Optimal Gradient Compression for Distributed and Federated Learning

Alyazeed Albasyoni   **Mher Safaryan**   **Laurent Condat**[*]   **Peter Richtárik**
King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia

Abstract

Communicating information, like gradient vectors, between computing nodes in distributed and federated learning is an unavoidable burden, resulting in scalability issues. Indeed, communication might be slow and costly. Recent advances in communication-efficient training algorithms have reduced this bottleneck by using compression techniques, in the form of sparsification, quantization, or low-rank approximation. Since compression is inexact, the iteration complexity is typically worsened; but the communication complexity can improve significantly, leading overall to large computation time savings. We investigate the trade-off between the number of bits needed to encode compressed vectors and the corresponding compression error. In a worst-case analysis, we introduce an efficient compression operator, *Sparse Dithering*, which is very close to a lower bound we have derived. In an average-case analysis, we design the new *Spherical Compression* scheme, which is simple and optimal. Our new compressors significantly outperform the state of the art, as illustrated by numerical experiments in Figures 1 and 2.

1 Introduction

Due to the necessity of huge amounts of data to achieve high-quality machine learning models [1, 2], modern large-scale training procedures are executed in a distributed environment [3, 4]. In such a setup, both storage and computation needs are reduced, as the overall data is partitioned among the nodes and computation is carried out in parallel. However, it is necessary for the nodes to exchange information about their local progress [5–9]. This need for communication is typically a burden, resulting in a scalability issue commonly referred to as *communication bottleneck* [10–12]. To reduce its amount, the information can be communicated in a compressed or inexact form. *Information lossy compression* is a common practice, where original information is encoded approximately with essentially fewer bits, while introducing additional controllable distortion into the decoded message.

In the context of *Federated Learning* [13–15], communication between devices arises naturally, as data is initially decentralized and should remain so, for privacy purposes. It might be even desirable for each unit, or client, to compress/encode/encrypt the information they are going to share, to minimize private data disclosures. Another practical scenario where compression methods are useful is when storage capabilities are scarce or there is no need to save complete versions of the data. In such cases, the representation of the data (encoding and decoding schemes) can be optimized and with little or no precision loss, one can allocate significantly less memory space.

Related work. Recently, substantial amount of work has been devoted to the advances of communication-efficient training algorithms by utilizing various types of compression mechanisms, such as sparsification [16–18], quantization [19–21] and low-rank approximation [4]. Typically, the information communicated by computing nodes consists of local gradients, to which compression operators are applied. For example, one popular example of such compression operator is Top-k [22],

[*]corresponding author. Contact: see `https://lcondat.github.io/`

which transfers only k coordinates of the gradient with largest magnitudes. The theoretical foundations of lossy compression have long history and are based on *Rate-Distortion Theory* rooted in Shannon's work [23,24]. Recently, this theory has been exploited in the context of model compression [25,26]. Typically, the statistical properties of the input messages are supposed known and are exploited. This differs from our setting: we do not assume any knowledge on the encoded vectors.

We investigate the problem of lossy compression, namely encoding vectors $x \in \mathbb{R}^d$, without prior knowledge on the distribution, for any $d \geq 1$, into as few bits as possible, while introducing as little distortion as possible. Formally, we measure the distortion of a (possibly randomized) compression operator $\mathcal{C}\colon \mathbb{R}^d \to \mathbb{R}^d$ by its constant $\alpha \in [0,1]$ such that $\mathbb{E}\left[\|\mathcal{C}(x) - x\|^2\right] \leq \alpha\|x\|^2$ for every x, where the norm is the Euclidean norm (see Definitions 1, 2 and 3 for details). We denote by b the number of bits (in the worst case or in expectation) needed to encode $\mathcal{C}(x)$. Intuitively, b and α cannot be too small at the same time: they are antagonistic and ruled by a fundamental rate-distortion trade-off. As a matter of fact, as shown in [27], the following lower bound, referred to as *uncertainty principle for communication compression*, holds (if omitted, the base of $\log$ is assumed to be 2):

$$\alpha\, 4^{b/d} \geq 1 \quad \text{or, equivalently,} \quad b \geq d\,\tfrac{1}{2}\log\tfrac{1}{\alpha}. \tag{1}$$

In this work, we investigate this trade-off more deeply. We perform two types of analyses: *worst case analysis (WCA)* and *average case analysis (ACA)*, and we we design new efficient compression schemes in both cases. Note that our derivations deal with real numbers, compressed using a finite number of bits. We should keep in mind that numbers are represented by finite-precision, say 32 bits, floats in computers. We can safely omit this aspect in the derivations, without loss of rigor.

