Quantized Distributed Training of Large Models with Convergence Guarantees

Ilia Markov [1] **Adrian Vladu** [2 3] **Qi Guo** [4] **Dan Alistarh** [1]

Abstract

Communication-reduction techniques are a popular way to improve scalability in data-parallel training of deep neural networks (DNNs). The recent emergence of large language models such as GPT has created the need for new approaches to exploit data-parallelism. Among these, fully-sharded data parallel (FSDP) training is highly popular, yet it still encounters scalability bottlenecks. One reason is that applying compression techniques to FSDP is challenging: as the vast majority of the communication involves the model's weights, direct compression alters convergence and leads to accuracy loss. We present QSDP, a variant of FSDP which supports both gradient and weight quantization with theoretical guarantees, is simple to implement and has essentially no overheads. To derive QSDP we prove that a natural modification of SGD achieves convergence even when we only maintain quantized weights, and thus the domain over which we train consists of quantized points and is, therefore, highly non-convex. We validate this approach by training GPT-family models with up to 1.3 billion parameters on a multi-node cluster. Experiments show that QSDP preserves model accuracy, while completely removing the communication bottlenecks of FSDP, providing end-to-end speedups of up to 2.2x.

1. Introduction

The impressive recent progress of Deep Learning in tasks such as natural language processing and computer vision has been accompanied by massive increases in parameter counts. For instance, large language models (LLMs) from Transformer family, such as GPT (Radford et al., 2018), OPT (Zhang et al., 2022) and BLOOM (Laurençon et al., 2022) easily count billions of trainable parameters, which induces tremendous computational and memory costs. Training such models can easily exceed the memory capacity of a single computational unit, such as a GPU.

As a consequence, standard distribution strategies such as *data-parallel training* (Bottou, 2010), which require each node to be able to keep all parameters in memory, are no longer directly applicable. Several novel distribution strategies have been proposed to mitigate this challenge, such as *model-parallel training* (Shoeybi et al., 2019; Raffel et al., 2020), *pipeline-parallel training* (Huang et al., 2019; Harlap et al., 2018) and *model sharding* (Ren et al., 2021; Rajbhandari et al., 2020; Rasley et al., 2020; FairScale, 2021).

We consider the communication costs of distribution strategies for massive models, and focus on *Fully-Sharded Data-Parallel (FSDP)* distributed training, which is among the most popular and user-friendly approaches to mitigate per-node memory limitations. FSDP is supported natively by Pytorch (Paszke et al., 2019), Facebook `fairscale` (FairScale, 2021), and Microsoft DeepSpeed (Ren et al., 2021), where it is known as ZeRO-3.

The main idea behind FSDP is that both the training data *and the model parameters* are partitioned among the P nodes. That is, only a $1/P$ partition of the parameters of each layer is stored at a node. Then, both for the forward and for the backward pass, nodes proceed synchronously layer-by-layer, gathering full weights for the current layer, via *all-to-all communication*, before executing its forward or backward operation. After this operation is complete, nodes can discard the current layer's received weights partitions, and move to the next layer. (Please see Figure 1 for an illustration, and Section 4.1 for a detailed description.)

The key advantage of this pattern is that it reduces memory usage linearly in P. Thus, it enables running models with billions of parameters on small or medium-sized clusters (FairScale, 2021; MosaicML, 2022). At the same time, FSDP faces challenges in terms of *communication efficiency*: since every forward and backward pass relies on all-to-all weight exchanges, FSDP can put massive pressure on the network bandwidth, which becomes a bottleneck.

As we will show, all-to-all communication leads to significant communication bottlenecks when training LLMs on

[1]Institute of Science and Technology Austria [2]CNRS [3]IRIF, Université Paris Cité [4]Max Planck Institute for Informatics. Correspondence to: Ilia Markov <ilia.markov@ist.ac.at>, Dan Alistarh <dan.alistarh@ist.ac.at>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

multi-node clusters. Two key challenges to removing this communication bottleneck are that 1) a majority of FSDP's communication are *layer weights*: quantizing them naively loses theoretical convergence, and can easily lead to practical divergence; 2) the FSDP setting poses stringent compute and memory constraints, restricting the set of approaches.

Contribution. We propose the first communication-efficient variant of FSDP, called QSDP, which provides both convergence guarantees, and strong practical performance. QSDP is inspired by a new analysis of SGD convergence with *full quantization of transmitted model state*. That is, we show that a simple modified variant of SGD can allow both weights and gradients to be quantized during training, without additional per-node memory, nor costly local computation. We find the fact that this is possible with convergence guarantees surprising, since nodes only observe *biased estimators* of the gradients, taken over quantized weights, without any error-correction (Karimireddy et al., 2019). From the practical perspective, our approach is fast and easy to implement, and completely removes the communication bottlenecks of FSDP, while recovering accuracy for billion-parameter GPT models.

At a high level, the QSDP algorithm simply performs weight and gradient quantization before the corresponding FSDP all-to-all communication steps. While gradient compression can be performed using standard unbiased compressors, e.g. (Alistarh et al., 2017), weight compression is performed using a carefully-designed unbiased estimator. Our key contribution is in the analysis: we model the training process as a new instance of sparse recovery (Blumensath & Davies, 2008; Foucart, 2012), in which 1) the projection step is performed via quantization and not sparsification, and 2) the gradient step is itself quantized. This connection allows us to prove, under analytic assumptions, that QSDP converges towards a minimizer of the loss over the set of lattice points corresponding to the quantization being employed. We believe this is the first instance of such an analysis. The analysis could potentially be improved by further leveraging structure, as it currently requires increasing the sparsity by a quadratic factor involving the "condition number" of the underlying objective, which, although inherent to this type of approach (Axiotis & Sviridenko, 2022), may in theory be quite large. However, despite this shortcoming, it paves the way towards a principled algorithm, with excellent practical behavior.

We complement our analysis with an efficient implementation of QSDP in Pytorch (Paszke et al., 2019), which we validate by training LLMs from the GPT family (Radford et al., 2018; Zhang et al., 2022) between 125M and 1.3B parameters, on a multi-node multi-GPU environment on Amazon EC2. Our experiments first show that communication bottlenecks can significantly impact standard FSDP in this standard practical setting, and that QSDP essentially removes such bottlenecks, without impact on accuracy. For example, QSDP can train GPT-1.3B to essentially the same perplexity up to 2.2x faster than standard FSDP on a 10Gbps network. In addition, we also introduce a "learned" adaptive weight quantization approach which can further reduce bit-width, without significant accuracy impact.

2. Related Work

Over the past decade, there has been a massive amount of work on communication-efficient variants of Data-Parallel SGD, e.g. (Seide et al., 2014; Dryden et al., 2016; Alistarh et al., 2017; Vogels et al., 2019; Tang et al., 2019; Wang et al., 2018). (Please see (Ben-Nun & Hoefler, 2019) for a survey.) The vast majority of this work focuses on gradient compression, the main communication cost of SGD, and is thus mostly orthogonal to our work. The massive scale of recent deep models, e.g. (Chowdhery et al., 2022; Brown et al., 2020) has led to significant work on novel distribution strategies (Ren et al., 2021; Rajbhandari et al., 2020; Rasley et al., 2020; FairScale, 2021) adapted to the requirements of these models, among which FSDP is a standard approach, e.g. (Chowdhery et al., 2022). While there is recent work on further optimizing FSDP, e.g. (Jiang et al., 2022; Miao et al., 2022), we are the first to investigate and address its communication costs. Our results are part of a broader line of work using different techniques to make the training of massive models amenable to standard infrastructure, e.g. (Wang et al., 2022; Yuan et al., 2022; Borzunov et al., 2022).

Quantized weight exchange during training has been investigated independently in the context of decentralized distributed learning. Tang et al. (2018) presents a scheme which supports quantized weight exchange by having each node extrapolate each of its neighbors' model values; yet, this would require unrealistic $\Theta(Pd)$ extra memory in our case. Similarly, other work in this vein (Koloskova et al., 2019; Nadiradze et al., 2021; Lu & De Sa, 2020) either requires additional storage, or would not fit the FSDP algorithm structure. Both our analysis approach and our algorithms' guarantees are different relative to this line of work.

Recently, there has been a surge of interest in *post-training* quantization approaches for large language models, which reduce the deployment costs of already trained models (Yao et al., 2022; Dettmers et al., 2022; Frantar et al., 2022; Xiao et al., 2022). Our work is complementary, in the sense that we show that quantized weights and gradient representations can be applied *during training*, without accuracy loss, leading to training speedup. On the other hand, these post-training approaches would be too expensive to be executed for compression at training time.

A parallel line of work aims to perform *fully-quantized* training of DNNs (Banner et al., 2018; Zhu et al., 2020). One

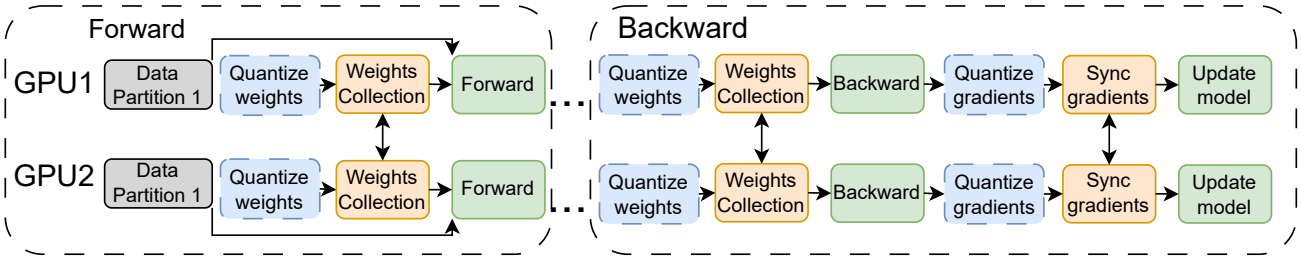


Figure 1. Scheme of (Quantized) Fully Sharded Data Parallel algorithm. During forward pass we collect the missing partitions of layer's weights, compute its activations and discard the partitions. At backward pass, we collect the weights again, compute the gradients, synchronize the gradients corresponding to our partition.

general finding from this line of work is that integrating weight and gradient quantization *into training* is extremely challenging, even when using 8bit precision, from both accuracy and performance perspectives. Specifically, this line of work investigates model modifications via e.g. parameter tuning and specialized normalization layers, in order to recover accuracy. By contrast, we preserve model structure, and do not modify hyper-parameter values, although we only quantize transmitted state.

3. Background and Motivation

3.1. Data-Parallel Training

In this classic SGD distribution pattern (Bottou, 2010), each node (e.g., GPU) holds a copy of the model, and the data is partitioned among the nodes. Each training step samples a subset of the data called a *batch*, performs a *forward pass* over the batch to obtain model predictions, and then performs a *backward pass* to obtain gradient updates. Finally, nodes communicate their local gradient updates in all-to-all fashion to keep the model in sync.

3.2. Gradient Compression

Transmitting gradients is the key communication cost of Data-Parallel SGD, and there has been a tremendous amount of work on addressing the resulting bandwidth bottleneck (Seide et al., 2014; Dryden et al., 2016; Strom, 2015). (As this area is extremely vast, we refer to Ben-Nun & Hoefler (2019); Liu et al. (2020b) for a full overview.) Of these, gradient quantization is a particularly-popular technique, which has the advantage that variants of it can be implemented without additional memory cost. A simple example is the QSGD technique (Alistarh et al., 2017), which is essentially a codebook compression method which maps each gradient value to a point on a uniform grid, via randomized rounding. For this, values are first scaled to the range $[-1, 1]$, and then each scaled coordinate v is mapped to one of the endpoints of its quantization interval $v \in [q_i, q_{i+1}]$ via the following rule:

$$q(v) = \begin{cases} q_i, \text{with probability } \frac{v - q_i}{q_{i+1} - q_i}, \\ q_{i+1}, \text{otherwise.} \end{cases}$$

It is easy to see that this gradient estimator is unbiased with respect to the stochastic quantization, and that its variance can be bounded by the norm of the original gradient. We will revisit this scheme in Sections 4.3 and 5.

3.3. Fully-Sharded Data-Parallel Training

As the name suggests, FSDP starts from the Data-Parallel (DP) approach. The main observation is that nodes do not necessarily need to store the full set of parameters at every stage of training, in particular during the backward pass. Specifically, we use the scarce GPU memory to represent only those layers which are in the forward-backward "working set" at a given moment of time.

Initially, model parameters are partitioned, so that each of the P workers is assigned a distinct $1/P$ fraction of each layer's weights. At each optimization step (see Figure 1, ignoring the dashed quantization operations), before the forward pass on a layer, each worker collects the missing partitions from other workers, computes the output activations, discards the received partitions and proceeds to the next layer. For the backward pass, workers again collect all layer weights, compute the gradients, synchronize them, discard the layer weights and proceed to the next layer. Technically, each optimization step consists of two AllGather collective operations for weights, and one Reduce-Scatter to sync gradients (full pseudocode in Appendix A).

One can easily check that the above approach implements the standard SGD iteration one-to-one, relative to a sequential execution. If we denote by $\boldsymbol{y}_t$ the model's parameter vector used at iteration t, and by $\boldsymbol{g}(\boldsymbol{y}_t)$ the average of the nodes' stochastic gradients at step t, taken at $\boldsymbol{y}_t$, then, for learning rate η, we can model the iteration as

$$\boldsymbol{y}_{t+1} = \boldsymbol{y}_t - \eta \boldsymbol{g}(\boldsymbol{y}_t). \tag{1}$$

FSDP with Compression. The natural way to reduce the cost of weight and gradient transmission in the above scheme would be to simply quantize them before transmission. (Please see the full Figure 1.) To examine the impact of adding compression on the above SGD iteration, let us consider abstract quantization operators $\mathbf{Q}^{\mathbf{w}}$ applied to the weights, and $\mathbf{Q}^{\mathbf{g}}$ applied to the gradients. (We will specify these quantization functions precisely in Section 4.3, and the exact implementation in Section 5.) For iteration $t \geq 0$, let $\boldsymbol{v}_t$ be a "virtual" view of the model weights at the beginning of iteration t, obtained by aggregating all the weights, across all the weight partitions, *in full precision*.

First, notice that, if we apply $\mathbf{Q}^{\mathbf{w}}$ before all transmissions, then the algorithm will only observe the *quantized* version of $\boldsymbol{v}_t$, which we denote by $\mathbf{Q}^{\mathbf{w}}(\boldsymbol{v}_t)$. Then, we can re-write one iteration of the algorithm as

$$\boldsymbol{v}_{t+1} = \mathbf{Q}^{\mathbf{w}}(\boldsymbol{v}_t) - \eta \mathbf{Q}^{\mathbf{g}}(\boldsymbol{g}\left(\mathbf{Q}^{\mathbf{w}}(\boldsymbol{v}_t)\right)).$$

This formulation inspires the notation $\boldsymbol{x}_t = \mathbf{Q}^{\mathbf{w}}(\boldsymbol{v}_t)$, as the algorithm only "sees" the quantized version of the full-precision weights. Then, we get the following iteration:

$$\boldsymbol{x}_{t+1} = \mathbf{Q}^{\mathbf{w}}(\boldsymbol{x}_t - \eta \mathbf{Q}^{\mathbf{g}}(\boldsymbol{g}\left(\boldsymbol{x}_t\right))), \tag{2}$$

which would correspond to an abstractly-quantized version of FSDP. This iteration is the starting point for our analysis in the next section.

4. SGD with Quantized Weights and Provable Convergence

The cornerstone of our method consists of a stochastic gradient method that provably converges to a good *quantized* iterate, under reasonable analytic assumptions. One main novelty is that it converges despite the fact that the domain is *non-convex*. At a very high level, it is similar to the iterative hard thresholding (IHT) method, which achieves provable guarantees despite the fact that it seeks a good iterate in the set of vectors of bounded sparsity (Blumensath & Davies, 2008). Throughout this section, we abstract away specifics of the system architecture, since they are not relevant to our analysis. We explain their relationship to the actual implementation in Section 5.

