# Problem-dependent convergence bounds for randomized linear gradient compression

Thomas Flynn, Patrick Johnstone, and Shinjae Yoo



**Abstract**

In distributed optimization, the communication of model updates can be a performance bottleneck. Consequently, gradient compression has been proposed as a means of increasing optimization throughput. In general, due to information loss, compression introduces a penalty on the number of iterations needed to reach a solution. In this work, we investigate how the iteration penalty depends on the interaction between compression and problem structure, in the context of non-convex stochastic optimization. We focus on linear compression schemes, where compression and decompression can be modeled as multiplication with a random matrix. We consider several distributions of matrices, among them random orthogonal matrices and matrices with random Gaussian entries. We find that in each case, the impact of compression on convergence can be quantified in terms of the norm of the Hessian of the objective, using a norm defined by the compression scheme. The analysis reveals that in certain cases, compression performance is related to low-rank structure or other spectral properties of the problem. In these cases, our bounds predict that the penalty introduced by compression is significantly reduced compared to worst-case bounds that only consider the compression level, ignoring problem data. We verify the theoretical findings on several optimization problems, including fine-tuning an image classification model.

## I. Introduction

Training machine learning models on large datasets often involves distributed optimization. In this context distributed training means using multiple compute nodes to train a machine learning model collaboratively. The purpose could be to increase the overall throughput of training, and thus reach a solution in a shorter amount of time compared to single-node training, or it could be to increase accuracy, by indirectly combining datasets across nodes.

A significant performance bottleneck in distributed training comes from communication among training processes. In algorithms like Parallel SGD, model updates need to be synchronized at every optimization step, and this is typically carried out using a collective communication primitive like *all-reduce* [1], [2], [3]. In many HPC settings, there is an imbalance between local compute power and network bandwidth, such that time spent in network communication accounts for the majority of time in each iteration. One approach to address this bottleneck is to reduce communication by compressing model updates. In these schemes, model updates are compressed before communication, and these compressed representations are synchronized through collective communication and then decompressed. When the savings in communication time is greater than the overhead from compression and decompression, the result is faster training iterations. That is, we achieve higher throughput. However, higher throughput alone is not sufficient to achieve a benefit in machine learning problems. Practical compression schemes are lossy, meaning they introduce approximation errors into the training algorithm. This reduces the effectiveness of individual model updates. Hence in order for compression to be useful, the increase in throughput needs to be of sufficient magnitude to make up for the increased number of training steps that may be required to compensate for these approximation errors.

In this work we examine the performance of several compression schemes. We analyze how compression impacts the convergence of the algorithms, in terms of the number of iterations required to reach a solution. We focus our attention on three schemes in particular. In the first scheme, referred to as `rand-`$k$, at each step of optimization a random subset of $k$ elements of the model update is synchronized. The second scheme, `haar-`$k$, involves compressing gradients by projection to a random $k$-dimensional subspace, using random matrices distributed according to the Haar distribution on Orthogonal matrices. The third scheme we consider is `norm-`$k$, and involves compression using matrices with i.i.d. entries from the normal distribution $\mathcal{N}(0,1)$. Our optimization objective is to find a local minimum of a function $f : \mathbb{R}^m \to \mathbb{R}$:

$$\min_{x \in \mathbb{R}^m} f(x)$$

We allow $f$ to be non-convex, and our analyses bounds the number of iterations needed to find approximate stationary points. These are values of the parameter $x$ where $\|\nabla f(x)\|^2$ is small. The pseudo-code for the compression approach we study is listed in Algorithm 1. We assume that optimization is carried out by $r$ tasks. Iteration $t$ begins with a stochastic gradient computation at each task, producing the gradient estimates $g_t^i$. These gradients are then compressed locally, using a linear operator $Q_t$ which is assumed to be the same across tasks. These compressed gradients are synchronized via all-reduce, and then decompressed into the vector $\Delta_t$. This $\Delta$ is then combined with a step size $\epsilon_t$ to update the parameter, obtaining $x_{t+1}$. Note that depending on the type of compression used, it is not necessary to actually form the compression matrix $Q_t$ (for instance, when using `rand-`$k$).

T. Flynn (tflynn@bnl.gov) and S. Yoo are (sjyoo@bnl.gov) with the Computing & Data Sciences directorate, Brookhaven National Laboratory, Upton, NY, 11973 USA.

P. Johnstone (patrick.r.johnstone@gmail.com) is with Meta, New York, NY, 10001 USA. Work performed while at Brookhaven National Laboratory.

**Algorithm 1** SGD with compression (runs on $r$ nodes in parallel)

**input:** Initial point $x_1 \in \mathbb{R}^m$, same for all tasks.
**for** $t = 1, 2, \ldots$ **do**
  Compute stochastic gradient $g_t^i$
  Compress: $\Delta_t^i \leftarrow Q_t^T g_t^i$
  All-reduce: $\Delta_t \leftarrow \frac{1}{r} \sum_{i=1}^{r} \Delta_t^i$
  Decompress $h_t \leftarrow Q_t \Delta_t$
  Perform update: $x_{t+1} = x_t - \epsilon_t h_t$

The forms of compression we consider in this work can all be described with linear transformations. That is, in each case the operations of compression and decompression can be expressed as linear maps applied to the original vector $g_t$ or it's compressed representation $\Delta$, respectively. No matter which compression method is used, the next iterate $x_{t+1}$ takes the form

$$x_{t+1} = x_t - \epsilon Q_t Q_t^T g_t \tag{1}$$

where $g_t = \frac{1}{r} \sum_{i=1}^{r} g_t^i$.

### A. Related work

*1) Compression for distributed optimization:* To address the communication bottleneck, a wide variety of compression schemes have been considered. These include quantization [4], such as single-bit quantization [5], ternary quantization [6], random rounding [7]. Other classes include sparsification [8], count sketches [9]. Low-rank projections were proposed in [10]. Follow up work such as [11] demonstrates it's utility in large scale training.

Various theoretical analyses have focused on the convergence of algorithms that exploit compressed communication, including analyses focusing on quantization [4], generic sparsity preserving compressors [12], error-feedback and biased compressors [13] including the non-convex setting [14]. A systematic approach to bias compression approaches appears in [15].

A variety of characterizations of compression schemes appear in theoretical analysis. For instance, [16] defines an "$\omega$-unbiased compressor". Adapted to our setting of linear compressors (in the original definition of [16], the compressor need not be a linear mapping), an $\omega$-unbiased compressor is any distribution on $Q$ such that the following two properties hold, for any $g \in \mathbb{R}^m$:

$$\mathbb{E}[QQ^T] = I \tag{2a}$$

$$\mathbb{E}\left[\left\|g - QQ^T g\right\|^2\right] \leq \omega \|g\|^2 \tag{2b}$$

Equation 2a guarantees that the overall update in (1) is unbiased, while Equation (2b) bounds the variance. We note that the compressors studied in the present work, `norm-`$k$, `rand-`$k$, and `haar-`$k$ are all $\omega$-unbiased, with $\omega = \frac{m}{k} - 1$ for `rand-`$k$ and `haar-`$k$ and $\omega = \frac{m+1}{k}$ for `norm-`$k$. The compressor `norm-`$k$ and `rand-`$k$ also satisfy, after appropriate scaling, the criteria of $k$-contractions, as defined in [13]. While these abstractions facilitate analysis, they leave open the question of whether the performance bounds can be improved by taking into account the interaction between the compressor and the objective function. As can be seen below in Example III.6, the bounds that we derive can lead to tighter performance estimates compared to using (2b) alone.

Using the concept of matrix smoothness to define and analyze novel compressors was introduced in [17] and further studied in [18], [19]. Given a positive semidefinite matrix $\mathbb{L}$, let $\|\cdot\|_{\mathbb{L}}$ denote the seminorm $\|x\|_{\mathbb{L}} = \sqrt{x^T \mathbb{L} x}$. A function $f : \mathbb{R}^n \to \mathbb{R}$ is $\mathbb{L}$-smooth if $\forall x, y$,

$$f(x) \leq f(y) + \nabla f(y)^T (x - y) + \frac{1}{2} \|x - y\|_{\mathbb{L}}^2. \tag{3}$$

We adopt this formulation in our analysis. Intuitively, this matrix $\mathbb{L}$ can be taken to be a positive semidefinite matrix that majorizes the Hessian matrix (2nd derivative) of the objective function. An important difference in the present work is that we are not dealing with matrix step-sizes; rather, the matrix smoothness concept is used to analyzing the interaction between the compressor and the problem structure in the scalar step-size setting. Furthermore, in this work we specifically investigate the performance of the compressors `haar-`$k$ and `norm-`$k$.

Lower-bounds on the performance (overall communication, or number of iterations) of compressed training schemes have been considered in [20], which considered a detailed analysis of the trade off between compression accuracy and compression level. They consider $\alpha$-contractive compression operators, deriving relations between the $\alpha$, dimension $d$, and bits of the compressed representation $b$. [21] gives lower bounds on the overall amount of communication (in terms of number of bits)

needed for distributed optimization, independent of the algorithm. [22] gives lower bounds in terms of iteration numbers for compression. Considers "$\omega$-unbiased" and "contractive" compression.

Notable work focusing on constrained optimization proved that, the geometry of thee constraint set impacts the overall iteration penalty of compression [23].

A closely related algorithm is PowerSGD [10]. PowerSGD is also based on compression using linear projections. In PowerSGD, the initial values of $Q$ are random matrices, but later the compression matrices adapt to the observed stochastic gradients, using a variant of subspace iteration. By comparison, in this work we consider the sequence of projection matrices to be i.i.d.

