Wyner-Ziv Estimators: Efficient Distributed Mean Estimation with Side Information

Prathamesh Mayekar
Indian Institute of Science
prathamesh@iisc.ac.in

Ananda Theertha Suresh
Google Research
theertha@google.com

Himanshu Tyagi
Indian Institute of Science
htyagi@iisc.ac.in

Abstract

Communication efficient distributed mean estimation is an important primitive that arises in many distributed learning and optimization scenarios such as federated learning. Without any probabilistic assumptions on the underlying data, we study the problem of distributed mean estimation where the server has access to side information. We propose *Wyner-Ziv estimators*, which are communication and computationally efficient and near-optimal when an upper bound for the distance between the side information and the data is known. In a different direction, when there is no knowledge assumed about the distance between side information and the data, we present an alternative Wyner-Ziv estimator that uses correlated sampling. This latter setting offers *universal recovery guarantees*, and perhaps will be of interest in practice when the number of users is large and keeping track of the distances between the data and the side information may not be possible.

1 Introduction

1.1 Background

Consider the problem of distributed mean estimation for n vectors $\{x_i\}_{i=1}^n$ in $\mathbb{R}^d$, where x_i is available to client i. Each client communicates to a server using a few bits to enable the server to compute the empirical mean

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i. \tag{1}$$

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

This estimation problem has become a crucial primitive for distributed optimization scenarios such as federated learning, where the data is distributed across multiple clients (see Bottou (2010), Kairouz et al. (2019), Konečný et al. (2016), Alistarh et al. (2017), Ramezani-Kebrya et al. (2019), Gandikota et al. (2019), Basu et al. (2019), Seide et al. (2014), Wang et al. (2018), Stich et al. (2018), Wen et al. (2017), Wangni et al. (2018), Lu and De Sa (2020), Vogels et al. (2019), Acharya et al. (2019)). One of the main bottlenecks in such distributed scenarios is the significant communication cost incurred due to client communication at each iteration of the distributed algorithm. This has spurred a recent line of work which seeks to design quantizers to express x_is using a low precision and, yet, enable the server to compute a high accuracy estimate of $\bar{x}$ (see Suresh et al. (2017), Konečný and Richtárik (2018), Chen et al. (2020), Huang et al. (2019), Mayekar and Tyagi (2020b), Safaryan et al. (2020), Albasyoni et al. (2020), and the references therein).

Most of the recent works on distributed mean estimation focus on the setting where the server must estimate the sample mean based on the client vectors, and nothing else. However, in practice, the server may also have access to some side information. For example, consider the task of training a machine learning model based on remote client data as well as some publicly accessible data. At each iteration, the server communicates its global model to the client, based on which the clients compute their updates (the gradient estimates based on their local data), compress them, and then send them to the server. The server may choose to compute its own update using the publicly available dataset to complement the updates from the client. In a related setting, the server can use the previously received gradients as side information for the next gradients expected from the clients. Similarly, distributed mean estimation with side information can be used for variance reduction in other problems such as power iteration or

The full version of this paper Mayekar et al. (2020) contains the proofs of all results discussed here, as well as other additional details.

parallel SGD (*cf.* Davies et al. (2020)).

Motivated by these observations, for the distributed mean estimation problem described at the start of the section, we study the setting in which the server has access to the side information $\{y_i\}_{i=1}^n$ in $\mathbb{R}^d$, in addition to the communication from clients. Here, y_i can be viewed as server's initial estimate (guess) of x_i. We emphasize that the side information y_i is available only to the sever and can, therefore, be used for estimating the mean at the server, but is not available to the clients while quantizing the updates $\{x_i\}_{i=1}^n$.

1.2 The model

Consider the input $\mathbf{x} := (x_1, \ldots, x_n)$ and the side information $\mathbf{y} := (y_1, \ldots, y_n)$. The clients use a communication protocol to send r bits each about their observed vector to the server. For the ease of implementation, we restrict to non interactive protocols. Specifically, we allow *simultaneous message passing* (SMP) protocols $\pi = (\pi_1, ..., \pi_n)$ where the communication $C_i = \pi_i(x_i, U) \in \{0,1\}^r$ of client[1] i, $i \in [n]$, can only depend on its local observation x_i and public randomness U. Note that the clients are not aware of side information $\mathbf{y}$, which is available only to the server. In effect, the message C_i is obtained by *quantizing* x_i using an appropriately chosen randomized quantizer. Denoting the overall communication by $C^n := (C_1, C_2, ..., C_n)$, the server uses the transcript (C^n, U) of the protocol and the side information $\mathbf{y}$ to form the estimate of the sample mean[2] $\hat{\bar{x}} = \hat{\bar{x}}(C^n, U, \mathbf{y})$; see Figure 1 for a depiction of our setting. We call such a π an *r-bit SMP protocol* with input $(\mathbf{x}, \mathbf{y})$ and output $\hat{\bar{x}}$.

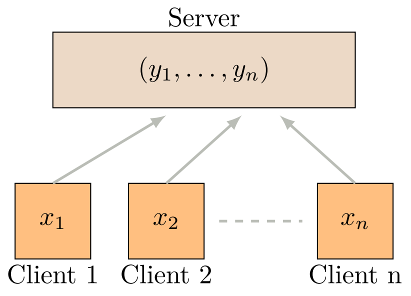

Figure 1: Problem setting of mean estimation with side information

We measure the performance of protocol π for inputs $\mathbf{x}$ and $\mathbf{y}$ and output $\hat{\bar{x}}$ using mean squared error (MSE) given by

$$\mathcal{E}(\pi, \mathbf{x}, \mathbf{y}) := \mathbb{E}\left[\|\hat{\bar{x}} - \bar{x}\|_2^2\right],$$

where the expectation is over the public randomness U and $\bar{x}$ is given in (1). We study the MSE of protocols for $\mathbf{x}$ and $\mathbf{y}$ such that the Euclidean distance between x_i and y_i is at most Δ_i, i.e.,

$$\|x_i - y_i\|_2 \le \Delta_i, \quad \forall\, i \in [n]. \tag{2}$$

Denoting $\boldsymbol{\Delta} := (\Delta_1, \ldots, \Delta_n)$, we are interested in the performance of our protocols for the following two settings:

1. The *known* $\boldsymbol{\Delta}$ setting, where Δ_i is known to client i and the server;

2. The *unknown* $\boldsymbol{\Delta}$ setting, where Δ_is are unknown to everyone.

In both these settings, we seek to find efficient r-bit quantizers for x_i that will allow accurate sample mean estimation. In the known $\boldsymbol{\Delta}$ setting, the quantizers of different clients can be chosen using the knowledge of $\boldsymbol{\Delta}$; in the unknown $\boldsymbol{\Delta}$ setting, they must be fixed irrespective of $\boldsymbol{\Delta}$.

In another direction, we distinguish the *low-precision* setting of $r \le d$ from the *high-precision* setting of $r > d$. The former is perhaps of more relevance for federated learning and high-dimensional distributed optimization, while the latter has received a lot of attention in the information theory literature on rate-distortion theory.