WCA: Contributions. We will not detail this study by lack of space, but we have constructed a compression scheme with α distortion and b encoding bits (in the worst case), satisfying

$$\alpha\, 4^{b/d} \leq \text{poly}(d)^{1/d} \quad \text{or} \quad b \leq d\,\tfrac{1}{2}\log\tfrac{1}{\alpha} + \mathcal{O}(\log d). \tag{2}$$

This implies the asymptotic tightness of the bound (1) as the dimension d grows. Also, investigating the minimal number of bits (in the worst case) $b^*(\alpha, d)$ as a function of distortion α and dimension d, we proved that

$$b^*(\alpha, d) = -\log P(\alpha, d) + \log d + \tfrac{1}{2}\log\log d + e,$$

where $P(\alpha, d) := \frac{1}{2} I_\alpha(\frac{d-1}{2}, \frac{1}{2})$, with I_α the regularized incomplete beta function, and e a negligible additive error term with $|e| \leq \frac{1}{2}\log\log d + \mathcal{O}(1)$, as opposed to $\mathcal{O}(\log d)$ in (1) and (2).

Motivated by these lower bounds, we consider in Section 3 the construction of a compression method which is optimal and implementable in high dimensions. For this, we slightly depart from the optimal boundary and propose a new efficient compression method—*Sparse Dithering (SD)*. Both deterministic (biased) and randomized (unbiased) versions of SD are analyzed, and comparisons with existing methods are made, showing that we outperform the state of the art. In the special case, the encoding of deterministic SD with $\alpha = 1/10$ distortion requires at most $30 + \log d + 3.35d$ bits, which is optimal within $1.69d$ additional bits (see Theorem 1).

ACA: Contributions. In the average-case analysis, we have established a lower bound $-\log P(\alpha, d) \leq B$ on the expected number of bits B needed to encode a compression operator from $\mathbb{C}(\alpha)$ (see Definition 3). To reach this lower bound, in Section 4, we first analyze the randomized (and unbiased) version of SD. It is good but still suboptimal with respect to the lower bound. Then we present a simple compression operator–*Spherical Compression*–which attains the lower bound with less than 3 extra bits, (see Theorem 4).

Our findings are developed and illustrated by experiments in the long version of the paper [28].

2 Discussion about Compression Operators

First, we formally define three general classes of compression operators, that will be considered throughout the paper. We start with the well-studied class of unbiased compressors [16,20,29,30].

Definition 1 (ω-compressors)**.** *We denote by $\mathbb{U}(\omega)$ the class of unbiased compression operators $\mathcal{C}\colon \mathbb{R}^d \to \mathbb{R}^d$ with variance $\omega \geq 0$; that is, $\mathbb{E}\left[\mathcal{C}(x)\right] = x$ and $\mathbb{E}\left[\|\mathcal{C}(x) - x\|^2\right] \leq \omega\|x\|^2$, $\forall x \in \mathbb{R}^d$.*

Another broad class of compressions operators, for which compressed learning algorithms have been successfully analysed [31–34], is the class of biased operators, which are contractive in expectation.

Definition 2 (α-contractive operators)**.** *We denote by $\mathbb{B}(\alpha)$ the class of (possibly biased and randomized) compression operators $\mathcal{C}\colon \mathbb{R}^d \to \mathbb{R}^d$ with $\alpha \in [0,1]$-contractive property; that is, $\mathbb{E}\left[\|\mathcal{C}(x) - x\|^2\right] \le \alpha\|x\|^2$, $\forall x \in \mathbb{R}^d$.*

Similar to ω for the variance, the parameter α is referred to as normalized variance or distortion threshold. It has been shown that the class $\mathbb{U}(\omega)$ can be embedded into $\mathbb{B}(\alpha)$. Specifically, if $\mathcal{C} \in \mathbb{U}(\omega)$, then $\frac{1}{\omega+1}\mathcal{C} \in \mathbb{B}(\frac{\omega}{\omega+1})$ (see e.g. Lemma 1 in [27]). We will also consider the subclass of strictly contractive operators which, compared to operators from $\mathbb{B}(\alpha)$, are contractive for all realizations rather than in expectation:

Definition 3 (Strictly α-contractive operators)**.** *We denote by $\mathbb{C}(\alpha)$ the class of (possibly biased and randomized) compression operators $\mathcal{C}\colon \mathbb{R}^d \to \mathbb{R}^d$ with $\alpha \in [0,1]$-strictly contractive property; that is, $\|\mathcal{C}(x) - x\|^2 \le \alpha\|x\|^2$, $\forall x \in \mathbb{R}^d$.*