To address the loss in accuracy of the gradients associated with compression, the technique of error feedback involves saving the error vector that captures the difference between the local gradient and it's compressed counterpart, and using this extra information to inform the model update at successive iterations. [5][24]. The convergence properties of error-feedback schemes were considered in [13] and in [14] for the case of non-convex functions. See also [16] for a unified analysis that can handle variance reduction and error-feedback. Specialized compressors have been designed for error feedback, including cyclic local top-k [25]. Error-feedback was also important in PowerSGD [10].

We also note that compression for distributed learning has been specialized and extended to a variety of settings, including differential privacy using quantization [26], handling heterogeneous data across the nodes [27], compression in federated learning (FL) [28], [29], and decentralized training [30], [31]. Variations of adaptive gradient methods like 1-bit Adam [32]. Additionally, there are adaptive schemes which adjust the compression level based on gradient statistics [33], and specific strategies for doing compression in cloud scenarios [34].

*2) Random orthogonal matrices:* In the compression scheme `haar-`$k$, the matrix $Q_t$ is constructed using the Haar measure on orthogonal matrices. The Haar measure can be viewed as the natural generalization of the uniform distribution on the unit sphere to matrices. A defining property of this distribution is that if $Q$ is Haar-distributed and $U$ is any orthogonal matrix, then $UQ$ and $QU$ are Haar-distributed as well [35]. That is, for any measurable function $h : \mathbb{R}^{n\times n} \to \mathbb{R}$, we have

$$\mathbb{E}[h(UQ)] = \mathbb{E}[h(QU)] = \mathbb{E}[h(Q)].$$

In the compression method `haar-`$k$, each $Q_t$ is made up of the first $k$ columns of an $n \times n$ Haar-distributed matrix (after scaling). An intuitive algorithm for sampling from the Haar distribution is to fill a matrix with i.i.d. samples from the standard normal distribution and then apply the Gram-Schmidt orthogonalization procedure; we use a more efficient algorithm based on the QR decomposition, following [36].

One result of our work is that using random orthogonal matrices leads to slightly better convergence bounds compared to Gaussian matrices. This is in line with the signal reconstruction results in [37]. That work analyzes a signal reconstruction when the observation is the product of a random matrix and the signal, considering both Gaussian and random orthogonal matrices for the observation matrix. Analysis shows that optimal reconstruction error is better under orthogonal matrices.

### *B. Our contributions*

- We derive optimization bounds for `haar-`$k$, `norm-`$k$, and `rand-`$k$ that reflect how the compressor interacts with problem structure, leading to bounds that can be better than those that solely account for compression without considering problem data.
- The analytical formulas for the $\mathcal{Q}$-norm for different schemes (`rand-`$k$, `haar-`$k$, `norm-`$k$) can potentially give insight into how the compressors compare against each other.
- Experiments on a linear regression task and fine-tuning the last layer of a neural network model suggest that the theory captures some of the real differences in performance among compressors.

## II. Properties of random projection matrices

We first define the three compression approaches we consider.

**Definition II.1.** For $m \geq 1$ and $1 \leq k \leq m$ we define three distributions on matrices $Q \in \mathbb{R}^{m\times k}$:

- `haar-`$k$: The matrices $Q_t$ are such that $\sqrt{\frac{k}{m}}Q_t$ is distributed according to the Haar measure on $m \times k$ matrices [35].
- `norm-`$k$: Each entry of $Q_t$ is an independent normal random variable with standard deviation $\frac{1}{\sqrt{k}}$.
- `rand-`$k$: The matrices $Q_t$ are such that $\sqrt{\frac{k}{m}}Q_t$ has the distribution resulting from the removal of $m-k$ random columns from the identity matrix $I_{m\times m}$.

Note that the scaling factors in the above definitions guarantee that the resulting estimators are unbiased when used for optimization. We obtain different convergence bounds depending on the type of compression used. and in each case the impact of compression is quantified by the $\mathcal{Q}$*-seminorm*:

**Definition II.2.** Given a distribution $\mathcal{Q}$ over $\mathbb{R}^{m\times k}$, the **$\mathcal{Q}$-seminorm** is defined for matrices $A \in \mathbb{R}^{m\times m}$ as

$$\|A\|_{\mathcal{Q}} = \left\|\mathbb{E}_{Q\sim\mathcal{Q}}\left[QQ^TAQQ^T\right]\right\|. \tag{4}$$

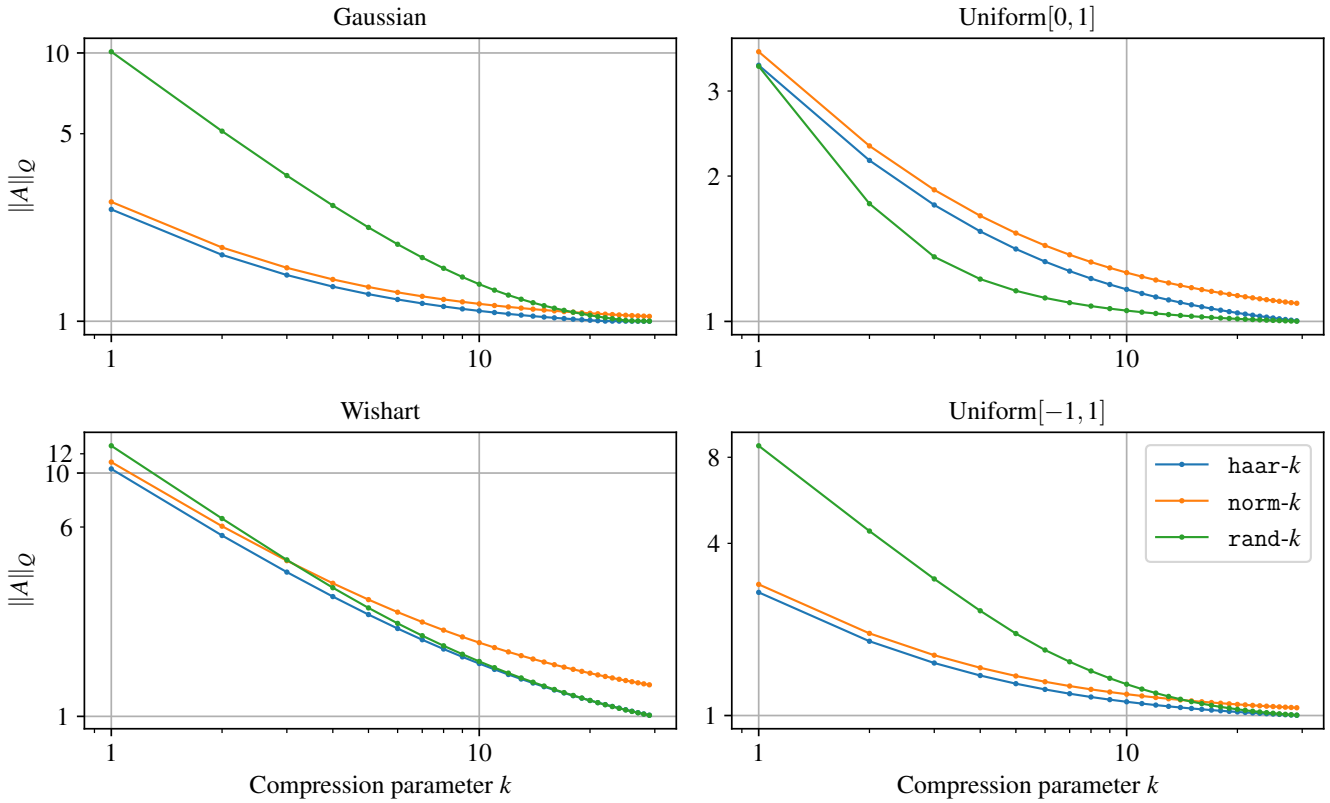


Fig. 1. Empirical values of the $\mathcal{Q}$-seminorm for matrices from several distributions, as the compression parameter varies. Upper left: Symmetric matrix with $\mathcal{N}(0,1)$ entries. Upper right: Symmetric matrix with $\mathrm{Unif}[0,1]$ entries. Lower left: Rank $r = 30$. Wishart matrix. Lower right: Symmetric matrix with $\mathrm{Unif}[-1,1]$-entries. All matrices are $30 \times 30$.

The $\mathcal{Q}$-seminorm plays a key role in our analyses below. When specialized to the case if linear regression, our convergence result depends on the $\mathcal{Q}$-seminorm of the data covariance matrix. In the general case of non-convex objectives, the convergence depends, roughly speaking, on a uniform bound on the $\mathcal{Q}$-seminorm of the Hessian matrix of the objective. In essence, the $\mathcal{Q}$-seminorm replaces the Lipschitz constant of the gradient in the analyses, and smaller values imply faster convergence.

We now state our results on the $\mathcal{Q}$-seminorm for the various compression schemes we consider. In each case, we provide an exact formula for the inner expectation and a corresponding upper bound.