4.1. Background and Assumptions

The main challenge we face is to obtain quantized solutions to optimization problems that seek to minimize a function $f : \mathbb{R}^n \to \mathbb{R}$:

$$\min_{\boldsymbol{x} \in \mathcal{G}} f(\boldsymbol{x}), \tag{3}$$

where the domain $\mathcal{G}$ is a lattice that allows for an efficient communication of its elements. We restrict our attention to shifts of the lattice $\delta\mathbb{Z}^n$ along the direction of the all-ones vector. Formally, $\mathcal{G} = \{\delta\mathbb{Z}^n + r\mathbf{1} : r \in [-\delta/2, \delta/2)\}$.

Overview. Even in the case where f is convex, the non-convex structure of $\mathcal{G}$ makes it incredibly difficult to obtain a good minimizer to f without suffering a large loss. In fact, problems of this form are generally NP-hard. However, we show that when f is reasonably well-conditioned we can obtain strong convergence guarantees. The idea consists of alternating stochastic gradient descent steps with applications of a quantization operator $\mathbf{Q}^{\mathbf{w}}$ which projects the new iterate back onto a certain subset of $\mathcal{G}$. Letting $\boldsymbol{g}\left(\boldsymbol{x}_t\right)$ be a stochastic gradient, and δ a parameter which determines the coarseness of the quantization grid that we project onto, our update at step $t+1$ has the following form:

$$\boldsymbol{x}_{t+1} = \mathbf{Q}^{\mathbf{w}}_{\delta}\left(\boldsymbol{x}_t - \eta \boldsymbol{g}\left(\boldsymbol{x}_t\right)\right). \tag{4}$$

This formulation covers the practical case where the stochastic gradient $\boldsymbol{g}\left(\boldsymbol{x}_t\right)$ corresponds to a mini-batch stochastic gradient. Indeed, as in practice f takes the form $f(\boldsymbol{x}) = \frac{1}{Pm}\sum_{i=1}^{P}\sum_{j=1}^{m} f(\boldsymbol{x}; \boldsymbol{y}_j)$, where $S = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_m\}$ are data samples, and $f_i(\boldsymbol{x})$ are loss functions at individual nodes, the stochastic gradients obtained via backpropagation takes the form $\frac{1}{|B|}\sum_{j \in B} \nabla f_i(\boldsymbol{x}; \boldsymbol{y}_j)$, where i is a random node, and B is a sampled mini-batch.

Quantization by Random Shift. For weight quantization, we consider the following unbiased stochastic quantization method. To quantize a vector, we first sample a single fixed random scalar r, then shift all the coordinates of the vector by r. For vector encoding, it rounds each coordinate to the nearest neighbor on the quantization grid, and sends the lattice coordinates of the resulting vector, together with the scalar r. For decoding, it takes the lattice point, and undoes the shift on the quantized coordinates. The notable difference between this and more standard quantization methods, e.g.(Alistarh et al., 2017), is that quantization is *dependent* across coordinates. In exchange for losing independence, it allows us to provide stronger guarantees in the context of weight quantization. We define it formally:

Definition 1 (quantization by random shift)**.** Let $\delta > 0$ be a scalar defining the coarseness of the quantization grid. Let a scalar $r \in [-\delta/2, \delta/2)$, and let the deterministic operator $\mathbf{Q}^{\mathbf{w}}_{r,\delta} : \mathbb{R} \to \mathbb{R}$ which rounds to the nearest element in $\delta\mathbb{Z} + r$:

$$\mathbf{Q}^{\mathbf{w}}_{r,\delta}\left(x\right) = \delta \cdot \left\lfloor \frac{x - r}{\delta} \right\rceil + r\,.$$

Define the randomized quantization operator $\mathbf{Q}^{\mathbf{w}}_{\delta} : \mathbb{R} \to \mathbb{R}$ via $\mathbf{Q}^{\mathbf{w}}_{\delta}\left(x\right) = \mathbf{Q}^{\mathbf{w}}_{r,\delta}\left(x\right)$, for a random $r \sim \mathrm{Unif}\left([-\delta/2, \delta/2)\right)$. We apply $\mathbf{Q}^{\mathbf{w}}_{\delta}$ to vectors, with the meaning that it is dependently applied to each coordinate for a single random shift r.

We use this quantization method to show that, for a well conditioned loss function, and appropriate grid parameters, the iteration (4) converges, under reasonable analytical assumptions, to a set of weights that are comparable in quality to the best possible quantized weights from a slightly coarser grid. We note that to further reduce communication costs, our method also supports gradient quantization in addition to weight quantization, provided that gradients are quantized using an (arbitrary) unbiased estimator.

Analytical Assumptions. Formally, our analysis uses the following assumptions on f.

1. Unbiased gradient estimators with variance σ: $\mathbb{E}\left[\boldsymbol{g}\left(\boldsymbol{x}\right)|\boldsymbol{x}\right] = \nabla f\left(\boldsymbol{x}\right)$.
2. For $\beta > 0$, the β-smoothness condition: for all $\boldsymbol{x}, \boldsymbol{\Delta}$,
$$f\left(\boldsymbol{x}+\boldsymbol{\Delta}\right) \leq f\left(\boldsymbol{x}\right) + \left\langle \nabla f\left(\boldsymbol{x}\right), \boldsymbol{\Delta}\right\rangle + \frac{\beta}{2}\left\|\boldsymbol{\Delta}\right\|_2^2 .$$
3. For $\alpha > 0$, the Polyak-Łojasiewicz (α-PL) condition:
$$\frac{1}{2}\left\|\nabla f\left(\boldsymbol{x}\right)\right\|_2^2 \geq \alpha\left(f\left(\boldsymbol{x}\right) - f^*\right) ,$$
where $f^* = \min_{\boldsymbol{x}} f\left(\boldsymbol{x}\right)$.

The first two assumptions are standard in stochastic optimization (see e.g. (Lin et al., 2019)). The Polyak-Łojasiewicz (PL) condition (Karimi et al., 2016) is common in non-convex optimization, and versions of it are essential in the analysis of DNN training (Liu et al., 2020a; Allen-Zhu et al., 2019). In words, it states that small gradient norm, i.e. approximate stationarity, implies closeness to optimum in function value.

4.2. Main Theoretical Results

We are now ready to state our main analytical result.

Theorem 2. *Let $\alpha, \beta, \delta_\star, \varepsilon > 0$ and $\sigma \geq 0$ be real parameters, and let $\eta = \min\left\{\frac{3}{10}\frac{\varepsilon\alpha}{\sigma^2}, 1\right\}$. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a β-smooth and α-PL function, with access to a stochastic gradient $\boldsymbol{g}\left(\boldsymbol{x}\right)$, i.e. $\mathbb{E}\left[\boldsymbol{g}\left(\boldsymbol{x}\right)|\boldsymbol{x}\right] = \nabla f\left(\boldsymbol{x}\right)$ with bounded variance $\mathbb{E}\left\|\boldsymbol{g}\left(\boldsymbol{x}\right) - \nabla f\left(\boldsymbol{x}\right)\right\|_2^2 \leq \sigma^2$. For each $r \in [-\delta_\star/2, \delta_\star/2)$, let $\boldsymbol{x}^\star_{r,\delta_\star}$ be any minimizer of f over $\delta_\star\mathbb{Z}^n + r\mathbf{1}$. Let $\delta = \frac{\eta}{\lceil 16(\beta/\alpha)^2\rceil}\cdot\delta_\star$. Consider the iteration:*
$$\boldsymbol{x}_{t+1} = \mathbf{Q}^{\mathbf{w}}_\delta\left(\boldsymbol{x}_t - \frac{\eta}{\beta}\boldsymbol{g}\left(\boldsymbol{x}_t\right)\right) .$$
In $T = \frac{10}{\eta}\cdot\frac{\beta}{\alpha}\ln\frac{f(\boldsymbol{x}_0)-\mathbb{E}f(\boldsymbol{x}^\star_{r,\delta_\star})}{\varepsilon}$ iterations we obtain a point $\boldsymbol{x}_T$ satisfying $\mathbb{E}f\left(\boldsymbol{x}_T\right) - \mathbb{E}f(\boldsymbol{x}^\star_{r,\delta_\star}) \leq \varepsilon$.

Discussion. To understand the convergence of this method, let us establish as benchmark the expected value $\mathbb{E}f(\boldsymbol{x}^\star_{r,\delta_\star})$ of the best iterate on the lattice $\delta_\star\mathbb{Z}^n + r\mathbf{1}$, where the expectation is taken over the shift r. Our method finds a point in a slightly finer grid $\boldsymbol{x}_T \in \delta\mathbb{Z}^n + r'\mathbf{1}$, such that in expectation over the randomness in the algorithm, the value of the function is at most ϵ larger than our benchmark. The sacrifice we have to make in exchange for this surprisingly strong convergence is an increase in resolution for the iterates we maintain, which is dependent, among others, on the condition number of f. It appears that without further assumptions, the dependence in condition number can be quite large – unfortunately even the best-known analyses of "standard" iterative hard thresholding (the sparsity correspondent to our projection approach) have quadratic dependency in the condition number, as shown in (Axiotis & Sviridenko, 2022), so bounds of this type are inherent even for simple examples, like ℓ_2 regression.

Since our method works with stochastic gradients, we can additionally quantize gradients to further reduce communication. In fact any quantization method that compresses gradients to an unbiased estimator with low variance can be directly plugged into Theorem 2.

We state in the following corollary a generic bound for quantized gradients, which highlights the trade-off between variance and communication for the quantization method.

Corollary 3 (Gradient Quantization)**.** *Let $\alpha, \beta, \delta_\star, \varepsilon, b > 0$ and $\sigma, \sigma_\nabla \geq 0$ be real parameters. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a β-smooth and α-PL function, with access to a stochastic gradient estimator $\boldsymbol{g}\left(\boldsymbol{x}\right)$, i.e. $\mathbb{E}\left[\boldsymbol{g}\left(\boldsymbol{x}\right)|\boldsymbol{x}\right] = \nabla f\left(\boldsymbol{x}\right)$ with bounded variance $\mathbb{E}\left\|\boldsymbol{g}\left(\boldsymbol{x}\right) - \nabla f\left(\boldsymbol{x}\right)\right\|_2^2 \leq \sigma^2$. Let $\mathbf{Q}^{\mathbf{g}} : \mathbb{R}^n \to \mathbb{R}$ be a gradient quantizer which for any stochastic gradient $\boldsymbol{g}\left(\boldsymbol{x}\right)$ encountered during the execution of the algorithm, ensures:*

1. *unbiased estimator: $\mathbb{E}\left[\mathbf{Q}^{\mathbf{g}}(\boldsymbol{g}\left(\boldsymbol{x}\right))|\boldsymbol{g}\left(\boldsymbol{x}\right)\right] = \boldsymbol{g}\left(\boldsymbol{x}\right)$,*
2. *variance: $\mathbb{E}\left[\left\|\mathbf{Q}^{\mathbf{g}}(\boldsymbol{g}\left(\boldsymbol{x}\right)) - \boldsymbol{g}\left(\boldsymbol{x}\right)\right\|_2^2 |\boldsymbol{g}\left(\boldsymbol{x}\right)\right] \leq \sigma_\nabla^2$,*
3. *requires b bits to communicate $\mathbf{Q}^{\mathbf{g}}(\boldsymbol{g}\left(\boldsymbol{x}\right))$.*

For each $r \in [-\delta_\star/2, \delta_\star/2)$, let $\boldsymbol{x}^\star_{r,\delta_\star}$ be any minimizer of f over $\delta_\star\mathbb{Z}^n + r\cdot\mathbf{1}$. Let $\eta = \min\left\{\frac{3}{10}\frac{\varepsilon\alpha}{\sigma^2+\sigma_\nabla^2}, 1\right\}$, $\delta = \frac{\eta}{\lceil 16(\beta/\alpha)^2\rceil}\cdot\delta_\star$, and consider the iteration:
$$\boldsymbol{x}_{t+1} = \mathbf{Q}^{\mathbf{w}}_\delta\left(\boldsymbol{x}_t - \frac{\eta}{\beta}\mathbf{Q}^{\mathbf{g}}\left(\boldsymbol{g}\left(\boldsymbol{x}_t\right)\right)\right) .$$
In $T = \frac{10}{\eta}\cdot\frac{\beta}{\alpha}\ln\frac{f(\boldsymbol{x}_0)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)}{\varepsilon}$ iterations we obtain a point $\boldsymbol{x}_T$ satisfying $\mathbb{E}f\left(\boldsymbol{x}_T\right) - \mathbb{E}f(\boldsymbol{x}^\star_{r,\delta_\star}) \leq \varepsilon$. Furthermore, the entire algorithm requires $O\left(b\cdot\frac{\sigma^2+\sigma_\nabla^2}{\varepsilon\alpha}\frac{\beta}{\alpha}\ln\frac{f(\boldsymbol{x}_0)-\mathbb{E}f(\boldsymbol{x}^\star_{r,\delta_\star})}{\varepsilon}\right)$ bits to communicate the quantized gradients.

We notice that since b and σ_∇ are inversely associated, we can establish a trade-off between the number of iterations and the total communication. As a matter of fact, this trade-off kicks in only at the point where the variance of the quantized gradient estimator becomes as large as that of the stochastic gradient, as the number of iterations scales linearly with $\sigma^2 + \sigma_\nabla^2$. For example, given a resolution parameter $\delta_\nabla > 0$, a simple gradient quantization scheme such as the one employed by QSGD, quantizes gradient entries to $\delta_\nabla \mathbb{Z}^n$, while guaranteeing $\sigma_\nabla^2 \leq \delta_\nabla G_{\ell_1}$, where $G_{\ell_1} \geq \|\boldsymbol{g}(\boldsymbol{x})\|_1$, and $b = O(G_{\ell_1}/\delta_\nabla \cdot (\ln n + \ln G_{\ell_1}))$ bits required for communication. See a more detailed discussion in Section E.3. By varying δ_∇ we distinguish between the extreme cases, corresponding to the scenarios where the quantized gradients are dense and sparse, respectively. While the total communication does not improve by varying δ_∇, by doing so we are able to reduce the communication performed in each iteration, in practice (Alistarh et al., 2017).

Dense gradients: setting $\delta_\nabla = \sigma^2/G_{\ell_1}$, we obtain exactly the same number of iterations as in the basic case without quantized gradients, but the communication per iteration is reduced to $O(G_{\ell_1}^2/\sigma^2 \cdot (\ln n + \ln G_{\ell_1}))$.

Sparse gradients: setting $\delta_\nabla = G_{\ell_1}$, the number of iterations scales with $\max\left\{\sigma^2, G_{\ell_1}^2\right\}$ rather than σ^2, but the pre-step communication is reduced to $O\left(\ln n + \ln G_{\ell_1}\right)$ bits.

4.3. Analysis Overview

Let us briefly explain the intuition behind our theoretical analyses. We view our iteration as a version of projected gradient descent, where iterates are projected onto the non-convex domain of quantized vectors. In general, when the domain is convex, projections do not hurt convergence. But in our setting the distance to the optimal solution can increase and drastically affect the loss. However, we can show a trade-off between how much this distance increases and the ratio between the target and optimal resolution $\delta/\delta_\star$.

To understand this better, consider a point $\boldsymbol{x'}$ obtained by taking a step $\boldsymbol{x'} = \mathbf{Q}_\delta^{\mathbf{w}}(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x}))$. Using smoothness, we can verify that this significantly decreases the loss, provided that the quantization operator does not perturb its input by too much in ℓ_2 norm. Formally, using Lemma 7 we see that

$$f(\boldsymbol{x'}) \leq f(\boldsymbol{x}) - \frac{1}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2 + \frac{\beta}{2}\left\|\mathbf{Q}_\delta^{\mathbf{w}}\left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right) - \left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right)\right\|_2^2 . \tag{5}$$

Since compared to a vanilla gradient method, this suffers a reduction in the progress made in a single iteration, we can force this to be significantly smaller, so as not to undo more than a fraction of the progress we would ideally make. To do so we notice that we can change the last term in (5) to the current error in function value, and we can make this dependence be arbitrarily small by using a finer resolution δ for our quantization grid. This is captured by the following crucial lemma, which we prove in Section E.4.2.