As a benchmark, we recall the result for distributed mean estimation with no side-information from Suresh et al. (2017). When all x_is lie in the Euclidean ball of radius 1, Suresh et al. (2017) showed that the minmax MSE in the no side-information case is

$$\Theta\left(\frac{d}{nr}\right). \tag{3}$$

1.3 Our contributions

Drawing on ideas from distributed quantization problem in information theory (*cf.* Wyner and Ziv (1976)), specifically the Wyner-Ziv problem, we present *Wyner-Ziv estimators* for distributed mean estimation. In the known $\boldsymbol{\Delta}$ setting, for a fixed $\boldsymbol{\Delta}$, and the low-precision setting of $r \le d$, we propose an *r-bit SMP protocol* π^*_{k} which satisfies[3]

$$\mathcal{E}(\pi^*_{\mathrm{k}}, \mathbf{x}, \mathbf{y}) = O\left(\sum_{i=1}^n \frac{\Delta_i^2}{n} \cdot \frac{d \log\log n}{nr}\right),$$

for all $\mathbf{x}$ and $\mathbf{y}$ satisfying (2). Thus, in the case where all x_is lie in the Euclidean ball of radius 1, we improve

[1] $[n] := \{1, \ldots, n\}$.
[2] While side information y_i is associated with client i, we do not enforce this association in our general formulation at this point.

[3] We denote by $\log(\cdot)$ logarithm to the base 2 and by $\ln(\cdot)$ logarithm to the base e.

upon the optimal estimator for distributed mean estimation (3) in the regime $\sum_{i=1}^{n} \frac{\Delta_i^2 \log\log n}{n} \leq 1$. Our estimator is motivated by the classic Wyner-Ziv problem, and hence, we refer to it as the *Wyner-Ziv estimator*. The details of the algorithm are given in Section 3.3.

Our protocol uses the same (randomized) r-bit quantizer for each client's data and simply uses the sample mean of the quantized vectors as the estimate for $\bar{x}$. Furthermore, the common quantizer used by the clients is efficient and has nearly linear time-complexity of $O(d\log d)$. Our proposed quantizer first applies a random rotation using the randomized Walsh-Hadamard transform (see Ailon and Chazelle (2006)) to the input vectors x_i at client i and the side information vector y_i at the server. This ensures that the Δ_i upper bound on the ℓ_2 distance of x_i and y_i is converted to roughly a $\Delta_i/\sqrt{d}$ upper bound on the ℓ_∞ distance between x_i and y_i. This then enables us to use efficient one-dimensional quantizers for each coordinate of the x_i, which can now operate with the knowledge that the server knows a y_i with each coordinate within roughly $\Delta_i/\sqrt{d}$ of x_i's coordinates.

Moreover, we show that this protocol π_{k}^* has optimal (worst-case) MSE up to an $O(\log\log n)$ factor. That is, we show that for any other r-bit SMP protocol π for $r \leq d$, we can find $\mathbf{x}$ and $\mathbf{y}$ satisfying (2) such that

$$\mathcal{E}(\pi, \mathbf{x}, \mathbf{y}) = \Omega\left(\min_{i\in\{1,\dots,n\}} \Delta_i^2 \cdot \frac{d}{nr}\right).$$

In the unknown $\boldsymbol{\Delta}$ setting, we propose a protocol π_{u}^* which adapts to the unknown distance Δ_i between x_i and y_i and, remarkably, provides MSE guarantees dependent on $\boldsymbol{\Delta}$. Specifically, for the low-precision setting of $r \leq d$, the protocol satisfies[4]

$$\mathcal{E}(\pi_u^*, \mathbf{x}, \mathbf{y}) = O\left(\sum_{i=1}^{n} \frac{\Delta_i}{n} \cdot \frac{d \ln^* d}{nr}\right),$$

for all $\mathbf{x}$ and $\mathbf{y}$ in the unit Euclidean ball $\mathcal{B} := \{x \in \mathbb{R}^d : \|x\|_2 \leq 1\}$ and satisfying (2). Thus, we improve upon the optimal estimator for the no side information counterpart (3) in the regime $\sum_{i=1}^{n} \frac{\Delta_i \ln^* d}{n} \leq 1$. Once again, the quantizer employed by the protocol is efficient and has nearly linear time-complexity of $O(d\log d)$. At the heart of our proposed quantizer is the technique of correlated sampling from Holenstein (2009) which enables us to derive a $\boldsymbol{\Delta}$ dependent MSE bound.

Due to space constraints, we defer details of some of the results to the supplementary material. We leave the reader with a summary of key takeaways from these results below:

1. High-precision setting. Both our quantizers can be extended to the high-precision regime of $r > d$. The quantizer for the known $\boldsymbol{\Delta}$ setting directly extends by using r/d bits per dimension. The MSE of the SMP protocol using this quantizer for all the clients is only a factor of $\log n + r/d$ from the lower bound in Davies et al. (2020). The quantizer for the unknown $\boldsymbol{\Delta}$ setting can be extended by sending the "type" of the communication vector, following an idea proposed in Mayekar and Tyagi (2020a). The MSE of the SMP protocol using this quantizer for all the clients falls as $2^{-r/d\ln^* d}$ as opposed to d/r that can be obtained using naive extensions of our quantizer.

2. Gaussian Wyner-Ziv. Finally, in a different direction, we revisit the classic Gaussian rate-distortion problem (*cf.* Oohama (1997)) in information theory. In this problem, the encoder observing an Gaussian vector X wants to send it to a decoder observing a correlated Gaussian vector Y using r bits. Using the quantizer developed in the known $\boldsymbol{\Delta}$ setting, we obtain an efficient scheme for this classic problem which requires a minuscule excess rate over the optimal asymptotic rate. Our scheme for this classic problem is very interesting for two reasons: the first that it gives almost optimal result while using "covering" for each coordinate separately. All existing schemes rely on high-dimensional covering constructed using structured codes; and the second that we do not require the distribution to be exactly Gaussian and subgaussianity suffices.

1.4 Prior work

The known $\boldsymbol{\Delta}$ setting described above was first considered in Davies et al. (2020). The scheme of Davies et al. (2020) relies on lattice quantizers with information theoretically optimal covering radius. Explicit lattices to be used and computationally efficient decoding is not provided.

In contrast, we provide explicit computationally efficient protocols for both low- and high-precision settings. Also, we establish lower bounds showing the optimality of our quantizer upto a multiplicative factor of $\log\log n$ in the low-precision regime of $r \leq d$. In comparison, the scheme of Davies et al. (2020) is off by a factor of $\frac{d}{r}$ from this lower bound. Thus, when $r \ll d$, our scheme performs significantly better than that in Davies et al. (2020). We remark that the unknown $\boldsymbol{\Delta}$ setting, which is perhaps more important in certain applications where estimating the distance of side information of each client is infeasible, has not been considered before.

In the classic information theoretic setting, related

[4] We denote by $\ln^*(a)$ the minimum number of iterated logarithms to the base e that must be applied to a to make it less than 1.

problems of quantization with side information at the decoder have been considered in rate-distortion theory starting with the seminal work of Wyner and Ziv (Wyner and Ziv, 1976). Practical codes for settings where the observations are generated from known distributions have been constructed using channel codes; see, for instance, Korada and Urbanke (2010); Ling et al. (2012); Liu and Ling (2015); Pradhan and Ramchandran (2003); Zamir et al. (2002). However, these codes are computationally too expensive for our setting, cannot be directly used for our distribution-free setup, and are designed for the high-precision setting of $r > d$. We remark that the scheme proposed in Davies et al. (2020) is similar to lattice schemes in Ling et al. (2012); Liu and Ling (2015); Zamir et al. (2002).

