Randomized Distributed Mean Estimation: Accuracy vs. Communication

Citation for published version:

Digital Object Identifier (DOI):
10.3389/fams.2018.00062

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Publisher's PDF, also known as Version of record

Published In:
Frontiers in Applied Mathematics and Statistics

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
Randomized Distributed Mean Estimation: Accuracy vs. Communication

Jakub Konečný* and Peter Richtárik¹,²,³

¹ School of Mathematics, The University of Edinburgh, Edinburgh, United Kingdom, ² Moscow Institute of Physics and Technology, Dolgoprudny, Russia, ³ King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

We consider the problem of estimating the arithmetic average of a finite collection of real vectors stored in a distributed fashion across several compute nodes subject to a communication budget constraint. Our analysis does not rely on any statistical assumptions about the source of the vectors. This problem arises as a subproblem in many applications, including reduce-all operations within algorithms for distributed and federated optimization and learning. We propose a flexible family of randomized algorithms exploring the trade-off between expected communication cost and estimation error. Our family contains the full-communication and zero-error method on one extreme, and an $\epsilon$-bit communication and $O\left(\frac{1}{\epsilon n}\right)$ error method on the opposite extreme. In the special case where we communicate, in expectation, a single bit per coordinate of each vector, we improve upon existing results by obtaining $O(\frac{r}{n})$ error, where $r$ is the number of bits used to represent a floating point value.

Keywords: communication efficiency, distributed mean estimation, accuracy-communication tradeoff, gradient compression, quantization

1. INTRODUCTION

We address the problem of estimating the arithmetic mean of $n$ vectors, $X_1, \ldots, X_n \in \mathbb{R}^d$, stored in a distributed fashion across $n$ compute nodes, subject to a constraint on the communication cost.

In particular, we consider a star network topology with a single server at the centre and $n$ nodes connected to it. All nodes send an encoded (possibly via a lossy randomized transformation) version of their vector to the server, after which the server performs a decoding operation to estimate the true mean

$$X \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} X_i.$$

The purpose of the encoding operation is to compress the vector so as to save on communication cost, which is typically the bottleneck in practical applications.

To better illustrate the setup, consider the naive approach in which all nodes send the vectors without performing any encoding operation, followed by the application of a simple averaging decoder by the server. This results in zero estimation error at the expense of maximum communication cost of $ndr$ bits, where $r$ is the number of bits needed to communicate a single floating point entry/coordinate of $X_i$. 
This operation appears as a computational primitive in numerous cases, and the communication cost can be reduced at the expense of accuracy. Our proposal for balancing accuracy and communication is in practice relevant for any application that uses the MPI_Gather or MPI_Allgather routines [1], or their conceptual variants, for efficient implementation and can tolerate inexactness in computation, such as many algorithms for distributed optimization.

1.1. Background and Contributions
The distributed mean estimation problem was recently studied in a statistical framework where it is assumed that the vectors $X_i$ are independent and identically distributed samples from some specific underlying distribution. In such a setup, the goal is to estimate the true mean of the underlying distribution [2–5]. These works formulate lower and upper bounds on the communication cost needed to achieve the minimax optimal estimation error.

In contrast, we do not make any statistical assumptions on the source of the vectors, and study the trade-off between expected communication costs and mean square error of the estimate. Arguably, this setup is a more robust and accurate model of the distributed mean estimation problems arising as subproblems in applications such as reduce-all operations within algorithms for distributed and federated optimization [6–10]. In these applications, the averaging operations need to be done repeatedly throughout the iterations of a master learning/optimization algorithm, and the vectors $\{X_i\}$ correspond to updates to a global model/variable. In such cases, the vectors evolve throughout the iterative process in a complicated pattern, typically approaching zero as the master algorithm converges to optimality. Hence, their statistical properties change, which renders fixed statistical assumptions not satisfied in practice.

For instance, when training a deep neural network model in a distributed environment, the vector $X_i$ corresponds to a stochastic gradient based on a minibatch of data stored on node $i$. In this setup we do not have any useful prior statistical knowledge about the high-dimensional vectors to be aggregated. It has recently been observed that when communication cost is high, which is typically the case for commodity clusters, and even more so in a federated optimization framework, it is can be very useful to sacrifice on estimation accuracy in favor of reduced communication [11, 12].

In this paper we propose a parametric family of randomized methods for estimating the mean $\mu$, with parameters being a set of probabilities $p_{ij}$ for $i = 1, \ldots, n$ and $j = 1, 2, \ldots, d$ and node centers $\mu_i \in \mathbb{R}^n$ for $i = 1, 2, \ldots, n$. The exact meaning of these parameters is explained in section 3. By varying the probabilities, at one extreme, we recover the exact method described, enjoying zero estimation error at the expense of full communication cost. At the opposite extreme are methods with arbitrarily small expected communication cost, which is achieved at the expense of suffering an exploding estimation error. Practical methods appear somewhere on the continuum between these two extremes, depending on the specific requirements of the application at hand. Suresh et al. [13] propose a method combining a pre-processing step via a random structured rotation, followed by randomized binary quantization. Their quantization protocol arises as a suboptimal special case of our parametric family of methods\(^1\).

To illustrate our results, consider the special case presented in Example 7, in which we choose to communicate a single bit per element of $X_i$ only. We then obtain an $O\left(\frac{d}{\epsilon^2}\right)$ bound on the mean square error, where $r$ is number of bits used to represent a floating point value, and $R = \frac{1}{n} \sum_{i=1}^n \|X_i - \mu_i\|^2$ with $\mu_i \in \mathbb{R}$ being the average of elements of $X_i$, and $1$ the all-ones vector in $\mathbb{R}^d$. Note that this bound improves upon the performance of the method of Suresh et al. [13] in two aspects. First, the bound is independent of $d$, improving from logarithmic dependence, as stated in Remark 4 in detail. Further, due to a preprocessing rotation step, their method requires $O(d \log d)$ time to be implemented on each node, while our method is linear in $d$. This and other special cases are summarized in Table 1 in section 5.

While the above already improves upon the state of the art, the improved results are in fact obtained for a suboptimal choice of the parameters of our method (constant probabilities $p_{ij}$, and node centers fixed to the mean $\mu_i$). One can decrease the MSE further by optimizing over the probabilities and/or node centers (see section 6). However, apart from a very low communication cost regime in which we have a closed form expression for the optimal probabilities, the problem needs to be solved numerically, and hence we do not have expressions for how much improvement is possible. We illustrate the effect of fixed and optimal probabilities on the trade-off between communication cost and MSE experimentally on a few selected datasets in section 6 (see Figure 1).

