IntSGD: Floatless Compression of Stochastic Gradients

Konstantin Mishchenko 1  Bokun Wang 1  Dmitry Kovalev 1  Peter Richtárik 1

Abstract
We propose a family of lossy integer compressions for Stochastic Gradient Descent (SGD) that do not communicate a single float. This is achieved by multiplying floating-point vectors with a number known to every device and then rounding to an integer number. Our theory shows that the iteration complexity of SGD does not change up to constant factors when the vectors are scaled properly. Moreover, this holds for both convex and non-convex functions, with and without overparameterization. In contrast to other compression-based algorithms, ours preserves the convergence rate of SGD even on non-smooth problems. Finally, we show that when the data is significantly heterogeneous, it may become increasingly hard to keep the integers bounded and propose an alternative algorithm, IntDIANA, to solve this type of problems.

1. Introduction
Many recent breakthroughs in machine learning were made possible due to the introduction of large, sophisticated and high capacity supervised models whose training requires days or even weeks of computation (Hinton et al., 2015; He et al., 2016; Huang et al., 2017; Devlin et al., 2018). However, it would not be possible to train them without corresponding advances in parallel and distributed algorithms capable of taking advantage of modern hardware. Very large models are typically trained on vast collections of training data stored in a distributed fashion across a number of compute nodes that need to communicate throughout the training process. In this scenario, reliance on efficient communication protocols is of utmost importance.

Theory vs. practice. However, due to the understandable and in a strict sense unavoidable tendency of theoreticians to resort to abstractions when designing new algorithms, various idiosyncrasies and constraints of real systems are often ignored, and methods that may seem efficient in theory are not necessarily efficient in practice (Dutta et al., 2020). For instance, large swaths of recent literature on communication efficient distributed training attributes the cost of sending a single vector from a worker to the server to the number of bits needed to represent it. Based on this abstraction, various elaborate vector compression techniques (see Table 1 in (Beznosikov et al., 2020; Xu et al., 2020; Safaryan et al., 2020)) and algorithms have been designed.

However, in real systems, efficiency of sending a vector is not fully characterized by the number of bits alone. Factors such as memory constraints of GPUs, design of message passing backends, and the unstable behavior of neural networks pose many restrictions on which communication-reduction techniques are useful in practice and which are not. For example, the low memory budget of GPUs often prevents us from introducing extra sequences such as those needed for error-feedback (Stich et al., 2018). Further, the implementation of communication primitives prioritizes algorithms that do not rely on elaborate decompression techniques such as those needed to support the random dithering compression of QSGD (Alistarh et al., 2017). Compression techniques such as rand-k or top-k sparsification can also suffer from long compression times (Dutta et al., 2020). Finally, training of neural networks is known to degrade once the mini-batch sizes exceed a certain limit, or if communication becomes too lossy.

SwitchML. Formally, the problem we study is that of parallel/distributed minimization of the average of $n$ functions, each of which is given as expectation:

$$
\min_{x \in \mathbb{R}^d} \left[ f(x) \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(x) \right], \quad f_i(x) \overset{\text{def}}{=} E_{\xi}[f_i(x; \xi)].
$$

These empirical risk minimization problems form the key formalism behind modern supervised machine learning. The recently proposed SwitchML framework of Sapio et al. (2021) for solving (1) is an example of a framework that works empirically well on real-world hardware. At its core is the integer compression of SGD updates combined with efficient in-network aggregation via communication primitives implemented on the switch. Before aggregation, each SwitchML worker scales its update vector by a float factor $\alpha$ available to all devices, after which the entries of the

---

1King Abdullah University of Science and Technology. Correspondence to: Konstantin Mishchenko <konsta.mish@gmail.com>.
scaled vector are rounded to integers and communicated. As no additional scaling or decompression is needed before aggregating the communicated vectors, their sums can be computed on the fly. Furthermore, since by design even a computationally-weak device such as a network switch can perform integer addition, communication compression to integers offers the user ample flexibility in choosing low-cost or non-traditional hardware. This property distinguishes the SwitchML compressor from other popular compression operators.

Sapio et al. (2021) remark that the choice of the scaling factors requires special care, and suggest to keep the rounded integers small to fit in 32 bits. However, their method is a heuristic, no efficient rule for making this possible was proposed, and in their presentation, one of the authors notes: "A bad choice of scaling factor can reduce the performance." This is where our theory comes to rescue. By rigorously and exhaustively analyzing integer compression based on scaling, we find adaptive parameter rules that do not require the expensive profiling employed by (Sapio et al., 2021). However, our work is also of independent interest, and can possibly find applications beyond the SwitchML framework.

1.1. Contributions

Our contributions can be summarized as follows.

- **IntSGD.** We develop IntSGD: a novel variant of distributed SGD using cheap integer communication compression based on scaling and randomized rounding (see Section 2). As a special case, we obtain the integer compression used in SwitchML (Sapio et al., 2021).

- **Integers both ways.** IntSGD offers bidirectional compression—from devices to the server and back—that can be efficiently combined with Reduce and All-Reduce primitives of message passing backends, such as MPI.

- **Scaling.** We develop several scaling rules, each giving rise to a different compressor, with different uses and properties. Our proposed rules include: 1–adaptive, 2–block-wise, 3–moving-averages, and 4–constant safeguard (see Section 4).

- **Rates.** For all of the proposed compressions we prove convergence rates of IntSGD that match those of full-precision SGD up to constant factors. Our results are tight and apply to both convex and non-convex problems. Our analysis does not require any extra assumption compared to those typically invoked for SGD. We show that there is a linear improvement in compression noise, which means that more aggressive integer compression can be used when the number of devices $n$ increases. In contrast to other compression-based methods, IntSGD has the same rate as that of full-precision SGD even on non-smooth problems. For example, the rate of error-feedback is $\frac{1}{n}$ times worse, where $\delta \approx 0$ is the compression ratio (Karimireddy et al., 2019). As a special case of our results, we obtain the first analysis of the integer compression used in SwitchML (Sapio et al., 2021).

- **IntDIANA.** We observe empirically that IntSGD struggles when the devices have heterogeneous (non-identical) data—an issue it shares with vanilla SGD—and propose an alternative method, IntDIANA, that can provably alleviate this issue. We also show that our tools are useful for extending the methodology beyond SGD methods, for example, to variance reduced methods (Johnson & Zhang, 2013; Allen-Zhu & Hazan, 2016; Kovalev et al., 2020) with integer compression. We refer the reader to the recent survey of Gower et al. (2020).  

2. Integer Compression and IntSGD

By randomized integer rounding we mean the mapping $\mathbb{I}: \mathbb{R} \rightarrow \mathbb{Z}$ defined by

$$
\mathbb{I}(t) = \begin{cases} 
\lfloor t \rfloor + 1, & \text{with probability } p_t = t - \lfloor t \rfloor, \\
\lfloor t \rfloor, & \text{with probability } 1 - p_t,
\end{cases}
$$

where $\lfloor t \rfloor$ denotes the floor of $t \in \mathbb{R}$, i.e., $\lfloor t \rfloor = k \in \mathbb{Z}$, where $k$ is such that $k \leq t < k + 1$. Note that

$$
\mathbb{E}[\mathbb{I}(t)] = (t - \lfloor t \rfloor)(\lfloor t \rfloor + 1) + (\lfloor t \rfloor + 1 - t)\lfloor t \rfloor = t.
$$

We extend this mapping to vectors $x \in \mathbb{R}^d$ by applying in element-wise: $\mathbb{I}(x)_i = \mathbb{I}(x_i)$.

2.1. Integer compression

Given a scaling vector $\alpha \in \mathbb{R}^d$ with nonzero entries, we further define the integer compression operator $Q : \mathbb{R}^d \rightarrow \mathbb{R}^d$ by

$$
Q(x) = \frac{1}{\alpha} \circ \mathbb{I}(\alpha \circ x),
$$

where $a \circ b \coloneqq (a_1 b_1, \ldots, a_d b_d) \in \mathbb{R}^d$ denotes the Hadamard product of two vectors $a = (a_1, \ldots, a_d) \in \mathbb{R}^d$ and $b = (b_1, \ldots, b_d) \in \mathbb{R}^d$.

As we show below, the integer compression operator (2) has a number of properties which will be useful in our analysis. In particular, it is unbiased, and its variance can be controlled by the choice of the scaling vector $\alpha \in \mathbb{R}^d_{++}$.

**Lemma 1.** For any $x \in \mathbb{R}^d$ and $\alpha \in \mathbb{R}^d_{++}$, we have

$$
\frac{1}{\alpha} \circ \mathbb{E}[\mathbb{I}(\alpha \circ x)] = x,
$$

and

$$
\mathbb{E}
\left[
\left\|
\frac{1}{\alpha} \circ \mathbb{I}(\alpha \circ x) - x
\right\|^2
\right]
\leq \frac{1}{\sum_{j=1}^{d} \frac{1}{\alpha_j^2}}.
$$

**Proof.** The variance of the rounding error is bounded as

$$
\frac{1}{\alpha} \circ \mathbb{I}(\alpha \circ x) - x
\leq \frac{1}{\alpha} \circ \mathbb{I}(\alpha \circ x) - \mathbb{E}[\mathbb{I}(\alpha \circ x)] + \mathbb{E}[\mathbb{I}(\alpha \circ x)] - x,
$$

where the second term is zero. The LHS can be bounded as

$$
\left\|
\frac{1}{\alpha} \circ \mathbb{I}(\alpha \circ x) - \mathbb{E}[\mathbb{I}(\alpha \circ x)]
\right\|
\leq \frac{1}{\sum_{j=1}^{d} \frac{1}{\alpha_j^2}}.
$$

The expected value of the rounding error can be bounded as

$$
\mathbb{E}
\left[
\left\|
\frac{1}{\alpha} \circ \mathbb{I}(\alpha \circ x) - x
\right\|^2
\right]
\leq \frac{1}{\sum_{j=1}^{d} \frac{1}{\alpha_j^2}}.
$$

\[\Box\]
Prior to communication, each worker $i$ adding them up forms aggregation by a master (e.g., a switch or a parameter server), which performs the randomized rounding operator $R$. We are ready to present our algorithm, IntSGD.

3. Analysis of IntSGD

To establish convergence of IntSGD, we introduce the following assumption on the scaling vector $\alpha_k = (\alpha_k, \ldots, \alpha_k, d)^T \in \mathbb{R}^{d+1}$.

Assumption 1. There exists $\sigma_Q > 0$ and $\beta \in [0, 1]$ such that

$$\sum_{t=0}^{d} \mathbb{E} \left[ \frac{n_t^2}{\alpha_{t,j}} \right] \text{ is bounded above by } \sum_{t=0}^{d} \beta^t \mathbb{E} \left[ \| x^{k-t} - x^{k-t-1} \|_2^2 \right].$$

While this assumption may look exotic, it captures precisely what we need to establish our convergence results, and it as we shall see in detail in Section 4, it holds for several practical choices of $\alpha_k$. In particular, $\sigma_Q$ will be useful for constant choice of $\alpha_k$ and $\beta > 0$ for moving average update.

3.1. Decomposition: quantized SGD recursion

The staring point of our analysis is the following recursion.

