PACE: PARAMETER AVERAGING WITH CONSENSUS EXTRAPOLATION FOR DISTRIBUTED TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing methods for training language models across slow, heterogeneous, or unreliable networks utilize infrequent or compressed gradient exchanges to reduce communication traffic. However, this creates two problems. *a)* They still pause computation while communicating and *b)* work best only when every worker remains available throughout training. Asynchronous sparse parameter averaging avoids both constraints: workers exchange a random subset of model weights in the background, absent workers contribute nothing, and stale workers are naturally pulled back when they return. Its weakness is optimization quality. We show that the missing ingredient is *momentum on the mean motion* of the worker population. That motion is *already present* in each sparse average, so PACE recovers it as a shared consensus-velocity buffer without sending any additional data. Every worker reconstructs the same buffer and uses it in a Nesterov-style update. We also propose accumulate-to-match (ATM) for handling heterogeneous hardware. ATM lets each worker take an optimizer step only after processing the same number of samples, so slower workers become stale instead of providing noisy updates due to lower batch sizes. Sparse averaging handles this bounded staleness much better than such gradient noise. Together, these changes turn sparse parameter averaging from a weaker optimizer into the strongest communication-efficient method in heterogeneous and unreliable settings.

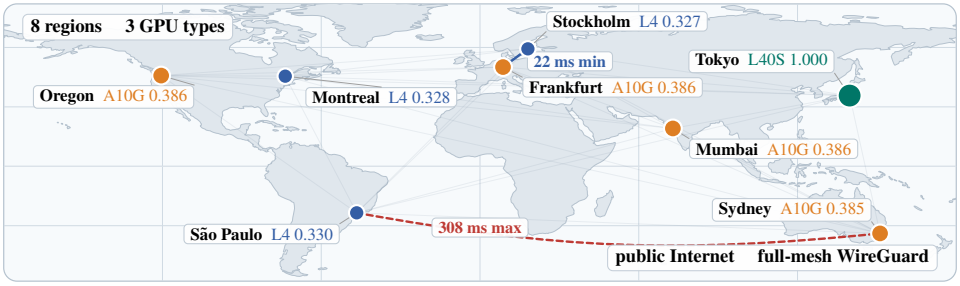
1 INTRODUCTION

Scaling language models usually relies on synchronous data parallelism (DDP), where every worker exchanges a full gradient before the next step. This is suitable for a fast cluster, but not viable when workers spread across regions with Internet-grade links or drawn from mixed hardware: the network has to carry a full-model message every step, and the fastest worker has to wait for the slowest. Communication-efficient methods tackle this by communicating infrequently or compressing updates, overlapping transfers with computation, or replacing global collectives with asynchronous server- or peer-to-peer exchange (Douillard et al., 2023; Sarfi et al., 2025; Douillard et al., 2025; Liu et al., 2024; Lian et al., 2018; Ryabinin et al., 2021; **?**).

We separate two types of blocking that are often conflated. *Quorum blocking* asks whether an exchange requires every worker to be present; if so, one straggler stalls the fleet and a returning worker must wait for a round boundary. *Communication blocking* asks whether a worker stops computing while its message is transferred; if so, slow links directly reduce utilization. Existing methods typically remove only one of the two (Table 1). Streaming DiLoCo Douillard et al. (2025) overlaps communication but still needs a quorum. Asynchronous DiLoCo Liu et al. (2024) and gossip remove the global quorum, but either rely on a server or pause the workers involved in an exchange. More fundamentally, all of these methods communicate a *change* of state, accumulated over a fixed local budget. That change is difficult to compare when workers run at different speeds or disappear partway through a round.

Instead, sparse parameter averaging communicates *state*. SPARTA (Beton et al., 2025; Fournier et al., 2024) has every worker average the same randomly selected fraction of its parameters at each exchange. It was introduced as a lightweight exchange between DiLoCo outer communication steps, and its authors found that the selected parameters could be several steps stale without hurting train-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



	blocks on	needs a coordinator	held-out loss $\bar{\theta}$	% wall blocked	wall-clock
PACE (ours)	**neither**	**no**	3.478	**0.9%**	**17.4 h**
async DiLoCo + DyLU	B	yes[†]	3.520	37.7%	27.8 h
SparseLoCo[‡]	A + B	no	3.59	56%	36 h
cannot be run on this fleet					
DiLoCo	A + B	no	—	every round gated by the slowest replica; fastest idles 67%	
DDP	A + B	no	—	no step completed within the 900 s timeout	

Figure 1: Eight-region WAN deployment with a 500M model. Marker color identifies the GPU type and marker size shows measured throughput. The public-internet WireGuard mesh spans 22–308 ms RTT. PACE reaches 3.478 held-out loss in 17.4 hours while exposing only 0.9% communication time; DDP does not complete a step within the 900-second timeout.

ing. Async SPARTA (Ajanthan et al., 2026) later overlapped sparse averaging with local training and introduced a delay corrected sparse averaging with EMA of local staleness for robustness.

State averaging is attractive on an imperfect fleet for three reasons. First, the average is well defined for any set of participants: an absent worker adds neither a value nor a weight. A missing gradient or pseudo-gradient, by contrast, changes the update and must be handled by a barrier, a retry, or a server. Second, the same correction naturally repairs a returning worker. Its local state may be stale or may have drifted while disconnected, but subsequent averages pull it back toward the fleet without a checkpoint or a special rejoining protocol. Third, parameter averaging already tolerates delay, so the transfer can be overlapped with computation.

Nonetheless, these systems properties do not make (async) SPARTA a strong optimizer. In our fault-free experiments, it is beaten by DiLoCo Douillard et al. (2023) and its variants in nearly every regime and at every scale. The difference is what each method retains from an exchange. DiLoCo carries the average change made by the workers into an outer momentum step. Sparse parameter averaging observes the same population trajectory, but uses each observation (i.e., weight average) only to reduce disagreement. Interestingly, we note that its correction is mean-zero across workers (Eq. 3), so the direction in which the fleet moved is already present in the exchanged values. We highlight this as a crucial signal: *momentum on the mean motion of the replica population*. Further, uniform parameter averaging has a second weakness under heterogeneity: if slow workers use smaller batches to keep pace, their updates have higher variance. The average remains unbiased, but its noise floor is tied to which workers are slow.

We introduce PACE[1] to recover the discarded motion without changing the sparse exchange. Consecutive observations of the averaged coordinates give a per-coordinate fleet velocity, which we smooth into a momentum buffer M. Every worker receives the same averaged values and uses the same mask, so it reconstructs the same buffer locally with no additional communication. In cases where a worker drops out and its local momentum becomes stale, we show that the workers' momentum states quickly return to consensus after it rejoins. Each local step evaluates its gradient at a look-ahead, takes an ordinary AdamW step, and then applies the same velocity as an assist. We also introduce *echo cancellation*, which removes motion caused by earlier assists, while the preserving the norm of the weight matrices to prevent them from growing in magnitude.

In PACE, the weight exchange remains fully asynchronous (overlaped with computation), and the average is normalized by the mass of the workers. Specifically, no worker waits for a global quorum, no coordinator owns the model, and a worker that returns with stale weights is automatically repaired. Thus PACE keeps the desirable systems properties of async SPARTA while recovering the optimization signal that it loses.

[1]Parameter Averaging with Consensus Extrapolation.

We propose a further correction to the heterogeneity. If a worker with relative speed ρ uses a proportionally smaller batch, its gradient has roughly $1/\rho$ times the variance. Weighting can reduce that worker's influence, but only by discarding some of its signal. Sparse state averaging is much better suited to bounded staleness: repeated weight mixing reduces a stale worker's distance from the fleet consensus, and we show that this gap approaches a fixed point within a bounded-delay model.

We propose accumulate-to-match (ATM) to tackle this. Every worker forms an optimizer step from the same number of samples; a worker at relative speed ρ simply takes $1/\rho$ times as long for a step. This enforces the optimizer steps to have the same noise scale, and therefore, heterogeneity appears only as bounded staleness. When an accumulation window spans several ticks, the worker also holds its look-ahead fixed so that all microbatches in that step are evaluated at one point. ATM therefore converts speed-dependent gradient noise, which persists in the fleet average, into staleness, which sparse weight averaging continuously reduces.

Contributions. (i) We introduce PACE, a fully non-blocking, coordinator-free method that reconstructs consensus-velocity momentum from sparse delayed parameter averages without sending another message. (ii) We introduce ATM, which equalizes the sample count of every optimizer step and replaces speed-dependent gradient noise with bounded staleness. (iii) We evaluate the method across 226M, 500M, and 1B models, heterogeneous hardware, outages and session churn, larger replica counts, and a real eight-region deployment, and we analyze the effect of consensus correction, velocity estimator, and sparse delay both theoretically and empirically.

2 Related work

Local updates and compressed synchronization. Local SGD and FedAvg take several local steps between averages (Stich, 2019; McMahan et al., 2017). DiLoCo makes long horizons practical for language-model pretraining with an outer Nesterov update on the average model change (Douillard et al., 2023); SparseLoCo compresses that change with sparsification, quantization, and error feedback (Sarfi et al., 2025), Streaming DiLoCo spreads it across the inner steps (Douillard et al., 2025), and FG-DiLoCo splits synchronization into an overlapped parameter-mixing step and a smaller blocking gradient-mixing step (Koneputugodage et al., 2026). All of them communicate a *change* measured from a shared anchor, whose summands are hard to compare once workers make different progress or hold different anchors.

Table 1: The two blocking axes. Async DiLoCo removes the quorum by using a parameter server; the state-mixing methods do not require one.