**Proposition II.3.** *Let $A$ be a matrix in $\mathbb{R}^{m\times m}$. Define the the constant $\beta$ as*

$$\beta = \frac{m(m-k)}{(m+2)(m-1)}. \tag{5}$$

*Then $\|A\|_{\mathcal{Q}}$ for $Q \in \{\texttt{haar-}k, \texttt{rand-}k, \texttt{norm-}k\}$ may be computed as follows:*

$$\|A\|_{\texttt{haar-}k} = \frac{m}{k}\left\|(1-\beta)A + \beta\frac{\operatorname{tr} A}{m}I\right\|, \tag{6}$$

$$\|A\|_{\texttt{rand-}k} = \frac{m}{k}\left\|\frac{k-1}{m-1}A + \frac{m-k}{m-1}\operatorname{diag} A\right\|, \tag{7}$$

$$\|A\|_{\texttt{norm-}k} = \frac{m}{k}\left\|\frac{k+1}{m}A + \frac{\operatorname{tr} A}{m}I\right\|. \tag{8}$$

The derivations for these formulas may be found in the Appendix A; see Propositions A.1, A.3, and A.4 for $\texttt{rand-}k, \texttt{haar-}k$ and $\texttt{norm-}k$ respectively. An empirical comparison of these bounds for several classes of matrices is shown in Figure 1. Each plot in the figure indicates the $\mathbb{Q}$-seminorm of a matrix drawn from a specific distribution, for the three values of $\mathcal{Q}$ and increasing values of $k$. In general, the $\texttt{haar-}k$ norm of the matrices is smallest, followed closely by $\texttt{norm-}k$. We see that the relative performance of $\texttt{rand-}k$ depends heavily on the type of matrix, and in some cases on the compression level. We note that it is also possible to generate matrices where $\|\cdot\|_{\texttt{norm-}k} \leq \|\cdot\|_{\texttt{haar-}k}$, although this is not possible for the smallest values $k$ - for instance, when $k = 1$, we can see that $\|\cdot\|_{\texttt{haar-}k} = \frac{m}{m+2}\|\cdot\|_{\texttt{norm-}k}$.

It can be shown that, at least for $\mathcal{Q} \in \{\texttt{haar-}k, \texttt{norm-}k, \texttt{rand-}k\}$, the seminorm $\|\cdot\|_{\mathcal{Q}}$ is indeed a norm. See Proposition A.5 for details.

To motivate the use of the $\mathcal{Q}$-norm for analyzing compression, consider a simple example of optimizing a quadratic function:

**Example II.4.** Let $A$ be an $m \times m$ positive definite matrix. The objective function $f : \mathbb{R}^m \to \mathbb{R}$ is defined as $f(x) = \frac{1}{2}x^T Ax + b^T x$. Starting from an initial parameter value $x_1$, we generate a sequence of parameter values $x_1, x_2, \ldots, x_N$ according to (1). From Proposition III.4, we can upper bound the convergence rate of the gradients on the order of $\mathcal{O}\left(\frac{\|A\|_{\mathcal{Q}}}{N}\right)$. Specifically, let $\mathcal{Q}$ be a distribution on $m \times k$ matrices satisfying (11) and consider running Algorithm 1 on $f$ with exact gradients $g_t = \nabla f(x_t)$. Then, choosing the step-size $\epsilon = \frac{1}{\|A\|_{\mathcal{Q}}}$ and setting $D = [f(x_1) - \inf_{x \in \mathbb{R}^m} f(x)]$ leads to the convergence bound

$$\frac{1}{N}\sum_{t=1}^{N} \mathbb{E}\left[\|\nabla f(x_t)\|^2\right] \leq \frac{2D\|A\|_{\mathcal{Q}}}{N}. \tag{9}$$

In the case of $\texttt{haar-}k$, using (6) this is

$$\frac{1}{N}\sum_{t=1}^{N} \mathbb{E}[\|\nabla f(x_t)\|^2] \leq \frac{2D}{N}\frac{m}{k}\left((1-\beta)\|A\| + \beta\frac{\operatorname{tr} A}{m}\right)$$

On the other hand, if the only information one has is that $\texttt{haar-}k$ determines an $\omega$-unbiased compressor - which is indeed the case for $\omega = (\frac{m}{k} - 1)$ - then the above bound (9) reduces to

$$\frac{1}{N}\sum_{t=1}^{N} \mathbb{E}[\|\nabla f(x_t)\|^2] \leq \frac{2D}{N}\frac{m}{k}\|A\|,$$

which can be significantly worse depending on the matrix $A$ and the compression level. This motivates investigating convergence bounds based on the $\mathcal{Q}$-norm. □

### *A. Relation between $\mathcal{Q}$-norm and other compressor characterizations*

*1) $\omega$-unbiased compressors:* As we show below, establishing that a compressor $\mathcal{Q}$ is $\omega$-unbiased is equivalent to determining that $\|I\|_{\mathcal{Q}} \leq (\omega + 1)$.

**Proposition II.5.** *Suppose that $\mathcal{Q}$ is a distribution on matrices such that $\mathbb{E}[QQ^T] = I$. Then $\mathcal{Q}$ determines an $\omega$-unbiased compressor according to 2b iff $\|I\|_{\mathcal{Q}} \leq (\omega + 1)$.*

*Proof.* By definition, a distribution $\mathcal{Q}$ determines a $\omega$-unbiased compressor iff for any $g$ we have $\mathbb{E}[\|g - QQ^T g\|^2] \leq \omega\|g\|^2$. Expanding out the left-hand side of this equation, and using the unbiasedness property, this is equivalent to $\mathbb{E}[g^T QQ^T QQ^T g] \leq (\omega+1)\|g\|^2$. Since $g$ is arbitrary, this equation is equivalent to stating $\|\mathbb{E}[QQ^T QQ^T]\| \leq (\omega+1)$ which is exactly the condition for boundedness of $\|I\|_{\mathcal{Q}}$. □

Furthermore, it can be shown that any $\omega$-unbiased compressor satisfies $\|A\|_{\mathcal{Q}} \leq (\omega+1)\|A\|$. However, as alluded to in the Example, the benefit of the $\mathcal{Q}$ norm is that by taking into account more information about the problem it may lead to tighter optimization bounds.

*2) The quantity $\mathcal{L}(C, \mathbb{L})$ :* Another approach to capturing the interaction between a (randomized) compressor $C$ and the function $f$ is captured via the quantity $\mathcal{L}(C, \mathbb{L})$, as defined in [17], [18].

$$\mathcal{L}(C, \mathbb{L}) = \inf\{\omega \geq 0 : \forall x, \mathbb{E}\left[\|C(x) - x\|_{\mathbb{L}}^2\right] \leq \omega\|x\|^2\} \tag{10}$$

This key quantity captures the interaction between compression and these smoothness matrices, and is a dominant factor in the convergence bounds in [17], [18]. In case the underlying compressor $C$ is one of $\texttt{haar-}k$ or $\texttt{norm-}k$, the results we used to bound the $\mathcal{Q}$-norm (Propositions A.3 and A.4) could also be applied to compute $\mathcal{L}(C, \mathbb{L})$. We leave the exploration of the relationship between the $\mathcal{Q}$-norm and matrix step-sizes to future work.

## III. Optimization analysis

The following assumption that will be used in the optimization analysis:

**Assumption III.1.** Let $\mathcal{Q}$ be a distribution on matrices $Q \in \mathbb{R}^{m \times k}$ such that

$$\mathbb{E}_{Q \sim \mathcal{Q}}\left[QQ^T\right] = I_{m \times m}. \tag{11}$$

The primary assumption on our objective function $f : \mathbb{R}^n \to \mathbb{R}$ is as follows:

**Assumption III.2.** The function $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable, bounded from below by $f^*$, and, for some $\mathbb{L} \succeq 0$, satisfies the $\mathbb{L}$-smoothness criteria (3). The constant $P \geq 0$ is an upper bound on the $\mathcal{Q}$-norm of $\mathbb{L}$. That is, $\|\mathbb{L}\|_{\mathcal{Q}} \leq P$.

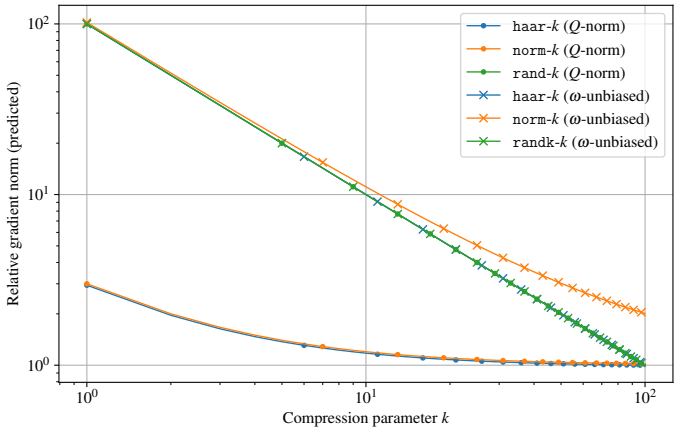


Fig. 2. Predicted convergence penalty vs compression parameter, for analyses based on the $\mathcal{Q}$-norm, and a problem-independent approach, in the case of a binary logistic regression problem.

Note that if a twice differentiable function is $L$-smooth in the usual sense (that is, having an $L$-Lipschitz continuous gradient), then it is $(L', \mathcal{Q})$-smooth for some $L'$ (e.g. by taking $L' = \|I\|_{\mathcal{Q}} L$). However, one may be able to get improved bounds on the $\mathcal{Q}$-norm by considering the distribution in the context of the specific structure of the problem; see our calculations for logistic regression in Example (III.6) below.

Next, we state our assumption on the stochastic gradients $g_i$:

**Assumption III.3.** For all $t \geq 1$ the gradient estimates $g_t$ are unbiased, meaning $\mathbb{E}[g_t - \nabla f(x_t)] = 0$. Additionally, the gradient estimates have bounded variance: there is some $\sigma \geq 0$ so that $\mathbb{E}\left[\|\nabla f(x_t) - g_t\|^2\right] \leq \sigma^2$ for all $t \geq 1$.

Using these assumptions, we can state our result on optimization performance using compression.

**Proposition III.4.** *Let Assumptions III.1, III.2 and III.3 hold. Let $\epsilon$ be chosen as*

$$\epsilon = \min\left\{\frac{1}{P}, \frac{1}{\sigma\sqrt{N}}\right\}. \tag{12}$$

*Then the iterates $x_1, x_2, \ldots$ generated by Algorithm* (1) *satisfy*

$$\begin{aligned} &\frac{1}{N}\sum_{t=1}^{N}\mathbb{E}\left[\|\nabla f(x_t)\|^2\right] \\ &\leq \frac{2DP}{N} + \left(D + \frac{P}{2}\right)\frac{2\sigma}{\sqrt{N}}. \end{aligned} \tag{13}$$

Asymptotically, this represents a $\mathcal{O}\left(\frac{P}{N} + \frac{\sigma P}{\sqrt{N}}\right)$ convergence rate, which is consistent with existing upper-bounds on non-convex SGD [38], but with the $(P, \mathcal{Q})$-smoothness constant replacing the usual gradient smoothness term. Note that as $\sigma$ tends to zero in 12 and 13, we recover a result for the case of exact gradient descent with compression, as presented below in Corollary III.5.