Encoding and Decoding Generally speaking, compression is a two-sided notion in the sense that one end encodes the message, while the other end decodes it to estimate the original information. An encoder is any mapping $E\colon \mathbb{R}^d \to \{0,1\}^*$ which maps a given vector $x \in \mathbb{R}^d$ to some finite word from the set of all finite words $\{0,1\}^*$ with the binary alphabet $\{0,1\}$. A decoder, on the other hand, is a mapping $D\colon \{0,1\}^* \to \mathbb{R}^d$ which aims to reconstruct the initial vector $x \in \mathbb{R}^d$ from the finite binary codeword $E(x)$. Thus, a compression operator $\mathcal{C}\colon \mathbb{R}^d \to \mathbb{R}^d$ can be decomposed into an encoder and decoder so that $\mathcal{C}(x) = D(E(x))$. The number of bits needed to transfer a compressed version of $x \in \mathbb{R}^d$ is the length $|E(x)|$ of the binary word $E(x)$. In the worst case analysis we are interested in the length of the longest codeword $\sup_{x,\mathcal{C}(x)} |E(x)|$, while in average case analysis we investigate the size of the longest expected codeword $\sup_x \mathbb{E}_{\mathcal{C}}\left[|E(x)|\right]$.

From now on, we exclude the trivial cases $\omega = 0,\ \alpha \in \{0,1\}$ and we assume $\omega > 0,\ \alpha \in (0,1)$.

Two notions of optimality. It is worth distinguishing between optimality within a class in a single step of communication and optimality of total communication throughout the optimization process leading to ϵ-accuracy, e.g. $\frac{\|x^t - x^\star\|^2}{\|x^0 - x^\star\|^2} \le \epsilon$ for a prescribed ϵ, where t is the iteration counter. Our theoretical contributions mainly deal with the first sense of optimality. Regarding the second view of optimality, the following proposition shows that Compressed Gradient Descent (CGD) can converge at significantly different speeds for different operators from $\mathbb{B}(\alpha)$.

Proposition 1. *If $\mathcal{C} \in \mathbb{B}(\alpha)$, the iteration complexity of CGD is $\frac{1}{1-\alpha}$ times bigger than for GD; that is CGD needs $\frac{1}{1-\alpha}$ times more iterations than GD to obtain the same ϵ-accuracy. Moreover, if $\mathcal{C}$ is additionally unbiased, then only $1+\alpha$ times more iterations are sufficient.*

Thus, if we aim to minimize the total communication complexity ensuring convergence to ϵ-accuracy, then the optimal operator $\mathcal{C}^*$ should be either unbiased, or it will need to satisfy not only the direct condition, $\mathbb{E}[|E_{\mathcal{C}^*}(x)|] \le \mathbb{E}[|E_{\mathcal{C}}(x)|]$ for all operators $\mathcal{C} \in \mathbb{B}(\alpha)$, but also the additional condition $\mathbb{E}[|E_{\mathcal{C}^*}(x)|] \le (1-\alpha^2)\mathbb{E}[|E_{\mathcal{U}}(x)|]$ for all unbiased operators $\mathcal{U} \in \mathbb{B}(\alpha)$. It is important to see that when α is close to 1, then this additional constraint is hard to satisfy when $\mathcal{C}^*$ is not unbiased. For $\alpha < 1$, we show that this is indeed the case by obtaining an optimal biased operator $\mathcal{C}^* \in \mathbb{B}(\alpha)$, which we call *Spherical Compression*, and another unbiased one, which we call *Sparse Dithering*. We show that the latter is more suitable in practice due to its unbiasedness, and hence, convergence occurs in much fewer iterations, and that this is most pronounced when α is close to 1. In addition to being computationally efficient, Sparse Dithering is guaranteed to reduce the total training communication by $\approx 9.9\times$ compared to full precision gradient communication of 32-bits floats.