Method	Communicates	No quorum	Overlaps traffic
DDP	gradients	✗	✗
DiLoCo / SparseLoCo	changes	✗	✗
Streaming DiLoCo	changes	✗	✓
Async DiLoCo	changes	✓†	✗
AD-PSGD / Moshpit / NoLoCo	state	✓	partial
Async SPARTA	sparse state	✓	✓
PACE	sparse state	✓	✓

Decentralized parameter mixing. AD-PSGD, Moshpit SGD, and NoLoCo mix model states through peers or small groups (Lian et al., 2018; Ryabinin et al., 2021; Kolehmainen et al., 2025); SPARTA mixes the same randomly selected coordinates across workers (Beton et al., 2025); Async SPARTA introduces delay corrected sparse averaging to overalp communicaiotn with computation and considers an EMA of local stalenss for further robustness in a fully asynchronous PP x DP mesh. (Ajanthan et al., 2026); LOSCAR-SGD adds worker-specific amounts of local computation and a convergence analysis for that setting (Maziane et al., 2026). PACE keeps this substrate and extend what is learned from an exchange: the average restores consensus and also reveals the direction in which the fleet is moving.

Heterogeneity and momentum. FedProx and SCAFFOLD correct heterogeneous local training through server-mediated updates (Li et al., 2020b; Karimireddy et al., 2020), and dynamic local updates scale work with device speed (Liu et al., 2024). ATM addresses a different error: it equalizes the number of samples behind each optimizer step. The consensus-velocity buffer is likewise distinct from Polyak or Nesterov momentum (Polyak, 1964; Nesterov, 1983), tracking cross-worker model motion rather than one worker's gradient history.

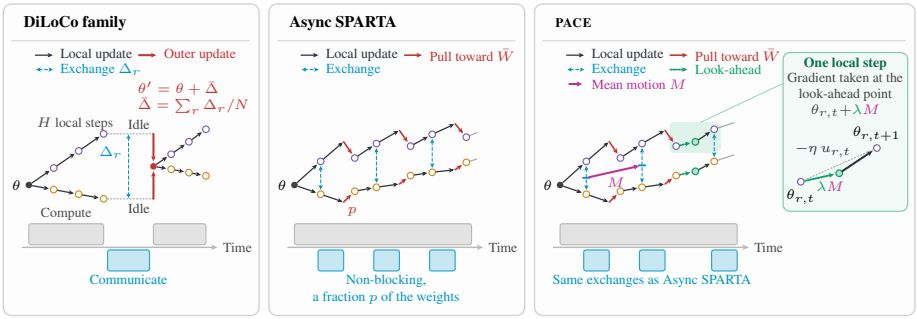


Figure 2: PACE vs. DiLoCo and Async SPARTA. The parameters of two workers (violet, amber) over the same time axis as the compute / communication bar below; all three start from θ. DiLoCo idles during the exchange and the mean change $\bar{\Delta}$ resets both workers to $\theta' = \theta + \bar{\Delta}$. Async SPARTA keeps stepping and pulls each worker a fraction p of the way toward the delayed average $\bar{W}$, so the gap shrinks but never closes. PACE uses the same exchanges; consecutive averages define the mean motion $M = \bar{W}_t - \bar{W}_{t-1}$, and every local step is taken Nesterov style (enlarged at right): shift by λM, take the gradient there, and step $-\eta u_{r,t}$ with $u_{r,t}$ the resulting AdamW direction.

3 Problem Setting

We consider N workers with parameters $\boldsymbol{\theta}_r \in \mathbb{R}^d$. Workers read disjoint IID data shards and run AdamW locally (Loshchilov & Hutter, 2019). Using the terminology from Section 1, we call a method fully non-blocking only when it avoids both quorum and communication blocking. The distinction matters because overlap alone does not survive a missing participant, and asynchrony alone can still leave workers idle on slow links.

Table 1 also separates methods by what they send. Change-based methods aggregate gradients or pseudo-gradients measured relative to an anchor. State-based methods average parameters. We note that an average of the workers that are currently present in a fleet is still a valid state average, and repeated state mixing can repair a stale worker (when dropped and rejoined). These properties motivate our choice of substrate. For comparison, DiLoCo communicates the anchored change $\Delta_r = \theta^{\text{anchor}} - \theta_r^{(H)}$ and applies $m \leftarrow \beta m + \bar{\Delta}$, $\theta^{\text{anchor}} \leftarrow \theta^{\text{anchor}} - \alpha(\bar{\Delta} + \beta m)$ with $\bar{\Delta} = \frac{1}{N}\sum_r \Delta_r$. This outer step needs comparable changes and a synchronized aggregate; PACE instead estimates the same fleet-level motion from state (weight) averages.

4 Method

Figure 2 shows the progression from the DiLoCo family to PACE. DiLoCo obtains a fleet-level update at a blocking round boundary. Async SPARTA removes that barrier by mixing delayed model states in the background. PACE retains that mechanism and leaves its communication unchanged. It additionally recovers the motion of the fleet mean from consecutive returned averages and uses this signal as a shared look-ahead for local optimization.

Writing $u_{r,t}$ for the AdamW direction evaluated at the look-ahead $\theta_{r,t} + \lambda_t M_t$, the core parameter update is

$$\theta_{r,t+1} = \mathcal{A}_{\lambda_t M_t}(\theta_{r,t} - \eta_t u_{r,t}) + \mathbf{1}\{t \bmod R = 0\}\ (\bar{W} - W_r^{\text{snap}})\big|_{\mathcal{I}_{t-D}}, \tag{1}$$

where $\mathcal{A}$ is the tensor-wise assist in Eq. (6). Thus one step consists of local optimization, a shared consensus-velocity assist, and—only every R steps—a sparse correction.

4.1 Sparse, Delayed Parameter Averaging

Every R steps, a shared random seed selects a fraction p of model coordinates, denoted by $\mathcal{I}_t$. Worker r snapshots those coordinates and starts a weighted all-reduce. Training continues for D steps while the transfer runs. When it completes, each participating worker applies

$$\bar{W}[i] = \frac{\sum_{r\in\mathcal{P}_t} \rho_r W_r^{\text{snap}}[i]}{\sum_{r\in\mathcal{P}_t} \rho_r}, \qquad \theta_r[i] \leftarrow \theta_r[i] + \bar{W}[i] - W_r^{\text{snap}}[i], \quad i \in \mathcal{I}_{t-D}, \tag{2}$$

where $\mathcal{P}_t$ is the set of workers that contributed and ρ_r is worker r's relative throughput. A missing worker does not add any value or mass, so the exchange remains well defined. This is the delay-corrected sparse average introduced by Async SPARTA (Ajanthan et al., 2026) and adopted by LOSCAR-SGD (Maziane et al., 2026): applying $\bar{W} - W_r^{\text{snap}}$ to the current parameters replaces

the transmitted snapshot with the fleet average while preserving every local update made during the delay. The correction is mean-zero under the same weights:

$$\sum_r w_r(\bar{W}[i] - W_r^{\text{snap}}[i]) = \bar{W}[i] - \bar{W}[i] = 0. \tag{3}$$

It therefore reduces worker disagreement without moving the fleet mean. The amortized payload is p/R of the model per worker per step, and a coordinate is refreshed every R/p steps on average. Appendix D varies the sparse fraction; its stability boundary and measured R/p scaling are in Appendices D.1 and H.2. Effect of delay is shown in Appendix E.

4.2 Consensus-velocity momentum

Async SPARTA introduced this delay-corrected sparse average and adds an exponential moving average of local staleness on top of it (Ajanthan et al., 2026); LOSCAR-SGD adopts the same correction for its consensus analysis (Maziane et al., 2026). Neither method extracts the fleet-mean velocity from consecutive sparse averages. We do so by comparing two observations of the same coordinate. Let $\bar{W}^-[i]$ be the previous observation and Δt_i the elapsed number of averaging windows. If $X_{r,s}$ denotes worker r's parameters after AdamW but before the assist at local step s, the applied assist is $d_{r,s} = \mathcal{A}_{\lambda_s M_s}(X_{r,s}) - X_{r,s}$ and its coordinate-wise accumulator is $Q_i \leftarrow Q_i + d_{r,s}[i]$, with Q_i reset to zero whenever coordinate i is observed. The velocity update is then

$$v_i^{\text{raw}} = \frac{\bar{W}[i] - \bar{W}^-[i]}{\Delta t_i}, \qquad \widehat{v}_i = v_i^{\text{raw}} - \frac{Q_i}{\Delta t_i}, \qquad M_i \leftarrow \mu M_i + (1-\mu)\widehat{v}_i. \tag{4}$$

The subtraction removes motion caused by the buffer itself. Without it, the estimator reinforces its own output which causes the update to blow-up. This exact *echo cancellation* is what allows the strong coupling used in our runs. Every term is derived from the shared mask, the all-reduced average, or deterministic local bookkeeping, so workers construct the same M without any additional data transfer.

At each shared training tick t, the gradient is evaluated at $\theta_r + \lambda_t M$. After AdamW, the same displacement is applied using

$$\lambda_t = \lambda_{\text{hi}} + (\lambda_{\text{lo}} - \lambda_{\text{hi}}) \min(1, t/W). \tag{5}$$

Here W is the number of shared ticks in the learning-rate warmup. Thus the coupling moves linearly from λ_{hi} to λ_{lo} during warmup and remains at λ_{lo} afterwards. For a hidden two-dimensional tensor X, the deployed assist preserves its pre-assist Frobenius norm; embeddings, the output head, and one-dimensional tensors use the additive branch:

$$\mathcal{A}_\delta(X) = \begin{cases} (X+\delta)\dfrac{\|X\|_F}{\|X+\delta\|_F}, & X \text{ hidden and 2D}, \\ X + \delta, & \text{otherwise}. \end{cases} \qquad \delta = \lambda_t M_X. \tag{6}$$

Norm preservation guards against magnitude growth. At window scale,

$$\mu_{\text{eff}} = \lambda_t R(1-\mu), \tag{7}$$

which can exceed one, whereas the classical Nesterov momentum coefficient is restricted to $0 \leq \beta < 1$. Appendix F ablates the effect of coupling schedule, and Appendices C.2 and C.3 examine assist magnitude and alignment.