**Corollary III.5.** *Let Assumptions III.1, III.2 and III.3 hold with $\sigma = 0$. Let $\epsilon = \frac{1}{P}$. Then the iterates $x_1, x_2, \ldots$ from Algorithm* (1) *satisfy*

$$\frac{1}{N}\sum_{t=1}^{N}\mathbb{E}[\|\nabla f(x_t)\|^2] \leq \frac{2DP}{N}.$$

We use this corollary to illustrate some features of the bounds one can obtain in the example of binary logistic regression.

**Example III.6** (Binary logistic regression)**.** Here, $f : \mathbb{R}^m \to \mathbb{R}$ is $f(x) = \frac{1}{B}\sum_{i=1}^{B} f_i(x)$, where each $f_i$ is defined as follows: $f_i(x) = h(y_i x^T z_i)$ where $h$ is the the log softmax function $h(w) = \log\left(1 + e^{-w}\right)$, $z_i$ is a feature vector and $y_i \in \{-1, 1\}$ is the target label for the $i$th example. For the purposes of this example, we assume $\|z_i\|_2 \leq 1$. The function $f$ is bounded from below by zero and we may take $D = f(x_1)$. Noting that $|h''(w)| \leq \frac{1}{4}$ for all $w$, it can be shown (Lemma 1 in [17]) that $f$ is $\mathbb{L}$-smooth, where $\mathbb{L}$ is the matrix

$$\mathbb{L} = \frac{1}{4}\frac{1}{B}\sum_{i=1}^{B} z_i z_i^T.$$

One can use properties of $\mathbb{L}$ to potentially bound $\|\mathbb{L}\|_{\mathcal{Q}}$ and thereby obtain bounds on optimization performance. In this example we demonstrate one such derivation, although we make no claim that these are optimal bounds; they merely illustrate the ability

of this approach to differentiate among the compressors and incorporate problem information. Primarily, we use that $\|\mathbb{L}\| \leq \frac{1}{4}$ and $\operatorname{tr}\mathbb{L} = \frac{1}{4}$. Furthermore, $\|\operatorname{diag}\mathbb{L}\| \leq \frac{1}{4}$. Plugging these values into the formulas for the $\mathcal{Q}$-norms in Proposition II.3, we obtain the following upper bounds for $\|\mathbb{L}\|_{\mathcal{Q}}$:

$$\|\mathbb{L}\|_{\texttt{haar-}k} \leq \frac{m}{k}\left[(1-\beta)\|\mathbb{L}\| + \beta\frac{\operatorname{tr}\mathbb{L}}{m}\right] \leq \frac{1}{4}\frac{m}{k}\frac{k+2}{m+2},$$
$$\|\mathbb{L}\|_{\texttt{rand-}k} \leq \frac{m}{k}\left[\frac{k-1}{m-1}\|\mathbb{L}\| + \frac{m-k}{m-1}\max_i \mathbb{L}_{i,i}\right] \leq \frac{1}{4}\frac{m}{k},$$
$$\|\mathbb{L}\|_{\texttt{norm-}k} \leq \frac{m}{k}\left[\frac{k+1}{m}\|\mathbb{L}\| + \frac{\operatorname{tr}\mathbb{L}}{m}\right] \leq \frac{1}{4}\frac{k+2}{k}.$$

These estimates for $\|\mathbb{L}\|_{\mathcal{Q}}$ may in turn be combined with Corollary III.5 to obtain bounds on optimization performance. These bounds are plotted in Figure 2. They have been uniformly scaled by dividing the curves by the corresponding upper bound bound the Corollary III.5 would provide in the absence of compression, which is $\frac{D}{2N}$.

On the other hand, if no problem information is available aside from $\|\mathbb{L}\| \leq \frac{1}{4}$, we can obtain performance bounds by combining Corollary III.5 with the the inequality $\|\mathbb{L}\|_{\mathcal{Q}} \leq \|I\|_{\mathcal{Q}}\|\mathbb{L}\|$, essentially ignoring the interaction of the problem and the compressor. For this case, we use that $\|I\|_{\texttt{haar-}k} = \|I\|_{\texttt{rand-}k} = \frac{m}{k}$, while $\|I\|_{\texttt{norm-}k} = \frac{m+k+1}{k}$. Bounds derived from these inequalities are included in (2), and correspond to the $\omega$-unbiased curves.

There are two main observations from Figure (2). The first is that, for `haar-`$k$, and `norm-`$k$, the performance guarantee is significantly better when using the analysis based on the $\mathcal{Q}$-norm, especially at small values of $k$. The second is that the $\mathcal{Q}$-norm analysis provides predictions that are qualitatively different: Under the $\omega$-unbiased analysis, `norm-`$k$ is predicted to be the worst performer and `haar-`$k$ is predicted to be on-par with `rand-`$k$, while in the $\mathcal{Q}$-norm analysis, both `haar-`$k$ and `norm-`$k$ are predicted to perform significantly better than `rand-`$k$. □

## IV. Experimental results

In this section we present some experimental results on least squares and logistic regression problems. The aim of these experiments is to empirically investigate whether the bounds derived above correlate with real-world performance. We find that the experiments confirm the theory above: As a rule, `rand-`$k$ is the worst performer among the three compression schemes, while `norm-`$k$ and `haar-`$k$ have similar performance, with `haar-`$k$ having a slight advantage.

### A. Linear regression

In this setting, we consider a linear regression problem. The task is to learn a set of regression coefficients (and a bias term) that map an $n = 100$ dimensional vector to a scalar. The number of input/output pairs is 10 and they are randomly sampled from a Gaussian distribution on $\mathbb{R}^{100+1}$. In these experiments we use Corollary III.5 to compute the step-sizes that minimize the upper bound on performance. The numerical results are presented in the top row of Figure 3. Each of the plots shows a comparison of non-compressed and compressed training at various values of $k$. On the top left, we plot the ratio of the (squared) gradient norm returned by compressed and non-compressed training after a fixed number of steps $N = 40$. As would be expected due to the lossy nature of compression, non-compressed training finds a parameter with a smaller gradient and the gap between compressed and non-compressed training narrows as the compression parameter $k$ is increased. In this relatively simple case, we can exactly calculate the right-hand side of Corollary III.5 under both compressed and non-compressed training, and obtain an estimate of the relative performance of each compression scheme. This is indicated by an '$\times$' in the plots, and we see that the predicted performance comparison is within the error bars of performance in each case.

On the top right we compare the algorithms by examining the number of iterations needed by compressed and non-compressed optimization to find a point with a small gradient, defined as $\|\nabla f(x)\|^2 < 10^{-3}$. Specifically, we plot the ratio of the number of iterations needed with and without compression in order to meet this performance criteria. We find that the relative performance of the compression schemes follows a similar pattern to what was observed with the relative gradient norm. For reference, non-compressed optimization required an average of $N = 15$ iterations to meet this criteria. In both the left and right plots, the error bars represent one standard deviation of performance, measured by repeating the experiment across 500 random datasets.

### B. ImageNet fine-tuning

As a second example, we considered fine-tuning the last layer of a ResNet-18 model for image classification [39]. The initial model was pre-trained (without compression) on the ImageNet-21k dataset, after removing the 1000 classes that determine the ImageNet-1k dataset [40]. For our fine-tuning, we take this pre-trained model and train the parameters of the last layer to classify images from ImageNet-1k, using the standard training and validation split for that dataset. This fine-tuning step is equivalent to optimizing a multinomial logistic regression model, where the input features are provided by the activations at the penultimate layer of a ResNet model. For each value of $k \in [2, 4, 8, 16, 32, 64, 128]$ and each compression type (including no compression) we identified the best learning rate using a grid search in the set $\eta \in \{10^{-2+3i/14}\}_{0\leq i\leq 14}$. After the optimal

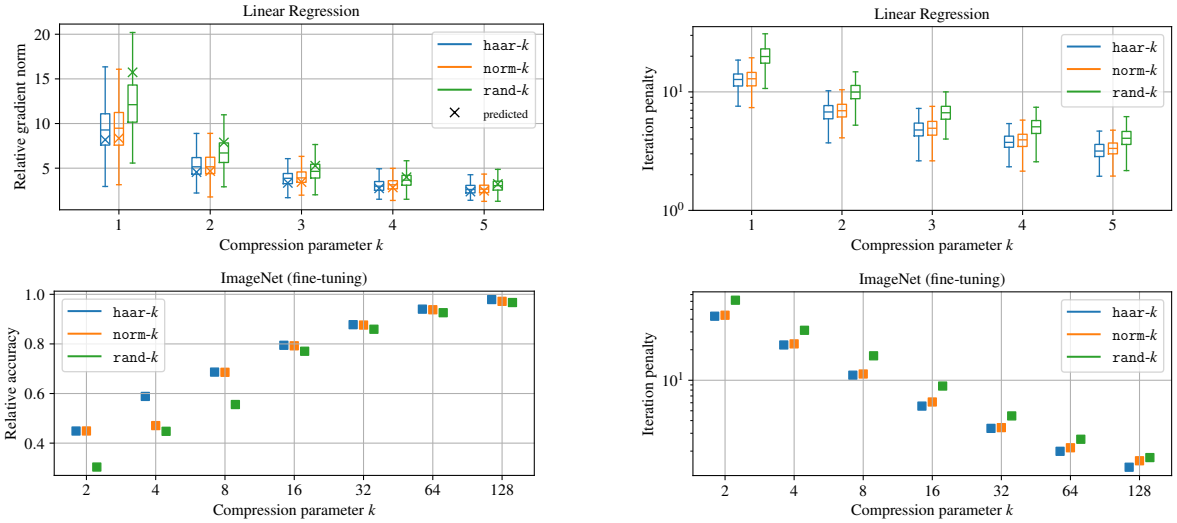


Fig. 3. Progress of optimization with and without compression. The top plots indicate performance on a linear regression task, while the bottom plots show performance on a neural network training task. Note that in the ImageNet experiments, we indicate only the mean performance at each compression parameter due to the low variability of the results.