Lemma 4. *Let $\delta_\star > \delta > 0$, such that $\delta_\star/\delta \in \mathbb{Z}$. Let $\boldsymbol{x} \in \mathbb{R}^n$, and for all $r \in [-\delta_\star/2, \delta_\star/2)$, let an arbitrary $\boldsymbol{x}_{r,\delta_\star}^\star \in \delta_\star\mathbb{Z}^n + r\mathbf{1}$. Then*

$$\mathbb{E}\left[\|\mathbf{Q}_\delta^{\mathbf{w}}(\boldsymbol{x}) - \boldsymbol{x}\|_2^2\right] \leq \frac{\delta}{\delta_\star}\mathbb{E}_r\left[\left\|\boldsymbol{x}_{r,\delta_\star}^\star - \boldsymbol{x}\right\|_2^2\right] .$$

Using Lemma 4, together with the α-PL condition, we can change the extra error term to

$$\frac{\beta}{2} \cdot \frac{\delta}{\delta_\star}\mathbb{E}_r\left[\frac{2}{\alpha}\left(f\left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right) - f(\boldsymbol{x}_{r,\delta_\star}^\star)\right)\right] ,$$

where $\boldsymbol{x}_{r,\delta_\star}^\star$ are picked to be the best minimizers in $\delta_\star\mathbb{Z}^n + r\mathbf{1}$. To simplify the exposition and highlight the main ideas, let us assume that $\mathbb{E}_r[f(\boldsymbol{x}_{r,\delta_\star}^\star)] = f(\boldsymbol{x}^\star)$. Since by the α-PL condition we know that the gradient norm is large compared to the error in function value, we conclude that

$$\begin{aligned} f(\boldsymbol{x'}) - f(\boldsymbol{x}^\star) &\leq f(\boldsymbol{x}) - f(\boldsymbol{x}^\star) - \frac{\alpha}{\beta}\left(f(\boldsymbol{x}) - f(\boldsymbol{x}^\star)\right) \\ &\quad + \frac{\beta}{\alpha} \cdot \frac{\delta}{\delta_\star}\left(f\left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right) - f(\boldsymbol{x}^\star)\right) \\ &\leq \left(f(\boldsymbol{x}) - f(\boldsymbol{x}^\star)\right)\left(1 - \frac{\alpha}{\beta} + \frac{\beta}{\alpha}\frac{\delta}{\delta_\star}\right) . \end{aligned}$$

This shows that by setting the $\delta \leq \delta_\star \cdot (\alpha/\beta)^2/2$, in each iteration the error contracts by a $1 - \Theta(\alpha/\beta)$ factor, which allows us to conclude that this algorithm converges linearly to a minimizer. We provide full proofs in Section E.

5. QSDP Implementation

5.1. Overview

We implemented a practical version of the QSDP algorithm described in the previous section, supporting both weight and gradient quantization, in Pytorch (Paszke et al., 2019) starting from the PyTorch FSDP support. Our implementation uses the CGX framework (Markov et al., 2022) as a communication backend, to which we added support for quantized AllGather and Reduce-Scatter collectives.

In the original FSDP implementation, layers are packed into groups: weights and gradients of layers in the same group are concatenated before communication. In QSDP, we compress layers separately, filtering out normalization layers and biases, which are communicated in full precision. This filtering is implemented at the level of the CGX communication backend. The quantized AllGather and Reduce-Scatter operations are implemented by leveraging peer-to-peer NVIDIA

NCCL primitives. For multi-node (inter-server) communication, we used hierarchical versions of the algorithms, to reduce the size of inter-node transmissions.

One important optimization regards the granularity at which quantization is performed. Specifically, applying quantization over large tensors suffers from scaling issues, which results in accuracy degradation. To address this, we perform compression independently into equally-sized "buckets" of fixed size, and compress each bucket independently. This approach sacrifices compression by a negligible amount (as we need to transmit min-max scaling meta-information for each bucket), but helps avoid loss in terms of model quality.

Bucketing (or grouping) on the weights is known to be necessary for good accuracy when quantizing *pre-trained* LLMs (Dettmers & Zettlemoyer, 2022). It is also justified theoretically (Theorem 2), as it both reduces compression variance, and allows us to explore solutions over finer-grained lattices. We observed experimentally that bucket size 1024 provides a good balance between compression and accuracy, and use it as a universal hyper-parameter. In the context of this optimization, we observed that the impact of stochasticity in the quantization becomes minimal.

5.2. Learned Weight Quantization

We now describe an additional (optional) optimization, which allows us to further reduce practical bit-width, at little to no accuracy loss. The motivating observation behind this optimization is that the quantization schemes we use for weights and gradients assume *uniform* locations of the quantization levels. Yet, this uniform grid does not take the distribution of values into account. The idea of adapting the locations of quantization levels to the data distribution has already been studied (Zhang et al., 2017; Faghri et al., 2020). However, existing dynamic-programming approaches Zhang et al. (2017) have high computational cost (quadratic in the number of data points); thus, we use a fast version of gradient-descent-based optimization over the quantization levels (Faghri et al., 2020).

The goal of the distribution-aware quantizer in Algorithm 1 is to select new locations for a fixed number of quantization points, and weight values, so as to minimize the error introduced by quantization. The algorithm runs iteratively across all values, finds the locations of quantization points for each value, and updates the *quantization points* using the gradient descent update rule. We run this heuristic periodically after a warmup period, separately for the weights and gradients of each layer. We save the derived locations of the quantization levels, and use them for quantization until the next re-computation.

Algorithm 1 Gradient-based Optimization of the Levels

Input: values V, initial levels Q_0, learning rate α.
Output: optimized quantization levels Q.
Normalize values V bucket-wise.
for each value v_i from V **do**
  $q_i = \mathsf{find_closest}(v, Q_i)$ // Quantize using current level
  $q_i = q_i - \alpha(q_i - v_i)$ // Update chosen quantization level
end for

6. Experimental Validation

6.1. Experimental setup

Infrastructure. We evaluate QSDP for training GPT-scale LLMs using multiple cloud-grade Amazon EC2 `p3dn.24xlarge` machines, with 8 V100 SXM2 GPUs each. Each GPU has 32GB memory. The inter-GPU interconnect is provisioned by NVLinks with 200Gbps, while the inter-server bandwidth is of 100 Gbps.

Environment and Tasks. We use the official NGC PyTorch 22.05-py3 Docker image with PyTorch 1.12, CUDA 11.6.2, NCCL 2.12, and the MosaicML Composer library (version 0.12), as well as a fork of the CGX communication library (Markov et al., 2022). All experiments were run with with MosaicML Large Language Models implementation (Mos, 2022). The benchmarks run the pre-training of different version of GPT-family models (Radford et al., 2018; Brown et al., 2020), varying the sizes of the models, on the C4 dataset (Raffel et al., 2020). Specifically, we examine accuracy on GPT models with 125M, 350M, and 1.3B parameters. For benchmarks, we use 4 servers with 8 GPUs each. See Appendix A for training details.

Baselines. As a baseline we use training with default parameters, which is already highly-optimized by MosaicML (Mos, 2022). We note that directly using INT8 quantization, without bucketing, resulted in very poor accuracy, and therefore we do not use it as a baseline. In terms of communication, the baseline transmits weights in full (FP32) precision, and gradients in half (FP16) precision. In QSDP experiments, we *do not* modify any hyperparameters. We convert gradients to full precision before quantization. For all timing experiments, the reported numbers are averaged over 50 training steps after warm-up of 10 iterations. Our main accuracy measure is *perplexity*, which is known to be a very stringent accuracy measure in this setting, and correlates extremely well with zero-shot performance (Dettmers et al., 2022).

Accuracy Recovery. We first examine the effect of quantization on model quality, i.e. final model perplexity, in the end-to-end experiments. The default bit-width for weights and gradients quantization is 8 bits, using 1024 bucket size, which we illustrate as W8G8. We communicate normalization layers and biases in full precision. We emphasize

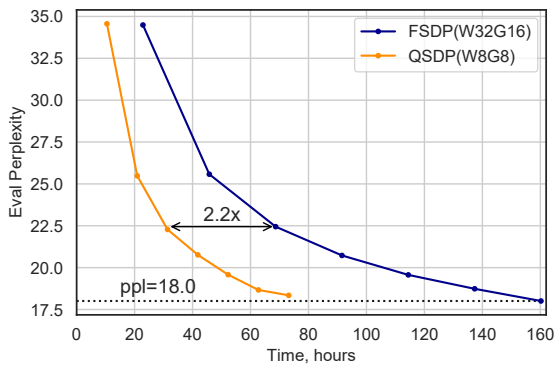


Figure 2. Perplexity vs time for standard FSDP (FP32 weights and FP16 gradients) and QSDP (both weights and gradients quantized to 8 bits) for the 1.3B model in the 10Gbps bandwidth setup.

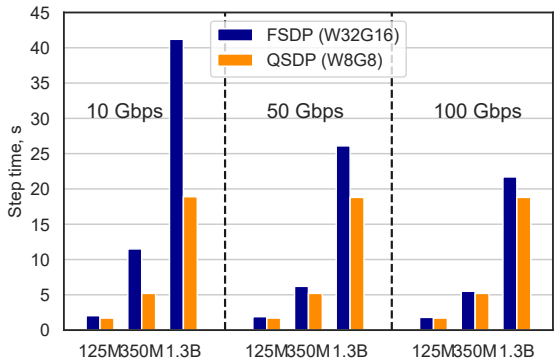


Figure 3. Training step time for different models at various inter-node bandwidth with and without QSDP enabled. The fact that QSDP step time is constant across considered bandwidths means that QSDP successfully tackles bandwidth bottlenecks.

that straightforward round-to-nearest or stochastic quantization *does not converge* to reasonable final perplexity in this setup: Naive quantization without bucketing loses more than 2 units of perplexity on GPT-125M, a model on which W8G8 with 1024 bucket size *improves* perplexity.

The accuracy results are presented in Table 1. The QSDP final perplexity is almost identical to that of regular training, and QSDP can even *slightly improve* the baseline accuracy. We stress that we did not perform any parameter tuning: quantization parameters are the same across all layers.

End-to-end Speedup. For end-to-end training speedup improvements, we use multi-node GPT pretraining under standard hyperparameters. We examine speedup for different inter-node bandwidths: 10 Gbits, 50 Gbits and 100 Gbits. For that, we artificially reduce input-output bandwidth on each node, using the UNIX `tc` tool (TC, 2001). The results are presented in Figure 3. First, please notice that standard FSDP training has a non-trivial bandwidth bottleneck even at 100Gbps bandwidth as we increase model size, and that this bandwidth bottleneck can dominate training time on the lower 10Gbps bandwidth. Second, the running time of QSDP is *essentially constant* across all three scenarios, showing that it has essentially removed the bandwidth bottleneck. More precisely, QSDP outperforms the baseline by up to 15% in the 100Gbps scenario (a non-trivial reduction of 12 hours of training time or 1.5k$ of cloud costs[1]), and by 2.25x in the 10Gbps scenario.

Learned quantization. We examined the performance of learned quantization for the small 125M parameters model. We ran the optimization algorithm after 400, 1900 and 3800 training steps, and noticed that optimizing the locations of quantization levels has no effect for bit-widths higher than 6 bits, but leads to noticeable improvements for lower bit-widths. Please see Table 3. Learned weight quantization allows to improve the final model performance for different weight and gradient quantization parameter pairs, reaching perplexities that are close to the baseline. Specifically, using learned quantization results in reaching the highest compression ratio for weights and gradient in training (i.e. 5 and 4 bits respectively) without substantial accuracy impact. We expand upon these experiments in Appendix C.

[1]price of 12 hours training on 4 AWS p3dn.24xlarge instances

Table 1. Perplexities recoveries for different models end-to-end training using QSDP. Weights and gradients quantized to 8 bits, uniform quantization.

	125M	350M	1.3B
Baseline	35.81	23.94	18.00
QSDP	35.58	23.95	18.34

Table 2. Final perplexities of training 125m GPT-2 model with combinations of weights and gradients low-bits uniform (not learned) quantization.

Weights bits \ Gradients bits	6	5	4
6	35.74	36.08	35.84
5	36.01	35.94	36.36
4	37.11	37.38	37.61

7. Conclusion

Motivated by the efficient distributed training of large language models, we have explored the feasibility of fully-quantized training for such models, with respect to both weights and gradients. This led to an interesting new analysis, showing that SGD can indeed converge with strong convergence guarantees even with quantized iterates, as long as a good quantized solution exists.

Complementing this analysis, we proposed QSDP, a quan-

Table 3. Final perplexities of low-bits quantization of 125m GPT-2 model using the learned quantization levels. Learned quantization in the W6G4 configuration provides lower perplexity than the baseline.

	baseline	w6g4	w5g4	w4g4	w4g32
Uniform	35.81	35.81	36.34	37.61	37.11
Learned		**35.75**	**36.01**	**36.94**	**36.55**

tized extension of the popular Fully Sharded Data Parallel (FSDP) distributed training approach, in which *all transmitted state is in quantized form*. We also provided a highly-efficient implementation of QSDP in Pytorch, which we showed to successfully eliminate the bandwidth bottleneck in large-scale distributed training of modern language models, without sacrificing accuracy. Specifically, our experimental validation across three model sizes showed that training with QSDP reaches up to 2.2× speedup.

Our results suggest that communication compression can be an effective tool in the context of novel distribution schemes motivated by large-scale training. Specifically, we believe we are the first to show both convergence guarantees and strong practical performance for simple *weight compression* schemes being applied during SGD-based training, which should motivate further work in this direction. In particular, an interesting extension of our work would be to examine whether the lower-precision weight representation can also be exploited for faster runtimes.

Acknowledgments

The authors gratefully acknowledge funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 805223 ScaleML), as well as experimental support from the IST Austria IT department, in particular Stefano Elefante, Andrei Hornoiu, and Alois Schloegl.

AV acknowledges the support of the French Agence Nationale de la Recherche (ANR), under grant ANR-21-CE48-0016 (project COMCOPT), the support of Fondation Hadamard with a PRMO grant, and the support of CNRS with a CoopIntEER IEA grant (project ALFRED).

References

Mosaicml examples. https://github.com/mosaicml/examples, 2022.

Alistarh, D., Grubic, D., Li, J., Tomioka, R., and Vojnovic, M. QSGD: Communication-efficient sgd via gradient quantization and encoding. *Advances in Neural Information Processing Systems*, 30:1709–1720, 2017.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. pp. 242–252. PMLR, 2019.

Axiotis, K. and Sviridenko, M. Iterative hard thresholding with adaptive regularization: Sparser solutions without sacrificing runtime. In *International Conference on Machine Learning*, pp. 1175–1197. PMLR, 2022.

Banner, R., Hubara, I., Hoffer, E., and Soudry, D. Scalable methods for 8-bit training of neural networks. *Advances in neural information processing systems*, 31, 2018.

Ben-Nun, T. and Hoefler, T. Demystifying parallel and distributed deep learning: An in-depth concurrency analysis. *ACM Computing Surveys (CSUR)*, 52(4):1–43, 2019.

Blumensath, T. and Davies, M. E. Iterative thresholding for sparse approximations. *Journal of Fourier analysis and Applications*, 14(5-6):629–654, 2008.

Borzunov, A., Baranchuk, D., Dettmers, T., Ryabinin, M., Belkada, Y., Chumachenko, A., Samygin, P., and Raffel, C. Petals: Collaborative inference and fine-tuning of large models. *arXiv preprint arXiv:2209.01188*, 2022.