4.3 Accumulate-to-match

Figure 3 shows the entire mechanism: ATM uses gradient accumulation to keep the local optimizer batch fixed. Suppose worker r processes samples at relative rate ρ_r. A common way to keep a shared step clock is to give that worker a batch proportional to ρ_r. Its gradient variance then grows roughly as $1/\rho_r$, so the slowest devices inject the noisiest updates.

In contrast, ATM keeps the target batch B fixed. If worker r processes ν_r samples per second and $\rho_r = \nu_r / \max_s \nu_s$, then $B_r = B$, its step time is $\tau_r = B/\nu_r = \tau/\rho_r$, and it steps once every $k_r = \tau_r/\tau = 1/\rho_r$ ticks. Every optimizer step therefore averages B samples and has variance scale σ^2/B; slow workers take fewer steps in wall-clock time, so their states are older when mixed.

Algorithm 1 PACE on worker r at shared tick t.

Require: $\widetilde{\lambda}_r$ initialized to λ_{hi}
Require: $\widetilde{M}_r$ initialized to zero
$\lambda_t \leftarrow \lambda_{\mathrm{hi}} + (\lambda_{\mathrm{lo}} - \lambda_{\mathrm{hi}}) \min(1, t/W)$
$k_r \leftarrow \max(1, \mathrm{round}(1/\rho_r))$ ▷ ATM accumulation length
$G_r \mathrel{+}= \sum_{z \in \mathcal{B}_{r,t}} \nabla \ell(\theta_r + \widetilde{\lambda}_r \widetilde{M}_r; z)$ ▷ fixed look-ahead within the window
if $t \bmod k_r = 0$ **then**
  $X_r \leftarrow \mathrm{AdamW}(\theta_r, G_r/B)$ ▷ take the local optimizer step
  $\theta_r \leftarrow \mathcal{A}_{\lambda_t M}(X_r)$ ▷ apply the shared-velocity assist
  $Q \mathrel{+}= \theta_r - X_r$ ▷ accumulate the local echo
  $G_r \leftarrow 0$
  $(\widetilde{\lambda}_r, \widetilde{M}_r) \leftarrow (\lambda_t, M)$ ▷ look-ahead for the next window
end if
if $t \bmod R = 0$ and $t \geq D$ **then**
  receive $\bar{W}$ ▷ completed sparse average
  $\theta_r[\mathcal{I}_{t-D}] \mathrel{+}= \bar{W} - W_r^{\mathrm{snap}}$ ▷ repair the selected coordinates
  for $i \in \mathcal{I}_{t-D}$ **do**
    $\Delta t_i \leftarrow (t - D - \tau_i)/R$
    $\widehat{v}_i \leftarrow (\bar{W}_i - \bar{W}_i^- - Q_i)/\Delta t_i$ ▷ remove the assist echo
    $M_i \leftarrow \mu M_i + (1-\mu)\widehat{v}_i$ ▷ update consensus velocity
    $(\bar{W}_i^-, \tau_i, Q_i) \leftarrow (\bar{W}_i, t - D, 0)$
  end for
end if
if $(t + D) \bmod R = 0$ **then**
  $W_r^{\mathrm{snap}} \leftarrow \theta_r[\mathcal{I}_{t+D}]$ ▷ snapshot the shared mask
  start $\mathrm{AllReduce}(\rho_r W_r^{\mathrm{snap}}, \rho_r)$ ▷ nonblocking weighted exchange
end if

This is a favorable trade for state averaging. If local training creates a gap Δ between mixing events and an exchange removes a fraction α of the current gap, then

$$\Gamma_{t+1} = (1-\alpha)\Gamma_t + \Delta, \qquad \Gamma_\infty = \Delta/\alpha. \tag{8}$$

Staleness therefore approaches a finite level and falls as the learning rate decays. Note that speed-dependent gradient noise has no such contraction. In the implementation, capacity weights in Eq. (2) make the consensus represent work completed per unit wall-clock time. Appendix K derives the fixed point and its bounded-delay extension.

4.4 Fixed look-ahead within an accumulation window

Under ATM, a worker with capacity ρ_r accumulates samples for $k_r = \mathrm{round}(1/\rho_r)$ ticks before taking an optimizer step. During those ticks, both λ_t and M may change. Evaluating each microbatch at the current $\theta_r + \lambda_t M$ would therefore combine gradients from several look-ahead points in one optimizer step.

Instead, the worker records $(\widetilde{\lambda}_r, \widetilde{M}_r)$ after each step and uses that pair throughout the next accumulation window. All microbatches in the window are then evaluated at one shifted point. For a full-speed worker, $k_r = 1$, so the pair is refreshed every tick and the update is unchanged. The same is true throughout the homogeneous and outage settings, where every worker has unit capacity.

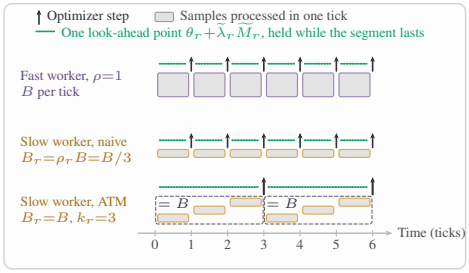


Figure 3: ATM converts gradient noise to staleness. Blocks are the samples a worker processes in one tick of the shared clock; each arrow marks an optimizer step at the instant it happens. The fast worker (violet, ρ=1) sees B samples per tick and steps every tick. Scaling the slow worker's batch to $B_r = \rho_r B$ (middle) keeps it on the clock, but every step then averages only a third of the samples: the cost of its slowness is paid as gradient noise. ATM (bottom) keeps $B_r = B$ instead: the slow worker accumulates $B/3$ per tick and steps once every $k_r = 3$ ticks with the full batch (dashed outline), so each step is as clean as the fast worker's and the cost is paid as fewer, older states, which state averaging repairs (Eq. 8). Each green segment is one look-ahead point $\theta_r + \widetilde{\lambda}_r \widetilde{M}_r$: refreshed every tick when the worker steps every tick, and held for the whole window under ATM so that every microbatch gradient in the window is taken at the same point (§4.4).

5 Analysis

Our analysis formalizes four properties. Appendix H checks the bounds empirically (i.e., the bound level, batch-size scaling, estimator margin, and summary). Appendix L contains the proofs. First, the correction in Eq. (2) preserves the weighted mean and reduces disagreement on every selected coordinate. Second, a velocity estimator that fails to subtract its own assist is stable only for $\lambda < 1/R$. Exact echo cancellation removes this restriction. Third, sparse

delayed averaging has consensus error controlled by the coordinate refresh horizon R/p, matching dense periodic averaging at the corresponding horizon up to delay terms. Fourth, under the usual smoothness and bounded variance assumptions, a hard-clipped additive PACE variant retains the standard non-convex decentralized-SGD rate, with additional terms for consensus error and clipping. In our setting, $q_t = \lambda_t M_t \min\{1, \kappa_t \eta G / \|\lambda_t M_t\|\}$, so $\|q_t\| \le \kappa_t \eta G$. ATM exactly equalizes sample counts apart from integer rounding; its staleness result is proved in a bounded-delay model.

For completeness, the main statements are summarized below. Let $\bar{\theta}_t = N^{-1} \sum_r \theta_{r,t}$ and $\Xi_t = N^{-1} \sum_r \mathbb{E}\|\theta_{r,t} - \bar{\theta}_t\|^2$.

Lemma 1 (Mean preservation and contraction)**.** *For any participating set with normalized non-negative weights, the correction in Eq. (2) leaves the weighted mean unchanged. On a selected coordinate it removes the snapshot disagreement exactly, i.e., only the disagreement accumulated during the delay remains.*

Lemma 2 (Echo stability)**.** *Within the fractional-deduction family, an undeducted estimator is stable only when $\lambda < 1/R$. Exact deduction is stable for any finite coupling when $0 \le \mu < 1$ and the external velocity has bounded second moment: with $M_{w+1} = a_\gamma M_w + (1-\mu)u_w$ and $a_\gamma = \mu + (1-\mu)(1-\gamma)\lambda R$, deduction $\gamma = 1$ gives $a_\gamma = \mu < 1$, while $\gamma = 0$ requires $\lambda < 1/R$.*

Proposition 1 (Consensus error)**.** *Under bounded stochastic gradients, sparse delayed averaging keeps the time-averaged disagreement finite, with its leading refresh term proportional to the expected coordinate horizon R/p and an additive delay term: $\Xi_t \le C\eta^2\left(\frac{\sigma^2}{B}\frac{R}{p} + G^2 D^2\right)$.*

Theorem 1 (Convergence of the analyzed variant)**.** *With smooth objective, unbiased bounded-variance gradients, diminishing step size, and a hard-clipped additive assist,* PACE*-SGD reaches the usual $O(T^{-1/2})$ average-gradient rate up to the consensus and clipping terms.*

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\|\nabla f(\bar{\theta}_t)\|^2 \le C\left[\frac{f(\bar{\theta}_0) - f^\star}{\eta T} + \frac{L\eta\sigma^2}{NB} + L^2\bar{\Xi} + G^2\bar{\kappa}_T^2\right], \quad \bar{\kappa}_T^2 = \frac{1}{T}\sum_{t=1}^{T}\kappa_t^2. \tag{9}$$

Proposition 2 (ATM)**.** *ATM gives every optimizer step the same sample count, up to integer rounding. In the bounded-delay model, the resulting cross-worker disparity approaches a finite level proportional to the largest delay. The update variance drops from $\mathrm{Var}_{\mathrm{naive}} \le \frac{\sigma^2}{B}\sum_r w_r^2/\rho_r$ to $\mathrm{Var}_{\mathrm{ATM}} \le \frac{\sigma^2}{B}\sum_r w_r^2$.*

6 Experiments

6.1 Setup

We train LLaMA-3-style models (Grattafiori et al., 2024) at 226M, 500M, and 1B parameters. The main 500M comparison uses eight replicas, batch 1,024, AdamW, and 30,000 steps, with the model, data order, schedule, validation set, and token budget matched. We report held-out loss on C4 (Raffel et al., 2020) and FineWeb-Edu (Penedo et al., 2024). PACE uses $p = 0.05$, $R = D = 5$, $\mu = 0.5$, and $(\lambda_{\mathrm{hi}}, \lambda_{\mathrm{lo}}) = (4, 1)$, with W equal to warmup; this gives one-percent payload per step and $\mu_{\mathrm{eff}} \in [2.5, 10]$. The communication-efficient baselines use the same amortized budget: about one dense bf16 model-equivalent per 100 optimizer steps. Results average the final eight validations.