learning rate is identified, we repeat each training scenario three times. The batch size used is $4096$, and hence our experiments model a distributed training scenario where $\ell$ nodes collaborate by providing gradients from a local minibatch of size $4096/\ell$. We plot two indicators of performance in the bottom row of Figure 3. The bottom left plot compares the accuracy of the models obtained by compressed and non-compressed training, for the various compressors and values of $k$, after running for a fixed number of iterations $N = 840$. The $y$-axis there represents the ratio of the accuracy of the compressed and non-compressed model. We see that at each $k$, the ranking of compression types agrees with what was observed for the linear regression case. For an alternative view of performance, in the right plot of Figure 3 we look at the relative number of iterations needed to reach 50% test accuracy by compressed and non-compressed optimization. That is, at each $k$ we indicate the iteration penalty for each compression scheme to obtain 50% accuracy. For reference, in these experiments it required an average of $N = 470$ iterations for non-compressed training to reach the target accuracy level. Similarly to the linear regression case, we find that `haar`-$k$ has a slight advantage over `norm`-$k$, while `rand`-$k$ performs the worst out of the three.

## V. Conclusion

In this paper, we explored the relationship between the performance of compressed training and problem structure. We looked at random linear compression schemes and determined upper bounds on optimization performance in terms of problem data. These data included properties of the Hessian, such as the norm and the trace. Our motivation was a better understanding of distributed training procedures that make use of projections. In general, the performance bounds for `norm`-$k$ and `haar`-$k$ are very similar unless the problem size is particularly small, while bounds for `rand`-$k$ are significantly worse. We believe that the results discussed here at not limited to random compression, but can be used to gain insight into more complex schemes such as PowerSGD.

## Acknowledgement

This research was supported by the U.S. Department of Energy's Office of Science under Contract No. DE-SC0012704, and used resources of the National Energy Research Scientific Computing Center (NERSC), a Department of Energy Office of Science User Facility using NERSC award ASCR-ERCAP0027539. The authors would also like to acknowledge Roxana Geambasu (Columbia University) and Pierre Tholoniat (Columbia University) for helpful discussions. In particular, P. Tholoniat developed the initial model used for this paper's fine-tuning experiments.

## References

[1] J. Dean, G. S. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Y. Ng, "Large scale distributed deep networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, USA, 2012, pp. 1223–1231.

[2] M. Zinkevich, M. Weimer, L. Li, and A. J. Smola, "Parallelized stochastic gradient descent," in *Advances in neural information processing systems*, 2010, pp. 2595–2603.

[3] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server," in *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'14. USA: USENIX Association, 2014, p. 583–598.

[4] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[5] 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," *Microsoft Research*, Sep. 2014.

[6] 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 *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., Dec. 2017, pp. 1508–1518.

[7] S. Horvóth, C.-Y. Ho, L. Horvath, A. N. Sahu, M. Canini, and P. Richtarik, "Natural Compression for Distributed Deep Learning," in *Mathematical and Scientific Machine Learning*. PMLR, Sep. 2022, pp. 129–141.

[8] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," in *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Dec. 2018, pp. 1306–1316.

[9] N. Ivkin, D. Rothchild, E. Ullah, V. Braverman, I. Stoica, and R. Arora, "Communication-efficient distributed SGD with sketching," in *NIPS'19: Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Dec. 2019, pp. 13 142–13 152.

[10] T. Vogels, S. P. Karimireddy, and M. Jaggi, "Powersgd: Practical low-rank gradient compression for distributed optimization," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.

[11] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-Shot Text-to-Image Generation," in *International Conference on Machine Learning*. PMLR, Jul. 2021, pp. 8821–8831.

[12] S. Khirirat, H. R. Feyzmahdavian, and M. Johansson, "Distributed learning with compressed gradients," *arXiv*, Jun. 2018.

[13] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified sgd with memory," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.

[14] S. P. Karimireddy, Q. Rebjock, S. U. Stich, and M. Jaggi, "Error feedback fixes SignSGD and other gradient compression schemes," in *ICML - Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 3252–3261.

[15] A. Beznosikov, S. Horváth, P. Richtárik, and M. Safaryan, "On Biased Compression for Distributed Learning," *Journal of Machine Learning Research*, vol. 24, no. 276, pp. 1–50, 2023.

[16] L. Condat, K. Yi, and P. Richtarik, "Ef-bv: A unified theory of error feedback and variance reduction mechanisms for biased and unbiased compression in distributed optimization," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 17 501–17 514.

[17] M. Safaryan, F. Hanzely, and P. Richtarik, "Smoothness Matrices Beat Smoothness Constants: Better Communication Compression Techniques for Distributed Optimization," *Advances in Neural Information Processing Systems*, vol. 34, pp. 25 688–25 702, Dec. 2021.

[18] B. Wang, M. Safaryan, and P. Richtarik, "Theoretically Better and Numerically Faster Distributed Optimization with Smoothness-Aware Quantization Techniques," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9841–9852, Dec. 2022.

[19] H. Li, A. Karagulyan, and P. Richtárik, "Det-CGD: Compressed gradient descent with matrix stepsizes for non-convex optimization," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=ZEZ0CPmoSI

[20] A. Albasyoni, M. Safaryan, L. Condat, and P. Richtárik, "Optimal Gradient Compression for Distributed and Federated Learning," *arXiv*, Oct. 2020.

[21] J. H. Korhonen and D. Alistarh, "Towards Tight Communication Lower Bounds for Distributed Optimisation," *Advances in Neural Information Processing Systems*, vol. 34, pp. 7254–7266, Dec. 2021.

[22] X. Huang, Y. Chen, W. Yin, and K. Yuan, "Lower Bounds and Nearly Optimal Algorithms in Distributed Learning with Communication Compression," *Advances in Neural Information Processing Systems*, vol. 35, pp. 18 955–18 969, Dec. 2022.

[23] S. P. Kasiviswanathan, "SGD with low-dimensional gradients with applications to private and distributed learning," in *Uncertainty in Artificial Intelligence*. PMLR, Dec. 2021, pp. 1905–1915.

[24] N. Strom, "Scalable distributed DNN training using commodity GPU cloud computing," in *Proc. Interspeech 2015*, 2015, pp. 1488–1492.

[25] C.-Y. Chen, J. Ni, S. Lu, X. Cui, P.-Y. Chen, X. Sun, N. Wang, S. Venkataramani, V. V. Srinivasan, W. Zhang, and K. Gopalakrishnan, "ScaleCom: Scalable Sparsified Gradient Compression for Communication-Efficient Distributed Training," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13 551–13 563, 2020.

[26] N. Agarwal, A. T. Suresh, F. Yu, S. Kumar, and H. B. McMahan, "cpSGD: communication-efficient and differentially-private distributed SGD," in *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Dec. 2018, pp. 7575–7586.

[27] S. U. Stich, "On Communication Compression for Distributed Optimization on Heterogeneous Data," *arXiv*, Sep. 2020.

[28] Z. Li, H. Zhao, B. Li, and Y. Chi, "SoteriaFL: A Unified Framework for Private Federated Learning with Communication Compression," *Advances in Neural Information Processing Systems*, vol. 35, pp. 4285–4300, Dec. 2022.

[29] D. Rothchild, A. Panda, E. Ullah, N. Ivkin, I. Stoica, V. Braverman, J. Gonzalez, and R. Arora, "FetchSGD: communication-efficient federated learning with sketching," in *ICML'20: Proceedings of the 37th International Conference on Machine Learning*. JMLR.org, Jul. 2020, vol. 119, pp. 8253–8265.

[30] T. Vogels, S. P. Karimireddy, and M. Jaggi, "Practical Low-Rank Communication Compression in Decentralized Deep Learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14 171–14 181, 2020.

[31] H. Zhao, B. Li, Z. Li, P. Richtarik, and Y. Chi, "BEER: Fast $O(1/T)$ Rate for Decentralized Nonconvex Optimization with Communication Compression," *Advances in Neural Information Processing Systems*, vol. 35, pp. 31 653–31 667, Dec. 2022.

[32] H. Tang, S. Gan, A. A. Awan, S. Rajbhandari, C. Li, X. Lian, J. Liu, C. Zhang, and Y. He, "1-bit Adam: Communication Efficient Large-Scale Training with Adam's Convergence Speed," in *International Conference on Machine Learning*. PMLR, Jul. 2021, pp. 10 118–10 129.

[33] S. Agarwal, H. Wang, K. Lee, S. Venkataraman, and D. Papailiopoulos, "Adaptive Gradient Communication via Critical Learning Regime Identification," *Proceedings of Machine Learning and Systems*, vol. 3, pp. 55–80, Mar. 2021.

[34] S. Shi, X. Zhou, S. Song, X. Wang, Z. Zhu, X. Huang, X. Jiang, F. Zhou, Z. Guo, L. Xie, R. Lan, X. Ouyang, Y. Zhang, J. Wei, J. Gong, W. Lin, P. Gao, P. Meng, X. Xu, C. Guo, B. Yang, Z. Chen, Y. Wu, and X. Chu, "Towards Scalable Distributed Training of Deep Learning on Public Cloud Clusters," *Proceedings of Machine Learning and Systems*, vol. 3, pp. 401–412, Mar. 2021.