Remark 1. Since the initial version of this work, an updated version of Suresh et al. [13] contains a rate similar to Example 7, using variable length coding. That work also formulates lower bounds, which are attained by both their and our results. Other works that were published since, such as [14, 15], propose algorithms that can also be represented as a particular choice of protocols $\alpha$, $\beta$, $\gamma$, demonstrating the versatility of our proposal.

1.2. Outline
In section 2 we formalize the concepts of encoding and decoding protocols. In section 3 we describe a parametric family of randomized (and unbiased) encoding protocols and give a simple formula for the mean squared error. Subsequently, in section 4 we formalize the notion of communication cost, and describe several communication protocols, which are optimal under different circumstances. We give simple instantiations of our protocol in section 5, illustrating the trade-off between communication costs and accuracy. In section 6 we address the question of the optimal choice of parameters of our protocol. Finally, in section 7 we comment on possible extensions we leave out to future work.

2. THREE PROTOCOLS
In this work we consider (randomized) encoding protocols $\alpha$, communication protocols $\beta$, and decoding protocols $\gamma$ using which

\(^1\)See Remark 4.

---

This text is a part of a larger research paper on randomized distributed mean estimation, discussing the trade-offs between communication and estimation accuracy in distributed learning and optimization frameworks. The authors propose a parametric family of randomized methods to estimate the mean of high-dimensional vectors, achieving a balance between communication cost and estimation error, and illustrate their methods through specific examples and case studies. The paper includes theoretical analyses, experimental results, and discussions on the applications and implications of their proposed methods.
the averaging is performed inexact as follows. Node $i$ computes a (possibly stochastic) estimate of $X_i$ using the encoding protocol, which we denote $Y_i = \alpha(X_i) \in \mathbb{R}^d$, and sends it to the server using communication protocol $\beta$. By $\beta(Y_i)$ we denote the number of bits that need to be transferred under $\beta$. The server then estimates $X$ using the decoding protocol $\gamma$ of the estimates:

$$Y \overset{\text{def}}{=} \gamma(Y_1, \ldots, Y_n).$$

The objective of this work is to study the trade-off between the (expected) number of bits that need to be communicated, and the accuracy of $Y$ as an estimate of $X$.

In this work we focus on encoders which are unbiased, in the following sense.

Definition 2.1 (Unbiased and Independent Encoder): We say that encoder $\alpha$ is unbiased if $E_{\alpha}[\alpha(X_i)] = X_i$ for all $i = 1, 2, \ldots, n$. We say that it is independent, if $\alpha(X_i)$ is independent from $\alpha(X_j)$ for all $i \neq j$.

Example 1 (Identity Encoder). A trivial example of an encoding protocol is the identity function: $\alpha(X_i) = X_i$. It is both unbiased and independent. This encoder does not lead to any savings in communication that would be otherwise infeasible through.

Another examples of unbiased and independent Encoders include the protocols introduced in section 3, or other existing techniques [12, 14, 15].

We now formalize the notion of accuracy of estimating $X$ via $Y$. Since $Y$ can be random, the notion of accuracy will naturally be probabilistic.

Definition 2.2 (Estimation Error / Mean Squared Error): The mean squared error of protocol $(\alpha, \gamma)$ is the quantity

$$MSE_{\alpha,\gamma}(X_1, \ldots, X_n) = E_{\alpha,\gamma}
\left[\|Y - X\|^2\right]$$

To illustrate the above concept, we now give a few examples:

Example 2 (Averaging Decoder). If $\gamma$ is the averaging function, i.e., $\gamma(Y_1, \ldots, Y_n) = \frac{1}{n} \sum_{i=1}^{n} Y_i$, then

$$MSE_{\alpha,\gamma}(X_1, \ldots, X_n) = \frac{1}{n^2} E_{\alpha}
\left[\left\|\sum_{i=1}^{n} \alpha(X_i) - X\right\|^2\right].$$

The next example generalizes the identity encoder and averaging decoder.

Example 3 (Linear Encoder and Inverse Linear Decoder). Let $A : \mathbb{R}^d \rightarrow \mathbb{R}^d$ be linear and invertible. Then we can set $Y_i = \alpha(X_i) \overset{\text{def}}{=} AX_i$ and $\gamma(Y_1, \ldots, Y_n) \overset{\text{def}}{=} A^{-1} \left(\frac{1}{n} \sum_{i=1}^{n} Y_i\right)$. If $A$ is random, then $\alpha$ and $\gamma$ are random (e.g., a structured random rotation, see [16]). Note that

$$\gamma(Y_1, \ldots, Y_n) = \frac{1}{n} \sum_{i=1}^{n} A^{-1} Y_i = \frac{1}{n} \sum_{i=1}^{n} X_i = X,$$

and hence the MSE of $(\alpha, \gamma)$ is zero.

We shall now prove a simple result for unbiased and independent encoders used in subsequent sections.

Lemma 2.3 (Unbiased and Independent Encoder + Averaging Decoder): If the encoder $\alpha$ is unbiased and independent, and $\gamma$ is the averaging decoder, then

$$MSE_{\alpha,\gamma}(X_1, \ldots, X_n) = \frac{1}{n^2} E_{\alpha}
\left[\left\|\sum_{i=1}^{n} Y_i - X\right\|^2\right]$$

Proof: Note that $E_{\alpha}[Y_i] = X_i$ for all $i$. We have

$$MSE_{\alpha,\gamma}(X_1, \ldots, X_n) = E_{\alpha}
\left[\|Y - X\|^2\right]$$

(\quad) $$\frac{1}{n^2} E_{\alpha}
\left[\left\|\sum_{i=1}^{n} Y_i - X\right\|^2\right]$$

One may wish to define the encoder as a combination of two or more separate encoders: $\alpha(X_i) = \alpha_2(\alpha_1(X_i))$. See Suresh et al. [13] for an example where $\alpha_1$ is a random rotation and $\alpha_2$ is binary quantization.

3. A FAMILY OF RANDOMIZEDENCODING PROTOCOLS

Let $X_1, \ldots, X_n \in \mathbb{R}^d$ be given. We shall write $X_i = (X_i(1), \ldots, X_i(d))$ to denote the entries of vector $X_i$. In addition, with each $i$ we also associate a parameter $\mu_i \in \mathbb{R}$. We refer to $\mu_i$ as the center of data at node $i$, or simply as node center. For now, we assume these parameters are fixed. As a special case, we recover for instance classical binary quantization, see section 5.1. We shall comment on how to choose the parameters optimally in section 6.

