LOSCAR SGD: Local SGD with Communication-Computation Overlap and Delay-Corrected Sparse Model Averaging

Yassine Maziane
KAUST
yassine.maziane@kaust.edu.sa

Ammar Mahran
KAUST
majiedammar.mahran@kaust.edu.sa

Artavazd Maranjyan
KAUST
artavazd.maranjyan@kaust.edu.sa

Peter Richtárik
KAUST
peter.richtarik@kaust.edu.sa

Abstract

Communication is a major bottleneck in distributed learning, especially in large-scale settings and in federated learning environments with slow links. Three standard ways to reduce this cost are communication compression, local training, and communication-computation overlap. Methods that combine these ingredients are used in practice and have been found to be effective for large-scale training, but there is little theory for methods that combine all three. We study a heterogeneous-compute setting in which different workers may take different numbers of local steps, and we propose LOSCAR SGD, a Local SGD method that communicates only a sparse subset of model coordinates and continues optimizing while communication is in flight. A key ingredient is a delay-corrected merge rule that incorporates delayed synchronized information without discarding the progress made during the overlap phase. We give convergence guarantees for smooth non-convex shared objectives and show how sparsity, overlap, and worker heterogeneity affect the rate. To the best of our knowledge, this is the first joint finite-time convergence and wall-clock analysis for this setting. Experiments further show that communication-computation overlap can reduce training time.

1 Introduction

Distributed learning enables the training of large models across many workers. The most common setup is *data parallelism*, where the data is split across workers, each worker keeps a copy of the model, and the workers communicate to train one shared model [Goyal et al., 2017, Li et al., 2020a, Zhao et al., 2023]. A simple baseline in this setting is synchronized Minibatch SGD: at every iteration, each worker computes one stochastic gradient, the gradients are averaged, and one global update is taken. This is simple, but it requires communication at every iteration. As models become larger and the number of workers grows, communication can become the main factor limiting training speed. Federated learning is an especially clear example of this issue, since communication often happens over the internet and is much slower than local computation [McMahan et al., 2017, Kairouz et al., 2021, Wang et al., 2021]. Here, we focus on communication-limited regimes, including bandwidth-constrained or geographically distributed deployments.

There are several common ways to reduce this communication cost. One is **local training**: instead of synchronizing after every stochastic-gradient step, workers perform several local SGD steps before they communicate. Because the local steps are simple SGD steps, this approach is often called

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Table 1: Algorithmic features of representative methods.

Method	Local steps	Compression	Comm./ comp. overlap	Worker-specific steps
Minibatch SGD	✗	✗	✗	✗
Local SGD [Zinkevich et al., 2010]	✔	✗	✗	✗
FedAvg [McMahan et al., 2017]	✔	✗	✗	✔[(a)]
SPARTA (with DiLoCo) [Beton et al., 2025, Douillard et al., 2023]	✔	✔	✔	✗
LoCoDL [Condat et al., 2025]	✔	✔	✗	✗
Streaming DiLoCo [(b)] [Douillard et al., 2025]	✔	✔	✔	✔
Overlap-Local-SGD, CO2 [Wang et al., 2020a, Sun et al., 2024]	✔	✗	✔	✗
FedProx, GradSkip [Li et al., 2020b, Maranjyan et al., 2025c]	✔	✗	✗	✔
LoSCAR SGD (Algorithm 1)	✔	✔	✔	✔

A checkmark denotes a feature of the method or a reported variant. Local steps: multiple updates between full-model or same-fragment synchronizations. Compression: sparse or otherwise compressed messages. Overlap: computation continues during communication. Worker-specific steps: local or overlap step counts may differ across workers. Marks describe algorithmic features, not joint theoretical guarantees; see Appendix A. For SPARTA (with DiLoCo), local steps refer to the interval between full-model synchronizations.
(a) FedAvg's unequal step counts can result from unequal dataset sizes, without adapting to worker speed.
(b) Streaming DiLoCo uses FP4 outer-update quantization; its standard schedule uses a common delay, while an ablation tests different overlap delays across workers.

Local SGD [Zinkevich et al., 2010], and in the federated learning literature it became popular through FedAvg [McMahan et al., 2017]. Another idea is **communication-computation overlap**. In standard synchronized methods such as Minibatch SGD and plain FedAvg-style local training, workers may sit idle while messages are being sent, aggregated, and returned. With overlap, workers keep optimizing locally while communication is in flight, which can hide part of the communication delay and reduce training time [Wang et al., 2020a, Sun et al., 2024, Kale et al., 2025]. The returned average is then based on models that have already changed, so the workers must decide how to combine the delayed result with their current models. A third idea is to reduce the amount of communicated information through **compression**, for example via quantization or sparsification [Alistarh et al., 2017, Wangni et al., 2018]. Quantization reduces the bits used per value, while sparsification sends only selected coordinates; either gradients or model parameters can be compressed. In local training, workers often synchronize model weights by averaging them; with sparsification, only a subset of model coordinates is synchronized [Fournier et al., 2024, Beton et al., 2025].

These techniques are each well studied in both theory and practice. There is also practical evidence that combining them can work well [Douillard et al., 2025, Ajanthan et al., 2026]. However, to the best of our knowledge, there is no joint finite-time convergence and wall-clock analysis of sparse model averaging with communication-computation overlap and worker-specific local-step counts. The problem is further complicated when workers have different computation speeds, as in heterogeneous clusters or federated settings, because different workers may perform different amounts of local work in the same amount of time. A natural way to handle this is to let different workers take different numbers of local steps so that they remain roughly aligned in time [Li et al., 2020b, Maranjyan et al., 2025c]. ATA adapts task allocation to heterogeneous worker computation times [Maranjyan et al., 2025b].

Table 1 compares the algorithmic ingredients of representative methods, including their reported variants. Its marks describe algorithmic features rather than joint theoretical guarantees; Appendix A gives the extended discussion and update-level comparisons.

In this work, we study exactly this regime in the data-homogeneous setting: all workers optimize a shared objective. We propose LoSCAR SGD, a Local SGD-type method with infrequent sparse model averaging, communication-computation overlap, and worker-specific local-step counts. Workers may take different numbers of steps both before and during communication, and we model the time to send, average, and return the updates by a common communication delay (Assumption 2.1). We also

study a delay-corrected merge rule that preserves the progress made during the overlap phase instead of discarding it through a naive averaging overwrite.

1.1 Contributions

- We introduce a general Local SGD framework (Algorithm 1) for data-homogeneous distributed optimization that combines four ingredients in one model: local training, sparse model averaging, communication-computation overlap, and worker-specific local-step counts.
- We study a delay-corrected merge rule for sparse synchronization. When delayed synchronized information arrives, the rule retains the local progress made during the overlap phase. Instead of simply overwriting coordinates with a delayed average, the rule keeps the overlap progress and corrects only the synchronization disagreement.
- In Section 4, we prove convergence guarantees for smooth non-convex shared objectives (Theorem 4.1) and derive a wall-clock time-complexity bound (Corollary 4.2). The leading stochastic term retains linear-in-n minibatch speedup. The bounds show how sparsity, overlap, and heterogeneity affect convergence. They also show that the extra error caused by overlap-induced staleness appears as a controlled higher-order term.
- Experiments in Section 5 show that communication-computation overlap reduces training time, with regime-dependent merge-rule preferences and explicit failure cases. On four NVIDIA V100 GPUs across two nodes, with a controlled $250\,\mathrm{ms}$ minimum communication wait, overlap yields $1.39\times$–$1.72\times$ measured training-time speedups over work-matched blocking.

1.2 Related work

Local training Local training reduces communication by performing multiple stochastic-gradient steps between synchronizations. It was popularized in federated learning by FedAvg [McMahan et al., 2017], building on earlier distributed-training ideas [Zinkevich et al., 2010, Povey et al., 2014, Moritz et al., 2015]. Work addressing heterogeneous data includes Li et al. [2020c], Karimireddy et al. [2020], Mishchenko et al. [2022b], Douillard et al. [2023]; these papers include both theoretical and empirical studies. For our shared-objective setting, the most relevant theory is the homogeneous-data Local SGD line [Yu et al., 2019, Haddadpour et al., 2019, Wang and Joshi, 2021, Woodworth et al., 2020]; for broader background, see Malinovsky et al. [2022].

Compression and sparsification Another standard way to reduce communication is to transmit compressed information. Classical work studies gradient compression, sparsification, and quantization [Alistarh et al., 2017, Wangni et al., 2018, Lin et al., 2017, Beznosikov et al., 2023]. For methods based on periodic model averaging, closer references combine local training with partial model synchronization or compressed communication [Beton et al., 2025, Filippova et al., 2025, Douillard et al., 2025, Condat et al., 2025, 2026]. SPARTA asynchronously shares subsets of model parameters, whereas LoCoDL combines local training with unbiased compression [Beton et al., 2025, Condat et al., 2025].

Communication-computation overlap Communication-computation overlap is one of the main ingredients of our method. Instead of leaving workers idle while messages are sent, aggregated, and returned, overlap lets them continue taking useful local updates while communication is in flight, which can reduce training time. Overlap-Local-SGD and CO2 apply this idea to local training [Wang et al., 2020a, Sun et al., 2024]. CoCoD-SGD adds the local displacement to the averaged launch model: under equal weights and common step counts, this is the dense counterpart of our merge [Shen et al., 2019]. Streaming DiLoCo synchronizes different model fragments while training continues [Douillard et al., 2025]; eager DiLoCo combines a current local outer-gradient contribution with delayed remote contributions [Kale et al., 2025]. ACCO instead overlaps delayed-gradient synchronization with new gradient computation in sharded-optimizer training [Nabli et al., 2025]. Overlap and preservation of local progress are therefore established ideas; our contribution is their joint finite-time convergence and wall-clock analysis with sparse averaging and worker-specific steps.

Asynchronous methods Asynchronous SGD also avoids worker idling, but it does so by removing round-level synchronization altogether [Agarwal and Duchi, 2011, Niu et al., 2011, Maranjyan, 2025]. The motivation is therefore similar to overlap, but the mechanism is different. Our method keeps explicit rounds and a shared server message at the end of each round, while allowing controlled staleness through delayed sparse synchronization. Workers receive a common sparse average after taking different numbers of local steps during the same communication window. Structured coordination also appears in optimal asynchronous methods [Tyurin and Richtárik, 2023, Maranjyan et al., 2025d, Maranjyan and Richtárik, 2026]. A close comparison in this direction is Tyurin and Sivtsov [2026], which combines local training with asynchronous model averaging but does not include model sparsification. Decoupled DiLoCo asynchronously aggregates model fragments across independent learners [Douillard et al., 2026].

2 Problem setup

We consider the stochastic optimization problem

$$\min_{x\in\mathbb{R}^d} \left\{ f(x) \coloneqq \mathbb{E}_{\xi\sim\mathcal{D}}\left[F(x;\xi)\right]\right\}, \tag{1}$$

where $x \in \mathbb{R}^d$ is the model parameter, ξ is a random sample from $\mathcal{D}$, $F(x;\xi)$ is the sample loss, and f the population objective. We work in a distributed, data-homogeneous setting with n workers. Each worker keeps its own copy of the model and draws data from the same distribution $\mathcal{D}$. Every worker can therefore, in principle, optimize (1) on its own. We use multiple workers to speed up training by parallelizing computation. Workers may have different computation speeds. Worker i evaluates stochastic gradients $g_i(x, \xi_i)$ using fresh samples ξ_i. To measure progress over time, we also need a model for computation and communication times.

Assumption 2.1 (Worker timing model)**.** *For each worker $i \in \{1, \dots, n\}$, one stochastic-gradient computation takes τ_i seconds, where τ_i is a positive integer. Let*

$$\tau \coloneqq \operatorname{lcm}(\tau_1, \dots, \tau_n)$$

be the least common multiple of these computation times. We also assume that one communication phase, from the moment workers send their messages until the aggregated message is available again at the workers, takes ζ seconds, where

$$\zeta \in \{0, \tau, 2\tau, \dots\}.$$

Assumption 2.1 is a modeling simplification for the analysis. Real computation times need not be integer-valued, and communication delays need not be constant. Since every τ_i divides τ, intervals that are multiples of τ allow every worker to complete an integer number of local steps. Using such intervals for local computation and communication lets us define rounds and count updates without tracking partially completed steps or variable message-arrival times. The model retains the effects we want to study: faster workers can take more local steps, communication takes time, and workers can continue optimizing while messages are in flight. Practical implementations can approximate this schedule with fixed time budgets and count the local steps completed within each budget; the guarantees below apply to the stated timing model.

We make the following standard assumptions on the objective:

Assumption 2.2 (Lower boundedness)**.** *The objective $f : \mathbb{R}^d \to \mathbb{R}$ is bounded from below, i.e., there exists $f^\star \in \mathbb{R}$ such that*

$$f(x) \geq f^\star \qquad \forall x \in \mathbb{R}^d.$$

Assumption 2.3 (Smoothness)**.** *The objective f is differentiable and L-smooth, that is, there exists $L > 0$ such that*

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \qquad \forall x, y \in \mathbb{R}^d.$$

Randomness enters our model through the random data samples used to compute stochastic gradients, as well as the random masks sampled for the coordinate sparsifier (cf. Section 3). We make the natural assumption that these sources of randomness are independent.

Assumption 2.4 (Independent samples and masks)**.** *The collection of all data samples $\{\xi_{i,t}^r\}$ and all sparsification masks $\{S_r\}$ across all workers, local steps, and rounds is mutually independent.*

Moreover, we make the following assumptions about the stochastic gradients:

Assumption 2.5 (Unbiased stochastic gradients)**.** *For every worker i and every $x \in \mathbb{R}^d$,*

$$\mathbb{E}\left[\, g_i(x, \xi_i) \middle|\, x \right] = \nabla f(x).$$

Assumption 2.6 (Bounded variance)**.** *There exists $\sigma^2 \geq 0$ such that for every worker i and every $x \in \mathbb{R}^d$,*

$$\mathbb{E}\left[\, \|g_i(x, \xi_i) - \nabla f(x)\|^2 \middle|\, x \right] \leq \sigma^2.$$

Assumption 2.7 (Bounded second moment)**.** *There exists $G > 0$ such that for every worker i and every $x \in \mathbb{R}^d$,*

$$\mathbb{E}\left[\, \|g_i(x, \xi_i)\|^2 \middle|\, x \right] \leq G^2.$$

These assumptions concern a shared objective and uniform moment bounds; non-IID experiments are stress tests outside the guarantees. The proof notation is defined in Appendix C.

3 Method

LOSCAR SGD combines local SGD, sparse model averaging, and communication-computation overlap. The aim is to use the workers' computation time: they keep taking local SGD steps while their messages are sent, averaged, and returned. As explained in Section 2, workers can have different computation speeds.

The method uses two timing parameters. The integer $M \geq 1$ determines how long workers train locally before communicating: the first phase of each round lasts $M\tau$ seconds. Workers then send sparse model information to the server and continue training during the ζ seconds needed for communication. When the averaged message returns, each worker merges it with its current model and starts the next round. Algorithm 1 summarizes the method; we explain one round below.

Algorithm 1 LOSCAR SGD (Local SGD with Overlapped Sparse Corrected AveRaging)

Input: Number of workers n; worker compute times $\tau_1, \ldots, \tau_n \in \mathbb{N}$; $\tau \leftarrow \operatorname{lcm}(\tau_1, \ldots, \tau_n)$ (least common multiple of the worker compute times)
integer $M \geq 1$ (local-computation parameter)
communication duration $\zeta \in \{0, \tau, 2\tau, \ldots\}$
sparsification level $K \in \{1, \ldots, d\}$
stepsize $\eta > 0$
$N_i \leftarrow M\tau/\tau_i$ (# local steps before communication starts) for each worker i
$Q_i \leftarrow \zeta/\tau_i$ (# local steps during communication) for each worker i
Initialize $x_i^0 \in \mathbb{R}^d$ for $i = 1, \ldots, n$
for $r = 0, 1, 2, \ldots$ **do**
  Sample a common Rand-K mask $S_r \subseteq [d]$
  for all workers $i = 1, \ldots, n$ **in parallel do**
    Starting from x_i^r, run N_i local SGD steps with stepsize η to obtain y_i^r
    Form compressed message $m_i^r \leftarrow \mathcal{C}_{S_r}(y_i^r)$
    Send m_i^r to the server
    While communication is in flight, run Q_i further local SGD steps, using stepsize η, from y_i^r to obtain z_i^r
  end for
  Server computes $m^r \leftarrow \frac{1}{n}\sum_{i=1}^n m_i^r$
  Server broadcasts m^r to all workers
  for all workers $i = 1, \ldots, n$ **in parallel do**
    Merge delayed average with latest local model:

$$x_{i,j}^{r+1} \leftarrow \begin{cases} m_j^r + (z_{i,j}^r - y_{i,j}^r), & j \in S_r, \\ z_{i,j}^r, & j \notin S_r \end{cases} \qquad \forall j \in [d]$$

  end for
end for

One round in detail At the beginning of round r, worker i holds its local model x_i^r. During the first $M\tau$ seconds, it takes

$$N_i = \frac{M\tau}{\tau_i} \tag{2}$$

local SGD steps to obtain y_i^r, the model it prepares for communication. It sends a sparse version of this model to the server. While the server averages the messages and returns the result, worker i continues training for another

$$Q_i = \frac{\zeta}{\tau_i} \tag{3}$$

steps, reaching z_i^r. Thus y_i^r is the model when communication starts, whereas z_i^r is the newer model when the result arrives. The total number of local steps is

$$H_i = N_i + Q_i = \frac{M\tau+\zeta}{\tau_i}. \tag{4}$$

Faster workers therefore take more steps within the same round duration. Finally, each worker combines the delayed average with z_i^r to form the next starting point x_i^{r+1}, using the merge rule below.

Compressed synchronization Sparsification reduces the message size by communicating only K of the d model coordinates. In each round, we sample a uniform K-coordinate mask S_r, independently of previous masks and all gradient samples. Every worker uses the same mask, so the server receives the same coordinates from all workers and can average them coordinate by coordinate. Writing $P_r = \mathrm{Proj}_{S_r}$, the compressor $\mathcal{C}_{S_r}(x) = P_r x$ keeps the selected coordinates and sets the others to zero. Worker i sends $m_i^r = P_r y_i^r$; the server returns $m^r = \frac{1}{n}\sum_{i=1}^n m_i^r = P_r \bar{y}^r$, where $\bar{y}^r = \frac{1}{n}\sum_{i=1}^n y_i^r$. Only the selected entries are transmitted; the full-dimensional notation is for convenience. The message averages the launch models y_i^r, not the newer models z_i^r. Each coordinate is selected with probability $p = K/d$. We do not rescale selected values by $1/p$, so this is an unscaled projector rather than an unbiased compressor:

$$p = K/d, \qquad q = 1 - p, \qquad \mathbb{E}[P_r x] = px. \tag{5}$$

Merge rules The two merge rules can be written using the shared projector P_r:

$$\begin{aligned} \text{Overwrite:} \quad & x_i^{r+1} = z_i^r + P_r(\bar{y}^r - z_i^r), \\ \text{Delay-corrected:} \quad & x_i^{r+1} = z_i^r + P_r(\bar{y}^r - y_i^r). \end{aligned}$$

Overwrite replaces selected coordinates with the delayed average and discards their overlap-phase progress. The **delay-corrected merge** starts from the current model z_i^r and adds the selected-coordinate correction $\bar{y}^r - y_i^r$. It therefore preserves the displacement $z_i^r - y_i^r$ accumulated during communication and satisfies $\bar{x}^{r+1} = \bar{z}^r$ for every sampled mask. With dense communication, equal weights, and common step counts, this is the CoCoD-SGD merge [Shen et al., 2019].

Special cases Setting $K = d$ removes sparsification. Setting $\zeta = 0$ removes overlap and models communication with no added time cost, rather than a positive-delay blocking implementation. With equal worker times, these choices recover local SGD; taking $M = 1$ further recovers synchronized minibatch SGD. Unlike asynchronous SGD, our workers receive a common synchronization result at each round boundary.

4 Theory

For the local-step counts defined in Section 3, let

$$\bar{H} := \tfrac{1}{n}\textstyle\sum_{i=1}^n H_i, \qquad H_{\max} := \max_{1\le i\le n} H_i, \tag{6}$$

$$S_N := \textstyle\sum_{i=1}^n N_i^2, \qquad S_Q := \sum_{i=1}^n Q_i^2, \tag{7}$$

$$\Psi_H := \textstyle\sum_{i=1}^n \sum_{t=0}^{H_i-1} t^2. \tag{8}$$

Here $\bar{H}, H_{\max}$ summarize total local work, S_N, S_Q summarize work before and during communication, and Ψ_H controls cumulative within-round drift. We can now state the main convergence guarantee for Algorithm 1.

Theorem 4.1 (Convergence guarantee). *Consider Algorithm 1, initialized with $x_i^0 = x^0 \in \mathbb{R}^d$, $i = 1, \dots, n$. Let Assumptions 2.2 to 2.7 hold. Choose a stepsize η satisfying $0 < \eta \leq 1/8LH_{\max}$, where $H_{\max}$ is defined in (6). Choose the sparsification level $K \in \{1, \dots, d\}$, let $q = 1 - \frac{K}{d}$ as in (5), and choose analysis parameters $\alpha > 0$ and $\beta > 0$ such that $c := q(1+\alpha)(1+\beta) < 1$. Further, define $B := q(1+\beta)\left(1+\frac{1}{\alpha}\right)$, $D := 1 + \frac{q}{\beta}$. Then for every $R \geq 1$, the average iterate $\bar{x}^r := \frac{1}{n}\sum_{i=1}^n x_i^r$ satisfies*

$$\begin{aligned} \frac{1}{R}\sum_{r=0}^{R-1} \mathbb{E}\left[\|\nabla f(\bar{x}^r)\|^2\right] \quad &\leq \quad \frac{4(f(x^0)-f^\star)}{\eta \bar{H} R} + \frac{4L\eta\sigma^2}{n} \\ &\quad + \frac{6L^2\eta^2 G^2 \Psi_H}{n\bar{H}} + \frac{6L^2\eta^2 G^2 H_{\max}}{n\bar{H}} \frac{BS_N + DS_Q}{1-c}, \end{aligned} \tag{9}$$

where $\bar{H}$ is defined in (6), S_N and S_Q are defined in (7), and Ψ_H is defined in (8).

Interpretation The bound in (9) separates four effects. The first term decreases with $\eta\bar{H}R$: more effective local progress per round can reduce the number of communication rounds needed. However, the stepsize restriction $\eta \leq 1/(8LH_{\max})$ limits this benefit through the worker-imbalance ratio $\bar{H}/H_{\max}$. The second term, $4L\eta\sigma^2/n$, is the stochastic noise term and retains the familiar $1/n$ variance reduction of minibatch SGD [Cotter et al., 2011, Gower et al., 2019].

The third term, involving Ψ_H, is the local-drift penalty: even with full communication and no delay, workers drift apart when they take several local steps before synchronizing. The fourth term captures disagreement from partial or delayed synchronization. Its S_N-part accounts for local work before communication, while its S_Q-part is the additional cost of continuing to optimize during communication; these quantities are step-count aggregates, not measured disagreements. Synchronizing fewer coordinates makes this residual disagreement harder to control through B, D, and c. Importantly, both drift and disagreement penalties scale as η^2, so these additional costs enter at higher order than the leading stochastic term.

With $K = d$, $c = B = 0$ and $D = 1$; with $\zeta = 0$, $S_Q = 0$. Together these remove synchronization disagreement, leaving the local-SGD drift term Ψ_H. For equal worker times and $M = 1$, $\Psi_H = 0$ as well, recovering the standard non-convex minibatch rate.

Time complexity We now translate the convergence bound into round and time complexity.

Corollary 4.2 (Time complexity; proof in Appendix C.8). *Let $\Delta := f(x^0) - f^\star$ and $X := \Psi_H + H_{\max}\frac{BS_N + DS_Q}{1-c}$. Let $\hat{r}$ be sampled uniformly at random from $\{0, 1, \dots, R-1\}$, independently of the algorithmic history. Under the assumptions of Theorem 4.1, for every $\varepsilon > 0$ there are a constant stepsize and an integer number of rounds $R \geq 1$ such that $\mathbb{E}\left[\left\|\nabla f(\bar{x}^{\hat{r}})\right\|^2\right] \leq \varepsilon$ within wall-clock time*

$$\mathcal{O}\left(\frac{\Delta L\sigma^2}{n\varepsilon^2}\cdot\tau_H + \frac{\Delta LG\sqrt{X}}{\varepsilon^{3/2}\sqrt{n\bar{H}}}\cdot\tau_H + \frac{\Delta LH_{\max}}{\varepsilon}\cdot\tau_H\right),$$

up to at most one additional round for integer rounding, where $\tau_H := \frac{M\tau+\zeta}{\bar{H}} = \frac{n}{\sum_{i=1}^n \tau_i^{-1}}$ is the harmonic mean of worker computation times. An explicit stepsize and round count are given in Appendix C.8.

The leading term $\Delta L\sigma^2\tau_H/(n\varepsilon^2)$ retains linear-in-n speedup and depends on worker speeds through their harmonic mean, as in recent asynchronous-SGD bounds [Mishchenko et al., 2022a, Koloskova et al., 2022]. Local work, sparsity, and overlap enter the higher-order terms through X and $H_{\max}$.

Optimized sparsity dependence Appendix B develops further theoretical extensions. First, Appendix B.1 optimizes α, β in closed form, removing these free proof parameters and making the dependence on $p = K/d$ explicit. At fixed local-step counts, the optimized disagreement factor scales as p^{-2}, so its contribution to the $\varepsilon^{-3/2}$ time term scales as p^{-1}; the leading ε^{-2} stochastic term is independent of p. The appendix gives both the exact optimum and a time bound written directly in terms of p.

Comparison with optimized blocking Second, Appendix B.2 compares overlap with blocking at the same payload and positive communication duration. In an explicit high-accuracy regime, a fixed overlap configuration has a smaller sufficient-time upper bound even after optimizing the blocking

synchronization interval and its proof parameters. This compares theoretical upper bounds, not actual runtimes; outside that regime the extra disagreement from overlap can make its bound larger.

Communication time depends on sparsity Third, Appendix B.3 couples the communication duration to the payload through $\zeta_{\text{sys}}(p) = \zeta_{\text{lat}} + (sd/\nu)p$, followed by rounding to the common time grid. Substituting this model into the time bound makes the local-step counts depend on p as well, and yields an explicit regime in which a sparse configuration has a smaller sufficient-time bound than its dense counterpart.

5 Experiments

We evaluate LOSCAR SGD in controlled simulations and on a multi-node GPU cluster. In the single-process PyTorch simulator, workers are separate model copies and communication delays are imposed explicitly. This isolates the effects of sparsity, overlap, merge rule, and worker speed from hardware and networking noise; the GPU experiments then measure actual training time. Local Sparse (LS) is the blocking baseline: it communicates the same shared Rand-K coordinates but waits during communication. **Overwrite** (OW) and **delay-corrected** (DC) continue local training using the merges in Section 3. Within each seed, comparisons share initialization, data partitions, minibatches, and masks where applicable. We report validation nAUC, the accuracy averaged over the common time horizon, and speedup, the LS time to a fixed accuracy target divided by the compared method's time to that target. nAUC summarizes the trajectory rather than final accuracy. Plots against processed examples compare progress at a fixed amount of data, separating sample efficiency from the timing benefit of computing during communication. Appendix D gives complete protocols, configurations, and results.

5.1 Mechanism: overlap and correction

We first use four-worker logistic regression on `a9a` with heterogeneous step times to illustrate the overlap mechanism; the setup is detailed in Appendix D.3.1. At matched round duration and payload, both overlap variants reduce loss faster than LS because they convert waiting time into local steps. Correction performs best in this controlled setting: it preserves the progress made on synchronized coordinates during communication instead of overwriting it (Figure 1(a)). Delay, local-budget, and sparsity ablations show how the benefit changes (Appendix D.4).

5.2 Neural networks

On four-worker CIFAR-10, three paired seeds give median simulated-time speedups to 70% validation accuracy of $1.19\times/1.25\times$ (OW/DC) in the normal regime and $2.85\times/3.10\times$ in the communication-stress regime. Exact configurations are given in Appendix D.3. Every method reaches the target in every seed. Tiny ImageNet also benefits most under communication stress (Appendix D.3).

A single-seed CIFAR-10 suite retains the gains when scaling from four to sixteen workers. At sixteen workers, validation-nAUC gains are 0.009 in the reference setting, 0.029–0.032 under longer delay, and 0.004–0.006 under compute heterogeneity (Appendix D.4.2 and table 4). A Tiny ImageNet grid covers 18 single-seed combinations of sparsity, delay, local work, and worker speeds: both overlap variants improve nAUC in all twelve settings with $p \in \{0.01, 0.25\}$, but only one of six with $p = 0.001$ (Appendix D.4.3). At this extreme sparsity, only 0.1% of coordinates are synchronized per round. A plausible explanation is that the extra local steps during communication increase worker disagreement faster than such sparse averaging can reduce it, offsetting the benefit of avoiding idle time; the results do not isolate this mechanism. The timing gains come from processing additional minibatches during communication; at equal processed-example counts, curves are much closer and their ordering is mixed (Appendix D.3).

5.3 Comparison with external overlap methods

We reimplement Overlap-Local-SGD, CoCoD-SGD, and CO2 [Wang et al., 2020a, Shen et al., 2019, Sun et al., 2024] alongside LS, OW, and DC on sixteen-worker IID CIFAR-10: five regimes, six methods, four paired seeds, 120 runs. All use the same model, inner SGD, and 400-effective-epoch

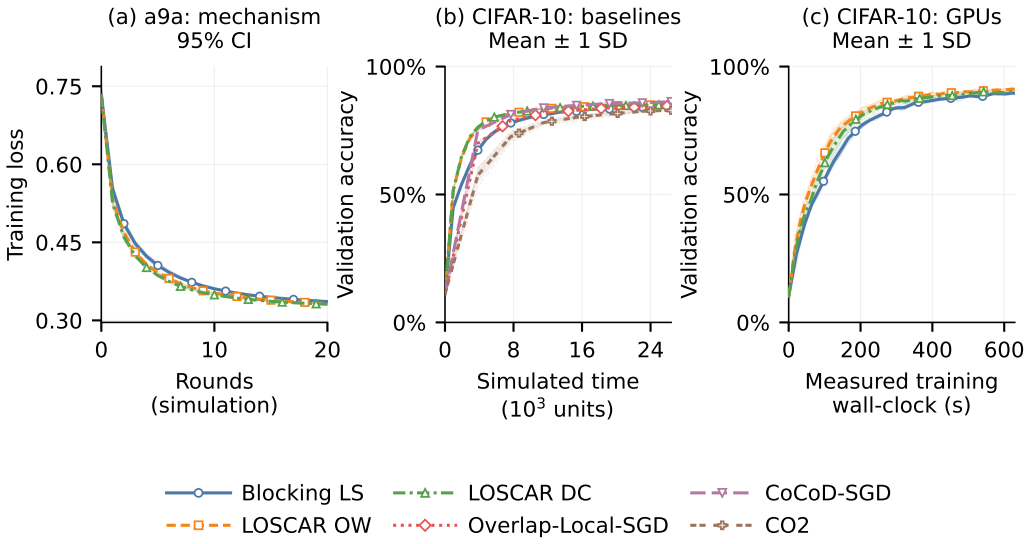


Figure 1: Overlap mechanism, external baselines, and measured hardware performance. (a) `a9a` logistic-regression training loss versus simulated rounds, with matched round duration and payload. Each round lasts 24 simulated time units; bands are pointwise 95% confidence intervals. (b) Sixteen-worker CIFAR-10 validation accuracy versus simulated time, with cumulative model-value payload matched between LOSCAR SGD and the dense overlap methods (about 38.6–38.7 GB); blocking LS is the work-matched reference. (c) CIFAR-10 validation accuracy for ResNet18-GN at $p = 0.25$ on four V100 GPUs across two nodes, with a 250 ms minimum wait before applying aggregated updates. The horizontal axis measures training wall-clock time, excluding dense evaluation. In (b,c), bands show one sample standard deviation over four paired seeds; pale lines are seed trajectories and opaque lines are means. OW/DC denote overwrite/delay correction. Protocols and uncertainty definitions appear in Appendices D.3.1, D.7 and D.9.

budget. A latency–bandwidth model makes dense synchronization slower than sparse synchronization. Four regimes share the pre-communication local-work parameter M; the fifth reduces dense communication frequency to match cumulative model-value payload (Appendix D.7).

Corrected LOSCAR SGD leads mean validation nAUC in four regimes. In the payload-matched setting (Figure 1(b)), the methods communicate nearly equal totals of model values: 38.60 GB for corrected LOSCAR SGD and 38.74 GB for each dense baseline. Corrected LOSCAR SGD obtains nAUC 0.7977, versus 0.7773 for CoCoD-SGD and 0.7546 for Overlap-Local-SGD; it reaches 80% validation accuracy 1.22× faster than CoCoD-SGD. CoCoD-SGD wins the separate high-sparsity regime (0.7911 versus 0.7884), where total payload is not matched: its dense messages carry ten times as many model values per synchronization as the sparse methods at $p = 0.10$, but it synchronizes less often. Across the five regimes, it also has the highest mean final accuracy (85.70% versus 84.75% for corrected LOSCAR SGD). Trajectory and endpoint rankings therefore differ.

These simulated payload totals exclude coordinate indices, assuming the shared sparse mask is already known. In a separate two-seed long-delay comparison that includes indices in both payload and delay, corrected LOSCAR SGD retains higher nAUC with 40.5% less communication than CoCoD-SGD, although CoCoD-SGD has higher final accuracy (Appendix D.7).

5.4 Real multi-node hardware

We run PyTorch Distributed with asynchronous NCCL over Ethernet on four NVIDIA V100 GPUs across two nodes (Appendix D.9). Two CIFAR-10 models, $p \in \{0.25, 0.50\}$, and up to four paired seeds use 100 effective epochs and SGD with momentum 0.9. LS, OW, and DC are matched in total local steps, communication count, and payload. Alongside native communication, we study longer delays by requiring workers to wait at least 25, 100, or $250\,\mathrm{ms}$ after communication starts before applying the aggregated update. Workers continue training during this wait in the overlap methods. These waits are imposed by the training code, not by emulating a wide-area network. Table 2 summarizes the $250\,\mathrm{ms}$ results.

Table 2: Measured four-V100 results at the $250\,\mathrm{ms}$ minimum result-availability window, as overwrite / delay-corrected. $Q^\star$ is the calibrated number of overlap steps. Training time excludes periodic dense evaluation. Speedups are paired-seed medians relative to work-matched LS; hidden fractions and final-accuracy differences (overlap minus LS) are means over four paired seeds.