We test equal speed (HOMOGENEOUS), two heterogeneous fleets (MIXED-SPEED and ONE-FAST), stale rejoin (OUTAGE), and churn with equal or mixed speeds (SESSION and SESS-HET) using independent Markov processes (i.e., workers keep dropping and joining following a Markov process). Baselines use their original heterogeneity treatment; PACE uses ATM. Appendices A.1 and A.2 give the protocol, and Appendix A shows extra results.

6.2 Main comparison

Table 2 gives the 500M comparison. We compare against the baselines DDP (Li et al., 2020a), SparseLoCo (Sarfi et al., 2025), DiLoCo (Douillard et al., 2023), its asynchronous variant (Liu et al., 2024), NoLoCo (Kolehmainen et al., 2025), Streaming DiLoCo (Douillard et al., 2025), AD-PSGD (Lian et al., 2018), Moshpit SGD (Ryabinin et al., 2021), MT-DAO (Iacob et al., 2026), and LoRDO (Jovanović et al., 2026). Our async-SPARTA ablation uses SPARTA's sparse state

exchange (Beton et al., 2025) but removes ATM and consensus velocity. Figure 4 shows the static regimes. Full curves are in Appendix B.

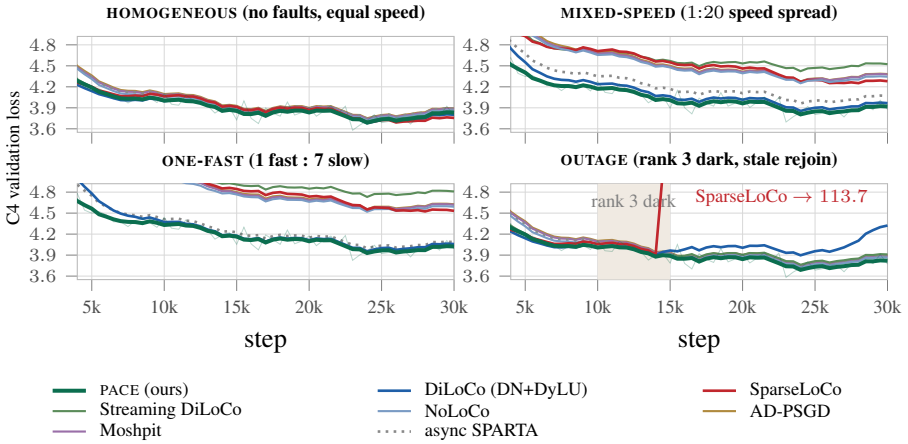


Figure 4: 500M C4 validation curves. SparseLoCo leads on homogeneous hardware; PACE leads both heterogeneous regimes and survives the outage. Thin traces are raw evaluations; heavy lines are smoothed.

Table 2: Complete 500M validation matrix; lower is better. Values are means over the final eight evaluations. Bold marks the best communication-efficient method, italic marks DDP, and red marks collapse. Async DiLoCo uses delayed Nesterov and dynamic local updates; DDP cannot complete missing-worker rows.

	ours	sync. ref.	communication-efficient baselines								ablation
Setting	PACE	DDP†	SparseLoCo	async DiLoCo	NoLoCo	Streaming DiLoCo	AD-PSGD	Moshpit	MT-DAO	LoRDO	async SPARTA
C4 held-out											
HOMOGENEOUS	3.662	*3.491*	**3.595**	3.618	3.688	3.667	3.731	3.721	3.837	3.700	3.731
MIXED-SPEED	**3.786**	*3.692*	4.138	3.811	4.195	4.393	4.240	4.224	4.222	3.855	3.826
ONE-FAST	**3.876**	*3.787*	4.412	3.899	4.459	4.686	4.508	4.492	4.361	3.926	3.883
OUTAGE	**3.644**	*n/a*‡	**div.**	3.877	3.691	3.769	3.733	3.723	3.833	3.701	3.731
SESSION	**3.642**	*n/a*‡	**div.**	4.009	3.692	3.717	3.731	3.721	3.813	3.701	3.726
SESS-HET	**3.755**	*n/a*‡	**div.**	3.981	4.199	4.544	4.237	4.226	4.215	3.853	3.804
FineWeb-Edu held-out											
HOMOGENEOUS	3.030	*2.885*	**2.962**	2.988	3.058	3.035	3.101	3.091	3.237	3.099	3.105
MIXED-SPEED	**3.161**	*3.073*	3.486	3.179	3.553	3.760	3.601	3.584	3.627	3.256	3.205
ONE-FAST	**3.248**	*3.162*	3.772	3.271	3.833	4.085	3.885	3.867	3.760	3.326	3.256
OUTAGE	**3.014**	*n/a*‡	**div.**	3.216	3.062	3.100	3.103	3.093	3.239	3.100	3.103
SESSION	**3.011**	*n/a*‡	**div.**	3.354	3.061	3.078	3.102	3.092	3.219	3.100	3.099
SESS-HET	**3.131**	*n/a*‡	**div.**	3.342	3.559	3.917	3.601	3.587	3.616	3.256	3.179

We highlight three points. SparseLoCo is strongest in the reliable homogeneous case, where barriers are cheap and ATM has nothing to correct. PACE leads every heterogeneous row; async SPARTA is close but consistently worse, isolating the gain from consensus velocity and ATM. Under outage or churn, DDP blocks and SparseLoCo loses its shared anchor and collapses, while state-mixing methods are more robust to them.

6.3 Ablations

Table 3 shows ablations. Note that removing echo cancellation diverges. Norm preservation gives a smaller gain. The effectiveness of ATM is ablated in Table 15. Full ablations cover shared state during outage (Appendix C.1), assist magnitude (Appendix C.2), and coupling stability (Appendix C.3). Six trustless weighting rules are compared in Appendix J. The 1B decomposition is in Appendix M.

Table 3: Selected 226M mechanism ablations. Mixed-speed C4 loss; red marks divergence.

Configuration	C4 loss	Change
PACE	**4.460**	–
No echo cancellation	**diverged**	–
No coupling decay ($\lambda = 4$)	6.490	+2.030
No consensus velocity	4.599	+0.140
No norm preservation	4.481	+0.021

6.4 FAULTS, DEPLOYMENT, AND SCALE

Geo-distributed deployment. Figure 1 shows the eight-region 500M run. Across three GPU types and 22–308 ms RTTs, PACE reaches 3.478 loss in 17.4 hours with only 0.9% blocking. Async DiLoCo needs 27.8 hours and blocks for 37.7%; SparseLoCo needs 36 hours and blocks for 56%; DDP times out before one step. Full network measurements are in Appendix A.4.

During the outage, the stale worker's disparity spikes on return and contracts over subsequent exchanges without reconfiguration (Figure 5). Under repeated node drops, PACE also leads LoRDO, the strongest baseline, in both SESSION and SESS-HET. SparseLoCo collapses on both validation sets.

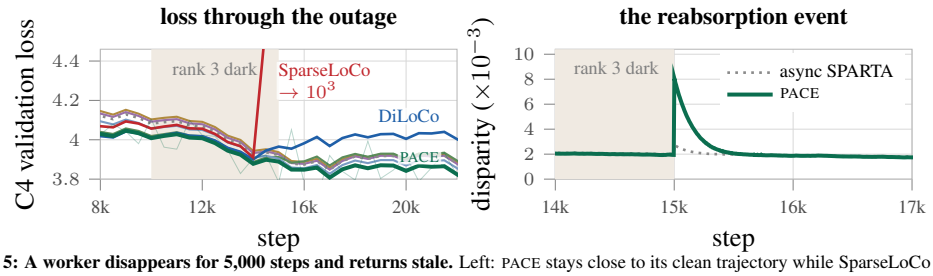


Figure 5: A worker disappears for 5,000 steps and returns stale. Left: PACE stays close to its clean trajectory while SparseLoCo diverges. Right: the returning worker causes a short disparity spike; repeated state averaging removes it over roughly 400 steps without an explicit recovery procedure (the rank is dark until step 15k, shaded).

At 500M, all communication-efficient methods receive the same amortized payload budget (Table 4). PACE sends 23.7 MB per exchange, or 4.74 MB per optimizer step; consensus velocity adds no extra communication.

Table 4: Matched amortized communication budget at 500M. One model-equivalent is one dense bf16 model vector; every communication-efficient method receives about one per 100 steps, and DDP is the unmatched synchronous reference. SparseLoCo's packet budget counts its transmitted indices.

Method	Payload	Every	Model-eq. /100 steps	Overlap
DDP	dense gradient	1 step	100	no
DiLoCo	dense change	100 steps	≈ 1	no
Streaming DiLoCo	1/4 dense change	25 steps	≈ 1	yes
SparseLoCo	compressed packet	5 steps	≈ 1	no
async SPARTA	5% sparse state	5 steps	1	yes
PACE	5% sparse state	5 steps	1	yes

The 226M replication is in Appendix I. At 1B, Table 5 shows the selected results; the full six-benchmark evaluation is in Appendix A.6. The zero-shot benchmarks are HellaSwag (Zellers et al., 2019), ARC-Easy and ARC-Challenge (Clark et al., 2018), PIQA (Bisk et al., 2020), WinoGrande (Sakaguchi et al., 2020), and LAMBADA (Paperno et al., 2016). PACE has the lowest held-out loss and the numerically best result on every reported task. Async SPARTA, the sparse-exchange baseline without consensus momentum, trails PACE by 0.16 (C4) and 0.25 (FineWeb-Edu) nats in MIXED-SPEED and by 0.19 and 0.28 in OUTAGE. Appendix A.5 studies worker-count scaling and shows that PACE leads at 16 and 32 workers.

