$) is the following: Partition the interval $[0, k]$ into d sub-intervals of lengths $r_1, r_2, \ldots, r_d$. Each of these correspond to an element in $\mathcal{I}$. Define

$$\mathcal{R}_i \triangleq \left(\sum_{l=1}^{i-1} r_l, \quad \sum_{l=1}^{i} r_l\right).$$

Now we generate a random variable $\mathbf{x} \sim \text{Unif}[0,1]$. The element i belongs to the subset $\mathcal{T}$ if there exists $j \in \mathbb{Z}$ such that $\mathbf{x} + j \in \mathcal{R}_i$. Therefore $\mathcal{T}$ can be defined as:

$$\mathcal{T} \triangleq \{i \in \mathcal{I} : \mathbf{x} + j \in \mathcal{R}_i \text{ for some } j \in \mathbb{Z}\}. \tag{8}$$

7.3.2 Connection to Quantization

The connection between quantizing $\{m\underline{p}\}$ using $\underline{\mathbf{a}}$ and the sampling with specified marginals problem is quite straightforward. The vector $\{m\underline{p}\}$ corresponds to $\underline{r}$, since each element lies within $[0,1]$ and they sum to an integer k. Similarly, $\mathbf{a}_i \in \{0,1\}^d$, which contains k ones and $d-k$ zeros, represents a k-element subset, where a coordinate value of 1 indicates the presence of that element, while a coordinate value of 0 indicates its absence in the subset. The criterion $\mathbb{P}(i \in \mathcal{T}) = r_i \quad \text{for} \quad 1 \leqslant i \leqslant d$ aligns with the condition of unbiased quantization. Specifically, $\mathbb{P}(i \in \mathcal{T}) = r_i$ is equivalent to $\mathbb{P}(\mathbf{a}_i = 1) = \mathbb{E}[\mathbf{a}_i] = \{mp_i\}$. Therefore, $\mathbb{P}(i \in \mathcal{T}) = r_i$ for $1 \leqslant i \leqslant d$ is equivalent to $\mathbb{E}[\underline{\mathbf{a}}] = \{m\underline{p}\}$.

Lemma 7.2. *Fix any $\underline{p}$ with $k_{\underline{p}} = k$. Let $\underline{\mathbf{a}}$ be the output of Algorithm 2. Then, $\mathbb{E}[\underline{\mathbf{a}}] = \{m\underline{p}\}$.*

Proof. First, observe that $\lfloor \mathbf{s} \rfloor + 1 = \lfloor \mathbf{s} + \{mp_i\} \rfloor$ if and only if $\exists\, l \in \mathbb{Z}$ such that $\mathbf{x} + l \in \left[\sum_{j=1}^{i-1} \{mp_j\}, \sum_{j=1}^{i} \{mp_j\}\right)$.

We need to show that $\mathbb{P}(\mathbf{a}_i = 1) = \{mp_i\}$ for all i. Since $0 \leqslant \{mp_i\} \leqslant 1$ for all i, the interval $\mathcal{R}_i$ could contain at most one integer. We split the analysis into two cases, based on whether $\mathcal{R}_i$ contains none or one integer.

Case 1: There is no integer in the interval $\mathcal{R}_i$.

In this case, if $\mathbf{x} + l \in \mathcal{R}_i$ for some integer l, then it must be the case that $l = \left\lfloor \sum_{j=1}^{i-1} \{mp_j\} \right\rfloor$. Therefore, if $\mathcal{R}_i$ does not contain an integer, then

$$\begin{aligned}
\mathbb{P}(\mathbf{a}_i = 1) &= \mathbb{P}\left(\mathbf{x} \in \left[\left\{\sum_{j=1}^{i-1} \{mp_j\}\right\}, \quad \left\{\sum_{j=1}^{i-1} \{mp_j\}\right\} + \{mp_i\}\right]\right) \\
&= \{mp_i\}.
\end{aligned} \tag{9}$$

Case 2 : There is exactly one integer in $\mathcal{R}_i$.

In this case, if $\mathbf{x} + l \in \mathcal{R}_i$ for some integer l, then l could be either $\left\lfloor \sum_{j=1}^{i-1} \{mp_j\} \right\rfloor$ or $\left\lfloor \sum_{j=1}^{i-1} \{mp_j\} \right\rfloor + 1$.