Model	p	$Q^\star$	Communication hidden	Training-time speedup	Final val. difference
ResNet18-GN	0.25	6	$57\%/59\%$	$1.40\times/1.39\times$	$+0.63/-0.40$ pp
ResNet18-GN	0.50	9	$57\%/49\%$	$1.56\times/1.57\times$	$+0.86/-0.07$ pp
Small CNN	0.25	14	$94\%/88\%$	$1.72\times/1.66\times$	$+0.68/-0.09$ pp
Small CNN	0.50	13	$95\%/90\%$	$1.70\times/1.67\times$	$+0.31/-0.41$ pp

Overlap hides 49–95% of communication time and achieves $1.39\times$–$1.72\times$ training-time speedups (Figure 1(c)). If we include periodic dense evaluation, end-to-end speedups fall to $1.16\times$–$1.27\times$. At matched effective epochs, overwrite is approximately neutral and correction somewhat worse; overwrite has higher mean final validation accuracy than correction in all four configurations. The full delay sweep, target times, and momentum-free comparison are in Appendix D.9.

5.5 Limitations and failure regimes

Sparsity and heterogeneity. At $p = 0.001$, Tiny ImageNet median nAUC gains are negative ($-0.006/-0.008$, OW/DC). Overlap also reaches targets later than LS in sixteen-worker logistic regression with heterogeneous speeds and moderate delay (Appendix D.4.2). Non-IID stress tests lie outside our homogeneous-data guarantees: extreme `a9a` drift makes both overlap methods plateau below LS, while moderate-overlap CIFAR-10 retains timing gains (Appendix D.6).

Merge rule. In Tiny ImageNet, corrected-minus-overwrite nAUC changes from -0.0065 at short delay to $+0.00391$ at long delay, but correction wins only three of four long-delay seeds. Removing momentum at medium delay favors correction in all four seeds (Appendix D.5), whereas momentum-based hardware favors overwrite. Long delay alone does not guarantee a correction benefit.

Masks and compressors. At a fixed per-worker coordinate budget, independent masks generally weaken CIFAR-10 trajectories relative to a shared mask (Appendix D.8.1). At matched communication budgets, Top-K can improve early progress but increases disagreement and diverges once; all eight $1/p$-scaled Rand-K runs diverge (Appendix D.8.2). Dense INT8 has lower final accuracy and nAUC than shared Rand-K, but its deliberately simple quantizer (one model-wide scale, no error feedback) limits general conclusions. These ablations lie outside our shared-support theory.

6 Conclusion

We introduced LOSCAR SGD and established convergence and wall-clock complexity guarantees for sparse model averaging with communication–computation overlap and heterogeneous worker speeds. The leading stochastic term retains linear-in-n speedup, while local drift, sparsity, and overlap contribute higher-order terms to the convergence bound. Simulated and real-hardware experiments demonstrate substantial training-time gains, including 1.39–1.72× speedups on four GPUs, and competitive performance against existing overlap methods. The gains are strongest in communication-limited regimes, while aggressive sparsification and unfavorable worker heterogeneity can reduce them. The preferred merge rule also depends on the training regime. Extending the analysis to heterogeneous objectives is a natural direction for future work.

Acknowledgments and Disclosure of Funding

The research reported in this publication was supported by funding from King Abdullah University of Science and Technology (KAUST): i) KAUST Baseline Research Scheme, ii) CRG Grant ORFS-CRG12-2024-6460, and iii) Center of Excellence for Generative AI, under award number 5940.

The authors declare no competing interests.

References

Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. *Advances in Neural Information Processing Systems*, 24, 2011. (Cited on pages 4 and 20.)

Thalaiyasingam Ajanthan, Sameera Ramasinghe, Gil Avraham, Hadi Mohaghegh Dolatabadi, Chamin P Hewa Koneputugodage, Violetta Shevchenko, Yan Zuo, and Alexander Long. AsyncMesh: Fully asynchronous optimization for data and pipeline parallelism. *arXiv preprint arXiv:2601.22442*, 2026. (Cited on pages 2 and 18.)

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1709–1720, 2017. (Cited on pages 2, 3, and 17.)

Yossi Arjevani, Yair Carmon, John C Duchi, Dylan J Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, pages 1–50, 2022. (Cited on page 24.)

Debraj Basu, Deepesh Data, Can Karakus, and Suhas Diggavi. Qsparse-local-SGD: Distributed SGD with quantization, sparsification, and local computations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. (Cited on page 18.)

Matt Beton, Matthew Reed, Seth Howes, Alex Cheema, and Mohamed Baioumy. Improving the efficiency of distributed training using sparse parameter averaging. In *ICLR 2025 Workshop on Modularity for Collaborative, Decentralized, and Continual Deep Learning*, 2025. (Cited on pages 2, 3, and 17.)

Aleksandr Beznosikov, Samuel Horváth, Peter Richtárik, and Mher Safaryan. On biased compression for distributed learning. *Journal of Machine Learning Research*, 24(276):1–50, 2023. (Cited on pages 3 and 17.)

Laurent Condat, Artavazd Maranjyan, and Peter Richtárik. LoCoDL: Communication-efficient distributed learning with local training and compression. In *The Thirteenth International Conference on Learning Representations*, 2025. (Cited on pages 2, 3, and 17.)

Laurent Condat, Artavazd Maranjyan, and Peter Richtárik. BiCoLoR: Communication-efficient optimization with bidirectional compression and local training. *arXiv preprint arXiv:2601.12400*, 2026. (Cited on pages 3 and 17.)

Andrew Cotter, Ohad Shamir, Nati Srebro, and Karthik Sridharan. Better mini-batch algorithms via accelerated gradient methods. *Advances in Neural Information Processing Systems*, 24, 2011. (Cited on page 7.)

Arthur Douillard, Qixuan Feng, Andrei A Rusu, Rachita Chhaparia, Yani Donchev, Adhiguna Kuncoro, Marc'Aurelio Ranzato, Arthur Szlam, and Jiajun Shen. DiLoCo: Distributed low-communication training of language models. *arXiv preprint arXiv:2311.08105*, 2023. (Cited on pages 2, 3, and 17.)

Arthur Douillard, Yanislav Donchev, Keith Rush, Satyen Kale, Zachary Charles, Zachary Garrett, Gabriel Teston, Dave Lacey, Ross McIlroy, Jiajun Shen, et al. Streaming DiLoCo with overlapping communication: Towards a distributed free lunch. *arXiv preprint arXiv:2501.18512*, 2025. (Cited on pages 2, 3, 17, and 19.)

Arthur Douillard, Keith Rush, Yani Donchev, Zachary Charles, Nova Fallen, Ayush Dubey, Ionel Gog, Josef Dean, Blake Woodworth, Zachary Garrett, et al. Decoupled diloco for resilient distributed pre-training. *arXiv preprint arXiv:2604.21428*, 2026. (Cited on pages 4 and 20.)

Anastasiia Filippova, Angelos Katharopoulos, David Grangier, and Ronan Collobert. Partial parameter updates for efficient distributed training. *arXiv preprint arXiv:2509.22418*, 2025. (Cited on pages 3 and 17.)

Louis Fournier, Adel Nabli, Masih Aminbeidokhti, Marco Pedersoli, Eugene Belilovsky, and Edouard Oyallon. WASH: Train your ensemble with communication-efficient weight shuffling, then average. 2024. (Cited on page 2.)

Robert Mansel Gower, Nicolas Loizou, Xun Qian, Alibek Sailanbayev, Egor Shulgin, and Peter Richtárik. SGD: General analysis and improved rates. In *International Conference on Machine Learning*, pages 5200–5209. PMLR, 2019. (Cited on page 7.)

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: Training ImageNet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. (Cited on page 1.)

Farzin Haddadpour, Mohammad Mahdi Kamani, Mehrdad Mahdavi, and Viveck R. Cadambe. Local SGD with periodic averaging: Tighter analysis and adaptive synchronization. In *Neural Information Processing Systems*, 2019. (Cited on pages 3 and 17.)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. (Cited on page 41.)

Alex Iacob, Andrej Jovanovic, Mher Safaryan, Meghdad Kurmanji, Lorenzo Sani, Samuel Horváth, William Shen, Xinchi Qiu, and Nic Lane. MT-DAO: Multi-timescale distributed adaptive optimizers with local updates. In *International Conference on Learning Representations*, 2026a. (Cited on page 20.)

Alex Iacob, Lorenzo Sani, Mher Safaryan, Paris Giampouras, Samuel Horváth, Andrej Jovanovic, Meghdad Kurmanji, Preslav Aleksandrov, William Shen, Xinchi Qiu, and Nic Lane. DES-LOC: Desynced low communication adaptive optimizers for foundation models. In *International Conference on Learning Representations*, 2026b. (Cited on page 20.)

Andrej Jovanovic, Alex Iacob, Mher Safaryan, Ionut-Vlad Modoranu, Lorenzo Sani, William F. Shen, Xinchi Qiu, Dan Alistarh, and Nicholas D. Lane. LoRDO: Distributed low-rank optimization with infrequent communication. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 54741–54775. PMLR, 2026. (Cited on page 20.)

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021. (Cited on page 1.)

Satyen Kale, Arthur Douillard, and Yanislav Donchev. Eager updates for overlapped communication and computation in DiLoCo. *arXiv preprint arXiv:2502.12996*, 2025. (Cited on pages 2, 3, and 19.)

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pages 5132–5143. PMLR, 2020. (Cited on pages 3 and 17.)

Anastasiia Koloskova, Sebastian U Stich, and Martin Jaggi. Sharper convergence guarantees for asynchronous SGD for distributed and federated learning. *Advances in Neural Information Processing Systems*, 35: 17202–17215, 2022. (Cited on page 7.)

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto, 2009. (Cited on page 40.)

Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, et al. Pytorch distributed: Experiences on accelerating data parallel training. *arXiv preprint arXiv:2006.15704*, 2020a. (Cited on page 1.)

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020b. (Cited on page 2.)

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of FedAvg on non-IID data. In *International Conference on Learning Representations*, 2020c. (Cited on pages 3 and 17.)

Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*, 2017. (Cited on pages 3 and 17.)

Ammar Mahran, Artavazd Maranjyan, and Peter Richtárik. Rescaled asynchronous SGD: Optimal distributed optimization under data and system heterogeneity. *arXiv preprint arXiv:2605.13434*, 2026. (Cited on page 20.)

Grigory Malinovsky, Kai Yi, and Peter Richtárik. Variance reduced ProxSkip: Algorithm, theory and application to federated learning. *Advances in Neural Information Processing Systems*, 35:15176–15189, 2022. (Cited on pages 3 and 17.)

Artavazd Maranjyan. *First Provably Optimal Asynchronous SGD for Homogeneous and Heterogeneous Data*. PhD thesis, King Abdullah University of Science and Technology, 2025. (Cited on pages 4 and 20.)

Artavazd Maranjyan and Peter Richtárik. Ringleader ASGD: The first asynchronous SGD with optimal time complexity under data heterogeneity. In *The Fourteenth International Conference on Learning Representations*, 2026. (Cited on pages 4 and 20.)

Artavazd Maranjyan, Omar Shaikh Omar, and Peter Richtárik. Mindflayer SGD: Efficient parallel SGD in the presence of heterogeneous and random worker compute times. In *The 41st Conference on Uncertainty in Artificial Intelligence*, 2025a. (Cited on page 20.)

Artavazd Maranjyan, El Mehdi Saad, Peter Richtárik, and Francesco Orabona. ATA: Adaptive task allocation for efficient resource management in distributed machine learning. In *International Conference on Machine Learning*, 2025b. (Cited on page 2.)

Artavazd Maranjyan, Mher Safaryan, and Peter Richtárik. GradSkip: Communication-accelerated local gradient methods with better computational complexity. *Transactions on Machine Learning Research*, 2025c. ISSN 2835-8856. (Cited on page 2.)

Artavazd Maranjyan, Alexander Tyurin, and Peter Richtárik. Ringmaster ASGD: The first asynchronous SGD with optimal time complexity. In *International Conference on Machine Learning*, 2025d. (Cited on pages 4 and 20.)

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017. (Cited on pages 1, 2, 3, and 17.)

Konstantin Mishchenko, Francis Bach, Mathieu Even, and Blake E Woodworth. Asynchronous SGD beats minibatch SGD under arbitrary delays. *Advances in Neural Information Processing Systems*, 35:420–433, 2022a. (Cited on page 7.)

Konstantin Mishchenko, Grigory Malinovsky, Sebastian Stich, and Peter Richtárik. ProxSkip: Yes! Local gradient steps provably lead to communication acceleration! Finally! In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15750–15769. PMLR, 17–23 Jul 2022b. (Cited on pages 3 and 17.)

Philipp Moritz, Robert Nishihara, Ion Stoica, and Michael I Jordan. Sparknet: Training deep networks in spark. *arXiv preprint arXiv:1511.06051*, 2015. (Cited on pages 3 and 17.)

Adel Nabli, Louis Fournier, Pierre Erbacher, Louis Serrano, Eugene Belilovsky, and Edouard Oyallon. ACCO: Accumulate while you communicate for communication-overlapped sharded LLM training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. (Cited on pages 3 and 20.)

Feng Niu, Benjamin Recht, Christopher Ré, and Stephen J Wright. HOGWILD!: A lock-free approach to parallelizing stochastic gradient descent. *Advances in Neural Information Processing Systems*, 24, 2011. (Cited on pages 4 and 20.)

Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur. Parallel training of DNNs with natural gradient and parameter averaging. *arXiv preprint arXiv:1410.7455*, 2014. (Cited on pages 3 and 17.)

Abdurakhmon Sadiev, Artavazd Maranjyan, Ivan Ilin, and Peter Richtárik. Ringmaster LMO: Asynchronous linear minimization oracle momentum method. *arXiv preprint arXiv:2605.18174*, 2026. (Cited on page 20.)

Shuheng Shen, Linli Xu, Jingchang Liu, Xianfeng Liang, and Yifei Cheng. Faster distributed deep net training: Computation and communication decoupled stochastic gradient descent. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 4582–4589, 2019. doi: 10.24963/ijcai.2019/637. (Cited on pages 3, 6, 8, 17, and 64.)

Weigao Sun, Zhen Qin, Weixuan Sun, Shidi Li, Dong Li, Xuyang Shen, Yu Qiao, and Yiran Zhong. CO2: Efficient distributed training with full communication-computation overlap. In *The Twelfth International Conference on Learning Representations*, 2024. (Cited on pages 2, 3, 8, 18, 19, and 64.)

Zhirayr Tovmasyan, Artavazd Maranjyan, and Peter Richtárik. Rennala MVR: Improved time complexity for parallel stochastic optimization via momentum-based variance reduction. *arXiv preprint arXiv:2605.08871*, 2026. (Cited on page 20.)

Alexander Tyurin and Peter Richtárik. Optimal time complexities of parallel stochastic optimization methods under a fixed computation model. In *Advances in Neural Information Processing Systems*, volume 36, pages 16515–16577, 2023. (Cited on pages 4 and 20.)

Alexander Tyurin and Danil Sivtsov. Birch SGD: A tree graph framework for local and asynchronous SGD methods. In *The Fourteenth International Conference on Learning Representations*, 2026. (Cited on pages 4 and 20.)

Jianyu Wang and Gauri Joshi. Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. *Journal of Machine Learning Research*, 22:1–50, 2021. (Cited on pages 3 and 17.)

Jianyu Wang, Hao Liang, and Gauri Joshi. Overlap local-SGD: An algorithmic approach to hide communication delays in distributed SGD. *arXiv preprint arXiv:2002.09539*, 2020a. (Cited on pages 2, 3, 8, 18, 19, and 64.)

Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat. SlowMo: Improving communication-efficient distributed SGD with slow momentum. In *International Conference on Learning Representations*, 2020b. (Cited on page 20.)

Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Blaise Agüera y Arcas, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021. (Cited on page 1.)

Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. Gradient sparsification for communication-efficient distributed optimization. *Advances in Neural Information Processing Systems*, 31, 2018. (Cited on pages 2, 3, and 17.)

Blake Woodworth, Kumar Kshitij Patel, Sebastian Stich, Zhen Dai, Brian Bullins, Brendan McMahan, Ohad Shamir, and Nathan Srebro. Is local SGD better than minibatch SGD? In *International Conference on Machine Learning*, pages 10334–10343. PMLR, 2020. (Cited on pages 3 and 17.)

Hao Yu, Sen Yang, and Shenghuo Zhu. Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5693–5700, 2019. (Cited on pages 3 and 17.)

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023. (Cited on page 1.)

Yuhao Zhou, Qing Ye, and Jiancheng Lv. Communication-efficient federated learning with compensated Overlap-FedAvg. *IEEE Transactions on Parallel and Distributed Systems*, 33(1):192–205, 2022. doi: 10.1109/TPDS.2021.3090331. (Cited on page 18.)

Ligeng Zhu, Hongzhou Lin, Yao Lu, Yujun Lin, and Song Han. Delayed gradient averaging: Tolerate the communication latency for federated learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021. (Cited on page 18.)

Martin Zinkevich, Markus Weimer, Lihong Li, and Alex Smola. Parallelized stochastic gradient descent. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. (Cited on pages 2, 3, and 17.)

Contents

1 Introduction **1**

1.1 Contributions . . . 3

1.2 Related work . . . 3

2 Problem setup **4**

3 Method **5**

4 Theory **6**

5 Experiments **8**

5.1 Mechanism: overlap and correction . . . 8

5.2 Neural networks . . . 8

5.3 Comparison with external overlap methods . . . 8

5.4 Real multi-node hardware . . . 10

5.5 Limitations and failure regimes . . . 10

6 Conclusion **10**

A Extended related work and method comparisons **17**

B Additional theoretical results **21**

B.1 Explicit dependence on the communicated fraction $p = K/d$ . . . 21

B.2 Comparison with an optimized blocking baseline . . . 22

B.3 Linking sparsity to communication time: a latency-bandwidth specialization . . . 23

B.4 An idle-time lower bound for blocking . . . 24

C Proofs **25**

C.1 Within-round dynamics . . . 25

C.2 Exact evolution of the average . . . 26

C.3 Variance reduction by worker averaging . . . 26

C.4 Within-round drift bounds . . . 27

C.5 Disagreement recursion . . . 29

C.6 One-round descent inequality . . . 32

C.7 Proof of Theorem 4.1 . . . 35

C.8 Proof of Corollary 4.2 . . . 37

D Additional experimental results **38**

D.1 Methodology and shared protocol . . . 38

D.2 Tasks, models, and configurations . . . 40

D.2.1 Logistic regression . . . 40

D.2.2 CIFAR-10 . . . 40
D.2.3 Tiny ImageNet . . . 41
D.3 Does overlap reduce training time, and why? . . . 41
D.3.1 Mechanism on logistic regression . . . 41
D.3.2 Neural networks . . . 43
D.3.3 What overlap buys: processed examples . . . 45
D.4 How do delay, local work, workers, and sparsity shape the gain? . . . 45
D.4.1 Communication delay and local work . . . 45
D.4.2 Worker count and compute heterogeneity . . . 47
D.4.3 A joint grid on Tiny ImageNet . . . 49
D.4.4 Communicated fraction . . . 55
D.5 When does delay correction help? . . . 56
D.6 What happens when worker data are heterogeneous? . . . 59
D.6.1 Logistic regression . . . 59
D.6.2 CIFAR-10 . . . 60
D.6.3 Summary . . . 64
D.7 How does LOSCAR SGD compare with existing overlap methods? . . . 64
D.8 Is a shared Rand-K support the right communication operator? . . . 67
D.8.1 Shared versus independent Rand-K masks . . . 67
D.8.2 Alternative communication operators . . . 67
D.9 Do the gains hold on real hardware? . . . 69

A Extended related work and method comparisons

Communication overlap and preservation of local progress are not new in themselves. Table 1 describes algorithmic features, including reported variants; it does not assert that each combination has a joint convergence guarantee. We first place the method in the broader literature, then distinguish the closest update rules and the scope of their analyses.

Local training Local training reduces communication by performing multiple stochastic-gradient steps between synchronizations. FedAvg popularized this approach in federated learning [McMahan et al., 2017], building on earlier distributed-training methods [Zinkevich et al., 2010, Povey et al., 2014, Moritz et al., 2015]. Work addressing heterogeneous data includes [Li et al., 2020c, Karimireddy et al., 2020, Mishchenko et al., 2022b, Douillard et al., 2023]; these papers include both theoretical and empirical studies. Our shared-objective assumptions instead connect most directly to homogeneous-data Local SGD analyses [Yu et al., 2019, Haddadpour et al., 2019, Wang and Joshi, 2021, Woodworth et al., 2020]; see also Malinovsky et al. [2022].

Compression and sparsification Gradient compression includes quantization, sparsification, and biased compression mechanisms [Alistarh et al., 2017, Wangni et al., 2018, Lin et al., 2017, Beznosikov et al., 2023]. Closer to our setting are methods combining local training with partial model synchronization or compressed communication [Beton et al., 2025, Filippova et al., 2025, Douillard et al., 2025, Condat et al., 2025, 2026]. SPARTA asynchronously shares sparse subsets of model parameters to reduce worker divergence, whereas LoCoDL combines local training with unbiased compression, including sparsification and quantization [Beton et al., 2025, Condat et al., 2025]. Partial synchronization and quantization are distinct mechanisms; for example, the compression mark for Streaming DiLoCo in Table 1 refers to its quantized outer updates. Here the communicated object is a model to be synchronized, rather than a stochastic-gradient estimator. For a vector a independent of the sampled mask, the shared Rand-K projector has the properties

$$P_r^2 = P_r, \qquad \mathbb{E}[P_r] = pI, \qquad \mathbb{E}\|(I - P_r)a\|^2 = (1-p)\|a\|^2.$$

Selected coordinates are averaged exactly across all workers. Rescaling by $1/p$ changes selected-coordinate synchronization into an amplified update, so unbiasedness alone does not preserve the projector structure or the exact-averaging interpretation used by our analysis. Scaling both the averaged and local launch contributions consistently can preserve the pathwise mean, but changes the selected-coordinate update and the mask-contraction factor. Other compressors may require a different update, memory, or analysis, as illustrated by the operator ablations in Appendix D.8.

Communication overlap and delayed updates Overlap allows local computation to continue while synchronization is in flight. The comparisons below cover dense local-model averaging, delayed outer updates, and sharded-gradient methods. Let $P_r = \mathrm{Proj}_{S_r}$ be the shared coordinate projector. The LOSCAR SGD merge is

$$x_i^{r+1} = z_i^r + P_r(\bar{y}^r - y_i^r) = P_r\bar{y}^r + (I - P_r)y_i^r + (z_i^r - y_i^r).$$

It averages the launch models on selected coordinates while preserving each worker's realized displacement during communication; in particular, $\bar{x}^{r+1} = \bar{z}^r$ for every sampled mask. The correction uses quantities already stored locally and adds no communication. The same physical execution can be indexed from a merge boundary (local computation, launch, overlap, merge) or from a launch boundary (overlap, merge, local computation, launch); after initialization, shifting the round boundary changes neither the communicated models nor the local updates. Our analysis uses common logical merge boundaries and fixed completed-step counts, rather than arbitrary event-driven message arrivals.

Local training with overlap or compression

- CoCoD-SGD [Shen et al., 2019]: uses $x_{t+k}^i = \widehat{x}_t + (\widehat{x}_{t+k}^i - x_t^i)$ with $\widehat{x}_t = \sum_i w_i x_t^i$. With equal weights and a common local-step count this is exactly the dense ($P_r = I$) endpoint of our merge. It proves finite-time smooth nonconvex convergence under bounded dissimilarity but does not analyze sparse shared-support synchronization or worker-specific overlap counts Q_i.

- Overlap-Local-SGD [Wang et al., 2020a]: after a local step $x_{k+1/2}^{(i)} = x_k^{(i)} - \gamma g_i(x_k^{(i)})$, its update at a synchronization boundary is

$$x_{k+1}^{(i)} = (1-\rho)x_{k+1/2}^{(i)} + \rho z_k.$$

The anchor z_k is refreshed by a parallel dense-averaging thread and used at the next pullback. This mixes toward a stale anchor rather than applying the exact synchronization displacement. Its guarantee assumes bounded objective heterogeneity, a common local period, and dense synchronization, with no sparse support or worker-specific Q_i.

- CO2 [Sun et al., 2024]: using its local/averaged endpoint notation, the delayed outer update has the form

$$\mu_t = \beta_c \mu_{t-1} + \Lambda_t^{-1}(x_{t-1,0} - x_{t-1,\ell}),$$
$$x_{t+1,0} = x_{t,0} - \alpha_c \,\mathrm{Clip}(\mu_t, \phi),$$

where Λ_t is its coordinate-wise staleness penalty and α_c, β_c are outer-optimizer parameters. This is delayed dense outer momentum rather than a delayed sparse model-space merge. Its analysis uses a constant staleness coefficient and omits clipping.

- AsyncMesh [Ajanthan et al., 2026]: let P project onto the shared delayed support, let $\widehat{w}_i^t$ be the pre-averaging local model, and write $\bar{w}^s = n^{-1}\sum_i \widehat{w}_i^s$. Its data-parallel correction is

$$Pd_i^t = (1-\lambda_t)Pd_i^{t-1} + \lambda_t P(\widehat{w}_i^t - \widehat{w}_i^{t-\tau}),$$
$$w_i^t = (I-P)\widehat{w}_i^t + P(\bar{w}^{t-\tau} + d_i^t).$$

At the no-smoothing boundary $\lambda_t = 1$, this is algebraically our merge when the snapshots, shared support, and averaging weights coincide. This identifies the merge operation, not the complete algorithms or guarantees. The paper specifies $0 < \lambda_t < 1$; its asymptotic consensus theorem assumes a homogeneous setup, bounded expected model displacement, and diminishing-EMA and drift conditions, including $\sum_t \lambda_t = \infty$ and $\sum_t \lambda_t^2 < \infty$. Thus the boundary observation is not covered by that theorem, which also does not provide our finite-time stationarity and wall-clock bound with explicit N_i, Q_i.

- Qsparse-Local-SGD [Basu et al., 2019]: in its synchronous version, let u_t be the server model, $\widehat{x}_{i,t+1/2}$ the local model after an SGD step, $e_{i,t}$ its residual, and C its compressor. At communication times,

$$v_{i,t} = e_{i,t} + u_t - \widehat{x}_{i,t+1/2}, \qquad c_{i,t} = C(v_{i,t}),$$
$$e_{i,t+1} = v_{i,t} - c_{i,t}, \qquad u_{t+1} = u_t - \frac{1}{n}\sum_i c_{i,t},$$

and all workers reset to $\widehat{x}_{i,t+1} = u_{t+1}$. Between communications, the server and residual stay fixed while local SGD continues. Its asynchronous variant uses each worker's last received anchor, updates the server with the communicating subset, and refreshes only those workers, under a shared local-iteration clock and bounded synchronization gaps. It therefore compresses accumulated model displacements with error feedback, but does not expose a physical in-flight communication duration through post-launch counts Q_i or establish the shared-support cancellation used here.

- Delayed Gradient Averaging [Zhu et al., 2021]: at receipt of a delayed average, its update can be written as

$$w_{k+1}^{(i)} = w_k^{(i)} - \eta\big(g_k^{(i)} - m_u^{(i)} + \bar{m}_u\big), \qquad \bar{m}_u = \frac{1}{m}\sum_j m_u^{(j)},$$

where $m_u^{(i)}$ is the accumulated gradient from worker i's earlier communication interval and m is the worker count. Between such receipts it performs ordinary local SGD. It replaces a stale local accumulated gradient by its delayed dense average, rather than applying our shared-support model-synchronization displacement.

- Compensated Overlap-FedAvg [Zhou et al., 2022]: its analyzed gradient compensation takes the form

$$c_{i,t} = g_{i,t-1} + \lambda g_{i,t-1} \odot g_{i,t-1} \odot (w_t - w_{t-1}),$$
$$w_{t+1} = w_t - \eta_t \sum_i a_i c_{i,t},$$

where $g_{i,t-1} = \nabla F_i(w_{t-1})$, a_i are aggregation weights, and $\odot$ is coordinate-wise multiplication. The correction approximates the fresh gradient through a diagonal Hessian surrogate; it is not the exact realized local displacement used in our merge. The practical algorithm additionally uses Nesterov acceleration, whereas the displayed equations describe its analyzed update.

Finite-time guarantees and iteration counters The prior overlap guarantees use different iterates and counters. For Overlap-Local-SGD, let m be the worker count, J the total number of local iterations, ℓ the common synchronization period, and $\rho \in (0, 1)$ the anchor-pullback coefficient. From a common initialization, under smoothness, unbiased gradients, bounded stochastic variance σ^2, and bounded average squared gradient dissimilarity κ^2, Theorem 1 of Wang et al. [2020a] uses $\gamma = L^{-1}\sqrt{m/J}$ and $J \geq 60m\ell^2/\rho^2$ to obtain

$$\frac{1}{J}\sum_{k=0}^{J-1}\mathbb{E}\|\nabla F(y_k)\|^2 \leq \frac{4L\Delta_{\mathrm{ol}}}{(1-\rho)\sqrt{mJ}} + \frac{2(1-\rho)\sigma^2}{\sqrt{mJ}} + \frac{2m\sigma^2}{J}\left(\frac{2\ell}{(2-\rho)\rho} - 1\right) + \frac{2m\ell^2\kappa^2}{\rho^2 J}. \tag{10}$$

Here $\Delta_{\mathrm{ol}} = F(y_0) - F_{\mathrm{inf}}$, and $y_k = (1-\rho)m^{-1}\sum_i x_k^{(i)} + \rho z_k$ is its anchor-weighted virtual iterate. For CO2, let g be the worker count, T the number of outer iterations, and ℓ the common inner-loop length. Under the parameter choices and iteration-budget conditions of its Theorem 1, with clipping omitted, Sun et al. [2024] establish

$$\frac{1}{T\ell}\sum_{t=0}^{T-1}\sum_{k=0}^{\ell-1}\mathbb{E}\|\nabla f(x_{t,k})\|^2 = \mathcal{O}\big((gT\ell)^{-1/2}\big) + \mathcal{O}\big(g\ell/T\big). \tag{11}$$

Its assumptions include smooth local objectives, bounded stochastic-gradient variance, a bounded variance for the aggregated update direction, and bounded average squared gradient dissimilarity. These prior works permit heterogeneous local objectives under their dissimilarity conditions, whereas our theorem assumes a shared objective. Moreover, J counts local iterations and $T\ell$ counts inner iterations, while our R counts communication rounds containing an average of $\bar{H}$ local steps per worker. The displayed rates therefore cannot be identified as the same bound by equating their iteration counters. Our analysis instead exposes shared-support sparsity and the worker-specific pre-communication and overlap counts separately.

DiLoCo and sharded-optimizer methods

- Streaming DiLoCo [Douillard et al., 2025]: communicates outer gradients for different model fragments at different times while training continues, and mixes the delayed outer update with the current local fragment. Its FP4 communication and heterogeneous-overlap-delay ablation explain the corresponding marks in Table 1. For one fragment, under outer SGD with unit step and a common previous outer fragment, the mixing rule becomes

  $$x_i^{r+1} = \bar{y}^r + \alpha(z_i^r - \bar{y}^r).$$

  Unlike our dense merge $\bar{y}^r + (z_i^r - y_i^r)$, it scales both the local displacement and pre-launch disagreement by α; no fixed α makes the two update rules identical in general. The cited work evaluates the streaming scheme empirically rather than providing a finite-time convergence bound for it.

- Eager and delayed DiLoCo [Kale et al., 2025]: at h-step boundaries, define the outer gradients before the outer update by $\Delta_i^t = \theta_i^{t-h} - \theta_i^t$ and $\Delta^t = n^{-1}\sum_i \Delta_i^t$. For $t > h$, the eager update uses

  $$\widetilde{\Delta}_i^t = \Delta^{t-h} + \frac{\Delta_i^t - \Delta_i^{t-h}}{n} = \frac{1}{n}\left(\Delta_i^t + \sum_{j\neq i}\Delta_j^{t-h}\right),$$
  $$\theta_i^t \leftarrow \mathrm{OuterOpt}(\theta_i^{t-h}, \widetilde{\Delta}_i^t).$$

The naive delayed variant instead supplies Δ^{t-h} to the outer optimizer. Eager DiLoCo combines the current local contribution with delayed remote contributions through a full-model outer optimizer, rather than our additive sparse model-space merge. The paper evaluates this update experimentally and gives no convergence theorem for it.

- ACCO [Nabli et al., 2025]: communicates accumulated stochastic gradients and their counts while computing new gradients, supporting sharded optimizer states. Its correction combines gradients evaluated at current and lookahead parameters. It establishes smooth nonconvex convergence for an idealized SGD recursion, not for independently trained local models with sparse model averaging.
- Decoupled DiLoCo [Douillard et al., 2026]: independent learners send parameter fragments and progress information to a central synchronizer, which performs quorum-based, token-weighted aggregation. Learners replace their fragments with returned updates. The work demonstrates robustness and training quality empirically, without a finite-time convergence guarantee for this scheme; conversely, our analysis does not cover quorum-based asynchronous aggregation.

Other communication-efficient optimizers SlowMo periodically applies a slow momentum update after several iterations of a base optimizer and proves smooth nonconvex convergence [Wang et al., 2020b]. MT-DAO combines slow- and fast-moving first-moment estimates, or a momentum estimate with the current gradient, to adapt optimizer dynamics to infrequent synchronization [Iacob et al., 2026a]. DES-LOC assigns independent synchronization periods to model parameters and momentum states [Iacob et al., 2026b]. LoRDO combines low-rank optimization with infrequent synchronization; its LoRDO-Global variant constructs projections from aggregated pseudo-gradients and uses a full-rank quasi-hyperbolic update to retain exploration outside the projected subspace [Jovanovic et al., 2026]. These methods address complementary aspects of communication-efficient training: momentum dynamics, optimizer-state synchronization, and low-rank updates. They are relevant precedents, but not controlled comparisons of the sparse delayed merge alone.

Asynchronous methods Asynchronous SGD applies worker updates without requiring all workers to finish a common synchronization round [Agarwal and Duchi, 2011, Niu et al., 2011, Maranjyan, 2025]. Recent analyses study how scheduling and stale gradients affect convergence and time complexity [Tyurin and Richtárik, 2023, Maranjyan et al., 2025a,d, Maranjyan and Richtárik, 2026, Mahran et al., 2026, Sadiev et al., 2026, Tovmasyan et al., 2026]. In LOSCAR SGD, workers instead receive a common sparse synchronization result after performing worker-specific amounts of local work; the result is delayed, but its use follows the timing model in Assumption 2.1. The method of Tyurin and Sivtsov [2026] combines local training with asynchronous model averaging without model sparsification.