Let $\rho^{k+1} = \rho^k - 2\eta_k \delta_k + A^k + B^k,$

where $A^k = 2\eta_k^2 \mathbb{E} [\zeta^k]$ and $B^k = \frac{1}{\alpha^2} \sum_{j=1}^{d} \frac{n_j^2}{\alpha_{t,j}^2} - \| x^{k+1} - x^k \|_2^2$ are the SGD and quantization error terms, respectively.

We now show how to control the quantization error by choosing the scaling vector $\alpha_k$ in accordance with Assumption 1.

Lemma 3. If the assumptions of Lemma 2 hold together with Assumption 1, then

$$\mathbb{E} \left[ \rho^{k+1} \right] \leq \rho^0 - 2 \sum_{t=0}^{k} \eta_t \mathbb{E} [\delta^t] + 2 \sum_{t=0}^{k} \eta_t^2 \mathbb{E} [\zeta^t] + \frac{\sigma^2}{\alpha^2} \sum_{t=1}^{k} \frac{1}{\alpha_{t,j}^2}.$$ 

The above lemma shows that the quantization error is controlled by the size of $\sigma_Q$.

3.2. Non-smooth analysis: generic result

Let us now show that IntSGD works well even on non-smooth functions.

Assumption 2. Stochastic (sub)gradients $g_1^k, \ldots, g_n^k$ sampled at iteration $k$ satisfy the inequalities

$$\mathbb{E} \left[ \sum_{i=1}^{n} \mathbb{E}_k (g_i^k) \right] \leq G^2,$$

$$\mathbb{E}_k \left[ \| g_i^k - \mathbb{E}_k (g_i^k) \|_2^2 \right] \leq \sigma^2,$$

which is exactly what we need to establish our convergence results, and it as we shall see in detail in Section 4, it holds for several practical choices of $\alpha_k$. In particular, $\sigma_Q$ will be useful for constant choice of $\alpha_k$ and $\beta > 0$ for moving average update.
where the former inequality corresponds to G-Lipschitzness of \( f \) and the latter to bounded variance of stochastic (sub)gradients.

**Theorem 1.** Let functions \( f_1, \ldots, f_n \) be convex and Assumptions 1 and 2 be satisfied. Then

\[
\mathbb{E} \left[ (\hat{x}^k) - f(x^*) \right] \leq \frac{\mu + 2 \left( \sum_{i=1}^n \gamma_i^2 \right) \sum_{t=0}^k \eta_t^2}{2 \sum_{t=0}^k \eta_t},
\]

where \( \hat{x}^k = \frac{1}{\sum_{t=0}^k \eta_t} \sum_{t=0}^k \eta_t x^t \) is a weighted average of iterates.

The fact that Algorithm 1 does not rely on smoothness properties of the objective illustrates that it is more general and robust than error-feedback.

### 3.3. Smooth analysis: generic result

We now develop a theory for smooth objectives.

**Assumption 3.** There exist constants \( \mathcal{L}, \sigma_* \geq 0 \) such that the stochastic gradients \( g^*_1, \ldots, g^*_n \) at iteration \( k \) satisfy

\[
\mathbb{E}_k [g^*_k] = \nabla f(x^k) \quad \text{and} \quad \mathbb{E}_k \left[ \left\| \frac{1}{n} \sum_{i=1}^n g^*_i \right\|^2 \right] \leq \mathcal{L}(f(x^k) - f(x^*)) + \frac{\sigma^2_*}{n},
\]

(7)

Assumption 3 is known as the expected smoothness assumption (Gower et al., 2019). In its formulation, we divide the constant term \( \sigma^2_* \) by \( n \), which is justified by the following proposition.

**Proposition 1** (Section 3.3 in (Gower et al., 2019)). Let \( f_i(x) = \mathbb{E}_\xi[f_i(x; \xi)] \), \( g^*_i = \nabla f_i(x^*; \xi^*) \), and \( f_i(\cdot; \xi) \) be convex and its gradient be \( L_i \)-Lipschitz for any \( \xi \). Then, the second part of Assumption 3 is satisfied with \( \sigma^2_* \) defined as

\[
\frac{1}{n} \sum_{i=1}^n \mathbb{E}_\xi \left[ \left\| \nabla f_i(x^*; \xi) \right\|^2 \right] \text{ and } \mathcal{L} \text{ defined as } \max_{i=1, \ldots, n} L_i.
\]

In (Gower et al., 2019), this result is stated and proved in a more general form so for the reader’s convenience we provide a proof in the appendix.

**Theorem 2.** Assume that \( f \) is convex and Assumption 3 holds. If \( \eta_k \leq \frac{1}{2 \mathcal{L}} \) and \( \hat{x}^k = \frac{1}{\sum_{t=0}^k \eta_t} \sum_{t=0}^k \eta_t x^t \) is a weighted average of iterates, then

\[
\mathbb{E} \left[ (\hat{x}^k) - f(x^*) \right] \leq \frac{\mu + 2 \left( \sum_{i=1}^n \gamma_i^2 \right) \sum_{t=0}^k \eta_t^2}{2 \sum_{t=0}^k \eta_t}.
\]

**Corollary 1** (Overparameterized regime). When the model is overparameterized (i.e., the losses can be minimized to optimality simultaneously: \( \sigma_* = 0 \)), we can set \( \sigma^*_Q = 0 \) and obtain \( O \left( \frac{1}{k} \right) \) convergence.

### 3.4. Non-convex analysis: generic result

We now develop a theory for non-convex objectives.

**Assumption 4.** The gradient of \( f \) is \( L \)-Lipschitz and there exists \( f^\text{inf} \in \mathbb{R} \) such that \( f^\text{inf} \leq f(x) \) for all \( x \). Furthermore, for all \( i \) and \( k \) we have

\[
\mathbb{E} \left[ \left\| g^*_i - \nabla f_i(x^k) \right\|^2 \right] \leq \sigma^2.
\]

(8)

Our main result in the non-convex regime follows.

**Theorem 3.** Let \( f \) be \( L \)-smooth and let Assumption 1 hold. If \( \eta_k \leq \frac{1}{2L} \) for all \( k \), then

\[
\mathbb{E} \left[ \left\| \nabla f(\hat{x}^k) \right\|^2 \right] \leq \frac{f(x^0) - f^\text{inf} + \left( \frac{\mu^2}{\sigma^2_*} + \frac{\sigma^2}{\mathcal{L}^2} \right) \sum_{t=0}^k \eta_t^2}{\sum_{t=0}^k \eta_t}.
\]

where \( \hat{x}^k \) is sampled from \( \{x^0, \ldots, x^k\} \) with probabilities proportional to \( \eta_0, \ldots, \eta_k \).

### 3.5. Explicit complexity results

Having developed generic complexity results for IntSGD in the non-smooth (Section 3.2), smooth (Section 3.3) and non-convex (Section 3.4) regimes, we now derive explicit convergence rates.

**Corollary 2.** For any sequence of scaling vectors \( \alpha_k \) satisfying Assumption 1, we recover the following complexities:

(i) if \( f_1, \ldots, f_n \) are convex, Assumption 2 is satisfied and

\[
\eta_k = \eta = \frac{\sqrt{\mu^2}}{\sqrt{k(\gamma^2 + \sigma^2/n)}} = O \left( \frac{1}{\sqrt{k}} \right)
\]

for \( t = 0, \ldots, k \), then

\[
\mathbb{E} \left[ f(\hat{x}^k) - f(x^*) \right] = O \left( \frac{\sigma + \sigma_0}{\sqrt{k}} + \frac{\sigma^*_Q}{\sqrt{k}} \right);
\]

(9)

(ii) if \( f \) is convex, Assumption 3 holds and

\[
\eta_k = \min \left\{ \frac{1}{2 \mathcal{L}}, \frac{\sqrt{\mu^2}}{2 \sqrt{\mathbb{E}(\sigma_*, \sigma^*_Q)}} \right\},
\]

then

\[
\mathbb{E} \left[ f(\hat{x}^k) - f(x^*) \right] = O \left( \frac{\sigma + \sigma_Q}{\sqrt{k}} + \frac{\sigma^*_Q}{\sqrt{k}} \right);
\]

(iii) if \( f \) is non-convex, Assumption 4 holds and

\[
\eta_k = \min \left\{ \frac{1}{2 \mathcal{L}}, \frac{\sqrt{f(x^0) - f^\text{inf}} n}{2 \sqrt{k(\sigma^*_Q + \sigma^*_Q)}} \right\},
\]

then

\[
\mathbb{E} \left[ \left\| \nabla f(\hat{x}^k) \right\|^2 \right] = O \left( \frac{\sigma + \sigma_0}{\sqrt{k}} + \frac{f(x^0) - f^\text{inf}}{\sqrt{k}} \right).
\]

Note that when \( n \) grows, the compression noise \( \sigma^*_Q \) improves at the same rate as \( \sigma \) or \( \sigma_* \). This is the analogue of linear speed up and it allows us to use more aggressive compression.

### 4. Design of Scaling Vectors

From Corollary 2 one can see that by adjusting \( \sigma^*_Q \), we can control the impact of quantization on the complexity of
IntSGD. We now present several ways of picking $\alpha_k$ during the training. For simplicity, we assume that the first communication is exact, which allows us to estimate $\alpha_k$ adaptively without worrying about $\alpha_0$.

4.1. Adaptive choice of $\alpha_k$

**Proposition 2** (Adaptive $\alpha_k$). If we choose

$$\alpha_k = \frac{n_k \sqrt{d}}{\sqrt{2n_k \| x^k - x^{k-1} \|}} \sim \frac{\sqrt{d}}{\sqrt{n}}$$

then Assumption 1 holds with $\sigma_Q = 0$ and $\beta = 0$.

4.2. Constant choice of $\alpha_k$

**Proposition 3** (Constant $\alpha_k$). If $\alpha_k$ has all entries equal to the same number, $\alpha_k = \frac{\sqrt{d}}{\sqrt{n}} (1, \ldots, 1)^T$ (or $\alpha_k = \frac{\sqrt{d}}{\sigma} (1, \ldots, 1)^T$), then Assumption 1 holds with $\beta = 1$ and $\sigma_Q = \sigma_n$ (or $\sigma_Q = \sigma_n$, respectively).

4.3. Coordinate- and layer-wise $\alpha_{k,l}$

One can also consider applying an integer quantization with individual values of $\alpha_l$ for each coordinate or block, for instance, with an $\alpha_{l,i}$ corresponding to the $l$-th layer in a neural network. It is straightforward to see that this modification leads to the error $\sum_{i=1}^{B} \| \frac{\partial f_i}{\partial x_i} \|$, where $B$ is the total number of blocks and $d_l$ is the dimension of the $l$-th block.

**Proposition 4** (Adaptive block $\alpha_k$). Assume we are given a partition of all coordinates into $B \leq d$ blocks with dimensions $d_1, \ldots, d_B$, and denote by $(x^k)_l$ the $l$-th block of coordinates of $x^k$. Then Assumption 1 holds with

$$\alpha_{k,(l)} = \frac{n_k \sqrt{d_l}}{\sqrt{2n_k \| (x^k)_l - (x^{k-1})_l \|}} \sim \frac{\sqrt{d}}{\sqrt{n}}$$

for $l = 1, \ldots, B$.