Bottou, L. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Dettmers, T. and Zettlemoyer, L. The case for 4-bit precision: k-bit inference scaling laws. *arXiv preprint arXiv:2212.09720*, 2022.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. LLM.int8(): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

Dryden, N., Jacobs, S. A., Moon, T., and Van Essen, B. Communication quantization for data-parallel training of deep neural networks. In *Proceedings of the Workshop on Machine Learning in High Performance Computing Environments*, pp. 1–8. IEEE Press, 2016.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Faghri, F., Tabrizian, I., Markov, I., Alistarh, D., Roy, D. M., and Ramezani-Kebrya, A. Adaptive gradient quantization for data-parallel sgd. *Advances in neural information processing systems*, 33:3174–3185, 2020.

FairScale. Fairscale: A general purpose modular pytorch library for high performance and large scale training. https://github.com/facebookresearch/fairscale, 2021.

Foucart, S. Sparse recovery algorithms: sufficient conditions in terms of restricted isometry constants. In *Approximation Theory XIII: San Antonio 2010*, pp. 65–77. Springer, 2012.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Harlap, A., Narayanan, D., Phanishayee, A., Seshadri, V., Devanur, N., Ganger, G., and Gibbons, P. Pipedream: Fast and efficient pipeline parallel dnn training. *arXiv preprint arXiv:1806.03377*, 2018.

Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, D., Chen, M., Lee, H., Ngiam, J., Le, Q. V., Wu, Y., et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *Advances in neural information processing systems*, 32, 2019.

Jiang, Y., Miao, X., Nie, X., and Cui, B. Osdp: Optimal sharded data parallel for distributed deep learning. *arXiv preprint arXiv:2209.13258*, 2022.

Karimi, H., Nutini, J., and Schmidt, M. Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 795–811. Springer, 2016.

Karimireddy, S. P., Rebjock, Q., Stich, S., and Jaggi, M. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, pp. 3252–3261. PMLR, 2019.

Koloskova, A., Stich, S., and Jaggi, M. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, pp. 3478–3487. PMLR, 2019.

Laurençon, H., Saulnier, L., Wang, T., Akiki, C., del Moral, A. V., Le Scao, T., Von Werra, L., Mou, C., Ponferrada, E. G., Nguyen, H., et al. The bigscience roots corpus: A 1.6 tb composite multilingual dataset. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Lin, T., Stich, S. U., Barba, L., Dmitriev, D., and Jaggi, M. Dynamic model pruning with feedback. 2019.

Liu, C., Zhu, L., and Belkin, M. Toward a theory of optimization for over-parameterized systems of non-linear equations: the lessons of deep learning. *arXiv preprint arXiv:2003.00307*, 2020a.

Liu, J., Zhang, C., et al. Distributed learning systems with first-order methods. *Foundations and Trends® in Databases*, 9(1):1–100, 2020b.

Lu, Y. and De Sa, C. Moniqua: Modulo quantized communication in decentralized sgd. In *International Conference on Machine Learning*, pp. 6415–6425. PMLR, 2020.

Markov, I., Ramezanikebrya, H., and Alistarh, D. Cgx: Adaptive system support for communication-efficient deep learning, 2022. URL https://arxiv.org/abs/2111.08617.

Miao, X., Wang, Y., Jiang, Y., Shi, C., Nie, X., Zhang, H., and Cui, B. Galvatron: Efficient transformer training over multiple gpus using automatic parallelism. *arXiv preprint arXiv:2211.13878*, 2022.

MosaicML. Mosaic LLMs (part 2): Gpt-3 quality for <$500k, 2022. URL https://www.mosaicml.com/blog/gpt-3-quality-for-500k.

Nadiradze, G., Sabour, A., Davies, P., Li, S., and Alistarh, D. Asynchronous decentralized sgd with quantized and local updates. *Advances in Neural Information Processing Systems*, 34:6829–6842, 2021.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training. 2018.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P. J., et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

Rajbhandari, S., Rasley, J., Ruwase, O., and He, Y. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16. IEEE, 2020.

Rasley, J., Rajbhandari, S., Ruwase, O., and He, Y. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3505–3506, 2020.

Ren, J., Rajbhandari, S., Aminabadi, R. Y., Ruwase, O., Yang, S., Zhang, M., Li, D., and He, Y. {ZeRO-Offload}: Democratizing {Billion-Scale} model training. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pp. 551–564, 2021.

Seide, F., Fu, H., Droppo, J., Li, G., and Yu, D. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In *Fifteenth annual conference of the international speech communication association*. Citeseer, 2014.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Strom, N. Scalable distributed dnn training using commodity gpu cloud computing. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

Tang, H., Zhang, C., Gan, S., Zhang, T., and Liu, J. Decentralization meets quantization. *arXiv preprint arXiv:1803.06443*, 2018.

Tang, H., Yu, C., Lian, X., Zhang, T., and Liu, J. Doublesqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression. In *International Conference on Machine Learning*, pp. 6155–6165. PMLR, 2019.

TC, U. *tc(8) Linux User's Manual*, December 2001.

Vogels, T., Karinireddy, S. P., and Jaggi, M. Powersgd: Practical low-rank gradient compression for distributed optimization. *Advances In Neural Information Processing Systems 32 (Nips 2019)*, 32, 2019.

Wang, H., Sievert, S., Charles, Z., Liu, S., Wright, S., and Papailiopoulos, D. ATOMO: Communication-efficient learning via atomic sparsification. 2018.

Wang, J., Yuan, B., Rimanic, L., He, Y., Dao, T., Chen, B., Re, C., and Zhang, C. Fine-tuning language models over slow networks using activation compression with guarantees. *arXiv preprint arXiv:2206.01299*, 2022.

Xiao, G., Lin, J., Seznec, M., Demouth, J., and Han, S. Smoothquant: Accurate and efficient post-training quantization for large language models. *arXiv preprint arXiv:2211.10438*, 2022.

Yao, Z., Aminabadi, R. Y., Zhang, M., Wu, X., Li, C., and He, Y. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. *arXiv preprint arXiv:2206.01861*, 2022.

Yuan, B., He, Y., Davis, J. Q., Zhang, T., Dao, T., Chen, B., Liang, P., Re, C., and Zhang, C. Decentralized training of foundation models in heterogeneous environments. *arXiv preprint arXiv:2206.01288*, 2022.

Zhang, H., Li, J., Kara, K., Alistarh, D., Liu, J., and Zhang, C. Zipml: Training linear models with end-to-end low precision, and a little bit of deep learning. In *International Conference on Machine Learning*, pp. 4035–4043. PMLR, 2017.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Zhu, F., Gong, R., Yu, F., Liu, X., Wang, Y., Li, Z., Yang, X., and Yan, J. Towards unified int8 training for convolutional neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1969–1979, 2020.

Algorithm 2 Pseudocode of QSDP for a Fixed Layer

Input: worker p, layer input x_p, worker weight partition w_p.
function ExecuteForwardPass
  $qw_p \leftarrow \mathsf{QuantizeWeights}(w_p)$ // Quantize p's weights
  $qw \leftarrow \mathsf{AllGather}(qw_i \text{ for all } i)$ // Collect quantized weights
  $o_p \leftarrow \mathsf{Layer}(qw, x_p)$ // Compute output for p
  $\mathsf{free}(qw)$ // Discard aggregated layer weights
end function
function ExecuteBackwardPass
  $qw_p \leftarrow \mathsf{QuantizeWeights}(w_p)$ // Quantize p's weights
  $qw \leftarrow \mathsf{AllGather}(qw_i \text{ for all } i)$ // Collect quantized weights
  $g_p \leftarrow \mathsf{Gradient}(qw, o_p)$ // Compute gradient for p
  $\mathsf{free}(qw)$ // Discard aggregated layer weights
  $qg_p \leftarrow \mathsf{QuantizeGradients}(g_p)$ // Quantize p's gradient
  $qg_p \leftarrow \mathsf{ReduceScatter}(qg_i \text{ for each } i)$ // Distribute gradients
  $w_p \leftarrow \mathsf{WeightUpdate}(qg_p, w_p)$ // Update p's weights
  $\mathsf{free}(qg)$ // Discard aggregated gradients
end function

A. Training details.

For training of GPT-2 models we were using MosaicML (Mos, 2022) examples. The global batch size for 125M and 350M models was 256, for 1.3B - 512, resulting in 4 gradient accumulations at each iteration. For all models AdamW optimizer was used, the optimizer parameters are presented in the Table 4. 125M model was trained in 4800 steps, 350M model in 13400 steps, 1.3B model in 14000 steps.

Table 4. AdamW optimizer parameters.

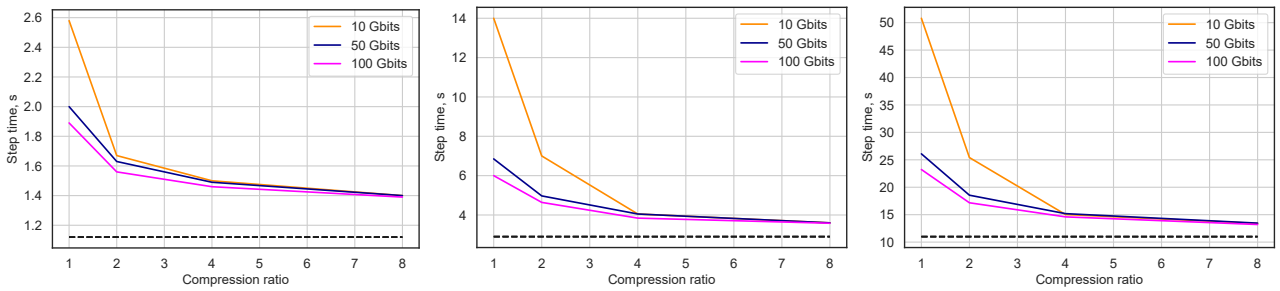
	125M	350M	1.3B
learning rate	6e-4	3e-4	2e-4
betas	0.9, 0.95	0.9, 0.95	0.9, 0.95
epsilon	1e-8	1e-8	1e-8

B. Network overhead experiments

In order to evaluate the effect on communications in FSDP training we conducted the synthetic experiment which reduces the bandwidth costs in each iteration. Specifically, given the buffer of size N which is about to be communicated, and compression ratio γ we only transmit the first N/γ elements. The results for our setup (4 x 8V100-32G GPUs) at different internode bandwidths is shown in the Figures.4, communication weights and gradients are reduced to the same compression ratio. We see that the most effect of compression is reached as expected for the largest 1.3B model and at lowest bandwidth. However, one can get around 80% speedup at high bandwidth when up to 8x compression ratio is applied. Also, we notice that $8\times$ compression almost reaches the ideal scaling for large model and has a evident overhead over the no-communication training in case of the small model. It infers that the large models have a bottleneck in bandwidth component of the communication and the small model has a dominating latency part.

To see the variance of the compression effects on weights and gradients we conducted the similar experiment for different combinations of compression ratio pairs (see 5). We observe that weight compression gives more performance profits than gradient compression. This can be naturally explained by the fact that weights are communicated more frequently than gradients in FSDP (in this particular experiment weights are communicated 5 times per one gradient exchange) and the amount of transmissions per communication is similar.

The difference between the synthetic experiment and QSDP performance numbers with the same compression ratios can be justified by the performance inefficiency of NCCL point-to-point communication primitives on which QSDP compressed communication is based on - the compression overhead in our experiments was verified to be negligible (less than 1% per iteration).



(a) 125M (b) 350M (c) 1.3B

Figure 4. Compression vs average step time for different models at different inter-node bandwidths with fake compression (weights and gradients have the same compression ratio). Lower is better. The dashed line represents ideal scaling - training without communication.

Table 5. Training step timings (in seconds) for 1.3B model at 100 Gbps bandwidth with various combinations of weights and gradient compression ratio.

Weights ratio \ Gradients ratios	1	2	4	8
1	23.23	21.36	20.62	20.2
2	19.27	17.17	16.26	15.95
4	17.50	15.35	14.6	14.08
8	16.62	14.52	13.66	13.21

C. Learned quantization

We implemented stochastic gradient descent optimization of quantization levels in PyTorch. We use learning rate 0.01, batch size 1024. We run the learning for each layer larger than 1e5 parameters, for other layers uniform quantization was used. We evaluate the quality of quantization levels by comparing L2 of compression error introduced by quantizing a buffer using the levels. We conducted the such evaluation for weights quantized to 5 bits and gradients quantized to 4 bits during the training of GPT 125M model. The results for one of the attention layers and LM head layer are shown in the Figures 5 and 6. The dashed vertical lines show the moment of running learning quantization levels algorithm. We see that compression error of learned quantization levels is constantly lower for the learned algorithm, and the lower bits-width (for gradients we use 4 bits quantizaton) the larger the gap between the considered methods. Also, we see that the compression error of the learned quantization only increases in sync with uniform quantization over time. It means that learning algorithm can be run only once, at the start of the training.

Also, we measured overhead of running learning algorithm for GPT 125M with weights quantized to 5 bits, gradients to 4 bits. The overhead of learning algorithm amounts to around 9 minutes, whereas the full training takes lasts 5 hours.

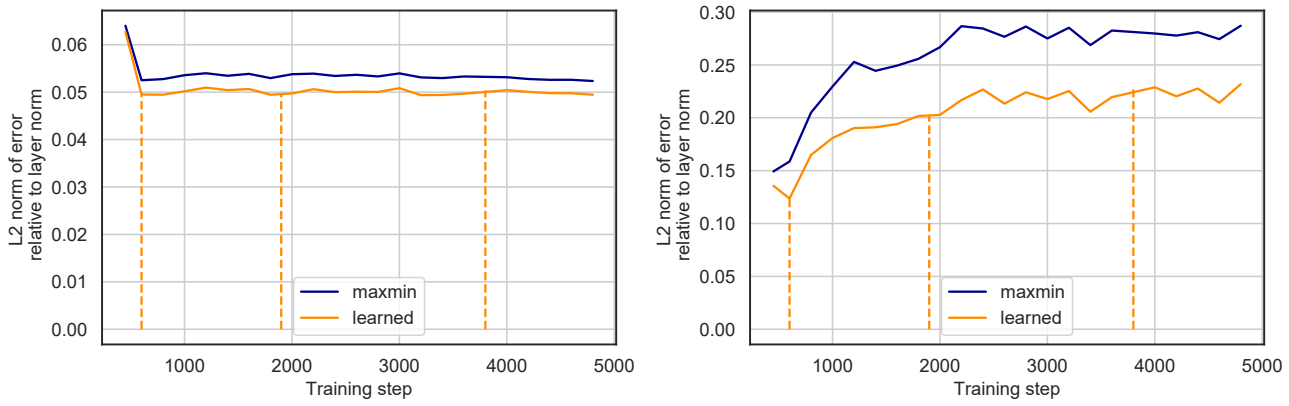
The extra experiments results with low bit-width quantization are shown in the Table. 6. The number doesn't show full perplexity recovery but they represent the improvements achieved by learned levels algorithm. We can see that with learned quantization levels one can reduce up to 3 units of perplexity.