Scope of the contribution These guarantees concern different algorithms and assumptions and should not be compared term by term. Our contribution is the joint finite-time convergence and wall-clock analysis of shared-support sparse model averaging, computation during communication, and worker-specific counts N_i, Q_i, using a displacement-preserving merge. This yields the separate dependence on S_N, S_Q, Ψ_H and worker speeds in Theorem 4.1 and corollary 4.2; it is not a novelty claim for overlap or delay correction alone.

B Additional theoretical results

Here, we present the additional analyses discussed at the end of Section 4. The proofs of the main convergence and time-complexity results are collected in Appendix C. Each subsection is self-contained and uses the notation of Theorem 4.1 and Corollary 4.2: we write $\Delta \coloneqq f(x^0) - f^\star$, $p \coloneqq K/d$, $q \coloneqq 1-p$, $\tau_{\min} \coloneqq \min_i \tau_i$, and use the aggregate quantities $\bar{H}, H_{\max}, S_N, S_Q, \Psi_H$ and

$$X \coloneqq \Psi_H + H_{\max}\frac{BS_N + DS_Q}{1-c}, \qquad a \coloneqq \sqrt{X/\bar{H}}.$$

It is convenient to record explicit sufficient-time bounds from Theorem 4.1. Choose

$$\eta = \min\left\{\frac{\varepsilon n}{16L\sigma^2}, \sqrt{\frac{\varepsilon n\bar{H}}{24L^2G^2X}}, \frac{1}{8LH_{\max}}\right\}, \qquad R_0 = \frac{8\Delta}{\eta\bar{H}\varepsilon}.$$

Entries with zero denominators are interpreted as $+\infty$. The stochastic and disagreement terms are then at most $\varepsilon/4$ each, and $R \geq R_0$ makes the optimization term at most $\varepsilon/2$. Before integer rounding, the sufficient time is $\widehat{T} = (M\tau+\zeta)R_0 = \tau_H \max\{C_0/\varepsilon^2, C_1 a/\varepsilon^{3/2}, C_2 H_{\max}/\varepsilon\}$. It is bounded above by the sum expression

$$\begin{aligned} T &= \tau_H\left[\frac{C_0}{\varepsilon^2} + \frac{C_1\, a}{\varepsilon^{3/2}} + \frac{C_2 H_{\max}}{\varepsilon}\right] \\ C_0 &\coloneqq \frac{128\,\Delta L\sigma^2}{n}, \quad C_1 \coloneqq \frac{8\sqrt{24}\,\Delta LG}{\sqrt{n}}, \quad C_2 \coloneqq 64\,\Delta L. \end{aligned} \tag{12}$$

The three terms are the stochastic (minibatch) term, the disagreement term, and the stepsize-ceiling term, respectively. Taking $R = \max\{1, \lceil R_0 \rceil\}$ adds at most one round duration to $\widehat{T}$. The sparsity and optimized-blocking comparisons below concern these unrounded upper-bound expressions, not measured runtimes.

B.1 Explicit dependence on the communicated fraction $p = K/d$

The parameters α, β in Theorem 4.1 are proof parameters, not algorithmic hyperparameters, and can be optimized in closed form. We fix the round structure (the counts N_i, Q_i, hence $\bar{H}, H_{\max}, S_N, S_Q$) and make the dependence on the sparsity level $p = K/d$ explicit. Writing $q = 1-p$, the disagreement factor entering X is

$$F_p(\alpha,\beta) \coloneqq \frac{BS_N + DS_Q}{1-c}, \qquad B = q(1+\beta)\Big(1+\tfrac{1}{\alpha}\Big), \quad D = 1+\tfrac{q}{\beta}, \quad c = q(1+\alpha)(1+\beta).$$

Corollary B.1 (Optimized sparsity dependence)**.** *Let $S_N, S_Q > 0$ and $0 < p < 1$, and set $r \coloneqq \sqrt{S_Q/S_N}$. Over the feasible set $\{\alpha > 0, \beta > 0, c < 1\}$, the minimum of F_p is attained at*

$$\alpha^\star = \frac{p}{q(1+r) + \sqrt{q(1+r)^2 + p^2r^2}}, \qquad \beta^\star = \frac{r\,\alpha^\star}{1+\alpha^\star},$$

and equals

$$F_p^\star = \frac{S_N}{p^2}\Big[q(1+r) + \sqrt{q(1+r)^2 + p^2r^2}\Big]^2.$$

In particular, as $p \to 0$,

$$F_p^\star = \frac{4\big(\sqrt{S_N} + \sqrt{S_Q}\big)^2}{p^2} + O\Big(\tfrac{1}{p}\Big).$$

Thus the optimized disagreement factor scales as p^{-2} and $\sqrt{F_p^\star}$ as p^{-1}. Consequently, using $X = \Psi_H + H_{\max}F_p^\star$ in (12), the leading ε^{-2} term is independent of p, the $\varepsilon^{-3/2}$ term scales as p^{-1}, and the ε^{-1} term is independent of p. For fixed $\varepsilon > 0$ and other problem parameters, the sufficient stepsize chosen above satisfies $\eta = \Theta(p)$ as $p \to 0$, and every $p > 0$ yields a valid finite stepsize and complexity guarantee.

Proof. Put $u := \alpha$ and $v := (1+\alpha)\beta$. The feasible region $\{\alpha > 0, \beta > 0, c < 1\}$ is equivalent to $\{u > 0, v > 0, u + v < p/q\}$, and

$$BS_N + DS_Q = q(1+u+v)\Big(\frac{S_N}{u} + \frac{S_Q}{v}\Big) + pS_Q, \qquad 1 - c = p - q(u+v).$$

For fixed $u+v$, Cauchy–Schwarz gives $\big(\frac{S_N}{u} + \frac{S_Q}{v}\big)(u+v) \geq (\sqrt{S_N} + \sqrt{S_Q})^2$, with equality if and only if $v/u = r$. Substituting $v = ru$ and minimizing the resulting one-dimensional function of u yields $\alpha^\star, \beta^\star$ and $F_p^\star$. The $p \to 0$ expansion follows by expanding the square root ($q \to 1$, bracket $\to 2(1+r)$). Finally, balancing the terms of (9) gives an admissible stepsize whose binding entry as $p \to 0$ is $\sqrt{\varepsilon n \bar{H}/(24L^2G^2X)} = \Theta(p)$, since $F_p^\star = \Theta(p^{-2})$. □

A direct sparsity bound For all $0 < p < 1$, the optimized factor also obeys

$$\sqrt{F_p^\star} = \frac{q(\sqrt{S_N} + \sqrt{S_Q}) + \sqrt{q(\sqrt{S_N} + \sqrt{S_Q})^2 + p^2 S_Q}}{p} \leq \frac{2(\sqrt{S_N} + \sqrt{S_Q})}{p}.$$

Indeed, $S_Q \leq (\sqrt{S_N} + \sqrt{S_Q})^2$ and $q + p^2 \leq 1$, so the square-root term in the numerator is at most $\sqrt{S_N} + \sqrt{S_Q}$. With $(\alpha, \beta) = (\alpha^\star, \beta^\star)$, we have $X = \Psi_H + H_{\max} F_p^\star$ and $\sqrt{X} \leq \sqrt{\Psi_H} + \sqrt{H_{\max} F_p^\star}$. The sufficient-time expression in (12) therefore satisfies

$$\begin{aligned} T \leq \tau_H \Bigg[&\frac{C_0}{\varepsilon^2} + \frac{C_1\sqrt{\Psi_H}}{\varepsilon^{3/2}\sqrt{\bar{H}}} + \frac{C_2 H_{\max}}{\varepsilon} \\ &+ \frac{2C_1\sqrt{H_{\max}}(\sqrt{S_N} + \sqrt{S_Q})}{p\varepsilon^{3/2}\sqrt{\bar{H}}} \Bigg]. \end{aligned} \tag{13}$$

This separates local drift from the explicit p^{-1} sparsification contribution and eliminates the free analysis parameters. Integer rounding adds at most one round duration. The displayed sparsity scaling holds for fixed local-step counts; under a system model $\zeta = \zeta(K)$, the step-count aggregates must also be evaluated at their resulting K-dependent values.

The p^{-2} scaling is already present in this bound without overlap: when $S_Q = 0$, taking the boundary infimum gives $F_p^\star = S_N(q + \sqrt{q})^2/p^2$. It is not a lower bound on actual disagreement. At fixed pre-communication work, S_N is independent of K and ζ, whereas $S_Q = \zeta^2 \sum_i \tau_i^{-2}$ decreases as the communication duration decreases. This reduction can offset part of the sparsification cost, but does not imply that the full factor $DS_Q/(1-c)$ decreases with p. At the dense endpoint $p = 1$, $F_1 = S_Q$.

B.2 Comparison with an optimized blocking baseline

Fix a sparsity level K with communication duration ζ_K, and let $a_{\mathrm{ov}} = \sqrt{X/\bar{H}}$ be the disagreement coefficient of LOSCAR SGD at a given operating point. A *blocking* baseline communicates the same K coordinates but performs no computation during communication: for any $M \geq 1$ with $A := M\tau$ it takes $N_i(A) = A/\tau_i$ local steps and then idles for ζ_K. Writing $\bar{N}(A) = n^{-1}\sum_{i=1}^n N_i(A)$, its iterate dynamics are the $Q_i = 0$ specialization of Theorem 4.1, so $S_Q = 0$ and its disagreement coefficient is $a_{\mathrm{bl}}(A) = \sqrt{X_{\mathrm{bl}}(A)/\bar{N}(A)}$ with $X_{\mathrm{bl}}(A) = \Psi_N(A) + N_{\max}(A)\, BS_N(A)/(1-c)$ (all quantities defined as in Theorem 4.1 with $N_i(A)$ in place of H_i); each of its rounds nevertheless lasts $A + \zeta_K$ in wall-clock. For $0 < \eta \leq 1/(8LN_{\max}(A))$ and feasible proof parameters, Theorem 4.1 gives the corresponding blocking convergence bound

$$\begin{aligned} \frac{1}{R}\sum_{r=0}^{R-1} \mathbb{E}\|\nabla f(\bar{x}^r)\|^2 \leq &\frac{4\Delta}{\eta\bar{N}(A)R} + \frac{4L\eta\sigma^2}{n} \\ &+ \frac{6L^2\eta^2G^2}{n\bar{N}(A)}\left[\Psi_N(A) + \frac{N_{\max}(A)BS_N(A)}{1-c}\right]. \end{aligned} \tag{14}$$

Here $\Psi_N(A) = \sum_i \sum_{t=0}^{N_i(A)-1} t^2$. By (12),

$$\widehat{T}_{\mathrm{ov}} = \tau_H \max\Big\{\tfrac{C_0}{\varepsilon^2}, \tfrac{C_1 a_{\mathrm{ov}}}{\varepsilon^{3/2}}, \tfrac{C_2 H_{\max}}{\varepsilon}\Big\}, \quad \widehat{T}_{\mathrm{bl}}(A) = \tau_H\Big(1 + \tfrac{\zeta_K}{A}\Big)\max\Big\{\tfrac{C_0}{\varepsilon^2}, \tfrac{C_1 a_{\mathrm{bl}}(A)}{\varepsilon^{3/2}}, \tfrac{C_2 A}{\tau_{\min}\varepsilon}\Big\}.$$

Thus LOSCAR SGD pays the overlap disagreement contained in a_{ov} (through S_Q), whereas blocking pays the idle-time factor $1 + \zeta_K/A$.

Proposition B.2 (A conditional strict improvement over optimized blocking). *Suppose $\Delta, \sigma^2 > 0$ and $\zeta_K > 0$. If the operating point of* LOSCAR SGD *satisfies the high-accuracy condition $\varepsilon \le \min\{(C_0/(C_1 a_{\mathrm{ov}}))^2,\ C_0/(C_2 H_{\max})\}$, then $\widehat{T}_{\mathrm{ov}} = \tau_H C_0/\varepsilon^2$ and, for every feasible blocking choice $A = M\tau$ ($M \in \mathbb{N}$) and every feasible α, β,*

$$\inf_{M,\alpha,\beta} \widehat{T}_{\mathrm{bl}} \ \ge\ \tau_H \Big[\frac{C_0}{\varepsilon^2} + \frac{C_2\zeta_K}{\tau_{\min}\varepsilon}\Big] \ >\ \frac{\tau_H C_0}{\varepsilon^2} = \widehat{T}_{\mathrm{ov}}.$$

Hence, in this explicit non-empty regime, a fixed LOSCAR SGD *configuration has a strictly smaller sufficient-time bound than the blocking bound optimized over its synchronization interval M and its proof parameters α, β (and over K, replacing ζ_K by $\zeta_{\min} = \min_{1\le K\le d}\zeta_K$, when $\zeta_{\min} > 0$). This is not a lower bound on blocking's actual runtime. If $\zeta_K = 0$, overlap reduces to blocking at the same local-work setting.*

Proof. Under the stated condition the first term of $\widehat{T}_{\mathrm{ov}}$ dominates the other two, giving $\widehat{T}_{\mathrm{ov}} = \tau_H C_0/\varepsilon^2$. Write $P = C_0/\varepsilon^2$ and $g(A) = C_2 A/(\tau_{\min}\varepsilon)$. Dropping the middle term of $\widehat{T}_{\mathrm{bl}}$ and using $(1+\zeta_K/A)\max\{P, g(A)\} \ge P + (\zeta_K/A)g(A)$, which holds by cases on whether $g(A) \gtrless P$, together with the identity $(\zeta_K/A)g(A) = C_2\zeta_K/(\tau_{\min}\varepsilon)$, independent of A, gives $\widehat{T}_{\mathrm{bl}}(A)/\tau_H \ge C_0/\varepsilon^2 + C_2\zeta_K/(\tau_{\min}\varepsilon)$ for every A. The right-hand side is independent of M, α, β; taking the infimum and using $\zeta_K > 0$ for strictness completes the proof. □

We note that this is a conditional, certificate-based comparison: it compares theorem-derived upper bounds in an explicit high-accuracy regime and does not assert general dominance. In particular, at moderate target accuracy the additional S_Q disagreement can make the blocking certificate smaller. A complementary lower bound that does not rely on the (possibly loose) upper bounds is given in Appendix B.4 below.

B.3 Linking sparsity to communication time: a latency-bandwidth specialization

Theorem 4.1 treats K and ζ as independent. In practice a sparser message is cheaper to transmit, so ζ should decrease as K decreases. Because Theorem 4.1 holds for *every* pair (K, ζ), any system model $\zeta = \zeta(K)$ can be substituted directly into (12); no new convergence proof is needed. Fix M, and for $K \in \{1, \dots, d\}$ with duration ζ_K set $p_K = K/d$, $Q_i(K) = \zeta_K/\tau_i$, $H_i(K) = (M\tau + \zeta_K)/\tau_i$,

$$\bar{H}_K = \frac{M\tau + \zeta_K}{\tau_H}, \quad H_{\max,K} = \frac{M\tau + \zeta_K}{\tau_{\min}}, \quad S_Q(K) = \zeta_K^2 \sum_i \tau_i^{-2},$$

and let X_K, $a_K = \sqrt{X_K/\bar{H}_K}$ and T_K be the corresponding quantities in (12) (with α, β optimized as in Corollary B.1 for $0 < p_K < 1$ and $S_Q(K) > 0$, using the boundary infimum when $S_Q(K) = 0$ and $F_1 = S_Q$ at $K = d$). Consider the latency-bandwidth model

$$\zeta_{\mathrm{sys}}(K) = \zeta_{\mathrm{lat}} + \frac{sK}{\nu} = \zeta_{\mathrm{lat}} + \frac{sd}{\nu}p_K, \qquad \zeta_K = \tau\Big\lceil \zeta_{\mathrm{sys}}(K)/\tau \Big\rceil,$$

where ν is the link bandwidth and s is the effective number of bits per retained coordinate, including its value and coordinate index when sent. Fixed encoding, aggregation, and broadcast costs can enter ζ_{lat}; costs proportional to the retained-coordinate count can be absorbed into s. Dense communication is captured by $K = d$, and the best value of this sufficient-time expression within the model can be selected by the finite search $K^\star \in \arg\min_{1\le K\le d} T_K$.

Corollary B.3 (A regime where the sparse bound is smaller). *Assume $\Delta > 0$. For any sparse choice $K < d$,*

$$T_K - T_d = \tau_H\, \varepsilon^{-3/2}\Big[C_1(a_K - a_d) - \tfrac{C_2\sqrt{\varepsilon}}{\tau_{\min}}(\zeta_d - \zeta_K)\Big],$$

so $T_K < T_d$ if $C_1(a_K - a_d) < \frac{C_2\sqrt{\varepsilon}}{\tau_{\min}}(\zeta_d - \zeta_K)$. If $\zeta_K < \zeta_d$ and $a_K \le a_d$, this holds for every $\varepsilon > 0$; if $a_K > a_d$, it holds for

$$\varepsilon > \Big(\frac{C_1\tau_{\min}(a_K - a_d)}{C_2(\zeta_d - \zeta_K)}\Big)^2.$$

When both unrounded durations $\zeta_{\mathrm{sys}}(d)$ and $\zeta_{\mathrm{sys}}(K)$ are integer multiples of τ, $\zeta_d - \zeta_K = s(d-K)/\nu$. The comparison regime is non-empty whenever $\zeta_d > \zeta_K$.

Proof. The stochastic term C_0/ε^2 is independent of K and ζ, so it cancels in $T_K - T_d$. The stepsize-ceiling term contributes $C_2(H_{\max,K} - H_{\max,d})/\varepsilon = -C_2(\zeta_d - \zeta_K)/(\tau_{\min}\varepsilon)$, since $H_{\max,K} = (M\tau + \zeta_K)/\tau_{\min}$. Combining with the disagreement term $C_1(a_K - a_d)/\varepsilon^{3/2}$ and factoring $\varepsilon^{-3/2}$ gives the stated identity; the equivalence and the threshold follow immediately. □

The corollary gives an explicit regime in which the sparse sufficient-time bound is smaller than the dense bound; it does not order actual runtimes. The same latency-bandwidth model is used, at the level of the simulator, in our external-baseline experiments (cf. Appendix D.7).

B.4 An idle-time lower bound for blocking

The comparisons offered in Appendix B.2 above are between upper bounds. We complement them with a structural lower bound showing that a blocking method must waste wall-clock that overlap does not. Call a method *blocking* if each round contains $M\tau$ seconds of local computation followed by a communication phase of duration ζ_K during which no worker computes. Its iterate dynamics are the $Q_i = 0$ specialization of Algorithm 1.

Proposition B.4 (Idle-time lower bound)**.** *Under Assumptions 2.2 to 2.6, let $L, \Delta, \sigma^2 > 0$, with Δ interpreted here as an upper bound on initial suboptimality, and let $N_{\max} = \max_i M\tau/\tau_i$. There exist universal constants $c, c_0 > 0$ such that, for $0 < \varepsilon \le c_0 L\Delta$, every fixed-budget stochastic first-order blocking method that guarantees $\mathbb{E}\left[\|\nabla f(\bar{x})\|^2\right] \le \varepsilon$ over this class requires, on some sufficiently high-dimensional instance, at least*

$$R \;\ge\; R_\star := c\,\frac{\Delta L \sigma^2}{n\,\varepsilon^2\,N_{\max}}$$

rounds, and therefore incurs at least $T_{\mathrm{idle}} \ge R_\star \zeta_K$ of idle wall-clock. LOSCAR SGD *computes during every communication phase and incurs no idle time.*

Proof. Each of the R blocking rounds contains a communication phase of length ζ_K with no gradient computation, so its total idle time is $R\zeta_K$. Sequentializing the workers' gradient evaluations gives an adaptive stochastic first-order algorithm using at most $R\sum_i N_i \le RnN_{\max}$ independent oracle queries; communication and random masks add no oracle information. The randomized stochastic first-order lower bound of Arjevani et al. [2022], including its statistical-learning-oracle extension, requires at least $c\Delta L\sigma^2/\varepsilon^2$ such queries on some sufficiently high-dimensional instance in the stated accuracy regime. Indeed, our squared-gradient criterion implies $\mathbb{E}\|\nabla f(\bar{x})\| \le \sqrt{\varepsilon}$ by Jensen's inequality. Thus $RnN_{\max} \ge c\Delta L\sigma^2/\varepsilon^2$, yielding the rounds bound and its idle-time consequence. □

The bound is worst-case over the class and its constants are not optimized; it separates overlap from blocking only on the idle axis, since LOSCAR SGD converts the same ζ_K into computation that carries the S_Q disagreement cost of Theorem 4.1. It nonetheless establishes, by a lower bound rather than a certificate comparison, that blocking pays an unavoidable idle overhead that overlap avoids.

C Proofs

This section collects the proofs of the main technical results. The within-round drift quantity from (8) has the closed form

$$\Psi_H = \sum_{i=1}^{n} \sum_{t=0}^{H_i-1} t^2 = \sum_{i=1}^{n} \frac{(H_i-1)H_i(2H_i-1)}{6}.$$

We use the following notation throughout the proofs. Intermediate lemmas allow deterministic round-dependent stepsizes η_r; the main theorem uses $\eta_r \equiv \eta$. In round r, worker i starts from the local model $x_i^r \in \mathbb{R}^d$ it holds at the beginning of the round (see Algorithm 1) and performs H_i local steps, where H_i is defined later in (4). For round r and local step index t, let $w_{i,t}^r$ denote the point at which worker i evaluates its t-th stochastic gradient in that round, so that $w_{i,0}^r = x_i^r$. We also define the active worker set at local time t by

$$A_t \coloneqq \{i \in \{1,\dots,n\} : H_i > t\}, \qquad m_t \coloneqq |A_t|. \tag{15}$$

So A_t is exactly the set of workers that still perform a local step at local time t.

We define the sigma-field

$$\mathcal{F}_{r,t} \coloneqq \sigma\big(\{x_i^r\}_{i=1}^n,\ \{w_{i,s}^r, \xi_{i,s}^r :\ i \in A_s,\ s = 0,\dots,t-1\},\ \{w_{i,t}^r :\ i \in A_t\}\big)\,,$$

which contains the history up to local time t in round r, except for the fresh samples drawn at that time.

Under Assumptions 2.4 and 2.5, conditioned on $\mathcal{F}_{r,t}$, the random vectors

$$g_i(w_{i,t}^r, \xi_{i,t}^r) - \nabla f(w_{i,t}^r), \qquad i \in A_t\ ,$$

are independent and have conditional mean zero.

C.1 Within-round dynamics

At the beginning of round r, worker i holds a model $x_i^r \in \mathbb{R}^d$. Its local trajectory within the round is

$$w_{i,0}^r \coloneqq x_i^r, \qquad w_{i,t+1}^r = w_{i,t}^r - \eta_r\, g_i(w_{i,t}^r, \xi_{i,t}^r), \qquad t = 0,1,\dots,H_i-1. \tag{16}$$

We define

$$y_i^r \coloneqq w_{i,N_i}^r, \qquad z_i^r \coloneqq w_{i,H_i}^r\ . \tag{17}$$

We also define the round averages

$$\bar{x}^r \coloneqq \frac{1}{n}\sum_{i=1}^n x_i^r, \qquad \bar{y}^r \coloneqq \frac{1}{n}\sum_{i=1}^n y_i^r, \qquad \bar{z}^r \coloneqq \frac{1}{n}\sum_{i=1}^n z_i^r. \tag{18}$$

The delay-corrected merge is

$$x_i^{r+1} = \mathrm{Proj}_{S_r}\big(\bar{y}^r + z_i^r - y_i^r\big) + (I - \mathrm{Proj}_{S_r}) z_i^r = z_i^r + \mathrm{Proj}_{S_r}(\bar{y}^r - y_i^r), \tag{19}$$

where $S_r \subseteq [d]$ is a uniformly random subset of cardinality K, shared by all workers.

C.2 Exact evolution of the average

We now state and prove our first lemma, characterizing the exact evolution of the average $\bar{x}^r$.

Lemma C.1 (Exact evolution of the average)**.** *For every round r,*

$$\bar{x}^{r+1} = \bar{z}^r = \bar{x}^r - \eta_r G_r, \tag{20}$$

where

$$G_r \coloneqq \sum_{t=0}^{H_{\max}-1} \bar{g}_t^r, \qquad \bar{g}_t^r \coloneqq \frac{1}{n}\sum_{i\in A_t} g_i(w_{i,t}^r, \xi_{i,t}^r). \tag{21}$$

Proof. Averaging the corrected merge formula gives

$$\begin{aligned}
\bar{x}^{r+1} \overset{(18)}{=} \frac{1}{n}\sum_{i=1}^n x_i^{r+1} \quad &\overset{(19)}{=} \quad \frac{1}{n}\sum_{i=1}^n \Big(z_i^r + \mathrm{Proj}_{S_r}(\bar{y}^r - y_i^r)\Big) \\
&= \quad \frac{1}{n}\sum_{i=1}^n z_i^r + \frac{1}{n}\sum_{i=1}^n \mathrm{Proj}_{S_r}(\bar{y}^r - y_i^r) \\
&= \quad \bar{z}^r + \mathrm{Proj}_{S_r}\left(\frac{1}{n}\sum_{i=1}^n (\bar{y}^r - y_i^r)\right) \\
&= \quad \bar{z}^r + \mathrm{Proj}_{S_r}(0) = \bar{z}^r.
\end{aligned}$$

Next,

$$z_i^r \overset{(17)}{=} w_{i,H_i}^r \overset{(16)}{=} x_i^r - \eta_r \sum_{t=0}^{H_i-1} g_i(w_{i,t}^r, \xi_{i,t}^r). \tag{22}$$

Averaging over i, we get

$$\begin{aligned}
\bar{z}^r \quad &\overset{(18)}{=} \quad \frac{1}{n}\sum_{i=1}^n z_i^r \\
&\overset{(22)}{=} \quad \frac{1}{n}\sum_{i=1}^n x_i^r - \eta_r \frac{1}{n}\sum_{i=1}^n \sum_{t=0}^{H_i-1} g_i(w_{i,t}^r, \xi_{i,t}^r) \\
&\overset{(18)}{=} \quad \bar{x}^r - \eta_r \frac{1}{n}\sum_{i=1}^n \sum_{t=0}^{H_i-1} g_i(w_{i,t}^r, \xi_{i,t}^r) \\
&\overset{(15)}{=} \quad \bar{x}^r - \eta_r \sum_{t=0}^{H_{\max}-1} \frac{1}{n}\sum_{i\in A_t} g_i(w_{i,t}^r, \xi_{i,t}^r) \\
&\overset{(21)}{=} \quad \bar{x}^r - \eta_r G_r.
\end{aligned}$$

Combining the two identities proves the claim. □

C.3 Variance reduction by worker averaging

For $i \in A_t$, define

$$\varepsilon_{i,t}^r \coloneqq g_i(w_{i,t}^r, \xi_{i,t}^r) - \nabla f(w_{i,t}^r), \tag{23}$$

and further let

$$\bar{\varepsilon}_t^r \quad \coloneqq \quad \frac{1}{n}\sum_{i\in A_t} \varepsilon_{i,t}^r. \tag{24}$$

Plugging (23) into (24) gives

$$\begin{aligned}
\bar{\varepsilon}_t^r &= \frac{1}{n}\sum_{i\in A_t} \Big(g_i(w_{i,t}^r, \xi_{i,t}^r) - \nabla f(w_{i,t}^r)\Big) \\
&= \bar{g}_t^r - \frac{1}{n}\sum_{i\in A_t} \nabla f(w_{i,t}^r).
\end{aligned}$$

Lemma C.2 (Variance reduction at local time t)**.** *Under Assumptions 2.4 to 2.6 (independent samples, unbiased gradients, and bounded variance), for every round r and local time t, we have*

$$\mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2 \middle| \mathcal{F}_{r,t}\right] \leq \frac{m_t}{n^2}\sigma^2$$

and

$$\sum_{t=0}^{H_{\max}-1} \mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2 \middle| \{x_i^r\}_{i=1}^n\right] \leq \frac{\bar{H}}{n}\sigma^2.$$

Proof. Conditioned on $\mathcal{F}_{r,t}$, the vectors $\{\varepsilon_{i,t}^r, i \in A_t\}$ are independent and conditionally mean zero. Therefore,

$$\mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2 \middle| \mathcal{F}_{r,t}\right] \overset{(24)}{=} \frac{1}{n^2}\mathbb{E}\left[\left\|\sum_{i\in A_t}\varepsilon_{i,t}^r\right\|^2 \middle| \mathcal{F}_{r,t}\right] = \frac{1}{n^2}\sum_{i\in A_t}\mathbb{E}\left[\|\varepsilon_{i,t}^r\|^2 \middle| \mathcal{F}_{r,t}\right], \tag{25}$$

because the cross terms vanish. By Assumption 2.6, we have

$$\mathbb{E}\left[\|\varepsilon_{i,t}^r\|^2 \middle| \mathcal{F}_{r,t}\right] \leq \sigma^2.$$

Plugging this into (25), and recalling that $m_t = |A_t|$, we get

$$\mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2 \middle| \mathcal{F}_{r,t}\right] \leq \frac{m_t}{n^2}\sigma^2. \tag{26}$$

Summing the above inequality over $t \in \{0, \ldots, H_{\max}-1\}$ and using

$$\begin{aligned}
\sum_{t=0}^{H_{\max}-1} m_t &\overset{(15)}{=} \sum_{t=0}^{H_{\max}-1} |A_t| \\
&\overset{(15)}{=} \sum_{t=0}^{H_{\max}-1} |\{i \in \{1,\ldots,n\} : H_i > t\}| \\
&= \sum_{t=0}^{H_{\max}-1}\sum_{i=1}^{n} \mathbf{1}(H_i > t) \\
&= \sum_{i=1}^{n}\sum_{t=0}^{H_{\max}-1} \mathbf{1}(H_i > t) \\
&= \sum_{i=1}^{n} H_i \overset{(6)}{=} n\bar{H},
\end{aligned} \tag{27}$$

we obtain

$$\sum_{t=0}^{H_{\max}-1} \mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2 \middle| \mathcal{F}_{r,t}\right] \overset{(26)}{\leq} \frac{1}{n^2}\left(\sum_{t=0}^{H_{\max}-1} m_t\right)\sigma^2 \overset{(27)}{=} \frac{\bar{H}}{n}\sigma^2.$$

Taking conditional expectation with respect to $\{x_i^r\}_{i=1}^n$ and using the tower property, we obtain

$$\sum_{t=0}^{H_{\max}-1} \mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2 \middle| \{x_i^r\}_{i=1}^n\right] = \sum_{t=0}^{H_{\max}-1} \mathbb{E}\left[\mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2 \middle| \mathcal{F}_{r,t}\right] \middle| \{x_i^r\}_{i=1}^n\right] \leq \frac{\bar{H}}{n}\sigma^2.$$

□

C.4 Within-round drift bounds

Define

$$u_i^r := y_i^r - x_i^r, \qquad v_i^r := z_i^r - y_i^r, \qquad \bar{u}^r := \frac{1}{n}\sum_{i=1}^n u_i^r, \qquad \bar{v}^r := \frac{1}{n}\sum_{i=1}^n v_i^r. \tag{28}$$

Lemma C.3 (Within-round local drift)**.** *Let Assumption 2.4 (independent samples) and Assumption 2.7 (bounded second moment) hold. Then, for every round r, worker i, and local time $t \in \{0, \dots, H_i\}$, we have*

$$\mathbb{E}\left[\|w_{i,t}^r - x_i^r\|^2 \middle| \{x_j^r\}_{j=1}^n\right] \le \eta_r^2 t^2 G^2, \tag{29}$$

$$\mathbb{E}\left[\sum_{i=1}^n \|u_i^r - \bar{u}^r\|^2 \middle| \{x_j^r\}_{j=1}^n\right] \le \eta_r^2 G^2 S_N, \tag{30}$$

$$\mathbb{E}\left[\sum_{i=1}^n \|v_i^r - \bar{v}^r\|^2 \middle| \{x_j^r\}_{j=1}^n\right] \le \eta_r^2 G^2 S_Q. \tag{31}$$

Moreover,

$$\sum_{i=1}^n \sum_{t=0}^{H_i-1} \mathbb{E}\left[\|w_{i,t}^r - x_i^r\|^2 \middle| \{x_j^r\}_{j=1}^n\right] \le \eta_r^2 G^2 \Psi_H. \tag{32}$$

Proof. We begin with the identity

$$w_{i,t}^r - x_i^r \overset{(16)}{=} -\eta_r \sum_{s=0}^{t-1} g_i(w_{i,s}^r, \xi_{i,s}^r). \tag{33}$$

Using Cauchy–Schwarz, we obtain

$$\|w_{i,t}^r - x_i^r\|^2 \overset{(33)}{=} \eta_r^2 \left\|\sum_{s=0}^{t-1} g_i(w_{i,s}^r, \xi_{i,s}^r)\right\|^2 \le \eta_r^2\, t \sum_{s=0}^{t-1} \|g_i(w_{i,s}^r, \xi_{i,s}^r)\|^2. \tag{34}$$

Taking conditional expectation and using the tower property together with Assumption 2.7, we get

$$\mathbb{E}\left[\|g_i(w_{i,s}^r, \xi_{i,s}^r)\|^2 \middle| \{x_j^r\}_{j=1}^n\right] = \mathbb{E}\left[\mathbb{E}\left[\|g_i(w_{i,s}^r, \xi_{i,s}^r)\|^2 \middle| \mathcal{F}_{r,s}\right] \middle| \{x_j^r\}_{j=1}^n\right] \le G^2. \tag{35}$$

Substituting into (34), we obtain

$$\mathbb{E}\left[\|w_{i,t}^r - x_i^r\|^2 \middle| \{x_j^r\}_{j=1}^n\right] \overset{(34)+(35)}{\le} \eta_r^2 t^2 G^2. \tag{36}$$

Summing (36) over i and t, we get

$$\sum_{i=1}^n \sum_{t=0}^{H_i-1} \mathbb{E}\left[\|w_{i,t}^r - x_i^r\|^2 \middle| \{x_j^r\}_{j=1}^n\right] \overset{(36)}{\le} \eta_r^2 G^2 \sum_{i=1}^n \sum_{t=0}^{H_i-1} t^2 \overset{(8)}{=} \eta_r^2 G^2 \Psi_H.$$

Next, by definition,

$$u_i^r \overset{(28)}{=} y_i^r - x_i^r \overset{(17)}{=} w_{i,N_i}^r - x_i^r \overset{(16)}{=} -\eta_r \sum_{t=0}^{N_i-1} g_i(w_{i,t}^r, \xi_{i,t}^r). \tag{37}$$