There are two extreme cases in terms of how we can choose the blocks. One extreme is to set $B = 1$, in which case we have a single scalar for the whole vector. The other extreme is to use $B = d$, which means that $\alpha_k = \frac{n_k \sqrt{d}}{\sqrt{2n_k \| x^k - x^{k-1} \|}}$, where the division and absolute values are computed coordinate-wise.

4.4. Updating $\alpha_k$ with moving average

**Proposition 5** (Adaptive moving average for $\alpha_k$). Assumption 1 holds if we choose $\beta \in [0, 1)$ and

$$\alpha_k = \frac{n_k \sqrt{d}}{\sqrt{2n_k \| x^k - x^{k-1} \|}}$$

where $r_k = \beta r_{k-1} + (1 - \beta) \| x^k - x^{k-1} \|^2$.

4.5. Constant safeguard with moving average

**Proposition 6** (Adaptive moving average with constant safeguard for $\alpha_k$). Assumption 1 holds if we choose

Algorithm 2 IntSGD: adaptive block quantization

1: **Input:** $x^0 \in \mathbb{R}^d$, $\beta \in [0, 1)$, $\sigma_Q \geq 0$, $x^1 = x^0 - \eta_0 \frac{\sum_{i=1}^{n} g_i^0}{\| g_i^0 \|}$ a partitioning of $\mathbb{R}^d$ into $B$ blocks of sizes $d_1, \ldots, d_B$ such that $\mathbb{R}^d = \mathbb{R}^{d_1} \times \cdots \times \mathbb{R}^{d_B}$
2: for $k = 1, 2, \ldots$ do
3: Compute independent stochastic gradients $g_i^k$
4: $r_{k,l} = \beta r_{k-1,l} + (1 - \beta) \| (x^k)_l - (x^{k-1})_l \|^2$ (for each block $l = 1, \ldots, B$)
5: $\alpha_{k,l} = \frac{n_k \sqrt{d}}{\sqrt{2n_k r_{k,l} + \| g_i^k \|_2 \sigma_Q}}$
6: Let $(Q(g_i^k)_l) = \frac{1}{\alpha_{k,l}} \text{Int} \alpha_{k,l}(g_i^k)_l$
7: $x^{k+1} = x^k - \frac{\eta_k}{\sqrt{n}} \sum_{i=1}^{n} Q(g_i^k)$
8: end for

$\beta \in [0, 1)$, $\sigma_Q \geq 0$ and

$$\alpha_k = \frac{n_k \sqrt{d}}{\sqrt{2n_k \| x^k - x^{k-1} \|}}$$

where $r_k = \beta r_{k-1} + (1 - \beta) \| x^k - x^{k-1} \|^2$.

The benefit of the proposed modification is that regardless of the value of $r_k$, we have $\| \alpha_k \circ g_i^k \|_\infty \leq \sqrt{d} \frac{\| g_i^k \|_\infty}{\sigma_Q}$ for any $i$. This way, we can verify that no integer larger than $2^{31}$ or even $2^{15}$ is ever used in communication, so that the compressed vector entries fit 32 or 16 bits.

4.6. Combining techniques

Clearly, if we combine the tricks introduced above, the proof will still go through. For illustration, we provide an algorithm statement that can be similarly shown to converge with the $O(1/\sqrt{d})$ rate in both smooth and non-smooth cases. See Algorithm 2 for the pseudocode.

4.7. Compression efficiency

Let us now discuss the number of bits needed for the compressed vectors. Although the main attraction of IntSGD is that it can perform efficient in-network communication, we may also hope to gain from smaller size of the updates.

Consider for simplicity the case where $\| x^k - x^{k-1} \| \approx \| \eta_k g_i^k \|$ with some $i$. The adaptive scheme with scalar $\alpha_k$ gives $\alpha_k = \frac{n_k \sqrt{d}}{\sqrt{2n_k \| x^k - x^{k-1} \|}} \approx \frac{n_k \sqrt{d}}{\sqrt{2n_k \| x^k - x^{k-1} \|}} = \frac{\sqrt{d}}{\sqrt{2n}}$, so that $\| \alpha_k g_i^k \|_\infty = \frac{\sqrt{d}}{\sqrt{2n}} \| g_i^k \|_\infty \leq \frac{\sqrt{d}}{\sqrt{2n}}$. Since we only use signed integers, we need at most $1 + \log_2 \frac{\sqrt{d}}{\sqrt{2n}}$ bits for each coordinate. For instance, for $d \sim 10^{10}$ and $n \sim 100$, the upper bound is $1 + \log_2 (\sqrt{5} \cdot 10^7) < 14$ bits.

The situation becomes even better when $\| g_i^k \| \ll \| g_i^k \|_\infty$, i.e., when the stochastic gradients are dense. This property has been observed in certain empirical evaluations for deep
neural networks; see for example the study in (Bernstein et al., 2018).

Our block-wise and coordinate-wise compression can further benefit from reduced dimension factors in the upper bounds, leading to the estimate of \( \log_2 \frac{\sqrt{d}}{\sigma} \) bits for block with dimension \( d_i \). However, for smaller blocks it is less likely to happen that \( \|x^k - (x^{k-1})_i\| \approx \|g_k(x^k)_i\| \), so the estimate should be taken with a grain of salt. We hypothesize that using \( \sigma_Q \) as in Proposition 6 is required to make block compression robust. Notice that if stochastic gradients have bounded coordinates, i.e., \( \|g_k\|_\infty \leq G_\infty \) for all \( i, k \), then we would need at most \( 1 + \log_2 \frac{\sqrt{d}}{\sigma} \) bits to encode the integers. Since any \( \sigma_Q \leq \sigma + \sqrt{nG} \) does not change the rate in the non-smooth case (see Equation (9)), we get for free the upper bound of \( 1 + \log_2 \frac{\sqrt{d}}{\sigma} \) bits.

Finally, we note that when there is no noise in the stochastic gradients and the objective is smooth, we have \( \|g_k\| = \nabla f_i(x^k) \) which might not converge to 0, while \( \|x^k - x^{k-1}\| \) may converge to 0. Therefore, we end up with \( \alpha_k \rightarrow +\infty \), which results in increasingly large integer numbers and expensive compressions.

5. Handling Heterogeneous Data

IntSGD can be equipped with the full gradient or variance-reduced gradient estimator to enjoy faster convergence than \( O\left( \frac{1}{\sqrt{\mu}} \right) \) shown in Corollary 2. For example, if we plug \( \sigma_s = 0 \) (no variance) and \( \sigma_Q = 0 \) (no safeguard of \( \alpha_k \)) into item 2 of Corollary 2, the convergence rate of IntSGD is \( O\left( \frac{1}{\sqrt{\mu}} \right) \). However, when the data is heterogeneous (i.e., minimizing \( f(x) = \frac{1}{n} \sum_{i=1}^n f_i(x) \) will not make \( \|\nabla f_i(x^*)\| = 0 \), \( \forall i \in [n] \), the transmitted integer of IntSGD with \( \sigma_s = 0 \), \( \sigma_Q = 0 \) can be gigantically large, which leads to very inefficient communications or even exception value error. E.g., if we choose the adaptive \( \alpha_k \) and the full gradient \( g_k = \nabla f_i(x^k) \), the largest integer to transmit from worker \( i \) to the master is \( \|\alpha_k \nabla f_i(x^k)\|_\infty \approx \|\nabla f_i(x^k)\|_\infty \frac{\sqrt{d}}{\|x^k - x^{k-1}\|} \), where the denominator is 0 while the nominator is nonzero as the iterate converges to the optimum.

5.1. New algorithm: IntDIANA

To alleviate this issue, one needs to compress gradient differences as is done for example by (Mishchenko et al., 2019) in their DIANA method. By marrying IntSGD with the DIANA trick, we obtain IntDIANA (Algorithm 3).

For IntDIANA with adaptive \( \alpha_k \), the largest transmitted integer from worker to the master is \( \|\alpha_k(g_k - h_k)\|_\infty \approx \|g_k - h_k\|_\infty \frac{\sqrt{d}}{\|x^k - x^{k-1}\|} \). We will show that both the nominator and the denominator are infinitesimal when \( x^k \) converges to the optimum, such that the issue mentioned above can be hope-
2. If $\mu = 0$, the iterates of IntDIANA with adaptive $
abla k = \frac{\eta}{\sqrt{n} \|x_k - x_{k-1}\|}$ satisfy

$$E\left[f(x^k) - f(x^*)\right] \leq \frac{\psi^0}{\eta(k+1)},$$

where $x^k = \frac{1}{k+1} \sum_{i=0}^{k} x^i$.

- IntDIANA with the GD estimator requires that $\eta_k = \eta \leq \frac{1}{(L+\frac{32}{3} \eta)^2}$.
- IntDIANA with the L-SVRG estimator requires that $\eta_k = \eta \leq \frac{1}{(L+2\varepsilon / n)^2}$.

The above theorem establishes linear convergence of two versions of IntDIANA in the strongly convex regime, and sublinear convergence in the convex regime.

5.3. Compression efficiency

If $\mu > 0$, for IntDIANA with adaptive $\alpha_k$ and either GD or L-SVRG estimator, both $\|h_k^i - \nabla f_i(x^*)\|^2$ and $\|x^k - x^k\|^2$ converge to 0 linearly at the same rate, while $g_i^k \rightarrow \nabla f_i(x^*)$. Thus, the largest integer to transmit is $\|\alpha_k(g_i^k - h_i^k)\|_\infty \sim \frac{\|h_i^k\|_\infty}{\sqrt{x^k}}$ is hopefully upper bounded.

6. Experiments

We numerically evaluate our proposed algorithm IntSGD on several popular convex and non-convex benchmarks, including logistic regression, training convolutional networks for image classification, and training Variational Autoencoders (VAEs). We compare IntSGD (Algorithm 1) with the following baselines: (1) SGD without any compression; (2) SGD with natural compression (NatSGD) (Horváth et al., 2019), which also works in the SwitchML framework and has convergence theory; (3) SGD with the heuristic-based integer quantization (HintSGD) in (Sapio et al., 2021), which does not have convergence guarantee. Communication cost of IntSGD can be discussed in two possible cases of the SwitchML framework:

- Primitive integer dtypes (int8, int16, etc). Sometimes, bit-level operations could be expensive and make the algorithm quite inefficient. For example, in C++, bool has size of 1 byte and its vector<bool> should be avoided at all costs. It is worth noting that IntSGD works with the primitive integer data types while NatSGD does not, where bit-level operations may result in the long compression/decompression time of NatSGD.

- Bit-level operations. If the bit-level operations are allowed, the cost of IntSGD to communicate a compressed vector is its length times the number of bits to represent the largest integer in that vector. The cost of NatSGD to transmit a compressed vector is 9 bits times its length (Horváth et al., 2019) while that of SGD is 32 bits times its length.

For fair comparison, we only consider the second case in our experiments, where the transmitted megabytes (MB) of different algorithms are compared.

6.1. Logistic regression

Extensive experimentation with IntSGD and several variants of IntDIANA against several benchmarks on logistic regression problems can be found in Appendix A.