The version of the distributed mean estimation problem with no side information at the server has been extensively studied. For any protocol in this setting operating with a precision constraint of $r \leq d$ bits per client, using a strong data processing inequality from Duchi et al. (2014), Suresh et al. (2017) shows a lower bound on MSE of $\Omega\left(\frac{d}{nr}\right)$, when all x_is lie in the Euclidean ball of radius one. Suresh et al. (2017) propose a rotation based uniform quantization scheme which matches this lower bound up to a factor of $\log \log d$ for any precision constraint r. This upper bound is further improved by a random rotation based adaptive quantizer in Mayekar and Tyagi (2020b) to a much tighter $\log \log^* d$ factor. For a precision constraint of $r = \Theta(d)$, the variable-length quantizers proposed in Suresh et al. (2017), Alistarh et al. (2017), Ramezani-Kebrya et al. (2019) as well as the fixed-length quantizers in Mayekar and Tyagi (2020a), Gandikota et al. (2019) are orderwise optimal.

Finally, we note that one of the central components of our proposed algorithms is the idea of randomly rotating the data using the randomized Walsh-Hadamard transform. This idea was first used in Ailon and Chazelle (2006) for a Fast Johnson Lindenstrauss transform. Hadad and Erez (2016), Suresh et al. (2017), and, more recently, Mayekar and Tyagi (2020b) have all used this idea for quantization without any side-information.

Our results for the known $\boldsymbol{\Delta}$ setting are provided in Section 3 and for the unknown $\boldsymbol{\Delta}$ setting are provided in Section 4. Before presenting these results, we review some preliminaries in the next section.

2 Preliminaries and the structure of our protocols

While our lower bound for the known $\boldsymbol{\Delta}$ setting holds for an arbitrary SMP protocol, both the protocols we propose in this paper, for the known $\boldsymbol{\Delta}$ and the unknown $\boldsymbol{\Delta}$ settings, have a common structure. We use r-bit quantizers to form estimates of x_is at the server and then compute the sample mean of the estimates of x_is. To describe our protocols and facilitate our analysis, we begin by concretely defining the distributed quantizers needed for this problem. Further, we present a simple result relating the performance of the resulting protocol to the parameters of the quantizer.

An r-bit quantizer Q for input vectors in $\mathcal{X} \subset \mathbb{R}^d$ and side information $\mathcal{Y} \subset \mathbb{R}^d$ consists of randomized mappings[5] $(Q^{\mathsf{e}}, Q^{\mathsf{d}})$ with the encoder mapping $Q^{\mathsf{e}} : \mathcal{X} \to \{0,1\}^r$ used by the client to quantize and the decoder mapping $Q^{\mathsf{d}} : \{0,1\}^r \times \mathcal{Y} \to \mathcal{X}$ used by the server to aggregate quantized vectors. The overall quantizer Q is given by the composition mapping $Q(x,y) = Q^{\mathsf{d}}((Q^{\mathsf{e}}(x), y)$.

In our protocols, for input $\mathbf{x}$ and side information $\mathbf{y}$, client i uses the encoder Q_i^{e} for the r-bit quantizer Q_i to send $Q_i^{\mathsf{e}}(x_i)$. The server uses $Q_i^{\mathsf{e}}(x_i)$ and y_i to form the estimate $\hat{x}_i = Q_i(x_i, y_i)$ of x_i. We assume that the randomness used in quantizers Q_i for different i is independent, whereby $\hat{x}_i$ are independent of each other for different i. Then server finally forms the estimate of the sample mean as

$$\hat{\bar{x}} := \frac{1}{n}\sum_{i=1}^{n} \hat{x}_i. \tag{4}$$

For any quantizer Q, the following two quantities will determine its performance when used in our distributed mean estimation protocol:

$$\alpha(Q;\Delta) := \sup_{x\in\mathcal{X}, y\in\mathcal{Y}: \|x-y\|_2\leq\Delta} \mathbb{E}\left[\|Q(x,y)-x\|_2^2\right],$$
$$\beta(Q;\Delta) := \sup_{x\in\mathcal{X}, y\in\mathcal{Y}: \|x-y\|_2\leq\Delta} \|\mathbb{E}\left[Q(x,y)-x\right]\|_2^2,$$

where the expectation is over the randomization of the quantizer. Note that $\alpha(Q;\Delta)$ can be interpreted as the worst-case MSE and $\beta(Q,\Delta)$ the worst-case bias over $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ such that $\|x-y\|_2 \leq \Delta$.

The result below will be very handy for our analysis.

Lemma 2.1. *For $\mathbf{x} \in \mathcal{X}^n$ and $\mathbf{y} \in \mathcal{Y}^n$ satisfying (2) and r-bit quantizers Q_i, $i \in [n]$, using independent randomness for different $i \in [n]$, the estimate $\hat{\bar{x}}$ in (4) and the sample mean $\bar{x}$ in (1) satisfy*

$$\mathbb{E}\left[\|\hat{\bar{x}} - \bar{x}\|_2^2\right] \leq \sum_{i=1}^{n} \frac{\alpha(Q_i;\Delta_i)}{n^2} + \sum_{i=1}^{n} \frac{\beta(Q_i;\Delta_i)}{n}.$$

[5]We can use public randomness U for randomizing.

3 Distributed mean estimation with known $\boldsymbol{\Delta}$

In this section, we present our Wyner-Ziv estimator for the known $\boldsymbol{\Delta}$ setting. As described in Section 2, we use the the same (randomized) quantizer across all the clients and form the estimate of sample mean as in (4). We only need to define the common quantizer used by all the clients, which we do in Section 3.3. In Sections 3.1 and 3.2, we provide the basic building blocks of our final quantizer. Further, in Section 3.4, we derive a lower bound for the worst-case MSE that establishes the near-optimality of our protocol. Throughout we restrict to the low-precision setting of $r \le d$.

3.1 Modulo Quantizer (MQ)

The first subroutine used by our larger quantizer is the *Modulo Quantizer* (MQ). MQ is a one dimensional distributed quantizer that can be applied to the input $x \in \mathbb{R}$ with side information $y \in \mathbb{R}$. We give an input parameter Δ' to MQ where $|x-y| \le \Delta'$. In addition to Δ', MQ also has the resolution parameter k and the lattice parameter ε as inputs.

For an appropriate ε to be specified later, we consider the lattice $\mathbb{Z}_\varepsilon = \{\varepsilon z : z \in \mathbb{Z}\}$. For a given input x, the encoder $Q_{\mathtt{M}}^{\mathtt{e}}$ finds the closest points in $\mathbb{Z}_\varepsilon$ larger and smaller than x. Then, one of these points is sampled randomly to get an unbiased estimate of x. The sampled point will be of the form $\tilde{z}\varepsilon$, where $\tilde{z}$ is in $\mathbb{Z}$. We note that the chosen point $\tilde{z}$ satisfies $\varepsilon\mathbb{E}[\tilde{z}] = x$ and $|x - \varepsilon\tilde{z}| < \varepsilon$, almost surely. The encoder sends $w = \tilde{z} \mod k$ to the decoder, which requires $\log k$ bits.