We shall define support of $\alpha$ on node $i$ to be the set $S_i \overset{\text{def}}{=} \{j : Y_j(j) \neq \mu_i\}$. We now define two parametric families of randomized encoding protocols. The first results in $S_i$ of random size, the second has $S_i$ of a fixed size.

3.1. Encoding Protocol With Variable-Size Support

With each pair $(i,j)$ we associate a parameter $0 < p_{ij} \leq 1$, representing a probability. The collection of parameters $[p_{ij}, \mu_i]$ defines an encoding protocol $\alpha$ as follows:

$$Y_i(j) = \begin{cases} \frac{X_i(j)}{\mu_i} - \frac{1-p_{ij}}{p_{ij}} \mu_i & \text{with probability } p_{ij}, \\ \mu_i & \text{with probability } 1 - p_{ij}. \end{cases}$$
Proof: Using Lemma 2.3, we have
\[
MSE_\alpha(X_1, \ldots, X_n) = \frac{1}{n^2} \sum_{i=1}^{n} \mathbf{E}_\alpha \left[ \| Y_i - X_i \|^2 \right]
\]
\[
= \frac{1}{n^2} \sum_{i=1}^{n} \mathbf{E}_\alpha \left[ \sum_{j=1}^{d} (Y_{ij} - X_{ij})^2 \right]
\]
\[
= \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{d} \mathbf{E}_\alpha \left[ (Y_{ij} - X_{ij})^2 \right].
\]  \hfill (3)

For any \( i, j \) we further have
\[
\mathbf{E}_\alpha \left[ (Y_{ij} - X_{ij})^2 \right] = p_{ij} \left( \frac{X_{ij}}{p_{ij}} - \frac{1 - p_{ij}}{p_{ij}} \mu_i - X_{ij} \right)^2
\]
\[
+ (1 - p_{ij}) (\mu_i - X_{ij})^2
\]
\[
= (1 - p_{ij}) (\mu_i - X_{ij})^2.
\]
It suffices to substitute the above into (3). \( \square \)

3.2. Encoding Protocol With Fixed-Size Support

Here we propose an alternative encoding protocol, one with deterministic support size. As we shall see later, this results in deterministic communication cost.

Let \( \sigma_k(d) \) denote the set of all subsets of \( \{1, 2, \ldots, d\} \) containing \( k \) elements. The protocol \( \alpha \) with a single integer parameter \( k \) is then working as follows: First, each node \( i \) samples \( D_i \in \sigma_k(d) \) uniformly at random, and then sets
\[
Y_i(j) = \begin{cases} 
\frac{dX_{ij}}{k} - \frac{d-k}{k} \mu_i & \text{if } j \in D_i, \\
\mu_i & \text{otherwise}.
\end{cases}
\]  \hfill (4)

Note that due to the design, the size of the support of \( Y_i \) is always \( k \), i.e., \( |S_i| = k \). Naturally, we can expect this protocol to perform practically the same as the protocol (1) with \( p_{ij} = k/d \), for all \( i, j \).

Lemma 3.4 indeed suggests this is the case. While this protocol admits a more efficient communication protocol (as we shall see in section 4.4), protocol (1) enjoys a larger parameters space, ultimately leading to better MSE. We comment on this tradeoff in subsequent sections.

As for the data-dependent protocol, we prove basic properties. The proofs are similar to those of Lemmas 3.1 and 3.2 and we defer them to Appendix A.

Lemma 3.3 (Unbiasedness): The encoder \( \alpha \) defined in (4) is unbiased. That is, \( \mathbf{E}_\alpha [\alpha(X_i)] = X_i \) for all \( i \). As a result, \( Y \) is an unbiased estimate of the true average: \( \mathbf{E}_\alpha [Y] = X \).

Lemma 3.4 (Mean Squared Error): Let \( \alpha = \alpha(k) \) be encoder defined as in (4). Then
\[
MSE_\alpha(X_1, \ldots, X_n) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{d} \left( \frac{d-k}{k} \mu_i \right)^2.
\]  \hfill (5)

4. COMMUNICATION PROTOCOLS

Having defined the encoding protocols \( \alpha \), we need to specify the way the encoded vectors \( Y_i = \alpha(X_i) \), for \( i = 1, 2, \ldots, n \), are communicated to the server. Given a specific communication protocol \( \beta \), we write \( \beta(Y_i) \) to denote the (expected) number of bits that are communicated by node \( i \) to the server. Since \( Y_i = \alpha(X_i) \) is in general not deterministic, \( \beta(Y_i) \) can be a random variable.

Definition 4.1 (Communication Cost): The communication cost of communication protocol \( \beta \) under randomized encoding \( \alpha \) is the total expected number of bits transmitted to the server:
\[
C_{\alpha, \beta}(X_1, \ldots, X_n) = \mathbf{E}_\alpha \left[ \sum_{i=1}^{n} \beta(\alpha(X_i)) \right].
\]  \hfill (6)

Given \( Y_i \), a good communication protocol is able to encode \( Y_i = \alpha(X_i) \) using a few bits only. Let \( r \) denote the number of bits used...
to represent a floating point number. Let \( r \) be the the number of bits representing \( \mu_i \).

In the rest of this section we describe several communication protocols \( \beta \) and calculate their communication cost.

### 4.1. Naive
Represent \( Y_i = \alpha(X_i) \) as \( d \) floating point numbers. Then for all encoding protocols \( \alpha \) and all \( i \) we have \( \beta(\alpha(X_i)) = dr \), whence

\[
C_{\alpha, \beta} = E_{\alpha} \left[ \sum_{i=1}^{n} \beta(\alpha(X_i)) \right] = ndr.
\]

### 4.2. Varying-Length
We will use a single variable for every element of the vector \( Y_i \), which does not have constant size. The first bit decides whether the value represents \( \mu_i \) or not. If yes, end of variable, if not, next \( r \) bits represent the value of \( Y_i(j) \). In addition, we need to communicate \( \mu_i \), which takes \( r \) bits\(^2\). We thus have

\[
\beta(\alpha(X_i)) = \bar{r} + \sum_{j=1}^{d} (1_{Y_i(j)=\mu_i}) + (r + 1) \times 1_{Y_i(j) \neq \mu_i}), \tag{7}
\]

where \( 1_{c} \) is the indicator function of event \( c \). The expected number of bits communicated is given by

\[
C_{\alpha, \beta} = E_{\alpha} \left[ \sum_{i=1}^{n} \beta(\alpha(X_i)) \right] = n\bar{r} + \sum_{i=1}^{n} \sum_{j=1}^{d} (1 - p_{ij} + (r + 1)p_{ij})
\]

\[
= n\bar{r} + \sum_{i=1}^{n} \sum_{j=1}^{d} (1 + rp_{ij})
\]