Using Cauchy–Schwarz again, we obtain

$$\|u_i^r\|^2 \overset{(37)}{=} \eta_r^2 \left\|\sum_{t=0}^{N_i-1} g_i(w_{i,t}^r, \xi_{i,t}^r)\right\|^2 \le \eta_r^2 N_i \sum_{t=0}^{N_i-1} \|g_i(w_{i,t}^r, \xi_{i,t}^r)\|^2. \tag{38}$$

Taking conditional expectation and using (35), we get

$$\mathbb{E}\left[\|u_i^r\|^2 \middle| \{x_j^r\}_{j=1}^n\right] \overset{(38)+(35)}{\le} \eta_r^2 N_i^2 G^2. \tag{39}$$

Recall that $\bar{u}^r := \frac{1}{n}\sum_{i=1}^n u_i^r$. Since the variance of the vectors $\{u_i^r\}_{i=1}^n$ can be bounded by their second moment, i.e.,

$$\frac{1}{n}\sum_{i=1}^n \|u_i^r - \bar{u}^r\|^2 \le \frac{1}{n}\sum_{i=1}^n \|u_i^r\|^2, \tag{40}$$

we obtain

$$\mathbb{E}\left[\sum_{i=1}^{n}\|u_i^r-\bar{u}^r\|^2\,\middle|\,\{x_j^r\}_{j=1}^n\right] \overset{(40)+(39)}{\leq} \eta_r^2G^2\sum_{i=1}^{n}N_i^2 \overset{(7)}{=} \eta_r^2G^2S_N.$$

Finally, by definition,

$$v_i^r \overset{(28)}{=} z_i^r - y_i^r \overset{(17)}{=} w_{i,H_i}^r - w_{i,N_i}^r \overset{(16)}{=} -\eta_r\sum_{t=N_i}^{H_i-1} g_i(w_{i,t}^r,\xi_{i,t}^r), \tag{41}$$

which contains exactly $Q_i = H_i - N_i$ terms. Using (41) and Cauchy–Schwarz, we obtain

$$\|v_i^r\|^2 \overset{(41)}{=} \eta_r^2\left\|\sum_{t=N_i}^{H_i-1} g_i(w_{i,t}^r,\xi_{i,t}^r)\right\|^2 \leq \eta_r^2Q_i\sum_{t=N_i}^{H_i-1}\|g_i(w_{i,t}^r,\xi_{i,t}^r)\|^2. \tag{42}$$

Taking conditional expectation and using (35) and (3), we get

$$\mathbb{E}\left[\,\|v_i^r\|^2\,\middle|\,\{x_j^r\}_{j=1}^n\right] \overset{(42)+(35)+(3)}{\leq} \eta_r^2Q_i^2G^2. \tag{43}$$

Using

$$\sum_{i=1}^{n}\|v_i^r-\bar{v}^r\|^2 \leq \sum_{i=1}^{n}\|v_i^r\|^2, \tag{44}$$

which is an analogue of the bound (40) applied to the vectors $\{v_i^r\}_{i=1}^n$, we obtain

$$\mathbb{E}\left[\sum_{i=1}^{n}\|v_i^r-\bar{v}^r\|^2\,\middle|\,\{x_j^r\}_{j=1}^n\right] \overset{(44)+(43)}{\leq} \eta_r^2G^2\sum_{i=1}^{n}Q_i^2 \overset{(7)}{=} \eta_r^2G^2S_Q.$$

This proves (29)–(32). □

C.5 Disagreement recursion

Define the *disagreement* related to the vectors $\{x_i^r\}_{i=1}^n$ and $\{y_i^r\}_{i=1}^n$ as

$$X^r \coloneqq \sum_{i=1}^{n}\|x_i^r-\bar{x}^r\|^2, \qquad Y^r \coloneqq \sum_{i=1}^{n}\|y_i^r-\bar{y}^r\|^2. \tag{45}$$

Analogously, define the disagreement quantities

$$U^r \coloneqq \sum_{i=1}^{n}\|u_i^r-\bar{u}^r\|^2, \qquad V^r \coloneqq \sum_{i=1}^{n}\|v_i^r-\bar{v}^r\|^2, \qquad Z^r \coloneqq \sum_{i=1}^{n}\|z_i^r-\bar{z}^r\|^2. \tag{46}$$

Lemma C.4 (Disagreement recursion)**.** *Fix any parameters $\alpha > 0$ and $\beta > 0$, and define*

$$c \coloneqq q(1+\alpha)(1+\beta), \qquad B \coloneqq q(1+\beta)\left(1+\frac{1}{\alpha}\right), \qquad D \coloneqq 1+\frac{q}{\beta}.$$

Then, under Assumption 2.4 (mutual independence) and Assumption 2.7 (bounded second moment), for every round r, we have

$$\mathbb{E}\left[X^{r+1}\middle|\{x_i^r\}_{i=1}^n\right] \le cX^r + \eta_r^2 G^2\big(BS_N + DS_Q\big). \tag{47}$$

Proof. By Lemma C.1, we have $\bar{x}^{r+1} = \bar{z}^r$. Therefore, for each worker i,

$$\begin{aligned} x_i^{r+1} - \bar{x}^{r+1} &\overset{(19)}{=} z_i^r + \mathrm{Proj}_{S_r}(\bar{y}^r - y_i^r) - \bar{z}^r \\ &= z_i^r - \bar{z}^r - \mathrm{Proj}_{S_r}(y_i^r - \bar{y}^r) \\ &\overset{(28)}{=} (y_i^r - \bar{y}^r) + (v_i^r - \bar{v}^r) - \mathrm{Proj}_{S_r}(y_i^r - \bar{y}^r) \\ &= (I - \mathrm{Proj}_{S_r})(y_i^r - \bar{y}^r) + (v_i^r - \bar{v}^r). \end{aligned} \tag{48}$$

Because y_i^r and z_i^r are deterministic functions of the initial points x_i^0, historical masks, and data samples up to round r, Assumption 2.4 ensures that the mask S_r is independent of $\{y_i^r, z_i^r\}_{i=1}^n$. Including the starting models $\{x_i^r\}$ in the conditioning makes the tower-property step below valid. The mask's conditional distribution remains uniform and each coordinate is not selected with probability $q = 1 - p$. For any fixed vectors a, b, we have

$$\mathbb{E}\left[\|(I - \mathrm{Proj}_{S_r})a + b\|^2\right] = q\|a\|^2 + 2q\langle a, b\rangle + \|b\|^2.$$

Applying this with

$$a = y_i^r - \bar{y}^r, \qquad b = v_i^r - \bar{v}^r,$$

and using (48), we get

$$\begin{aligned} \mathbb{E}\left[\|x_i^{r+1} - \bar{x}^{r+1}\|^2\middle|\{x_i^r, y_i^r, z_i^r\}_{i=1}^n\right] &= q\|y_i^r - \bar{y}^r\|^2 + 2q\langle y_i^r - \bar{y}^r, v_i^r - \bar{v}^r\rangle + \|v_i^r - \bar{v}^r\|^2 \\ &= q\|(y_i^r - \bar{y}^r) + (v_i^r - \bar{v}^r)\|^2 + p\|v_i^r - \bar{v}^r\|^2 \\ &= q\|z_i^r - \bar{z}^r\|^2 + p\|v_i^r - \bar{v}^r\|^2. \end{aligned} \tag{49}$$

Summing (49) over i, we obtain

$$\begin{aligned} \mathbb{E}\left[X^{r+1}\middle|\{x_i^r, y_i^r, z_i^r\}_{i=1}^n\right] &= q\sum_{i=1}^n \|z_i^r - \bar{z}^r\|^2 + p\sum_{i=1}^n \|v_i^r - \bar{v}^r\|^2 \\ &\overset{(46)}{=} qZ^r + pV^r. \end{aligned} \tag{50}$$

Next, using the identity

$$z_i^r - \bar{z}^r = (y_i^r - \bar{y}^r) + (v_i^r - \bar{v}^r)$$

which follows from (28), we estimate $qZ^r + pV^r$. For any vectors a, b, we can estimate

$$\begin{aligned} q\|a+b\|^2 + p\|b\|^2 &= q\|a\|^2 + 2q\langle a, b\rangle + (q+p)\|b\|^2 \\ &= q\|a\|^2 + 2q\langle a, b\rangle + \|b\|^2 \\ &\le q(1+\beta)\|a\|^2 + \left(1+\frac{q}{\beta}\right)\|b\|^2, \end{aligned} \tag{51}$$

where we used the inequality

$$2q\langle a, b\rangle \le q\beta\|a\|^2 + \frac{q}{\beta}\|b\|^2.$$

Applying (51) with

$$a = y_i^r - \bar{y}^r, \qquad b = v_i^r - \bar{v}^r,$$

and summing over i gives

$$\begin{aligned} qZ^r + pV^r &\le q(1+\beta)Y^r + \left(1+\frac{q}{\beta}\right)V^r \\ &= q(1+\beta)Y^r + DV^r. \end{aligned} \tag{52}$$

It remains to control Y^r. Since

$$y_i^r - \bar{y}^r \overset{(28)}{=} (x_i^r - \bar{x}^r) + (u_i^r - \bar{u}^r),$$

using Young's inequality leads to

$$\begin{aligned} Y^r &\overset{(45)}{=} \sum_{i=1}^n \|y_i^r - \bar{y}^r\|^2 \\ &= \sum_{i=1}^n \|(x_i^r - \bar{x}^r) + (u_i^r - \bar{u}^r)\|^2 \\ &\leq \sum_{i=1}^n \left[(1+\alpha)\|x_i^r - \bar{x}^r\|^2 + \left(1 + \frac{1}{\alpha}\right)\|u_i^r - \bar{u}^r\|^2\right] \\ &\overset{(45)+(46)}{=} (1+\alpha)X^r + \left(1 + \frac{1}{\alpha}\right)U^r. \end{aligned} \tag{53}$$

Combining (50), (52), and (53), we obtain

$$\begin{aligned} \mathbb{E}\left[X^{r+1} \middle| \{x_i^r, y_i^r, z_i^r\}_{i=1}^n\right] &\overset{(50)}{=} qZ^r + pV^r \\ &\overset{(52)}{\leq} q(1+\beta)Y^r + DV^r \\ &\overset{(53)}{\leq} q(1+\beta)(1+\alpha)X^r + q(1+\beta)\left(1 + \frac{1}{\alpha}\right)U^r + DV^r \\ &= cX^r + BU^r + DV^r. \end{aligned} \tag{54}$$

Finally, taking conditional expectation with respect to $\{x_i^r\}_{i=1}^n$ and using (30)–(31), we get

$$\begin{aligned} \mathbb{E}\left[X^{r+1} \middle| \{x_i^r\}_{i=1}^n\right] &\overset{(54)}{\leq} cX^r + B\mathbb{E}\left[U^r \middle| \{x_i^r\}_{i=1}^n\right] + D\mathbb{E}\left[V^r \middle| \{x_i^r\}_{i=1}^n\right] \\ &\overset{(30)+(31)}{\leq} cX^r + \eta_r^2 G^2(BS_N + DS_Q). \end{aligned} \tag{55}$$

This proves (47). □

Corollary C.5 (Accumulated disagreement bound)**.** *Assume $x_i^0 = x^0$ for all i, so that $X^0 = 0$, and assume $\eta_r \equiv \eta$ for all r. If*

$$c = q(1+\alpha)(1+\beta) < 1,$$

then

$$\sum_{r=0}^{R-1} \mathbb{E}\left[X^r\right] \leq R\,\frac{\eta^2 G^2\left(BS_N + DS_Q\right)}{1-c} \tag{56}$$

for every $R \geq 1$.

Proof. Let $\delta^r := \mathbb{E}\left[X^r\right]$. Taking full expectation in (47), and using $\eta_r \equiv \eta$, gives

$$\delta^{r+1} \overset{(47)}{\leq} c\delta^r + C, \tag{57}$$

where $C := \eta^2 G^2(BS_N + DS_Q)$. Since $x_i^0 = x^0$ for all i, we have

$$X^0 = 0, \qquad \delta^0 = \mathbb{E}\left[X^0\right] = 0.$$

We now prove by induction that, for every $r \geq 0$,

$$\delta^r \leq C\sum_{j=0}^{r-1} c^j, \tag{58}$$

where the sum is interpreted as 0 when $r = 0$. For $r = 0$, (58) gives $\delta^0 \leq 0$, which holds since $\delta^0 = 0$. Suppose now that (58) holds for some $r \geq 0$. Then

$$\delta^{r+1} \overset{(57)}{\leq} c\delta^r + C \overset{(58)}{\leq} cC\sum_{j=0}^{r-1} c^j + C = C\sum_{j=1}^{r} c^j + C = C\sum_{j=0}^{r} c^j.$$

Thus (58) holds for all $r \geq 0$. Since $c < 1$, we have

$$\sum_{j=0}^{r-1} c^j \leq \frac{1}{1-c},$$

and therefore

$$\delta^r \overset{(58)}{\leq} \frac{C}{1-c} = \frac{\eta^2 G^2 (BS_N + DS_Q)}{1-c}. \tag{59}$$

Summing (59) over $r = 0, \ldots, R-1$, we get

$$\begin{aligned} \sum_{r=0}^{R-1} \mathbb{E}\left[X^r\right] &= \sum_{r=0}^{R-1} \delta^r \\ &\overset{(59)}{\leq} R \frac{\eta^2 G^2 (BS_N + DS_Q)}{1-c}. \end{aligned}$$

This proves (56). □

C.6 One-round descent inequality

Define

$$\bar{\nabla}_t^r \coloneqq \frac{1}{n} \sum_{i \in A_t} \nabla f(w_{i,t}^r), \qquad \bar{g}_t^r \coloneqq \frac{1}{n} \sum_{i \in A_t} g_i(w_{i,t}^r, \xi_{i,t}^r), \qquad G_r \coloneqq \sum_{t=0}^{H_{\max}-1} \bar{g}_t^r. \tag{60}$$

Lemma C.6 (One-round descent)**.** *Let Assumption 2.3 (smoothness), Assumption 2.4 (independent samples), Assumption 2.5 (unbiasedness), Assumption 2.6 (bounded variance), and Assumption 2.7 (bounded second moment) be satisfied. Then for every round r, we have*

$$\begin{aligned} \mathbb{E}\left[f(\bar{x}^{r+1}) \middle| \{x_i^r\}_{i=1}^n \right] \leq f(\bar{x}^r) &- \eta_r \left(\frac{\bar{H}}{2} - 2L\eta_r \bar{H} H_{\max} \right) \|\nabla f(\bar{x}^r)\|^2 \\ &+ \eta_r \frac{L^2 H_{\max}}{n} \left(1 + 4L\eta_r H_{\max}\right) X^r \\ &+ L\eta_r^2 \bar{H} \frac{\sigma^2}{n} \\ &+ L^2 \eta_r^3 G^2 \frac{\Psi_H}{n} \left(1 + 4L\eta_r H_{\max}\right). \end{aligned} \tag{61}$$

Proof. By Lemma C.1, we have

$$\bar{x}^{r+1} \overset{(20)}{=} \bar{x}^r - \eta_r G_r.$$

Hence, by L-smoothness of f (Assumption 2.3),

$$f(\bar{x}^{r+1}) \leq f(\bar{x}^r) - \eta_r \left\langle \nabla f(\bar{x}^r), G_r \right\rangle + \frac{L\eta_r^2}{2} \|G_r\|^2. \tag{62}$$

Taking conditional expectation given $\{x_i^r\}_{i=1}^n$, we obtain

$$\begin{aligned} \mathbb{E}\left[f(\bar{x}^{r+1}) \middle| \{x_i^r\}_{i=1}^n \right] \leq f(\bar{x}^r) &- \eta_r \mathbb{E}\left[\left\langle \nabla f(\bar{x}^r), G_r \right\rangle \middle| \{x_i^r\}_{i=1}^n \right] \\ &+ \frac{L\eta_r^2}{2} \mathbb{E}\left[\|G_r\|^2 \middle| \{x_i^r\}_{i=1}^n \right]. \end{aligned} \tag{63}$$

Step 1: lower bound the inner-product term. Using unbiasedness, we get

$$\mathbb{E}\left[\bar{g}_t^r \middle| \mathcal{F}_{r,t} \right] \overset{\text{Assumption 2.5}}{=} \bar{\nabla}_t^r, \tag{64}$$

and hence

$$\mathbb{E}\left[\left\langle \nabla f(\bar{x}^r), G_r \right\rangle \middle| \{x_i^r\}_{i=1}^n \right] \overset{(64)+(60)}{=} \sum_{t=0}^{H_{\max}-1} \mathbb{E}\left[\left\langle \nabla f(\bar{x}^r), \bar{\nabla}_t^r \right\rangle \middle| \{x_i^r\}_{i=1}^n \right]. \tag{65}$$

Further,

$$\left\langle \nabla f(\bar{x}^r), \bar{\nabla}_t^r \right\rangle \overset{(60)}{=} \frac{1}{n} \sum_{i \in A_t} \left\langle \nabla f(\bar{x}^r), \nabla f(w_{i,t}^r) \right\rangle. \tag{66}$$

Using the inequality

$$\langle a, b \rangle \geq \frac{1}{2}\|a\|^2 - \frac{1}{2}\|a-b\|^2,$$

with $a = \nabla f(\bar{x}^r),\ b = \nabla f(w_{i,t}^r)$, and then using L-Lipschitz continuity of ∇f (cf. Assumption 2.3), we obtain

$$\left\langle \nabla f(\bar{x}^r), \nabla f(w_{i,t}^r) \right\rangle \geq \frac{1}{2}\|\nabla f(\bar{x}^r)\|^2 - \frac{L^2}{2}\|w_{i,t}^r - \bar{x}^r\|^2. \tag{67}$$

Plugging (67) into (66), and using $m_t = |A_t|$, we get

$$\left\langle \nabla f(\bar{x}^r), \bar{\nabla}_t^r \right\rangle \quad \geq \quad \frac{m_t}{2n}\|\nabla f(\bar{x}^r)\|^2 - \frac{L^2}{2n} \sum_{i \in A_t} \|w_{i,t}^r - \bar{x}^r\|^2. \tag{68}$$

Taking conditional expectation and summing over t,

$$\begin{aligned}
\mathbb{E}\left[\langle \nabla f(\bar{x}^r), G_r \rangle \middle| \{x_i^r\}_{i=1}^n\right] &\overset{(65)+(68)}{\geq} \left(\frac{1}{2n}\sum_{t=0}^{H_{\max}-1} m_t\right)\|\nabla f(\bar{x}^r)\|^2 \\
&\quad -\frac{L^2}{2n}\sum_{i=1}^n \sum_{t=0}^{H_i-1} \mathbb{E}\left[\|w_{i,t}^r - \bar{x}^r\|^2 \middle| \{x_i^r\}_{i=1}^n\right] \\
&= \frac{\bar{H}}{2}\|\nabla f(\bar{x}^r)\|^2 \\
&\quad -\frac{L^2}{2n}\sum_{i=1}^n \sum_{t=0}^{H_i-1} \mathbb{E}\left[\|w_{i,t}^r - \bar{x}^r\|^2 \middle| \{x_i^r\}_{i=1}^n\right],
\end{aligned} \tag{69}$$

where in the last step we have used the identity

$$\sum_{t=0}^{H_{\max}-1} m_t \overset{(15)}{=} \sum_{i=1}^n H_i \overset{(6)}{=} n\bar{H}.$$

Next, using the decomposition

$$w_{i,t}^r - \bar{x}^r = (x_i^r - \bar{x}^r) + (w_{i,t}^r - x_i^r),$$

Young's inequality

$$\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2,$$

and Lemma C.3, we get

$$\mathbb{E}\left[\|w_{i,t}^r - \bar{x}^r\|^2 \middle| \{x_i^r\}_{i=1}^n\right] \leq 2\|x_i^r - \bar{x}^r\|^2 + 2\eta_r^2 t^2 G^2. \tag{70}$$

Summing over i and t, we obtain

$$\begin{aligned}
\sum_{i=1}^n \sum_{t=0}^{H_i-1} \mathbb{E}\left[\|w_{i,t}^r - \bar{x}^r\|^2 \middle| \{x_i^r\}_{i=1}^n\right] &\overset{(70)}{\leq} \sum_{i=1}^n \sum_{t=0}^{H_i-1} \left(2\|x_i^r - \bar{x}^r\|^2 + 2\eta_r^2 t^2 G^2\right) \\
&= 2\sum_{i=1}^n \sum_{t=0}^{H_i-1} \|x_i^r - \bar{x}^r\|^2 + 2\eta_r^2 G^2 \sum_{i=1}^n \sum_{t=0}^{H_i-1} t^2 \\
&\overset{(8)}{=} 2\sum_{i=1}^n H_i \|x_i^r - \bar{x}^r\|^2 + 2\eta_r^2 G^2 \Psi_H \\
&\leq 2H_{\max}\sum_{i=1}^n \|x_i^r - \bar{x}^r\|^2 + 2\eta_r^2 G^2 \Psi_H \\
&\overset{(45)}{=} 2H_{\max} X^r + 2\eta_r^2 G^2 \Psi_H.
\end{aligned} \tag{71}$$

Substituting (71) into (69), we get

$$\mathbb{E}\left[\langle \nabla f(\bar{x}^r), G_r \rangle \middle| \{x_i^r\}_{i=1}^n\right] \quad \geq \quad \frac{\bar{H}}{2}\|\nabla f(\bar{x}^r)\|^2 - \frac{L^2 H_{\max}}{n} X^r - \frac{L^2 \eta_r^2 G^2}{n} \Psi_H. \tag{72}$$

Step 2: upper bound the quadratic term. Let us write

$$G_r \overset{(60)+(24)}{=} \sum_{t=0}^{H_{\max}-1} \bar{\nabla}_t^r + \sum_{t=0}^{H_{\max}-1} \bar{\varepsilon}_t^r.$$

Using Young's inequality $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$, we get

$$\mathbb{E}\left[\|G_r\|^2 \middle| \{x_i^r\}_{i=1}^n\right] \leq 2\mathbb{E}\left[\left\|\sum_{t=0}^{H_{\max}-1} \bar{\nabla}_t^r\right\|^2 \middle| \{x_i^r\}_{i=1}^n\right] + 2\mathbb{E}\left[\left\|\sum_{t=0}^{H_{\max}-1} \bar{\varepsilon}_t^r\right\|^2 \middle| \{x_i^r\}_{i=1}^n\right]. \tag{73}$$

First, let's bound the first term on the right-hand side of (73). Using Jensen's inequality

$$\left\|\sum_{t=0}^{H_{\max}-1} a_t\right\|^2 \leq H_{\max} \sum_{t=0}^{H_{\max}-1} \|a_t\|^2,$$

and arguments identical to those leading to (71) (we repeat the same decomposition and apply Lemma C.3), we obtain the bound

$$\begin{aligned}\mathbb{E}\left[\left\|\sum_{t=0}^{H_{\max}-1} \bar{\nabla}_t^r\right\|^2 \middle| \{x_i^r\}_{i=1}^n\right] &\leq 2\bar{H}H_{\max}\|\nabla f(\bar{x}^r)\|^2 \\ &\quad + \frac{4L^2H_{\max}^2}{n}X^r + \frac{4L^2\eta_r^2G^2H_{\max}}{n}\Psi_H.\end{aligned} \tag{74}$$

Second, we bound the second term on the right-hand side of (73). We first explain why the cross terms vanish. Recall that

$$\bar{\varepsilon}_t^r = \frac{1}{n}\sum_{i\in A_t}\left(g_i(w_{i,t}^r, \xi_{i,t}^r) - \nabla f(w_{i,t}^r)\right).$$

By Assumption 2.5, and since $w_{i,t}^r$ is $\mathcal{F}_{r,t}$-measurable, we have

$$\mathbb{E}\left[\bar{\varepsilon}_t^r \middle| \mathcal{F}_{r,t}\right] = 0. \tag{75}$$

Moreover, for any $s < t$, the vector $\bar{\varepsilon}_s^r$ is $\mathcal{F}_{r,t}$-measurable, since it depends only on randomness revealed before time t. Therefore,

$$\begin{aligned}\mathbb{E}\left[\langle\bar{\varepsilon}_s^r, \bar{\varepsilon}_t^r\rangle \middle| \{x_i^r\}_{i=1}^n\right] &= \mathbb{E}\left[\mathbb{E}\left[\langle\bar{\varepsilon}_s^r, \bar{\varepsilon}_t^r\rangle \middle| \mathcal{F}_{r,t}\right] \middle| \{x_i^r\}_{i=1}^n\right] \\ &= \mathbb{E}\left[\langle\bar{\varepsilon}_s^r, \mathbb{E}\left[\bar{\varepsilon}_t^r \middle| \mathcal{F}_{r,t}\right]\rangle \middle| \{x_i^r\}_{i=1}^n\right] \\ &\overset{(75)}{=} 0.\end{aligned} \tag{76}$$

This shows that the sequence $\{\bar{\varepsilon}_t^r\}$ is a martingale difference sequence, and hence different-time noise terms are orthogonal in conditional expectation.

Now expand the squared norm:

$$\begin{aligned}\mathbb{E}\left[\left\|\sum_{t=0}^{H_{\max}-1} \bar{\varepsilon}_t^r\right\|^2 \middle| \{x_i^r\}_{i=1}^n\right] &= \mathbb{E}\left[\sum_{t=0}^{H_{\max}-1}\|\bar{\varepsilon}_t^r\|^2 + 2\sum_{0\leq s<t\leq H_{\max}-1}\langle\bar{\varepsilon}_s^r, \bar{\varepsilon}_t^r\rangle \middle| \{x_i^r\}_{i=1}^n\right] \\ &= \sum_{t=0}^{H_{\max}-1}\mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2 \middle| \{x_i^r\}_{i=1}^n\right] \\ &\quad + 2\sum_{0\leq s<t\leq H_{\max}-1}\mathbb{E}\left[\langle\bar{\varepsilon}_s^r, \bar{\varepsilon}_t^r\rangle \middle| \{x_i^r\}_{i=1}^n\right] \\ &\overset{(76)}{=} \sum_{t=0}^{H_{\max}-1}\mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2 \middle| \{x_i^r\}_{i=1}^n\right].\end{aligned} \tag{77}$$

Finally, applying Lemma C.2, we obtain

$$\begin{aligned}\mathbb{E}\left[\left\|\sum_{t=0}^{H_{\max}-1}\bar{\varepsilon}_t^r\right\|^2\middle|\{x_i^r\}_{i=1}^n\right] &\overset{(77)}{=} \sum_{t=0}^{H_{\max}-1}\mathbb{E}\left[\|\bar{\varepsilon}_t^r\|^2\middle|\{x_i^r\}_{i=1}^n\right]\\ &\overset{\text{Lemma C.2}}{\leq} \sum_{t=0}^{H_{\max}-1}\frac{m_t}{n^2}\sigma^2\\ &= \frac{\sigma^2}{n^2}\sum_{t=0}^{H_{\max}-1}m_t\\ &\overset{(27)}{=} \frac{\sigma^2}{n^2}\cdot n\bar{H} = \frac{\bar{H}}{n}\sigma^2. \end{aligned} \tag{78}$$

Combining (73), (74), and (78), we obtain

$$\mathbb{E}\left[\|G_r\|^2\middle|\{x_i^r\}_{i=1}^n\right] \leq 4\bar{H}H_{\max}\|\nabla f(\bar{x}^r)\|^2 + \frac{8L^2H_{\max}^2}{n}X^r \tag{79}$$

$$+ \frac{8L^2\eta_r^2G^2H_{\max}}{n}\Psi_H + 2\frac{\bar{H}}{n}\sigma^2. \tag{80}$$

Step 3: conclude. Substituting (72) and (80) into (63), and collecting terms, yields

$$\begin{aligned}\mathbb{E}\left[f(\bar{x}^{r+1})\middle|\{x_i^r\}_{i=1}^n\right] \leq f(\bar{x}^r) &- \eta_r\left(\frac{\bar{H}}{2} - 2L\eta_r\bar{H}H_{\max}\right)\|\nabla f(\bar{x}^r)\|^2\\ &+ \eta_r\frac{L^2H_{\max}}{n}\left(1+4L\eta_rH_{\max}\right)X^r + L\eta_r^2\bar{H}\frac{\sigma^2}{n}\\ &+ L^2\eta_r^3G^2\frac{\Psi_H}{n}\left(1+4L\eta_rH_{\max}\right),\end{aligned}$$

which is (61). □

C.7 Proof of Theorem 4.1

Theorem 4.1 (Convergence guarantee)**.** *Consider Algorithm 1, initialized with $x_i^0 = x^0 \in \mathbb{R}^d$, $i = 1,\dots,n$. Let Assumptions 2.2 to 2.7 hold. Choose a stepsize η satisfying $0 < \eta \leq 1/8LH_{\max}$, where $H_{\max}$ is defined in (6). Choose the sparsification level $K \in \{1,\dots,d\}$, let $q = 1 - \frac{K}{d}$ as in (5), and choose analysis parameters $\alpha > 0$ and $\beta > 0$ such that $c := q(1+\alpha)(1+\beta) < 1$. Further, define $B := q(1+\beta)\left(1+\frac{1}{\alpha}\right)$, $D := 1 + \frac{q}{\beta}$. Then for every $R \geq 1$, the average iterate $\bar{x}^r := \frac{1}{n}\sum_{i=1}^n x_i^r$ satisfies*

$$\begin{aligned}\frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}\left[\|\nabla f(\bar{x}^r)\|^2\right] \leq &\frac{4(f(x^0)-f^\star)}{\eta\bar{H}R} + \frac{4L\eta\sigma^2}{n}\\ &+\frac{6L^2\eta^2G^2\Psi_H}{n\bar{H}} + \frac{6L^2\eta^2G^2H_{\max}}{n\bar{H}}\frac{BS_N+DS_Q}{1-c},\end{aligned} \tag{9}$$

where $\bar{H}$ is defined in (6), S_N and S_Q are defined in (7), and Ψ_H is defined in (8).

Proof. Taking full expectation in Lemma C.6, and using $\eta_r \equiv \eta$, gives

$$\begin{aligned}\mathbb{E}\left[f(\bar{x}^{r+1})\right] \overset{(61)}{\leq} &\mathbb{E}\left[f(\bar{x}^r)\right] - \eta\left(\frac{\bar{H}}{2} - 2L\eta\bar{H}H_{\max}\right)\mathbb{E}\left[\|\nabla f(\bar{x}^r)\|^2\right]\\ &+\eta\frac{L^2H_{\max}}{n}\left(1+4L\eta H_{\max}\right)\mathbb{E}\left[X^r\right] + L\eta^2\bar{H}\frac{\sigma^2}{n}\\ &+L^2\eta^3G^2\frac{\Psi_H}{n}\left(1+4L\eta H_{\max}\right).\end{aligned} \tag{81}$$

Since $\eta \leq \frac{1}{8LH_{\max}}$, we have

$$4L\eta H_{\max} \leq \frac{1}{2}, \tag{82}$$

and therefore,

$$1 + 4L\eta H_{\max} \overset{(82)}{\leq} \frac{3}{2} \tag{83}$$

and

$$\frac{\bar{H}}{2} - 2L\eta\bar{H}H_{\max} = \bar{H}\left(\frac{1}{2} - 2L\eta H_{\max}\right) \overset{(82)}{\geq} \frac{\bar{H}}{4}. \tag{84}$$

Using (84) and (83) in (81), we obtain

$$\begin{aligned}\mathbb{E}\left[f(\bar{x}^{r+1})\right] \quad &\leq \quad \mathbb{E}\left[f(\bar{x}^r)\right] - \frac{\eta\bar{H}}{4}\mathbb{E}\left[\|\nabla f(\bar{x}^r)\|^2\right] \\ &\quad + \frac{3}{2}\eta\frac{L^2H_{\max}}{n}\mathbb{E}\left[X^r\right] + L\eta^2\bar{H}\frac{\sigma^2}{n} + \frac{3}{2}L^2\eta^3G^2\frac{\Psi_H}{n}.\end{aligned} \tag{85}$$

Rearranging (85), we get

$$\begin{aligned}\frac{\eta\bar{H}}{4}\mathbb{E}\left[\|\nabla f(\bar{x}^r)\|^2\right] \quad &\leq \quad \mathbb{E}\left[f(\bar{x}^r)\right] - \mathbb{E}\left[f(\bar{x}^{r+1})\right] + \frac{3}{2}\eta\frac{L^2H_{\max}}{n}\mathbb{E}\left[X^r\right] \\ &\quad + L\eta^2\bar{H}\frac{\sigma^2}{n} + \frac{3}{2}L^2\eta^3G^2\frac{\Psi_H}{n}.\end{aligned} \tag{86}$$

Summing (86) over $r = 0, \ldots, R-1$, we obtain

$$\begin{aligned}\frac{\eta\bar{H}}{4}\sum_{r=0}^{R-1}\mathbb{E}\left[\|\nabla f(\bar{x}^r)\|^2\right] \quad &\overset{(86)}{\leq} \quad \sum_{r=0}^{R-1}\left(\mathbb{E}\left[f(\bar{x}^r)\right] - \mathbb{E}\left[f(\bar{x}^{r+1})\right]\right) \\ &\quad + \frac{3}{2}\eta\frac{L^2H_{\max}}{n}\sum_{r=0}^{R-1}\mathbb{E}\left[X^r\right] \\ &\quad + RL\eta^2\bar{H}\frac{\sigma^2}{n} + \frac{3}{2}RL^2\eta^3G^2\frac{\Psi_H}{n} \\ &= \quad f(x^0) - \mathbb{E}\left[f(\bar{x}^R)\right] \\ &\quad + \frac{3}{2}\eta\frac{L^2H_{\max}}{n}\sum_{r=0}^{R-1}\mathbb{E}\left[X^r\right] \\ &\quad + RL\eta^2\bar{H}\frac{\sigma^2}{n} + \frac{3}{2}RL^2\eta^3G^2\frac{\Psi_H}{n}.\end{aligned} \tag{87}$$

By Assumption 2.2 (lower-boundedness), we get

$$f(x^0) - \mathbb{E}\left[f(\bar{x}^R)\right] \quad \leq \quad f(x^0) - f^\star. \tag{88}$$

Moreover, since $X^0 = 0$, $\eta_r \equiv \eta$, and $c < 1$, Corollary C.5 gives

$$\sum_{r=0}^{R-1}\mathbb{E}\left[X^r\right] \overset{Corollary\ C.5}{\leq} R\frac{\eta^2G^2(BS_N + DS_Q)}{1-c}. \tag{89}$$

Substituting (88) and (89) into (87), we get

$$\begin{aligned}\frac{\eta\bar{H}}{4}\sum_{r=0}^{R-1}\mathbb{E}\left[\|\nabla f(\bar{x}^r)\|^2\right] \quad &\leq \quad f(x^0) - f^\star + \frac{3}{2}R\eta^3\frac{L^2G^2H_{\max}}{n}\frac{BS_N + DS_Q}{1-c} \\ &\quad + RL\eta^2\bar{H}\frac{\sigma^2}{n} + \frac{3}{2}RL^2\eta^3G^2\frac{\Psi_H}{n}.\end{aligned} \tag{90}$$

Dividing (90) by $R\eta\bar{H}/4$, we obtain

$$\begin{aligned}\frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}\left[\|\nabla f(\bar{x}^r)\|^2\right] \quad &\leq \quad \frac{4(f(x^0) - f^\star)}{\eta\bar{H}R} + \frac{6L^2\eta^2G^2H_{\max}}{n\bar{H}}\frac{BS_N + DS_Q}{1-c} \\ &\quad + \frac{4L\eta\sigma^2}{n} + \frac{6L^2\eta^2G^2\Psi_H}{n\bar{H}}.\end{aligned}$$

Finally, reordering the terms gives (9). □

C.8 Proof of Corollary 4.2

Here we provide the proof of Corollary 4.2.