Upon receiving this w, the decoder $Q^{\mathtt{d}}$ looks at the set $\mathbb{Z}_{w,\varepsilon} = \{(zk+w)\cdot\varepsilon : z \in \mathbb{Z}\}$ and decodes the point closest to y, which we denote by $Q_{\mathtt{M}}(x,y)$. Note that declaring y will already give a MSE of less than Δ. A useful property of this decoder is that its output is always within a bounded distance from y; namely, since in Step 1 of Alg. 3 we look for the closest point to y in the lattice $Z_{w,\varepsilon} := \{(zk+w)\cdot\varepsilon : z \in \mathbb{Z}\}$, the output $Q_{\mathtt{M}}(x,y)$ satisfies $|Q_{\mathtt{M}}(x,y) - y| \le k\varepsilon$, almost surely.

We summarize MQ in Alg. 2 and 3.

```
Require: Input $x \in \mathbb{R}$, Parameters $k$, $\Delta'$, and $\varepsilon$
1: Compute $z_u = \lceil x/\varepsilon \rceil$, $z_l = \lfloor x/\varepsilon \rfloor$
2: Generate $\tilde{z} = \begin{cases} z_u, & w.p.\ x/\varepsilon - z_l \\ z_l, & w.p.\ z_u - x/\varepsilon \end{cases}$
3: Output: $Q_{\mathtt{M}}^{\mathtt{e}}(x) = \tilde{z} \bmod k$
```

Algorithm 2: Encoder $Q_{\mathtt{M}}^{\mathtt{e}}(x)$ of MQ

```
Require: Input $w \in \{0, \dots, k-1\}$, $y \in \mathbb{R}$
1: Compute $\hat{z} = \arg\min\{|(zk+w)\cdot\varepsilon - y| : z \in \mathbb{Z}\}$
2: Output: $Q_{\mathtt{M}}^{\mathtt{d}}(w,y) = (\hat{z}k+w)\varepsilon$
```

Algorithm 3: Decoder $Q_{\mathtt{M}}^{\mathtt{d}}(w,y)$ of MQ

The result below provides performance guarantees for $Q_{\mathtt{M}}$. The key observation is that the output $Q_{\mathtt{M}}(x,y)$ of the quantizer equals $\tilde{z}\varepsilon$ with $\tilde{z}$ found at the encoder, if ε is set appropriately.

Lemma 3.1. *Consider the Modulo Quantizer $Q_{\mathtt{M}}$ described in Alg. 2 and 3 with parameter ε set to satisfy*

$$k\varepsilon \ge 2(\varepsilon + \Delta'). \tag{5}$$

Then, for every x, y in $\mathbb{R}$ such that $|x-y| \le \Delta'$, the output $Q_{\mathtt{M}}(x,y)$ of MQ satisfies

$$\mathbb{E}[Q_{\mathtt{M}}(x,y)] = x \text{ and } |Q_{\mathtt{M}}(x,y) - x| \le \varepsilon, \text{ almost surely.}$$

In particular, we can set $\varepsilon = 2\Delta'/(k-2)$, to get $|Q_{\mathtt{M}}(x,y) - x| \le 2\Delta'/(k-2)$. Furthermore, the output of $Q_{\mathtt{M}}$ can be described in $\log k$ bits.

We close with a remark that the modulo operation used in our scheme is the simplest and easily implementable version of classic coset codes obtained using nested lattices used in distributed quantization (*cf.* Forney (1988); Liu (2016); Zamir et al. (2002)) and was used in Davies et al. (2020) as well.

3.2 Rotated Modulo Quantizer (RMQ)

We now describe *Rotated Modulo Quantizer (RMQ)*. RMQ and the subsequent quantizers in this section will be used to quantize input vector x in $\mathbb{R}^d$ with side information y in $\mathbb{R}^d$, where $\|x-y\|_2 \le \Delta$. RMQ first preprocesses the input x and side information y by randomly rotating them and then simply applies MQ for each coordinate. For rotation, we multiply both x and y with a matrix R given by

$$R = \frac{1}{\sqrt{d}} \cdot HD, \tag{6}$$

where H is the $d \times d$ Walsh-Hadamard Matrix (see Horadam (2012))[6] and D is a diagonal matrix with each diagonal entry generated uniformly from $\{-1,+1\}$. Note that we use public randomness[7] to generate the same D at both the encoder and the decoder. We formally describe the quantizer in[8] Alg. 4 and 5.

[6]We assume that d is a power of 2. If it isn't, we can pad the vector by zeros to make it a power of 2; even in the worst-case, this only doubles the required bits.

[7]In practice, this can be implemented by using the same seed for pseudo-random number generator at encoder and decoder.

[8]We denote by $(e_1, ..., e_d)$ the standard basis of $\mathbb{R}^d$.

Remark 1. We remark that the vector $R(x-y)$ has zero mean subgaussian coordinates with a variance factor of Δ^2/d. This implies that for all coordinates i in $[d]$, we have

$$P\left(|R\,(x-y)\,(i)| \geq \Delta'\right) \leq 2e^{-\frac{\Delta'^2 d}{2\Delta^2}}$$

(see, for instance, (Boucheron et al., 2013, Theorem 2.8)). This observation allows us to use $\Delta' \approx \Delta/\sqrt{d}$ for MQ applied to each coordinate.

```
Require: Input $x \in \mathbb{R}^d$, Parameters $k$ and $\Delta'$
1: Sample $R$ as in (6) using public randomness
2: $x' = Rx$
3: Output: $Q^{\mathtt{e}}_{\mathtt{M},R}(x) = [Q^{\mathtt{e}}_{\mathtt{M}}(x'(1)),\ldots,Q^{\mathtt{e}}_{\mathtt{M}}(x'(d)]^T$
   using parameters $k$, $\varepsilon$, and $\Delta'$ for $Q^{\mathtt{e}}_{\mathtt{M}}$ of Alg. 2
```

Algorithm 4: Encoder $Q^{\mathtt{e}}_{\mathtt{M},R}(x)$ of RMQ

```
Require: Input $w \in \{0,\ldots,k-1\}^d$, $y \in \mathbb{R}^d$,
         Parameters $k$ and $\Delta'$
1: Get $R$ from public randomness.
2: $y' = Ry$
3: Output: $Q^{\mathtt{d}}_{\mathtt{M},R}(w,y) = R^{-1}\sum_{i\in[d]} Q^{\mathtt{d}}_{\mathtt{M}}(w(i),y'(i))e_i$
   using parameters $k$, $\varepsilon$, and $\Delta'$ for $Q^{\mathtt{d}}_{\mathtt{M}}$ of Alg. 3,
```

Algorithm 5: Decoder $Q^{\mathtt{d}}_{\mathtt{M},R}(w,y)$ of RMQ

Lemma 3.2. *Fix $\Delta \geq 0$. Let $Q_{\mathtt{M},R}$ be RMQ described in Alg. 4 and 5. Then, for[9] $k \geq 4$, $\delta \in (0,\Delta)$, $\Delta' = \sqrt{6(\Delta^2/d)\ln(\Delta/\delta)}$ and the parameter ε of MQ set to $\varepsilon = 2\Delta'/(k-2)$, we have for $\mathcal{X} = \mathcal{Y} = \mathbb{R}^d$ that*

$$\alpha(Q_{\mathtt{M},R};\Delta) \leq \frac{24\,\Delta^2}{(k-2)^2}\ln\frac{\Delta}{\delta} + 154\,\delta^2 \quad \textit{and}$$
$$\beta(Q_{\mathtt{M},R};\Delta) \leq 154\,\delta^2.$$

Furthermore, the output of quantizer $Q_{\mathtt{M},R}$ can be described in $d\log k$ bits.