Table 5: Selected 1B results. Loss and perplexity are lower better; accuracy is higher better. Red marks divergence after stale rejoin.

	MIXED-SPEED				OUTAGE			
Metric	PACE	DiLoCo	SparseLoCo	async SPARTA	PACE	DiLoCo	SparseLoCo	async SPARTA
C4 loss	**3.434**	3.487	3.569	3.589	**3.311**	3.407	diverged	3.502
FineWeb-Edu loss	**2.823**	2.870	3.000	3.070	**2.702**	2.815	diverged	2.985
HellaSwag accuracy	**0.387**	0.371	0.384	0.298	**0.428**	0.424	diverged	0.303
PIQA accuracy	**0.696**	0.673	0.657	0.632	**0.694**	0.682	diverged	0.647
LAMBADA accuracy	**0.295**	0.271	0.282	0.214	**0.344**	0.320	diverged	0.249
LAMBADA perplexity	**55.0**	69.3	107.7	155.4	**36.3**	45.6	diverged	104.3

7 CONCLUSION

Sparse asynchronous parameter averaging is robust in imperfect fleets, but its optimization quality is subpar. PACE recovers the fleet-level motion hidden in each sparse average and uses it as shared momentum, while ATM turns hardware heterogeneity into staleness that ordinary state mixing can recover. This mechanism adds no extra communication. The proposed method is consistently strong across various settings, including heterogeneous, decentralized, and failure-prone fleets.

8 AI USE STATEMENT

The authors used AI assistants to inspect the implementation and recorded results, to locate related literature, to help formulate and verify mathematical arguments, and to prepare the manuscript and its LaTeX source. Responsibility for the correctness, attribution, and integrity of the work rests entirely with the authors.

REFERENCES

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. URL https://arxiv.org/abs/2601.22442.

Matt Beton, Seth Howes, Alex Cheema, Mohamed Bajoumy, and Matt Reed. Improving the efficiency of distributed training using sparse parameter averaging. In *ICLR Workshop on Modularity for Collaborative, Decentralized, and Continual Deep Learning*, 2025. URL https://openreview.net/forum?id=stFPf3gzq1.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239. URL https://ojs.aaai.org/index.php/AAAI/article/view/6239.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

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. URL https://arxiv.org/abs/2311.08105.

Arthur Douillard, Yanislav Donchev, Keith Rush, Satyen Kale, Zachary Charles, Zachary Garrett, Gabriel Teston, Dave Lacey, Ross McIlroy, Jiajun Shen, Alexandre Ramé, Arthur Szlam, Marc'Aurelio Ranzato, and Paul Barham. Streaming DiLoCo with overlapping communication. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=yYk3zK0X6Q.

Louis Fournier, Adel Nabli, Masih Aminbeidokhti, Marco Pedersoli, Eugene Belilovsky, and Edouard Oyallon. WASH: Train your ensemble with communication-efficient weight shuffling, then average. In *NeurIPS 2024 Workshop on Optimization for Machine Learning*, 2024. URL https://arxiv.org/abs/2405.17517.

Aaron Grattafiori, Abhimanyu Dubey, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Alex Iacob, Andrej Jovanović, Mher Safaryan, Meghdad Kurmanji, Lorenzo Sani, Samuel Horváth, William F. Shen, Xinchi Qiu, and Nicholas D. Lane. MT-DAO: Multi-timescale distributed adaptive optimizers with local updates. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=5yPP238v4c.

Andrej Jovanović, 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 *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.04396.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5132–5143, 2020. URL https://proceedings.mlr.press/v119/karimireddy20a.html.

Jari Kolehmainen, Nikolay Blagoev, Semih Kara, John Donaghy, Christopher Nies, and Oğuzhan Ersoy. NoLoCo: No-all-reduce low communication training method for large models. *arXiv preprint arXiv:2506.10911*, 2025. URL https://arxiv.org/abs/2506.10911.

Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian Stich. A unified theory of decentralized SGD with changing topology and local updates. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5381–5393, 2020. URL https://proceedings.mlr.press/v119/koloskova20a.html.

Chamin Hewa Koneputugodage, Thalaiyasingam Ajanthan, Sameera Ramasinghe, Hadi Mohaghegh Dolatabadi, Shamane Siriwardhana, Gil Avraham, Violetta Shevchenko, Karol Pajak, James Snewin, and Alexander Long. Factored gossip DiLoCo: Reducing blocking communication in DiLoCo. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2606.22768.

Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. PyTorch distributed: Experiences on accelerating data parallel training. *Proceedings of the VLDB Endowment*, 13(12):3005–3018, 2020a. doi: 10.14778/3415478.3415530. URL https://www.vldb.org/pvldb/vol13/p3005-li.pdf.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, volume 2, 2020b. URL https://proceedings.mlsys.org/paper_files/paper/2020/hash/1f5fe83998a09396ebe6477d9475ba0c-Abstract.html.

Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. Asynchronous decentralized parallel stochastic gradient descent. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3043–3052, 2018. URL https://proceedings.mlr.press/v80/lian18a.html.

Bo Liu, Rachita Chhaparia, Arthur Douillard, Satyen Kale, Andrei A. Rusu, Jiajun Shen, Arthur Szlam, and Marc'Aurelio Ranzato. Asynchronous local-SGD training for language modeling. *arXiv preprint arXiv:2401.09135*, 2024. URL https://arxiv.org/abs/2401.09135.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Yassine Maziane, Ammar Mahran, Artavazd Maranjyan, and Peter Richtárik. LOSCAR-SGD: Local SGD with communication-computation overlap and delay-corrected sparse model averaging. *arXiv preprint arXiv:2605.20866*, 2026. URL https://arxiv.org/abs/2605.20866.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282, 2017. URL https://proceedings.mlr.press/v54/mcmahan17a.html.

Yurii E. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1525–1534, 2016. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144/.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineWeb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, volume 37, 2024. doi: 10.52202/079017-0970. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/370df50ccfdf8bde18f8f9c2d9151bda-Abstract-Datasets_and_Benchmarks_Track.html.

Boris T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964. doi: 10.1016/0041-5553(64)90137-5.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL https://www.jmlr.org/papers/v21/20-074.html.

Max Ryabinin, Eduard Gorbunov, Vsevolod Plokhotnyuk, and Gennady Pekhimenko. Moshpit SGD: Communication-efficient decentralized training on heterogeneous unreliable devices. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/97275a23ca44226c9964043c8462be96-Abstract.html.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399. URL https://ojs.aaai.org/index.php/AAAI/article/view/6399.

Amir Sarfi, Benjamin Thérien, Joel Lidin, and Eugene Belilovsky. Overcoming the communication-performance tradeoff in LLM pretraining. *arXiv preprint arXiv:2508.15706*, 2025. URL https://arxiv.org/abs/2508.15706.

Sebastian U. Stich. Local SGD converges fast and communicates little. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1g2JnRcFX.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

A ADDITIONAL EXPERIMENTAL DETAILS

A.1 EXPERIMENTAL PROTOCOL

The 226M and 500M models are decoder-only, LLaMA-3-style transformers with a sequence length of 2,048. At each scale, all methods use the same initialization, token order, global batch, AdamW schedule, and validation batches. The 500M runs contain 30,000 optimizer steps, corresponding to approximately 10.3 billion tokens. We use a peak learning rate of 5.0×10^{-4}, 4,000 warmup steps, and linear decay. Unless stated otherwise, each run uses eight replicas.

We average validation loss over the final eight evaluations. Typical standard errors are 0.017–0.028 on C4 and 0.011–0.016 on FineWeb-Edu; consequently, we do not interpret an isolated difference below approximately 0.03. Both validation sets give the same ordering in every regime.

We consider six regimes. All replicas have equal throughput in HOMOGENEOUS, while MIXED-SPEED uses a graded 1:20 speed ratio. In ONE-FAST, one replica runs at full speed and the other seven are slow. In OUTAGE, replica 3 is absent from steps 10,000 to 15,000 and returns with its stale parameters and optimizer state, without a checkpoint reload or membership reconfiguration. In SESSION, each replica follows a two-state Markov chain over exchange events. The stationary absence probability is 0.05, the mean absence lasts approximately 500 steps, and at most two replicas are absent simultaneously. We match absence durations in physical steps for methods with different exchange periods. SESS-HET applies this process to the MIXED-SPEED speed ratios.

A.2 ADDITIONAL 500M REPORTING DETAILS

Table 2 reports the complete C4 and FineWeb-Edu results. DDP is a fully blocking reference. SparseLoCo and DiLoCo require external coordination when workers are missing, so their outage and churn runs use the simulator's absent-contributor semantics. SparseLoCo nevertheless diverges or collapses in each of these settings.

Figure 5 in the main paper depicts the stale worker's return. The left panel shows the loss trajectory, while the right panel shows its cross-replica disparity contracting after rejoin.

A.3 COMMUNICATION BUDGET

The controlled comparison matches the communication payload across methods. We define one model-equivalent as one dense bf16 model vector. DiLoCo sends one such vector every 100 steps, Streaming DiLoCo sends one quarter every 25 steps, and PACE and async SPARTA send five percent every five steps. SparseLoCo's 1.27% top-k, 2-bit packet was sized to the same amortized budget after including the transmitted indices. Thus every communication-efficient row in Table 4 uses approximately one model-equivalent every 100 optimizer steps. DDP is the unmatched synchronous reference.