Proof. In the following bound, the constants from Theorem 4.1 are $c_0 = c_1 = 4$ and $c_2 = 6$. Under the assumptions of Theorem 4.1, we have

$$\mathbb{E}\left[\left\|\nabla f(\bar{x}^{\hat{r}})\right\|^2\right] = \frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}\left[\left\|\nabla f(\bar{x}^{r})\right\|^2\right] \leq c_0 \cdot \frac{\Delta}{\eta \bar{H} R} + c_1 \cdot \frac{\eta L \sigma^2}{n} + c_2 \cdot \eta^2 \frac{L^2 G^2 X}{n\bar{H}}.$$

For the variance term to satisfy $c_1 \frac{\eta L \sigma^2}{n} \leq \frac{\varepsilon}{3}$, we require $\eta \leq \frac{1}{3c_1}\frac{\varepsilon n}{L\sigma^2}$ when $\sigma^2 > 0$. For the disagreement drift term to satisfy $c_2 \eta^2 \frac{L^2 G^2 X}{n\bar{H}} \leq \frac{\varepsilon}{3}$, we require $\eta \leq \frac{1}{\sqrt{3c_2}}\frac{\sqrt{\varepsilon n \bar{H}}}{LG\sqrt{X}}$ when $X > 0$. A stepsize satisfying these conditions as well as the stepsize bound from Theorem 4.1, after substituting $c_1 = 4$ and $c_2 = 6$, is

$$\hat{\eta} = \min\left\{\frac{\varepsilon n}{12 L\sigma^2}, \sqrt{\frac{\varepsilon n \bar{H}}{18 L^2 G^2 X}}, \frac{1}{8 L H_{\max}}\right\}. \tag{91}$$

We use $\eta = \hat{\eta}$; entries with zero denominators are interpreted as $+\infty$.

To ensure the remaining term satisfies $c_0 \frac{\Delta}{\eta \bar{H} R} \leq \frac{\varepsilon}{3}$, we must have $R \geq \frac{3c_0\Delta}{\varepsilon \hat{\eta} \bar{H}}$. Substituting the evaluated components of $\hat{\eta}$ directly yields the explicit round bound:

$$R \geq c_R \left(\frac{\Delta L \sigma^2}{n\varepsilon^2 \bar{H}} + \frac{\Delta L G \sqrt{X}}{\varepsilon^{3/2}\bar{H}\sqrt{n\bar{H}}} + \frac{\Delta L H_{\max}}{\varepsilon \bar{H}}\right)$$

for an appropriate absolute constant $c_R > 0$.

To convert this round complexity to wall-clock time complexity, we multiply the total elapsed rounds R by the round duration, $T = M\tau + \zeta$. Using $\frac{T}{\bar{H}} = \tau_H \coloneqq \frac{n}{\sum_{i=1}^n \tau_i^{-1}}$, we obtain the stated unrounded bound:

$$\mathcal{O}\left(\frac{\Delta L \sigma^2}{n\varepsilon^2} \cdot \tau_H + \frac{\Delta L G \sqrt{X}}{\varepsilon^{3/2}\sqrt{n\bar{H}}} \cdot \tau_H + \frac{\Delta L H_{\max}}{\varepsilon} \cdot \tau_H\right).$$

Taking $R = \max\{1, \lceil 3c_0\Delta/(\varepsilon\hat{\eta}\bar{H})\rceil\}$ adds at most $M\tau + \zeta$ to this sufficient time. □

D Additional experimental results

This section gives the protocols behind Section 5 and the complete results, organized by the research question each experiment answers. Appendix D.1 describes the simulator, the matching protocol, the metrics, and the communication accounting shared by all experiments. Appendix D.2 describes each task, model, and configuration once. The remaining subsections answer, in turn:

- Does overlap reduce training time, and why? (Appendix D.3)
- How do communication delay, local work, worker count, compute heterogeneity, and sparsity shape the gain? (Appendix D.4)
- When does delay correction help? (Appendix D.5)
- What happens when worker data are heterogeneous? (Appendix D.6)
- How does LoCoSGD compare with existing overlap methods? (Appendix D.7)
- Is a shared Rand-K support the right communication operator? (Appendix D.8)
- Do the gains hold on real hardware? (Appendix D.9)

Table 3 lists every study with its replication level. We keep three kinds of evidence separate throughout: simulated logical time, processed optimization work, and measured wall-clock time. Our theory assumes homogeneous data: all workers draw samples from the same distribution $\mathcal{D}$ (Section 2). Its guarantees therefore do not cover the label-Dirichlet experiments, which are empirical stress tests only.

Table 3: Overview of the experimental studies. "Simulated" studies use the logical-time simulator; the hardware study measures wall-clock time. Seeds are paired across methods.

Study	Model	Workers	Methods	Replication	Sections
Mechanism and ablations	`a9a` logistic	4	LS, OW, DC	seed means, 95% CI	D.3.1, D.4.1, D.4.4, D.5, D.6.1
Sixteen-worker suite	`a9a` logistic	4, 16	LS, OW, DC	5 seeds, 105 runs	D.4.2, D.6.1
Normal / stress regimes	CIFAR-10 CNN	4	LS, OW, DC	3 seeds (+ seed 11)	D.3.2
Targeted suite	CIFAR-10 CNN	4, 16	LS, OW, DC	1 seed, 21 runs	D.4.2, D.6.2
Dirichlet sweep	CIFAR-10 CNN	4	LS, OW, DC	1 seed, 5 partitions	D.6.2
External baselines	CIFAR-10 CNN	16	6 methods	4 seeds, 120 runs	D.7
Index-aware follow-up	CIFAR-10 CNN	16	8 variants	2 seeds, 16 runs	D.7
Normal / stress regimes	Tiny ImageNet ResNet-18	4	LS, OW, DC	1 seed	D.3.2
Grid	Tiny ImageNet ResNet-18	4	LS, OW, DC	1 seed, 54 runs	D.4.3
Delay sweep and momentum	Tiny ImageNet ResNet-18	4	OW, DC	4 seeds	D.5
Mask policy	`a9a`; CIFAR-10 CNN	16	OW, DC	5 seeds; 1 seed	D.8.1
Communication operators	CIFAR-10 CNN	16	DC	4 seeds	D.8.2
Real hardware (measured)	CIFAR-10 CNN, ResNet18-GN	4 GPUs	LS, OW, DC	4 seeds, 190 runs	D.9

D.1 Methodology and shared protocol

Methods. The core comparisons use blocking Local Sparse (LS) and LoCoSGD with the **overwrite** (OW) or **delay-corrected** (DC) merge of Section 3. Merge-only and communication-operator ablations use the subsets stated in their respective sections. LS performs local drift and sparse model averaging and waits while communication is in flight. LS communicates the same shared Rand-K coordinates as LoCoSGD. Unless a study states otherwise, it also has the same round duration. Its compute window equals the overlap methods' pre-communication window $M\tau$ and is followed by an idle communication window of length ζ. Comparing OW with LS therefore measures the effect of computing during communication, and comparing DC with OW measures the effect of the merge rule. Appendix D.7 adds three external overlap methods.

Logical-time simulator. The simulator is not a real distributed system: workers, communication, and delays are simulated deterministically inside one process. This isolates local computation, sparsity, communication delay, worker-speed heterogeneity, and overlap from hardware scheduling and networking noise. Each worker is a separate PyTorch model copy, and the simulator explicitly tracks the round states x_i^r, y_i^r, and z_i^r. Logical time is controlled by integer worker step times τ_i. The simulator computes $\tau = \mathrm{lcm}(\tau_1, \ldots, \tau_n)$, uses a local compute window of length $M\tau$, and takes ζ to be a multiple of τ. This gives $N_i = M\tau/\tau_i$ steps before and $Q_i = \zeta/\tau_i$ steps during

communication, so faster workers perform more local computation in the same window. The server averages the sparse messages on a shared random mask, and the workers merge the result by overwrite or delay correction. Each mask selects $K = \lceil pd \rceil$ of the d model coordinates, so for small models the realized fraction K/d can exceed the nominal p. The codebase separates timing, mask sampling, compression, communication accounting, and merge rules. Besides the engines used here, it includes dense FedAvg-Full and Sync SGD engines. Experiments are driven by saved configurations, and each run records optimization and validation metrics, worker disagreement, processed examples, cumulative logical time, and logical communication in coordinates and bits.

Matching protocol. Within each paired comparison, methods share initialization, data split, worker partition, and random seeds. The core LS/OW/DC comparisons also match minibatch streams and shared masks; methods can consume different prefixes when their local-step counts differ. External baselines and mask/compressor ablations deliberately change the communication or update rule, as specified below. Multi-seed results pair the methods within each seed. Curves show seed means, and the meaning of the bands is given with each study. Tabulated uncertainty is the sample standard deviation unless stated otherwise. Single-seed results are labeled as such, and they do not estimate variability across initializations or partitions.

Horizontal axes. Round-based plots measure progress per synchronization event. Logical-time plots capture the purpose of overlap, which is to reduce idle communication time. *Processed examples* (equivalently, effective epochs) count the training examples consumed by all logical workers and ignore simulated communication time. This axis is a proxy for the number of stochastic-gradient steps and approximately for local floating-point work. It asks whether overlap only looks better because it processes more minibatches during communication, or whether those extra updates remain useful at the same amount of data. It does not equalize the number of sparse merges. When LS is matched only in pre-communication work, overlap generally merges less often at the same processed-example count. The external-baseline study gives all methods the same nominal total-training-work budget; the hardware study additionally matches LS to the overlap methods' total local work per round.

Communication accounting. Logical communication is a model-level count, not measured bandwidth, and three conventions appear in this appendix.

1. *Logical bits*, used for the `a9a` figures (Appendices D.3.1 and D.4.4), include coordinate indices. For a model with d coordinates and K active coordinates in a sparse round, each value counts as 32 bits. The sparse uplink adds $\lceil \log_2 d \rceil$ index bits per active coordinate, so the n-worker uplink costs $nK(32 + \lceil \log_2 d \rceil)$ bits. The downlink broadcasts the aggregated values and counts $nK \cdot 32$ bits, multiplied by a downlink coefficient equal to 1 in our experiments. Dense communication sends all d values without index bits.
2. *Transmitted model values*, used for the headline payload totals of the external-baseline comparison (Appendix D.7), count only the 32-bit values and omit coordinate indices. A shared support can be regenerated from a seed, but these totals are an accounting convention, not a measurement of a particular wire format. A follow-up study in the same section charges an explicit 22-bit index ($\lceil \log_2 d \rceil$ for the 2.19M-parameter CNN) per sparse uplink value; the downlink counts values only. Its communication durations are recomputed from this index-aware payload.
3. *Bidirectional bits per worker per round*, used to match the communication operators of Appendix D.8.2, count each operator's own values, indices, and scales, as described there.

None of the conventions charges the shared-mask seed, transport headers, or local optimizer states such as momentum buffers. The hardware study (Appendix D.9) measures time rather than bits.

Trajectory summaries. For a set of compared methods, let t_m^- and t_m^+ be the first and last recorded times of method m on the chosen axis, and let $T_0 \coloneqq \max_m t_m^-$ and $T_1 \coloneqq \min_m t_m^+$. The *normalized AUC* of an accuracy curve a_m is

$$\mathrm{nAUC}_m \coloneqq \frac{1}{T_1 - T_0} \int_{T_0}^{T_1} a_m(t)\, dt,$$

the accuracy averaged over the common horizon. Unless stated otherwise, it uses validation accuracy and untransformed simulated logical time. Logarithmic axes in figures are only a visualization choice.

Because overlap methods can exhaust a fixed data budget in less logical time, their curves may end earlier. We therefore restrict to the common horizon. Dashed vertical lines in the figures mark common horizons. For a target accuracy q, the *speedup* of method m is $T_{\mathrm{LS}}(q)/T_m(q)$, where $T_m(q)$ is the first time m reaches q. Values above one favor m. Targets are absolute accuracies shared by the compared methods, except for the explicitly identified relative-progress targets in the rapidly saturating logistic-regression scaling suite.

D.2 Tasks, models, and configurations

D.2.1 Logistic regression

We use binary logistic regression on `a9a` from LIBSVM with labels in $\{-1, +1\}$. For an example (x, y), the loss is $\ell(w; x, y) = \log(1 + \exp(-yx^\top w))$, and the objective is

$$F(w) = \sum_{i=1}^{n} \frac{m_i}{m} F_i(w), \qquad F_i(w) = \frac{1}{m_i} \sum_{r=1}^{m_i} \ell(w; x_{i,r}, y_{i,r}) + R(w), \qquad m = \sum_{i=1}^{n} m_i.$$

For the nonconvex variant, we use the coordinate-wise Geman–McClure regularizer $R(w) = \lambda \sum_{j=1}^{d} w_j^2/(w_j^2 + \theta^2)$. The nonconvex runs produced the same qualitative ordering as the unregularized runs, so we report only the latter ($\lambda = 0$). Features are standardized with training-set statistics, 10% of the training data is held out for validation, and worker shards use a fixed partitioning seed. We report training and validation loss and accuracy and the full training gradient norm $\|\nabla F(w)\|$, which measures stationarity. In the loss above, x includes a constant feature for the bias. The linear classifier therefore has $d = 124$ coordinates (123 features and the bias), and each mask selects $K = \lceil pd \rceil$ of them. Unless stated otherwise, runs use four workers with step times $(1, 2, 3, 6)$, hence $\tau = 6$, 20 rounds, learning rate $\eta = 0.1$, batch size 256, and $p = 0.3$ ($K = 38$). These four-worker runs report means over seeds, with bands showing 95% confidence intervals (± 1.96 standard errors).

Sixteen-worker suite. A second set of runs tests whether the mechanism survives more workers and harder timing. It comprises 105 runs: seven configurations, three methods, and five paired seeds (11–15). We checked that every comparison shares its configuration and realized worker partition. All configurations use $p = 0.30$ ($K = 38$) and $M = 8$. The homogeneous configurations set $\tau_i = 8$ for every worker. They use four or sixteen workers with $\zeta = 8$, sixteen workers with label-Dirichlet partitions ($\alpha \in \{0.1, 0.03\}$) and $\zeta = 8$, or sixteen workers with a communication-stress delay $\zeta = 32$. The compute-heterogeneous configurations use sixteen workers, four at each $\tau_i \in \{1, 2, 4, 8\}$. They use either $\zeta = 8$ or, in the combined stress, $\zeta = 32$ and a Dirichlet partition with $\alpha = 0.1$. These are the same seven configuration types as the CIFAR-10 targeted suite (Table 4). Logistic-regression accuracy saturates within a few rounds. We therefore summarize this suite by validation nAUC and by mean paired speedups to progress targets. The targets are placed at 50%, 75%, 90%, and 95% of the progress that each paired LS run makes from initialization to its best validation accuracy.

D.2.2 CIFAR-10

On CIFAR-10 [Krizhevsky et al., 2009], the model is a convolutional classifier with 2.19M parameters, trained with cross-entropy, weight decay $5 \cdot 10^{-4}$, dropout 0.1, batch size 256, random crops with padding 4, and horizontal flips. We hold out 10% of the training set for validation and evaluate with batch size 512. Values are simulated as 32-bit numbers. Unless stated otherwise, all logical workers sample from the same training distribution. This matches the theory and focuses on optimization and communication effects rather than client drift. All CIFAR-10 runs use a budget of 400 effective epochs.

Four-worker regimes. Four workers with step times $(1, 2, 3, 6)$ use learning rate 0.05 in a normal regime ($p = 0.25$, $M = 4$, $\zeta = 6$) and in a communication-stress regime ($p = 0.10$, $M = 2$, $\zeta = 24$). The stress regime makes communication a larger share of the time budget. Both regimes were run with three paired seeds $0, 1, 2$, for $2 \times 3 \times 3 = 18$ runs. Representative trajectories use seed 11.

Targeted suite. A targeted suite of 21 runs fixes $p = 0.25$ and $M = 4$ and changes the worker count, the delay, the worker speeds, and the data partition one at a time. It covers the seven configurations of

Table 4, three methods, and one paired seed, 11. In the table, $c^{\times n}$ denotes n workers with step time c, and $(1, 2, 4, 8)^{\times 4}$ denotes four workers at each step time. Hence $\tau = 6$ in the homogeneous rows and $\tau = 8$ in the heterogeneous rows. The same configuration types define the sixteen-worker logistic-regression suite and the CIFAR-10 settings of the mask and operator ablations (Appendix D.8). The targeted suite is single-seed, so its differences carry no uncertainty estimate.

Table 4: Configurations of the targeted CIFAR-10 suite. τ_i is worker i's step time and ζ the synchronization delay. All rows use 400 effective epochs and one paired seed.

Label	Workers	Data partition	$\{\tau_i\}$	p	M	ζ
Clean, 4 workers	4	IID	$6^{\times 4}$	0.25	4	6
Clean, 16 workers	16	IID	$6^{\times 16}$	0.25	4	6
Communication stress	16	IID	$6^{\times 16}$	0.25	4	24
Compute heterogeneity	16	IID	$(1, 2, 4, 8)^{\times 4}$	0.25	4	8
Dirichlet $\alpha = 0.1$	16	label-Dirichlet, $\alpha = 0.1$	$6^{\times 16}$	0.25	4	6
Dirichlet $\alpha = 0.03$	16	label-Dirichlet, $\alpha = 0.03$	$6^{\times 16}$	0.25	4	6
Combined stress	16	label-Dirichlet, $\alpha = 0.1$	$(1, 2, 4, 8)^{\times 4}$	0.25	4	32

D.2.3 Tiny ImageNet

Tiny ImageNet has 200 classes at resolution 64×64. We optimize on the official training split, hold out 10% of it for validation, and reserve the official validation split for final testing. Images are normalized with ImageNet statistics. The model is a ResNet-18-style network trained from scratch [He et al., 2016]. For 64×64 inputs, it uses a 3×3 stride-1 stem without the initial max-pooling layer, a 200-class linear classifier, and no dropout. It uses GroupNorm instead of BatchNorm, because BatchNorm running statistics are not synchronized by sparse parameter averaging. All methods use 100 effective epochs, batch size 128, SGD with momentum 0.9, weight decay $5 \cdot 10^{-4}$, and initial learning rate 0.1 with cosine decay to zero. Training uses random crops with padding 4 and horizontal flips, and evaluation uses unaugmented images. Workers share the data distribution.

Two representative regimes. With four workers and step times $(1, 2, 3, 6)$, the normal regime uses $p = 0.25$, $M = 4$, and $\zeta = 6$, so LS computes for 24 units and waits 6. The communication-stress regime uses $p = 0.10$, $M = 2$, and $\zeta = 24$, so LS computes for 12 and waits 24.

Grid over sparsity, delay, and local work. A targeted grid varies sparsity, communication delay, local work, and worker-speed heterogeneity with the model, optimizer, partition, and budget above. The homogeneous profile has step times $(2, 2, 2, 2)$ and $\tau = 2$, and the heterogeneous profile has $(1, 2, 3, 6)$ and $\tau = 6$. Delays are relative to the profile, with $\zeta/\tau \in \{1, 4, 8\}$ called low, medium, and high. For example, a medium delay is $\zeta = 8$ for homogeneous and $\zeta = 24$ for heterogeneous workers, and a high delay is $\zeta = 16$ and $\zeta = 48$. Each profile is evaluated on nine settings:

$$
\begin{aligned}
(p, M, \zeta_{\text{level}}) \in \{&(0.25, 1, \text{medium}),\\
&(0.001, 4, \text{low}),\ (0.001, 4, \text{medium}),\ (0.001, 4, \text{high}),\\
&(0.01, 4, \text{medium}),\\
&(0.25, 4, \text{low}),\ (0.25, 4, \text{medium}),\ (0.25, 4, \text{high}),\\
&(0.25, 16, \text{medium})\}.
\end{aligned}
$$

This reduced grid is diagnostic rather than a full Cartesian product. It gives $2 \times 9 \times 3 = 54$ runs, all with seed 11. Runs are paired within a configuration, but there is no across-seed variability. In this grid, the nAUC of method m is $\frac{1}{T_1} \int_{t_m^-}^{T_1} a_m(t)\, dt$, evaluated by the trapezoidal rule over the recorded validation curve. This follows the implementation used for the aggregate summary and coincides with Appendix D.1 when all curves start at time zero.

D.3 Does overlap reduce training time, and why?

D.3.1 Mechanism on logistic regression

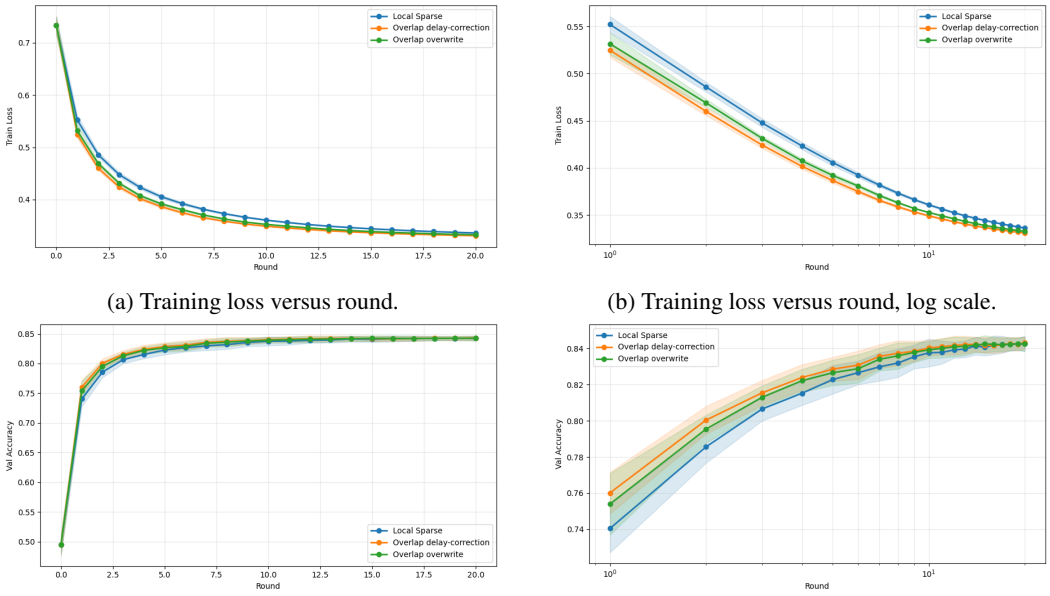
This experiment, summarized in Figure 1(a), isolates the use of the communication window for local computation. LS computes for 18 time units and then blocks for 6. The overlap methods use $M = 3$

and $\zeta = 6$, giving the same round duration $M\tau + \zeta = 24$ and

$$(N_1, N_2, N_3, N_4) = (18, 9, 6, 3), \qquad (Q_1, Q_2, Q_3, Q_4) = (6, 3, 2, 1).$$

The pre-communication phase is therefore identical across the three methods, and the overlap methods additionally compute during communication. Since the methods also communicate the same number of coordinates per round, logical time and logical communication are common rescalings of the round index. We include them only as consistency checks.

Across all metrics and parametrizations, delay correction outperforms overwrite, which outperforms LS (Figures 2 to 4). The delay-corrected variant reaches higher accuracy earlier and decreases the loss and gradient norm faster. The same ordering appeared with nonconvex regularized objectives and very heterogeneous worker speeds. It also appeared in compute-dominated regimes, where local steps are expensive relative to communication, and in communication-dominated regimes. LS wastes the communication window by waiting. Overwrite uses the window, but then discards part of this progress on the synchronized coordinates by replacing them with a delayed sparse average. Delay correction keeps the overlap-phase drift and corrects only the disagreement induced by sparse synchronization.



(a) Training loss versus round.

(b) Training loss versus round, log scale.

(c) Validation accuracy versus round.

(d) Validation accuracy versus round, log scale.

Figure 2: Controlled a9a comparison ($M = 3$, $\zeta = 6$, $p = 0.3$). Panels (a,b) show training loss, and (c,d) show validation accuracy, with linear and logarithmic round axes. One round is 24 simulated time units. Delay correction decreases loss and reaches higher validation accuracy earlier than overwrite, and both overlap rules improve over blocking LS.

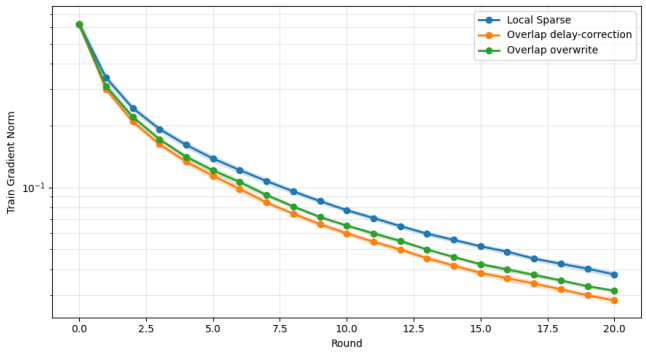
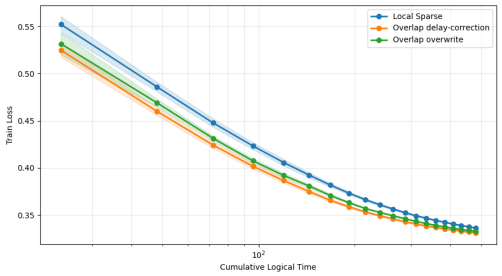
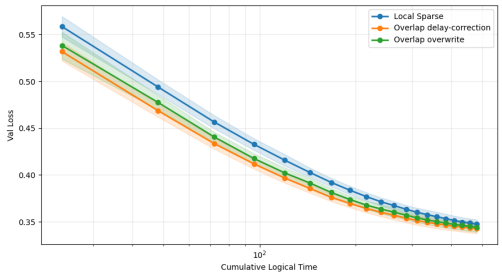


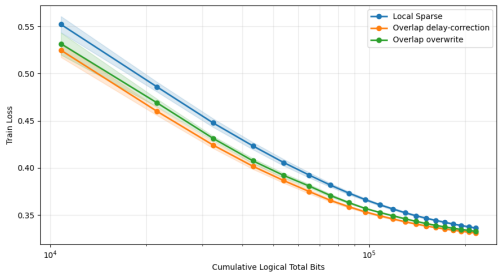
Figure 3: Training gradient norm in the controlled `a9a` comparison. Stationarity follows the same ordering as loss and accuracy.



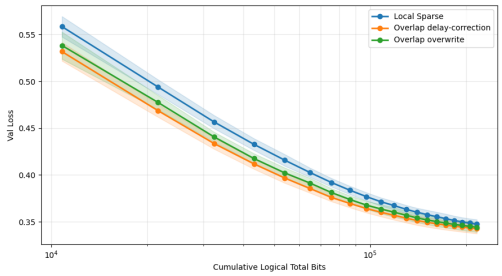
(a) Training loss versus logical time.



(b) Validation loss versus logical time.



(c) Training loss versus logical bits.



(d) Validation loss versus logical bits.

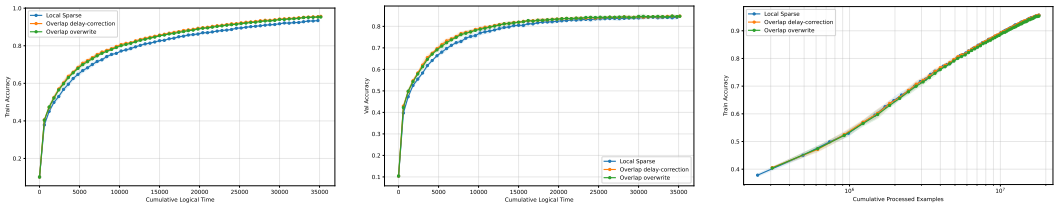
Figure 4: Loss under equivalent resource parametrizations of the controlled `a9a` comparison. Because round duration and per-round payload are matched, logical time and communication preserve the round-based ordering.

D.3.2 Neural networks

CIFAR-10. In both four-worker regimes, overlap improves training and validation accuracy in logical time. The advantage is much larger under communication stress, where blocking is more expensive (Table 5, Figures 5 and 6). Every method reaches both targets in all three seeds.

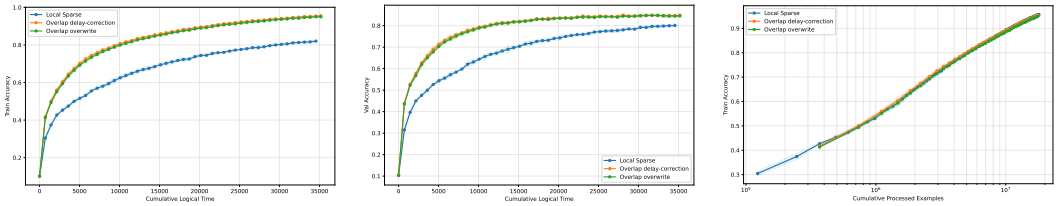
Table 5: Three-seed CIFAR-10 confirmation with four workers. Each entry is the median paired logical-time speedup $T_{\mathrm{LS}}/T_{\mathrm{method}}$ over seeds $0, 1, 2$ at fixed targets of 70% validation and 80% training accuracy. Every method reaches both targets in all three seeds.

	Validation target 70%		Training target 80%	
Regime	Overwrite	Delay-corrected	Overwrite	Delay-corrected
Normal communication	$1.19\times$	$1.25\times$	$1.22\times$	$1.26\times$
Communication stress	$2.85\times$	$3.10\times$	$2.90\times$	$3.07\times$



(a) Training vs. logical time. (b) Validation vs. logical time. (c) Training vs. processed examples.

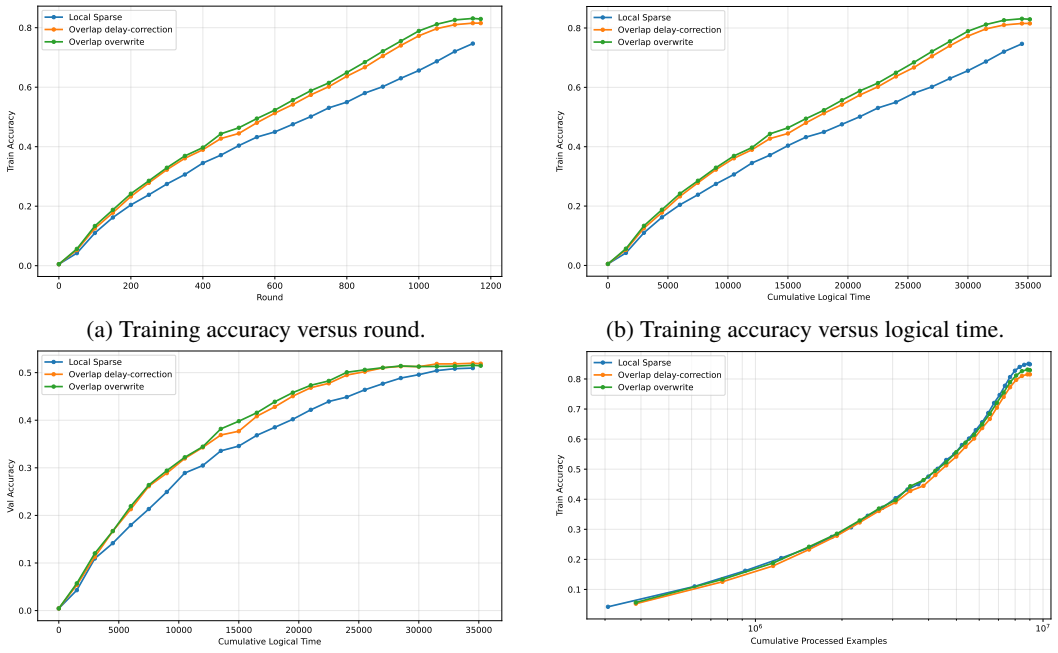
Figure 5: CIFAR-10, four workers with step times $(1, 2, 3, 6)$, normal regime ($p = 0.25$, $M = 4$, $\zeta = 6$), seed 11. Overlap improves over blocking LS in logical time, and the processed-example curves show no loss of accuracy at a fixed amount of data.



(a) Training vs. logical time. (b) Validation vs. logical time. (c) Training vs. processed examples.

Figure 6: CIFAR-10, four workers with step times $(1, 2, 3, 6)$, communication-stress regime ($p = 0.10$, $M = 2$, $\zeta = 24$), seed 11. The longer delay makes the logical-time benefit of overlap more pronounced.

Tiny ImageNet. Overlap is again the dominant effect (Figures 7 and 8). In the normal regime, both overlap variants improve substantially over LS in logical time and in rounds. Under communication stress, LS spends so much time waiting that it reaches much lower accuracy within the same time budget.