Remark 2. The choice of Δ' in the first statement of the Lemma 3.2 is based on Remark 1. We note that δ is a parameter to control the bias incurred by our quantizer. By setting $\Delta' = \Delta$ we can get an unbiased quantizer, but it only recovers the performance obtained by simply using MQ for each coordinate, an algorithm considered in Davies et al. (2020) as well.

[9] In the proof, we provide a general bound which holds for all k.

3.3 Subsampled RMQ: A Wyner-Ziv quantizer for $\mathbb{R}^d$

Our final quantizer is a modification of RMQ of previous section where we make the precision less than r bits by randomly sampling a subset of coordinates. Specifically, note that $Q^{\mathtt{e}}_{\mathtt{M},R}(x)$ sends d binary strings of $\log k$ bits each. We reduce the resolution by sending only a random subset S of these strings. This subset is sampled using shared randomness and is available to the decoder, too. Note that $Q^{\mathtt{d}}_{\mathtt{M},R}$ applies $Q^{\mathtt{d}}_{\mathtt{M}}$ to these strings separately; now, we use $Q^{\mathtt{d}}_{\mathtt{M}}$ to decode the entries in S alone. We describe the overall quantizer in Alg. 6 and 7.

```
Require: Input $x \in \mathbb{R}$, Parameters $k$, $\Delta'$, and $\mu$
1: Sample $S \subset [d]$ u.a.r. from all subsets of $[d]$
   of cardinality $\mu d$ and sample $R$ as in (6) using
   public randomness
2: Output: $Q^{\mathtt{e}}_{\mathtt{WZ}}(x) = \{Q^{\mathtt{e}}_{\mathtt{M}}(Rx(i)) : i \in S\}$ using
   parameters $k$, $\varepsilon$, and $\Delta'$ for $Q^{\mathtt{e}}_{\mathtt{M}}$ of Alg. 2
```

Algorithm 6: Encoder $Q^{\mathtt{e}}_{\mathtt{WZ}}(x)$ of subsampled RMQ

```
Require: Input $w \in \{0,\ldots,k-1\}^{\mu d}$, $y \in \mathbb{R}$
1: Get $S$ and $R$ from public randomness
2: Compute $\tilde{x} = (Q^{\mathtt{d}}_{\mathtt{M}}(w(i),Ry(i)), i \in S)$ using pa-
   rameters $k$, $\varepsilon$, and $\Delta'$ for $Q^{\mathtt{d}}_{\mathtt{M}}$ of Alg. 3
3: $\hat{x}_R = \frac{1}{\mu}\sum_{i\in S}\left(\tilde{x}(i) - Ry(i)\right)e_i + Ry$
4: Output: $Q^{\mathtt{d}}_{\mathtt{WZ}}(w,y) = R^{-1}\hat{x}_R$
```

Algorithm 7: Decoder $Q^{\mathtt{d}}_{\mathtt{WZ}}(w,y)$ of subsampled RMQ

Remark 3. We remark that, typically, when implementing random sampling, we set the unsampled components to 0. However, to get Δ dependent bounds on MSE, we set the unsampled coordinates to the corresponding coordinate of side information and center our estimate appropriately to only have small bias.

The result below relates the performance of our final quantizer $Q_{\mathtt{WZ}}$ to that of $Q_{\mathtt{M},R}$, which was already analysed in the previous section.

Lemma 3.3. *Fix $\Delta > 0$. Let $Q_{\mathtt{WZ}}$ and $Q_{\mathtt{M},R}$ be the quantizers described in Alg. 6 and 7 and Alg. 4 and 5, respectively. Then, for $\mu d \in [d]$, we have for $\mathcal{X} = \mathcal{Y} = \mathbb{R}^d$ that*

$$\alpha(Q_{\mathtt{WZ}};\Delta) \leq \frac{\alpha(Q_{\mathtt{M},R};\Delta)}{\mu} + \frac{\Delta^2}{\mu} \quad \textit{and}$$
$$\beta(Q_{\mathtt{WZ}};\Delta) = \beta(Q_{\mathtt{M},R};\Delta).$$

Furthermore, the output of quantizer $Q_{\mathtt{WZ}}$ can be described in $\mu d\log k$ bits.

We are now equipped to prove our first main result. Our protocol $\pi^*_{\mathtt{k}}$ uses $Q_{\mathtt{WZ}}$ for each client as described in Section 2 and forms the estimate $\hat{\hat{x}}$ as in (4). We set the parameters needed for $Q_{\texttt{WZ}}$ in Alg. 6 and 7 as follows: For client i, we set the parameters of MQ as

$$\delta = \frac{\Delta_i}{\sqrt{n}}, \quad \log k = \left\lceil \log(2 + \sqrt{12 \ln n}) \right\rceil,$$
$$\Delta' = \sqrt{6(\Delta_i^2/d)\ln(\Delta_i/\delta)}, \quad \varepsilon = 2\Delta'/(k-2), \qquad (7)$$

and set the parameter μ as

$$\mu d = \left\lfloor \frac{r}{\log k} \right\rfloor. \qquad (8)$$

We characterize the resulting error performance in the next result.

Theorem 3.4. *For a $n \geq 2$, a fixed $\mathbf{\Delta} = (\Delta_1, ..., \Delta_n)$, and $d \geq r \geq 2 \left\lceil \log(2 + \sqrt{12 \ln n}) \right\rceil$, the protocol π_k^* with parameters as set in (7) and (8) is an r-bit protocol which satisfies*

$$\mathcal{E}(\pi_k^*, \mathbf{x}, \mathbf{y})$$
$$\leq \left(79 \lceil \log(2 + \sqrt{12 \ln n}) \rceil + 26\right) \left(\sum_{i=1}^{n} \frac{\Delta_i^2}{n} \cdot \frac{d}{nr} \right),$$

for all $\mathbf{x}, \mathbf{y}$ satisfying (2).

Remark 4. We note that by using MQ for each coordinate without rotating (or even with rotation using R as above) and with $\Delta' = \Delta_i$ yields MSE less than

$$O\left(\sum_{i=1}^{n} \frac{\Delta_i^2}{n} \cdot \frac{d \log d}{nr} \right),$$

for $r \leq d$. Thus, our approach above allows us to remove the $\log d$ factor at the cost of a (milder for large d) $\log \log n$ factor.

Thus, as can be seen from the lower bound presented in Theorem 3.5 below, our Wyner-Ziv estimator π_k^* is nearly optimal. Moreover, $Q_{\texttt{WZ}}$ can be efficiently implemented as both the encoding and decoding procedures have nearly-linear time complexity[10] of $O(d \log d)$. Finally, the encoder and the decoder of $Q_{\texttt{WZ}}$ with a parameter μ uses at the most $d + \mu d \log(1/\mu)$ bits of public randomness, where d bits are used for generating R and $\mu d \log(1/\mu)$ are used for subsampling.

3.4 Lower bound

We now present a lower bound on the MSE incurred by any SMP protocol using r bits per client. The proof relies on the strong data processing inequality in Duchi et al. (2014) and is similar in structure to the lower bound for distributed mean estimation without side-information in Suresh et al. (2017).