For PACE, instrumentation gives 23.7 MB per exchange, or 4.74 MB per optimizer step. An eight-worker ring moves 41.5 MB per worker for that event, and consensus velocity adds no payload. We do not use an entropy-coded index floor for SparseLoCo because it would not match the transport used for the results in Table 2. The realized network traffic also depends on whether worker-specific supports are gathered, densified, or handled by a parameter server. We therefore report the matched experimental budget rather than a transport lower bound.

A.4 DEPLOYMENT ON A REAL WAN: EIGHT AWS REGIONS

We also ran the 500M model on eight single-GPU EC2 instances in Tokyo, Stockholm, Montreal, Oregon, Frankfurt, Mumbai, São Paulo, and Sydney (Figure 1). Communication runs over the public Internet through a full-mesh WireGuard overlay with MTU 1380. We use `gloo` on CPU tensors because NCCL over TCP was unreliable at these round-trip times and the sparse exchange already resides in host memory.

Hardware speeds. We measure each node in isolation and normalize its throughput by that of the fastest node to obtain ρ_r (Table 6). The four A10G nodes agree within 0.16%, and the three L4

nodes agree within 0.94%, despite running in different regions. The larger Mumbai instance also matches the other A10G nodes within 0.1%. These ratios primarily capture GPU throughput, rather than network location or CPU provisioning. On this workload, the L4 is slower than the A10G, and the measured fleet spread is 3.06:1.

Table 6: Isolated throughput of the WAN workers. Throughput is the median of the final five of fifteen steps. The resulting capacity ratios are used directly by weighted averaging and ATM. The fleet spans 3.06:1, compared with 20:1 in MIXED-SPEED.

region	instance	GPU	tok/s	ρ_r
Tokyo (ap-northeast-1)	`g6e.xlarge`	L40S 48GB	23,233	**1.000**
Oregon (us-west-2)	`g5.xlarge`	A10G 24GB	8,964	0.386
Frankfurt (eu-central-1)	`g5.xlarge`	A10G 24GB	8,961	0.386
Mumbai (ap-south-1)	`g5.2xlarge`	A10G 24GB	8,957	0.386
Sydney (ap-southeast-2)	`g5.xlarge`	A10G 24GB	8,950	0.385
São Paulo (sa-east-1)	`g6.xlarge`	L4 24GB	7,657	0.330
Montreal (ca-central-1)	`g6.xlarge`	L4 24GB	7,626	0.328
Stockholm (eu-north-1)	`g6.xlarge`	L4 24GB	7,586	0.327

Network latency. The median round-trip time is 153 ms over the 28 undirected links (Table 7). The fastest pair, Stockholm–Frankfurt, is 22 ms; the slowest, São Paulo–Sydney, is 308 ms. None of the 56 ordered paths lost a packet. Measurements during training match the idle baseline, indicating that the sparse traffic does not saturate the links.

Table 7: Pairwise RTT (ms) over the WireGuard mesh, mean of ten ICMP probes per ordered pair; 0% loss on all 56 paths.

	Tokyo	Stkhlm	Montrl	Oregon	Frnkft	Mumbai	SaoPlo	Sydney
Tokyo	—	244	142	99	236	125	259	103
Stkhlm	245	—	105	153	22	147	220	267
Montrl	141	105	—	61	92	186	123	196
Oregon	99	153	60	—	143	230	173	141
Frnkft	236	22	92	142	—	126	201	248
Mumbai	125	147	187	230	126	—	294	152
SaoPlo	259	220	123	190	200	294	—	308
Sydney	102	267	197	140	250	153	308	—

Synchronous DDP. The 500M model requires a 1.96 GB gradient all-reduce on every DDP step. No step finished within the 900 s collective timeout, so the watchdog stopped the run before the first optimizer update. In contrast, each sparse exchange is 49 MB and runs in the background.

Overlapping the sparse exchange. We use $R{=}D{=}5$, matching the controlled experiments. Across 44 consecutive step intervals, the mean step time is 6.91 s against a roughly 7 s compute floor. Applying a completed exchange is no slower than an ordinary step (6.33 s versus 7.06 s). The 49 MB transfer therefore fits within the five-step window, even on the slowest path of 38 Mbps at 308 ms RTT.

The bulk exchange uses a separate `c10d` process group. Sharing a process group serializes the smaller per-step collectives and reduces throughput from 4,615 to 955 tokens/s per replica. We also do not use a collective vote to decide when to apply an exchange. A four-byte vote on every step increases the mean step time from 6.91 s to 16.40 s because it queues behind the bulk traffic. Since the period is fixed, each replica can compute the apply step from the launch step without additional communication.

Offline validation. We disable validation during WAN training because one 20-batch evaluation takes 11 m38 s and transfers 4.5 GB per worker. Each replica instead saves its checkpoint, and we evaluate the uniform average $\bar{\theta} = \frac{1}{8}\sum_r \theta_r$ offline. The FineWeb-Edu protocol matches the main experiments, and both methods use the same batch stream.

End-to-end timing. Table 8 uses the step timestamps from each run. Although PACE exchanges 20 times as often as DiLoCo, it spends only 0.9% of its time blocked, compared with 37.7% for DiLoCo. The 10k-step run takes 17.4 h with PACE and 27.8 h with DiLoCo.

Table 8: WAN exchange cost from step timestamps. The 10k-step runs count an interval as blocked when it exceeds four times that run's median step.

method	period	median step	% wall blocked	blocked / exchange	wall-clock
PACE	R=5	6.0 s	**0.9%**	0.3 s	**17.4 h**
DiLoCo	H=100	5.0 s	37.7%	381 s	27.8 h

Baseline configuration. The WAN DiLoCo run uses DyLU, which scales each replica's inner-step count by its measured speed. With a common inner budget, every outer round would wait for the slowest node and the fastest node would idle for roughly 67% of the round. Batch scaling diverges under heterogeneity (Appendix I). We also provide DiLoCo with the coordinator state normally held by its parameter server, so this is its strongest practical configuration. PACE uses no coordinator.

A.5 SCALING THE REPLICA COUNT: $N = 8 \to 16 \to 32$

We repeat the 500M MIXED-SPEED experiment with 16 and 32 replicas. The model, seed, 30k-step horizon, global batch of 320, learning-rate schedule, exchange settings, and validation protocol remain fixed. Thus, increasing N reduces the batch available to each replica.

The accumulation depth changes the realized speed ratio. A batch-cutoff method uses $\lceil \rho A \rceil / A$, where $A = 320/(Nb_{\text{local}})$. For $N = 8$ and $N = 16$, $A = 20$ and the slowest replicas still realize the intended 1:20 spread. At $N = 32$, $A = 10$ and the smallest possible ratio is 0.1, so the profile becomes $[1, 0.5, 0.3, 0.2, 0.1\times4]$, or 1:10. We use this same realized profile for all three methods at $N = 32$, including DyLU, and keep their token budgets equal. The comparisons within each column are matched. Across columns, however, the severity of heterogeneity also changes between $N = 8$ and $N = 32$.

Table 9 reports the mean and standard error of the final eight evaluations.

Table 9: Replica-count scaling at fixed global batch. Values are the mean of the final eight evaluations ± standard error. The 8- and 16-replica runs use 1:20 heterogeneity; the 32-replica runs share a realized 1:10 profile because of the accumulation-depth floor.

method	N=8 (1:20)	N=16 (1:20)	N=32 (1:10)
(a) C4 held-out			
PACE	**3.786** ± 0.024	**3.992** ± 0.024	**4.200** ± 0.024
DiLoCo (DN+DyLU)	3.811 ± 0.024	4.061 ± 0.024	4.287 ± 0.023
SparseLoCo	4.138 ± 0.018	4.514 ± 0.018	4.946 ± 0.020
(b) FineWeb-Edu held-out			
PACE	**3.161** ± 0.011	**3.349** ± 0.009	**3.555** ± 0.006
DiLoCo (DN+DyLU)	3.179 ± 0.012	3.418 ± 0.009	3.627 ± 0.006
SparseLoCo	3.486 ± 0.015	3.863 ± 0.021	4.330 ± 0.026

SparseLoCo. PACE performs better in every column. Its FineWeb-Edu margin over SparseLoCo is 0.325 ± 0.019 at $N = 8$, 0.514 ± 0.023 at $N = 16$, and 0.775 ± 0.027 at $N = 32$. The corresponding C4 margins are 0.352 ± 0.030, 0.522 ± 0.030, and 0.746 ± 0.031. From $N = 8$ to $N = 32$, SparseLoCo's FineWeb-Edu loss increases by 0.844, compared with 0.394 for PACE; the difference between those increases is 0.450 ± 0.033.

DiLoCo. PACE also performs better than DiLoCo at all three replica counts. The FineWeb-Edu margins are 0.018 ± 0.016, 0.069 ± 0.013, and 0.072 ± 0.008 for $N = 8$, 16, and 32. On C4, the

Table 10: One-billion-parameter held-out loss and zero-shot results. Accuracy is higher better; loss and perplexity are lower better. Bold marks the best result in each regime.

		MIXED-SPEED			OUTAGE		
Task	Chance	PACE	DiLoCo	SparseLoCo	PACE	DiLoCo	SparseLoCo
FineWeb-Edu loss	–	**2.823**	2.870	3.000	**2.702**	2.815	**diverged**
C4 loss	–	**3.434**	3.487	3.569	**3.311**	3.407	**diverged**
HellaSwag	0.25	**0.387**	0.371	0.384	**0.428**	0.424	–
ARC-Easy	0.25	**0.580**	0.577	0.561	**0.619**	0.598	–
ARC-Challenge	0.25	**0.296**	0.281	0.281	**0.313**	0.305	–
PIQA	0.50	**0.696**	0.673	0.657	**0.694**	0.682	–
WinoGrande	0.50	**0.512**	0.496	0.511	**0.559**	0.534	–
LAMBADA accuracy	–	**0.295**	0.271	0.282	**0.344**	0.320	–
LAMBADA perplexity	–	**55.0**	69.3	107.7	**36.3**	45.6	–

margins are 0.025 ± 0.034, 0.069 ± 0.034, and 0.087 ± 0.033. The advantage is small at $N = 8$ but is larger at both $N = 16$ and $N = 32$; there is no reversal in either validation set.