6.2. Deep learning experiments

Setup: We evaluate the proposed IntSGD and compare it with the baseline algorithms SGD, NatSGD and HintSGD on both image classification and generative modeling tasks with deep neural nets, including training ResNet-18, ResNet-50 (He et al., 2016) on the Cifar10 dataset and training VAE (Kingma & Welling, 2013), $\beta$-TCVAE (Chen et al., 2018) on MNIST, FashionMNIST and CelebA datasets. All the algorithms are run with 4 random seeds and 8 local workers. No momentum or weight decay is used. ResNet-18, ResNet-50 and VAE use the batch normalization while the $\beta$-TCVAE does not. We tune the step-size in $\{0.0001, 0.001, 0.01, 0.1, 1\}$. We use permutations over the empirical data instead of uniformly sampling functions, which a standard trick to make SGD faster (Bottou, 2012; Mishchenko et al., 2020). The algorithms and architectures are implemented in PyTorch (Paszke et al., 2017). We consider the adaptive $\alpha_k$ with a constant factor $\alpha_0 \geq 1$ and a safeguard $\sigma_Q > 0$:

$$\alpha_k = \frac{\alpha_0 \eta_k \sqrt{Q}}{\sqrt{2 \alpha_0 \eta_k + \eta_k^2 \sigma_Q^2}},$$

It is worth noting that any $\alpha_0 \geq 1$ never hurts the convergence results in Section 3. Instead, larger $\alpha_0$ trades compression efficiency for faster convergence. We set $\beta = 0$, $\sigma_Q = 10^{-8}$ in all our experiments where they are not mentioned explicitly. $\alpha_0$ is selected according to Table 1.

Results:

Comparison with baselines: As shown in Figure 1 and Figure 2, IntSGD attains the same training loss or testing accuracy with notably fewer transmitted megabytes compared to SGD or NatSGD on both image classification and generative modeling with deep neural nets. Besides, our IntSGD almost recovers the results of HintSGD (Sapio et al., 2021). However, the proposed IntSGD has low convergence guarantees on a wide spectrum of problems while HintSGD is purely heuristic and needs costly profiling.

---

2Our code: https://github.com/bokunwang1/intsgd
Effect of moving average: The safeguard $\sigma_Q^2 = 10^{-8}$ can avoid the exception value. However, according to the last row in Figure 1, the value of the adaptive $\alpha_k$ could still oscillate a lot. To tackle this problem, we can utilize the technique of moving average ($\beta > 0$) introduced in Section 4.4. By observing Figure 1, we can conclude that moving average is effective to smooth the value of $\alpha_k$ without hurting the performance.

Table 1: The optimal $\alpha_0$ of the adaptive $\alpha_k$ and the optimal constant $\alpha_k$ for various architectures/datasets.

| Architecture | Dataset | Optimal $\alpha_0$ | Optimal constant $\alpha_k$ |
|--------------|---------|--------------------|-----------------------------|
| ResNet-18    | Cifar10 | 1                  | $2^{12}$                    |
| ResNet-50    | Cifar10 | 1                  | $2^{11}$                    |
| VAE          | MNIST   | 1                  | $2^{-2}$                    |
| VAE          | FMNIST  | 1                  | $2^{-2}$                    |
| $\beta$-TCVAE | CelebA  | 2                  | $2^9$                       |

Necessity of adaptivity: In Section 3, we theoretically demonstrate that the constant $\alpha_k$ (Section 4.2) and the adaptive one (Section 4.5, with a safeguard or not) share the same convergence result. However, one may still wonder whether the adaptive $\alpha_k$ has some advantage over the constant one in practice? Here we provide a confirmative answer. It is observed that IntSGD with a proper constant $\alpha_k = \alpha$ can lead to the performance comparable to the adaptive $\alpha_k$ (the optimal constant $\alpha$’s we found by grid search are listed in Table 1). Here “optimal” is in the sense of that $\alpha/2$ leads to a worse final loss and $2\alpha$ leads to almost no improvement. As shown in Table 1, the value of optimal constant $\alpha$ could lie in an extremely large search space on different architectures/datasets. Thus, it is quite expensive to do grid search for finding the optimal constant $\alpha$ without any prior knowledge.

On the contrary, the adaptive estimate can automatically adapt to the optimal value of $\alpha$ without the expensive grid search. Note that the safeguard $\sigma_Q^2$ can be any small enough positive number to avoid exception value and does not need to be tuned. Indeed, we may need to slightly adjust the constant factor $\alpha_0$. However, its optimal value is quite stable across different architectures/datasets and can be found nearly at no cost, even if the optimal constant $\alpha_k$’s are quite divergent.

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

Allen-Zhu, Z. and Hazan, E. Variance reduction for faster
non-convex optimization. In The 33th International Conference on Machine Learning, pp. 699–707, 2016.

Bernstein, J., Wang, Y.-X., Azizzadenesheli, K., and Anandkumar, A. SignSGD: Compressed optimisation for non-convex problems. In Dy, J. and Krause, A. (eds.), Proceedings of the 35th International Conference on Machine Learning, volume 80 of Proceedings of Machine Learning Research, pp. 560–569, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Beznosikov, A., Horváth, S., Richtárik, P., and Safaryan, M. On biased compression for distributed learning. arXiv:2002.12410, 2020.

Bottou, L. Stochastic gradient descent tricks. In Neural networks: Tricks of the trade, pp. 421–436. Springer, 2012.

Chen, R. T. Q., Li, X., Grosse, R., and Duvenaud, D. Isolating sources of disentanglement in VAEs. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 2615–2625, 2018.

Dalcín, L., Paz, R., and Storti, M. MPI for Python. Journal of Parallel and Distributed Computing, 65(9):1108–1115, 2005.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

Dutta, A., Bergou, E. H., Abdelmoniem, A. M., Ho, C.-Y., Sahu, A. N., Canini, M., and Kalnis, P. On the discrepancy between the theoretical analysis and practical implementations of compressed communication for distributed deep learning. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pp. 3817–3824, 2020.

Gower, R. M., Loizou, N., Qian, X., Sailanbayev, A., Shulgin, E., and Richtárik, P. SGD: General analysis and improved rates. In Chaudhuri, K. and Salakhutdinov, R. (eds.), Proceedings of the 36th International Conference on Machine Learning, volume 97 of Proceedings of Machine Learning Research, pp. 5200–5209, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

Gower, R. M., Schmidt, M., Bach, F., and Richtárik, P. Variance-reduced methods for machine learning. Proceedings of the IEEE, 108(11):1968–1983, 2020.

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

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015.

Horváth, S., Ho, C.-Y., Horváth, v., Sahu, A. N., Canini, M., and Richtárik, P. Natural compression for distributed deep learning. arXiv preprint arXiv:1905.10988, 2019.

Horváth, S., Kovalev, D., Mishchenko, K., Stich, S., and Richtárik, P. Stochastic distributed learning with gradient quantization and variance reduction. arXiv preprint arXiv:1904.05115, 2019.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700–4708, 2017.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. Advances in neural information processing systems, 26:315–323, 2013.

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, 2019.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

Kovalev, D., Horváth, S., and Richtárik, P. Don’t jump through hoops and remove those loops: SVRG and Katyusha are better without the outer loop. In Algorithmic Learning Theory, pp. 451–467. PMLR, 2020.

Mishchenko, K., Gorbunov, E., Takáč, M., and Richtárik, P. Distributed learning with compressed gradient differences. arXiv preprint arXiv:1901.09269, 2019.

Mishchenko, K., Khaled, A., and Richtárik, P. Random Reshuffling: Simple analysis with vast improvements. Advances in Neural Information Processing Systems, 33, 2020.

Nesterov, Y. Introductory lectures on convex optimization: A basic course, volume 87. Springer Science & Business Media, 2013.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. 2017.
Sapio, A., Canini, M., Ho, C.-Y., Nelson, J., Kalnis, P., Kim, C., Krishnamurthy, A., Moshref, M., Ports, D. R. K., and Richtárik, P. Scaling distributed machine learning with in-network aggregation. *To appear in 18th USENIX Symposium on Networked Systems Design and Implementation*, 2021.

Stich, S. U., Cordonnier, J.-B., and Jaggi, M. Sparsified SGD with memory. In *Advances in Neural Information Processing Systems*, pp. 4447–4458, 2018.

Xu, H., Ho, C.-Y., Abdelmoniem, A. M., Dutta, A., Bergou, E. H., Karatsenidis, K., Canini, M., and Kalnis, P. Compressed communication for distributed deep learning: Survey and quantitative evaluation. Technical report, 2020.
IntSGD: Floatless Compression of Stochastic Gradients

Appendix

A. Experiments with Logistic Regression

Setup: We run the experiments on the $\ell_2$-regularized logistic regression problem with four datasets (a5a, mushrooms, w8a, real-sim) from the LibSVM repository\(^3\), where

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x)$$

and

$$f_i(x) = \frac{1}{m} \sum_{l=1}^{m} \log(1 + \exp(-A_{i,l}^T x b_{i,l})) + \frac{\lambda_2}{2} \|x\|^2,$$

and $x \in \mathbb{R}^d$, $\lambda_2$ is chosen proportionally to $\frac{1}{mn}$ and $A_{i,l} \in \mathbb{R}^d$, $b_{i,l} \in \{-1, 1\}$ are the feature and label of $l$-th data point on the $i$-th worker. The experiments are performed on a machine with 24 Intel(R) Xeon(R) Gold 6246 CPU @ 3.30GHz cores, where 12 cores are connected to a socket (there are two sockets in total). All experiments use 12 cpu cores and each core is utilized as a worker. The communications are implemented based on the MPI4PY library (Dalcín et al., 2005). The “optimum” $x^*$ is obtained by running GD with the whole data using one cpu core until there are 5000 iterations or $\|\nabla f(x)\|^2 \leq 10^{-30}$.

Table 2: Information of the experiments on $\ell_2$-regularized logistic regression.

| Dataset   | #Instances N | Dimension d | $\lambda_2$  |
|-----------|--------------|-------------|--------------|
| a5a       | 6414         | 123         | $5 \times 10^{-4}$ |
| mushrooms | 8124         | 112         | $6 \times 10^{-4}$ |
| w8a       | 49749        | 300         | $10^{-4}$     |
| real-sim  | 72309        | 20958       | $5 \times 10^{-5}$ |

The whole dataset is splitted according to its original indices into $n$ folds and each fold is assigned to a local worker, i.e., the data is heterogeneous. There are $m = \lfloor \frac{n}{n} \rfloor$ data points on each worker. For each dataset, we run each algorithm multiples times with 20 random seeds for each worker. For the stochastic algorithms, we randomly sample 5% of the local data as a minibatch (i.e., batch size $\tau = \lfloor \frac{m}{20} \rfloor$) to estimate the stochastic gradient $g_i^k$ on each worker. We set $p = \frac{r}{n}$ in VR-IntDIANA.

Apart from IntSGD, we also evaluate IntDIANA (Algorithm 3) with the GD or L-SVRG estimator (called IntDIANA and VR-IntDIANA, respectively) and compare them with the baselines L-SVRG (Kovalev et al., 2020), VR-DIANA with natural compression (VR-NatDiana) (Horváth et al., 2019) and the heuristic-based integer quantization (VR-HIntDIANA).