[10]The most expensive operation at both the encoder and decoder of this estimator is the Hadamard matrix multiplication operation, which requires $d \log d$ real operations.

Theorem 3.5. *Fix $\mathbf{\Delta} = (\Delta_1, \ldots, \Delta_n)$. There exists a universal constant $c < 1$ such that for any r-bit SMP protocol π, with $r \leq cd$, there exists input $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{2d}$ satisfying (2) and such that*

$$\mathcal{E}(\pi, \mathbf{x}, \mathbf{y}) \geq c \min_{i \in [d]} \Delta_i^2 \cdot \frac{d}{nr}.$$

4 Distributed mean estimation for unknown $\mathbf{\Delta}$

Finally, we present our Wyner-Ziv estimator for the unknown $\mathbf{\Delta}$ setting. We first, in Section 4.1, describe the idea of correlated sampling from Holenstein (2009), which will serve as an essential building block for all our quantizers in this section. We then build towards our final quantizer, described in 4.4, by first describing its simpler versions in Section 4.2 and 4.3. Once again, we restrict to the low-precision setting of $r \leq d$.

4.1 The correlated sampling idea

Suppose we have two numbers x and y lying in $[0, 1]$. A 1-bit unbiased estimator for x is the random variable $\mathbb{1}_{\{U \leq x\}}$, where U is a uniform random variable in $[0, 1]$. The variance of such an estimator is $x - x^2$. We consider a variant of this estimator given by:

$$\hat{X} = \mathbb{1}_{\{U \leq x\}} - \mathbb{1}_{\{U \leq y\}} + y, \qquad (9)$$

where, like before, U is a uniform random variable in $[0, 1]$. Such an estimator still uses only 1-bit of information related to x. It is easy to check that this estimator unbiased estimator of x, namely $\mathbb{E}\left[\hat{X}\right] = x$. The variance of this estimator is given by

$$\texttt{Var}(\hat{X}) = \mathbb{E}\left[(\hat{X} - x)^2\right] = |x - y| - (x - y)^2,$$

which is lower than that of the former quantizer when x is close to y. We build-on this basic primitive to obtain a quantizer with MSE bounded above by a $\mathbf{\Delta}$-dependent expression, without requiring the knowledge of $\mathbf{\Delta}$.

4.2 Distance Adaptive Quantizer (DAQ)

DAQ and subsequent quantizers in this Section will be described for input x and side information y lying in $\mathbb{R}^d$. The first component of our quantizer, DAQ, which uses (9) and incorporates the correlated sampling idea discussed earlier. Both the encoder and the decoder of DAQ use the same d uniform random variables $\{U(i)\}_{i=1}^d$ between $[-1, 1]$, which are generated using public randomness. At the encoder, each coordinate of vector x is encoded to the bit $\mathbb{1}_{\{U(i) \leq x(i)\}}$. At the decoder, using the bits received from the encoder, side information y, and the public randomness $\{U(i)\}_{i=1}^d$, we first compute bits $\mathbb{1}_{\{U(i)\le y(i)\}}$ for each $i \in [d]$. Then, the estimate of x is formed as follows:

$$Q_{\mathtt{D}}(x, y) = \sum_{i=1}^{d} \left(\mathbb{1}_{\{U(i)\le x(i)\}} - \mathbb{1}_{\{U(i)\le y(i)\}}\right) e_i + y.$$

We formally describe the quantizer in Alg. 8 and 9.

```
Require: Input $x \in \mathbb{R}^d$
 1: Sample $U(i) \sim Unif[-1,1], \forall i \in [d]$
 2: $\tilde{x} = \sum_{i=1}^d \mathbb{1}_{\{U(i)\le x(i)\}} \cdot e_i$
 3: Output: $Q_{\mathtt{D}}^{\mathtt{e}}(x) = \tilde{x}$, where $\tilde{x}$ is viewed as binary vector of length $d$
```

Algorithm 8: Encoder $Q_{\mathtt{D}}^{\mathtt{e}}(x)$ of DAQ

```
Require: Input $w \in \{0,1\}^d$, $y \in \mathbb{R}^d$,
 1: Get $U(i), \forall i \in [d]$, using public randomness
 2: Set $\tilde{y} = \sum_{i=1}^d \mathbb{1}_{\{U(i)\le y(i)\}} \cdot e_i$
 3: Output: $Q_{\mathtt{D}}^{\mathtt{d}}(w, y) = 2(w - \tilde{y}) + y$, where $w$ is viewed as a vector in $\mathbb{R}^d$
```

Algorithm 9: Decoder $Q_{\mathtt{D}}^{\mathtt{d}}(w, y)$ of DAQ

The next result characterizes the performance for DAQ.

Lemma 4.1. *Let $Q_{\mathtt{D}}$ denote DAQ described in Algorithms 8 and 9. Then, for $\mathcal{X} = \mathcal{Y} = \mathcal{B}$ and every $\Delta > 0$, we have*

$$\alpha(Q_{\mathtt{D}}; \Delta) \le 2\Delta\sqrt{d} \quad \text{and} \quad \beta(Q_{\mathtt{D}}; \Delta) = 0.$$

Furthermore, the output of quantizer $Q_{\mathtt{D}}$ can be described in d bits.

4.3 Rotated Distance Adaptive Quantizer (RDAQ)

Next, we proceed as for the known $\boldsymbol{\Delta}$ setting and add a preprocessing step of rotating x and y using random matrix R of (6), which is sampled using shared randomness. We remark that here random rotation is used to exploit the subgaussianity of the rotated x and y, whereas in RMQ of previous section it was used to exploit the subgaussianity of $x - y$. After this rotation step, we proceed with a quantizer similar to DAQ, but we quantize each coordinate at multiple "scales." We describe this step in detail below.

Using multiple scales. In DAQ, we considered each coordinate x to be anywhere between $[-1, 1]$ and used one uniform random variable for each coordinate. Now, we will use h independent uniform random variables for each coordinate, each corresponding to a different scale $[-M_j, M_j]$, $j \in \{0, 1, 2, \ldots, h-1\}$. For convenience, we abbreviate $[h]_0 := \{0, 1, 2, \ldots, h-1\}$.

Specifically, let $U(i, j)$ be distributed uniformly over $[-M_j, M_j]$, independently for different $i \in [d]$ and different $j \in [h]_0$. The values M_js correspond to different scales and are set, along with h, as follows: For all $j \in [h]_0$,

$$M_j^2 := \frac{6}{d} \cdot e^{*j}, \quad \log h := \lceil \log(1 + \ln^*(d/6)) \rceil, \quad (10)$$

where e^{*j} denotes the jth iteration of e given by $e^{*0} := 1, \quad e^{*1} := e, \quad e^{*j} := e^{e^{*(j-1)}}$. All the dh uniform random variables are generated using public randomness and are available to both the encoder and the decoder.

The intervals $[-M_j, M_j]$ are designed to minimize the MSE of our quantizer by tuning its "resolution" to the "scale" of the input, and while still ensuring unbiased estimates. This idea of using multiple intervals $[-M_j, M_j]$ for quantizing the randomly rotated vector is from Mayekar and Tyagi (2020b), where it was used for the case with no side information.