[35] E. S. Meckes, *The Random Matrix Theory of the Classical Compact Groups*, ser. Cambridge Tracts in Mathematics. Cambridge University Press, 2019.

[36] F. Mezzadri, "How to generate random matrices from the classical compact groups," *Notices of the American Mathematical Society*, vol. 54, no. 5, pp. 592–604, 2007.

[37] C. Thrampoulidis and B. Hassibi, "Isotropically random orthogonal matrices: Performance of LASSO and minimum conic singular values," in *2015 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2015, pp. 14–19.

[38] S. Ghadimi and G. Lan, "Stochastic first- and zeroth-order methods for nonconvex stochastic programming," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[40] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

## APPENDIX A
## AUXILIARY RESULTS

**Proposition A.1.** *Let the $m \times k$ matrix $Q$ have the distribution of an identity matrix which has had $m-k$ random diagonal entries set to zero, and the resulting columns that are zero deleted. (That is $Q$ is distributed like the* `rand-k` *matrices described in Definition II.1, without the $\frac{m}{k}$ scaling factor.) Let $A$ be any matrix. Then $\mathbb{E}[QQ^TAQQ^T] = M \circ A$ where $\circ$ represents the element-wise product and $M$ is the matrix $M_{i,j} = \frac{k}{m}(\delta_{i,j} + (1-\delta_{i,j})\frac{k-1}{m-1})$. Consequently,*

$$\|\mathbb{E}[QQ^TAQQ^T]\| = \frac{k}{m}\left\|\frac{k-1}{m-1}A + \frac{m-k}{m-1}\operatorname{diag} A\right\|.$$

*Proof.* The random matrix $QQ^T$ is made from taking the identity matrix and setting to zero all columns that are not in the chosen set of parameters to update. Let $I$ be the indices of columns that are non-zero. We have $(QQ^T)_{i,i} = 1_{i\in I}$ and $\mathbb{E}[(QQ^T)_{i,i}] = p(i \in I)$ and $(QQ^TAQQ)_{i,j} = 1_{i\in I, j\in I}A_{i,j}$. Note that $p(i \in I) = \frac{k}{m}$. [1]. The probability $p(i \in I, j \in I)$ can be calculated according to the hypergeometric distribution. This gives

$$p(i \in I \,\&\, j \in I) = \begin{cases} \frac{k}{m} & \text{if } i = j, \\ \frac{k}{m}\frac{k-1}{m-1} & \text{if } i \neq j. \end{cases}$$

So $\mathbb{E}[QQ^TAQQ^T] = M \circ A$ where $M_{i,j} = \frac{k}{m}(\delta_{i,j} + (1-\delta_{i,j})\frac{k-1}{m-1}) = \frac{k}{m}(\frac{k-1}{m-1} + (1-\frac{k-1}{m-1})\delta_{i,j})$. Expanding the definition of $M$, we see that

$$M \circ A = \frac{k}{m}\frac{k-1}{m-1}A + \frac{k}{m}\left(1 - \frac{k-1}{m-1}\right)\operatorname{diag} A.$$

□

**Proposition A.2** ([35])**.** *Let $Q$ be distributed according to the Haar measure on $m \times m$ matrices. Then*

$$\mathbb{E}[Q_{1,1}] = 0, \tag{14}$$

$$\mathbb{E}[Q_{1,1}Q_{2,1}] = 0, \tag{15}$$

$$\mathbb{E}[Q_{1,1}^2] = \frac{1}{m}, \tag{16}$$

$$\mathbb{E}[Q_{1,1}^4] = \frac{3}{m(2+m)}, \tag{17}$$

$$\mathbb{E}[Q_{1,1}^2Q_{2,1}^2] = \frac{1}{m(m+2)}, \tag{18}$$

$$\mathbb{E}[Q_{1,1}^2Q_{2,2}^2] = \frac{m+1}{(m-1)m(m+2)}, \tag{19}$$

$$\mathbb{E}[Q_{1,1}Q_{1,2}Q_{2,1}Q_{2,2}] = -\frac{1}{(m-1)m(m+2)}. \tag{20}$$

*Proof.* These formulas can all be seen as consequences Since the Haar measure on $\mathbb{O}(m)$ is invariant under the map $Q \mapsto -Q$, we have $\mathbb{E}[Q_{1,1}] = -\mathbb{E}[Q_{1.1}]$, hence 14 holds. Equation 15 holds as the Haar measure is invariant under multiplying individual rows or columns by $-1$, since this operation can be expressed as multiplying the random element $Q$ by another orthogonal matrix. All entries of $Q$ follow the same distribution, implying that $\mathbb{E}[Q_{1,1}^2] = \mathbb{E}\left[\frac{1}{m}\sum_{i=1}^{m} Q_{i,1}^2\right] = \frac{1}{m}$, where we have used that each column has a unit norm. Hence 16 holds. Equations 17 and 18 follow from Proposition 2.5 of [35]. By symmetry, the joint distribution of $Q_{1.1}, Q_{2,2}$ is the same as that of $Q_{1,1}, Q_{i,2}$ for any $i \neq 1$. Hence

$$\begin{aligned}
\mathbb{E}[Q_{1,1}^2Q_{2,2}^2] &= \mathbb{E}\left[Q_{1,1}^2\frac{1}{m-1}\sum_{j=2}^{m}Q_{j,2}^2\right] \\
&= \mathbb{E}[Q_{1,1}^2\frac{1}{m-1}(1-Q_{1,2}^2)] \\
&= \frac{1}{m-1}\mathbb{E}[Q_{1,1}^2] - \frac{1}{m-1}\mathbb{E}[Q_{1,1}^2Q_{1,2}^2] \\
&= \frac{1}{(m-1)m} - \frac{1}{(m-1)m(m+2)} = \frac{m+1}{(m-1)m(m+2)}.
\end{aligned}$$

[1]We can show this by induction: Let $p_j$ be the probability that $i$ is in a random interval of size $j$. Clearly $p_1 = \frac{1}{m}$. For the induction, assume $p_{k-1} = \frac{k-1}{m}$. Then $p_k = p_{k-1} + (1-p_{k-1})\frac{1}{m-(k-1)} = \frac{k-1}{m} + (1-\frac{k-1}{m})\frac{1}{m-(k-1)} = \frac{k}{m}$

In the last step we used 16 and 18. This shows 19. Finally, for 20 we start with the symmetry that the joint distribution of $Q_{1,1}, Q_{1,2}, Q_{i,1}, Q_{i,2}$ is the same for all $i > 1$:

$$\begin{aligned}\mathbb{E}[Q_{1,1}Q_{1,2}Q_{2,1}Q_{2,2}] &= \frac{1}{m-1}\mathbb{E}\left[Q_{1,1}Q_{1,2}\sum_{i=2}^{m} Q_{i,1}Q_{i,2}\right] \\ &= \frac{1}{m-1}\mathbb{E}[Q_{1,1}Q_{1,2}(-Q_{1,1}Q_{1,2})] \\ &= -\frac{1}{m-1}\mathbb{E}[Q_{1,1}^2 Q_{1,2}^2].\end{aligned}$$

For the second step above, orthogonality of the columns yields $\sum_{i=2}^{m} Q_{i,1}Q_{i,2} = -Q_{1,1}Q_{1,2}$. The last step is to apply (18). □

**Proposition A.3.** *Let $A$ be an $m \times m$ symmetric matrix. Let $Q$ be distributed according to the Haar measure on $m \times k$ matrices. Then*

$$\mathbb{E}[QQ^T AQQ^T] = \frac{k}{m}\eta A + (\operatorname{tr} A)\frac{k}{m^2}\beta I \tag{21}$$

*Above, $\eta, \beta$ are as defined as:*

$$\begin{aligned}\beta &= \frac{m(m-k)}{(m-1)(m+2)} \\ \eta &= \frac{2(m-1)+(k-1)m}{(m-1)(m+2)} = 1-\beta\end{aligned} \tag{22}$$

*Proof.* Let $A$ have Eigendecomposition $A = \sum_{i=1}^{m} \lambda_i v_i v_i^T$, where $\{v_1, \ldots, v_n\}$ form an orthogonal basis. Assume that (21) holds for rank one matrices. Then

$$\begin{aligned}\mathbb{E}[QQ^T AQQ^T] &= \mathbb{E}\left[QQ^T\left(\sum_{i=1}^{m}\lambda_i v_i v_i^T\right)QQ^T\right] \\ &= \sum_{i=1}^{m}\lambda_i \mathbb{E}[QQ^T v_i v_i^T QQ^T] \\ &= \sum_{i=1}^{m}\lambda_i\left(\frac{k}{m}\eta v_i v_i^T + \frac{k}{m^2}\beta I\right) \\ &= \sum_{i=1}^{m}\lambda_i\frac{k}{m}\eta v_i v_i^T + \sum_{i=1}^{m}\lambda_i\frac{k}{m^2}\beta I\end{aligned}$$

Hence it suffices to show the claim for rank-one matrices of the form $A = vv^T$ where $\|v\| = 1$. To start, observe that

$$(QQ^T AQQ^T)_{i,j} = \sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\sum_{d=1}^{k}\sum_{c=1}^{k} Q_{i,c}Q_{r,c}Q_{a,d}Q_{j,d} \tag{23}$$