Results: First, we evaluate IntSGD and IntDIANA with different gradient estimators and various choices of $\alpha_k$ listed in Section 4. As shown in the first row of Figure 3, IntSGD with the full gradient (IntGD) and the adaptive $\alpha_k$ leads to extremely large integer value (up to $2^{30}$) to communicate, which is quite inefficient. Setting a proper safeguard $\sigma_Q > 0$ can solve this issue. However, unlike IntGD with adaptive $\alpha_k$, it only converges to a neighborhood of $x^*$ when the stepsize is not diminishing. By contrast, IntDIANA with L-SVRG estimator (VR-IntDIANA) is linearly convergent and efficient in terms of both compression and gradient oracles. Second, we compare VR-IntDIANA with the baselines L-SVRG, VR-NatDiana, and VR-HIntDIANA. It is demonstrated that the proposed VR-IntDIANA can reach the same objective gap by transmitting fewer bytes.

\(^3\)https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html
Figure 3: Experimental results on the $\ell_2$-regularized logistic regression problem. “Max integer to send” represents the largest transmitted integers in a communication round (including both worker to master and master to worker communications).

B. Proofs

B.1. Proofs for Lemma 1

Proof. Take $y = \alpha \circ x$ and let $p_y = y - \lfloor y \rfloor$, where $\lfloor y \rfloor$ is the coordinate-wise floor, and $p_y$ is the vector of probabilities in the definition of $\mathcal{I}(y)$. By definition it holds

$$\mathbb{E}[\mathcal{I}(y)] = p_y(\lfloor y \rfloor + 1) + (1 - p_y)\lceil y \rceil = p_y + \lfloor y \rfloor = y - \lfloor y \rfloor + \lceil y \rceil = y.$$ 

Plugging back $y = \alpha \circ x$, we obtain the first claim.

Similarly,

$$\|y - \mathcal{I}(y)\|_{\infty} = \max_{z \in \mathbb{R}} |y_j - \mathcal{I}(y_j)| = \max_{z \in \mathbb{R}} \max_{z \in \mathbb{R}} (z - \lfloor z \rfloor, \lceil z \rceil + 1 - z) = 1.$$ 

After substituting $y = \alpha \circ x$, it remains to mention

$$\left\| \frac{1}{\alpha} \circ \mathcal{I}(\alpha \circ x) - x \right\|_{\infty} \leq \|\mathcal{I}(\alpha \circ x) - \alpha \circ x\|_{\infty} \max_{j=1,\ldots,d} \frac{1}{\alpha_j}.$$ 

To obtain the last fact, notice that $\mathcal{I}(y) - \lfloor y \rfloor$ is a vector of Bernoulli random variables. Since the variance of any Bernoulli variable is bounded by $\frac{1}{4}$, we have

$$\mathbb{E}\left[\left\| \frac{1}{\alpha} \circ \mathcal{I}(\alpha \circ x) - x \right\|^2 \right] = \sum_{j=1}^{d} \frac{1}{\alpha_j^2} \mathbb{E}\left[ (\mathcal{I}(y_j) - y_j)^2 \right] \leq \sum_{j=1}^{d} \frac{1}{4\alpha_j^2}.$$ 

$\square$

B.2. Proof of Lemma 2

Proof. The last term in the expression that we want to prove is needed to be later used to compensate quantization errors. For this reason, let us save one $\|x^{k+1} - x^k\|^2$ for later when expanding $\|x^{k+1} - x^*\|^2$. Consider the IntSGD step $x^{k+1} - x^k = \eta_k \frac{1}{n} \sum_{i=1}^{n} Q(g_i^k)$, where $Q(g_i^k) = \frac{1}{\alpha_k} \circ \mathcal{I}(\alpha_k \circ g_i^k)$.

$$\|x^{k+1} - x^*\|^2 = \|x^k - x^*\|^2 + 2\langle x^{k+1} - x^k, x^k - x^* \rangle + \|x^{k+1} - x^k\|^2$$

$$= \|x^k - x^*\|^2 + 2\langle x^{k+1} - x^k, x^k - x^* \rangle + 2\|x^{k+1} - x^k\|^2 - \|x^{k+1} - x^k\|^2$$

$$= \|x^k - x^*\|^2 - 2\eta_k \frac{1}{n} \sum_{i=1}^{n} \langle Q(g_i^k), x^k - x^* \rangle + 2\eta_k \frac{1}{n} \sum_{i=1}^{n} Q(g_i^k) \|x^{k+1} - x^k\|^2.$$ 

(10)
Moreover, using the tower property of expectation, we can decompose the penultimate term in (10) as follows:

$$E_k \left[ -2 \frac{\eta_k}{n} \sum_{i=1}^{n} (Q(g_i^k), x^k - x^*) \right] = -2 \frac{\eta_k}{n} \sum_{i=1}^{n} E_k[g_i^k], x^k - x^* \leq -2 \eta_k (f(x^k) - f(x^*)).$$

By convexity, it is clear that the latter terms get cancelled when we plug this bound back into the first recursion.

Firstly, let us recur the bound in Lemma 2 from Proof.

**B.3. Proof of Lemma 3**

**Proof.** Firstly, let us recur the bound in Lemma 2 from $k$ to $0$:

$$E \left[ ||x^{k+1} - x^*||^2 \right] \leq ||x^0 - x^*||^2 - 2 \sum_{t=0}^{k} \eta_t E \left[ f(x^t) - f(x^*) \right] + 2 \sum_{t=0}^{k} \eta_t^2 E \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^t \right\|^2 \right]$$

$$+ \frac{1}{2n} \sum_{t=1}^{k} \sum_{j=1}^{d} \left[ \eta_t^2 \frac{\alpha_{t,j}^2}{\alpha_{t,j}} - \sum_{l=0}^{t} E \left[ ||x^{t+1} - x^t||^2 \right] \right].$$

Note that in the bound we do not have $\alpha_{0,j}$ for any $j$ as we assume that the first communication is done without compression. Assumption 1 implies for the quantization error

$$\sum_{j=1}^{d} E_k \left[ ||Q(g_i^k) - g_i^k||^2 \right] = \sum_{i=1}^{n} E_k \left[ \left\| \frac{1}{\alpha_k} \circ \text{Int} (\alpha_k \circ g_i^k) - g_i^k \right\|^2 \right] \leq \frac{n}{4} \sum_{j=1}^{d} \frac{1}{\alpha_{k,j}}.$$

Dividing both sides by $n^2$ and plugging it into (10), we obtain the desired decomposition into SGD and quantization terms.

It is clear that the latter terms get cancelled when we plug this bound back into the first recursion. 

$\square$
B.4. Proof of Theorem 1

Proof. Most of the derivation has been already obtained in Lemma 3 and we only need to take care of the SGD terms. To do that, we decompose the gradient error into expectation and variance:

\[
E_k \left[ \frac{1}{n} \sum_{i=1}^{n} g_i^k \right] = E_k \left[ \frac{1}{n} \sum_{i=1}^{n} E_k [g_i^k] \right]^2 + E_k \left[ \frac{1}{n} \sum_{i=1}^{n} (g_i^k - E_k [g_i^k]) \right]^2 \\
= \left\| \frac{1}{n} \sum_{i=1}^{n} E_k [g_i^k] \right\|^2 + \frac{1}{n^2} \sum_{i=1}^{n} E_k \left[ \left\| g_i^k - E_k [g_i^k] \right\|^2 \right] \\
\leq G^2 + \frac{\sigma^2}{n}.
\]

Thus, we arrive at the following corollary of Lemma 3:

\[
0 \leq E \left[ \|x^{k+1} - x^*\|^2 \right] \leq \|x^0 - x^*\|^2 - 2 \sum_{t=0}^{k} \eta_t E \left[ f(x^t) - f(x^*) \right] + 2 \sum_{t=0}^{k} \eta_t^2 \left( 4G^2 + \frac{\sigma^2}{n} + \frac{\sigma_Q^2}{4n} \right).
\]

Furthermore, by convexity of \( f \) we have

\[
f(x^k) - f(x^*) \leq \frac{1}{\sum_{t=0}^{k} \eta_t} \sum_{t=0}^{k} \eta_t (f(x^t) - f(x^*)).
\]

Plugging it back, rearranging the terms and dropping \( E \left[ \|x^{k+1} - x^*\|^2 \right] \) gives the result.

\( \Box \)

B.5. Proof of Proposition 1

Proof. Fix any \( i \). By Young’s inequality and independence of \( \xi_1^k, \ldots, \xi_n^k \) we have

\[
E \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x^k; \xi_i^k) \right\|^2 \right] = E \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x^k; \xi_i^k) \right\|^2 \right] \\
\leq 2E \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x^k; \xi_i^k) \right\|^2 \right] + 2E \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} (\nabla f_i(x^k; \xi_i^k) - \nabla f_i(x^*; \xi_i^k)) \right\|^2 \right] \\
= 2 \frac{n^2}{n} \sum_{i=1}^{n} E \left[ \| \nabla f_i(x^k; \xi_i^k) \|^2 \right] + 2E \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} (\nabla f_i(x^k; \xi_i^k) - \nabla f_i(x^*; \xi_i^k)) \right\|^2 \right].
\]

Substituting the definition of \( \sigma^2 \) and applying Jensen’s inequality, we derive

\[
E \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right] \leq \sigma^2 + \frac{2}{n} \sum_{i=1}^{n} E \left[ \| \nabla f_i(x^k; \xi_i^k) - \nabla f_i(x^*; \xi_i^k) \|^2 \right].
\]

By our assumption, \( f_i(\cdot; \xi) \) is convex and has \( L_i \)-Lipschitz gradient, so we can use Equation (2.1.7) in Theorem 2.1.5 in (Nesterov, 2013):

\[
2E \left[ \| \nabla f_i(x^k; \xi_i^k) - \nabla f_i(x^*; \xi_i^k) \|^2 \right] \leq 4L_i E \left[ f_i(x^k; \xi_i^k) - f_i(x^*; \xi_i^k) - (\nabla f_i(x^*; \xi_i^k), x^k - x^*) \right] \\
= 4L_i E \left[ f_i(x^k) - f_i(x^*) - (\nabla f_i(x^*), x^k - x^*) \right] \\
\leq L E \left[ f_i(x^k) - f_i(x^*) - (\nabla f_i(x^*), x^k - x^*) \right].
\]

Taking the average over \( i = 1,\ldots, n \) and noticing \( \sum_{i=1}^{n} \nabla f_i(x^*) = 0 \) yields

\[
E \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right] \leq \sigma^2 \frac{2}{n} + \frac{L n}{n} \sum_{i=1}^{n} E \left[ f_i(x^k) - f_i(x^*) - (\nabla f_i(x^*), x^k - x^*) \right] \\
= \sigma^2 \frac{2}{n} + L E \left[ f(x^k) - f(x^*) \right],
\]

which is exactly our claim.