Multiscale DAQ. After rotation, we proceed as in DAQ, except that we use different scale M_j for different coordinates. Ideally, for the ith coordinate, we would like to use $M_{z^*(i)}$, where $z^*(i)$ is the smallest index such that both $Rx(i)$ and $Ry(i)$ lie in $[-M_{z^*(i)}, M_{z^*(i)}]$. However, since y is not available to the encoder, we simply resort to sending the smallest value $z(i)$ which is the smallest index such that $Rx(i) \in [-M_{z(i)}, M_{z(i)}]$ and apply the encoder of DAQ h times to compress x at all scales, *i.e.*, we send h bits $(\mathbb{1}_{\{U(i,j)\le Rx(i)\}}, j \in [h]_0)$.

Thus, the overall number of bits used by RDAQ's encoder is $d \cdot (h + \lceil \log h \rceil)$. At RDAQ's decoder, using $z(i)$, we compute the smallest index $z^*(i)$ containing both $Rx(i)$ and $Ry(i)$. In effect, the decoder emulates the decoder for DAQ applied to Ry, but for scale $M_{z^*(i)}$. The encoding and decoding algorithm of RDAQ are described in Alg. 10 and 11, respectively.

Then, the quantized output $Q_{\mathtt{D},R}$ corresponding to input vector x and side-information y is

$$Q_{\mathtt{D},R}(x, y) = R^{-1}\left[\sum_{i=1}^{d} 2M_{z^*(i)} \left(\mathbb{1}_{\{U(i,z^*(i))\le Rx(i)\}} - \mathbb{1}_{\{U(i,z^*(i))\le Ry(i)\}}\right) + Ry\right].$$

We remark that since rotated coordinates $Rx(i)$ and $Ry(i)$ have subgaussian tails, with very high probability $M_{z^*(i)}$ will be much less than 1, which helps in reducing the overall MSE significantly. The performance of the algorithm is characterized below.

```
Require: Input $x \in \mathcal{B}$
1: Sample $U(i,j) \sim Unif[-M_j, M_j]$, $i \in [d], j \in [h]_0$, and sample $R$ as in(6) using public randomness.
2: $x_R = Rx$
3: for $i \in [d]$ do
       $z(i) = \min\{j \in [h]_0 : |x_R(i)| \leq M_j\}$
4: for $j \in [h]_0$ do
       $\tilde{x}_j = \sum_{i=1}^{d} \mathbb{1}_{\{U(i,j) \leq x_R(i)\}} e_i$
5: Output: $Q^{\mathtt{e}}_{\mathtt{D},R}(x) = ([\tilde{x}_0, \ldots, \tilde{x}_{h-1}], z)$, where we view $\tilde{x}_j$s as binary vectors
```

Algorithm 10: Encoder $Q^{\mathtt{e}}_{\mathtt{D},R}(x)$ at for RDAQ

```
Require: Input $(w, z) \in \{0,1\}^{d\times h} \times [h]_0^d$ and $y \in \mathcal{B}$
1: Get $U(i,j)$, $i \in [d]$, $j \in [h]_0$, and $R$ using public randomness.
2: $y_R = Ry$
3: for $i \in [d]$ do
       $z'(i) = \min\{j \in \{[h]_0\} : |y_R(i)| \leq M_j\}$
       $z^*(i) = \max\{z(i), z'(i)\}$
4: $w' = \sum_{i\in[d]} 2M_{z^*(i)} \left(w(i, z^*(i)) - \mathbb{1}_{\{U(i,z^*(i)) \leq y_R\}}\right)$
5: $\hat{x}_R = w' + Ry$
6: Output: $Q^{\mathtt{d}}_{\mathtt{D},R}(w, y) = R^{-1}\hat{x}_R$.
```

Algorithm 11: Decoder $Q^{\mathtt{d}}_{\mathtt{D},R}(x)$ for RDAQ

Lemma 4.2. *Let $Q_{\mathtt{D},R}$ be RDAQ described in Alg. 10 and 11. Then, for $\mathcal{X} = \mathcal{Y} = \mathcal{B}$ and every $\Delta > 0$, we have*

$$\alpha(Q_{\mathtt{D},R}; \Delta) \leq 16\sqrt{3}\Delta \quad \text{and} \quad \beta(Q_{\mathtt{D},R}; \Delta) = 0.$$

Furthermore, the output of quantizer Q can be described in $d(h + \log h)$ bits.

4.4 Subsampled RDAQ: A universal Wyner-Ziv quantizer for unit Euclidean ball

Finally, we bring down the precision of RDAQ to r, as before for the known $\mathbf{\Delta}$ setting, by retaining the output of RDAQ for only coordinates $i \in S$, where S is generated uniformly at random from all subsets of $[d]$ of cardinality μd using public randomness. Specifically, we execute Alg. 10 and 11 with S replacing $[d]$ and multiplying w' in Step 4 of Alg. 11 by normalization factor of $d/|S|$. The output of the resulting encoder is

$$Q^{\mathtt{e}}_{\mathtt{WZ},u}(x) = \{Q^{\mathtt{e}}_{\mathtt{D},R}(x)(i) : i \in S\}, \tag{11}$$

where $Q^{\mathtt{e}}_{\mathtt{D},R}(x)(i)$ represents the encoded bits $([\tilde{x}_0(i), \ldots, \tilde{x}_{h-1}(i)], z(i))$ for the ith coordinate using RDAQ, and the output of the resulting decoder is

$$Q_{\mathtt{WZ},u}(x, y) = R^{-1}\left[\frac{1}{\mu}\sum_{i\in S} 2M_{z^*(i)}\left(\mathbb{1}_{\{U(i,z^*(i)) \leq Rx(i)\}} - \mathbb{1}_{\{U(i,z^*(i)) \leq Ry(i)\}}\right) + Ry\right]. \tag{12}$$

Lemma 4.3. *Let $Q_{\mathtt{WZ},u}$ be the quantizers described in (11) and (12) and $Q_{\mathtt{D},R}$ be RDAQ described in Alg. 10 and 11. Then, for $\mu d \in [d]$, $\mathcal{X} = \mathcal{Y} = \mathcal{B}$, and every $\Delta > 0$, we have*

$$\alpha(Q_{\mathtt{WZ},u}; \Delta) \leq \frac{\alpha(Q_{\mathtt{D},R}; \Delta)}{\mu} \quad \text{and} \quad \beta(Q_{\mathtt{WZ},u}; \Delta) = 0.$$

Furthermore, the output of quantizer $Q_{\mathtt{WZ},u}$ can be described in $\mu d(h + \log h)$ bits.

We are now equipped to prove our second main result. Our protocol $\pi^*_{\mathtt{u}}$ uses $Q_{\mathtt{WZ},u}$ for each client as described in Section 2 and forms the estimate $\hat{\bar{x}}$ as in (4). Unlike for the known $\mathbf{\Delta}$ setting, we now use the same parameters for $Q_{\mathtt{WZ},u}$ for all clients, given by

$$\mu d = \left\lfloor \frac{r}{h + \log h} \right\rfloor. \tag{13}$$

Theorem 4.4. *For $d \geq r \geq 2(h + \log h)$ and h given in (10), the r-bit protocol π^*_u with parameters as set in (13) satisfies*

$$\mathcal{E}(\pi^*_u, \mathbf{x}, \mathbf{y}) \leq (128\sqrt{3}\,(1 + \ln^*(d/6))) \left(\sum_{i\in[n]} \frac{\Delta_i}{n} \cdot \frac{d}{nr}\right),$$

for all $\mathbf{x}, \mathbf{y}$ satisfying (2), for every $\mathbf{\Delta} = (\Delta_1, ..., \Delta_n)$.