In the special case when \( p_{ij} = p > 0 \) for all \( i, j \), we get

\[
C_{\alpha, \beta} = n(\bar{r} + d + pd). \tag{8}
\]

### 4.3. Sparse Communication Protocol for Encoder (1)
We can represent \( Y_i \) as a sparse vector; that is, a list of pairs \( (j, Y_i(j)) \) for which \( Y_i(j) \neq \mu_i \). The number of bits to represent each pair is \( \lfloor \log(d) \rfloor + r \). Any index not found in the list, will be interpreted by server as having value \( \mu_i \). Additionally, we have to communicate the value of \( \mu_i \) to the server, which takes \( r \) bits. We assume that the value \( d \), size of the vectors, is known to the server. Hence,

\[
\beta(\alpha(X_i)) = \bar{r} + \sum_{j=1}^{d} 1_{Y_i(j) \neq \mu_i} \times \lfloor \log(d) \rfloor + r \right).
\]

Summing up through \( i \) and taking expectations, the the communication cost is given by

\[
C_{\alpha, \beta} = E_{\alpha} \left[ \sum_{i=1}^{n} \beta(\alpha(X_i)) \right] = nd\bar{r} + \sum_{i=1}^{n} \sum_{j=1}^{d} p_{ij}. \tag{8}
\]

\(^2\)The distinction here is because \( \mu_i \) can be chosen to be data independent, such as 0, so we don’t have to communicate anything (i.e., \( \bar{r} = 0 \)).

In the special case when \( p_{ij} = p > 0 \) for all \( i, j \), we get

\[
C_{\alpha, \beta} = n(\bar{r} + (\lfloor \log d \rfloor + r)d).
\]

**Remark 3.** A practical improvement upon this could be to (without loss of generality) assume that the pairs \( (j, Y_i(j)) \) are ordered by \( j \), i.e., we have \( \{j_k, Y_i(j_k)\}_{k=1}^{k} \) for some \( k \) and \( j_1 < j_2 < \cdots < j_k \). Further, let us denote \( j_0 = 0 \). We can then use a variant of variable-length quantity [17] to represent the set \( \{(j_k - j_{k-1}, Y_i(j_k))\}_{k=1}^{k} \). With careful design one can hope to reduce the \( \log(d) \) factor in the average case. Nevertheless, this does not improve the worst case analysis we focus on in this paper, and hence we do not delve deeper in this. After the first version of this work was posted on arXiv, such an idea was independently proposed and analyzed in Alistarh et al. [14].

### 4.4. Sparse Communication Protocol for Encoder (4)
We now describe a sparse communication protocol compatible with fixed length encoder defined in (4). Note that the selection of set \( D_i \) is independent of the values \( X_i(j) \) being compressed. We can utilize this fact, and instead of communicating index-value pairs \( (j, Y_i(j)) \) as above, we can only communicate the values \( Y_i(j) \), and the indices they correspond to can be reconstructed from a shared random seed. This lets us avoid the \( \log(d) \) factor in (8). Apart from protocol (4), this idea is also applicable to protocol (1) with uniform probabilities \( p_{ij} \).

In particular, we represent \( Y_i \) as a vector containing the list of the values for which \( Y_i(j) \neq \mu_i \), ordered by \( j \). Additionally, we communicate the value \( \mu_i \) (using \( r \) bits) and a random seed (using \( r_s \) bits), which can be used to reconstruct the indices \( j \), corresponding to the communicated values. Note that for any fixed \( k \) defining protocol (4), we have \( |S_i| = k \). Hence, communication cost is deterministic:

\[
C_{\alpha, \beta} = \sum_{i=1}^{n} \beta(\alpha(X_i)) = n(\bar{r} + r_s) + nkr. \tag{9}
\]

In the case of the variable-size-support encoding protocol (1) with \( p_{ij} = p > 0 \) for all \( i, j \), the sparse communication protocol described here yields expected communication cost

\[
C_{\alpha, \beta} = E_{\alpha} \left[ \sum_{i=1}^{n} \beta(\alpha(X_i)) \right] = n(\bar{r} + r_s) + ndpr. \tag{10}
\]

### 4.5. Binary
If the elements of \( Y_i \) take only two different values, \( Y_i^{\text{min}} \) or \( Y_i^{\text{max}} \), we can use a binary communication protocol. That is, for each node \( i \), we communicate the values of \( Y_i^{\text{min}} \) and \( Y_i^{\text{max}} \) (using \( 2r \) bits), followed by a single bit per element of the array indicating whether \( Y_i^{\text{max}} \) or \( Y_i^{\text{min}} \) should be used. The resulting (deterministic) communication cost is

\[
C_{\alpha, \beta} = \sum_{i=1}^{n} \beta(\alpha(X_i)) = n(2r) + nd. \tag{11}
\]
4.6. Discussion
In the above, we have presented several communication protocols of different complexity. However, it is not possible to claim any of them is the most efficient one. Which communication protocol is the best, depends on the specifics of the used encoding protocol. Consider the extreme case of encoding protocol (1) with \( p_{ij} = 1 \) for all \( i,j \). The naive communication protocol is clearly the most efficient, as all other protocols need to send some additional information.

However, in the interesting case when we consider small communication budget, the sparse communication protocols are the most efficient. Therefore, in the following sections, we focus primarily on optimizing the performance using these protocols.

5. EXAMPLES
In this section, we highlight on several instantiations of our protocols, recovering existing techniques and formulating novel ones. We comment on the resulting trade-offs between communication cost and estimation error.

5.1. Binary Quantization
We start by recovering an existing method, which turns every element of the vectors \( X_i \) into a particular binary representation.