Therefore, if $\mathcal{R}_i$ contains one integer, then

$$\begin{aligned}
\mathbb{P}[\mathbf{a}_i = 1] &= \mathbb{P}[\mathbf{x} + l \in \mathcal{R}_i \text{ for some integer } l] \\
&= \mathbb{P}\left[\mathbf{x} + \left\lfloor \sum_{j=1}^{i-1} \{mp_j\} \right\rfloor \in \mathcal{R}_i, \quad \text{or} \quad \mathbf{x} + \left\lfloor \sum_{j=1}^{i-1} \{mp_j\} \right\rfloor + 1 \in \mathcal{R}_i\right] \\
&= \mathbb{P}\left[\mathbf{x} + \left\lfloor \sum_{j=1}^{i-1} \{mp_j\} \right\rfloor \in \mathcal{R}_i\right] + \mathbb{P}\left[\mathbf{x} + \left\lfloor \sum_{j=1}^{i-1} \{mp_j\} \right\rfloor + 1 \in \mathcal{R}_i\right] \\
&= \mathbb{P}\left[\mathbf{x} \in \left[\left\{\sum_{j=1}^{i-1} \{mp_j\}\right\}, \left\{\sum_{j=1}^{i-1} \{mp_j\}\right\} + \{mp_i\}\right]\right] \\
&\quad + \mathbb{P}\left[\mathbf{x} \in \left[\left\{\sum_{j=1}^{i-1} \{mp_j\}\right\} - 1, \left\{\sum_{j=1}^{i-1} \{mp_j\}\right\} + \{mp_i\} - 1\right]\right]
\end{aligned} \tag{10}$$

In the third equality above, we have used the fact that the two events are disjoint.

If $l = \left\lfloor \sum_{j=1}^{i-1} \{mp_j\} \right\rfloor$, then $\left\{\sum_{j=1}^{i-1} \{mp_j\}\right\} + \{mp_i\} \geqslant 1$. Likewise, if $l = \left\lfloor \sum_{j=1}^{i-1} \{mp_j\} \right\rfloor + 1$, then $\left\{\sum_{j=1}^{i-1} \{mp_j\}\right\} - 1 \leqslant 0$. Therefore

$$\mathbb{P}[\mathbf{a}_i = 1] = \mathbb{P}\left[\mathbf{x} \in \left[\left\{\sum_{j=1}^{i-1} \{mp_j\}\right\}, 1\right]\right] + \mathbb{P}\left[\mathbf{x} \in \left[0, \left\{\sum_{j=1}^{i-1} \{mp_j\}\right\} + \{mp_i\} - 1\right]\right]$$

$$= 1 - \left\{ \sum_{j=1}^{i-1} \{mp_j\} \right\} + \left\{ \sum_{j=1}^{i-1} \{mp_j\} \right\} + \{mp_i\} - 1$$
$$= \{mp_i\}. \tag{11}$$

Combining (9) and (11) completes the proof. □

7.4 Proof of Theorem 5.3

$$\mathrm{f}(m,d) = \sum_{j=1}^{\min(d,m)} 2^j \binom{d}{j}\binom{m-1}{j-1}$$
$$= \sum_{j=1}^{\min(d,m)} 2^j \frac{j}{m}\binom{d}{j}\binom{m}{j}$$

Define,

$$h(j) = 2^j \binom{d}{j}\binom{m-1}{j-1}$$

We first find the j that maximizes $h(j)$.

$$\frac{h(j+1)}{h(j)} = 2\frac{(d-j)(m-j)}{j(j+1)}$$

Clearly, $\frac{h(j+1)}{h(j)}$ is a decreasing function in j. Let us find the j for which $\frac{h(j+1)}{h(j)} = 1$.

$$\frac{h(j+1)}{h(j)} = 1 \implies 2\frac{(d-j)(m-j)}{j(j+1)} = 1$$
$$2j^2 - 2(m+d)j + 2md = j^2 + j$$
$$j^2 - (2(m+d)+1)j + 2md = 0$$

$$j = \frac{(2(d+m)+1) \pm \sqrt{(2(d+m)+1)^2 - 8dm}}{2}$$
$$= \left((d+m)+\frac{1}{2}\right) \pm \sqrt{\left((d+m)+\frac{1}{2}\right)^2 - 2dm}$$
$$j^* = d+m+\frac{1}{2} - \sqrt{d^2+m^2+d+m+\frac{1}{4}}$$

$\forall j \leqslant j^*$, we have $\frac{h(j+1)}{h(j)} \geqslant 1$, hence $h(j)$ attains maximum at j^*. But j^* need not be an integer. Hence, either $\lfloor j^* \rfloor$ or $\lceil j^* \rceil$ is the integer that maximizes $h(j)$. Now, we can bound $\mathrm{f}(m,d)$ by,

$$h(\lfloor j^* \rfloor) \leqslant \mathrm{f}(m,d) \leqslant d.\max\{h(\lfloor j^* \rfloor), h(\lceil j^* \rceil)\}$$

$$\frac{\log_2(h(\lfloor j^* \rfloor))}{d} \leqslant \frac{\log_2(\mathrm{f}(m,d))}{d} \leqslant \frac{\log_2(d.\max\{h(\lfloor j^* \rfloor), h(\lceil j^* \rceil)\})}{d} \tag{12}$$

$$\frac{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right)}{d} \leqslant \frac{\log_2\left(\mathrm{f}\left(m,d\right)\right)}{d} \leqslant \max\left\{\frac{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right)}{d}, \frac{\log_2\left(h\left(\lceil j^* \rceil\right)\right)}{d}\right\} + \frac{\log_2 d}{d} \tag{13}$$