Compressed learning algorithms. To highlight the importance of investigating the communication-variance trade-off of compression operators, we present how these operators affect the performance of compressed learning algorithms. For the sake of simplicity, consider distributed *Compressed Gradient Descent (CGD)* with compression operator $\mathcal{C} \in \mathbb{U}(\omega)$ solving the following smooth non-convex optimization problem

$$\min_{x\in\mathbb{R}^d} f(x) := \tfrac{1}{n}\textstyle\sum_{i=1}^n f_i(x),$$

where n is the number of nodes or machines available and $f_i(x)$ is the loss function corresponding to the data stored at node i. Hence, CGD algorithm iteratively performs the updates $x^{t+1} = x^t - \gamma_t g^t$ with unbiased gradient estimator

$$g^t = \tfrac{1}{n}\textstyle\sum_{i=1}^n g_i^t := \tfrac{1}{n}\sum_{i=1}^n \mathcal{C}(\nabla f_i(x^t)).$$

Using smoothness of the loss function $f(x)$, the expected loss is upper bounded as follows:

$$\begin{aligned}
\mathbb{E}[f(x^{t+1})|x^t] &= \mathbb{E}_t[f(x^t - \gamma_t g^t)] \\
&\overset{L\text{-smoothness}}{\leq} f(x^t) - \gamma_t \|\nabla f(x^t)\|^2 + \tfrac{L\gamma_t^2}{2}\mathbb{E}_t\left[\|g^t\|^2\right] \\
&= f(x^t) - \tfrac{2\gamma_t - L\gamma_t^2}{2}\|\nabla f(x^t)\|^2 + \tfrac{L\gamma_t^2}{2}\mathbb{E}_t\left[\|g^t - \nabla f(x^t)\|^2\right],
\end{aligned}$$

where $L > 0$ is the smoothness parameter. Now, the term that is affected by compression and slowing down the convergence is the last one, namely the variance of estimator g^t, which can be transform into

$$\begin{aligned}
\mathbb{E}_t\left[\|g^t - \nabla f(x^t)\|^2\right] &= \mathbb{E}_t\left[\|\tfrac{1}{n}\textstyle\sum_{i=1}^n \left(g_i^t - \nabla f_m(x^t)\right)\|^2\right] \\
&= \tfrac{1}{n^2}\textstyle\sum_{i=1}^n \mathbb{E}_t\left[\|g_i^t - \nabla f_i(x^t)\|^2\right] \leq \tfrac{\omega}{n^2}\textstyle\sum_{i=1}^n \|\nabla f_i(x^t)\|^2.
\end{aligned}$$

Clearly, without compression ($\omega = 0$), this term vanishes. Thus, the slowdown caused by the compression operator $\mathcal{C} \in \mathbb{U}(\omega)$ is controlled by its parameter ω. Similarly, for compression operators from $\mathbb{B}(\alpha)$ or $\mathbb{C}(\alpha)$, the slowdown is controlled by α. Hence, the goal is to design compression operators minimizing both variance and number of encoding bits to get efficient learning algorithms.

3 Sparse Dithering: An Almost Optimal Scheme in the Worst Case

We introduce a new compression scheme–*Sparse Dithering (SD)*–which is efficient in high dimension and nearly optimal. In some sense, SD can viewed as an effective combination of Top-k sparsification [22] and random dithering with uniform levels [29]. The essential novelty is the encoding scheme and better upper bound on the number of communicated bits. In this section, we present a deterministic and hence biased version of SD.

Construction and variance bound. To compress a given nonzero vector $x \in \mathbb{R}^d$, we first compress the normalized vector $u = x/\|x\| \in \mathbb{S}^d$ and then rescale it. To quantize the coordinates of the unit vector u, we apply dithering with levels $2k_i h,\ k_i \geq 0$, where $h = \sqrt{\nu/d}$ is the half-step and $\nu > 0$ is a free parameter. For each coordinate $u_i, i \in [d]$ we choose the nearest level so that $||u_i| - 2k_i h| \leq h$. Letting $\hat{u}_i = \operatorname{sign}(u_i)\, 2k_i h$ we have $|u_i - \hat{u}_i| \leq h$ for all $i \in [d]$. Therefore,

$$\|u - \hat{u}\|^2 = \textstyle\sum_{i=1}^d (u_i - \hat{u}_i)^2 \leq dh^2 = \nu.$$