Example 4. If we set the parameters of protocol (1) as \( \mu_i = X_i^{\text{min}} \) and \( p_{ij} = \frac{X_i(j) - X_i^{\text{max}}}{X_i^{\text{max}} - X_i^{\text{min}}} \), where \( \Delta_i \) is the range (for simplicity, that \( \Delta_i \neq 0 \)), we exactly recover the quantization algorithm proposed in Suresh et al. [13]:

\[
Y_i(j) = \begin{cases} 
X_i^{\text{max}} & \text{with probability } \frac{X_i(j) - X_i^{\text{min}}}{X_i^{\text{max}} - X_i^{\text{min}}}, \\
X_i^{\text{min}} & \text{with probability } \frac{X_i^{\text{max}} - X_i(j)}{X_i^{\text{max}} - X_i^{\text{min}}}. 
\end{cases}
\] (12)

Using the formula (2) for the encoding protocol \( \alpha \), we get

\[
\text{MSE}_{\alpha} = \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^d X_i^{\text{max}} - X_i(j) \left( X_i(j) - X_i^{\text{min}} \right)^2 \\
\leq \frac{d}{2n} \cdot \frac{1}{n} \sum_{i=1}^n \|X_i\|^2.
\]

This exactly recovers the MSE bound established in Suresh et al. [13, Theorem 1]. Using the binary communication protocol yields the communication cost of 1 bit per element of \( X_i \), plus a two real-valued scalars (11).

Remark 4. If we use the above protocol jointly with randomized linear encoder and decoder (see Example 3), where the linear transform is the randomized Hadamard transform, we recover the method described in Suresh et al. [13, section 3] which yields improved \( \text{MSE}_\alpha = \frac{2\log d + 2}{n} \cdot \frac{1}{n} \sum_{i=1}^n \|X_i\|^2 \) and can be implemented in \( \mathcal{O}(d \log d) \) time.

5.2. Sparse Communication Protocols
Now we move to comparing the communication costs and estimation error of various instantiations of the encoding protocols, utilizing the deterministic sparse communication protocol and uniform probabilities.

For the remainder of this section, let us only consider instantiations of our protocol where \( p_{ij} = p > 0 \) for all \( i,j \), and assume that the node centers are set to the vector averages, i.e.,

\[
\mu_i = \frac{1}{d} \sum_{j=1}^d X_i(j). 
\]

Denote \( R = \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^d (X_i(j) - \mu_i)^2 \). For simplicity, we also assume that \( |S| = nd \), which is what we can in general expect without any prior knowledge about the vectors \( X_i \).

The properties of the following examples follow from Equations (2) to (10). When considering the communication costs of the protocols, keep in mind that the trivial benchmark is \( C_{\alpha,\beta} = ndr \), which is achieved by simply sending the vectors unmodified. Communication cost of \( C_{\alpha,\beta} = nd \) corresponds to the interesting special case when we use (on average) one bit per element of each \( X_i \).

Example 5 (Full communication). If we choose \( p = 1 \), we get

\[
C_{\alpha,\beta} = n(r_s + \bar{r}) + ndr, \quad \text{MSE}_{\alpha,\gamma} = 0.
\]

In this case, the encoding protocol is lossless, which ensures \( \text{MSE} = 0 \). Note that in this case, we could get rid of the \( n(r_s + \bar{r}) \) factor by using naive communication protocol.

Example 6 (Log MSE). If we choose \( p = 1/\log d \), we get

\[
C_{\alpha,\beta} = n(r_s + \bar{r}) + \frac{ndr}{\log d}, \quad \text{MSE}_{\alpha,\gamma} = \frac{\log(d) - 1}{n} R.
\]

This protocol order-wise matches the MSE of the method in Remark 4. However, as long as \( d > 2 \), this protocol attains this error with smaller communication cost. In particular, this is on expectation less than a single bit per element of \( X_i \). Finally, note that the factor \( R \) is always smaller or equal to the factor \( \frac{1}{n} \sum_{i=1}^n \|X_i\|^2 \) appearing in Remark 4.

Example 7 (1-bit per element communication). If we choose \( p = 1/r \), we get

\[
C_{\alpha,\beta} = n(r_s + \bar{r}) + nd, \quad \text{MSE}_{\alpha,\gamma} = \frac{r - 1}{n} R.
\]

This protocol communicates on expectation single bit per element of \( X_i \) (plus additional \( r_s + \bar{r} \) bits per client), while attaining bound on \( \text{MSE} \) of \( \mathcal{O}(r/n) \). To the best of our knowledge, this is the first method to attain this bound without additional assumptions.

Example 8 (Alternative 1-bit per element communication). If we choose \( p = \frac{d - r_s - \bar{r}}{dr} \), we get

\[
C_{\alpha,\beta} = nd, \quad \text{MSE}_{\alpha,\gamma} = \frac{dr}{d - r_s - \bar{r} - 1} R.
\]

This alternative protocol attains on expectation exactly single bit per element of \( X_i \), with (a slightly more complicated) \( \mathcal{O}(r/n) \) bound on \( \text{MSE} \).

Example 9 (Below 1-bit communication). If we choose \( p = 1/d \), we get

\[
C_{\alpha,\beta} = n(r_s + \bar{r}) + nr, \quad \text{MSE}_{\alpha,\gamma} = \frac{d - 1}{n} R.
\]
This protocol attains the MSE of protocol in Example 4 while at the same time communicating on average significantly less than a single bit per element of X_i.

We summarize these examples in Table 1.

Using the deterministic sparse protocol, there is an obvious lower bound on the communication cost — n(\bar{r} + \bar{r}). We can bypass this threshold by using the sparse protocol, with a data-independent choice of \mu, such as 0, setting \bar{r} = 0. By setting \rho = \epsilon/d(\log d) + r, we get arbitrarily small expected communication cost of C_{\alpha, \beta} = \epsilon, and the cost of exploding estimation error MSE_{\alpha, \beta} = O(1/\epsilon n).

Note that all of the above examples have random communication costs. What we present is the expected communication cost of the protocols. All the above examples can be modified to use the encoding protocol with fixed-size support defined in (4) with the parameter k set to the value of pd for corresponding \rho used above, to get the same results. The only practical difference is that the communication cost will be deterministic for each node, which can be useful for certain applications.

### 6. Optimal Parameters for Encoder \( \alpha(p_{ij}; \mu) \)

Here we consider (\alpha, \beta, \gamma), where \alpha = \alpha(p_{ij}, \mu_i) is the encoder defined in (1), \beta is the associated the sparse communication protocol, and \gamma is the averaging decoder. Recall from Lemma 2 and (8) that the mean square error and communication cost are given by:

\[
MSE_{\alpha, \gamma} = \frac{1}{n^2} \sum_{ij} \left( \frac{1}{p_{ij}} - 1 \right) (X_i(j) - \mu_i)^2,
\]

\[
C_{\alpha, \beta} = n\bar{r} + (\log d + r) \sum_{i=1}^{n} \sum_{j=1}^{d} p_{ij}.
\]

Having these closed-form formulae as functions of the parameters \{p_{ij}, \mu_i\}, we can now ask questions such as:

1. Given a communication budget, which encoding protocol has the smallest mean squared error?
2. Given a bound on the mean squared error, which encoder suffers the minimal communication cost?