\( \Box \)
B.6. Proof of Theorem 2

Proof. The proof is almost identical to that of Theorem 1, but now we directly use Assumption 3 and plug it in inside Lemma 3 to get

\[
E \left[ \left\| x^{k+1} - x^* \right\|^2 \right] \leq \| x^0 - x^* \|^2 - 2 \sum_{t=0}^{k} \eta_t E \left[ f(x^t) - f(x^*) \right] + 2 \sum_{t=0}^{k} \eta_t^2 E \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^t \right\|^2 \right] + \frac{\sigma^2}{2n} \sum_{t=1}^{k} \eta_t^2
\]

\[
\leq \| x^0 - x^* \|^2 - \sum_{t=0}^{k} 2\eta_t (1 - \eta_t L) E \left[ f(x^t) - f(x^*) \right] + 2 \left( \frac{\sigma^2}{n} + \frac{\sigma^2 Q}{4n} \right) \sum_{t=1}^{k} \eta_t^2
\]

\[
\leq \| x^0 - x^* \|^2 - \sum_{t=0}^{k} \eta_t E \left[ f(x^t) - f(x^*) \right] + 2 \left( \frac{\sigma^2}{n} + \frac{\sigma^2 Q}{4n} \right) \sum_{t=1}^{k} \eta_t^2.
\]

Rearranging this inequality yields

\[
\sum_{t=0}^{k} \eta_t E \left[ f(x^t) - f(x^*) \right] \leq \| x^0 - x^* \|^2 - \| x^{k+1} - x^* \|^2 + 2 \left( \frac{\sigma^2}{n} + \frac{\sigma^2 Q}{4n} \right) \sum_{t=1}^{k} \eta_t^2
\]

\[
\leq \| x^0 - x^* \|^2 + 2 \left( \frac{\sigma^2}{n} + \frac{\sigma^2 Q}{4n} \right).
\]

To finish the proof, it remains to upper bound \( f(\hat{x}^k) \) using convexity the same way as it was done in Equation (12). 

B.7. Proof of Theorem 3

Proof. By L-smoothness of \( f \) we have

\[
E_k[f(x^{k+1})] \leq f(x^k) + E_k[\langle \nabla f(x^k), x^{k+1} - x^k \rangle] + \frac{L}{2} E_k[\| x^{k+1} - x^k \|^2]
\]

\[
= f(x^k) - \frac{\eta_k}{n} \sum_{i=1}^{n} E_k[\langle \nabla f(x^k), Q(g_i^k) \rangle] + \frac{L}{2} E_k[\| x^{k+1} - x^k \|^2]
\]

\[
\leq f(x^k) - \eta_k \| \nabla f(x^k) \|^2 + \frac{L}{2} E_k[\| x^{k+1} - x^k \|^2]
\]

\[
= f(x^k) - \eta_k \| \nabla f(x^k) \|^2 + L E_k[\| x^{k+1} - x^k \|^2] - \frac{L}{2} E_k[\| x^{k+1} - x^k \|^2]
\]

\[
= f(x^k) - \eta_k \| \nabla f(x^k) \|^2 + \eta_k^2 L E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} Q(g_i^k) \right\|^2 \right] - \frac{L}{2} E_k[\| x^{k+1} - x^k \|^2].
\]

Similarly to Lemma 2, we get a decomposition into SGD and quantization errors:

\[
E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} Q(g_i^k) \right\|^2 \right] = E_k \left[ E_Q \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} Q(g_i^k) \right\|^2 \right] \right]
\]

\[
= E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right] + \frac{1}{n^2} \sum_{i=1}^{n} E_k[\| Q(g_i^k) - g_i^k \|^2]
\]

\[
\leq E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right] + \frac{1}{4n} \sum_{j=1}^{d} \frac{1}{\alpha_{k,j}}.
\]
IntSGD: Floatless Compression of Stochastic Gradients

We proceed with the two terms separately. To begin with, we further decompose the SGD error into its expectation and variance:

\[
E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right] = \left\| \frac{1}{n} \sum_{i=1}^{n} E_k [g_i^k] \right\|^2 + E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} (g_i^k - E_k [g_i^k]) \right\|^2 \right]
\]

\[
= \|\nabla f(x^k)\|^2 + \frac{1}{n^2} \sum_{i=1}^{n} E_k \left[ \|g_i^k - \nabla f_i(x^k)\|^2 \right]
\]

\[
\leq \|\nabla f(x^k)\|^2 + \frac{\sigma^2}{n}. \quad (8)
\]

Moving on, we plug it back into the upper bound \(E_k [f(x^{k+1})]\). Assuming \(\eta_k \leq \frac{1}{2L}\), we get

\[
E_k [f(x^{k+1})] \leq f(x^k) - \eta_k (1 - \eta_k L)\|\nabla f(x^k)\|^2 + \frac{\eta_k^2 L \sigma^2}{n} + \frac{L}{4n} \sum_{j=1}^{d} \frac{\eta_k^2}{\alpha_{k,j}^2} - \frac{L}{2} E_k [\|x^{k+1} - x^k\|^2]
\]

\[
\leq f(x^k) - \frac{\eta_k}{2} \|\nabla f(x^k)\|^2 + \frac{\eta_k^2 L \sigma^2}{n} + \frac{L}{4n} \sum_{j=1}^{d} \frac{\eta_k^2}{\alpha_{k,j}^2} - \frac{L}{2} E_k [\|x^{k+1} - x^k\|^2].
\]

Finally, reusing Equation (11) produces the bound

\[
E [f(x^{k+1})] \leq f(x^0) - \sum_{t=0}^{k} \frac{\eta_t}{2} E [\|\nabla f(x^t)\|^2] + \frac{\sigma^2}{n} \sum_{t=0}^{k} \eta_t^2 L + \frac{\sigma_Q^2}{4n} \sum_{t=0}^{k} \eta_t^2 L.
\]

Notice that by Assumption 4, \(f^{\inf} \leq f(x^{k+1})\), so we have

\[
\frac{1}{\sum_{t=0}^{k} \eta_t} \sum_{t=0}^{k} \eta_t E [\|\nabla f(x^t)\|^2] \leq 2 \frac{f(x^0) - f^{\inf} + \left(\frac{\sigma^2}{n} + \frac{\sigma_Q^2}{4n}\right) \sum_{t=0}^{k} \eta_t^2 L}{\sum_{t=0}^{k} \eta_t}.
\]

The left-hand side is equal to \(E [\|\nabla f(\hat{x}^k)\|^2]\) by definition of \(\hat{x}^k\), and we conclude the proof.

**B.8. Proof of Corollary 2**

**Proof.** For the first part, we have

\[
\frac{\|x^0 - x^*\|^2 + 2 \left( G^2 + \frac{\sigma^2}{n} + \frac{\sigma_Q^2}{4n} \right) \sum_{t=0}^{k} \eta_t^2}{2 \sum_{t=0}^{k} \eta_t} = O \left( \frac{1}{\sum_{t=0}^{k} \eta_t} \right) = O \left( \frac{G + \sigma + \sigma_Q}{\sqrt{k}} \right).
\]

The other complexities follow similarly.\[\square\]
C. Proofs for Section 4: New Integer Compression Operators

C.1. Proof of Proposition 2

Proof. Indeed, we only need to plug in the values of $\alpha_{k,j}$:

$$\sum_{j=1}^{d} \mathbb{E} \left[ \frac{\eta_k^2}{\alpha_{k,j}^2} \right] = 2n \mathbb{E} \left[ \|x^k - x^{k-1}\|^2 \right] \overset{\beta = 0}{=} 2n (1 - \beta) \sum_{t=0}^{k-1} \beta^t \mathbb{E} \left[ \|x^{k-t} - x^{k-t-1}\|^2 \right].$$

\[\square\]

C.2. Proof of Proposition 3

Proof. If $\alpha_k = \frac{\sqrt{d}}{\sqrt{nG}} (1, \ldots, 1)^\top$, we have

$$\sum_{j=1}^{d} \mathbb{E} \left[ \frac{\eta_k^2}{\alpha_{k,j}^2} \right] = \sum_{j=1}^{d} \frac{\eta_k^2 nG^2}{d} = \eta_k^2 nG^2,$nG^2$$

which corresponds to $\beta = 1$ and $\sigma_Q = \sqrt{nG}$. The other choice follows the same way.

\[\square\]

C.3. Proof of Proposition 4

Proof. Since the $l$-th block has $d_l$ coordinates, we get

$$\sum_{j=1}^{d} \mathbb{E} \left[ \frac{\eta_k^2}{\alpha_{k,j}^2} \right] = \sum_{l=1}^{B} d_l \mathbb{E} \left[ \frac{\eta_k^2}{\alpha_{k,(l)}^2} \right] = 2n \sum_{l=1}^{B} \mathbb{E} \left[ \|x^k_l - x^{k-1}_l\|^2 \right] = 2n \mathbb{E} \left[ \|x^k - x^{k-1}\|^2 \right].$$

\[\square\]

C.4. Proof of Proposition 5

Proof. We have already seen that if one uses $\beta = 0$, then the bound holds. If $\beta > 0$, then one can mention that

$$\sum_{j=1}^{d} \mathbb{E} \left[ \frac{\eta_k^2}{\alpha_{k,j}^2} \right] = 2n \mathbb{E} [r_k] = 2n (1 - \beta) \sum_{t=0}^{k-1} \beta^t \|x^{k-t} - x^{k-t-1}\|^2.$$

\[\square\]

C.5. Proof of Proposition 6

Proof. By definition of $\alpha_k$

$$\sum_{j=1}^{d} \mathbb{E} \left[ \frac{\eta_k^2}{\alpha_{k,j}^2} \right] = \eta_k^2 \sigma_Q^2 + 2n \mathbb{E} [r_k] = \eta_k^2 \sigma_Q^2 + 2n (1 - \beta) \sum_{t=0}^{k-1} \beta^t \|x^{k-t} - x^{k-t-1}\|^2.$$
D. Proofs for IntDIANA

Assumption 5. $f_{il}(x)$ has $L_{il}$-Lipschitz gradient. We define $\mathcal{L} \overset{\text{def}}{=} 4 \max_{i \in [n]} \max_{l \in [m]} L_{il}$.

Proposition 7. Suppose that Assumption 5 holds. Then, we have the following for IntDIANA and any $x \in \mathbb{R}^d$:

\[
\frac{1}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \|\nabla f_{il}(x) - \nabla f_{il}(x^*)\|^2 \leq \frac{\mathcal{L}}{2} (f(x) - f(x^*))
\]  

(13)

Proof. Based on Assumption 5 and Theorem 2.1.5 (Nesterov, 2013), we have:

\[
\|\nabla f_{il}(x) - \nabla f_{il}(x^*)\|^2 \leq 2L_{il} (f_{il}(x) - f_{il}(x^*) - \langle \nabla f_{il}(x^*), x - x^* \rangle)
\]

Thus, double averaging leads to:

\[
\frac{1}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \|\nabla f_{il}(x) - \nabla f_{il}(x^*)\|^2 \\
\leq \frac{2}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} L_{il} (f_{il}(x) - f_{il}(x^*) - \langle \nabla f_{il}(x^*), x - x^* \rangle)
\]

\[
\leq 2 \max_i \max_l L_{il} \left( f(x) - f(x^*) - \langle \frac{1}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \nabla f_{il}(x^*), x - x^* \rangle \right)
\]

Considering that $\frac{1}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \nabla f_{il}(x^*) = 0$ and defining $4 \max_{i \in [n]} \max_{l \in [m]} L_{il}$ leads to the claim in the proposition.