Note that, after applying the scaling factor $\|x\|$, this gives a compression with variance at most ν. However, $\|x\|$ is not always the best option. Specifically, we can choose the scaling factor $\gamma > 0$ so to minimize the variance $\|x - \gamma\hat{u}\|^2$, which yields the optimal factor $\gamma^* = \frac{\langle x, \hat{u}\rangle}{\|\hat{u}\|^2}$ with the optimal variance of $\|x - \gamma^*\hat{u}\|^2 = \sin^2\varphi\,\|x\|^2$, where $\varphi \in [0, \pi/2]$ is the angle between x and $\hat{u}$ (in case of $\mathcal{C}(x) = 0$ we set $\varphi = \pi/2$). Hence, defining the compression operator as $\mathcal{C}(x) = \gamma^*\hat{u}$, we have the following bound on the variance:

$$\|\mathcal{C}(x) - x\|^2 \leq \min\left(\nu, \sin^2\varphi\right)\|x\|^2.$$

Encoding scheme. We now describe the corresponding encoding scheme into a sequence of bits. With the following notations:

$$\gamma := 2h\gamma^* \in \mathbb{R}_+, \quad k := (k_i)_{i=1}^d \in \mathbb{N}_+^d, \quad s := (\operatorname{sign}(u_i k_i))_{i=1}^d \in \{-1, 0, 1\}^d, \tag{3}$$

the compression operator can be written as $\mathcal{C}(x) = \gamma^*\hat{u} = 2h\gamma^*\ \operatorname{sign}(u)\, k = \gamma\, s\, k$. So, we need to encode the triple (γ, s, k). As $\gamma \in \mathbb{R}_+$, we need only $\underline{31}$ bits for the scaling factor. Next we encode s. Let

$$n_0 := |\{i \in [d] \colon s_i = 0\}| = |\{i \in [d] \colon k_i = 0\}|$$

be the number of coordinates u_i that are compressed to 0. To communicate s, we first send the locations of those n_0 coordinates and then $\underline{d - n_0}$ bits for the values ± 1. Sending n_0 positions can be done by sending $\underline{\log d}$ bits representing the number n_0, afterwards sending $\underline{\log\binom{d}{n_0}}$ bits for the positions. Finally, it remains to encode k, for which we only need to send nonzero entries, since the positions of $k_i = 0$ are already encoded. We encode $k_i \geq 1$ with k_i bits: $k_i - 1$ ones followed by a zero. Hence, encoding k requires $\underline{\sum k_i}$ bits.

We have derived a theoretical upper bound on the total number of bits for any choice of parameter $\nu > 0$. Let us just highlight the special case of $\nu = 1/10$:

Theorem 1. *Deterministic SD compression operator with parameter $\nu = 1/10$ belongs to $\mathbb{C}(1/10)$, communicating $30 + \log d + 3.35d$ bits at most.*

Ignoring the $30 + \log d$ negligible bits, SD is within a factor of $\log_4(\alpha\, 4^{b/d}) = \log_4\left(\frac{1}{10}4^{3.35}\right) \approx 1.69$ of optimality; that is, at most $1.69d$ more bits are sent in comparison to optimal compression with the same normalized variance $1/10$.

4 Average-Case Analysis

Now we switch to the average-case analysis for the class $\mathbb{C}(\alpha)$. Let us first investigate the trade-off between the normalized variance α and the expected number of bits $B = \sup_{\|x\|=1} \mathbb{E}_{\mathcal{C}}\left[|E(x)|\right]$. In other words, we study the trade-off for strictly α-contractive operators that encode any unit vector with no more than B bits in expectation. We have:

Theorem 2. *Let $\mathcal{C} \in \mathbb{C}(\alpha)$ be a compression operator such that $\mathcal{C}(x) \in \mathbb{R}^d$ can be transferred with B bits in expectation for any unit vector $x \in \mathbb{S}^d$. Then $-\log P(\alpha, d) \le B$.*

4.1 Randomized version of Sparse Dithering

Here we randomize Sparse Dithering to make it unbiased and estimate the number of encoding bits it needs in expectation. First we decompose the to-be-compressed vector $x \in \mathbb{R}^d$ into the magnitude and unit direction $u = x/\|x\|$ as before. To randomize the scheme, each coordinate u_i gets rounded to one of the two nearest neighbors, so as to preserve unbiasedness; that is, if $2k_ih \le |u_i| \le 2(k_i+1)h$ for some $k_i \ge 0$, then $\hat{u}_i = \operatorname{sign}(u_i)2\hat{k}_ih$ where

$$\hat{k}_i = \left\{k_i \text{ with prob. } \tfrac{2(k_i+1)h-|u_i|}{2h},\ k_i+1 \text{ with prob. } \tfrac{|u_i|-2k_ih}{2h}\right\}.$$

Clearly, $\mathbb{E}\left[\hat{u}\right] = u$ and defining $\mathcal{C}(x) = \|x\|\hat{u}$, we maintain unbiasedness $\mathbb{E}\left[\mathcal{C}(x)\right] = x$. The encoding scheme is the same as in the deterministic case. Upper bounding the expected number of bits and the variance, we obtain:

Theorem 3. *Randomized SD compression with parameter $\nu = \omega$ belongs to $\mathbb{U}(\omega)$, communicating at most $30 + \log d + \left(\log 3 + 1/(2\sqrt{\omega})\right)d$ bits in expectation. In particular, with $\omega = 1/4$ variance (ignoring $30 + \log d$ negligible factors), it uses $(1 + \log 3)\, d \approx 2.6d$ bits in each iteration (about 12 times less than full precision case) and forces up to $1 + \omega = 5/4$ times more iterations, leading to ≈ 9.9 times bandwidth savings.*

As mentioned earlier, SD is similar to random dithering with uniform levels, namely with $\sqrt{d}$ levels. However, with a different parametrization ν and better encoding strategy, SD provides better theoretical guarantees. Indeed, random dithering with $\sqrt{d}$ levels communicates $\approx 2.8d$ bits in expectation and requires $1 + \omega = 2$ times more iterations, resulting in a factor of ≈ 5.7 in bandwidth saving (see Theorem 3.2 and Corollary 3.3 of [29]).

Thus, the randomized version of Sparse Dithering is better than random dithering, but is suboptimal w.r.t. the established lower bound. In the next section, we present a new compression operator from $\mathbb{C}(\alpha)$, *Spherical Compression*, which is provably optimal.

4.2 New Compressor: Spherical Compression (SC)

It can be shown that randomized SD compression discussed in the previous section is suboptimal with respect to the lower bound of Theorem 2. Here we provide a simple compression operator–*Spherical Compression (SC)*–that achieves this lower bound with less than 3 overhead bits.

Construction and variance bound. As before, we transmit the magnitude and direction separately. For a given unit vector $x \in \mathbb{S}^d$, SC generates a sequence $(x^t)_{t=1}^T$ of i.i.d. points with $\|x^t\|^2 = 1 - \alpha$, and terminates once $\|x^T - x\|^2 \le \alpha$ for some $T \ge 1$. The last generated point x^T is the compressed version of x we need to communicate, that is $\mathcal{C}(x) = x^T$. It follows directly from this construction that $\mathcal{C} \in \mathbb{C}(\alpha)$.

Encoding scheme. The crucial observation is that it is sufficient to communicate the value of T. Importantly, the emitter and receiver have agreed on using the same random seed for generating i.i.d.

points (x^t), before the compression of any vector is performed. Thus, upon receiving the number of trials T, the decoder can reproduce the same sequence $x^1, x^2, \ldots, x^T$ and recover x^T. Consequently, it remains to encode the integer T into a binary code.

Upper bound on $\boldsymbol{B}$. T follows a geometric distribution with parameter $p = P(\alpha, d)$. In our case, the trials correspond to generating i.i.d. points x^t, until $x^t \in C^d(x^t, \sqrt{\alpha})$, which happens with proba. $P(\alpha, d) \in (0, 1/2)$. Therefore, the expected number of points x^t we need to generate until we get into α-vicinity of the initial point x is $\mathbb{E}[T] = 1/p = 1/P(\alpha,d) > 2$. Next, we encode T with the Golomb–Rice coding scheme. Define the integer $m \geq 0$ from $1/2p \leq 2^m < 1/p$ and decompose T as $T = 2^m q + r$ with $q \geq 0,\ 0 \leq r < 2^m$. The quotient q is encoded with unary coding as a string of q zeros followed by a 1. The remainder r is communicated with exactly m bits using truncated binary coding. There is no need to send the value of m as it can be computed from p, which depends only on α and d. Hence, the total number of bits to encode T is no more than $q + m + 1$. Note that $m < \log 1/p = -\log p$ is fixed, while q depends on T and $q \leq 2^{-m} T$. Hence,

$$\begin{aligned} B &= \mathbb{E}[q + m + 1] < 2^{-m}\mathbb{E}[T] - \log p + 1 \\ &= \tfrac{1}{2^m p} - \log p + 1 \leq -\log p + 3, \end{aligned} \tag{4}$$

which implies:

Theorem 4. *In the average-case analysis, Spherical Compression is optimal up to* 3 *extra bits; that is, it communicates $B < -\log P(\alpha, d) + 3$ bits in expectation.*

It is worth mentioning that the above compression operator satisfies $\|\mathcal{C}(x) - x\|^2 \leq \alpha\|x\|^2$ in the worst case, not in expectation. Moreover, because of the symmetry of spheres and caps $C^d(x^t, \sqrt{\alpha})$, it can be seen from the construction that $\mathbb{E}[\mathcal{C}(x)]$ points to the same direction as the initial vector x. Thus, with an appropriate (fixed) scaling factor, it can be made unbiased as well.