Taking the limit as $d \to \infty$ gives us,

$$\lim_{d\to\infty} \frac{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right)}{d} \leqslant \lim_{d\to\infty} \frac{\log_2\left(\mathrm{f}\left(m,d\right)\right)}{d} \leqslant \lim_{d\to\infty} \max\left\{\frac{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right)}{d}, \frac{\log_2\left(h\left(\lceil j^* \rceil\right)\right)}{d}\right\}$$

Now we will show that the limit of the lower bound and the limit of the upper bound converge to the same value. From Stirling's approximation, we know that

$$\sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n} - \frac{1}{360n^3}} < n! < \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n}}$$

If we recursively use the above relation we can upper bound

$$\binom{d}{j} < \sqrt{\frac{d}{2\pi j\left(d-j\right)}} \frac{d^d}{j^j\left(d-j\right)^{d-j}} e^{\frac{1}{12d} - \left(\frac{1}{12j} - \frac{1}{360j^3} + \frac{1}{12(d-j)} - \frac{1}{360(d-j)^3}\right)}$$

$$\binom{m}{j} < \sqrt{\frac{m}{2\pi j\left(m-j\right)}} \frac{m^m}{j^j\left(m-j\right)^{m-j}} e^{\frac{1}{12m}} e^{-\left(\frac{1}{12j} - \frac{1}{360j^3} + \frac{1}{12(m-j)} - \frac{1}{360(m-j)^3}\right)}$$

We can upper bound the exponential term at the end by a constant $e^{\frac{1}{12}}$. So we can upper bound $h\left(j\right)$ by,

$$h\left(j\right) < \frac{2^j}{2\pi} \frac{d^{d+\frac{1}{2}}}{j^{j+\frac{1}{2}}\left(d-j\right)^{d-j+\frac{1}{2}}} \frac{m^{m-\frac{1}{2}}}{j^{j-\frac{1}{2}}\left(m-j\right)^{m-j+\frac{1}{2}}} e^{\frac{1}{6}}$$

Similarly, we can use Stirling's approximation to get lower bounds,

$$\binom{d}{j} > \sqrt{\frac{d}{2\pi j\left(d-j\right)}} \frac{d^d}{j^j\left(d-j\right)^{d-j}} e^{\frac{1}{12d} - \frac{1}{360d^3} - \left(\frac{1}{12j} + \frac{1}{12(d-j)}\right)}$$

$$\binom{m}{j} > \sqrt{\frac{m}{2\pi j\left(m-j\right)}} \frac{m^m}{j^j\left(m-j\right)^{m-j}} e^{\frac{1}{12m} - \frac{1}{360m^3} - \left(\frac{1}{12j} + \frac{1}{12(m-j)}\right)}$$

We can further lower bound the exponential term with $e^{-\frac{1}{12}}$ and get

$$\binom{d}{j} > \sqrt{\frac{d}{2\pi j\left(d-j\right)}} \frac{d^d}{j^j\left(d-j\right)^{d-j}} e^{-\frac{1}{12}}$$
$$\binom{m}{j} > \sqrt{\frac{m}{2\pi j\left(m-j\right)}} \frac{m^m}{j^j\left(m-j\right)^{m-j}} e^{-\frac{1}{12}}$$

So, we can lower bound $h\left(j\right)$ by,

$$h\left(j\right) > \left(\frac{2^j}{2\pi} \frac{d^{d+\frac{1}{2}}}{j^{j+\frac{1}{2}}\left(d-j\right)^{d-j+\frac{1}{2}}}\right)\left(\frac{m^{m-\frac{1}{2}}}{j^{j-\frac{1}{2}}\left(m-j\right)^{m-j+\frac{1}{2}}} e^{-\frac{1}{6}}\right)$$

Let us define,

$$g\left(j\right) = 2^j \frac{d^{d+\frac{1}{2}}}{j^{j+\frac{1}{2}}\left(d-j\right)^{d-j+\frac{1}{2}}} \frac{m^{m-\frac{1}{2}}}{j^{j-\frac{1}{2}}\left(m-j\right)^{m-j+\frac{1}{2}}}$$

Hence we can write,

$$\frac{1}{2\pi} g\left(j\right) e^{-\frac{1}{6}} < h\left(j\right) < \frac{1}{2\pi} g\left(j\right) e^{\frac{1}{6}}$$

$$\begin{aligned} h\left(\lfloor j^* \rfloor\right) &> \frac{1}{2\pi} g\left(\lfloor j^* \rfloor\right) e^{-\frac{1}{6}} \\ \frac{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right)}{d} &> \frac{\log_2\left(g\left(\lfloor j^* \rfloor\right)\right)}{d} + \frac{O\left(1\right)}{d} \\ \lim_{d\to\infty} \frac{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right)}{d} &\geqslant \lim_{d\to\infty} \frac{\log_2\left(g\left(\lfloor j^* \rfloor\right)\right)}{d} \end{aligned}$$