For a fixed $i, j, r, a$ the inner sum on the right of (23) has expectation

$$\begin{aligned}\sum_{d=1}^{k}\sum_{c=1}^{k}\mathbb{E}[Q_{i,c}Q_{r,c}Q_{a,d}Q_{j,d}] &= \sum_{d=1}^{k}\mathbb{E}[Q_{i,d}Q_{r,d}Q_{a,d}Q_{j,d}] + \sum_{d=1}^{k}\sum_{c\neq d}^{k}\mathbb{E}[Q_{i,c}Q_{r,c}Q_{a,d}Q_{j,d}] \\ &= k\mathbb{E}[Q_{i,1}Q_{r,1}Q_{a,1}Q_{j,1}] + k(k-1)\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}]\end{aligned}$$

Above we split up the sum based on the cases $c = d$ and $c \neq d$. Combining this with 23 gives

$$\begin{aligned}\mathbb{E}[(QQ^T AQQ^T)_{i,j}] = k\sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,1}Q_{r,1}Q_{a,1}Q_{j,1}] \\ + k(k-1)\sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}]\end{aligned} \tag{24}$$

Assume $i = j$. Then this is

$$\mathbb{E}[(QQ^T AQQ^T)_{i,i}] = k\sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,1}^2 Q_{r,1}Q_{a,1}] + k(k-1)\sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{i,1}] \tag{25}$$

For the first term on the right of (25), (ignoring the scalar $k$)

$$\begin{aligned}
&\sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,1}^2 Q_{r,1}Q_{a,1}] \\
&= \sum_{r=1}^{m} A_{r,r}\mathbb{E}[Q_{i,1}^2 Q_{r,1}^2] + \sum_{r=1}^{m}\sum_{a\neq r}^{m} A_{r,a}\mathbb{E}[Q_{i,1}^2 Q_{r,1}Q_{a,1}] \\
&= A_{i,i}\mathbb{E}[Q_{i,1}^4] + \sum_{r\neq i}^{m} A_{r,r}\mathbb{E}[Q_{1,1}^2 Q_{2,1}^2] + \sum_{r=1}^{m}\sum_{a\neq r}^{m} A_{r,a}\mathbb{E}[Q_{i,1}^2 Q_{r,1}Q_{a,1}] \\
&= A_{i,i}\mathbb{E}[Q_{i,1}^4] + \sum_{r\neq i}^{m} A_{r,r}\mathbb{E}[Q_{1,1}^2 Q_{2,1}^2] + \sum_{a\neq i}^{m} A_{i,a}\mathbb{E}[Q_{i,1}^3 Q_{a,1}] + \sum_{r\neq i}^{m}\sum_{a\neq r}^{m} A_{r,a}\mathbb{E}[Q_{i,1}^2 Q_{r,1}Q_{a,1}] \\
&= A_{i,i}\mathbb{E}[Q_{i,1}^4] + \sum_{r\neq i}^{m} A_{r,r}\mathbb{E}[Q_{1,1}^2 Q_{2,1}^2] + \sum_{r\neq i}^{m} A_{r,i}k\mathbb{E}[Q_{i,1}^3 Q_{r,1}] + \sum_{r\neq i}^{m}\sum_{a\neq r, a\neq i}^{m} A_{r,a}\mathbb{E}[Q_{i,1}^2 Q_{r,1}Q_{a,1}] \\
&= A_{i,i}k\mathbb{E}[Q_{i,1}^4] + \sum_{r\neq i}^{m} A_{r,r}\mathbb{E}[Q_{1,1}^2 Q_{2,1}^2] + \sum_{r\neq i}^{m}\sum_{a\neq r, a\neq i}^{m} A_{r,a}\mathbb{E}[Q_{1,1}^2 Q_{2,1}Q_{3,1}] \\
&= A_{i,i}\mathbb{E}[Q_{1,1}^4] + \sum_{r\neq i}^{m} A_{r,r}\mathbb{E}[Q_{1,1}^2 Q_{2,1}^2] \\
&= A_{i,i}\frac{3}{m(m+2)} + \sum_{r\neq i}^{m} A_{r,r}\frac{1}{m(m+2)}.
\end{aligned} \tag{26}$$

For the second term on the right of (25), we have

$$\begin{aligned}
\sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{i,1}] &= \sum_{r=1}^{m} A_{r,r}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{r,1}Q_{i,1}] + \sum_{r=1}^{m}\sum_{a\neq r}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{i,1}] \\
&= \sum_{r=1}^{m} A_{r,r}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{r,1}Q_{i,1}] \\
&= A_{i,i}\mathbb{E}[Q_{1,2}^2 Q_{1,1}^2] + \sum_{r\neq i}^{m} A_{r,r}\mathbb{E}[Q_{1,2}Q_{2,2}Q_{2,1}Q_{1,1}] \\
&= A_{i,i}\mathbb{E}[Q_{1,2}^2 Q_{1,1}^2] - \sum_{r\neq i}^{m} A_{r,r}\frac{1}{m-1}\mathbb{E}[Q_{1,1}^2 Q_{1,2}^2] \\
&= A_{i,i}\frac{1}{m(m+2)} - \sum_{r\neq i}^{m} A_{r,r}\frac{1}{m-1}\frac{1}{m(m+2)}
\end{aligned} \tag{27}$$

Combining (25) with (26) and (27) we get that for $i = j$,

$$\begin{aligned}
&\mathbb{E}[(QQ^T AQQ^T)_{i,i}] \\
&= kA_{i,i}\frac{3}{m(m+2)} + k\sum_{r\neq i}^{m} A_{r,r}\frac{1}{m(2+m)} + k(k-1)A_{i,i}\frac{1}{m(m+2)} - k(k-1)\sum_{r\neq i}^{m} A_{r,r}\frac{1}{m-1}\frac{1}{m(m+2)} \\
&= \frac{k}{m}\frac{(k+2)}{(m+2)}A_{i,i} + \frac{k}{m}\frac{(m-k)}{(m-1)(m+2)}\sum_{r\neq i}^{m} A_{r,r}
\end{aligned}$$

Next, assume that $i \neq j$. For the first term on the right of (24),

$$\begin{aligned}
&\sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,1}Q_{r,1}Q_{a,1}Q_{j,1}] \\
&= \sum_{r=1}^{m} A_{r,r}\mathbb{E}[Q_{i,1}Q_{r,1}^2Q_{j,1}] + \sum_{r=1}^{m}\sum_{a\neq r}^{m} A_{r,a}\mathbb{E}[Q_{i,1}Q_{r,1}Q_{a,1}Q_{j,1}] \\
&= \sum_{a\neq i}^{m} A_{i,a}\mathbb{E}[Q_{i,1}^2Q_{a,1}Q_{j,1}] + \sum_{r\neq i}^{m}\sum_{a\neq r}^{m} A_{r,a}\mathbb{E}[Q_{i,1}Q_{r,1}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{i,1}^2Q_{j,1}^2] + \sum_{r\neq i}^{m}\sum_{a\neq r}^{m} A_{r,a}\mathbb{E}[Q_{i,1}Q_{r,1}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{i,1}^2Q_{j,1}^2] + \sum_{a\neq j}^{m} A_{j,a}\mathbb{E}[Q_{i,1}Q_{a,1}Q_{j,1}^2] + \sum_{r\neq i,r\neq j}^{m}\sum_{a\neq r}^{m} A_{r,a}\mathbb{E}[Q_{i,1}Q_{r,1}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{i,1}^2Q_{j,1}^2] + A_{j,i}\mathbb{E}[Q_{i,1}^2Q_{j,1}^2] + \sum_{r\neq i,r\neq j}^{m}\sum_{a\neq r}^{m} A_{r,a}\mathbb{E}[Q_{i,1}Q_{r,1}Q_{a,1}Q_{j,1}] \\
&= 2A_{i,j}\mathbb{E}[Q_{1,1}^2Q_{2,1}^2] \\
&= A_{i,j}\frac{2}{m(m+2)}.
\end{aligned} \tag{28}$$

Above, on the first equality we split the sum based on $a = r$ and $a \neq r$, while on the second equality we split the resulting sum based on $r = i$ and $\neq i$. On the fourth we split based on $r = j$ and $r \neq j$.

For the second term on the right of 24 we have

$$\begin{aligned}
&\sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}] \\
&= \sum_{a=1}^{m} A_{i,a}\mathbb{E}[Q_{i,2}^2Q_{a,1}Q_{j,1}] + \sum_{r\neq i}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{i,2}^2Q_{j,1}^2] + \sum_{a\neq j}^{m} A_{i,a}\mathbb{E}[Q_{i,2}^2Q_{a,1}Q_{j,1}] + \sum_{r\neq i}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{1,2}^2Q_{2,1}^2] + \sum_{r\neq i}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{1,2}^2Q_{2,1}^2] + \sum_{r\neq i}^{m} A_{r,j}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{j,1}^2] + \sum_{r\neq i}^{m}\sum_{a\neq j}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{1,2}^2Q_{2,1}^2] + \sum_{r\neq i}^{m}\sum_{a\neq j}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{1,2}^2Q_{2,1}^2] + \sum_{a\neq j}^{m} A_{j,a}\mathbb{E}[Q_{i,2}Q_{j,2}Q_{a,1}Q_{j,1}] + \sum_{r\neq i,r\neq j}^{m}\sum_{a\neq j}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}].
\end{aligned}$$