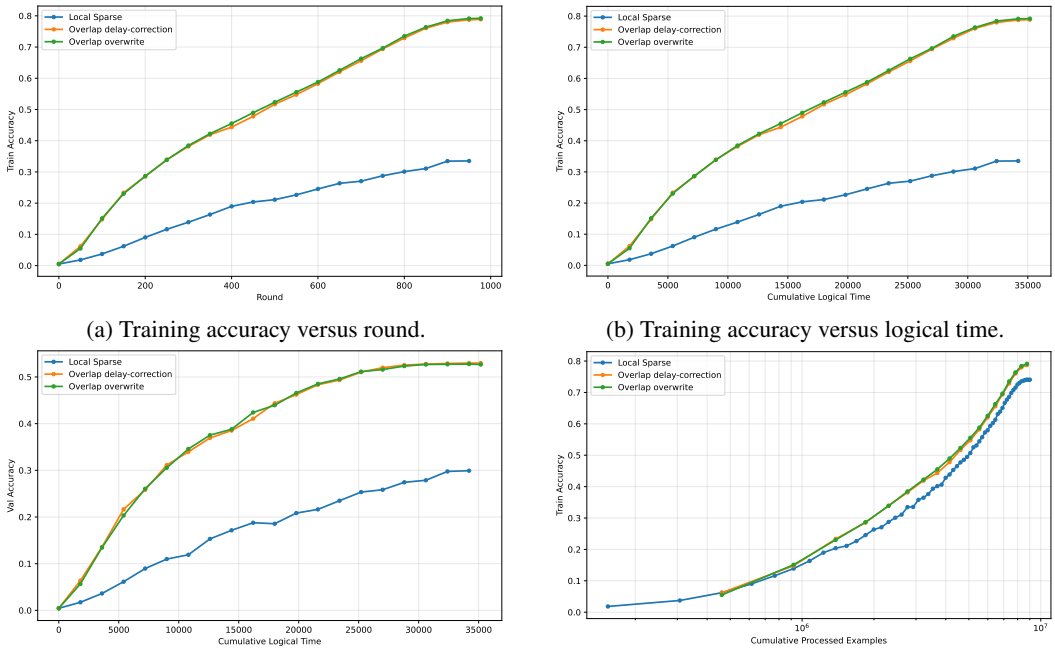
(a) Training accuracy versus round.

(b) Training accuracy versus logical time.

(c) Validation accuracy versus logical time.

(d) Training accuracy versus processed examples.

Figure 7: Tiny ImageNet, normal regime ($p = 0.25$, $M = 4$, $\zeta = 6$). Overlap improves over blocking LS in logical time and in rounds; against processed examples, the curves are much closer.



(a) Training accuracy versus round.

(b) Training accuracy versus logical time.

(c) Validation accuracy versus logical time.

(d) Training accuracy versus processed examples.

Figure 8: Tiny ImageNet, communication-stress regime ($p = 0.10$, $M = 2$, $\zeta = 24$). Both overlap variants substantially outperform blocking LS in logical time and remain ahead against processed examples.

D.3.3 What overlap buys: processed examples

Plotting against processed examples removes the time saved by computing during communication. On this axis, the four-worker CIFAR-10 curves are close, especially in the normal regime. The overlap variants neither lag behind nor become unstable, so the steps taken during communication remain useful optimization steps (Figures 5 and 6). With sixteen IID workers, overlap is slightly behind LS on this axis (Appendix D.4.2). On measured hardware, overwrite is about neutral and correction somewhat worse per effective epoch (Appendix D.9). On Tiny ImageNet in the normal regime, the processed-example plot shows similar sample efficiency once the computation performed during communication is accounted for. In the grid, the large logical-time separation largely disappears on this axis (Appendix D.4.3).

The Tiny ImageNet stress regime is the exception. There, the overlap methods perform the same pre-communication steps as LS and then two more such blocks during communication. Hence, at a comparable number of processed examples, LS has applied about three times as many sparse merges. We did not expect LS to be worse on this axis, and we do not fully understand the effect. One plausible explanation is that, with only $p = 0.10$ of the coordinates communicated and a deep network, very frequent partial merges perturb the trajectory. The overlap methods, by contrast, implicitly use a longer local window between sparse synchronizations.

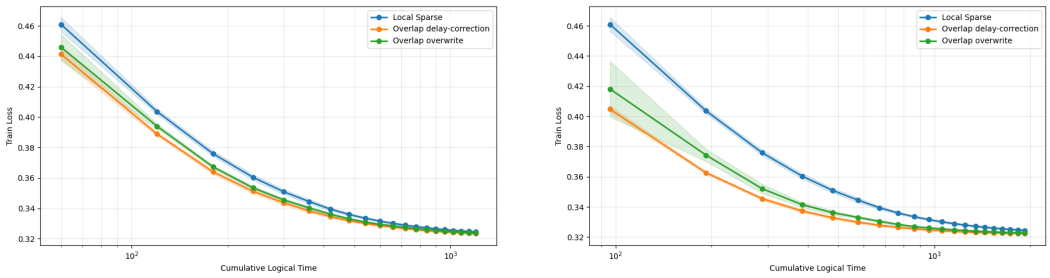
Overall, the main gain of overlap is in elapsed time rather than in the amount of data needed to reach a given accuracy. Overlap uses otherwise idle communication time, sometimes at a small per-example cost.

D.4 How do delay, local work, workers, and sparsity shape the gain?

D.4.1 Communication delay and local work

Communication delay. With $\zeta = 0$, the overlap phase has length zero, and the three methods coincide under the same seed, so we show only positive delays. Figure 9 compares the three methods on `a9a` at $\zeta = 12$ and $\zeta = 48$. At $\zeta = 12$, both overlap methods improve over LS for most of training. Correction is best, but its final margin over overwrite is small. When the window is long enough to

make overlap useful but not dominant, most of the gain therefore comes from avoiding idle time. At $\zeta = 48$, LS falls behind because it waits through a long window every round, and correction's advantage over overwrite is visibly larger (Appendix D.5). The neural-network experiments show the same dependence on the delay. On CIFAR-10, the four-worker speedups grow from $1.19\times$–$1.26\times$ to $2.85\times$–$3.10\times$ under communication stress (Table 5). The sixteen-worker results follow below.

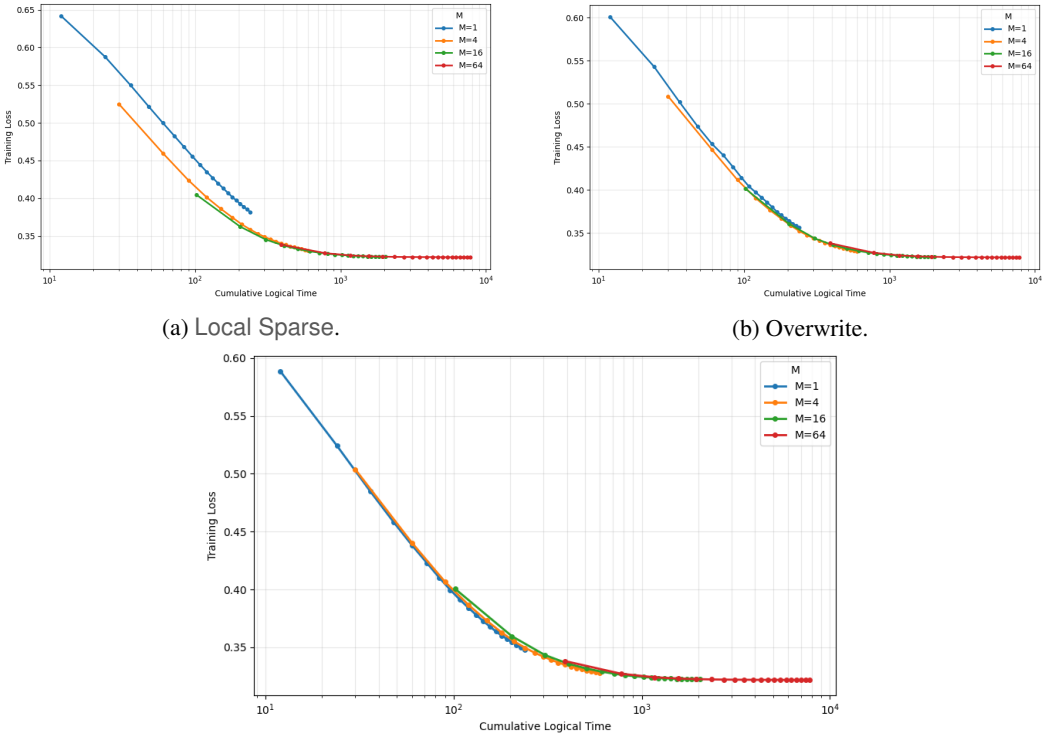


(a) Moderate delay, $\zeta = 12$.

(b) Large delay, $\zeta = 48$.

Figure 9: Training loss versus logical time for two communication delays on `a9a`. Overlap improves over blocking LS, and the advantage of delay correction grows with the delay.

Local computation budget. We vary $M \in \{1, 4, 16, 64\}$ on `a9a` with $\zeta = 6$ and plot training loss against cumulative logical time (Figure 10). All curves start from the same model at time zero, which the logarithmic axis cannot show. For larger M, the first visible point appears later because a round lasts longer. For blocking LS, early progress is sensitive to M. Small M synchronizes often and spends a larger fraction of time blocked, while $M = 16$ and $M = 64$ amortize communication and reach lower loss sooner. The overwrite curves are much closer across M, with differences mostly early in training, and the delay-corrected curves almost collapse. Overlap thus removes most of the logical-time penalty of small M, while large M introduces no visible loss penalty in this regime. All methods eventually approach a similar loss floor, so M mainly affects early progress. This conclusion is for homogeneous data. With strongly non-IID data, long local phases can harm all methods through client drift (Appendix D.6).



(a) Local Sparse.

(b) Overwrite.

(c) Delay-corrected.

Figure 10: Local computation budget $M \in \{1, 4, 16, 64\}$ on `a9a` with $\zeta = 6$. Blocking LS is sensitive to M early in logical time, overlap reduces this sensitivity, and the delay-corrected curves nearly collapse.

D.4.2 Worker count and compute heterogeneity

Sixteen-worker logistic regression. The delay dependence persists at sixteen workers. In the suite's communication-stress configuration ($\zeta = 32$ instead of 8), both overlap variants reach the 90% and 95% progress targets $1.21\times$ and $1.49\times$ sooner than LS. These are the largest speedups in the suite. The clean configurations with four or sixteen workers give $0.99\times$–$1.08\times$. On this quickly saturating problem, however, overlap's time-averaged validation accuracy over the common horizon is slightly lower than LS's. The gap is 0.09–0.12 percentage points in the clean configurations and 0.31–0.32 points under communication stress. The benefit is thus confined to the time needed to reach intermediate targets.

Worker-speed heterogeneity can turn overlap from a gain into a cost. With sixteen workers at $\tau_i \in \{1, 2, 4, 8\}$ and $\zeta = 8$, both overlap variants reach every progress target slightly later than LS (speedup $0.97\times$). Correction needs about 1.6 times LS's time for the 95% target ($0.62\times$), while the nAUC differences stay below 0.03 points. In this regime, the additional overlap updates cost more than the waiting time they hide, so we do not claim that overlap is uniformly faster in every timing regime.

Sixteen-worker CIFAR-10. The targeted suite gives a clearer neural-network picture. More workers do not remove the benefit. With homogeneous workers and the reference delay, the validation-nAUC gain over LS is 0.005 for overwrite and 0.008 for correction with four workers, and 0.009 for both with sixteen (Figure 11). Lengthening the delay from $\zeta = 6$ to $\zeta = 24$ raises the sixteen-worker gain to 0.029 for overwrite and 0.032 for correction (Figure 12, top). Heterogeneous speeds $(1, 2, 4, 8)^{\times 4}$ with $\zeta = 8$ give a smaller gain, 0.004 for overwrite and 0.006 for correction (Figure 12, bottom). Unlike in the sixteen-worker logistic suite, the gain stays positive. Both variants reach targets placed at 50–95% of LS's progress, and they do so $1.03\times$–$1.23\times$ sooner. Against processed examples, overlap is slightly behind LS at sixteen workers. Its validation accuracy averaged over the common example budget is 0.3 points lower in the clean configuration and 0.7–1.3 points lower

under communication stress and compute heterogeneity. The gain is therefore one of elapsed time. Overlap trades a small per-example cost for the use of otherwise idle communication time.

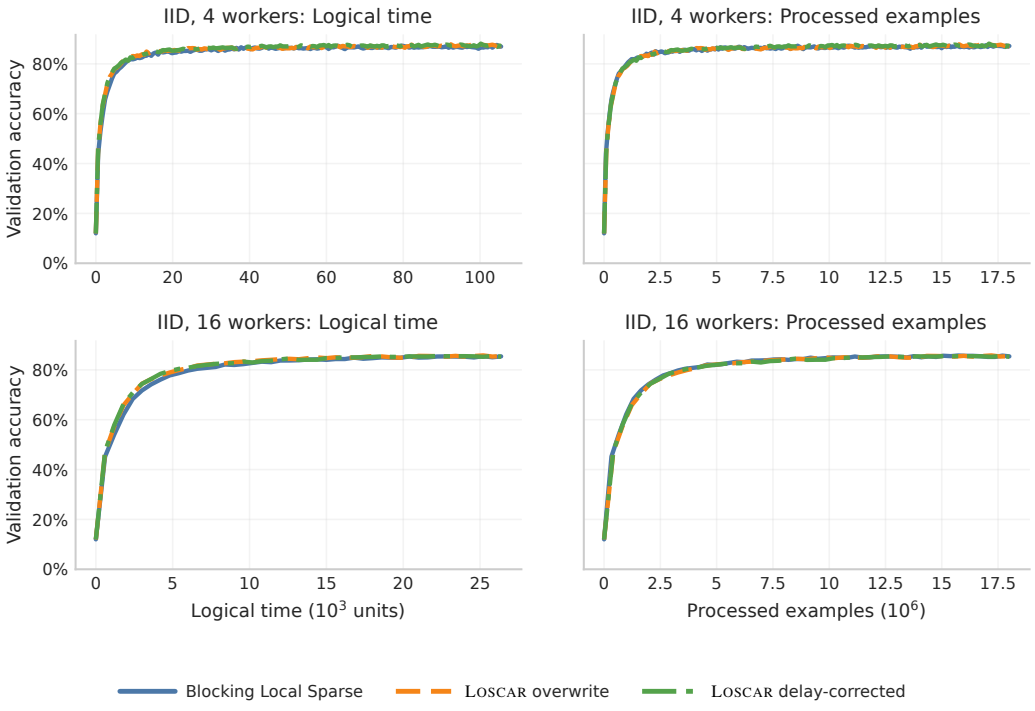


Figure 11: Targeted suite, clean IID CIFAR-10 with four (top) and sixteen (bottom) homogeneous workers. Columns: validation accuracy against simulated logical time (10^3 units) and processed examples (10^6). Curves are truncated to the common horizon; one paired seed (11), so no uncertainty band.

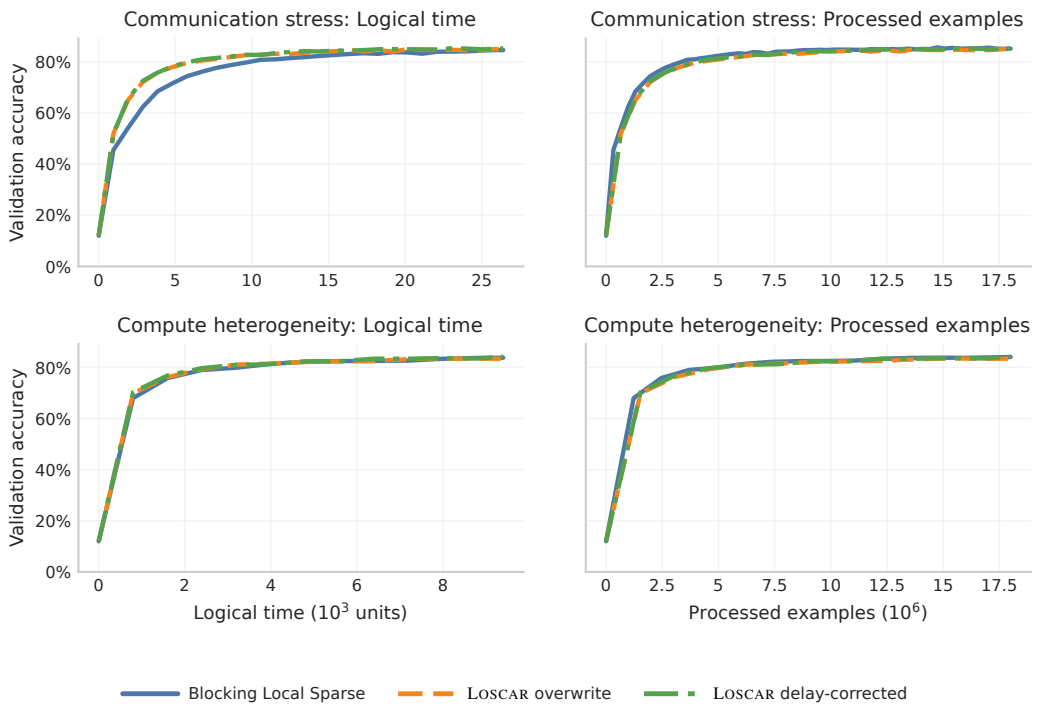


Figure 12: Targeted suite, sixteen workers: longer delay with homogeneous workers (top) and heterogeneous compute with the reference delay (bottom). Columns: validation accuracy against simulated logical time and processed examples. Common horizon; one paired seed (11).

D.4.3 A joint grid on Tiny ImageNet

Representative trajectories. At the central setting ($p = 0.25$, $M = 4$, medium delay), both overlap variants make substantially more training and validation progress than LS before the common horizon. This holds in both worker profiles (Figure 13). The two overlap curves remain close, so the gain is attributable to overlap rather than to either merge rule. Against processed examples, the large logical-time separation largely disappears, with small differences that vary across profiles and splits (Figure 14). Raising the delay to $\zeta/\tau = 8$ puts the overlap variants about twenty validation-accuracy points ahead of LS at the common horizon in both profiles (Figure 15). This is the setting where hiding a long communication interval matters most. Against processed examples, the curves are again much closer, and no method dominates across both profiles and both metrics (Figure 16). The high-delay advantage is therefore not produced by a larger effective-epoch budget.

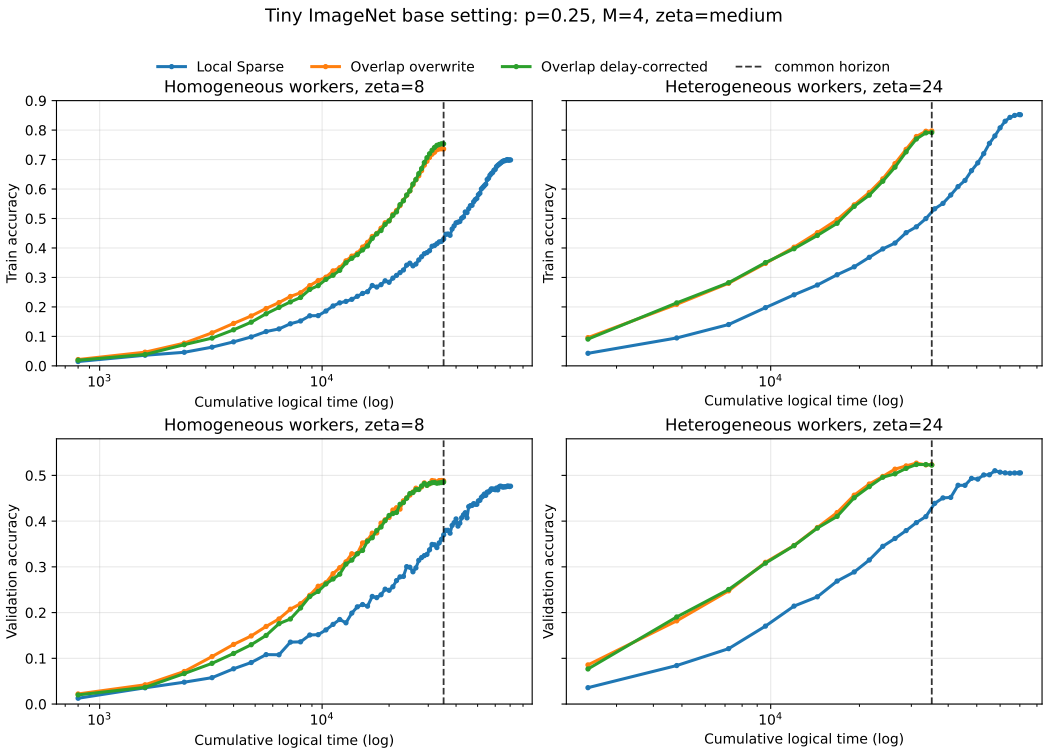


Figure 13: Tiny ImageNet grid at $p = 0.25$, $M = 4$, and medium delay $\zeta/\tau = 4$. Left: homogeneous workers with $\zeta = 8$; right: heterogeneous workers with $\zeta = 24$. Rows: training and validation accuracy against logical time. Dashed lines mark the common endpoint T_1.

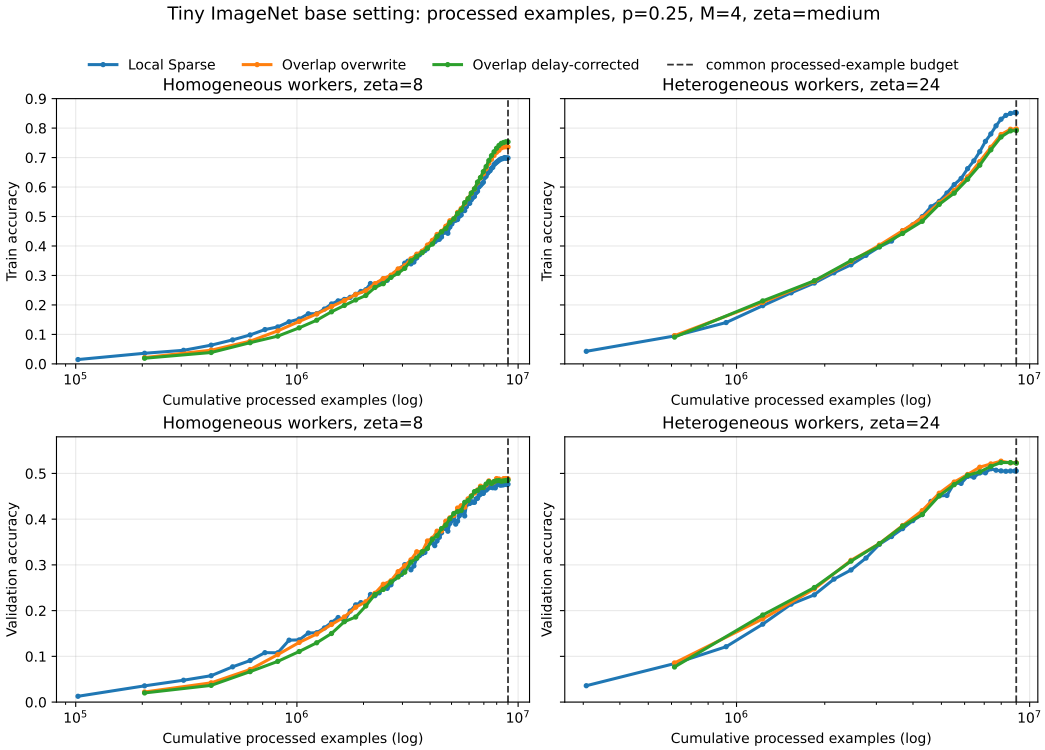


Figure 14: The runs of Figure 13 against processed examples. The trajectories are much closer than in logical time. This view matches data processing, but not the number of sparse synchronizations.

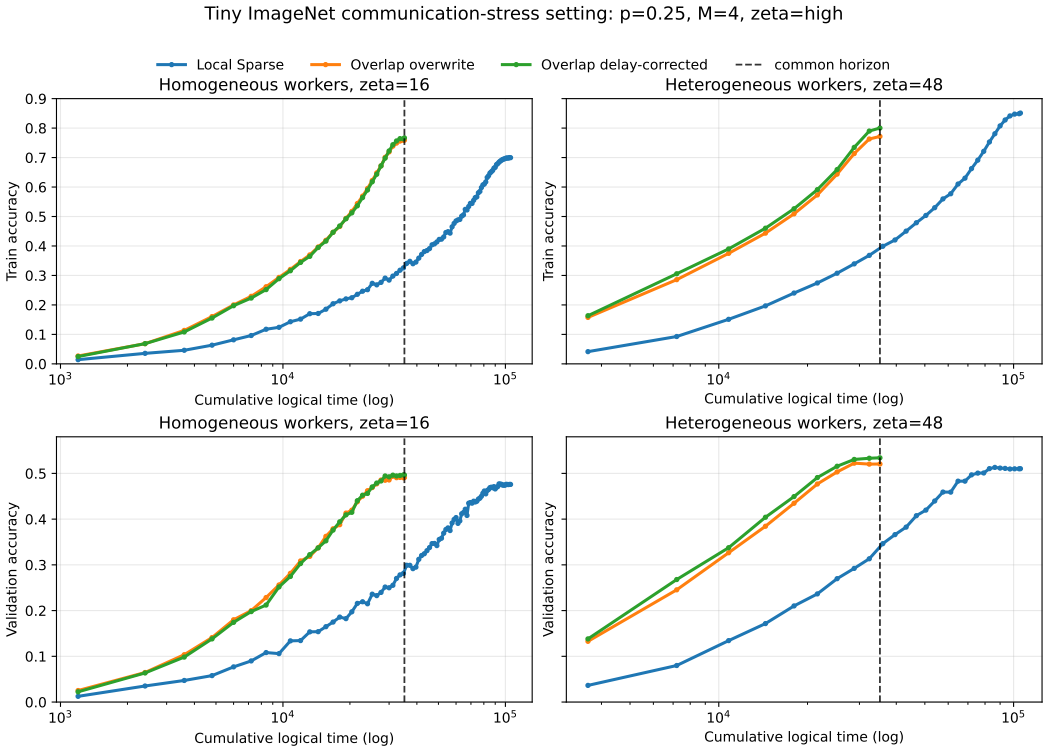


Figure 15: Tiny ImageNet grid at $p = 0.25$, $M = 4$, and high delay $\zeta/\tau = 8$ ($\zeta = 16$ for homogeneous and $\zeta = 48$ for heterogeneous workers). Overlap has a large training- and validation-accuracy advantage at matched logical time.

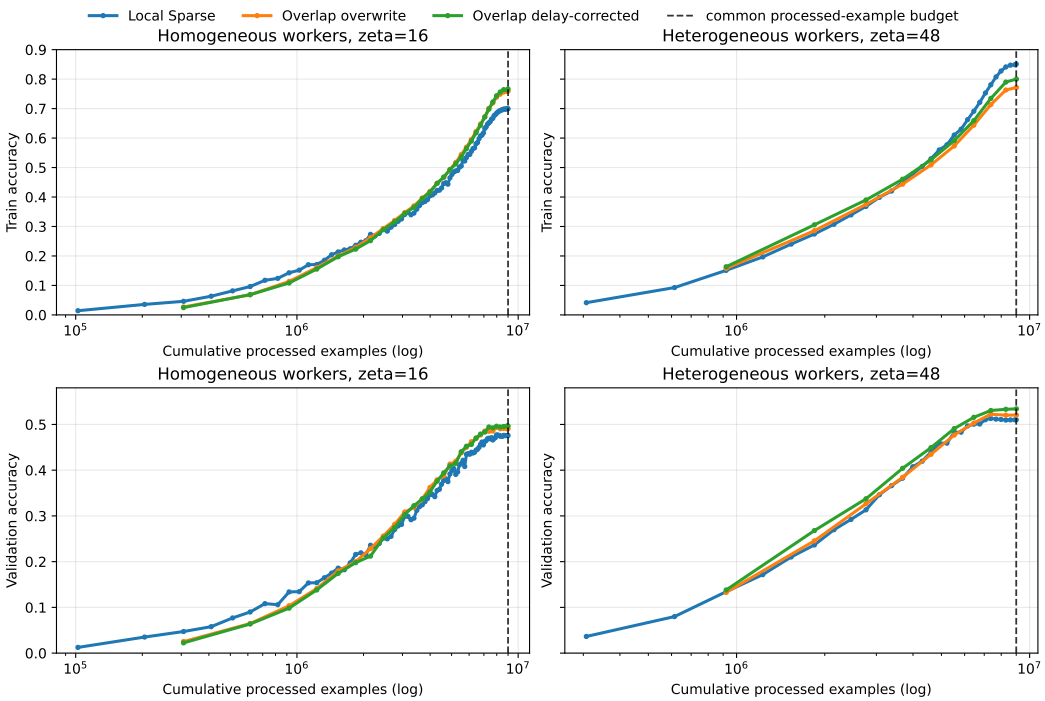


Figure 16: The runs of Figure 15 against processed examples. The methods are much closer than in logical time, and their ordering is not uniform across panels.

Aggregate results. Table 6 reports median paired nAUC differences across the 18 configurations. Overwrite and correction improve on LS by 0.041 and 0.037 overall. For each overlap method, the difference is positive in 13 of 18 settings: all 10 with $p = 0.25$, both with $p = 0.01$, and only one of the six with $p = 0.001$. The median gains are therefore large for $p \in \{0.01, 0.25\}$ but negative at $p = 0.001$, and the two worker profiles have nearly identical medians. The delay rows are marginal summaries rather than isolated delay effects, because the grid is not fully crossed. The corrected-minus-overwrite medians stay near zero in every subset. For time to accuracy, Tables 7 and 8 use only configurations in which both methods reach the target. Median validation speedups are about $1.6\times$ at 20%, $2.0\times$ at 30–40%, and $1.9\times$ at 45%. Training speedups are close to $2\times$ at 40–50% and about $1.8\times$ at 60–70%. At higher targets, the matched sample shrinks because difficult settings, especially at extreme sparsity, do not reach them. Those rows characterize an easier subset, and their counts can differ between the overlap methods.

Table 6: Median paired differences in validation nAUC up to each configuration's common logical-time horizon in the Tiny ImageNet grid. Positive entries in the middle columns favor overlap over LS; the last column compares the merge rules. n is the number of matched configurations.

Group	n	Overwrite – Local	Corrected – Local	Corrected – Overwrite
All	18	0.041	0.037	-0.002
$p = 0.25$	10	0.124	0.119	-0.002
$p = 0.01$	2	0.116	0.115	-0.001
$p = 0.001$	6	-0.006	-0.008	-0.003
Low delay	4	0.034	0.032	-0.004
Medium/high delay	14	0.111	0.107	-0.002
Homogeneous	9	0.042	0.036	-0.001
Heterogeneous	9	0.040	0.037	-0.003

Table 7: Logical-time speedup over LS to reach each validation-accuracy target in the Tiny ImageNet grid, over matched configurations in which both methods reach the target (n is method-specific).

Target	Method	Median	Mean	n
20%	Overwrite	1.655	2.143	14
20%	Delay-corrected	1.573	2.142	14
30%	Overwrite	1.979	2.464	13
30%	Delay-corrected	1.906	2.465	13
40%	Overwrite	2.061	2.449	13
40%	Delay-corrected	2.008	2.449	13
45%	Overwrite	1.910	2.316	10
45%	Delay-corrected	1.880	2.367	10
50%	Overwrite	1.610	1.815	6
50%	Delay-corrected	1.582	1.813	6
52%	Overwrite	1.534	1.534	2
52%	Delay-corrected	1.553	1.553	2

Table 8: Logical-time speedup over LS to reach each training-accuracy target in the Tiny ImageNet grid, over matched configurations in which both methods reach the target (n is method-specific).

Target	Method	Median	Mean	n
40%	Overwrite	2.048	2.420	13
40%	Delay-corrected	1.981	2.407	13
50%	Overwrite	1.979	2.456	13
50%	Delay-corrected	1.923	2.480	13
60%	Overwrite	1.881	2.243	10
60%	Delay-corrected	1.841	2.246	10
70%	Overwrite	1.859	1.776	7
70%	Delay-corrected	1.839	1.777	7
80%	Overwrite	1.217	1.193	3
80%	Delay-corrected	1.178	1.508	4

Extreme sparsity: a failure regime. At $p = 0.001$, $M = 4$, and medium delay, overlap gives no reliable logical-time advantage (Figure 17). The homogeneous curves are close near the common horizon, and under heterogeneous speeds all methods make little progress. This boundary is specific to the tested recipe, but it shows that the gains at moderate sparsity do not extrapolate automatically to communicating 0.1% of the coordinates.

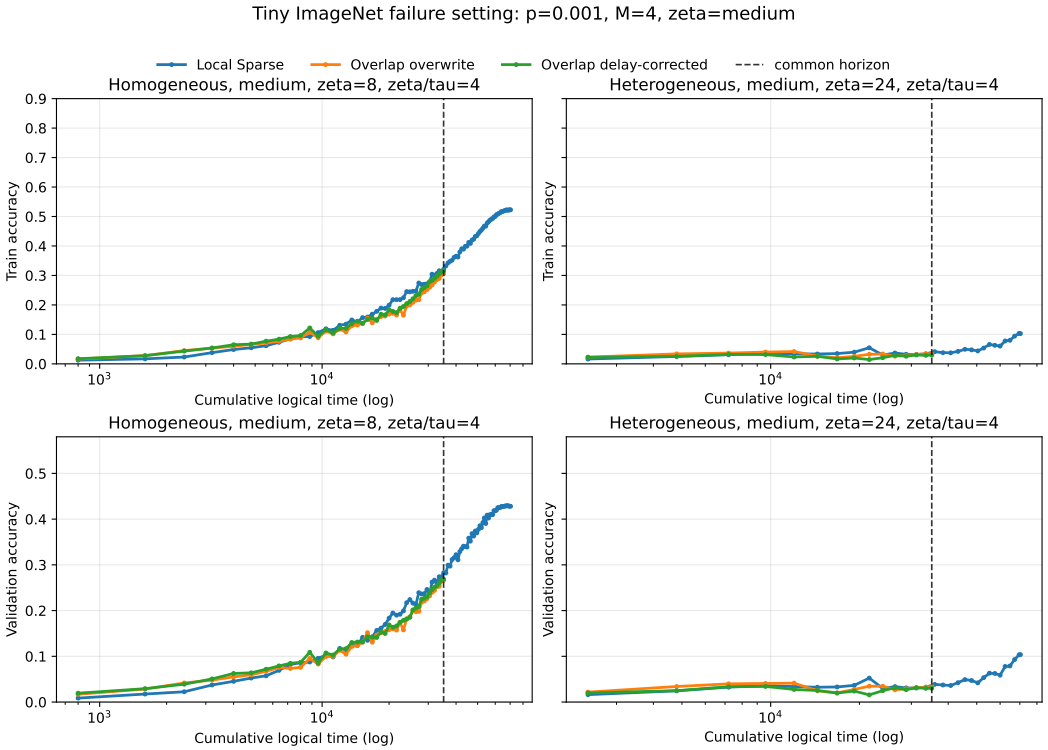
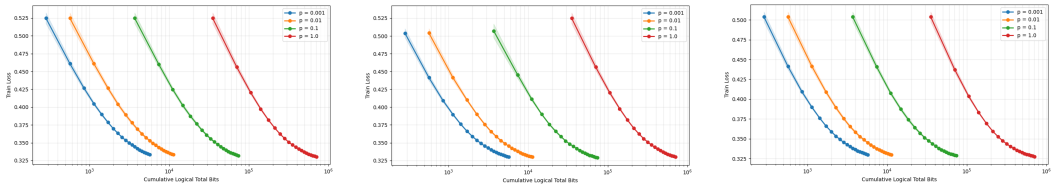


Figure 17: Tiny ImageNet grid at extreme sparsity, $p = 0.001$, $M = 4$, and medium delay. Overlap is not consistently beneficial, and all methods make little progress under the heterogeneous profile.