Similarly,

$$\max\left\{h\left(\lfloor j^* \rfloor\right), h\left(\lceil j^* \rceil\right)\right\} < \frac{1}{2\pi} \max\left\{g\left(\lfloor j^* \rfloor\right), g\left(\lceil j^* \rceil\right)\right\} e^{\frac{1}{6}} \tag{14}$$

$$\log_2\left(\max\left\{h\left(\lfloor j^* \rfloor\right), h\left(\lceil j^* \rceil\right)\right\}\right) < \log_2\left(\max\left\{g\left(\lfloor j^* \rfloor\right), g\left(\lceil j^* \rceil\right)\right\}\right) + O(1) \tag{15}$$

and

$$\max\left\{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right), \log_2\left(h\left(\lceil j^* \rceil\right)\right)\right\} < \max\left\{\log_2\left(g\left(\lfloor j^* \rfloor\right)\right), \log_2\left(g\left(\lceil j^* \rceil\right)\right)\right\} + O(1) \tag{16}$$

$$\lim_{d\to\infty} \max\left\{\frac{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right)}{d}, \frac{\log_2\left(h\left(\lceil j^* \rceil\right)\right)}{d}\right\} < \lim_{d\to\infty} \max\left\{\frac{\log_2\left(g\left(\lfloor j^* \rfloor\right)\right)}{d}, \frac{\log_2\left(g\left(\lceil j^* \rceil\right)\right)}{d}\right\} \tag{17}$$

$$\begin{aligned} &\lim_{d\to\infty} \frac{\log_2\left(g\left(\lfloor j^* \rfloor\right)\right)}{d} \\ &= \lim_{d\to\infty} \left(\frac{\lfloor j^* \rfloor}{d} + \left(1 + \frac{1}{2d}\right)\log_2(d)\right) - \left(\frac{\lfloor j^* \rfloor}{d} + \frac{1}{2d}\right)\log_2(\lfloor j^* \rfloor) - \left(1 - \frac{\lfloor j^* \rfloor}{d} + \frac{1}{2d}\right)\log_2(d - \lfloor j^* \rfloor) \\ &\quad + \frac{m}{d}\left(\left(1 - \frac{1}{2m}\right)\log_2(m) - \left(\frac{\lfloor j^* \rfloor}{m} - \frac{1}{2m}\right)\log_2(\lfloor j^* \rfloor)\right) - \left(1 - \frac{\lfloor j^* \rfloor}{m} + \frac{1}{2m}\right)\log_2\left(m - \lfloor j^* \rfloor\right) \\ &= \lim_{d\to\infty} \left(\frac{\lfloor j^* \rfloor}{d} - \frac{\lfloor j^* \rfloor}{d}\log_2\left(\frac{\lfloor j^* \rfloor}{d}\right) - \left(1 - \frac{\lfloor j^* \rfloor}{d}\right)\log_2\left(1 - \frac{\lfloor j^* \rfloor}{d}\right)\right) \\ &\quad + \frac{m}{d}\left(-\frac{\lfloor j^* \rfloor}{m}\log_2\left(\frac{\lfloor j^* \rfloor}{m}\right) - \left(1 - \frac{\lfloor j^* \rfloor}{m}\right)\log_2\left(1 - \frac{\lfloor j^* \rfloor}{m}\right)\right) \end{aligned} \tag{18}$$

$$\lim_{d\to\infty} \frac{\lfloor j^* \rfloor}{d} = \lim_{d\to\infty} \frac{j^*}{d} = \lim_{d\to\infty} \frac{d + m + \frac{1}{2} - \sqrt{d^2 + m^2 + d + m + \frac{1}{4}}}{d}$$

Substituting, $m = d\beta$ gives,

$$\lim_{d\to\infty} \frac{\lfloor j^* \rfloor}{d} = \beta + 1 - \sqrt{\beta^2 + 1}$$

Also,

$$\lim_{d\to\infty} \frac{\lfloor j^* \rfloor}{m} = \lim_{d\to\infty} \frac{d}{m}\frac{\lfloor j^* \rfloor}{d}$$

$$= \frac{\beta + 1 - \sqrt{\beta^2 + 1}}{\beta}$$

Let us define $\rho(\beta) = \beta + 1 - \sqrt{\beta^2 + 1}$. Note that both $\rho(\beta)$ and $\rho(\beta)/\beta$ are bounded within $[0, 1]$ for $\beta > 0$. Then,

$$\lim_{d\to\infty} \frac{\log_2\left(g\left(\lfloor j^* \rfloor\right)\right)}{d} = \rho\left(\beta\right) + h_2\left(\rho\left(\beta\right)\right) + \beta h_2\left(\frac{\rho\left(\beta\right)}{\beta}\right)$$

Similarly, we can show that,

$$\lim_{d\to\infty} \frac{\log_2\left(g\left(\lceil j^* \rceil\right)\right)}{d} = \rho\left(\beta\right) + h_2\left(\rho\left(\beta\right)\right) + \beta h_2\left(\frac{\rho\left(\beta\right)}{\beta}\right)$$