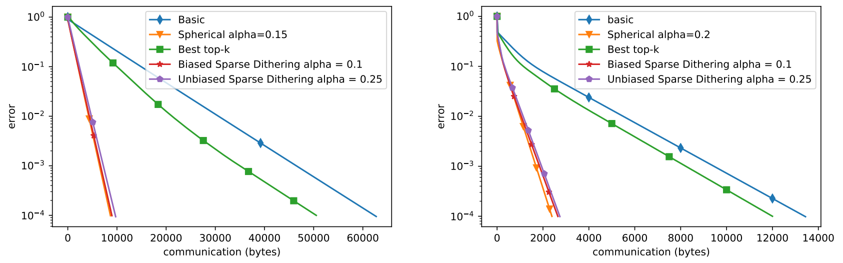

Figure 1: Communication vs. convergence: we look at convergence, measured as $\|x^t - x^\star\|^2 / \|x^0 - x^\star\|^2$ w.r.t. the number of bits communicated, for various compression operators. The benchmark 'Basic' sends a 32-bits float for every element in the gradient, with a total of $32d$ bits per iteration. We run Top-k for all $1 \leq k \leq d$ and pick the best representative, naming it 'Best Top$-k$'. Left: ridge regression on the Body-Fat dataset. Right: logistic regression on the Breast-Cancer dataset.

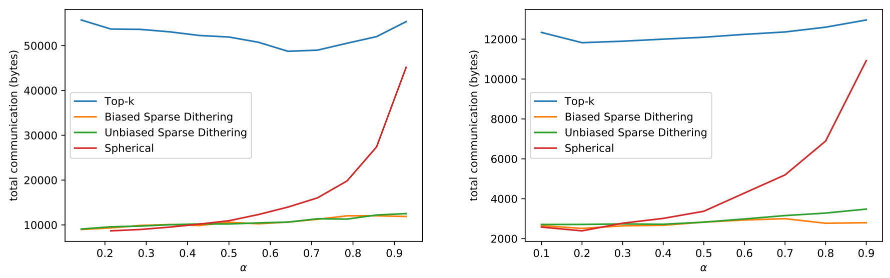

Figure 2: Total Communication as a Function of α: we let $\alpha = 1 - k/d$ vary and we show the total number of bits communicated before converging to ϵ-accuracy, with $\epsilon = 10^{-4}$. Note that for $1 \leq k \leq d$, we plot Top-k at $\alpha = 1 - k/d$. Left: ridge regression applied to the Body-Fat dataset. Right: logistic regression applied to the Breast-Cancer dataset.

References

[1] J. Schmidhuber, “Deep learning in neural networks: An overview,” in *Neural networks*, vol. 61, 2015, p. 85–117.

[2] S. Vaswani, F. Bach, and M. Schmidt, “Fast and faster convergence of SGD for over-parameterized models and an accelerated perceptron,” in *Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, 2019.

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

[4] T. Vogels, S. P. Karimireddy, and M. Jaggi, “PowerSGD: Practical low-rank gradient compression for distributed optimization,” in *Neural Information Processing Systems Conf. (NeurIPS)*, 2019.

[5] S. U. Stich, “Local SGD converges fast and communicates little,” in *Int. Conf. Learning Representations (ICLR)*, 2019.

[6] A. Khaled, K. Mishchenko, and P. Richtárik, “Tighter theory for local SGD on identical and heterogeneous data,” in *Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, 2020.

[7] D. Basu, D. Data, C. Karakus, and S. N. Diggavi, “Qsparse-local-SGD: Distributed SGD with quantization, sparsification, and local computations,” in *Neural Information Processing Systems Conf. (NeurIPS)*, 2019.

[8] G. Malinovsky, D. Kovalev, E. Gasanov, L. Condat, and P. Richtárik, “From local SGD to local fixed point methods for federated learning,” in *Proc. of 37th Int. Conf. Machine Learning (ICML)*, 2020.

[9] L. Condat, G. Malinovsky, and P. Richtárik, “Distributed proximal splitting algorithms with rates and acceleration,” 2020, preprint arXiv:2010.00952.