D.4.4 Communicated fraction

We vary the nominal communicated fraction $p \in \{0.001, 0.01, 0.1, 1.0\}$ on `a9a` with $M = 4$, $\zeta = 6$, and otherwise the settings of Appendix D.2.1. Each mask selects $K = \lceil pd \rceil$ of the $d = 124$ coordinates. These values are therefore realized as $K = 1$, 2, 13, and 124, that is, $K/d \approx 0.008$, 0.016, 0.105, and 1. The logical bits include index bits and are computed from the realized K. In particular, the nominal $p = 0.001$ synchronizes a single coordinate per round, about eight times the nominal fraction. Figure 18 plots training loss against cumulative logical bits for each method; its legends show the nominal p. The figure is best read horizontally, at a fixed communication budget. Reducing K shifts the curves left by orders of magnitude, because each round transmits fewer coordinates. The shape of the loss curve changes only mildly. For example, the $K = 1$ and $K = 2$ curves reach losses close to the denser settings with far fewer bits. More coordinates can slightly reduce the loss at the end of the run, but at a high communication cost. Dense averaging, $K = 124$, lies far to the right. The conclusion holds for all three methods, and delay correction remains at least as good as overwrite with the same savings. This small homogeneous logistic problem is benign even when one coordinate is synchronized per round. It does not reach the regime of the Tiny ImageNet grid (Appendix D.4.3). There, $p = 0.001$ means synchronizing 0.1% of the coordinates of a ResNet-18-style network, and overlap fails. The theory also predicts a disagreement cost growing as p^{-2} (Corollary B.1).



(a) Local Sparse. (b) Overwrite. (c) Delay-corrected.

Figure 18: Sparsity-level ablation on `a9a` ($M = 4$, $\zeta = 6$). The legends show the nominal $p \in \{0.001, 0.01, 0.1, 1.0\}$, realized as $K \in \{1, 2, 13, 124\}$ of $d = 124$ coordinates. Fewer coordinates sharply reduce the communicated bits while preserving a similar loss trajectory.

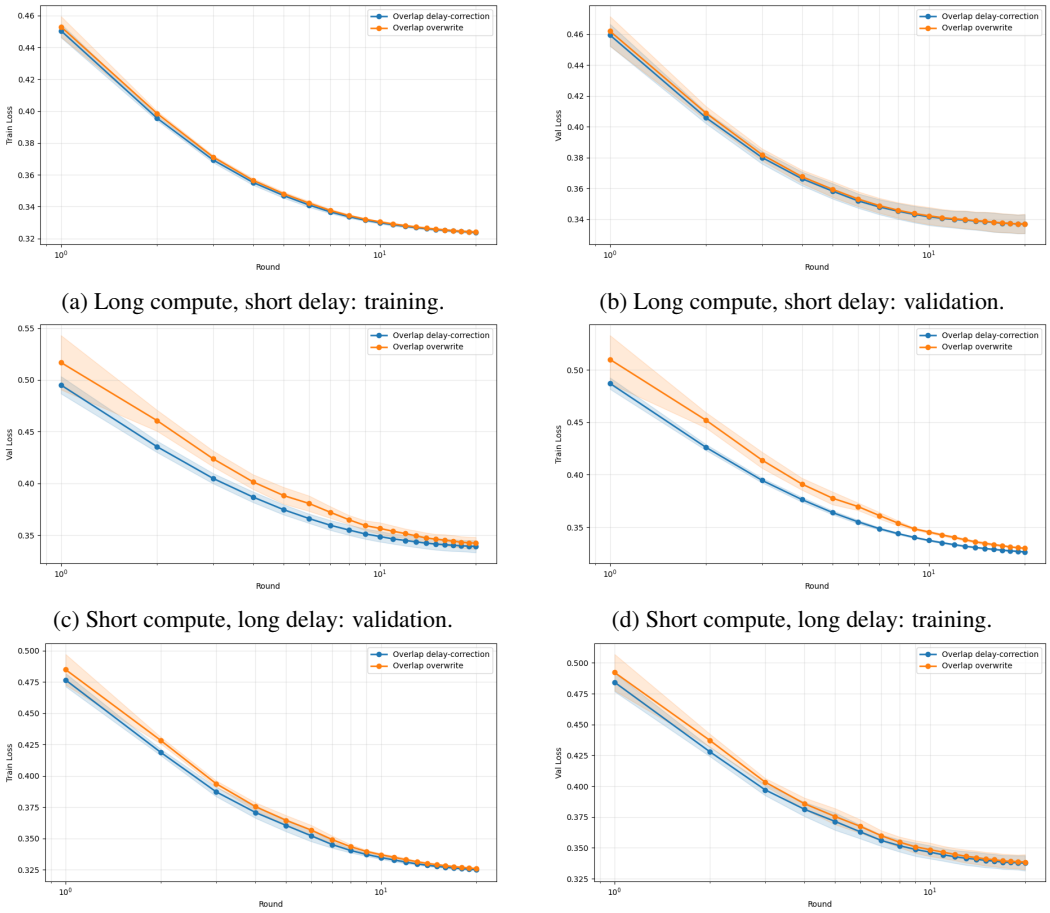
D.5 When does delay correction help?

This section collects all evidence on the merge rule, which is otherwise scattered across experiments. The working hypothesis, suggested by logistic regression, is that correction pays off when much local work accumulates during communication.

Logistic regression. To isolate the merge rule, we compare only the two overlap variants on `a9a` in three four-worker regimes that differ in local budget, delay, and worker speeds:

setting	M	ζ	$(\tau_1, \tau_2, \tau_3, \tau_4)$
long compute, short delay	8	6	$(1, 2, 3, 6)$
short compute, long delay	2	24	$(1, 2, 3, 6)$
very heterogeneous speeds	2	20	$(1, 2, 10, 20)$.

Delay correction gives lower training and validation loss in all three (Figure 19). The gain is small but persistent with long compute and a short delay, large throughout training with short compute and a long delay, and moderate under very heterogeneous speeds. Together with the delay ablation (Appendix D.4.1), this shows that in these four-worker regimes the advantage of correction scales with the amount of optimization performed during communication. In the sixteen-worker logistic suite, by contrast, the two merge rules are nearly indistinguishable: their speedups differ by at most $0.05\times$, and their nAUCs by less than 0.02 points.



(a) Long compute, short delay: training.

(b) Long compute, short delay: validation.

(c) Short compute, long delay: validation.

(d) Short compute, long delay: training.

(e) Very heterogeneous speeds: training.

(f) Very heterogeneous speeds: validation.

Figure 19: Merge-rule comparison on `a9a`. Rows: long compute and short delay ($M = 8$, $\zeta = 6$); short compute and long delay ($M = 2$, $\zeta = 24$), where overwrite discards the most overlap-phase progress and the gap is largest; and very heterogeneous speeds ($M = 2$, $\zeta = 20$, worker times $(1, 2, 10, 20)$).

Controlled Tiny ImageNet delay sweep. To test the hypothesis while holding the model, data, and optimizer fixed, we run a controlled Tiny ImageNet delay sweep. Four workers with step times $(1, 2, 3, 6)$ use $p = 0.25$, $M = 4$, the optimizer of Appendix D.2.3, the 100-effective-epoch budget, and four paired seeds 11–14. Only the delay changes, over $\zeta \in \{6, 24, 48\}$, giving mean overlap-step counts $\overline{Q} \in \{3, 12, 24\}$. In Table 9, positive nAUC and final-accuracy differences favor correction, and a corrected/overwrite target-time ratio below one favors correction. The 40% validation target is reached by both methods in every momentum-based run of the sweep.

Table 9: Controlled Tiny ImageNet delay sweep, four paired seeds per delay. Differences are corrected minus overwrite, as mean $\pm$ sample standard deviation. "nAUC wins" counts seeds with a positive validation-nAUC difference, and $T_{\mathrm{DC}}/T_{\mathrm{OW}}$ is the ratio of times to 40% validation accuracy.

ζ	$\overline{Q}$	Validation nAUC difference	nAUC wins	Final validation difference	$T_{\mathrm{DC}}/T_{\mathrm{OW}}$
6	3	-0.00650 ± 0.00424	$0/4$	-0.21 ± 0.74 pp	1.049 ± 0.022
24	12	-0.00080 ± 0.00316	$2/4$	$+0.87 \pm 0.52$ pp	1.009 ± 0.028
48	24	$+0.00391 \pm 0.00714$	$3/4$	$+0.61 \pm 0.94$ pp	0.978 ± 0.065

The mean correction-minus-overwrite validation nAUC changes from -0.00650 at the short delay ($\zeta = 6$) to $+0.00391$ at the long delay ($\zeta = 48$). At the short delay, overwrite has the better nAUC in all four seeds and reaches the target sooner. At the medium delay, the two are near parity on the

trajectory, although correction ends with slightly higher final validation accuracy. At the long delay, correction has a higher nAUC and a lower target time in three of four seeds. Within each seed, the nAUC interaction between $\overline{Q} = 24$ and $\overline{Q} = 3$ is positive for three of four seeds. Its mean is 0.0104, with paired bootstrap 95% interval $[0.0031, 0.0176]$. The sweep supports a regime-dependent benefit, not a universal advantage for correction at long delays (Figure 20).

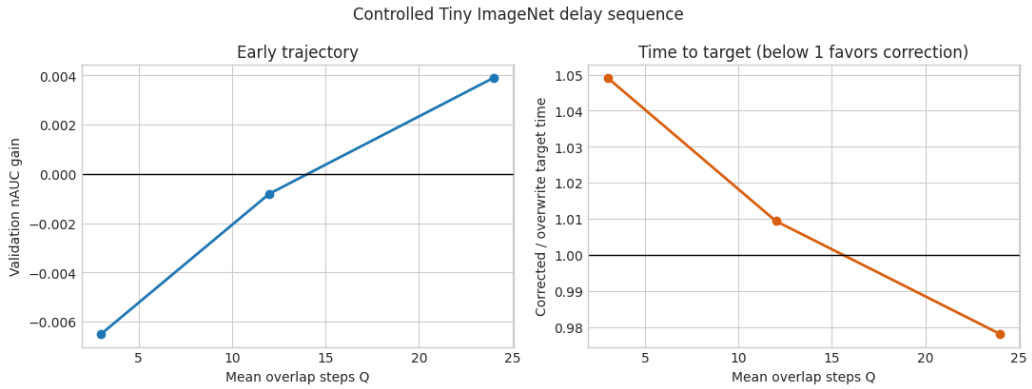


Figure 20: Effect of the overlap duration on the merge rule in the Tiny ImageNet delay sweep. Left: corrected-minus-overwrite validation nAUC. Right: corrected/overwrite time to 40% validation accuracy; values below one favor correction.

Optimizer state. We also ran an eight-run momentum ablation at the medium delay, changing only the optimizer momentum from 0.9 to 0 (Table 10). Without momentum, correction has a higher validation nAUC in all four paired seeds. However, both methods reach much lower accuracy, and overwrite never reaches the 40% target.

Table 10: Medium-delay Tiny ImageNet momentum ablation. Each row averages four paired seeds; differences are corrected minus overwrite.

Momentum	Final val. OW	Final val. DC	Final difference	Validation nAUC difference	nAUC wins
0.9	54.12%	55.00%	+0.87 pp	−0.00080	2/4
0	38.20%	40.33%	+2.13 pp	+0.01243	4/4

The hardware study points the same way. In the momentum-based runs at the $250\,\mathrm{ms}$ window (Appendix D.9), mean corrected-minus-overwrite final validation accuracy is -1.02, -0.93, -0.77, and -0.73 percentage points across the four model/sparsity configurations. Overwrite thus has higher mean final validation accuracy in all four, and it also has the higher validation nAUC in every paired seed. These runs use learning rate 0.05 and SGD momentum 0.9. Each worker keeps a velocity buffer v_i and updates $v_i \leftarrow 0.9v_i + g_i$ and $x_i \leftarrow x_i - 0.05v_i$. At synchronization, the selected coordinates of x_i are merged, but the velocity buffers are neither communicated nor reset. Each worker therefore continues with optimizer state accumulated along its own pre-merge trajectory, which can be temporarily inconsistent with the merged parameters. This parameter–momentum mismatch is one possible explanation for the weaker correction results, but it remains a hypothesis. An earlier four-V100 study used momentum-free SGD, learning rate 0.0125, native communication, and fixed $Q \in \{1, 2, 4, 8\}$. In its hardest $p = 0.50$ settings, the final validation accuracies for LS / overwrite / correction were $78.65\%/75.94\%/78.59\%$ at $Q = 4$ and $78.33\%/74.62\%/78.41\%$ at $Q = 8$ (Figure 21). Together, these results suggest an interaction with optimizer state. Without momentum, correction can protect long-overlap progress. With momentum, overwrite is stronger. They do not establish momentum as a complete explanation of the differences across experiments.

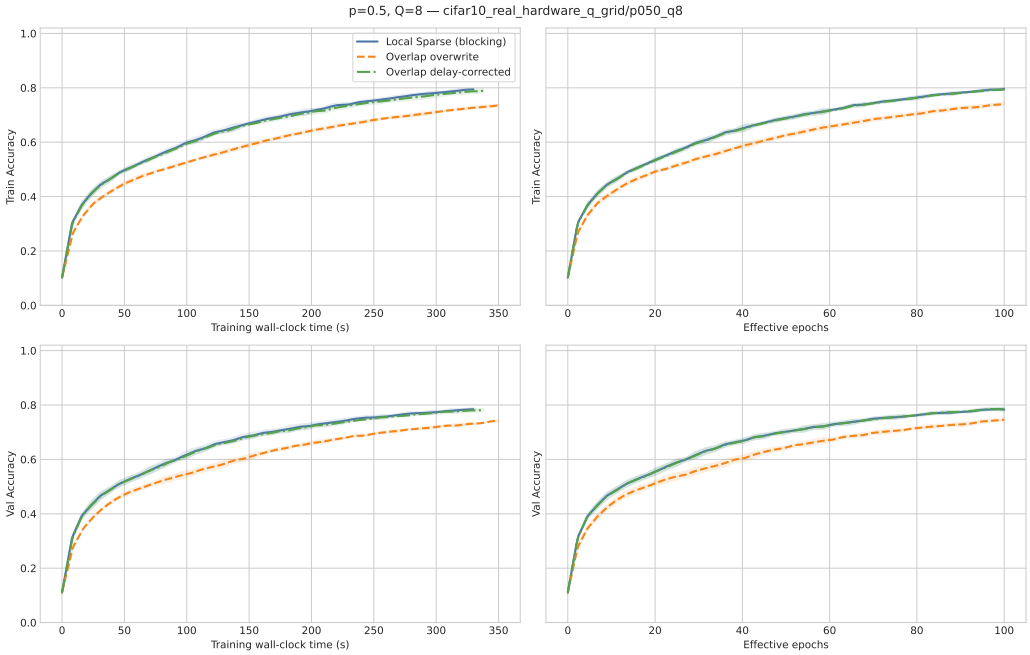


Figure 21: Earlier momentum-free four-V100 study at $p = 0.50$ and fixed $Q = 8$. Training and validation accuracy against measured training wall-clock time and effective epochs. This fixed-Q experiment is distinct from the controlled result-availability windows of Appendix D.9.

Other experiments. The merge rules are nearly tied in the Tiny ImageNet representative regimes and grid (Appendices D.3.2 and D.4.3). In the external-baseline grid, correction's regime-level nAUC gain over overwrite is only 0.0001–0.0019 (Appendix D.7). Under heterogeneous CIFAR-10 data, overwrite has an equal or higher validation nAUC than correction on every non-IID partition of the four-worker sweep (Table 11). The differences are -0.0043, -0.0010, -0.0008, and -0.0054 for $\alpha = 1, 0.3, 0.1, 0.03$. One possible mechanism is that correction preserves the displacement accumulated during communication, and heterogeneous data can bias this displacement toward each worker's local objective. Overwrite, in contrast, acts as a stronger realignment step. This is a hypothesis, and the ordering is not stable across worker counts. In the sixteen-worker suite, correction has the larger gain in both Dirichlet configurations (0.012 versus 0.010 at $\alpha = 0.1$, and 0.016 versus 0.010 at $\alpha = 0.03$). It has the smaller gain only under the combined stress. In the aggressive `a9a` non-IID stress test, correction is worst (Appendix D.6.1).

Summary. Correction helps in the four-worker heterogeneous logistic-regression regimes, and more so with longer delay. It also helps in the controlled Tiny ImageNet sweep at long delay and in runs without momentum. It is neutral or slightly worse at short delay, with momentum, in the sixteen-worker logistic suite, and in most neural-network comparisons. It is harmful under extreme client drift. The preferred merge rule is therefore regime-dependent.

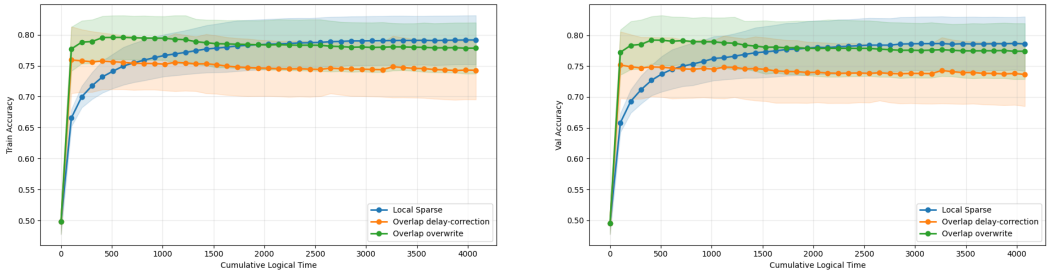
D.6 What happens when worker data are heterogeneous?

Our convergence theory assumes homogeneous data: every worker samples from the same distribution $\mathcal{D}$ and optimizes the same objective (Section 2). None of its guarantees applies to the label-Dirichlet partitions below. These partitions are empirical stress tests of how far the observed behavior extends beyond the analyzed setting.

D.6.1 Logistic regression

An aggressive non-IID stress test. With heterogeneous data, a long communication window is not always beneficial. We partition `a9a` by a label-Dirichlet split with $\alpha = 0.03$, which makes the worker objectives strongly heterogeneous. We run 40 rounds with $\eta = 0.3$, $p = 0.01$ ($K = 2$ of

$d = 124$), $M = 1$, $\zeta = 96$, and worker times $(1, 2, 3, 6)$. The workers then take $(N_i) = (6, 3, 2, 1)$ steps before and $(Q_i) = (96, 48, 32, 16)$ steps during communication. Both overlap methods improve very quickly at first, because the long window gives them many extra local updates (Figure 22). With heterogeneous data, however, these updates also pull each worker toward its own local objective, and the overlap curves plateau below LS. Delay correction is worst because it preserves exactly this locally biased progress. LS is slower initially but takes far fewer local steps between synchronizations and eventually reaches higher training and validation accuracy. In such regimes, one would need smaller learning rates, less aggressive overlap, stronger synchronization, or an analysis that controls client drift.



(a) Training accuracy.

(b) Validation accuracy.

Figure 22: Aggressive non-IID stress test on `a9a` (label-Dirichlet $\alpha = 0.03$, $M = 1$, $\zeta = 96$, $p = 0.01$). The overlap methods progress rapidly at first but plateau below blocking LS: too many local steps during communication drive the workers toward their own local optima.

Moderate non-IID partitions. The sixteen-worker suite does not reproduce this failure under moderate overlap. With Dirichlet partitions ($\alpha = 0.1$ or 0.03) and $\zeta = 8$, the overlap variants reach the progress targets $1.02\times$–$1.10\times$ sooner than LS. Their nAUCs differ from LS's by at most 0.11 points. In the combined stress (heterogeneous speeds, $\zeta = 32$, $\alpha = 0.1$), the speedups lie between $0.995\times$ and $1.017\times$, and correction's nAUC is 0.22 points below LS's. In these runs, the collapse above therefore appears only when strong heterogeneity is combined with very long overlap phases.

D.6.2 CIFAR-10

We partition the training examples among the four workers of the normal regime using label-Dirichlet distributions with $\alpha \in \{1, 0.3, 0.1, 0.03\}$. We keep the shared-data setting as an IID baseline. Smaller α makes the worker objectives more heterogeneous. Apart from the partition, these runs use the normal configuration ($p = 0.25$, $M = 4$, $\zeta = 6$, worker times $(1, 2, 3, 6)$). Every run uses seed 11, with initialization and partition matched across methods within each setting. The sweep is therefore a paired single-seed diagnostic.

Effect of heterogeneity. Increasing heterogeneity degrades all three methods (Figures 23 and 24). The logical time LS needs to reach 70% validation accuracy grows from 983 (IID) to 1,155, 2,356, 4,617, and 32,886 as α decreases through 1, 0.3, 0.1, and 0.03. Best validation accuracy falls from about 87% (IID) to about 71% at $\alpha = 0.03$. Heterogeneity thus both slows optimization and lowers the accuracy reached within the budget.

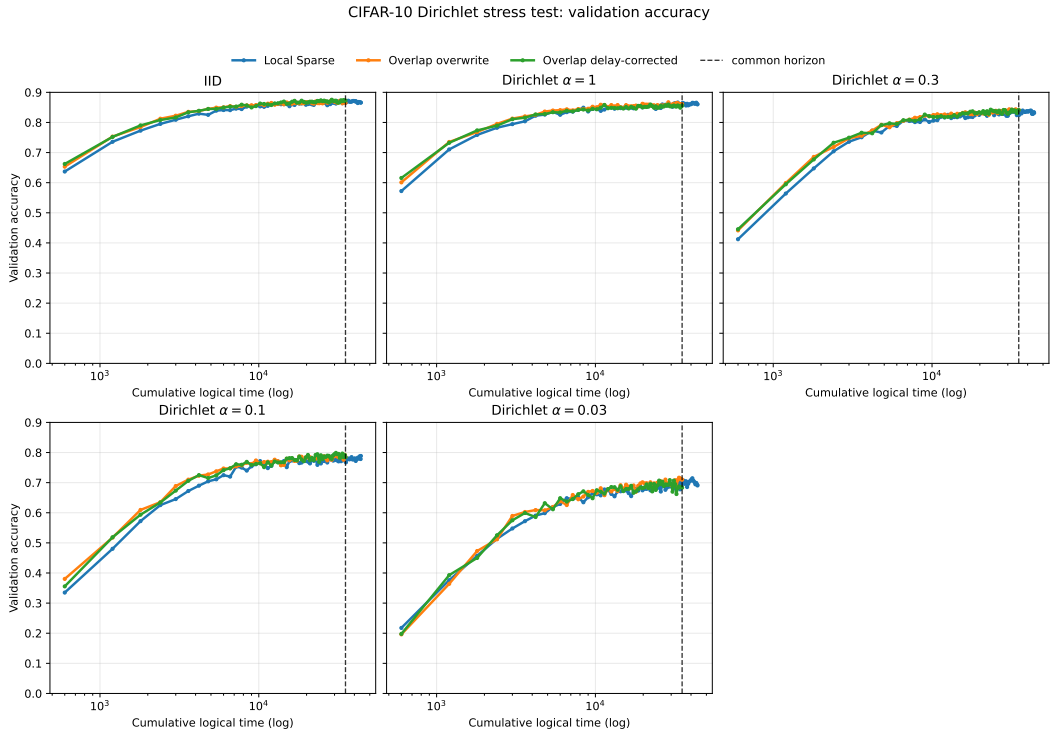


Figure 23: CIFAR-10 validation accuracy across the label-Dirichlet sweep. Smaller α degrades every method. Dashed lines mark the common horizon of each partition.

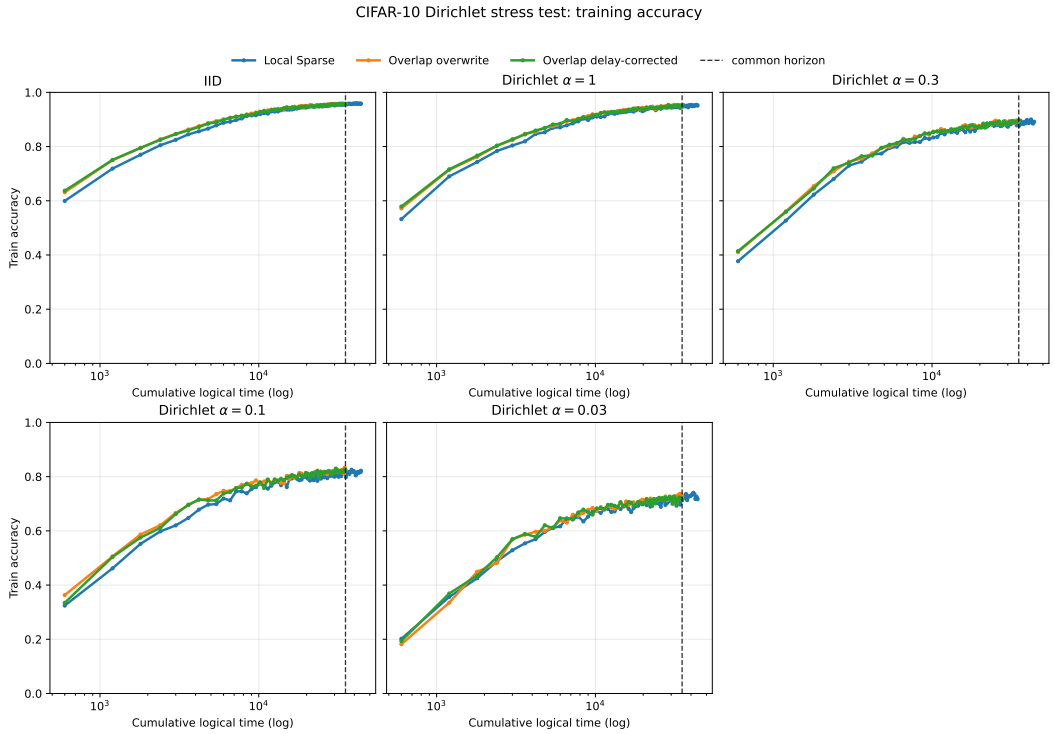


Figure 24: CIFAR-10 training accuracy across the label-Dirichlet sweep. The deterioration mirrors the validation results.

Common-horizon comparison. For these runs, the common horizon is $[T_0, T_1] = [0, 35{,}160]$ for every partition. The nAUC is computed by the trapezoidal rule with linear interpolation at the endpoints. Both overlap methods have positive validation-nAUC differences relative to LS in all five partitions and reach 70% validation accuracy earlier (Table 11). Their training-nAUC differences are also positive in every partition. Overlap therefore retains a logical-time benefit as the data become strongly heterogeneous. The 70% speedups at $\alpha = 0.03$ should be read cautiously, because the target is close to the terminal accuracy and its interpolated hitting time is sensitive to local fluctuations. The merge-rule columns are discussed in Appendix D.5.

Table 11: CIFAR-10 Dirichlet common-horizon comparison. T_{70}^{LS} is the logical time at which LS first reaches 70% validation accuracy, S_{OW} and S_{DC} are speedups over LS at that target, and the last three columns are paired validation-nAUC differences on $[0, 35{,}160]$.

Partition	T_{70}^{LS}	S_{OW}	S_{DC}	$\Delta_{\mathrm{OW/LS}}$	$\Delta_{\mathrm{DC/LS}}$	$\Delta_{\mathrm{DC/OW}}$
IID	983	1.11×	1.15×	0.0048	0.0066	0.0018
$\alpha = 1$	1,155	1.11×	1.12×	0.0054	0.0011	-0.0043
$\alpha = 0.3$	2,356	1.14×	1.15×	0.0094	0.0085	-0.0010
$\alpha = 0.1$	4,617	1.39×	1.32×	0.0132	0.0125	-0.0008
$\alpha = 0.03$	32,886	1.68×	1.32×	0.0114	0.0060	-0.0054

Logical time versus processed examples. At $\alpha = 0.03$, the two views diverge (Figure 25). In logical time, overlap progresses faster early because it computes while communication is in flight. At matched processed examples, LS is initially ahead, especially in training accuracy. Under heterogeneous objectives, the overlap-phase updates are therefore not uniformly as sample-efficient as LS updates. This is because some local progress is directed toward worker-specific objectives. The gap narrows later, and the three methods finish in the same accuracy range. The best validation accuracies are 71.46%, 71.64%, and 70.84% for LS, overwrite, and correction, and the final ones are 69.06%, 71.00%, and 68.34%. These single-seed differences do not establish a terminal-accuracy ordering. The supported conclusion is narrower. In this moderate-overlap configuration, strong heterogeneity harms every method, but overlap shows no clear additional terminal collapse relative to blocking.

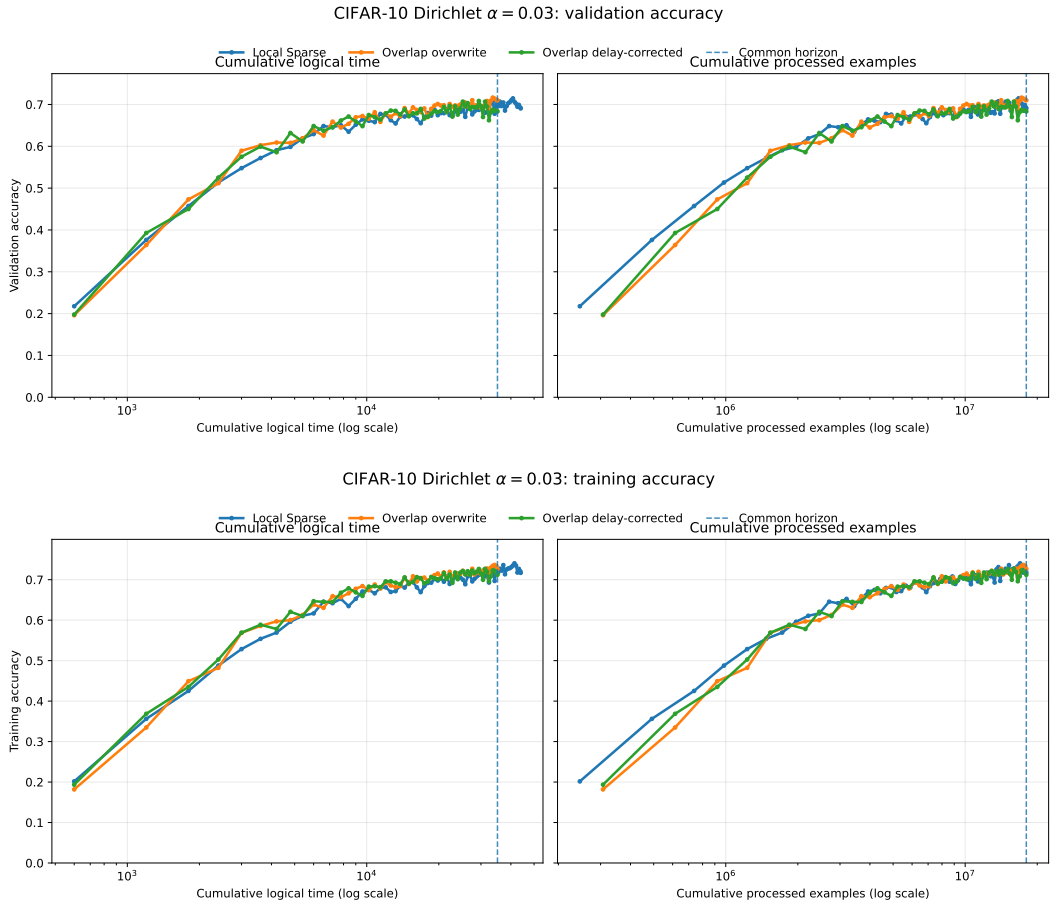


Figure 25: CIFAR-10 with strongly heterogeneous data ($\alpha = 0.03$). Top: validation accuracy; bottom: training accuracy. Within each pair, the left panel uses logical time and the right panel processed examples. The processed-example view exposes weaker early sample efficiency for overlap that its logical-time advantage hides. Dashed lines mark common horizons.

Sixteen workers. The targeted suite confirms the four-worker picture with more workers. With Dirichlet partitions at $\alpha = 0.1$ and 0.03 and the reference timing, overlap's validation-nAUC gains over LS are 0.010–0.012 and 0.010–0.016. These are comparable to the four-worker sweep at the same α (Table 11). The combined stress of heterogeneous speeds, a delay of $\zeta = 32$, and $\alpha = 0.1$ gives the largest gains in the suite, 0.039 for overwrite and 0.036 for correction (Figure 26). It also gives the largest per-example cost: at matched processed examples, overlap's averaged validation accuracy is 4.4–4.6 points below LS's. As in the four-worker case at $\alpha = 0.03$, heterogeneous data make the overlap-phase updates less sample-efficient, while elapsed time still favors overlap.

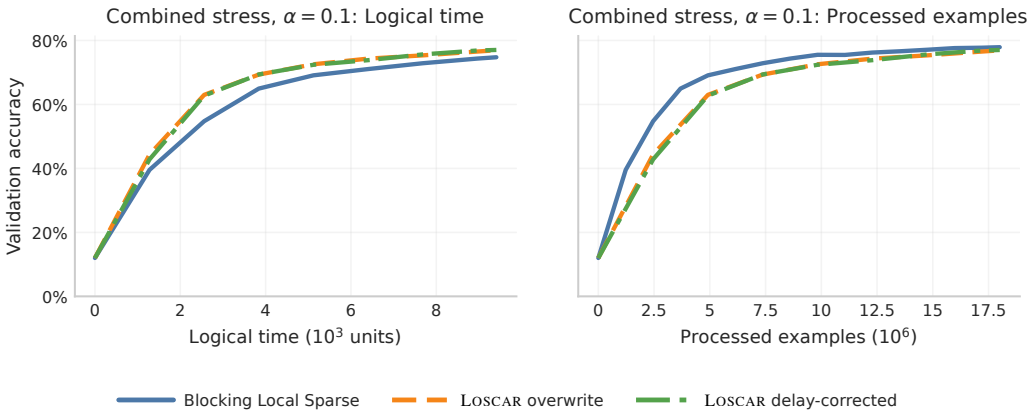


Figure 26: Sixteen-worker CIFAR-10 combined stress: heterogeneous compute, longer delay, and a label-Dirichlet partition with $\alpha = 0.1$. Validation accuracy against simulated logical time (left) and processed examples (right), truncated to the common horizon; one paired seed (11).