So,

$$\lim_{d\to\infty} \frac{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right)}{d} \geqslant \rho\left(\beta\right) + h_2\left(\rho\left(\beta\right)\right) + \beta h_2\left(\frac{\rho\left(\beta\right)}{\beta}\right)$$

$$\begin{aligned}\lim_{d\to\infty} \max\left\{\frac{\log_2\left(h\left(\lfloor j^* \rfloor\right)\right)}{d}, \frac{\log_2\left(h\left(\lceil j^* \rceil\right)\right)}{d}\right\} \\ \leqslant \rho\left(\beta\right) + h_2\left(\rho\left(\beta\right)\right) + \beta h_2\left(\frac{\rho\left(\beta\right)}{\beta}\right)\end{aligned} \tag{19}$$

Hence, by the sandwich theorem we have

$$\lim_{d\to\infty} \frac{\log_2\left(\mathrm{f}\left(m, d\right)\right)}{d} = \rho\left(\beta\right) + h_2\left(\rho\left(\beta\right)\right) + \beta h_2\left(\frac{\rho\left(\beta\right)}{l}\right)$$

This completes the proof. □

7.5 Proof of Lemma 5.4

If $\hat{\underline{\mathbf{x}}}$ is a random vector that is drawn from $\mathcal{C}$, then

$$\sup_{\underline{x}\in\mathcal{S}} \mathbb{E}[\|\hat{\underline{\mathbf{x}}} - \underline{x}\|_2^2] \geqslant \sup_{\underline{x}\in\mathcal{S}} \min_{\underline{y}\in\mathcal{C}} \|\underline{y} - \underline{x}\|_2^2 \tag{20}$$

which is the covering radius of $\mathcal{C}$. Denote the right hand side by $r_{\mathcal{C},\mathcal{S}}^2$.

It is clear that

$$\mathcal{S} \subset \bigcup_{\underline{y}\in\mathcal{C}} (\underline{y} + r_{\mathcal{C},\mathcal{S}} \mathcal{B}_{L_2})$$

which implies that

$$\mathrm{vol}(\mathcal{S}) \leqslant \mathrm{vol}\left(\bigcup_{\underline{y}\in\mathcal{C}} (\underline{y} + r_{\mathcal{C},\mathcal{S}} \mathcal{B}_{L_2})\right) \leqslant 2^{dR} \mathrm{vol}(\mathcal{B}_{L_2}) r_{\mathcal{C},\mathcal{S}}^d.$$

Therefore,

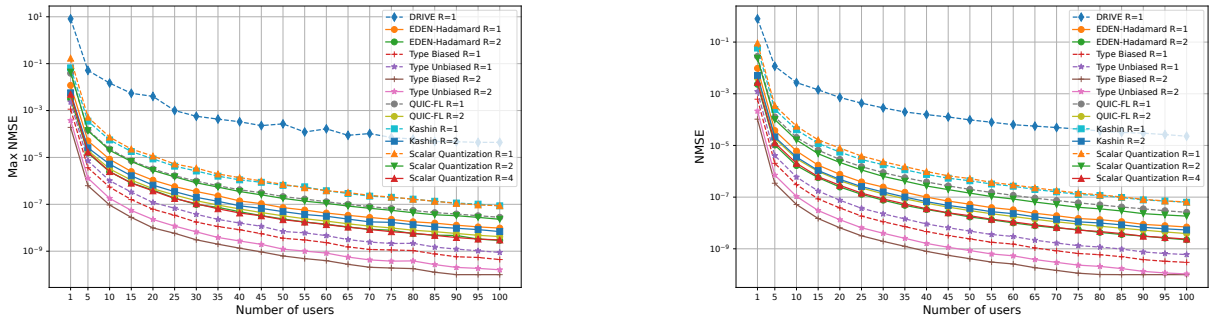
$$r_{\mathcal{C},\mathcal{S}}^2 \geqslant \left(\frac{\mathrm{vol}(\mathcal{S})}{2^{dR}\mathrm{vol}(\mathcal{B}_{L_2})}\right)^{2/d}$$

Using this in (20) gives the lemma. □

7.6 Additional Simulation Results

Figs. 4, 5, 6 and 7 are plots of the worst-case and average NMSE versus the number of users for $d = 2^{11}$, where the components of each vector were chosen i.i.d. Gaussian, Lognormal, Laplace, and Gamma respectively. These NMSE values are computed over 100 configurations per case, with the average computed across 50 independent trials for each configuration. In our simulation, for each configuration we generate up to 100 random vectors