Let us now address the first question; the second question can be handled in a similar fashion. In particular, consider the optimization problem

\[
\begin{aligned}
\text{minimize} & \quad \sum_{ij} \left( \frac{1}{p_{ij}} - 1 \right) (X_i(j) - \mu_i)^2 \\
\text{subject to} & \quad \mu_i \in \mathbb{R}, \quad i = 1, 2, \ldots, n \\
& \quad \sum_{ij} p_{ij} \leq B \\
& \quad 0 < p_{ij} \leq 1, \quad i = 1, 2, \ldots, n; \quad j = 1, 2, \ldots, d,
\end{aligned}
\]

where \( B > 0 \) represents a bound on the part of the total communication cost in (13) which depends on the choice of the probabilities \( p_{ij} \).

Note that while the constraints in (14) are convex (they are linear), the objective is not jointly convex in \{p_{ij}, \mu_i\}. However, the objective is convex in \{p_{ij}\} and convex in \{\mu_i\}. This suggests a simple alternating minimization heuristic for solving the above problem:

1. Fix the probabilities and optimize over the node centers.
2. Fix the node centers and optimize over probabilities.

These two steps are repeated until a suitable convergence criterion is reached. Note that the first step has a closed form solution. Indeed, the problem decomposes across the node centers to n univariate unconstrained convex quadratic minimization problems, and the solution is given by

\[
\mu_i = \frac{\sum_j w_{ij} X_i(j)}{\sum_j w_{ij}}, \quad w_{ij} = \frac{1}{p_{ij}} - 1.
\]

The second step does not have a closed form solution in general; we provide an analysis of this step in section 6.1.

**Remark 5.** Note that the upper bound \( \sum_{ij} (X_i(j) - \mu_i)^2 / p_{ij} \) on the objective is jointly convex in \{p_{ij}, \mu_i\}. We may therefore instead optimize this upper bound by a suitable convex optimization algorithm.

**Remark 6.** An alternative and a more practical model to (14) is to choose per-node budgets \( B_1, \ldots, B_n \) and require \( \sum_{j} p_{ij} \leq B_i \) for all i. The problem becomes separable across the nodes, and can therefore be solved by each node independently. If we set \( B = \sum_i B_i \), the optimal solution obtained this way will lead to MSE which is lower bounded by the MSE obtained through (14).

### 6.1. Optimal Probabilities for Fixed Node Centers

Let the node centers \( \mu_i \) be fixed. Problem (14) (or, equivalently, step 2 of the alternating minimization method described above) then takes the form

\[
\begin{aligned}
\text{minimize} & \quad \sum_{ij} \frac{(X_i(j) - \mu_i)^2}{p_{ij}} \\
\text{subject to} & \quad \sum_{ij} p_{ij} \leq B \\
& \quad 0 < p_{ij} \leq 1, \quad i = 1, 2, \ldots, n; \quad j = 1, 2, \ldots, d.
\end{aligned}
\]
Let \( S = \{(i,j) : X_i(j) \neq \mu_i\} \). Notice that as long as \( B \geq |S| \), the optimal solution is to set \( p_{ij} = 1 \) for all \((i,j) \in S\) and \( p_{ij} = 0 \) for all \((i,j) \notin S\). In such a case, we have \( \text{MSE}_{\alpha,\gamma} = 0 \). Hence, we can without loss of generality assume that \( B \leq |S| \).

While we are not able to derive a closed-form solution to this problem, we can formulate upper and lower bounds on the estimation error, given a bound on the communication cost formulated via \( B \).

**Theorem 6.1 (MSE-Optimal Protocols subject to a Communication Budget):** Consider problem (17) and fix any \( B \leq |S| \). Using the sparse communication protocol \( \beta \), the optimal encoding protocol \( \alpha \) has communication complexity
\[
C_{\alpha,B} = n\bar{r} + ((\log d) + r)B,
\]
and the mean squared error satisfies the bounds
\[
\left( \frac{1}{B} - 1 \right) \frac{R}{n} \leq \text{MSE}_{\alpha,\gamma} \leq \left( \frac{|S|}{B} - 1 \right) \frac{R}{n},
\]
where \( R = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{d} (X_i(j) - \mu_i)^2 = \frac{1}{n} \sum_{i=1}^{n} \|X_i - \mu_i\|^2 \). Let \( a_{ij} = |X_i(j) - \mu_i| \) and \( W = \sum_{i} a_{ij} \). If, moreover, \( B \leq \sum\sum a_{ij} / \max_{ij} a_{ij} \) (which is true, for instance, in the ultra-low communication regime with \( B \leq 1 \)), then
\[
\text{MSE}_{\alpha,\gamma} = \frac{W^2}{n^2 B} - \frac{R}{n}.
\]

**Proof:** Setting \( p_{ij} = B/|S| \) for all \((i,j) \in S\) leads to a feasible solution of (17). In view of (13), one then has
\[
\text{MSE}_{\alpha,\gamma} = \frac{1}{n^2} \left( \frac{|S|}{B} - 1 \right) \sum_{(i,j) \in S} (X_i(j) - \mu_i)^2 = \left( \frac{|S|}{B} - 1 \right) \frac{R}{n},
\]
where \( R = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{d} (X_i(j) - \mu_i)^2 = \frac{1}{n} \sum_{i=1}^{n} \|X_i - \mu_i\|^2 \).

If we relax the problem by removing the constraints \( p_{ij} \leq 1 \), the optimal solution satisfies \( a_{ij}/p_{ij} = \theta > 0 \) for all \((i,j) \in S\).

At optimality the bound involving \( B \) must be tight, which leads to \( \sum\sum a_{ij} / \theta = B \), whence \( \theta = \frac{1}{B} \sum\sum a_{ij} \). So, \( p_{ij} = a_{ij} B / \sum\sum a_{ij} \). The optimal MSE therefore satisfies the lower bound
\[
\text{MSE}_{\alpha,\gamma} \geq \frac{1}{n^2} \sum_{(i,j) \in S} \left( \frac{1}{p_{ij}} - 1 \right) (X_i(j) - \mu_i)^2 = \frac{1}{n^2} B W^2 - \frac{R}{n},
\]
where \( W = \sum\sum a_{ij} \). If \( B \leq \sum\sum a_{ij} / \max_{ij} a_{ij} \), then \( p_{ij} \leq 1 \) for all \((i,j) \in S\), and hence we have optimality. (Also note that, by Cauchy-Schwartz inequality, \( W^2 \leq nR|S| \).)