All methods use the same realized speed profile within a column. Across columns, the per-replica batch shrinks as N grows, and the $N = 32$ runs use a milder 1:10 profile instead of 1:20. We therefore report these as two additional replica-count measurements rather than as a scaling law. The shared mask still keeps PACE's communication volume independent of N.

A.6 ONE-BILLION-PARAMETER COMPARISON

At 1B parameters, we compare PACE, DiLoCo, and SparseLoCo in MIXED-SPEED and OUTAGE (Table 10). SparseLoCo diverges shortly after the stale worker returns and therefore has no usable outage checkpoint. Most accuracy differences are within sampling error; only the two LAMBADA metrics exceed 2.5 standard errors.

B FULL VALIDATION CURVES FOR EVERY 500M EXPERIMENT

Table 2 reports the final loss from each 30k-step run. Figure 6 shows the corresponding validation curves for the four operating regimes, nine methods, and both held-out sets.

Every 500 steps, we evaluate the uniform average of a method's replicas. Each evaluation covers about 82k tokens and is subject to sampling noise; the plots therefore show a centred five-point moving average. The faint line shows the raw PACE trace. We omit the first 4,000 warmup steps to keep the later differences visible. DDP is a synchronous reference and has no outage curve.

Homogeneous workers. The ranking stabilizes after about step 10k. DDP leads, followed by SparseLoCo, DiLoCo, and PACE. The gap between PACE and async SPARTA appears by step 6k and remains approximately constant thereafter.

Unequal worker speeds. In MIXED-SPEED and ONE-FAST, DDP, PACE, DiLoCo, and async SPARTA form the leading group. SparseLoCo, NoLoCo, Streaming DiLoCo, AD-PSGD, and Moshpit are about 0.4–0.7 worse. This separation is visible by step 6k and is larger in ONE-FAST. PACE remains within about 0.09 of DDP throughout training; the final gap is not due to a late recovery.

Outage and rejoin. Rank 3 leaves at step 10k and returns at step 15k. PACE shows little change at either point. Other state-averaging methods dip at rejoin and recover within a few hundred steps. DiLoCo moves to a worse trajectory. SparseLoCo continues to deteriorate after rejoin, peaks near 3.2×10^3, and ends near 113. Appendix C.1 examines the associated anchor failure.

The C4 and FineWeb-Edu panels show the same method ordering, separation, and fault response in all four regimes.

C ADDITIONAL DISCUSSIONS AND ABLATIONS

C.1 SHARED STATE DURING THE OUTAGE

We measure disagreement in the shared method state: $\mathbf{M}$ for PACE and the outer anchor for SparseLoCo. A value of zero means that all replicas hold the same state.

During the outage, disagreement in PACE's buffer rises from 7.7×10^{-3} to a median of 52 and a peak of 182 (Figure 7a), since the absent replica misses updates to $\mathbf{M}$. After it rejoins, the spread falls to 3.5×10^{-5}, below its pre-outage value. Ordinary parameter mixing therefore repairs both the weights and the method state.

SparseLoCo behaves differently. Its anchor spread grows from 0.19 to 2.5×10^{5} and remains near 4.1×10^{5} for the rest of training. The error-feedback residual grows only threefold, so the dominant failure is the persistent disagreement between anchors. The replicas therefore form pseudo-gradients relative to different reference models. After rejoin, SparseLoCo's gradient norm has a median of 122 and frequently exceeds 10^{3}, while PACE and DiLoCo remain between 0.54 and 0.59 (Figure 7b). DiLoCo stays finite but retains about ten times its pre-outage replica disparity.

C.2 ASSIST MAGNITUDE OVER TRAINING

Figure 8 plots $r_t = \|\lambda_t \mathbf{M}\|/\|\delta\|$ in all four regimes. The ratio is U-shaped: it is 1.6–4.6 during warmup, falls to 0.22–0.32 near the middle of training, and rises over the final 2k steps to 0.68 in HOMOGENEOUS and 1.48 in ONE-FAST. After warmup λ_t is fixed at 1. The late rise occurs because the optimizer step decays faster than the assist: between steps 20k–22k and 29.5k–30k, the assist falls to 0.22 of its earlier value while the optimizer step falls to 0.08.

Theorem 1 assumes $\|q_t\| \le \kappa_t \eta G$. The measured κ_t grows by about sevenfold over the final third of training because the local step decays more quickly than the assist. Thus, a guard schedule cannot be inferred from the learning-rate decay alone.

The realized step amplification is 1.007–1.039 late in training, well below the fixed-point value $1 + \lambda_t = 2$. Dispersion falls in every regime and ends between 0.17 and 0.37 $\times$ 10^{-3}, with the lowest final value in the outage run.

C.3 WHY THE COUPLING REMAINS STABLE

Figure 9 provides two empirical checks of the coupling argument. Replica disparity follows the learning rate, and the assist remains aligned with descent in every logged regime.

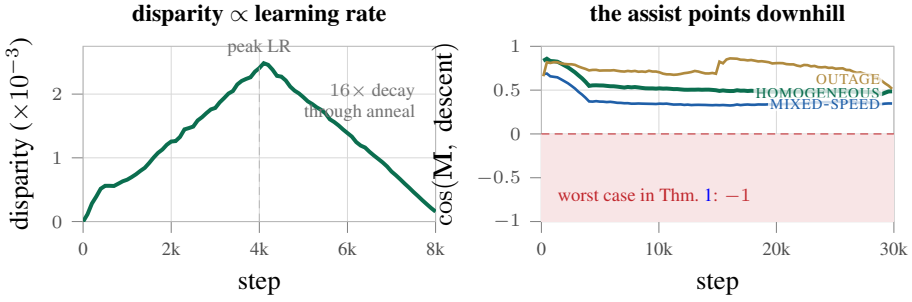


Figure 9: Two empirical checks on the coupling. Left: replica disparity in the 1B mixed-speed run peaks with the learning rate and falls about 16× during annealing. Right: the cosine between the applied assist and the local descent direction remains positive in all three 500M runs. The theorem allows the worst-case value -1, shown by the shaded region, but that case does not occur in these traces.

D ANALYSIS ON SPARSE FRACTION

We use $p = 0.05$ and $R = 5$, giving an expected coordinate refresh horizon of $R/p = 100$ steps. We test this choice by repeating the full 500M MIXED-SPEED run at $p \in \{0.01, 0.10, 0.25, 0.50, 1.0\}$. All other settings, including the seed, data order, R, D, ATM, weighting, and validation batches, are unchanged. Varying p changes both the traffic p/R and the refresh horizon R/p.

Each validation event covers 20 batches, so its absolute loss is noisy. The runs use the same validation examples at each matched step, and their paired difference removes most of this common variation. Table 11 reports paired differences over steps 20k–30k. The two unstable settings are discussed in §D.1.

Table 11: Effect of the sparse fraction. Differences are paired against $p = 0.05$ over 21 validation events from steps 20k–30k. Wire cost is the amortized fraction of model parameters sent per step.

	refresh	wire cost	Δ vs. $p = 0.05$	
sparse fraction	horizon R/p	p/R per step	FineWeb-Edu	C4
$p = 0.01$	500	0.2%	$+0.0554 \pm 0.0033$	$+0.0555 \pm 0.0037$
$p = 0.05$ (ours)	100	1.0%	—	—
$p = 0.10$	50	2.0%	$+0.0069 \pm 0.0004$	$+0.0053 \pm 0.0007$

Loss is U-shaped over the three converged settings. Reducing p to 0.01 saves fivefold communication but adds about 0.055 loss on both validation sets. Increasing p to 0.10 doubles communication and adds 0.0069 ± 0.0004 on FineWeb-Edu and 0.0053 ± 0.0007 on C4. Thus, the best measured value lies inside the tested range rather than at its communication-heavy end.

Figure 10 shows the corresponding method-state statistics. Parameter disparity falls monotonically as p rises, from 9.1×10^{-7} at $p = 0.01$ to 1.8×10^{-7} at $p = 0.10$. The spread of the consensus-velocity buffer does not: it is about 4.0×10^{-4} at $p = 0.05$ and roughly eleven times larger at either neighboring setting. With $p = 0.01$, coordinates are refreshed too slowly and the replicas drift apart. With $p = 0.10$, the shorter-window differences contain more local fluctuation, and the assist norm rises from 0.111 to 0.167. The operating point balances these two effects.

D.1 THE STABILITY BOUNDARY

The sweep becomes unstable between $p = 0.10$ and $p = 0.25$. At $p = 0.25$, loss turns upward at step 170; at $p = 0.50$, it turns at step 88. Both runs then grow geometrically until the parameters overflow. Repeating the runs from the same seed reproduces the onset step, including on a different machine. The applied correction grows from 3.125×10^{-2} to 8 in about 40 steps, while it remains at 3.125×10^{-2} in the convergent runs.

The buffer becomes unstable before the loss. For $p = 0.50$, the cross-rank spread of $\mathbf{M}$ grows from 8.8×10^{-6} to 3.8×10^{-3} by step 60, while the loss is still falling. The loss reaches 133 only 20 steps later. At $p = 0.25$, the spread similarly grows from 4.9×10^{-6} to 1.1×10^{-3} over the first 100 steps while the loss improves from 11.1 to 9.2. This ordering is consistent with the $a_\gamma > 1$ branch of Lemma 2: estimator instability precedes optimization failure.