Table 6. Final perplexities of low-bits quantization of 125m GPT-2 model using the learned quantization levels.

	baseline	w3g32	w2g32	w8g3	w8g2
Uniform	35.81	45.53	57.92	39.91	44.79
Learned		42.31	56.54	37.72	44.65

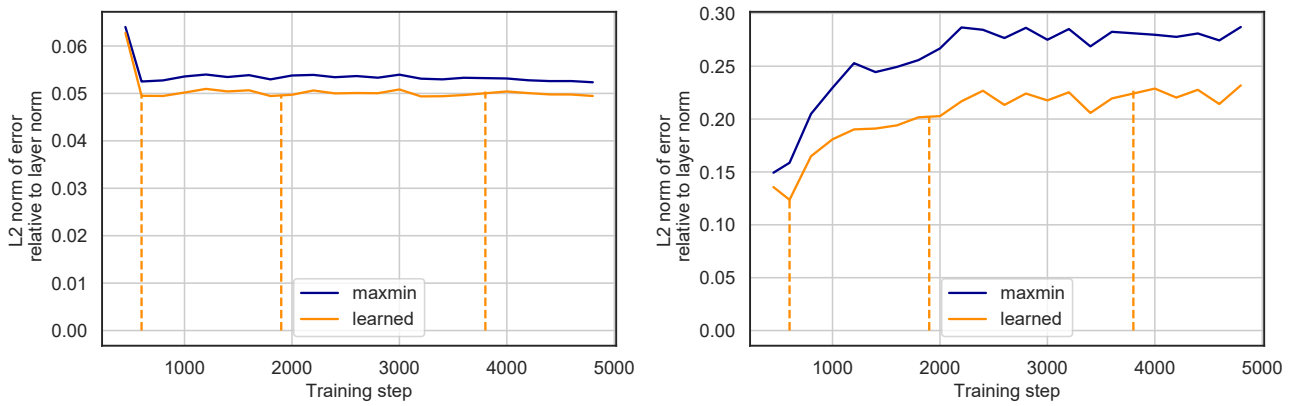
D. Convergence experiments.

Quantization bucket size. To highlight the importance of bucket size choice on practice we trained the 125M model quantizing weights and gradients to 8 bits with larger than default bucket size. The results are shown in Figure 7. We can see



(a) Weights (b) Gradients

Figure 5. Compression error (L2 norm of the error relative to L2 norm of the input) comparison with learned quantization levels for attention layer of 125M model, W5G4 quantization.



(a) Weights (b) Gradients

Figure 6. Compression error (L2 norm of the error relative to L2 norm of the input) comparison with learned quantization levels for LM-Head layerof 125M model, W5G4 quantization.

that training with quantization with bucket size 16284 has a remarkable gap in the convergence curve and does not recover full accuracy.

Quantization bu Random shift. We have conducted an empirical study of the Theorem 2 to show the impact of the random shift in quantization on the training convergence. We setup a linear regression problem on the YearPredictionMSD dataset from UCI Machine learning repository data (Dua & Graff, 2017). We trained the model quantizing weights and gradients of the linear model to 8 bits with bucket size 16 using the original QSGD and QSGD with random shift. We can see in the Figure. 8 that QSGD with random shift has better convergence than QSGD in the setup with convex problem.

E. Convergence Proofs

In this section we provide the convergence analysis for our algorithms.

E.1. Overview

We use the notation and assumptions defined in Section 4.1. As all of our analyses revolve around bounding the progress made in a single iteration, to simplify notation we will generally use $\boldsymbol{x}$ to denote the current iterate, and $\boldsymbol{x}'$ to denote the

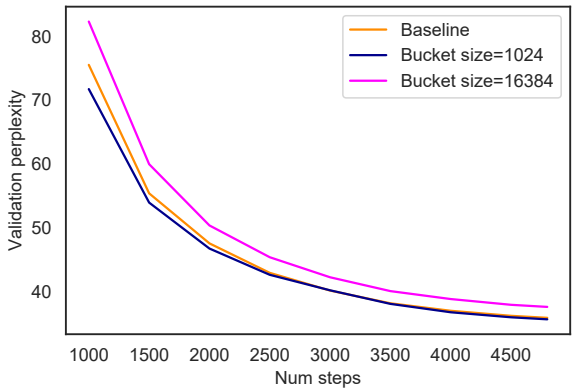


Figure 7. Validation perplexity vs number of steps in quantized training with default and larger bucket size.

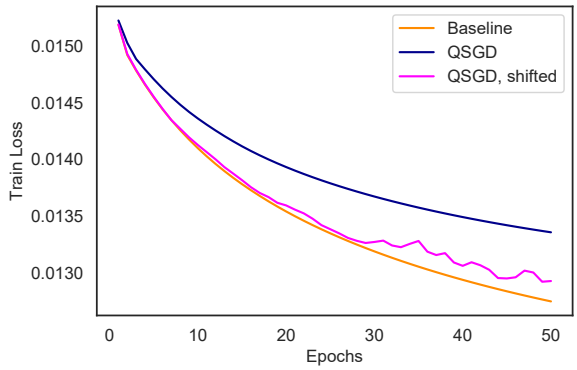


Figure 8. Training loss vs number of steps for convex problem comparing QSGD and QSGD with random shift.

iterate obtained after the generic update:

$$\boldsymbol{x}' = \mathbf{Q}_\delta^{\mathbf{w}}\left(\boldsymbol{x} - \frac{\eta}{\beta}\boldsymbol{g}\left(\boldsymbol{x}\right)\right),$$

where β is the smoothness parameter of f. In Section E.2 we will first prove convergence for the deterministic method, where we have direct access to the gradients of f. The analysis precisely follows the steps we described in Section 4.1. Then, we extend this analysis to the case of stochastic gradients, and provide the full proof for Theorem 2. Finally, in Section E.3 we show that given an appropriate gradient quantization method with bounded variance, we can use it on top of our iteration to further reduce the required amount of communication, and thus prove Corollary 3.

Before proceeding, we first formally analyze the quantization method we defined in Section 4.1, and show some additional properties that will be useful later.

Lemma 5. *Let $\boldsymbol{v} \in \mathbb{R}^n$, and let $\delta > 0$. Then,*

$$\mathbb{E}\left[\mathbf{Q}_\delta^{\mathbf{w}}\left(\boldsymbol{v}\right)\right] = \boldsymbol{v}\,,$$

$$\mathbb{E}\left[\left\|\mathbf{Q}_\delta^{\mathbf{w}}\left(\boldsymbol{v}\right) - \boldsymbol{v}\right\|_2^2\right] = \delta^2 \cdot \sum_{i=1}^{n}\left\{\frac{v_i}{\delta}\right\}\left(1 - \left\{\frac{v_i}{\delta}\right\}\right),$$

$$\mathbb{E}\left[\left\|\mathbf{Q}_{r,\delta}^{\mathbf{w}}\left(\boldsymbol{v}\right) - r\mathbf{1}\right\|_0\right] \leq \left\|\boldsymbol{v}\right\|_1/\delta\,.$$

Since the proofs are technical, we defer them to Section E.4.1. The most important feature of this quantization scheme is captured by Lemma 4, which is crucial for our convergence proof. We first restate it, and prove it formally in Section E.4.2.

Lemma 4. *Let $\delta_\star > \delta > 0$, such that $\delta_\star/\delta \in \mathbb{Z}$. Let $\boldsymbol{x} \in \mathbb{R}^n$, and for all $r \in [-\delta_\star/2, \delta_\star/2)$, let an arbitrary $\boldsymbol{x}^\star_{r,\delta_\star} \in \delta_\star\mathbb{Z}^n + r\mathbf{1}$. Then*

$$\mathbb{E}\left[\left\|\mathbf{Q}_\delta^{\mathbf{w}}\left(\boldsymbol{x}\right) - \boldsymbol{x}\right\|_2^2\right] \leq \frac{\delta}{\delta_\star}\mathbb{E}_r\left[\left\|\boldsymbol{x}^\star_{r,\delta_\star} - \boldsymbol{x}\right\|_2^2\right].$$

The proof crucially relies on the fact that $\delta/\delta_\star \in \mathbb{Z}$, and is rooted in the following inequality:

Lemma 6. *Let $y \in \mathbb{R}$ and $k \in \mathbb{Z}$. Then*

$$\left(1 - \{y\}\right)\{y\} \leq k\left(1 - \left\{\frac{y}{k}\right\}\right)\left\{\frac{y}{k}\right\}.$$

Proof. It suffices to consider $y \in [0, k]$, as both $\{y\}(1-\{y\})$ and $\{y/k\}(1-\{y/k\})$ are periodic within this interval. The function $\{y/k\}(1-\{y/k\})$ is a quadratic which is monotonically increasing over $[0, k/2]$ and symmetric around $k/2$. As $(1-\{y\})\{y\}$ is periodic on intervals of length 1 it suffices to show that $(1-\{y\})\{y\} \leq k\left(1-\left\{\frac{y}{k}\right\}\right)\left\{\frac{y}{k}\right\}$ on the interval $[0, 1]$. At this point we can drop the fractional part, and simply need to compare two quadratics over $[0, 1]$. Equivalently we need to show that $k(1-y/k)y/k \geq y(1-y)$ over $[0, 1]$, which after simplifying both sides is equivalent to $y^2(1-1/k) \geq 0$ over this interval, which is true. □

Finally, we provide some basic optimization inequalities, which will allow us to prove our theorems.

Optimization Basics. The first Lemma bounds the change in function value using smoothness, while the latter upper bounds the ℓ_2 distance to optimality using the error in function value. We provide the proofs in Sections E.4.3 and E.4.4.

Lemma 7. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a β-smooth function. Then for any $\boldsymbol{\Delta} \in \mathbb{R}^n$,*

$$f(\boldsymbol{x} + \boldsymbol{\Delta}) \leq f(\boldsymbol{x}) + (1-\eta)\langle \nabla f(\boldsymbol{x}), \boldsymbol{\Delta}\rangle - \frac{\eta^2}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2 + \frac{\beta}{2}\left\|\frac{\eta}{\beta}\nabla f(\boldsymbol{x}) + \boldsymbol{\Delta}\right\|_2^2 .$$

Lemma 8. *If $f : \mathbb{R}^n \to \mathbb{R}$ satisfies the α-PL condition, then for all $\boldsymbol{x} \in \mathbb{R}^n$,*

$$f(\boldsymbol{x}) - f^\star \geq \frac{\alpha}{2}\|\boldsymbol{x} - \boldsymbol{x}^\star\|_2^2 ,$$

where $\boldsymbol{x}^\star \in \arg\min_{\boldsymbol{x}} f(\boldsymbol{x})$.

We are now ready to prove the main theorems in this paper.

E.2. SGD with weight quantization

We first prove the stepping lemma for our quantized gradient method, in the case where full gradients are available. The steps are essentially the same we described in Section 4.1.

Lemma 9. *Let $f(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ be a β-smooth and α-PL function. For each $r \in [-\delta_\star/2, \delta_\star/2]$, let $\boldsymbol{x}^\star_{r,\delta_\star}$ be any minimizer of f over $\delta_\star\mathbb{Z} + r$. Let $\delta = \frac{\delta_\star}{\lceil 4(\beta/\alpha)^2 \rceil}$. Then letting $\boldsymbol{x'} = \mathbf{Q}^{\mathbf{w}}_\delta\left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right)$, one has that in expectation over the random bits used by the quantization operator:*

$$\mathbb{E}f(\boldsymbol{x'}) - \mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right) \leq \left(1 - \frac{\alpha}{2\beta}\right)\left(\mathbb{E}f(\boldsymbol{x}) - \mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)\right) .$$

Proof. Letting $\boldsymbol{\Delta} = \boldsymbol{x'} - \boldsymbol{x}$, we write:

$$\left\|\frac{1}{\beta}\nabla f(\boldsymbol{x}) + \boldsymbol{\Delta}\right\|_2^2 = \left\|(\boldsymbol{x} + \boldsymbol{\Delta}) - \left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right)\right\|_2^2 = \left\|\mathbf{Q}^{\mathbf{w}}_\delta\left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right) - \left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right)\right\|_2^2 .$$

Also using Lemma 4, we have that for any $\boldsymbol{x}^\star \in \arg\min_{\boldsymbol{x}} f(\boldsymbol{x})$,

$$\begin{aligned}\mathbb{E}\left\|\mathbf{Q}^{\mathbf{w}}_\delta\left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right) - \left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right)\right\|_2^2 &\leq \frac{\delta}{\delta_\star}\mathbb{E}_r\left[\left\|\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x}) - \boldsymbol{x}^\star_{r,\delta_\star}\right\|_2^2\right] \\ &\leq 2\frac{\delta}{\delta_\star}\left(\mathbb{E}_r\left[\left\|\boldsymbol{x}^\star_{r,\delta_\star} - \boldsymbol{x}^\star\right\|_2^2\right] + \left\|\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x}) - \boldsymbol{x}^\star\right\|_2^2\right) .\end{aligned}$$

Using the α-PL condition we upper bound distance from $\boldsymbol{x}^\star$ with function value i.e.

$$\begin{aligned}\left\|\boldsymbol{x}^\star_{r,\delta_\star} - \boldsymbol{x}^\star\right\|_2^2 &\leq \frac{2}{\alpha}\cdot\left(f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right) - f(\boldsymbol{x}^\star)\right) , \\ \left\|\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x}) - \boldsymbol{x}^\star\right\|_2^2 &\leq \frac{2}{\alpha}\left(f\left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right) - f(\boldsymbol{x}^\star)\right) .\end{aligned}$$

Combining these with Lemma 7 for $\eta = 1$ we conclude that

$$\begin{aligned}\mathbb{E}f(\boldsymbol{x'}) \leq\ & f(\boldsymbol{x}) - \frac{1}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2 \\ &+ 2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\left(f\left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right) - f(\boldsymbol{x}^\star)\right) + 2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\cdot\mathbb{E}_r\left[f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right) - f(\boldsymbol{x}^\star)\right] .\end{aligned}$$

Again, using the PL condition we lower bound $\frac{1}{2}\left\|\nabla f\left(\boldsymbol{x}\right)\right\|_2^2 \geq \alpha\left(f\left(\boldsymbol{x}\right)-f\left(\boldsymbol{x}^\star\right)\right)$, which gives

$$\begin{aligned}
&\mathbb{E}\left[f\left(\boldsymbol{x'}\right)-f\left(\boldsymbol{x}^\star\right)\right]\\
&\leq\left(1-\frac{\alpha}{\beta}\right)\left(f\left(\boldsymbol{x}\right)-f\left(\boldsymbol{x}^\star\right)\right)+2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\left(f\left(\boldsymbol{x}-\frac{1}{\beta}\nabla f\left(\boldsymbol{x}\right)\right)-f\left(\boldsymbol{x}^\star\right)\right)+2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\cdot\mathbb{E}_r\left[f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)-f\left(\boldsymbol{x}^\star\right)\right]\\
&\leq\left(1-\frac{\alpha}{\beta}+2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\right)\left(f\left(\boldsymbol{x}\right)-f\left(\boldsymbol{x}^\star\right)\right)+2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\cdot\mathbb{E}_r\left[f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)-f\left(\boldsymbol{x}^\star\right)\right]\,.
\end{aligned}$$

Equivalently we obtain

$$\begin{aligned}
\mathbb{E}f\left(\boldsymbol{x'}\right)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)&\leq\left(1-\frac{\alpha}{\beta}+2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)\right)+2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\cdot\mathbb{E}\left[f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)-f\left(\boldsymbol{x}^\star\right)\right]\\
&\quad+f\left(\boldsymbol{x}^\star\right)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)\\
&=\left(1-\frac{\alpha}{\beta}+2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)\right)+\left(2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}-\frac{\alpha}{\beta}+2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\right)\cdot\mathbb{E}\left[f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)-f\left(\boldsymbol{x}^\star\right)\right]\\
&=\left(1-\frac{\alpha}{\beta}+2\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)\right)+\left(4\frac{\delta}{\delta_\star}\cdot\frac{\beta}{\alpha}-\frac{\alpha}{\beta}\right)\cdot\mathbb{E}\left[f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)-f\left(\boldsymbol{x}^\star\right)\right]\,.
\end{aligned}$$

Since we set $\delta/\delta_\star=\frac{1}{\lceil 4(\beta/\alpha)^2\rceil}$, the second term is non-positive. Therefore in this case we have

$$\mathbb{E}f\left(\boldsymbol{x'}\right)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)\leq\left(1-\frac{\alpha}{2\beta}\right)\left(\mathbb{E}f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)\right)\,,$$

which concludes the proof. □

We now generalize the proof of Lemma 9 to the case where only stochastic gradients are available. The proof is essentially the same, the main difference being that we isolate terms involving the difference between the stochastic and the true gradient, which we bound separately using our variance bound.