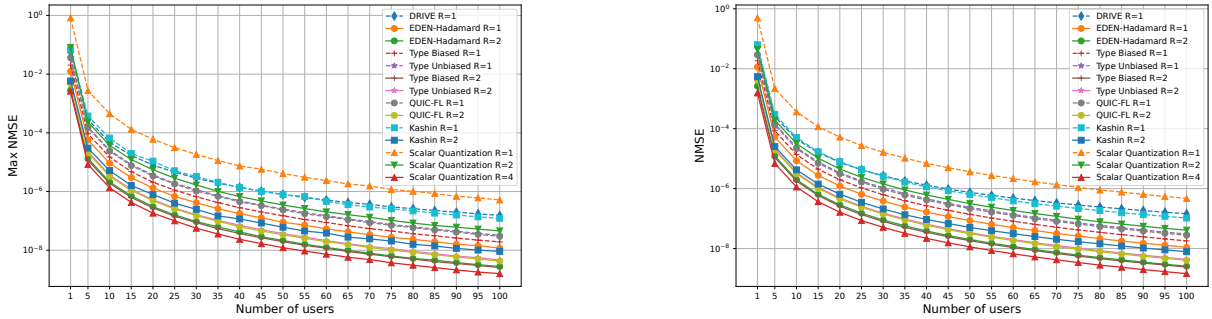
(each corresponding to a user as shown in the plots), compute their empirical mean, and then apply several quantization schemes to obtain quantized estimates. The NMSE is calculated as the squared norm of the difference between the quantized estimate and the true mean, normalized by the number of trials, the sum of squared vector norms, and the number of users. We see that when the components are i.i.d. Gaussian, EDEN and QUIC-FL perform extremely well as expected, since they are tailored to work well with Gaussian inputs. However, when the components are Lognormal, the type-based schemes outperform the rest. For Laplace and Gamma, EDEN performs well and is closely followed by the type-based method and QUIC-FL (all with rate =2).



(a) Maximum NMSE

(b) Average NMSE

Figure 5: NMSE performance versus the number of users with $d = 2^{11}$, where the components of each vector are selected i.i.d. from Lognormal$(1, 2)$.



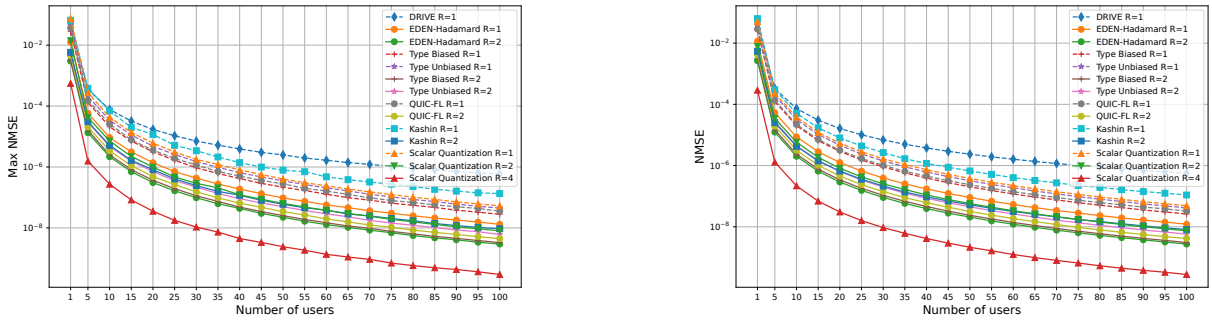
(a) Maximum NMSE

(b) Average NMSE

Figure 6: NMSE performance versus the number of users with $d = 2^{11}$, where the components of each vector are selected i.i.d. from Laplace$(1, 2)$.

We also perform federated learning simulations on the CIFAR10 Krizhevsky (2009) and Fashion MNIST Xiao et al. (2017) datasets. An i.i.d. split of the training dataset is used among 100 clients, with a subset of 10 clients sampled uniformly at random in each round. For CIFAR10, the model is a simple CNN designed for classification, consisting of two convolutional layers followed by three fully connected layers, with standard normalization applied using the dataset's mean and standard deviation. For Fashion MNIST, we employ a slightly deeper CNN architecture-(see the Github repository for details) as the `Net` class for Fashion MNIST—with four convolutional layers (two of which use max pooling and dropout for regularization) and two fully connected layers. The same architecture is used for MNIST. We use a learning rate of 0.01 for CIFAR10 and 0.1 for MNIST and Fashion MNIST, with a momentum of 0.9. Federated averaging runs for 300 rounds on CIFAR10 and 50 rounds on MNIST and Fashion MNIST using the Flower framework. Figs. 8 and 11 show the results for CIFAR10 and Fashion MNIST, respectively. We observe that our algorithms achieve competitive performance compared to other approaches.

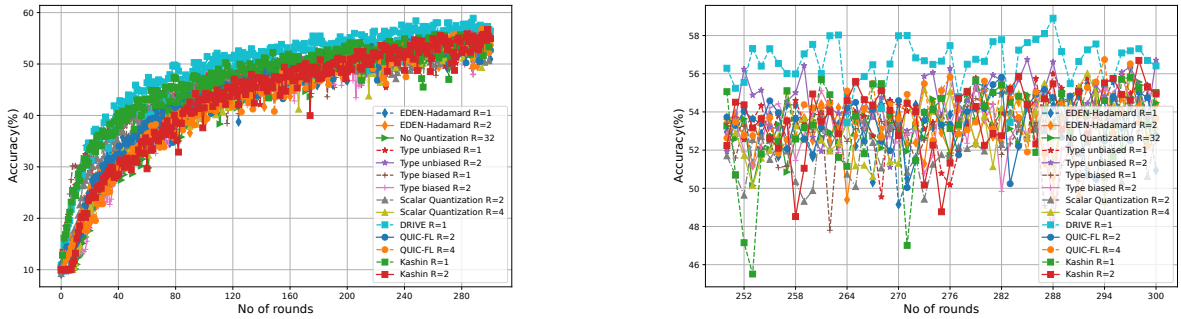
Table 2 presents a comparative analysis of the normalized mean squared error (NMSE) achieved by various