[10] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, “1-bit stochastic gradient descent and application to data-parallel distributed training of speech DNNs,” in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[11] H. Zhang, J. Li, K. Kara, D. Alistarh, J. Liu, and C. Zhang, “ZipML: Training linear models with end-to-end low precision, and a little bit of deep learning,” in *Int. Conf. Machine Learning (ICML)*, vol. 70, 2017, p. 4035–4043.

[12] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, “Deep gradient compression: Reducing the communication bandwidth for distributed training,” in *Int. Conf. Learning Representations (ICLR)*, 2018.

[13] J. Konečný, H. B. McMahan, F. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, “Federated learning: strategies for improving communication efficiency,” in *NIPS Private Multi-Party Machine Learning Workshop*, 2016.

[14] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, 2017.

[15] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, “SCAFFOLD: Stochastic controlled averaging for on-device federated learning,” 2019, preprint arXiv:1910.06378.

[16] J. Wangni, J. Wang, J. Liu, and T. Zhang, “Gradient sparsification for communication-efficient distributed optimization,” in *Neural Information Processing Systems Conf. (NeurIPS)*, 2018.

[17] H. Wang, S. Sievert, S. Liu, Z. Charles, D. Papailiopoulos, and S. Wright, “Atomo: Communication-efficient learning via atomic sparsification,” in *Neural Information Processing Systems Conf. (NeurIPS)*, 2018.

[18] N. Dryden, T. Moon, S. A. Jacobs, and B. V. Essen, “Communication quantization for data-parallel training of deep neural networks,” in *2nd Workshop on Machine Learning in HPC Environments (MLHPC)*, Nov 2016, pp. 1–8.

[19] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, “QSGD: Communication-efficient SGD via gradient quantization and encoding,” in *Neural Information Processing Systems Conf. (NeurIPS)*, 2017.

[20] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li, “Terngrad: Ternary gradients to reduce communication in distributed deep learning,” in *Neural Information Processing Systems Conf. (NeurIPS)*, 2017, p. 1509–1519.

[21] S. Horváth, C.-Y. Ho, L. Horváth, A. N. Sahu, M. Canini, and P. Richtárik, “Natural compression for distributed deep learning,” 2019, preprint arXiv:1905.10988.

[22] D. Alistarh, T. Hoefler, M. Johansson, S. Khirirat, N. Konstantinov, and C. Renggli, “The convergence of sparsified gradient methods,” in *Neural Information Processing Systems Conf. (NeurIPS)*, 2018.

[23] C. Shannon, “Coding theorems for a discrete source with a fidelity criterion.” *IRE Nat. Conv. Rec.*, vol. 27, pp. 379–423,623–656, 1948.

[24] ——, “A mathematical theory of communication,” *Bell Syst. Tech. J.*, vol. 4, pp. 142–163, 1959.

[25] W. Gao, Y.-H. Liu, C. Wang, and S. Oh, “Rate distortion for model Compression:From theory to practice,” in *Int. Conf. Machine Learning (ICML)*, vol. PMLR 97, 2019, pp. 2102–2111.

[26] Y. Bu, W. Gao, S. Zou, and V. V. Veeravalli, “Information-theoretic understanding of population risk improvement with model compression,” in *AAAI Conference on Artificial Intelligence*, February 2020, pp. 3300–3307.

[27] M. Safaryan, E. Shulgin, and P. Richtárik, “Uncertainty principle for communication compression in distributed and federated learning and the search for an optimal compressor,” 2020, preprint arXiv:2002.08958.

[28] A. Albasyoni, M. Safaryan, L. Condat, and P. Richtárik, “Optimal gradient compression for distributed and federated learning,” 2020, preprint arXiv:2010.03246.

[29] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, “QSGD: Communication-efficient sgd via gradient quantization and encoding,” in *Neural Information Processing Systems Conf. (NeurIPS)*, 2017.

[30] S. Khirirat, H. R. Feyzmahdavian, and M. Johansson, “Distributed learning with compressed gradients,” 2018, preprint arXiv:1806.06573.

[31] S. P. Karimireddy, Q. Rebjock, S. U. Stich, and M. Jaggi, “Error feedback fixes SignSGD and other gradient compression schemes,” 2019, preprint arXiv:1901.09847.

[32] S. U. Stich and S. P. Karimireddy, “The error-feedback framework: Better rates for SGD with delayed gradients and compressed communication,” 2019, preprint arXiv:1909.05350.

[33] S. Zheng, Z. Huang, and J. T. Kwok, “Communication-efficient distributed blockwise momentum SGD with error-feedback,” in *Neural Information Processing Systems Conf. (NeurIPS)*, 2019.

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