Lemma 10. *Let $f:\mathbb{R}^n\to\mathbb{R}$ be a β-smooth and α-PL function. For each $r\in[-\delta_\star/2,\delta_\star/2]$, let $\boldsymbol{x}^\star_{r,\delta_\star}$ be any minimizer of f over $\delta_\star\mathbb{Z}+r$. Let $\delta=\frac{\eta}{\lceil 16(\beta/\alpha)^2\rceil}\cdot\delta_\star$. Let $\boldsymbol{x'}=\mathbf{Q}^{\mathbf{w}}_\delta\left(\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}\left(\boldsymbol{x}\right)\right)$, where $\boldsymbol{g}\left(\boldsymbol{x}\right)$ is an unbiased estimator for $\nabla f\left(\boldsymbol{x}\right)$ i.e. $\mathbb{E}\left[\boldsymbol{g}\left(\boldsymbol{x}\right)|\boldsymbol{x}\right]=\nabla f\left(\boldsymbol{x}\right)$, and $0<\eta\leq 1$ is a step size parameter. Furthermore assume that the variance of $\boldsymbol{g}\left(\boldsymbol{x}\right)$ is bounded $\mathbb{E}\left\|\boldsymbol{g}\left(\boldsymbol{x}\right)-\nabla f\left(\boldsymbol{x}\right)\right\|_2^2\leq\sigma^2$, for a real parameter $\sigma>0$. Then, for $r\sim$ Unif$\left([-\delta_\star/2,\delta_\star/2)\right)$, in expectation over the gradient stochasticity:*

$$\mathbb{E}\left[f\left(\boldsymbol{x'}\right)|\boldsymbol{x}\right]-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)\leq\left(1-\frac{3}{4}\eta\frac{\alpha}{\beta}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)\right)+\frac{5}{4}\frac{\eta^2}{\beta}\sigma^2\,.$$

Proof. We follow the analysis from Lemma 9, while moving the stochastic gradients into expressions that involve the stochastic variance. Letting $\delta=\boldsymbol{x'}-\boldsymbol{x}$, we write:

$$\begin{aligned}
\left\|\frac{\eta}{\beta}\nabla f\left(\boldsymbol{x}\right)+\boldsymbol{\Delta}\right\|_2^2&=\left\|\left(\boldsymbol{x}+\boldsymbol{\Delta}\right)-\left(\boldsymbol{x}-\frac{\eta}{\beta}\nabla f\left(\boldsymbol{x}\right)\right)\right\|_2^2=\left\|\mathbf{Q}^{\mathbf{w}}_{r,\delta}\left(\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}\left(\boldsymbol{x}\right)\right)-\left(\boldsymbol{x}-\frac{\eta}{\beta}\nabla f\left(\boldsymbol{x}\right)\right)\right\|_2^2\\
&\leq 2\left\|\mathbf{Q}^{\mathbf{w}}_{r,\delta}\left(\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}\left(\boldsymbol{x}\right)\right)-\left(\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}\left(\boldsymbol{x}\right)\right)\right\|_2^2+2\left\|\frac{\eta}{\beta}\left(\boldsymbol{g}\left(\boldsymbol{x}\right)-\nabla f\left(\boldsymbol{x}\right)\right)\right\|_2^2\,,
\end{aligned}$$

where we used the inequality $\|\boldsymbol{a}+\boldsymbol{b}\|_2^2 \leq 2\|\boldsymbol{a}\|_2^2 + 2\|\boldsymbol{b}\|_2^2$. Also using Lemma 4, we have that for any $\boldsymbol{x}^\star \in \arg\min_{\boldsymbol{x}} f(\boldsymbol{x})$,

$$\begin{aligned}
&\mathbb{E}\left\|\mathbf{Q}_{r,\delta}^{\mathbf{w}}\left(\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}(\boldsymbol{x})\right)-\left(\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}(\boldsymbol{x})\right)\right\|_2^2 \\
&\leq \frac{\delta}{\delta_\star}\mathbb{E}\left[\left\|\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}(\boldsymbol{x})-\boldsymbol{x}_{r,\delta_\star}^\star\right\|_2^2\right] \\
&\leq 2\frac{\delta}{\delta_\star}\left(\mathbb{E}\left[\left\|\boldsymbol{x}_{r,\delta_\star}^\star-\boldsymbol{x}^\star\right\|_2^2\right]+\left\|\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}(\boldsymbol{x})-\boldsymbol{x}^\star\right\|_2^2\right) \\
&\leq 2\frac{\delta}{\delta_\star}\left(\mathbb{E}\left[\left\|\boldsymbol{x}_{r,\delta_\star}^\star-\boldsymbol{x}^\star\right\|_2^2\right]+2\left\|\boldsymbol{x}-\frac{\eta}{\beta}\nabla f(\boldsymbol{x})-\boldsymbol{x}^\star\right\|_2^2+2\left\|\frac{\eta}{\beta}\left(\boldsymbol{g}(\boldsymbol{x})-\nabla f(\boldsymbol{x})\right)\right\|_2^2\right).
\end{aligned}$$

Using the α-PL condition we upper bound distance from $\boldsymbol{x}^\star$ with function value i.e.

$$\begin{aligned}
\left\|\boldsymbol{x}_{r,\delta_\star}^\star-\boldsymbol{x}^\star\right\|_2^2 &\leq \frac{2}{\alpha}\cdot\left(f\left(\boldsymbol{x}_{r,\delta_\star}^\star\right)-f(\boldsymbol{x}^\star)\right), \\
\left\|\boldsymbol{x}-\frac{\eta}{\beta}\nabla f(\boldsymbol{x})-\boldsymbol{x}^\star\right\|_2^2 &\leq \frac{2}{\alpha}\left(f\left(\boldsymbol{x}-\frac{\eta}{\beta}\nabla f(\boldsymbol{x})\right)-f(\boldsymbol{x}^\star)\right).
\end{aligned}$$

Combining these with Lemma 7 we conclude that in expectation over the random shift:

$$\begin{aligned}
f(\boldsymbol{x}') &\leq f(\boldsymbol{x})+(1-\eta)\langle\nabla f(\boldsymbol{x}),\boldsymbol{\Delta}\rangle-\frac{\eta^2}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2 \\
&\quad+\frac{\beta}{2}\cdot\left(2\left\|\mathbf{Q}_{r,\delta}^{\mathbf{w}}\left(\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}(\boldsymbol{x})\right)-\left(\boldsymbol{x}-\frac{\eta}{\beta}\nabla f_i(\boldsymbol{x})\right)\right\|_2^2+2\left\|\frac{\eta}{\beta}\left(\boldsymbol{g}(\boldsymbol{x})-\nabla f(\boldsymbol{x})\right)\right\|_2^2\right) \\
&\leq f(\boldsymbol{x})+(1-\eta)\langle\nabla f(\boldsymbol{x}),\boldsymbol{\Delta}\rangle-\frac{\eta^2}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2 \\
&\quad+2\beta\frac{\delta}{\delta_\star}\cdot\left(\left\|\boldsymbol{x}_{r,\delta_\star}^\star-\boldsymbol{x}^\star\right\|_2^2+2\left\|x-\frac{\eta}{\beta}\nabla f(x)-\boldsymbol{x}^\star\right\|_2^2+2\left\|\frac{\eta}{\beta}\left(\boldsymbol{g}(\boldsymbol{x})-\nabla f(\boldsymbol{x})\right)\right\|_2^2\right) \\
&\quad+\frac{\eta^2}{\beta}\|\boldsymbol{g}(\boldsymbol{x})-\nabla f(\boldsymbol{x})\|_2^2 \\
&\leq f(\boldsymbol{x})+(1-\eta)\langle\nabla f(\boldsymbol{x}),\boldsymbol{\Delta}\rangle-\frac{\eta^2}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2 \\
&\quad+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\left(f\left(\boldsymbol{x}-\frac{\eta}{\beta}\nabla f(\boldsymbol{x})\right)-f(\boldsymbol{x}^\star)\right)+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\left(f\left(\boldsymbol{x}_{r,\delta_\star}^\star\right)-f(\boldsymbol{x}^\star)\right) \\
&\quad+\frac{\eta^2}{\beta}\left(1+4\frac{\delta}{\delta_\star}\right)\|\boldsymbol{g}(\boldsymbol{x})-\nabla f(\boldsymbol{x})\|_2^2.
\end{aligned}$$

Again, using the PL condition we lower bound $\frac{1}{2}\|\nabla f(\boldsymbol{x})\|_2^2 \geq \alpha\left(f(\boldsymbol{x})-f(\boldsymbol{x}^\star)\right)$, which gives that in expectation over the random shift:

$$\begin{aligned}
f(\boldsymbol{x}')-f(\boldsymbol{x}^\star) &\leq \left(1-\eta^2\frac{\alpha}{\beta}\right)\left(f(\boldsymbol{x})-f(\boldsymbol{x}^\star)\right)+(1-\eta)\langle\nabla f(\boldsymbol{x}),\boldsymbol{\Delta}\rangle \\
&\quad+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\left(f\left(\boldsymbol{x}-\frac{\eta}{\beta}\nabla f(\boldsymbol{x})\right)-f(\boldsymbol{x}^\star)\right)+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\left(f\left(\boldsymbol{x}_{r,\delta_\star}^\star\right)-f(\boldsymbol{x}^\star)\right) \\
&\quad+\frac{\eta^2}{\beta}\left(1+4\frac{\delta}{\delta_\star}\right)\|\boldsymbol{g}(\boldsymbol{x})-\nabla f(\boldsymbol{x})\|_2^2 \\
&\leq \left(1-\eta^2\frac{\alpha}{\beta}+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\right)\left(f(\boldsymbol{x})-f(\boldsymbol{x}^\star)\right)+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\left(f\left(\boldsymbol{x}_{r,\delta_\star}^\star\right)-f(\boldsymbol{x}^\star)\right) \\
&\quad+(1-\eta)\langle\nabla f(\boldsymbol{x}),\boldsymbol{\Delta}\rangle+\frac{\eta^2}{\beta}\left(1+4\frac{\delta}{\delta_\star}\right)\|\boldsymbol{g}(\boldsymbol{x})-\nabla f(\boldsymbol{x})\|_2^2.
\end{aligned}$$

and equivalently, in expectation over the random shift:

$$\mathbb{E}\left[f\left(\boldsymbol{x}'\right)-f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)\right] \leq \left(1-\eta^2\frac{\alpha}{\beta}+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)\right)+\left(8\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}-\eta^2\frac{\alpha}{\beta}\right)\left(\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)-f\left(\boldsymbol{x}^{\star}\right)\right)$$
$$+\left(1-\eta\right)\left\langle\nabla f\left(\boldsymbol{x}\right),\mathbb{E}\left[\boldsymbol{\Delta}\right]\right\rangle+\frac{\eta^2}{\beta}\left(1+4\frac{\delta}{\delta_\star}\right)\left\|\boldsymbol{g}\left(\boldsymbol{x}\right)-\nabla f\left(\boldsymbol{x}\right)\right\|_2^2 .$$

At this point we use Lemma 5 to write

$$\begin{aligned}\mathbb{E}\left[\boldsymbol{\Delta}\right] &= \mathbb{E}\left[\mathbf{Q}^{\mathbf{w}}_{r,\delta}\left(\boldsymbol{x}-\frac{\eta}{\beta}\boldsymbol{g}\left(\boldsymbol{x}\right)\right)-x\right]\\ &= -\frac{\eta}{\beta}\boldsymbol{g}\left(\boldsymbol{x}\right) ,\end{aligned}$$

and thus

$$\left(1-\eta\right)\left\langle\nabla f\left(\boldsymbol{x}\right),\mathbb{E}\left[\boldsymbol{\Delta}\right]\right\rangle=-\frac{\eta}{\beta}\left(1-\eta\right)\left\langle\nabla f\left(\boldsymbol{x}\right),\boldsymbol{g}\left(\boldsymbol{x}\right)\right\rangle .$$

Therefore, after taking expectation over both the random shift and gradient stochasticity we obtain:

$$\begin{aligned}&\mathbb{E}\left[f\left(\boldsymbol{x}'\right)-f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)|\boldsymbol{x}\right]\\ &\leq\left(1-\eta^2\frac{\alpha}{\beta}+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)\right)+\left(8\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}-\eta^2\frac{\alpha}{\beta}\right)\left(\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)-f\left(\boldsymbol{x}^{\star}\right)\right)\\ &-\frac{\eta}{\beta}\left(1-\eta\right)\left\|\nabla f\left(\boldsymbol{x}\right)\right\|_2^2+\frac{\eta^2}{\beta}\left(1+4\frac{\delta}{\delta_\star}\right)\sigma^2\\ &\leq\left(1-\eta^2\frac{\alpha}{\beta}+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)\right)+\left(8\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}-\eta^2\frac{\alpha}{\beta}\right)\left(\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)-f\left(\boldsymbol{x}^{\star}\right)\right)\\ &-2\eta\left(1-\eta\right)\frac{\alpha}{\beta}\left(f\left(\boldsymbol{x}\right)-f\left(\boldsymbol{x}^{\star}\right)\right)+\frac{\eta^2}{\beta}\left(1+4\frac{\delta}{\delta_\star}\right)\sigma^2\\ &=\left(1-\eta^2\frac{\alpha}{\beta}-2\eta\left(1-\eta\right)\frac{\alpha}{\beta}+4\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)\right)+\left(8\frac{\delta}{\delta_\star}\frac{\beta}{\alpha}-\eta^2\frac{\alpha}{\beta}-2\eta\left(1-\eta\right)\frac{\alpha}{\beta}\right)\left(\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)-f\left(\boldsymbol{x}^{\star}\right)\right)\\ &+\frac{\eta^2}{\beta}\left(1+4\frac{\delta}{\delta_\star}\right)\sigma^2 .\end{aligned}$$

Since we set $\delta/\delta_\star=\frac{\eta}{\lceil 16(\beta/\alpha)^2\rceil}$, the second term is non-positive. Therefore in this case we have

$$\begin{aligned}\mathbb{E}\left[f\left(\boldsymbol{x}'\right)-\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)|\boldsymbol{x}\right] &\leq\left(1-\eta^2\frac{\alpha}{\beta}-2\eta\left(1-\eta\right)\frac{\alpha}{\beta}+\frac{\eta}{4}\frac{\alpha}{\beta}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)\right)+\frac{\eta^2}{\beta}\left(1+\frac{\eta}{4}\left(\frac{\alpha}{\beta}\right)^2\right)\sigma^2\\ &\leq\left(1-\frac{7}{4}\eta\frac{\alpha}{\beta}+\eta^2\frac{\alpha}{\beta}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)\right)+\frac{\eta^2}{\beta}\left(1+\frac{\eta}{4}\left(\frac{\alpha}{\beta}\right)^2\right)\sigma^2\\ &\leq\left(1-\frac{3}{4}\eta\frac{\alpha}{\beta}\right)\left(f\left(\boldsymbol{x}\right)-\mathbb{E}f\left(\boldsymbol{x}^{\star}_{r,\delta_\star}\right)\right)+\frac{5}{4}\frac{\eta^2}{\beta}\sigma^2 ,\end{aligned}$$

as long as $\eta\leq 1$. This concludes the proof. □

Using Lemma 10 the proof of Theorem 2 follows very easily.