D.6.3 Summary

These results do not contradict the `a9a` failure of Figure 22. That test combines $\alpha = 0.03$ with $p = 0.01$, $M = 1$, and $\zeta = 96$, so overlap workers perform many more local updates per communication interval. The CIFAR-10 runs use a moderate configuration and also differ in model and training recipe. Data heterogeneity alone therefore does not decide whether overlap succeeds. What decides it is the interaction of heterogeneity with the overlap-phase work, delay, sparsity, model, and optimizer. Under moderate overlap, the main cost of heterogeneity here is weaker early sample efficiency. Under the aggressive `a9a` setting, client drift dominates and causes a terminal collapse. Because all of these settings lie outside the theory, extending the analysis to heterogeneous objectives is necessary before any guarantee can be claimed for them.

D.7 How does LoSCAR SGD compare with existing overlap methods?

We reimplemented Overlap-Local-SGD [Wang et al., 2020a], CoCoD-SGD [Shen et al., 2019], and CO2 [Sun et al., 2024] in the simulator. We compare them with LS and both LoSCAR SGD merge rules on sixteen-worker IID CIFAR-10 (Appendix D.2.2). All methods share the model, initialization, minibatch rules, inner SGD optimizer, and 400-effective-epoch budget. The grid contains 5 regimes, 6 methods, and 4 paired seeds, for 120 completed runs without numerical failure. The sparse methods communicate $p \in \{0.10, 0.25\}$, whereas the external methods communicate densely. CO2 uses outer learning rate 1.0, outer momentum 0.2, and clipping threshold 0.1.

Communication model. Let ℓ be a fixed startup latency, $\mathcal{S}$ the number of bits in a dense model-value payload, and ν the effective link bandwidth in bits per unit time. A method communicating a fraction p of the model has the continuous and grid-aligned durations

$$d_{\mathrm{comm}}(p) = \ell + \frac{p\mathcal{S}}{\nu}, \qquad \zeta(p) = \tau \left\lceil \frac{d_{\mathrm{comm}}(p)}{\tau} \right\rceil,$$

the simulator counterpart of Corollary B.3. Every synchronization pays the same startup cost, while serialization time grows with payload. The case $p = 1$ is dense communication. The five regimes are:

- **Reference:** all sixteen workers have $\tau_i = 6$. Sparse overlap methods use $p = 0.25$, $M = 4$, and $\zeta = 12$, giving $N_i = 4$ and $Q_i = 2$. Dense methods use $\zeta = 24$, giving $Q_i = 4$.
- **Bandwidth stress:** homogeneous workers and $M = 4$. Sparse overlap methods use $p = 0.25$ and $\zeta = 24$ ($Q_i = 4$); dense methods use $\zeta = 72$ ($Q_i = 12$).
- **High-sparsity stress:** homogeneous workers and $M = 4$. Sparse overlap methods use $p = 0.10$ and $\zeta = 18$ ($Q_i = 3$); dense methods use $\zeta = 72$ ($Q_i = 12$).
- **Cumulative value-payload matched:** sparse overlap methods use $p = 0.25$, $M = 4$, and $\zeta = 24$ ($Q_i = 4$). Dense methods use $M = 20$ and $\zeta = 72$ ($Q_i = 12$), so they synchronize

about four times less often, which compensates for their about four-times-larger value payload.

- **Heterogeneous long delay:** four workers at each step time in $\{1, 2, 4, 8\}$, hence $\tau = 8$, and $M = 4$. Sparse overlap methods use $p = 0.25$ and $\zeta = 32$, with $N_i = Q_i \in \{32, 16, 8, 4\}$. Dense methods use $\zeta = 96$, with $Q_i \in \{96, 48, 24, 12\}$.

LS uses the same sparse payload, communication duration, and pre-communication window as LOSCAR SGD, but waits during communication. The N_i, Q_i above describe overlap methods; LS takes only N_i steps before blocking, whereas overlap takes $N_i + Q_i$ steps within the same round duration. All methods have the same nominal 400-effective-epoch total-work budget, so LS needs more communication rounds to process that budget and can communicate more in total. The cumulative value-payload match in the fourth regime concerns the compared overlap methods, not LS; this suite does not match both per-round work and total communication across every method. In four regimes, sparse and dense overlap methods share the pre-communication parameter M, not their total time between synchronizations, and the latency–bandwidth model gives the larger dense payload a longer duration per synchronization. In the payload-matched regime, the total 32-bit model-value payload is nearly equal over training (about 38.6 versus 38.7 GB). Each dense synchronization still carries a larger payload and lasts longer. This match applies to LOSCAR SGD and the dense overlap methods, while LS retains the same nominal total-training-work budget while waiting during communication. Communication frequency, per-synchronization payload, and total payload cannot all be matched at once, so this is a resource-aware comparison of complete methods. The headline payload totals follow the value-only convention of Appendix D.1. The last paragraph reports a separate index-aware comparison.

Results. Table 12 reports the four-seed mean validation nAUC over the common simulated-time horizon of each regime. Figure 1(b) shows the payload-matched regime, and Figure 27 shows all five regimes, including heterogeneous long delay. Corrected LOSCAR SGD has the largest nAUC in four of five regimes. Across the 20 paired regime–seed comparisons, it exceeds LS, Overlap-Local-SGD, and CO2 in all 20 and CoCoD-SGD in 16. Its mean gains over LS, Overlap-Local-SGD, CoCoD-SGD, and CO2 are $+0.0249$, $+0.0263$, $+0.0110$, and $+0.0647$, respectively. In the payload-matched regime, corrected LOSCAR SGD communicates about 38.60 GB of model values, versus 38.74 GB for CoCoD-SGD and Overlap-Local-SGD. It obtains nAUC 0.7977, versus 0.7773 for CoCoD-SGD and 0.7546 for Overlap-Local-SGD. At 80% validation accuracy, it is $1.73\times$ faster than LS, $1.22\times$ faster than CoCoD-SGD, and $1.62\times$ faster than Overlap-Local-SGD in simulated time.

The result is not universal dominance. CoCoD-SGD is best in the high-sparsity stress regime (0.7911 versus 0.7884). It also has the largest mean final validation accuracy, 85.70%, followed by LS at 85.11% and corrected LOSCAR SGD at 84.75%. The comparison therefore exposes a trade-off between trajectory and endpoint. Correction contributes only modestly here (Appendix D.5).

Table 12: Mean validation nAUC in simulated time over the 120-run external-baseline grid (four paired seeds). Bold marks the largest value in each regime.

Regime	LS	LOSCAR OW	LOSCAR DC	Overlap-Local-SGD	CoCoD-SGD	CO2
Reference	0.7858	0.8011	**0.8024**	0.7899	0.8008	0.7288
Bandwidth stress	0.7654	0.7961	**0.7980**	0.7768	0.7910	0.7495
High-sparsity stress	0.7645	0.7883	0.7884	0.7777	**0.7911**	0.7478
Cumulative value-payload matched	0.7652	0.7962	**0.7977**	0.7546	0.7773	0.7113
Heterogeneous long delay	0.7380	0.7568	**0.7572**	0.7130	0.7282	0.6826
Mean	0.7638	0.7877	**0.7887**	0.7624	0.7777	0.7240

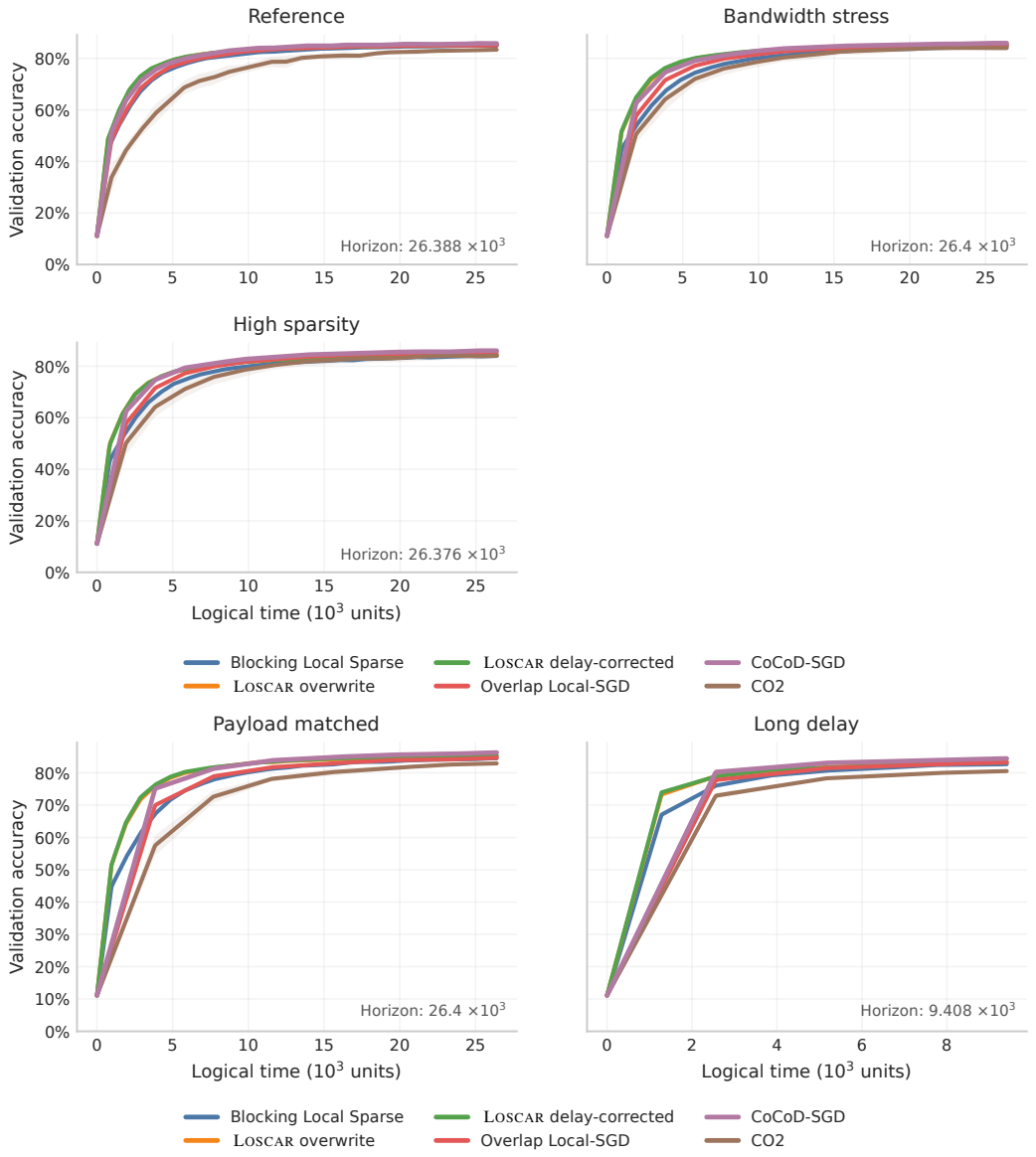


Figure 27: Validation accuracy against simulated logical time in all five external-baseline regimes: reference, bandwidth stress, high sparsity, cumulative value-payload matching, and heterogeneous long delay. Each panel ends at the common horizon used for nAUC; curves are means and bands one sample standard deviation over four paired seeds. The payload-matched panel is also shown in Figure 1(b). CoCoD-SGD has the largest nAUC in the high-sparsity regime.

Coordinate indices. The payload totals above count transmitted 32-bit model values but no coordinate identifiers. Dense messages need none, because a value's position determines its coordinate. A sparse receiver must either regenerate the support from a shared mask or seed, as shared Rand-K permits, or receive explicit identifiers. The value-only accounting corresponds to the former. To test the more conservative explicit representation, we run a focused heterogeneous-long-delay study with two paired seeds. It charges a 22-bit identifier per sparse uplink value; the downlink returns 32-bit values on the already known support. The two paired seeds are $21, 22$, with $M = 4$, $\tau = 8$, and worker times $(1, 2, 4, 8)^{\times 4}$. Unlike the value-only grid, the delay model includes index bits: a sparse bidirectional payload is a fraction $\rho = p(32 + 22 + 32)/(32 + 32)$ of the dense payload, and $\zeta = 8\lceil(8 + 88\rho)/8\rceil$. At $p = 0.25$, this gives $\zeta = 40$ for sparse methods, versus 96 for dense methods. This is a separately rerun, index-aware comparison, not a post-hoc relabeling of the headline totals. Transport headers and protocol overhead are not charged, so these remain logical totals rather

than measured traffic. At $p = 0.25$, corrected LOSCAR SGD obtains nAUC 0.7474, compared with 0.7257 for CoCoD-SGD and 0.7101 for Overlap-Local-SGD. It communicates 12.35 GB rather than 20.77 GB, a 40.5% reduction. CoCoD-SGD nevertheless ends at 84.14% validation accuracy, versus 82.84% for corrected LOSCAR SGD. This again shows the trade-off between trajectory and endpoint.

D.8 Is a shared Rand-K support the right communication operator?

How much of the model to communicate is studied in Appendices D.4.3 and D.4.4. This section asks whether the support must be shared across workers and whether shared Rand-K can be replaced by another operator with the same budget. Both ablations are design-choice studies, mainly on the CIFAR-10 settings of Table 4, not comprehensive benchmarks.

D.8.1 Shared versus independent Rand-K masks

LOSCAR SGD uses one shared support, so every selected coordinate is averaged across all workers and the corrections cancel exactly in the average (Lemma C.1). This may sacrifice the greater coverage obtained when workers select different supports. We therefore compare shared and independently sampled worker masks at a fixed number of transmitted coordinates per worker. This number is $K = 38$ ($p = 0.30$) for logistic regression and $K = 548{,}307$ ($p = 0.25$) for CIFAR-10. With independent masks, let P_j be the set of workers selecting coordinate j in a round. The server averages that coordinate over P_j only and returns it only to those workers. It does not broadcast the union of selected coordinates, which would increase the downlink budget. If $|P_j| = 1$, no cross-worker averaging occurs for coordinate j in that round. Masks are resampled every round, however, so synchronization groups change over time. A worker's current value can also already contain information from earlier synchronizations, so information still propagates across workers over several rounds.

For CIFAR-10, we evaluate five of the sixteen-worker settings in Table 4: clean IID, communication stress, compute heterogeneity, Dirichlet $\alpha = 0.1$, and combined stress. Each uses both overlap merge rules and one paired seed. Conditional on a worker selecting a coordinate, at least one of the other fifteen workers selects it with probability 98.66%, and the expected synchronization-group size is 4.75. Table 13 reports independent minus shared, so negative differences favor the shared mask.

Table 13: CIFAR-10 shared-versus-independent mask ablation over ten matched method/configuration comparisons. Differences are independent minus shared, with nAUC over the common simulated-time interval.

Metric	Independent minus shared	Shared wins
Final validation accuracy	-0.18 pp	$6/10$
Validation logical-time nAUC	-0.37 pp	$10/10$
Final training accuracy	-0.67 pp	$10/10$
Training logical-time nAUC	-0.67 pp	$10/10$

Shared masks are faster in $73/80$ time-to-target comparisons, with median $T_{\text{shared}}/T_{\text{independent}} = 0.967$, about 3.3% less simulated time. The largest difference occurs in the combined stress setting. There, independent masks reduce validation nAUC by 1.03–1.18 percentage points and training nAUC by 1.42–1.53 points, depending on the merge rule. In four sixteen-worker logistic-regression settings with five paired seeds each, the aggregate differences are close to zero. Independent masks thus remain functional. On CIFAR-10, the shared policy is modestly but consistently better, while the logistic problem shows no consistent preference. The CIFAR-10 comparisons use one seed per configuration, and only uniformly random supports were tested.

D.8.2 Alternative communication operators

Is the behavior specific to shared Rand-K? A communication-matched CIFAR-10 ablation separates the overlap framework from the choice of operator and exposes failure modes that a final-accuracy comparison would hide. It uses sixteen workers, four paired seeds 11–14, and two settings of Table 4: clean IID and combined stress. All operators use the same delay-corrected overlap framework and

are matched at about 35.09 Mbits per worker per round, counting both uplink and downlink. We test four operators:

- shared Rand-K with $p = 0.25$, which sends 32-bit values without indices because the shared support is regenerated from a seed;
- worker-wise Top-K, whose supports differ across workers and are selected from $|y_i^r - x_i^r|$; it pays a 32-bit index per value and therefore sends half as many coordinates ($p = 0.125$);
- dense INT8, which sends all coordinates as 8-bit values plus a 32-bit scale;
- $1/p$-scaled shared Rand-K, with the same support and payload as shared Rand-K.

One combined-stress Top-K run diverges, so its reported means and paired differences use the three completed seeds. Shared Rand-K and INT8 means use all four seeds; the scaled Rand-K runs diverge, as reported below. nAUC differences are relative to shared Rand-K.

Table 14: Communication-operator ablation. Accuracies are mean $\pm$ sample standard deviation in percent; nAUC differences are in percentage points relative to shared Rand-K.

Setting	Operator	Final train	Final validation	Train nAUC difference	Validation nAUC difference
Clean IID	Shared Rand-K	96.22 ± 0.05	85.42 ± 0.49	reference	reference
Clean IID	Worker Top-K	92.48 ± 0.10	82.79 ± 0.22	-2.91 ± 0.16	-2.43 ± 0.40
Clean IID	Dense INT8	61.98 ± 1.94	64.17 ± 1.73	-27.77 ± 1.67	-19.20 ± 1.43
Combined stress	Shared Rand-K	74.32 ± 2.73	73.89 ± 1.96	reference	reference
Combined stress	Worker Top-K	70.20 ± 2.10	69.85 ± 2.38	-1.87 ± 1.14	-2.16 ± 0.93
Combined stress	Dense INT8	54.31 ± 2.57	56.66 ± 2.52	-13.98 ± 1.63	-12.95 ± 1.46

Worker Top-K is slightly faster at low targets, about 1.15–$1.22\times$ at 25%–50% accuracy under combined stress, but the advantage later reverses. In clean validation, Top-K needs about 1.15, 1.58, and 2.43 times more simulated time than shared Rand-K to reach 75%, 80%, and 82%, and it never reaches 85%. Under combined stress, it needs about 1.37 times more time to reach 65% and never reaches 75%. Post-merge disagreement $\sum_{i=1}^{n} \|x_i - \bar{x}\|^2$ provides a diagnostic consistent with this deterioration, not a causal test. At the final finite iterate, disagreement is about $7.0\times$ larger for Top-K in clean IID and $5.8\times$ larger over the completed combined-stress pairs. All eight $1/p$-scaled Rand-K runs diverge. The rescaling amplifies the synchronization correction; these runs demonstrate instability for this operator and recipe, not for unbiased compression in general.

Limitations of the operator study. The INT8 comparator is deliberately simple: deterministic symmetric eight-bit quantization with one scale for the entire flattened model. It has no per-layer or per-channel scales, stochastic rounding, calibration, error feedback, or residual accumulation, all of which can materially improve low-precision communication. It is a transparent budget-matched reference, not a state-of-the-art quantizer, and its weak result is not a general limitation of INT8 training. Likewise, Top-K is tested only in its worker-wise form at the matched bit budget. The ablation is a design-choice study, not a comprehensive compressor benchmark.

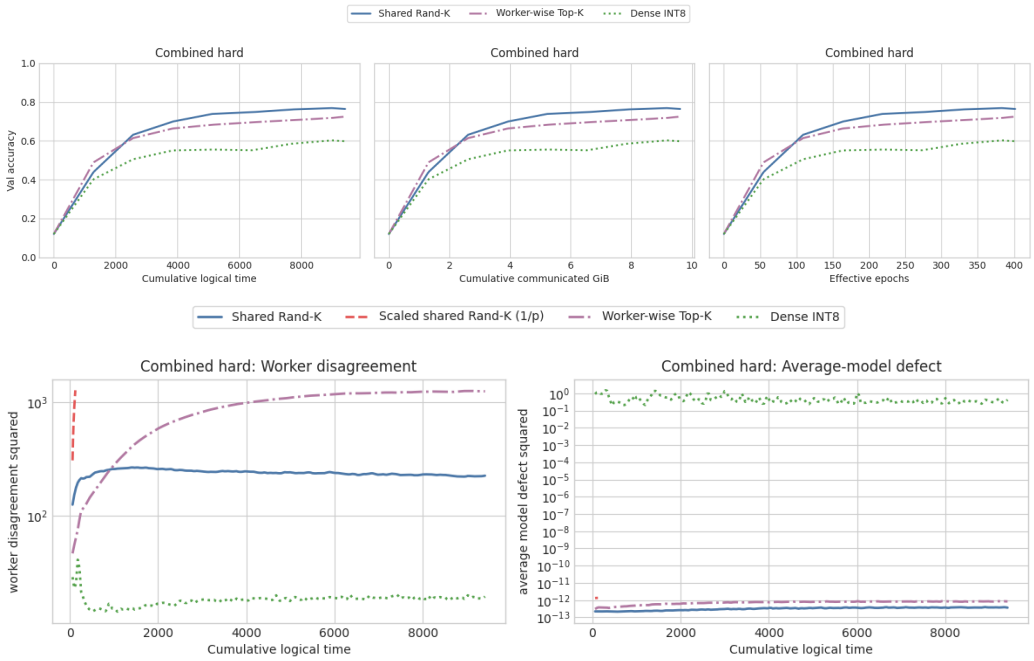


Figure 28: Communication-operator ablation under combined stress. Top: validation accuracy against simulated logical time, communicated GiB, and effective epochs. Bottom: worker disagreement and average-model defect against logical time. The short red dashed prefix records the divergent $1/p$-scaled Rand-K run rather than omitting it.

D.9 Do the gains hold on real hardware?

The simulator fixes the timing model by construction. This section checks the mechanism on real hardware: computation and collective completion are measured, while additional minimum result-availability windows are controlled experimentally.

Setup. We implemented work-matched LS, overwrite, and delay correction in PyTorch Distributed and ran them on four NVIDIA V100 GPUs across two compute nodes, using asynchronous NCCL all-reduce over Ethernet. We train the CIFAR-10 CNN of Appendix D.2.2 (2.19M parameters) and a CIFAR-adapted ResNet-18 with GroupNorm (11.17M parameters). Both use $p \in \{0.25, 0.50\}$ and 100 effective epochs, with four paired seeds planned per configuration. The suite contains 16 calibration jobs and 190 completed training runs, yielding 62 complete method triplets. The planned seeds are 11–14. At $p = 0.50$, the small CNN lacks the overwrite run for seed 13 under native communication and seed 11 at 100 ms; paired summaries use the three complete triplets in those two settings and four elsewhere. All 250 ms configurations have four complete triplets. Within a triplet, the methods share initialization, data split, minibatches, augmentation, Rand-K masks, optimizer, processed-example budget, communication count, and total number of local SGD steps.

Controlled result-availability windows and calibration. Besides native Ethernet communication, the implementation imposes minimum result-availability windows of 25, 100, and 250 ms. The NCCL collective must complete, but its result cannot be merged before the deadline. These windows are application-level controls, not WAN emulation or bandwidth throttling. For each model, p, and window, we calibrate the communication time T_{comm} and local-step time T_{step}, and set $Q^\star = \lceil T_{\mathrm{comm}}/T_{\mathrm{step}} \rceil$. At the 250 ms window, $Q^\star$ is 6 and 9 for ResNet18-GN at $p = 0.25$ and 0.50, and 14 and 13 for the small CNN (Table 2). The corresponding native sparse all-reduce times are about 102 ms, 287–380 ms, 30 ms, and 59 ms.

What is timed. The training timer includes local optimization, mask generation, sparse packing, NCCL communication, waiting, the controlled availability delay, and merging. It excludes initialization and periodic dense evaluation. The hidden fraction is the share of the measured com-

munication window covered by useful local computation; LS hides 0% by construction. End-to-end time additionally includes periodic dense evaluation, which all methods share and none overlaps.

Training-time results. Table 2 reports the $250\,\mathrm{ms}$ setting, and Figure 29 shows the trajectories. For the small CNN at $p = 0.50$, LS exposes about $256\,\mathrm{ms}$ of waiting per round, versus $13\,\mathrm{ms}$ for overwrite and $24\,\mathrm{ms}$ for correction. Throughput rises from about $7{,}580$ to $12{,}919$ and $12{,}694$ images/s. At $p = 0.25$, waiting falls from about $266\,\mathrm{ms}$ to 16 and $30\,\mathrm{ms}$, and throughput rises from about $7{,}591$ to $12{,}928$ and $12{,}528$ images/s. With periodic dense evaluation included, end-to-end speedups shrink to about $1.16\times$–$1.27\times$. Both overlap variants reach every target in Table 15 in all four paired seeds.

Table 15: Mean paired measured-wall-clock speedup over LS to fixed validation targets at the $250\,\mathrm{ms}$ window, as overwrite / delay-corrected.

Model, p	70%	80%	82%	85%
ResNet18-GN, 0.25	1.38/1.22	1.36/1.25	1.35/1.27	1.37/1.24
ResNet18-GN, 0.50	1.75/1.29	1.68/1.30	1.56/1.24	1.60/1.37
Small CNN, 0.25	1.49/1.33	1.68/1.40	1.55/1.37	1.93/1.27
Small CNN, 0.50	1.38/1.37	1.63/1.37	1.78/1.39	1.80/1.20

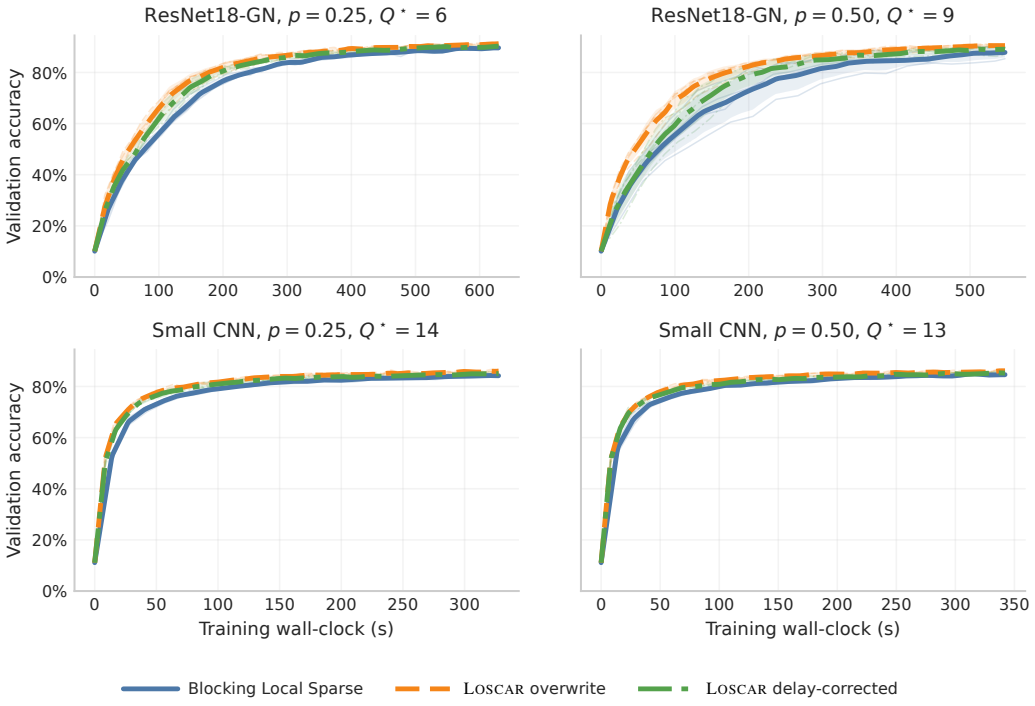


Figure 29: Measured validation trajectories at the $250\,\mathrm{ms}$ window. Rows: ResNet18-GN and the small CNN; columns: $p = 0.25$ and $p = 0.50$, with the calibrated $Q^\star$ in the panel titles. Pale curves are paired seeds, thick curves means, and bands one sample standard deviation. The horizontal axis is cumulative measured training wall-clock time, excluding periodic dense evaluation; the top-left panel is Figure 1(c).

Dependence on the communication window. Figure 30 shows the speedup and hidden fraction from native communication up to the $250\,\mathrm{ms}$ window. For both models, the median training-time speedup is already above one with native communication. For the small CNN, it increases with the imposed window, and so does the hidden fraction. For ResNet18-GN, the speedup is not monotone in the window: it falls to about parity at $25\,\mathrm{ms}$ and is highest at the longer windows. Hiding

communication therefore pays most when the exposed communication time is long relative to a local step.

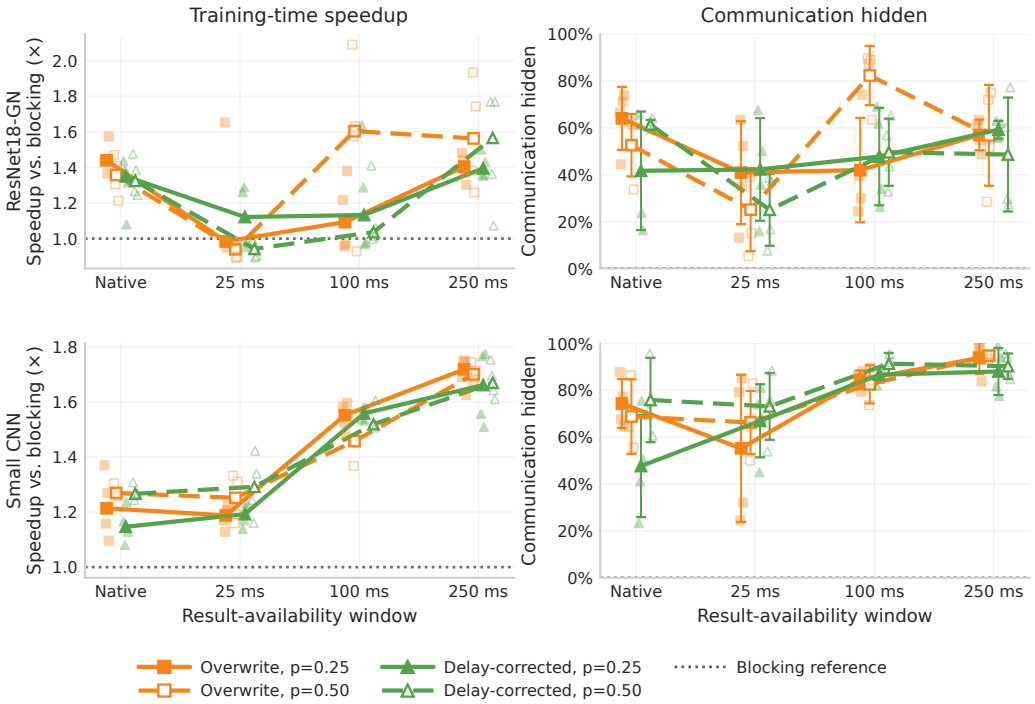


Figure 30: Measured delay response for ResNet18-GN (top) and the small CNN (bottom), from native communication to controlled 25, 100, and 250 ms windows. Left: full-training-time speedup over LS; right: fraction of communication hidden. Pale markers are the available paired seeds (three or four, as detailed above); speedup curves connect across-seed medians, and hidden-fraction curves show means with sample-standard-deviation error bars. Filled and open markers distinguish $p = 0.25$ and $p = 0.50$. The windows are controlled result-availability windows, not WAN emulation.

Wall-clock versus sample efficiency. Wall-clock validation nAUC combines systems and optimization effects. It improves relative to LS for overwrite in every seed and configuration, and for correction in $15/16$ comparisons. The mean overwrite/corrected differences are $+0.0408/+0.0252$, $+0.0661/+0.0284$, $+0.0287/+0.0189$, and $+0.0247/+0.0150$ for the four rows of Table 2. Against effective epochs, overwrite is about neutral and correction somewhat worse, with differences of $+0.0008/-0.0167$, $-0.0013/-0.0177$, $+0.0010/-0.0077$, and $-0.0023/-0.0113$. The main benefit is therefore measured wall-clock efficiency, not uniformly better sample efficiency. Correction is not preferred in these momentum-based runs. Appendix D.5 discusses this result together with an earlier momentum-free hardware study.

Scope. These are real measurements on a modest four-GPU, two-node Ethernet system. The controlled windows emulate slower result availability, not a particular wide-area network. The results confirm that overlap hides a substantial part of measured communication and shortens training time, but they do not support universal systems claims.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction describe the shared-objective theory, the communication-computation overlap mechanism, and the regime-dependent behavior of the merge rules. These claims are developed in Sections 3 to 5 and the appendices.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Sections 2, 5 and 6 discuss the fixed timing and shared-objective assumptions, the limits of the theoretical comparisons, failure regimes under sparsity and heterogeneity, and the modest scale of the hardware evaluation. The non-IID experiments are explicitly identified as stress tests outside the guarantees.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The assumptions are stated in Sections 2 and 4. The main convergence and time-complexity proofs appear in Appendix C, and the additional theoretical results and their proofs appear in Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 5 and appendix D specify the algorithms, datasets, models, logical-time simulator, hardware protocol, hyperparameters, paired-seed comparisons, and metric and communication-accounting conventions.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper identifies the public LIBSVM, CIFAR-10, and Tiny ImageNet datasets and gives implementation and experimental details in Appendix D. It does not provide a public code-release URL or complete run commands for the reported experiments.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 5 and appendix D give the objectives, data splits, models, augmentation, optimizers, learning rates, training budgets, sparsity, communication timing, worker profiles, and hardware settings for the reported studies.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: The multi-seed results report means with caption-defined uncertainty bands or error bars, distinguishing 95% confidence intervals from sample standard deviations. Seed counts and paired comparisons are specified in Appendix D; single-seed studies and configuration-grid summaries are identified as descriptive rather than estimates of across-seed variability.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: The hardware study identifies the four NVIDIA V100 GPUs across two nodes and reports measured timing and throughput. However, complete per-run memory and resource accounting for all studies, including the total research compute budget, is not provided.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work studies standard distributed optimization methods on public benchmark datasets, involves no human subjects, and does not introduce any procedure that conflicts with the NeurIPS Code of Ethics.

   Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This is a foundational optimization paper rather than an application or deployment paper, so we do not identify direct societal impacts that are specific to this work.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release high-risk models or datasets such as generative models or scraped web-scale data that would require special safeguards.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: The paper identifies the public LIBSVM, CIFAR-10, and Tiny ImageNet datasets and the software used, but it does not provide complete license and terms-of-use information for these assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce or release new datasets, models, or other assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects, so IRB approval or an equivalent review was not required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not part of the method, theory, or experiments in this work.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.