### 6.2. Trade-Off Curves

To illustrate the trade-offs between communication cost and estimation error (MSE) achievable by the protocols discussed in this section, we present simple numerical examples in Figure 1, on three synthetic data sets with \( n = 16 \) and \( d = 512 \). We choose an array of values for \( B \), directly bounding the communication cost via (18), and evaluate the MSE (2) for three encoding protocols (we use the sparse communication protocol and averaging decoder). All these protocols have the same communication cost, and only differ in the selection of the parameters \( p_{ij} \) and \( \mu_i \). In particular, we consider

(i) uniform probabilities \( p_{ij} = p > 0 \) with average node centers \( \mu_i = \frac{1}{d} \sum_{j=1}^{d} X_i(j) \) (blue dashed line),

(ii) optimal probabilities \( p_{ij} \) with average node centers \( \mu_i = \frac{1}{d} \sum_{j=1}^{d} X_i(j) \) (green dotted line), and

(iii) optimal probabilities with optimal node centers, obtained via the alternating minimization approach described above (red solid line).

In order to put a scale on the horizontal axis, we assumed that \( r = 16 \). Note that, in practice, one would choose \( r \) to be as small as possible without adversely affecting the application utilizing our distributed mean estimation method. The three plots represent \( X_i \) with entries drawn in an i.i.d. fashion from Gaussian (\( \mathcal{N}(0,1) \)), Laplace (\( \mathcal{L}(0,1) \)), and chi-squared (\( \chi^2(2) \)) distributions, respectively. As we can see, in the case of non-symmetric distributions, it is not necessarily optimal to set the node centers to averages.
As expected, for fixed node centers, optimizing over probabilities results in improved performance, across the entire trade-off curve. That is, the curve shifts downwards. In the first two plots based on data from symmetric distributions (Gaussian and Laplace), the average node centers are nearly optimal, which explains why the red solid and green dotted lines coalesce. This can be also established formally. In the third plot, based on the non-symmetric chi-squared data, optimizing over node centers leads to further improvement, which gets more pronounced with increased communication budget. It is possible to generate data where the difference between any pair of the three trade-off curves becomes arbitrarily large.

Finally, the black cross represents performance of the quantization protocol from Example 4. This approach appears as a single point in the trade-off space due to lack of any parameters to be fine-tuned.

7. FURTHER CONSIDERATIONS

In this section we outline further ideas worth consideration. However, we leave a detailed analysis to future work.

7.1. Beyond Binary Encoders

We can generalize the binary encoding protocol (1) to a $k$-ary protocol. To illustrate the concept without unnecessary notation overload, we present only the ternary (i.e., $k = 3$) case.

Let the collection of parameters $\{p_{ij}', p_{ij}'', \tilde{X}_i', \tilde{X}_i''\}$ define an encoding protocol $\alpha$ as follows:

$$Y_i(j) =\begin{cases} \tilde{X}_i' & \text{with probability } p_{ij}' \\ \tilde{X}_i'' & \text{with probability } p_{ij}'' \\ \frac{1}{1-p_{ij}'-p_{ij}''} (X_i(j) - p_{ij}' \tilde{X}_i' - p_{ij}'' \tilde{X}_i'') & \text{with probability } 1 - p_{ij}' - p_{ij}'' \end{cases}$$

(21)

It is straightforward to generalize Lemmas 3.1 and 3.2 to this case. We omit the proofs for brevity.

Lemma 7.1 (Unbiasedness): The encoder $\alpha$ defined in (21) is unbiased. That is, $E_{\alpha} [\alpha(X_i(j))] = X_i$ for all $i$. As a result, $Y$ is an unbiased estimate of the true average: $E_{\alpha} [Y] = X$.

Lemma 7.2 (Mean Squared Error): Let $\alpha = (p_{ij}', p_{ij}'', \tilde{X}_i', \tilde{X}_i'')$ be the protocol defined in (21). Then

$$MSE_{\alpha}(X_1, \ldots, X_n) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{d} \left( p_{ij}' (X_i(j) - \tilde{X}_i')^2 + p_{ij}'' (X_i(j) - \tilde{X}_i'')^2 + (p_{ij}' \tilde{X}_i' + p_{ij}'' \tilde{X}_i'')^2 \right).$$

We expect the $k$-ary protocol to lead to better (lower) MSE bounds, but at the expense of an increase in communication cost. Whether or not the trade-off offered by $k > 2$ is better than that for the $k = 2$ case investigated in this paper is an interesting question to consider.

7.2. Preprocessing via Random Transformations

Following the idea proposed in Suresh et al. [13], one can explore an encoding protocol $\alpha_Q$ which arises as the composition of a random mapping, $Q$, applied to $X_i$ for all $i$, followed by the protocol $\alpha$ described in section 3. Letting $Z_i = QX_i$ and $Z = \frac{1}{n} \sum_{i=1}^{n} Z_i$, we thus have

$$Y_i = \alpha(Z_i), \quad i = 1, 2, \ldots, n.$$

With this protocol we associate the decoder $\gamma(Y_1, \ldots, Y_n) = \frac{1}{n} \sum_{i=1}^{n} Q^{-1} Y_i$. Note that

$$MSE_{\alpha,Q} = E \left[ \|Y_1, \ldots, Y_n - X\|^2 \right] = E \left[ \|Q^{-1} Y_1, \ldots, Y_n - Q^{-1} Z\|^2 \right] = E \left[ \|\gamma(X_1, \ldots, X_n) - Z\|^2 \right] = E \left[ \|\gamma(X_1, \ldots, X_n) - Z\|^2 \mid Q \right].$$

This approach is motivated by the following observation: a random rotation can be identified by a single random seed, which is easy to communicate to the server without the need to communicate all floating point entries defining $Q$. So, a random rotation pre-processing step implies only a minor communication overhead. It is important to stress that the use of $Q$ and $Q^{-1}$ in particular, can incur a significant computational overhead. The randomized Hadamard transform used in Suresh et al. [13] requires $O(d \log d)$ to apply, but computation of an inverse matrix can be $O(n^3)$ in general. However, if the preprocessing step helps to dramatically reduce the MSE, we get an improvement. Note that the inner expectation above is the formula for MSE of our basic encoding-decoding protocol, given that the data is $Z_i = QX_i$ instead of $X_i$. The outer expectation is over $Q$. Hence, we would like to have a mapping $Q$ which tends to transform the data $X_i$ into new data $Z_i$ with better MSE, in expectation.