Theorem 2. *Let $\alpha,\beta,\delta_\star,\varepsilon>0$ and $\sigma\geq 0$ be real parameters, and let $\eta=\min\left\{\frac{3}{10}\frac{\varepsilon\alpha}{\sigma^2},1\right\}$. Let $f:\mathbb{R}^n\to\mathbb{R}$ be a β-smooth and α-PL function, with access to a stochastic gradient $\boldsymbol{g}\left(\boldsymbol{x}\right)$, i.e. $\mathbb{E}\left[\boldsymbol{g}\left(\boldsymbol{x}\right)|\boldsymbol{x}\right]=\nabla f\left(\boldsymbol{x}\right)$ with bounded variance $\mathbb{E}\left\|\boldsymbol{g}\left(\boldsymbol{x}\right)-\nabla f\left(\boldsymbol{x}\right)\right\|_2^2\leq\sigma^2$. For each $r\in[-\delta_\star/2,\delta_\star/2)$, let $\boldsymbol{x}^{\star}_{r,\delta_\star}$ be any minimizer of f over $\delta_\star\mathbb{Z}^n+r\mathbf{1}$. Let $\delta=\frac{\eta}{\lceil 16(\beta/\alpha)^2\rceil}\cdot\delta_\star$. Consider the iteration:*

$$\boldsymbol{x}_{t+1}=\mathbf{Q}^{\mathbf{w}}_{\delta}\left(\boldsymbol{x}_t-\frac{\eta}{\beta}\boldsymbol{g}\left(\boldsymbol{x}_t\right)\right) .$$

In $T=\frac{10}{\eta}\cdot\frac{\beta}{\alpha}\ln\frac{f(\boldsymbol{x}_0)-\mathbb{E}f(\boldsymbol{x}^{\star}_{r,\delta_\star})}{\varepsilon}$ iterations we obtain a point $\boldsymbol{x}_T$ satisfying $\mathbb{E}f\left(\boldsymbol{x}_T\right)-\mathbb{E}f(\boldsymbol{x}^{\star}_{r,\delta_\star})\leq\varepsilon$.

Proof. Plugging in Lemma 10 and applying it for $T = \frac{10}{\eta}\frac{\beta}{\alpha}\ln\frac{f(\boldsymbol{x}_0)-\mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right)}{\varepsilon}$ we obtain:

$$\mathbb{E}f\left(\boldsymbol{x}_T\right) - \mathbb{E}f\left(\boldsymbol{x}^\star_{r,\delta_\star}\right) \leq \frac{\varepsilon}{2} + \frac{5}{4}\frac{\eta^2}{\beta}\sigma^2 \cdot \sum_{k=0}^{T-1}\left(1 - \frac{3}{4}\eta\frac{\alpha}{\beta}\right)^k \leq \frac{\varepsilon}{2} + \frac{5}{4}\frac{\eta^2}{\beta}\sigma^2 \cdot \frac{4}{3}\frac{1}{\eta}\frac{\beta}{\alpha} = \frac{\varepsilon}{2} + \frac{5}{3}\frac{\eta}{\alpha}\sigma^2\,.$$

Since we set $\eta = \min\left\{\frac{3}{10}\frac{\varepsilon\alpha}{\sigma^2}, 1\right\}$, the entire quantity is at most ε, which concludes the proof. □

E.3. Reducing Communication by Quantizing Gradients

The approach described in Section E.2 maintains quantized weights, but communicating gradients may still be expensive. In this section we show that any reasonable quantization method for gradients can be used to reduce communication, while paying in exchange an increased variance. This trade-off is inherent, as the reduction in the number of bits requires injecting randomness, so as the entropy of the output is not smaller than that of the original message to be communicated.

To do so we use any gradient quantization method $\mathbf{Q}^{\mathbf{g}}$, as long as it is an unbiased estimator for the input it takes, and has bounded variance. Our formal requirements for $\mathbf{Q}^{\mathbf{g}}$ are the following.

Definition 11. We say that a gradient quantization operator $\mathbf{Q}^{\mathbf{g}}$ is a (σ_∇, b)-unbiased quantizer if it:

1. is an unbiased estimator: $\mathbb{E}\left[\mathbf{Q}^{\mathbf{g}}(\boldsymbol{g}\left(\boldsymbol{x}\right))|\boldsymbol{g}\left(\boldsymbol{x}\right)\right] = \boldsymbol{g}\left(\boldsymbol{x}\right)$,
2. has bounded variance on the stochastic gradients: $\mathbb{E}\left[\|\mathbf{Q}^{\mathbf{g}}(\boldsymbol{g}\left(\boldsymbol{x}\right)) - \boldsymbol{g}\left(\boldsymbol{x}\right)\|_2^2\,|\boldsymbol{g}\left(\boldsymbol{x}\right)\right] \leq \sigma_\nabla^2$,
3. requires b bits to communicate $\mathbf{Q}^{\mathbf{g}}(\boldsymbol{g}\left(\boldsymbol{x}\right))$.

By Lemma 5, these requirements are automatically satisfied by our shift-and-round quantization operator $\mathbf{Q}^{\mathbf{w}}$, and we can show that σ_∇ and b are determined by the ℓ_1 norm of $\boldsymbol{g}\left(\boldsymbol{x}\right)$.

Standard Quantization Schemes and Their Communication Cost. Another standard gradient quantization scheme can be obtained by independently rounding each coordinate to one of the neighboring points on the quantization grid, with an appropriate probability. An identical scheme has been previously used in other related works on gradient quantization (Alistarh et al., 2017).

Definition 12 (quantization by flipping a coin)**.** Let $\delta > 0$ be a scalar defining the coarseness of the quantization grid. Let the operator $\mathbf{Q}_\delta : \mathbb{R} \to \delta\mathbb{Z}$ defined as

$$\mathbf{Q}_\delta\left(x\right) = \begin{cases} \delta\left\lfloor\frac{x}{\delta}\right\rfloor & \text{with probability } 1 - \left(\frac{x}{\delta} - \left\lfloor\frac{x}{\delta}\right\rfloor\right) \\ \delta\left(\left\lfloor\frac{x}{\delta}\right\rfloor + 1\right) & \text{with probability } \frac{x}{\delta} - \left\lfloor\frac{x}{\delta}\right\rfloor \end{cases}$$

where $\delta > 0$. We apply $\mathbf{Q}_\delta$ to vectors, with the meaning that it is independently applied to each coordinate.

It is fairly easy to prove that this satisfies very similar properties to those proved for $\mathbf{Q}^{\mathbf{w}}$ in Lemma 5, which we quickly prove in Section E.4.5. We notice an important difference between these two quantization methods. While $\mathbf{Q}$ independently quantizes each coordinate, the quantization in $\mathbf{Q}^{\mathbf{w}}$ is done dependently across coordinates, and the output is always a vector in $\delta\mathbb{Z}^n + r\mathbf{1}$, for a randomly sampled scalar r. Although morally they are quite similar (in fact, the shift after rounding could just as well be ignored, and still have an unbiased extimator), it is important if we want to relate the quality of the final solution to the best set of weights from a reasonably chosen grid. This difference is apparent when trying to provide bounds of the type of Lemma 4, bu this attempt falls through in the case of the $\mathbf{Q}$ operator.

As we can naively relate the communication cost of a quantized gradient to its sparsity, it is important to discuss quantitative bounds. In both cases, the sparsity bound depends on the ℓ_1 norm of the quantized vector, and its easy to see that it is tight. By comparison, the bound from (Alistarh et al., 2017) is provided in terms of the ℓ_2 norm of the vector, but pays an additional $\sqrt{n}$ factor, which is suboptimal when the input is analytically sparse. For $\mathbf{Q}^{\mathbf{w}}$ and $\mathbf{Q}$, we see that the variance introduced by quantizing a generic vector $\boldsymbol{v}$ is bounded by $\delta\left\|\boldsymbol{v}\right\|_1$, while its sparsity is $\left\|\boldsymbol{v}\right\|_1/\delta$. Hence a naive encoding of this quantized gradient requires at most $O\left(\frac{\|\boldsymbol{v}\|_1}{\delta}\left(\ln n + \ln\left\|\boldsymbol{v}\right\|_1\right)\right)$ bits of communication.

SGD with Weight and Gradient Quantization. For gradient quantization operators that are unbiased estimators, we can use

them as stochastic gradients inside the scheme we derived in Theorem 2. To do so we crucially use the following identity involving conditional variance:

Lemma 13 (Law of total variance)**.** *Given random variables X and Y, one has that*

$$\mathit{Var}\left[Y\right] = \mathbb{E}\left[\mathit{Var}\left[Y|X\right]\right] + \mathit{Var}\left[\mathbb{E}\left[Y|X\right]\right] .$$

Corollary 14. *Consider a stochastic gradient estimator $\boldsymbol{g}\left(\boldsymbol{x}\right)$ such that $\mathbb{E}\left[\boldsymbol{g}\left(\boldsymbol{x}\right)|\boldsymbol{x}\right] = \nabla f\left(\boldsymbol{x}\right)$ and $\mathbb{E}\left[\left\|\boldsymbol{g}\left(\boldsymbol{x}\right) - \nabla f\left(\boldsymbol{x}\right)\right\|_2^2 |\boldsymbol{x}\right] \leq \sigma^2$. Consider (σ_∇, b)-unbiased quantizer (Definition 11). Then*

$$\mathbb{E}\left[\mathbf{Q}_\delta\left(\boldsymbol{g}\left(\boldsymbol{x}\right)\right)|\boldsymbol{x}\right] = \nabla f\left(\boldsymbol{x}\right) ,$$

i.e. it is an unbiased estimator for the gradient, and

$$\mathbb{E}\left[\left\|\mathbf{Q}_\delta\left(\boldsymbol{g}\left(\boldsymbol{x}\right)\right) - \nabla f\left(\boldsymbol{x}\right)\right\|_2^2\right] \leq \sigma_\nabla^2 + \sigma^2 .$$

Proof. The fact that the quantized gradient is an unbiased estimator for $\nabla f\left(\boldsymbol{x}\right)$ follows from the law of total expectation, as

$$\mathbb{E}\left[\mathbf{Q}_\delta\left(\boldsymbol{g}\left(\boldsymbol{x}\right)\right)|\boldsymbol{x}\right] = \mathbb{E}\left[\mathbb{E}\left[\mathbf{Q}_\delta\left(\boldsymbol{g}\left(\boldsymbol{x}\right)\right)|\boldsymbol{x}, \boldsymbol{g}\left(\boldsymbol{x}\right)\right]\right] = \mathbb{E}\left[\mathbb{E}\left[\boldsymbol{g}\left(\boldsymbol{x}\right)|\boldsymbol{x}\right]\right] = \nabla f\left(\boldsymbol{x}\right) .$$

For the variance, we use Lemma 13 to write:

$$\begin{aligned}
\mathbb{E}\left[\left\|\mathbf{Q}_\delta\left(\boldsymbol{g}\left(\boldsymbol{x}\right)\right) - \nabla f\left(\boldsymbol{x}\right)\right\|_2^2\right] &= \mathrm{Var}\left[\mathbf{Q}_\delta\left(\boldsymbol{g}\left(\boldsymbol{x}\right)\right)\right] \\
&= \mathbb{E}\left[\mathrm{Var}\left[\mathbf{Q}_\delta\left(\boldsymbol{g}\left(\boldsymbol{x}\right)\right)|\boldsymbol{g}\left(\boldsymbol{x}\right)\right]\right] + \mathrm{Var}\left[\mathbb{E}\left[\mathbf{Q}_\delta\left(\boldsymbol{g}\left(\boldsymbol{x}\right)\right)|\boldsymbol{g}\left(\boldsymbol{x}\right)\right]\right] \\
&\leq \sigma_\nabla^2 + \mathrm{Var}\left[\boldsymbol{g}\left(\boldsymbol{x}\right)\right] \\
&= \sigma_\nabla^2 + \sigma^2 .
\end{aligned}$$

□

Finally, combining Theorem 2 with Corollary 14, we obtain the final result from Corollary 3.

E.4. Deferred Proofs

E.4.1. Proof of Lemma 5

Proof. For both the mean and variance computation, it suffices to prove these bounds for the scalar operator.

We note that by definition $\mathbf{Q}_{r,\delta}^{\mathbf{w}}\left(x\right) - r = \mathbf{Q}_{0,\delta}^{\mathbf{w}}\left(x - r\right) = \delta \cdot \mathbf{Q}_{0,1}^{\mathbf{w}}\left(\frac{x-r}{\delta}\right) := \delta \cdot \left\lfloor \frac{x-r}{\delta} \right\rceil$. Also let $\{x\} = x - \lfloor x \rfloor$ denote the fractional part of x. We can easily verify that for any scalar $0 \leq z < 1$, we have

$$\mathbb{E}_{u \sim \mathrm{Unif}([-1/2, 1/2))}\left[\left\lfloor z + u \right\rceil\right] = z . \tag{6}$$

This is because $\lfloor z + u \rceil = 1$ if and only if $z + u \geq 1/2$ i.e. $u \geq \frac{1}{2} - z$, which happens with probability z. Now we can express the expectation of $\mathbf{Q}_{r,\epsilon}^{\mathbf{w}}\left(x\right)$ as follows:

$$\begin{aligned}
\mathbb{E}\left[\mathbf{Q}_\delta^{\mathbf{w}}\left(x\right)\right] &= \mathbb{E}_r\left[\mathbf{Q}_{0,\delta}^{\mathbf{w}}\left(x - r\right) + r\right] \\
&= \mathbb{E}_r\left[\mathbf{Q}_{0,\delta}^{\mathbf{w}}\left(x - r\right)\right] + \mathbb{E}_r\left[r\right] \\
&= \mathbb{E}_r\left[\mathbf{Q}_{0,\delta}^{\mathbf{w}}\left(\delta\left\lfloor\frac{x}{\delta}\right\rfloor + \delta\left\{\frac{x}{\delta}\right\} - r\right)\right] + \mathbb{E}_r\left[r\right] \\
&= \delta\left\lfloor\frac{x}{\delta}\right\rfloor + \mathbb{E}_r\left[\mathbf{Q}_{0,\epsilon}^{\mathbf{w}}\left(\delta\left\{\frac{x}{\delta}\right\} - r\right)\right] + \mathbb{E}_r\left[r\right] \\
&= \delta\left\lfloor\frac{x}{\delta}\right\rfloor + \mathbb{E}_r\left[\delta \cdot \mathbf{Q}_{0,1}^{\mathbf{w}}\left(\left\{\frac{x}{\delta}\right\} - \frac{r}{\delta}\right)\right] + \mathbb{E}_r\left[r\right] .
\end{aligned}$$

In the last line we used the fact that $\mathbf{Q}^{\mathbf{w}}_{0,\delta}(y) = \delta \cdot \mathbf{Q}^{\mathbf{w}}_{0,1}\left(\frac{y}{\delta}\right)$. Now we reparameterize by using $u := r/\delta$, where $r \sim \text{Unif}([-1/2, 1/2))$. This allows to write the term in the middle as:

$$\mathbb{E}_r\left[\delta \cdot \mathbf{Q}^{\mathbf{w}}_{0,1}\left(\left\{\frac{x}{\delta}\right\} - \frac{r}{\delta}\right)\right] = \delta \cdot \mathbb{E}_{u\sim\text{Unif}([-1/2,1/2))}\left[\mathbf{Q}^{\mathbf{w}}_{0,1}\left(\left\{\frac{x}{\delta}\right\} - u\right)\right] = \delta \cdot \left\{\frac{x}{\delta}\right\},$$

were we used (6). Plugging back in we obtain that

$$\mathbb{E}\left[\mathbf{Q}^{\mathbf{w}}_{r,\delta}(x)\right] = \delta\left\lfloor\frac{x}{\delta}\right\rfloor + \delta \cdot \left\{\frac{x}{\delta}\right\} + 0 = x\,.$$

Next we compute the scalar variance:

$$\begin{aligned}
\mathbb{E}\left[\left(\mathbf{Q}^{\mathbf{w}}_{\delta}(x) - x\right)^2\right] &= \mathbb{E}_r\left[\left(\mathbf{Q}^{\mathbf{w}}_{0,\delta}(x-r) - x\right)^2\right] \\
&= \mathbb{E}_r\left[\left(\delta\left\lfloor\frac{x}{\delta}\right\rfloor + \delta \cdot \mathbf{Q}^{\mathbf{w}}_{0,1}\left(\left\{\frac{x}{\delta}\right\} - \frac{r}{\delta}\right) - x\right)^2\right] \\
&= \mathbb{E}_r\left[\delta^2 \cdot \left(\left\lfloor\frac{x}{\delta}\right\rfloor + \mathbf{Q}^{\mathbf{w}}_{0,1}\left(\left\{\frac{x}{\delta}\right\} - \frac{r}{\delta}\right) - \frac{x}{\delta}\right)^2\right] \\
&= \mathbb{E}_{u\sim\text{Unif}([-1/2,1/2))}\left[\delta^2 \cdot \left(\left\lfloor\frac{x}{\delta}\right\rfloor + \mathbf{Q}^{\mathbf{w}}_{0,1}\left(\left\{\frac{x}{\delta}\right\} - u\right) - \frac{x}{\delta}\right)^2\right] \\
&= \delta^2 \cdot \mathbb{E}_{u\sim\text{Unif}([-1/2,1/2))}\left[\left(\mathbf{Q}^{\mathbf{w}}_{0,1}\left(\left\{\frac{x}{\delta}\right\} - u\right) - \left\{\frac{x}{\delta}\right\}\right)^2\right].
\end{aligned}$$

Now we use the fact that for any scalar $0 \le z < 1$ one has that

$$\mathbb{E}_{u\sim\text{Unif}([-1/2,1/2))}\left[\left(\lfloor z+u\rceil - z\right)^2\right] = z^2\,.$$

This follows from the fact that $\lfloor z+u\rceil = 1$ iff $u \ge 1/2 - z$, which happens with probability z, and makes the expectation equal to

$$\int_{-1/2}^{1/2-z} z^2 du + \int_{1/2-z}^{1/2} (1-z)^2\, du = z\,(1-z)\,,$$

which leads us to

$$\mathbb{E}\left[\left(\mathbf{Q}^{\mathbf{w}}_{\delta}(x) - x\right)^2\right] = \delta^2 \cdot \left\{\frac{x}{\delta}\right\}\left(1 - \left\{\frac{x}{\delta}\right\}\right),$$

which gives us what we needed.