The Wyner-Ziv estimator π^*_u is universal in $\mathbf{\Delta}$: it operates without the knowledge of the distance between the input and the side information and yet gets MSE depending on $\mathbf{\Delta}$. Moreover, it can be efficiently implemented as both the encoding and the decoding procedures of $Q_{\mathtt{WZ},u}$ have nearly linear time complexity of $O(d \log d)$. Finally, the encoder and the decoder of $Q_{\mathtt{WZ},u}$ with a parameter μ uses at the most $d + \mu d \log(1/\mu)$ bits of public randomness for generating R and subsampling. Additionally, $Q_{\mathtt{WZ},u}$ uses μdh uniform random variables for correlated sampling.

Acknowledgment

Prathamesh Mayekar would like to thank Shubham Jha for discussions on the Gaussian Wyner-Ziv problem.

The work of Prathamesh Mayekar was supported by the Ph.D. fellowship from Wipro Ltd. The work of Himanshu Tyagi was supported in part by a grant from the Robert Bosch Center for Cyberphysical Systems (RBCCPS), Indian Institute of Science, Bengaluru, and in part by the grant from the Department of Science and Technology (DST), India, under Grant EMR/2016/002569.

References

Acharya, J., De Sa, C., Foster, D. J., and Sridharan, K. (2019). Distributed Learning with Sublinear Communication. *arXiv:1902.11259*.

Ailon, N. and Chazelle, B. (2006). Approximate nearest neighbors and the fast johnson-lindenstrauss transform. *Proceedings of the ACM symposium on Theory of computing (STOC'06)*.

Albasyoni, A., Safaryan, M., Condat, L., and Richtárik, P. (2020). Optimal gradient compression for distributed and federated learning. *arXiv:2010.03246*.

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*.

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*.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. *Proceedings of COMPSTAT'2010*.

Boucheron, S., Lugosi, G., and Massart, P. (2013). *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press.

Chen, W.-N., Kairouz, P., and Özgür, A. (2020). Breaking the communication-privacy-accuracy trilemma. *arXiv:2007.11707*.

Davies, P., Gurunathan, V., Moshrefi, N., Ashkboos, S., and Alistarh, D. (2020). Distributed variance reduction with optimal communication. *arXiv:2002.09268*.

Duchi, J. C., Jordan, M. I., Wainwright, M. J., and Zhang, Y. (2014). Optimality guarantees for distributed statistical estimation. *arXiv:1405.0782*.

Forney, G. D. (1988). Coset codes. i. introduction and geometrical classification. *IEEE Transactions on Information Theory*, 34(5):1123–1151.

Gandikota, V., Kane, D., Maity, R. K., and Mazumdar, A. (2019). vqsgd: Vector quantized stochastic gradient descent. *arXiv:1911.07971*.

Hadad, R. and Erez, U. (2016). Dithered quantization via orthogonal transformations. *IEEE Transactions on Signal Processing*, 64:5887–5900.

Holenstein, T. (2009). Parallel repetition: Simplification and the no-signaling case. *Proceedings of the ACM symposium on Theory of computing (STOC'09)*.

Horadam, K. J. (2012). *Hadamard matrices and their applications*. Princeton university press.

Huang, Z., Yilei, W., Yi, K., et al. (2019). Optimal sparsity-sensitive bounds for distributed mean estimation. *Advances in Neural Information Processing Systems*.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2019). Advances and open problems in federated learning. *arXiv:1912.04977*.

Konečný, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. *arXiv:1610.05492*.

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

Korada, S. B. and Urbanke, R. L. (2010). Polar codes are optimal for lossy source coding. *IEEE Transactions on Information Theory*, 56(4):1751–1768.

Ling, C., Gao, S., and Belfiore, J. (2012). Wyner-ziv coding based on multidimensional nested lattices. *IEEE Transactions on Communications*, 60(5):1328–1335.

Liu, L. (2016). Polar codes and polar lattices for efficient communication and source quantization. *Ph.D. Thesis*.

Liu, L. and Ling, C. (2015). Polar lattices are good for lossy compression. *CoRR*, abs/1501.05683.

Lu, Y. and De Sa, C. (2020). Moniqua: Modulo quantized communication in decentralized sgd. *arXiv:2002.11787*.

Mayekar, P., Suresh, A. T., and Tyagi, H. (2020). Wyner-ziv estimators: Efficient distributed mean estimation with side information. *arXiv:2011.12160*.

Mayekar, P. and Tyagi, H. (2020a). Limits on gradient compression for stochastic optimization. *Proceedings of the IEEE International Symposium of Information Theory (ISIT' 20)*.

Mayekar, P. and Tyagi, H. (2020b). RATQ: A universal fixed-length quantizer for stochastic optimization. *arXiv.org:1908.08200*.

Oohama, Y. (1997). Gaussian multiterminal source coding. *IEEE Transactions on Information Theory*, 43(6):1912–1923.

Pradhan, S. S. and Ramchandran, K. (2003). Distributed source coding using syndromes (discus): design and construction. *IEEE Transactions on Information Theory*, 49(3):626–643.

Ramezani-Kebrya, A., Faghri, F., and Roy, D. M. (2019). Nuqsgd: Improved communication efficiency for data-parallel sgd via nonuniform quantization. *arXiv:1908.06077*.

Safaryan, M., Shulgin, E., and Richtárik, P. (2020). Uncertainty principle for communication compression in distributed and federated learning and the search for an optimal compressor. *arXiv:2002.08958*.

Seide, F., Fu, H., Droppo, J., Li, G., and Yu, D. (2014). 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. *Fifteenth Annual Conference of the International Speech Communication Association*.

Stich, S. U., Cordonnier, J.-B., and Jaggi, M. (2018). Sparsified sgd with memory. *Advances in Neural Information Processing Systems 31*.

Suresh, A. T., Yu, F. X., Kumar, S., and McMahan, H. B. (2017). Distributed mean estimation with limited communication. *Proceedings of the International Conference on Machine Learning (ICML' 17)*, 70:3329–3337.

Vogels, T., Karimireddy, S. P., and Jaggi, M. (2019). Powersgd: Practical low-rank gradient compression for distributed optimization. *arXiv:1905.13727*.

Wang, H., Sievert, S., Liu, S., Charles, Z., Papailiopoulos, D., and Wright, S. (2018). Atomo: Communication-efficient learning via atomic sparsification. *Advances in Neural Information Processing Systems*, pages 9850–9861.

Wangni, J., Wang, J., Liu, J., and Zhang, T. (2018). Gradient sparsification for communication-efficient distributed optimization. *Advances in Neural Information Processing Systems*.

Wen, W., Xu, C., Yan, F., Wu, C., Wang, Y., Chen, Y., and Li, H. (2017). TernGrad: Ternary gradients to reduce communication in distributed deep learning. *Advances in Neural Information Processing Systems*, pages 1509–1519.

Wyner, A. and Ziv, J. (1976). The rate-distortion function for source coding with side information at the decoder. *IEEE Transactions on information Theory*, 22(1):1–10.

Zamir, R., Shamai, S., and Erez, U. (2002). Nested linear/lattice codes for structured multiterminal binning. *IEEE Transactions on Information Theory*, 48(6):1250–1276.