Lemma 4. For IntDIANA (Algorithm 3) and $g^k \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} (h_i^k + Q(g_i^k))$, we have $\mathbb{E}_k [g^k] = \nabla f(x^k)$ and:

\[
\mathbb{E}_k [\|g^k\|^2] \leq \frac{1}{4n} \sum_{j=1}^{d} \frac{1}{\alpha_{k,j}^2} + \mathbb{E}_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right].
\]

(14)

Proof. By definition, $g^k = \frac{1}{n} \sum_{i=1}^{n} (h_i^k + Q(g_i^k))$, so

\[
\mathbb{E}_k [g^k] = \frac{1}{n} \sum_{i=1}^{n} h_i^k + \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_k [Q(g_i^k)] \overset{\text{(3)}}{=} \frac{1}{n} \sum_{i=1}^{n} h_i^k + \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_k [g_i^k] - \frac{1}{n} \sum_{i=1}^{n} h_i^k = \nabla f(x^k).
\]

Thus, we have shown that $g^k$ is an unbiased estimate of $\nabla f(x^k)$. Let us proceed with the second moment of $g^k$:

\[
\mathbb{E}_k [\|g^k\|^2] = \mathbb{E}_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} \left( \frac{1}{\alpha_k} \circ \mathbb{I}nt(\alpha_k \circ (g_i^k - h_i^k)) - (g_i^k - h_i^k) + g_i^k \right) \right\|^2 \right] \\
\overset{\text{(3)}}{=} \mathbb{E}_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} \left( \frac{1}{\alpha_k} \circ \mathbb{I}nt(\alpha_k \circ (g_i^k - h_i^k)) - (g_i^k - h_i^k) \right) \right\|^2 \right]
+ \mathbb{E}_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right]
\]

\[
= \frac{1}{n^2} \sum_{i=1}^{n} \mathbb{E}_k \left[ \left\| \frac{1}{\alpha_k} \circ \mathbb{I}nt(\alpha_k \circ (g_i^k - h_i^k)) - (g_i^k - h_i^k) \right\|^2 \right]
+ \mathbb{E}_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right]
\]

\[
\overset{\text{(4)}}{\leq} \frac{1}{4n} \sum_{j=1}^{d} \frac{1}{\alpha_{k,j}^2} + \mathbb{E}_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right].
\]
Lemma 5. If L-SVRG estimator $g^k_i = \nabla f_u(x^k; \xi^k) - \nabla f_u(w^k; \xi^k) + u^k$ is used in IntDIANA, we have $E_k [g^k_i] = \nabla f_i(x^k)$ and

$$
E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g^k_i \right\|^2 \right] \leq \left( 2L + \frac{L}{m} \right) (f(x^k) - f(x^*)) + \frac{2}{n} \sigma_1^k,
$$

(15)

$$
E_k [\sigma_1^{k+1}] \leq (1 - p) \sigma_1^k + \frac{pL}{2} (f(x^k) - f(x^*)) ,
$$

(16)

where $\sigma_1^k = \frac{1}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \|\nabla f_u(w^k_l) - \nabla f_u(x^k)\|^2$.

Proof. Recall that $E [\|X - E[X]\|^2] \leq E [\|X\|^2]$ for any random variable $X$. For the L-SVRG estimator $g^k_i = \nabla f_u(x^k; \xi^k) - \nabla f_u(w^k_t; \xi^k) + u^k$, we have:

$$
E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g^k_i \right\|^2 \right] = \left\| \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x^k) \right\|^2 + E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} (g^k_i - \nabla f_i(x^k)) \right\|^2 \right]
$$

\[ \leq 2L (f(x^k) - f(x^*)) + \frac{1}{n^2} \sum_{i=1}^{n} \frac{1}{m} \sum_{l=1}^{m} \|\nabla f_u(x^k) - \nabla f_u(w^k_l)\| \left\| \nabla f_i(x^k) - \nabla f_i(x^*) \right\|^2 \]

\[ \leq 2L (f(x^k) - f(x^*)) + \frac{1}{n^2} \sum_{i=1}^{n} \frac{1}{m} \sum_{l=1}^{m} \|\nabla f_u(x^k) - \nabla f_u(w^k_l)\|^2 \]

\[ \leq 2L (f(x^k) - f(x^*)) + \frac{2}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \|\nabla f_u(x^k) - \nabla f_u(x^*)\|^2 \]

\[ \leq (13) \left( 2L + \frac{L}{n} \right) (f(x^k) - f(x^*)) + \frac{2}{n} \sigma_1^k ,
\]

where $\sigma_1^k = \frac{1}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \|\nabla f_u(w^k_l) - \nabla f_u(x^k)\|^2$. Based on the update of control sequence in L-SVRG, we have:

$$
E_k [\sigma_1^{k+1}] = \frac{1 - p}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \|\nabla f_u(w^k_l) - \nabla f_u(x^*)\|^2 + \frac{p}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \|\nabla f_u(x^k) - \nabla f_u(x^*)\|^2
$$

\[ \leq (1 - p) \sigma_1^k + \frac{pL}{2} (f(x^k) - f(x^*)) .
\]

Lemma 6. Define $\sigma_2^k \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \|h^k_i - \nabla f_i(x^*)\|^2$. For IntDIANA algorithm, we have:

$$
E_k [\sigma_2^{k+1}] \leq \frac{1}{n} \sum_{i=1}^{n} E_k \left[ \|g^k_i - \nabla f_i(x^*)\|^2 \right] + \sum_{j=1}^{d} \frac{1}{\alpha_j^k} ,
$$

(17)

For the full gradient, we have $ \frac{1}{n} \sum_{i=1}^{n} E_k \left[ \|g^k_i - \nabla f_i(x^*)\|^2 \right] \leq \frac{\xi}{2} (f(x^k) - f(x^*))$. For the L-SVRG estimator, we have $ \frac{1}{n} \sum_{i=1}^{n} E_k \left[ \|g^k_i - \nabla f_i(x^*)\|^2 \right] \leq 4\sigma_1^k + 3L(f(x^k) - f(x^*))$.

Proof. We define $\sigma_2^k \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \|h^k_i - \nabla f_i(x^*)\|^2$. Consider the step $h^{k+1}_i = h^k_i + Q(g^k_i)$, where $Q(g^k_i) = \frac{1}{\alpha_k} \circ Int(\alpha_k \circ (g^k_i - h^k_i))$. Note that $E_k \left[ \|g^k_i - h^k_i\|^2 + \|g^k_i - 2\nabla f_i(x^*) + h^k_i\|^2 \right] = E_k \left[ \|g^k_i - \nabla f_i(x^*)\|^2 - \|h^k_i - \nabla f_i(x^*)\|^2 \right]$.  


which explains the last equality below:

$$E_k [\alpha_{2k+1}^{k+1}] = E_k \left[ \frac{1}{n} \sum_{i=1}^{n} \|h_i^k - \nabla f_i(x^*) + Q(g_i^k)\|^2 \right]$$

$$= \sigma_k^k + \frac{1}{n} \sum_{i=1}^{n} E_k \left[ \left\| \frac{1}{\alpha_k} \circ \text{Int} \left( \alpha_k \circ (g_i^k - h_i^k) \right) \right\|^2 \right] + \frac{2}{n} \sum_{i=1}^{n} E_k \left[ \langle Q(g_i^k), h_i^k - \nabla f_i(x^*) \rangle \right]$$

$$\leq \sigma_k^k + \frac{1}{n} \sum_{i=1}^{n} E_k \left[ \langle g_i^k - h_i^k, g_i^k - 2\nabla f_i(x^*) + h_i^k \rangle \right] + \frac{d}{\alpha_k^2}$$

$$= \frac{1}{n} \sum_{i=1}^{n} E_k \left[ \|g_i^k - \nabla f_i(x^*)\|^2 \right] + \frac{d}{\alpha_k^2}.$$

For the full gradient $g_i^k = \nabla f_i(x^k)$, we have:

$$\frac{1}{n} \sum_{i=1}^{n} E_k \left[ \|g_i^k - \nabla f_i(x^*)\|^2 \right] = \frac{1}{n} \sum_{i=1}^{n} E_k \left[ \|\nabla f_i(x^k) - \nabla f_i(x^*)\|^2 \right] \leq \frac{\mathcal{L}}{2} (f(x^k) - f(x^*))$$

For the L-SVRG estimator, we have by Young’s inequality:

$$\frac{1}{n} \sum_{i=1}^{n} E_k \left[ \|g_i^k - \nabla f_i(x^*)\|^2 \right] \leq \frac{2}{n} \sum_{i=1}^{n} E_k \left[ \|g_i^k - \nabla f_i(x^k)\|^2 \right] + \frac{2}{n} \sum_{i=1}^{n} \|\nabla f_i(x^k) - \nabla f_i(x^*)\|^2$$

$$\leq \frac{2}{mn} \sum_{i=1}^{m} \sum_{l=1}^{n} \|\nabla f_{il}(x^k) - \nabla f_{il}(w_i^k)\|^2 + \mathcal{L}(f(x^k) - f(x^*))$$

$$\leq \frac{4}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \|\nabla f_{il}(x^k) - \nabla f_{il}(x^*)\|^2 + \frac{4}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \|\nabla f_{il}(w_i^k) - \nabla f_{il}(x^*)\|^2 + \mathcal{L}(f(x^k) - f(x^*))$$

$$\leq 4\sigma_k^k + 3\mathcal{L}(f(x^k) - f(x^*)).$$

\[\square\]

**Lemma 7.** Suppose that Assumption 5 holds. Besides, we assume that $f(\cdot)$ is $\mu$-strongly convex ($\mu \geq 0$). For IntDIANA with adaptive $\alpha_k = \frac{\eta_k\sqrt{d}}{\sqrt{n\|x^k-x^k-1\|^2}}$ and GD gradient estimator, we have:

$$E_k \left[ \|x^{k+1} - x^*\|^2 \right] + E_k \left[ \|x^{k+1} - x^k\|^2 \right]$$

$$\leq (1 - \eta_k\mu)\|x^k - x^*\|^2 + \frac{1}{2}\|x^k - x^{k-1}\|^2 - 2\eta_k(1 - 2\eta_kL)(f(x^k) - f(x^*)),$$

$$E_k [\alpha_{2k+1}^{k+1}] \leq \frac{\mathcal{L}}{2} (f(x^k) - f(x^*)) + n\|x^k - x^{k-1}\|^2.$$
Proof. By $\mu$-strong convexity, we have:

$$
E_k \left[ -2\frac{n}{n} \sum_{i=1}^{n} (Q(g_i^k), x^k - x^*) \right]^{(3)} = -2\frac{n}{n} \sum_{i=1}^{n} \langle E_k g_i^k, x^k - x^* \rangle
$$

$$
\leq -2\eta k (f(x^k) - f(x^*)) - \eta k \mu \| x^k - x^* \|^2.
$$

Besides, $\| x^{k+1} - x^k \|^2 = 2\eta^2 \| g^k \|^2 - \| x^{k+1} - x^k \|^2$, so

$$
E_k \left[ \| x^{k+1} - x^* \|^2 \right] + E_k \left[ \| x^{k+1} - x^k \|^2 \right]
$$

$$
= (1 - \eta k \mu) \| x^k - x^* \|^2 - 2\eta k (f(x^k) - f(x^*)) + 2\eta^2 E_k \left[ \| g^k \|^2 \right]
$$

$$
\leq (1 - \eta k \mu) \| x^k - x^* \|^2 - 2\eta k (f(x^k) - f(x^*)) + 2\eta^2 E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right].
$$