(a) Maximum NMSE

(b) Average NMSE

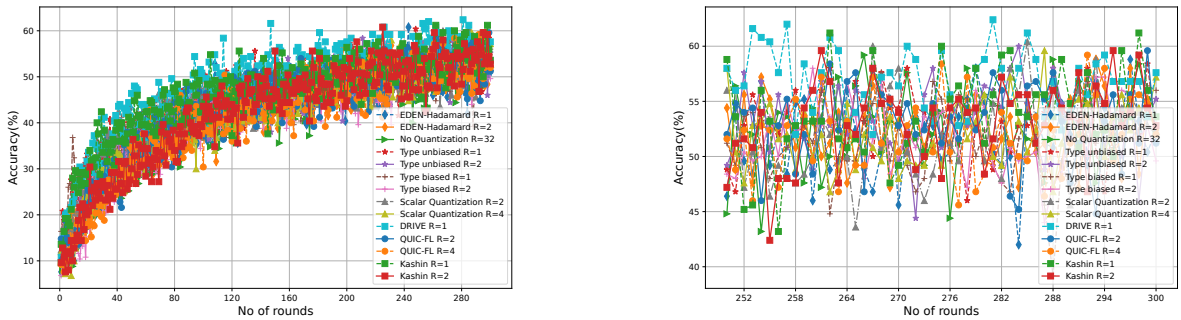
Figure 7: NMSE performance versus the number of users with $d = 2^{11}$, where the components of each vector are selected i.i.d. from Gamma$(2, 2)$.



(a) CIFAR10 Accuracy Plot

(b) CIFAR10 Accuracy Plot (Zoomed-in)

Figure 8: Performance of various algorithms for federated classification over the CIFAR10 dataset with testing done at the server.

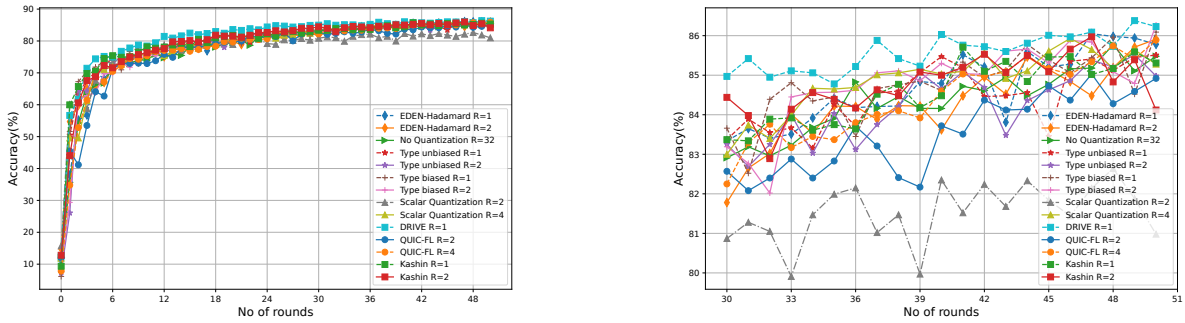


(a) CIFAR10 Accuracy Plot

(b) CIFAR10 Accuracy Plot (Zoomed-in)

Figure 9: Performance of various algorithms for federated classification over the CIFAR10 dataset with testing done at the clients.

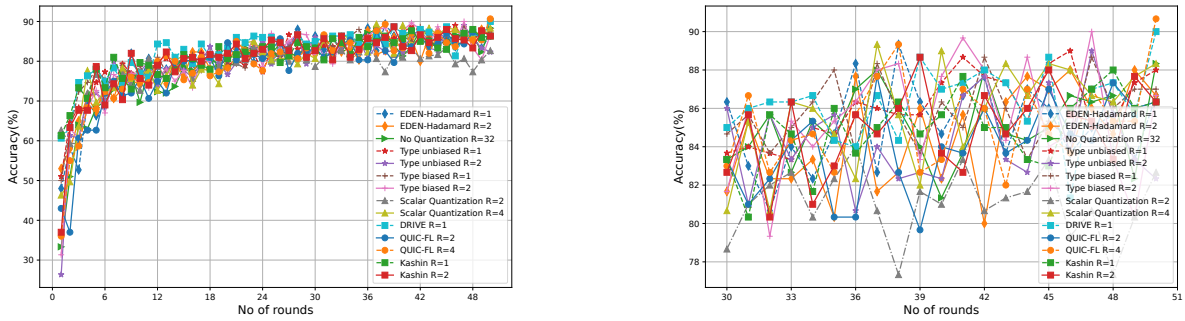
quantization schemes at different rate settings across three benchmark datasets: CIFAR10, MNIST, and Fashion MNIST. Each row lists a specific scheme along with its quantization rate, followed by the maximum and average NMSE values for each dataset. Schemes like DRIVE and EDEN display modest NMSE values even at lower rates of 1 or 2. In contrast, methods such as `Scalar_quantize` yield significantly higher NMSE as the rate decreases, which indicates a clear trade-off between compression and accuracy. Furthermore, the QUIC-FL,



(a) Fashion MNIST Accuracy Plot

(b) Fashion MNIST Accuracy Plot (Zoomed-in)

Figure 10: Performance of various algorithms for federated classification over the Fashion MNIST dataset with testing done at the server.