From now on, for simplicity assume the node centers are set to the average, i.e., $\bar{Z}_i = \frac{1}{n} \sum_{j=1}^{d} Z_i(j)$. For any vector $x \in \mathbb{R}^d$, define

$$\sigma(x) \overset{\text{def}}{=} \sum_{j=1}^{d} (x(j) - \bar{x})^2 = \|x - \bar{x}1\|^2,$$

where $\bar{x} = \frac{1}{n} \sum_{j=1}^{d} x(j)$ and $1$ is the vector of all ones. Further, for simplicity assume that $p_{ij} = p$ for all $i,j$. Then using Lemma 3.2, we get

$$MSE = \frac{1 - p}{pn^2} \sum_{j=1}^{n} E_Q \left[ \|Z_i - \bar{Z}_i1\|^2 \right] = \frac{1 - p}{pn^2} \sum_{i=1}^{n} E_Q \left[ \sigma(QX_i) \right].$$

It is interesting to investigate whether choosing $Q$ as a random mapping, rather than identity (which is the implicit choice done
In previous sections, leads to improvement in MSE, i.e., whether we can in some well-defined sense obtain an inequality of the type
\[
\sum_i E_Q [\sigma(QX_i)] \ll \sum_i \sigma(X_i).
\]

If \( Q \) was a tight frame satisfying the uncertainty principle, this could perhaps be realized by computing the Kashin representation of the vectors to be quantized \([18]\). However, as pointed out above, depending on the tight frame, this might come at a significant additional computational cost, and it is not obvious how much can the variance be reduced.

This is the case for the quantization protocol proposed in Suresh et al. \([13]\), which arises as a special case of our more general protocol. This is because the quantization protocol is suboptimal within our family of encoders. Indeed, as we have shown, with a different choice of the parameter we can obtain results which improve, in theory, on the rotation + quantization approach. This suggests that perhaps combining an appropriately chosen rotation pre-processing step with our optimal encoder, it may be possible to achieve further improvements in MSE for any fixed communication budget. Finding suitable random mappings \( Q \) requires a careful study which we leave to future research.

**AUTHOR CONTRIBUTIONS**

All authors listed have made a substantial, direct and intellectual contribution to the work, and approved it for publication.

**ACKNOWLEDGMENTS**

JK acknowledges support from Google via a Google European Doctoral Fellowship. Work done while at University of Edinburgh, currently at Google. PR acknowledges support from Amazon, and the EPSRC Grant EP/K02325X/1, Accelerated Coordinate Descent Methods for Big Data Optimization and EPSRC Fellowship EP/N00538/1, Randomized Algorithms for Extreme Convex Optimization.

**REFERENCES**


**Conflict of Interest Statement:** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2018 Konečný and Richtárik. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.
APPENDIX

A. ADDITIONAL PROOFS

In this section we provide proofs of Lemmas 3.3 and 3.4, describing properties of the encoding protocol α defined in (4). For completeness, we also repeat the statements.

Lemma A.1 (Unbiasedness): The encoder α defined in (1) is unbiased. That is, \( E_\alpha [\alpha(X)] = X_i \) for all \( i \). As a result, \( Y \) is an unbiased estimate of the true average: \( E_\alpha [Y] = X \).

Proof: Since \( Y(j) = \frac{1}{n} \sum_{i=1}^{n} Y_i(j) \) and \( X(j) = \frac{1}{n} \sum_{i=1}^{n} X_i(j) \), it suffices to show that \( E_\alpha [Y_i(j)] = X_i(j) \):

\[
E_\alpha [Y_i(j)] = \frac{1}{|\sigma_k(d)|} \sum_{\sigma \in \sigma_k(d)} 1_{j \in \sigma} \left( \frac{dX_i(j)}{k} - \frac{d-k}{k} \mu_i \right) + 1_{j \not\in \sigma} \mu_i
\]

\[
= \left( \frac{d}{k} \right)^{-1} \left( \frac{d}{k} - 1 \right) \left( \frac{dX_i(j)}{k} - \frac{d-k}{k} \mu_i \right) + \left( \frac{d}{k} - 1 \right) \mu_i
\]

\[
= \left( \frac{d}{k} \right)^{-1} \left( \frac{d}{k} - 1 \right) dX_i(j) + \left( \frac{d}{k} - 1 \right) \left( \frac{d-k}{k} \right) \mu_i
\]

\[
= X_i(j)
\]

and the claim is proved. \( \square \)

Lemma A.2 (Mean Squared Error): Let \( \alpha = \alpha(k) \) be encoder defined as in (4). Then

\[
MSE_\alpha(X_1, \ldots, X_n) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{d} \frac{d-k}{k} (X_i(j) - \mu_i)^2.
\]

Proof: Using Lemma 2.3, we have

\[
MSE_\alpha(X_1, \ldots, X_n) = \frac{1}{n^2} \sum_{i=1}^{n} E_\alpha [\|Y_i - X_i\|^2]
\]

\[
= \frac{1}{n^2} \sum_{i=1}^{n} E_\alpha \left[ \sum_{j=1}^{d} (Y_i(j) - X_i(j))^2 \right]
\]

\[
= \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{d} E_\alpha [(Y_i(j) - X_i(j))^2]. \tag{A1}
\]

Further,

\[
E_\alpha [(Y_i(j) - X_i(j))^2] = \left( \frac{d}{k} \right)^{-1} \sum_{\sigma \in \sigma_k(d)} 1_{j \in \sigma} \left( \frac{dX_i(j)}{k} - \frac{d-k}{k} \mu_i - X_i(j) \right)^2 + 1_{j \not\in \sigma} (\mu_i - X_i(j))^2
\]

\[
= \left( \frac{d}{k} \right)^{-1} \left( \frac{d}{k} - 1 \right) \left( \frac{d-k}{k^2} \right) (X_i(j) - \mu_i)^2 + \left( \frac{d}{k} - 1 \right) (\mu_i - X_i(j))^2
\]

\[
= \frac{d-k}{k} (X_i(j) - \mu_i)^2.
\]

It suffices to substitute the above into (A1). \( \square \)