Applying Proposition 2 to the obtained bound results in the following recursion

$$
E_k \left[ \| x^{k+1} - x^* \|^2 \right] + E_k \left[ \| x^{k+1} - x^k \|^2 \right]
$$

$$
\leq (1 - \eta k \mu) \| x^k - x^* \|^2 + \frac{1}{2} E_k \left[ \| x^k - x^{k-1} \|^2 \right] - 2\eta k (f(x^k) - f(x^*)) + 2\eta^2 E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right].
$$

With the GD estimator, the produced bound simplifies to

$$
E_k \left[ \| x^{k+1} - x^* \|^2 \right] + E_k \left[ \| x^{k+1} - x^k \|^2 \right]
$$

$$
\leq (1 - \eta k \mu) \| x^k - x^* \|^2 + \frac{1}{2} \| x^k - x^{k-1} \|^2 - 2\eta k (1 - 2\eta k L) (f(x^k) - f(x^*)).
$$

Based on Lemma 6, the following is satisfied for IntDIANA with GD estimator and adaptive $\alpha_k = \frac{\eta k \sqrt{\eta}}{\sqrt{n} \| x^k - x^* \|^2}$:

$$
E_k \left[ \sigma_2^{k+1} \right] \leq \frac{C}{2} (f(x^k) - f(x^*)) + \eta k \| x^k - x^{k-1} \|^2.
$$

In turn, Lemma 5 gives for L-SVRG estimator $E_k \left[ \left\| \frac{1}{n} \sum_{i=1}^{n} g_i^k \right\|^2 \right] \leq (2L + \frac{E}{n}) (f(x^k) - f(x^*)) + \frac{2}{n} \sigma_1^k$, so we can derive that

$$
E_k \left[ \| x^{k+1} - x^* \|^2 \right] + E_k \left[ \| x^{k+1} - x^k \|^2 \right]
$$

$$
\leq (1 - \eta k \mu) \| x^k - x^* \|^2 + \frac{1}{2} \| x^k - x^{k-1} \|^2 - 2\eta k \left( L + \frac{C}{2n} \right) (f(x^k) - f(x^*)) + \frac{4\eta^2}{n} \sigma_1^k.
$$

Let us now combine Equation (16) and Lemma 6:

$$
E_k \left[ \sigma_1^{k+1} \right] \leq (1 - p) \sigma_1^k + \frac{pC}{2} (f(x^k) - f(x^*)),
$$

$$
E_k \left[ \sigma_2^{k+1} \right] \leq 4\sigma_1^k + 3L (f(x^k) - f(x^*)) + n \| x^k - x^* \|^2.
$$

Lemma 8. We define the Lyapunov function as $\Psi^k \overset{\text{def}}{=} \| x^k - x^* \|^2 + \| x^k - x^{k-1} \|^2 + c_1 \eta k^2 \sigma_1^k + c_2 \eta k^2 \sigma_2^k$. Assume that the conditions of Lemma 7 hold. If $\mu > 0$, we have:

$$
E \left[ \Psi^{k+1} \right] \leq \theta E \left[ \Psi^k \right],
$$

where $\theta \overset{\text{def}}{=} \max \left\{ (1 - \eta k \mu), \frac{3}{4} \right\}, c_1 = 0, c_2 = \frac{L^2}{4n}$ and $\eta k \leq \frac{1}{2(L + L/32n)}$ for IntDIANA with GD estimator. Alternatively, for IntDIANA with L-SVRG estimator, we have $\theta \overset{\text{def}}{=} \max \left\{ (1 - \eta k \mu), \frac{3}{4} \right\}, (1 - \frac{3}{8m})$ and set $c_1 = \frac{8m}{n}, c_2 = \frac{L^2}{4m}, p = \frac{1}{m}$, and $\eta k \leq \frac{1}{2(L + L/32n)}$. If $\mu = 0$, we have

$$
\eta k E \left[ f(x^k) - f(x^*) \right] \leq E \left[ \Psi^k \right] - E \left[ \Psi^{k+1} \right],
$$

where $\eta k \leq \frac{1}{4(L + L/32n)}$ for the GD variant and $\eta k \leq \frac{1}{4(L + L/32n)}$ for the L-SVRG variant.
Proof. We define the Lyapunov function as \( \Psi^k \) \(= \|x^k - x^*\|^2 + c_1 \eta_k \sigma_1^2 + c_2 \eta_k^2 \sigma_2^2 \). For IntDIANA with GD estimator, we can set \( c_1 = 0 \) and derive the following inequality from Lemma 7 and \( \eta_{k+1} \leq \eta_k \):

\[
\begin{align*}
E[\Psi^{k+1}] & \leq (1 - \eta_k \mu) E[\|x^k - x^*\|^2] + \left( \frac{1}{2} + c_2 \eta_k^2 n \right) E[\|x^k - x^{k-1}\|^2] \\
& \quad - 2\eta_k \left( 1 - 2\eta_k \left( L + \frac{c_2 L}{8} \right) \right) E[f(x^k) - f(x^*)].
\end{align*}
\]

We first consider \( \mu > 0 \) case. Let \( c_2 = \frac{L^2}{8n} \), and \( \eta_k \leq \frac{1}{2(L + 2L/32n)} \). We have \( E[\Psi^{k+1}] \leq E[\Psi^k] \).

For IntDIANA with L-SVRG estimator, we have the following based on Lemma 7:

\[
E[\Psi^{k+1}] \leq (1 - \eta_k \mu) E[\|x^k - x^*\|^2] + \left( \frac{1}{2} + c_2 \eta_k^2 n \right) E[\|x^k - x^{k-1}\|^2] \\
+ \eta_k^2 \left( \frac{4}{n} + 4c_2 + (1 - p)c_1 \right) E[\sigma_k^2] \\
- 2\eta_k \left( 1 - 2\eta_k \left( L + \frac{L}{2n} + \frac{pc_1 L}{8} + \frac{3c_2 L}{4} \right) \right) E[f(x^k) - f(x^*)].
\]

Let \( c_1 = \frac{8m}{n}, c_2 = \frac{L^2}{4n}, p = \frac{1}{m} \), and \( \eta_k \leq \frac{1}{2(L + 2L/32n)} \). Plugging these values into the recursion, we get \( E[\Psi^{k+1}] \leq E[\Psi^k] \).

If \( \mu = 0 \), we instead let \( \eta_k \leq \frac{1}{4(L + 2L/32n)} \) for the GD variant and \( \eta_k \leq \frac{1}{4(L + 2L/32n)} \) for the L-SVRG variant to obtain from the same recursions:

\[
\eta_k E[f(x^k) - f(x^*)] \leq E[\Psi^k] - E[\Psi^{k+1}].
\]

\[\square\]

E. More Details of Experiments

E.1. Logistic Regression

Here, we show the full experimental results of logistic regression, including the omitted ones in Appendix A.

E.2. Details of the Deep Learning Experiments

Here we provide the omitted details of the deep learning experiments. Since we don’t have multiple GPUs connected by switch, the experiments are simulated on a machine with single NVIDIA V100 GPU and there are 8 “fake” workers in total. The machine runs Ubuntu 18.04.5 and CUDA 11.1. We used the implementation of ResNet-18 and ResNet-50 from the public repository https://github.com/kuangliu/pytorch-cifar and the implementation of VAE and \( \beta \)-TCVAE from https://github.com/AntixK/PyTorch-VAE. We utilize the standard data preprocessing schemes of those networks. Weight decay and momentum are not used. The curves with error bars are plotted based on the experiments from 4 different random seeds \( \{88508, 340925, 173557, 276948\} \). We tune the initial stepsize in \( \{0.0001, 0.001, 0.01, 0.1, 1\} \) with SGD and the stepsize is divided by 10 at epochs 120 and 160, which is shared by all algorithms. Finally, we choose initial step size \( \eta_0 = 0.1 \) for ResNet-18 and ResNet-50 and \( \eta_0 = 10^{-4} \) for VAE and \( \beta \)-TCVAE.

Randomized vs. Deterministic Rounding for IntSGD. The proposed IntSGD is built upon the rounding operator. In our theory, the rounding operator is based on a bernoulli random variable per coordinate (i.e., it is randomized). However, the deterministic rounding operator (e.g., torch.round) is faster in practice given that the number of coordinates is usually huge and drawing a high-dimensional random vector could be slow. E.g., IntSGD (\( \beta = 0, \sigma_Q^2 = 10^{-8} \)) with randomized rounding takes 3 more hours (execution time) than that with deterministic rounding to run 120 epochs on training ResNet-50 for classifying Cifar10. According to Figure 7, it can be seen that randomized rounding leads to slightly better performance than the deterministic counterpart. However, when comparing with other baseline algorithms in Figure 1 and Figure 2, we show IntSGD with deterministic rounding because it is faster and has simpler implementation.
Figure 4: Experimental results to demonstrate how $\beta$ and $\sigma^2_Q$ affect the performance of IntGD. “Max integer to send” represents the largest transmitted integers in a communication round (including both worker to master and master to worker communications).

Figure 5: Compare VR-IntDIANA with L-SVRG, VR-NatDIANA, and the heuristic-based VR-HintDIANA.
Figure 6: Compare four variants of IntSGD (full gradient or stochastic gradient) and IntDIANA (full gradient or variance-reduced stochastic gradient).

Figure 7: Experimental results of IntSGD with deterministic or randomized rounding (denoted as “rd”).

IntSGD: Floatless Compression of Stochastic Gradients

---

**IntSGD**

- Floatless Compression of Stochastic Gradients
- Compared with IntDIANA
- Benefits of reduced communication overhead
- Ideal for distributed machine learning scenarios

---

**IntDIANA**

- Full gradient or variance-reduced stochastic gradient
- Randomized rounding (denoted as “rd”)

---

**Real-world Applications**

- Improved efficiency in deep learning models
- Enhanced performance in various datasets

---

**Performance Metrics**

- Test Accuracy
- Total transmitted data (MB)
- Train Loss
- Test Loss

---

**Graphs**

- Log-log plots for easier visualization
- Comparison of different variants
- Key insights into the effectiveness of IntSGD and IntDIANA
Layer-wise vs. scalar $\alpha_k$ for IntSGD. We also investigate the empirical performance of the layer-wise $\alpha_{k,l}$ introduced in Section 4.3. As shown in Figure 8, we can observe that the layer-wise scaling vector does not have benefit over the scalar counterpart in communication efficiency. A tentative explanation has been given in Section 4.7.

Figure 8: Experimental results of IntSGD with scalar or layer-wise $\alpha_k$ (denoted as “lw”).