(a) Fashion MNIST Accuracy Plot

(b) Fashion MNIST Accuracy Plot (Zoomed-in)

Figure 11: Performance of various algorithms for federated classification over the Fashion MNIST dataset evaluated over the clients' test datasets.

Kashin, and Type-based unbiased quantization schemes exhibit varied performance; importantly, both Kashin and our Type-based unbiased quantization do not use shared randomness. This key design choice simplifies the implementation while maintaining competitive performance.

The table 3 presents a detailed comparison of the average quantization time per parameter (in microseconds) for various quantization schemes at different rate settings across the CIFAR10, MNIST, and Fashion MNIST datasets. In our experiments, we measure the quantization time for each scheme by capturing the elapsed time required to process the gradient tensor; this is done by recording the time immediately before and after the quantization operation. Notably, the table shows that some schemes run much faster even under high compression, while others take longer due to their greater computational complexity. All simulations were performed on a workstation equipped with dual AMD EPYC 7452 processors, 128GB of RAM, and an NVIDIA A6000 GPU. For the quantization algorithms in Vargaftik et al. (2021, 2022); Safaryan et al. (2022); Ben-Basat et al. (2024), we used the implementations provided by Ben-Basat et al. (2024).

Scheme Name	Rate	CIFAR10		MNIST		Fashion MNIST	
		Max	Avg	Max	Avg	Max	Avg
DRIVE	1	1.4500	0.8800	1.6250	1.0940	1.6310	0.8860
EDEN	1	0.1130	0.1070	0.0790	0.0780	0.0800	0.0780
EDEN	2	0.0250	0.0240	0.0180	0.0180	0.0180	0.0180
QUIC-FL	1	0.2720	0.2670	0.1890	0.1870	0.1900	0.1870
QUIC-FL	2	0.0390	0.0390	0.0270	0.0270	0.0270	0.0270
QUIC-FL	4	0.0017	0.0017	0.0012	0.0012	0.0012	0.0012
Kashin	1	0.8260	0.6090	1.0930	0.8500	1.0930	0.8690
Kashin	2	0.0780	0.0510	0.0860	0.0720	0.0870	0.0720
Type-biased	1	0.1100	0.0830	0.0240	0.0047	0.0270	0.0160
Type-biased	2	0.0160	0.0110	0.0025	0.0017	0.0020	0.0018
Type-unbiased	1	0.1970	0.1490	0.0400	0.0170	0.0400	0.0360
Type-unbiased	2	0.0320	0.0230	0.0050	0.0034	0.0042	0.0038
Scalar	1	8394.6690	2935.6560	2.4970	2.4970	173.7800	173.7800
Scalar	2	52.2530	10.2040	35.4480	23.0550	31.7990	24.1450
Scalar	4	8.5330	0.5350	0.8720	0.5040	1.8820	0.7210

Table 2: NMSE for various quantization schemes observed when simulating communication-constrained federated classification on CIFAR10, MNIST, and Fashion-MNIST. CIFAR10 uses 300 rounds, for MNIST and Fashion MNIST it runs for 50 rounds. "Average" NMSE is the mean over rounds, and "Max" NMSE is the highest error recorded.

Scheme Name	Rate	MNIST (μs)	Fashion MNIST (μs)	CIFAR10 (μs)
DRIVE	1	5.7896	5.7876	6.1127
EDEN	1	0.4253	0.4637	0.6575
EDEN	2	0.4318	0.4586	0.6288
QUIC-FL	1	1.0269	1.3437	1.3633
QUIC-FL	2	1.0033	1.0730	1.3622
QUIC-FL	4	0.9569	0.9999	1.4811
Kashin	1	1.1075	1.1000	1.5071
Kashin	2	1.0668	1.0733	1.6878
Type-biased	1	0.0974	0.1134	0.1099
Type-biased	2	0.0835	0.0918	0.1095
Type-unbiased	1	0.0792	0.0715	0.0798
Type-unbiased	2	0.0790	0.0714	0.0927
Scalar	1	0.0633	0.0639	0.0848
Scalar	2	0.0647	0.0619	0.0734
Scalar	4	0.0689	0.0626	0.0717

Table 3: Average quantization time per parameter per round (in microseconds) for various quantization schemes in our federated learning experiments. The values are based on models with 172,554 parameters for MNIST and Fashion MNIST, and 122,626 parameters for CIFAR10.