Finally, for the sparsity bound, let us understand when a single scalar gets rounded to zero (before shifting back by r). We have that for $x \in \mathbb{R}$,

$$\begin{aligned}
\mathbb{P}\left[\mathbf{Q}^{\mathbf{w}}_{r,\delta}(x) - r = 0\right] &= \mathbb{P}\left[\mathbf{Q}^{\mathbf{w}}_{r,1}\left(\frac{x}{\delta}\right) - r = 0\right] = \begin{cases}\int_{-1/2}^{1/2} \mathbf{1}_{\lfloor\frac{x}{\delta}-r\rceil=0} dr, & |x| < \delta, \\ 0, & \delta \le |x|\,,\end{cases} \\
&= \begin{cases}\int_{-1/2}^{1/2} \mathbf{1}_{-\frac{1}{2}\le\frac{x}{\delta}-r\le\frac{1}{2}} dr, & |x| < \delta, \\ 0, & \delta \le |x|\,,\end{cases} = \begin{cases}\int_{-1/2}^{1/2} \mathbf{1}_{\frac{x}{\delta}-\frac{1}{2}\le r\le\frac{x}{\delta}+\frac{1}{2}} dr, & \left|\frac{x}{\delta}\right| < 1, \\ 0, & 1 \le \left|\frac{x}{\delta}\right|,\end{cases} \\
&= \min\left\{\frac{x}{\delta} + \frac{1}{2}, \frac{1}{2}\right\} - \max\left\{\frac{x}{\delta} - \frac{1}{2}, -\frac{1}{2}\right\} = \frac{1}{2} + \min\left\{\frac{x}{\delta}, 0\right\} - \left(-\frac{1}{2} + \max\left\{\frac{x}{\delta}, 0\right\}\right) \\
&= 1 + \min\left\{\frac{x}{\delta}, 0\right\} - \max\left\{\frac{x}{\delta}, 0\right\} = 1 - \left|\frac{x}{\delta}\right|\,.
\end{aligned}$$

which shows that

$$\mathbb{E}\left[\left\|\mathbf{Q}^{\mathbf{g}}_{\delta}(\boldsymbol{v})\right\|_0\right] = \sum_{i=1}^{n}\left(1 - \mathbb{P}\left[\mathbf{Q}^{\mathbf{g}}_{\delta}(v_i) = 0\right]\right) = \sum_{i=1}^{n}\begin{cases}\left|\frac{v_i}{\delta}\right|, & |v_i| < \delta, \\ 1, & \delta \le |v_i|\,,\end{cases} \le \|\boldsymbol{v}\|_1/\delta\,.$$

This concludes the proof. □

E.4.2. Proof of Lemma 4

Proof. It suffices to prove this coordinate-wise. From Lemma 5 we have that for any $x \in \mathbb{R}$,

$$\mathbb{E}\left[\left(\mathbf{Q}_\delta^{\mathbf{w}}(x) - x\right)^2\right] = \delta^2\left(1 - \left\{\frac{x}{\delta}\right\}\right)\left\{\frac{x}{\delta}\right\}$$

and similarly for $\delta_\star$. Let $k = \delta_\star/\delta$. Then

$$\mathbb{E}\left[\left(\mathbf{Q}_{\delta_\star}^{\mathbf{w}}(x) - x\right)^2\right] = k^2\delta^2\left(1 - \left\{\frac{x/\delta}{k}\right\}\right)\left\{\frac{x/\delta}{k}\right\}$$

Applying the inequality from Lemma 6, we conclude that

$$\mathbb{E}\left[\left(\mathbf{Q}_\delta^{\mathbf{w}}(x) - x\right)^2\right] = \delta^2\left(1 - \left\{\frac{x}{\delta}\right\}\right)\left\{\frac{x}{\delta}\right\} \leq \delta^2 \cdot k\left(1 - \left\{\frac{x}{k\delta}\right\}\right)\left\{\frac{x}{k\delta}\right\} = \frac{1}{k}\mathbb{E}\left[\left(\mathbf{Q}_{\delta_\star}^{\mathbf{w}}(x) - x\right)^2\right] .$$

Applying this bound to all coordinates we obtain

$$\mathbb{E}\left[\left\|\mathbf{Q}_\delta^{\mathbf{w}}(\boldsymbol{x}) - \boldsymbol{x}\right\|_2^2\right] \leq \frac{\delta}{\delta_\star}\mathbb{E}\left[\left\|\mathbf{Q}_{r,\delta_\star}^{\mathbf{w}}(\boldsymbol{x}) - \boldsymbol{x}\right\|_2^2\right] .$$

Also since $\mathbf{Q}_{r,\delta_\star}^{\mathbf{w}}$ rounds to the nearest point in $\delta_\star\mathbb{Z} + r$, clearly $\left\|\mathbf{Q}_{r,\delta_\star}^{\mathbf{w}}(\boldsymbol{x}) - \boldsymbol{x}\right\|_2^2 \leq \left\|\boldsymbol{x}_{r,\delta_\star}^\star - \boldsymbol{x}\right\|_2^2$ for all r. Taking expectations on both sides and combining with the previous inequality concludes the proof. □

E.4.3. Proof of Lemma 7

Proof. Using smoothness we have

$$\begin{aligned}
f(\boldsymbol{x} + \boldsymbol{\Delta}) &\leq f(\boldsymbol{x}) + \langle\nabla f(\boldsymbol{x}), \boldsymbol{\Delta}\rangle + \frac{\beta}{2}\|\boldsymbol{\Delta}\|_2^2 \\
&= f(\boldsymbol{x}) + (1-\eta)\langle\nabla f(\boldsymbol{x}), \boldsymbol{\Delta}\rangle - \frac{\eta^2}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2 + \left(\frac{\eta^2}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2 + \eta\langle\nabla f(\boldsymbol{x}), \boldsymbol{\Delta}\rangle + \frac{\beta}{2}\|\boldsymbol{\Delta}\|_2^2\right) \\
&= f(\boldsymbol{x}) + (1-\eta)\langle\nabla f(\boldsymbol{x}), \boldsymbol{\Delta}\rangle - \frac{\eta^2}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2 + \frac{\beta}{2}\left\|\frac{\eta}{\beta}\nabla f(\boldsymbol{x}) + \boldsymbol{\Delta}\right\|_2^2 .
\end{aligned}$$

□

E.4.4. Proof of Lemma 8

The proof is standard and can be found in literature, such as (Karimi et al., 2016). However, for completeness we reproduce it here.

Proof. Let $g(\boldsymbol{x}) = \sqrt{f(\boldsymbol{x}) - f^*}$ for which we have

$$\nabla g(\boldsymbol{x}) = \frac{1}{2\sqrt{f(\boldsymbol{x}) - f^*}}\nabla f(\boldsymbol{x}) .$$

Using the α-PL condition we have

$$\|\nabla g(\boldsymbol{x})\|^2 = \frac{1}{4(f(\boldsymbol{x}) - f^*)} \cdot \|\nabla f(\boldsymbol{x})\|^2 \geq \frac{1}{2(f(\boldsymbol{x}) - f^*)} \cdot \alpha \cdot (f(\boldsymbol{x}) - f^*) = \frac{\alpha}{2} .$$

Now starting at some $\boldsymbol{x}_0$, we consider the dynamic $\dot{\boldsymbol{x}} = -\nabla g(\boldsymbol{x})$. We see that this always decreases function value until it reaches some $\boldsymbol{x}_T$ for which $\nabla g(\boldsymbol{x}_T) = 0$ and hence by the PL inequality, $\boldsymbol{x}_T$ is a minimizer i.e. $f(\boldsymbol{x}_T) = f^*$. Now we can write

$$\begin{aligned}
g(\boldsymbol{x}_T) &= g(\boldsymbol{x}_0) + \int_0^T \langle\nabla g(\boldsymbol{x}_t), \dot{\boldsymbol{x}}_t\rangle\, dt = g(\boldsymbol{x}_0) + \int_0^T \langle\nabla g(\boldsymbol{x}_t), -\nabla g(\boldsymbol{x}_t)\rangle\, dt \\
&= g(\boldsymbol{x}_0) - \int_0^T \|\nabla g(\boldsymbol{x}_t)\|^2\, dt .
\end{aligned}$$

Thus

$$g\left(\boldsymbol{x}_0\right)-g\left(\boldsymbol{x}_T\right)=\int_0^T\left\|\nabla g\left(\boldsymbol{x}_t\right)\right\|^2 dt \geq \sqrt{\frac{\alpha}{2}} \cdot \int_0^T\left\|\nabla g\left(\boldsymbol{x}_t\right)\right\| dt=\sqrt{\frac{\alpha}{2}} \cdot \int_0^T\left\|\dot{\boldsymbol{x}}_t\right\| dt ,$$

where we used our lower bound on the norm of $\nabla g\left(\boldsymbol{x}\right)$. Finally, we use the fact that the last integral lower bounds the total movement of $\boldsymbol{x}$ as it moves from $\boldsymbol{x}_0$ to $\boldsymbol{x}_T$. Thus

$$\int_0^T\left\|\dot{\boldsymbol{x}}_t\right\| dt \geq\left\|\boldsymbol{x}_0-\boldsymbol{x}_T\right\| ,$$

so

$$g\left(\boldsymbol{x}_0\right)-g\left(\boldsymbol{x}_T\right) \geq \sqrt{\frac{\alpha}{2}}\left\|\boldsymbol{x}_0-\boldsymbol{x}_T\right\| ,$$

which enables us to conclude that

$$f\left(\boldsymbol{x}_0\right)-f^* \geq \frac{\alpha}{2}\left\|\boldsymbol{x}_0-\boldsymbol{x}_T\right\|^2 ,$$

where $\boldsymbol{x}_T$ is some global minimizer of f. This concludes the proof. □

E.4.5. Bound for Quantization by Coin Flip

Lemma 15. *Let $\boldsymbol{v} \in \mathbb{R}^n$, and let $\delta>0$, and let $\mathbf{Q}_\delta$ be the quantization operator from Definition 12. Then,*

$$\begin{aligned}
\mathbb{E}\left[\mathbf{Q}_\delta(\boldsymbol{v})\right] &= \boldsymbol{v} , \\
\mathbb{E}\left[\left\|\mathbf{Q}_\delta(\boldsymbol{v})-\boldsymbol{v}\right\|_2^2\right] &= \delta^2 \cdot \sum_{i=1}^n\left\{\frac{v_i}{\delta}\right\}\left(1-\left\{\frac{v_i}{\delta}\right\}\right) , \\
\mathbb{E}\left[\left\|\mathbf{Q}_\delta(\boldsymbol{v})\right\|_0\right] &\leq\|\boldsymbol{v}\|_1 / \delta .
\end{aligned}$$

Proof. For the expectation and variance, it suffices to prove that these bound holds coordinate-wise. Let $x \in \mathbb{R}$, and write $x=\delta\left(\left\lfloor\frac{x}{\delta}\right\rfloor+\left\{\frac{x}{\delta}\right\}\right)$ so that

$$\begin{aligned}
\mathbb{E}\left[\mathbf{Q}_\delta(x)\right] &= \mathbb{E}\left[\mathbf{Q}_\delta\left(\delta\left(\left\lfloor\frac{x}{\delta}\right\rfloor+\left\{\frac{x}{\delta}\right\}\right)\right)\right] \\
&= \delta\left\lfloor\frac{x}{\delta}\right\rfloor+\mathbb{E}\left[\mathbf{Q}_\delta\left(\delta\left\{\frac{x}{\delta}\right\}\right)\right] \\
&= \delta\left\lfloor\frac{x}{\delta}\right\rfloor+\mathbb{E}\left[\mathbf{Q}_\delta\left(\delta\left\{\frac{x}{\delta}\right\}\right)\right] \\
&= \delta\left\lfloor\frac{x}{\delta}\right\rfloor+\delta \cdot\left\{\frac{x}{\delta}\right\} \\
&= x .
\end{aligned}$$

Similarly we write the variance as:

$$\begin{aligned}
\mathbb{E}\left[\left(\mathbf{Q}_\delta(x)-x\right)^2\right] &= \mathbb{E}\left[\left(\mathbf{Q}_\delta\left(\delta\left\{\frac{x}{\delta}\right\}\right)-\delta\left\{\frac{x}{\delta}\right\}\right)^2\right] \\
&= \left(1-\left\{\frac{x}{\delta}\right\}\right)\left(\delta\left\{\frac{x}{\delta}\right\}\right)^2+\left\{\frac{x}{\delta}\right\} \cdot\left(\delta-\delta\left\{\frac{x}{\delta}\right\}\right)^2 \\
&= \delta^2\left(\left(1-\left\{\frac{x}{\delta}\right\}\right)\left\{\frac{x}{\delta}\right\}^2+\left\{\frac{x}{\delta}\right\} \cdot\left(1-\left\{\frac{x}{\delta}\right\}\right)^2\right) \\
&= \delta^2\left(1-\left\{\frac{x}{\delta}\right\}\right)\left\{\frac{x}{\delta}\right\} ,
\end{aligned}$$

For the sparsity bound, we need to understand when a single scalar gets rounded to zero. We have that for $x \in \mathbb{R}$,

$$\mathbb{P}\left[\mathbf{Q}_\delta(x)=0\right]=\begin{cases}1-\left|\frac{x}{\delta}\right| , & |x|<\delta, \\ 0, & \delta \leq|x| ,\end{cases}$$

which shows that

$$\begin{aligned}
\mathbb{E}\left[\left\|\mathbf{Q}_\delta\left(\boldsymbol{v}\right)\right\|_0\right] &= \sum_{i=1}^{n}\left(1-\mathbb{P}\left[\mathbf{Q}_\delta\left(v_i\right)=0\right]\right) \\
&= \sum_{i=1}^{n}\begin{cases}\left|\frac{v_i}{\delta}\right|, & \left|v_i\right|<\delta, \\ 1, & \delta\leq\left|v_i\right|,\end{cases} \\
&\leq \left\|\boldsymbol{v}\right\|_1/\delta\,.
\end{aligned}$$

□