On the first equality, we split the sum based on $r = i$ and $r \neq i$. On the second equality we split the sum based on $a = j$ and $a \neq j$. On the fourth equality we split based on $a = j$ and $a \neq j$. On the sixth equality we split based on $r = j$, $r \neq j$. Noting

that the double sum on the final equation vanishes, we continue with

$$\begin{aligned}
&\sum_{r=1}^{m}\sum_{a=1}^{m} A_{r,a}\mathbb{E}[Q_{i,2}Q_{r,2}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{1,2}^2Q_{2,1}^2] + \sum_{a\neq j}^{m} A_{j,a}\mathbb{E}[Q_{i,2}Q_{j,2}Q_{a,1}Q_{j,1}] \\
&= A_{i,j}\mathbb{E}[Q_{1,2}^2Q_{2,1}^2] + A_{j,i}\mathbb{E}[Q_{i,2}Q_{j,2}Q_{i,1}Q_{j,1}] + \sum_{a\neq j, a\neq i}^{m} A_{j,a}\mathbb{E}[Q_{i,2}Q_{j,2}Q_{a,1}Q_{j,1}] \quad \text{(split on } a=j \text{ and} a\neq j) \\
&= A_{i,j}\mathbb{E}[Q_{1,2}^2Q_{2,1}^2] + A_{j,i}\mathbb{E}[Q_{1,2}Q_{2,2}Q_{1,1}Q_{2,1}] \\
&= A_{i,j}\mathbb{E}[Q_{1,2}^2Q_{2,1}^2] - A_{j,i}\frac{1}{m-1}\mathbb{E}[Q_{1,1}^2Q_{1,2}^2] \\
&= A_{i,j}\frac{m+1}{(m-1)m(m+2)} - A_{j,i}\frac{1}{(m-1)m(m+2)}.
\end{aligned} \tag{29}$$

Combining (24), (28), and (29) we see that when $i \neq j$,

$$\begin{aligned}
\mathbb{E}[(QQ^TAQQ^T)_{i,j}] &= \left(\frac{2k}{m(m+2)} + \frac{k(k-1)(m+1)}{(m-1)m(m+2)} - \frac{k(k-1)}{(m-1)m(m+2)}\right) v_iv_j \\
&= \left(\frac{2k(m-1)+k(k-1)(m+1)-k(k-1)}{(m-1)m(m+2)}\right) v_iv_j \\
&= \frac{k}{m}\left(\frac{2(m-1)+(k-1)m}{(m-1)(m+2)}\right) v_iv_j.
\end{aligned}$$

We summarize the above as follows:

$$\mathbb{E}[(QQ^TAQQ^T)_{i,j}] = \begin{cases} \frac{k}{m}\alpha v_i^2 + \frac{k}{m^2}\beta\sum_{r\neq i} v_r^2 & \text{if } i=j, \\ \frac{k}{m}\eta v_iv_j & \text{else} \end{cases} \tag{30}$$

where $\alpha = \frac{k+2}{m+2}$ and $\beta, \eta$ are as in (22). Note that we can use the fact that $\|v\| = 1$ and $\alpha - \frac{\beta}{m} = \eta$ to rewrite (30) as

$$\mathbb{E}[(QQ^TAQQ^T)_{i,j}] = \begin{cases} \frac{k}{m}\eta v_i^2 + \frac{k}{m^2}\beta & \text{if } i=j, \\ \frac{k}{m}\eta v_iv_j & \text{else.} \end{cases} \tag{31}$$

□

**Proposition A.4.** *Let $\mathcal{Q}$ be an $m \times k$ whose entries are i.i.d, following the normal distribution $\mathcal{N}(0,1)$. (That is, $\mathcal{Q}$ is distributed like the* `norm`*-$k$ matrices described in Definition II.1, without the $\frac{1}{\sqrt{k}}$ scaling factor.). Then*

$$\mathbb{E}[QQ^TAQQ^T] = k(k+1)A + k(\operatorname{tr} A)I$$

*Proof.* The proof proceeds along the same lines as the proof of Proposition A.3, with simplifications due to the fact the entries of the matrix $Q$ are now i.i.d. □

**Proposition A.5.** *Let $\mathcal{Q} \in \{$*`haar`*-$k$, `norm`*-$k$, `rand`*-$k\}$. Then $\|\cdot\|_{\mathcal{Q}}$ determines a norm on $m \times m$ matrices.*

*Proof.* Let $\mathcal{Q} =$ `haar`*-$k$* and suppose that $\|A\|_{\mathcal{Q}} = 0$. Equation (6) implies $A = -\frac{\beta}{1-\beta}\frac{\operatorname{tr} A}{m}I$. Hence $A$ is a scalar multiple of the identity matrix, and taking traces we see $\operatorname{tr} A = -\frac{\beta}{1-\beta}\operatorname{tr} A$, implying $\operatorname{tr} A = 0$. Since the diagonal elements of $A$ are all equal, we must have $A = 0$. The proof for `norm`*-$k$* is almost the same. Suppose that $\mathcal{Q} =$ `rand`*-$k$* and $\|A\|_{\mathcal{Q}} = 0$. According to (7), we must have $A = -\alpha \operatorname{diag} A$, for $\alpha = \frac{m-k}{k-1} > 0$. Hence $A$ is a diagonal matrix and $\operatorname{diag} A = -\alpha \operatorname{diag} A$, implying $A = 0$. □

## APPENDIX
## PROOFS FOR SECTION III (OPTIMIZATION ANALYSIS)

**Proposition III.4.** *Let Assumptions III.1, III.2 and III.3 hold. Let $\epsilon$ be chosen as*

$$\epsilon = \min\left\{\frac{1}{P}, \frac{1}{\sigma\sqrt{N}}\right\}. \tag{12}$$

*Then the iterates $x_1, x_2, \ldots$ generated by Algorithm* (1) *satisfy*

$$\begin{aligned}&\frac{1}{N}\sum_{t=1}^{N}\mathbb{E}\left[\|\nabla f(x_t)\|^2\right]\\ &\leq \frac{2DP}{N} + \left(D + \frac{P}{2}\right)\frac{2\sigma}{\sqrt{N}}.\end{aligned} \tag{13}$$

*Proof.* Each $x_{t+1}$ may be expressed as $x_{t+1} = x_t - \epsilon Q_t Q_t^T g_t$. By $\mathbb{L}$-smoothness, we have

$$\mathbb{E}[f(x_{t+1})] \leq f(x_t) - \epsilon\mathbb{E}[\nabla f(x_t)^T Q_t Q_t^T g_t] + \frac{\epsilon^2}{2}\mathbb{E}[g_t^T Q_t Q_t^T \mathbb{L} Q_t Q_t^T g_t]$$

Writing $g_t = \nabla f(x_t) + \delta_t$, where $\delta_t = g_t - \nabla f(x_t)$ is a mean-zero error term, and using the bound on the $\mathcal{Q}$-norm of $\mathbb{L}$,

$$\begin{aligned}\mathbb{E}[g_t^T Q_t Q_t^T \mathbb{L} Q_t Q_t^T g_t] &= \mathbb{E}[(\nabla f(x_t) + \delta_t)^T Q_t Q_t^T \mathbb{L} Q_t Q_t^T (\nabla f(x_t) + \delta_t)]\\ &= \mathbb{E}[\nabla f(x_t)^T Q_t Q_t^T \mathbb{L} Q_t Q_t^T \nabla f(x_t)] + \mathbb{E}[\delta_t Q_t Q_t^T \mathbb{L} Q_t Q_t^T \delta_t]\\ &\leq \mathbb{E}[\|\nabla f(x_t)\|^2]P + \sigma^2 P\end{aligned}$$

Hence

$$\mathbb{E}[f(x_{t+1})] \leq \mathbb{E}[f(x_t)] - \epsilon\mathbb{E}[\|\nabla f(x_t)\|^2] + \frac{\epsilon^2}{2}\mathbb{E}[\|\nabla f(x_t)\|^2]P + \frac{\epsilon^2}{2}\sigma^2 P.$$

From here, the proof proceeds along standard lines [38]. Rearranging the terms and simplifying, we get

$$\epsilon\left(1 - \frac{\epsilon}{2}P\right)\mathbb{E}[\|\nabla f(x_t)\|^2] \leq \mathbb{E}[f(x_t)] - \mathbb{E}[f(x_{t+1})] + \frac{\epsilon^2}{2}\sigma^2 P.$$

Summing this from $t = 1, \ldots N$ we get

$$\sum_{t=1}^{N}\epsilon\left(1 - \frac{\epsilon}{2}P\right)\mathbb{E}[\|\nabla f(x_t)\|^2] \leq \sum_{t=1}^{N}\mathbb{E}[f(x_t)] - \mathbb{E}[f(x_{t+1})] + N\frac{\epsilon^2}{2}\sigma^2 P.$$

This implies

$$\frac{1}{N}\sum_{t=1}^{N}\epsilon\left(1 - \frac{\epsilon}{2}P\right)\mathbb{E}[\|\nabla f(x_t)\|^2] \leq \frac{D}{N} + \frac{\epsilon^2}{2}\sigma^2 P$$

where $D = f(x_1) - f^*$. Let $\epsilon$ be as in (12). Then

$$\frac{1}{\epsilon} = \max\left\{P, \sigma\sqrt{N}\right\} \leq P + \sigma\sqrt{N}.$$

Combining the above, we obtain

$$\begin{aligned}\frac{1}{N}\sum_{t=1}^{N}\mathbb{E}[\|\nabla f(x_t)\|^2] &\leq \frac{D}{N\epsilon}\frac{1}{1 - \epsilon P/2} + \frac{P}{2}\sigma^2\epsilon\frac{1}{1 - \epsilon P/2}\\ &= \frac{D}{N}\frac{1}{1 - \epsilon P/2}\left(P + \sigma\sqrt{N}\right) + \frac{P}{2}\sigma^2\frac{1}{\sigma\sqrt{N}}\frac{1}{1 - \epsilon P/2} +\\ &= \frac{1}{N}\left(\frac{DP}{(1 - \epsilon P/2)}\right) + \frac{\sigma}{\sqrt{N}}\left(D + \frac{P}{2}\right)\frac{1}{1 - \epsilon P/2}\\ &\leq \frac{1}{N}(2DP) + \frac{2\sigma}{\sqrt{N}}\left(D + \frac{P}{2}\right).\end{aligned}$$

□