Table 12 measures all settings before any run has diverged. The assist norm increases from 0.020 at $p = 0.01$ to 4.603 at $p = 0.50$, and the buffer spread rises from 2.4×10^{-7} to 1.8×10^{-5}. Larger values of p move more coordinates at each event, increasing the displacement seen by the estimator and eventually pushing the feedback loop past its stability threshold.

Table 12: Stability across the sparse-fraction sweep. Assist norm and $\mathbf{M}$ spread are medians over steps 20–60, before any arm diverges. The last column reports paired FineWeb-Edu differences from $p = 0.05$ for settings that finish. The $p = 1$ row disables the momentum mechanism and is a control.

p	R/p	assist norm	$\mathbf{M}$ spread	outcome	Δ FineWeb
0.01	500	0.020	2.4×10^{-7}	converges	$+0.055$
0.05	100	0.033	8.9×10^{-7}	converges	**best**
0.10	50	0.439	3.0×10^{-6}	converges	$+0.007$
0.25	20	1.311	7.8×10^{-6}	**diverges** (step 170)	—
0.50	10	4.603	1.8×10^{-5}	**diverges** (step 88)	—
1.00	5	0	0	converges *(momentum off)*	$+0.187$

The $p = 1$ arm is a control, not a large-p PACE run. Selecting every coordinate leaves no subset on which to apply consensus momentum, and both the assist norm and $\mathbf{M}$ spread are exactly zero. This setting measures dense delayed parameter averaging without the proposed momentum. It is stable, but its FineWeb-Edu loss is 0.187 worse than the operating point.

Together, Tables 11 and 12 favor $p = 0.05$ in both quality and stability. The next setting, $p = 0.10$, is 0.007 worse on FineWeb-Edu and has an assist 13 times larger; $p = 0.25$ does not finish. The growth in assist is also consistent with the measured p-dependence in §H.2, which is steeper than the simplified consensus-error bound.

E Analysis on exchange delay

The main experiments use delay $D = 5$ and period $R = 5$. A replica deducts its own contribution from an average that is D steps old. Setting $D = R$ therefore reads each exchange exactly one period after it is written.

We repeat the 500M MIXED-SPEED run at $D \in \{1, 2, 5, 10, 20\}$ with $R = 5$ and $p = 0.05$ held fixed, varying only the staleness of the consensus estimate. All other settings, including the seed, data order, p, ATM, weighting, and validation batches, are unchanged. Holding R fixed while D moves separates staleness from traffic: the wire cost p/R is identical in every row, so any difference is attributable to reading an older average rather than to sending more or less.

As in §D, each validation event covers 20 batches and is noisy in isolation. Table 13 therefore reports differences paired against $D = 5$ over 20 validation events from steps 20k–29.5k.

Table 13: Effect of the exchange delay. Differences are paired against $D = 5$ over 20 paired validation events from steps 20k–29.5k, at fixed $R = 5$ and $p = 0.05$. The wire cost is identical in every row, so the comparison isolates staleness. $D = R$ is the operating point.

delay	exchanges read	Δ vs. $D = 5$	
D	at age (steps)	FineWeb-Edu	C4
$D = 1$	1	$+0.0002 \pm 0.0003$	$+0.0015 \pm 0.0007$
$D = 2$	2	$+0.0011 \pm 0.0002$	$+0.0024 \pm 0.0008$
$D = 5$ (ours)	5	—	—
$D = 10$	10	$+0.0092 \pm 0.0011$	$+0.0105 \pm 0.0011$
$D = 20$	20	*diverged*	

When $D < R$, a replica can deduct a contribution that has not yet entered the average it reads. The resulting echo mismatch has a small cost: on C4, $D = 1$ is $+0.0015 \pm 0.0007$ worse than the operating point, while the FineWeb-Edu difference is not significant. When $D > R$, the momentum estimate uses an older population mean. This is the $O(\eta^2 D^2)$ term in the bound of §H. At $D = 10$, the losses increase by $+0.0105 \pm 0.0011$ on C4 and $+0.0092 \pm 0.0011$ on FineWeb-Edu. The C4

increase is approximately seven times the penalty from moving the same distance below R, and both differences are resolved at about ten standard errors.

The results are nearly flat for $D \leq R$. All settings from $D = 1$ to $D = 5$ are within 0.0025 of the best measured value, and the difference between $D = 1$ and $D = 2$ is not resolved. In contrast, the degradation at $D = 10$ is clear on both validation sets. The data therefore support some flexibility below R, but not above it.

At $D = 20$, which corresponds to four exchange periods of staleness, the run becomes unstable. The training loss passes 100 by step 611, 10^6 by step 881, and reaches order 10^{17} by step 1700 before becoming non-finite, with gradient norms of order 10^{18}. Warmup lasts 4000 steps, so the failure occurs early in the learning-rate warmup. We report a divergence because there is no converged model to evaluate. This single run does not establish that $D = 20$ always diverges. It shows that, by $D/R = 4$, the effect of delay can extend beyond a small loss increase, consistent with the quadratic delay term in §H.

F THE λ SCHEDULE: PEAK, FLOOR, AND WHAT SETS THEM

The coupling λ is initially high and decreases during training. Section 4 motivates this schedule, while §C contains a 226M component ablation in which $\lambda \equiv 4$ increases C4 loss by 2.030. Here, we vary the schedule at fixed p, R, and D.

We compare three alternatives with the reported schedule. Two keep λ constant at either the initial or final value; the third anneals to zero. These runs separate the effect of annealing from those of its endpoints.

Table 14: Effect of the λ schedule. Differences are paired against the reported schedule over 20 paired validation events from steps 20k–29.5k, at fixed $p = 0.05$, $R = D = 5$. "$a \to b$" denotes a linear anneal from a to b over training.

schedule	what it tests	Δ vs. reported	
		FineWeb-Edu	C4
$4 \to 1$ (ours)	—	—	—
$\lambda \equiv 4$	anneal removed, peak kept	$+0.4468 \pm 0.0080$	$+0.4531 \pm 0.0084$
$\lambda \equiv 1$	no boost at all	$+0.0080 \pm 0.0004$	$+0.0077 \pm 0.0008$
$4 \to 0$	floor removed	$+0.0253 \pm 0.0014$	$+0.0243 \pm 0.0016$

Effect of annealing. Holding λ at its peak for the whole run increases loss by $+0.4531 \pm 0.0084$ on C4 and $+0.4468 \pm 0.0080$ on FineWeb-Edu. That is roughly forty times the largest penalty in the delay sweep of §E, and exceeds fifty standard errors on both validation sets. The 226M ablation in §C shows the same trend, with a $+2.030$ increase when the decay is removed. In contrast, using $\lambda \equiv 1$ throughout training increases loss by $+0.0077 \pm 0.0008$ on C4 and $+0.0080 \pm 0.0004$ on FineWeb-Edu. The large effect therefore comes from retaining high coupling late in training, not from removing the initial boost.

Final coupling. Annealing to zero removes the consensus velocity at the end of training, whereas annealing to one retains it without amplification. The zero-floor schedule is worse by $+0.0243 \pm 0.0016$ on C4 and $+0.0253 \pm 0.0014$ on FineWeb-Edu. This difference is resolved at approximately fifteen standard errors, but is nineteen times smaller than the cost of removing the anneal. Thus, reducing λ during training matters more than the precise value of the final coupling.

All four rows in Table 14 are uninterrupted 30k-step runs at 500M and use the same validation batches as the operating point. Data-dependent schedules based on the gradient norm or $1/\eta$ remain possible alternatives.

G ACCUMULATE-TO-MATCH VERSUS NAIVE BATCH CUTOFF

Under naive batch scaling, a slow replica follows the shared optimizer clock but processes a smaller batch. At capacity ρ_r, it uses the last $\rho_r A$ microbatches of an accumulation window of length A and discards the rest. Its updates therefore have roughly $1/\rho_r$ times the gradient variance. ATM instead keeps the number of samples per optimizer step fixed. A replica at capacity ρ_r takes one step every $k_r = \mathrm{round}(1/\rho_r)$ ticks, which converts the capacity difference into staleness that is contracted by sparse mixing.

As discussed in §L, this conversion is not free: slower replicas also contribute updates less often. Table 15 measures the net effect at matched wall-clock time. We cross ATM with capacity-weighted averaging because weighting can also reduce the influence of a noisy replica, but does so by a different mechanism.

Table 15: ATM against naive batch cutoff, crossed with the averaging rule. 226M values are held-out C4 loss after 6,000 steps. 500M values are paired differences against the operating point (ATM with capacity weighting) over 20 validation events from steps 20k–29.5k, so lower is better and the operating point reads 0 by construction. Both are the MIXED-SPEED regime.

batch handling	averaging	226M C4	500M Δ vs. ours	
			FineWeb-Edu	C4
naive cutoff	uniform	4.840	$+0.3611 \pm 0.0036$	$+0.3772 \pm 0.0050$
naive cutoff	capacity-weighted	4.531	$+0.1161 \pm 0.0033$	$+0.1300 \pm 0.0037$
ATM	uniform	4.510	$+0.0097 \pm 0.0013$	$+0.0016 \pm 0.0016$
ATM	capacity-weighted (ours)	**4.458**	—	—

226M results. With uniform averaging, ATM improves C4 loss by 0.330 over naive batch scaling. ATM with uniform averaging (4.510) also outperforms naive batches with capacity weighting (4.531). This comparison is useful because capacity weighting requires declared worker speeds, whereas ATM does not. The mechanisms are complementary: combining them gives the best loss, 4.458.

Section J reports six statistics tested for detecting noisy replicas from their observed dynamics. None improved on uniform averaging. ATM avoids the need for this classification.

500M results. The ordering is the same at 500M. Keeping capacity weighting but replacing ATM with the naive cutoff costs 0.130 ± 0.004 on C4 (0.116 ± 0.003 on FineWeb-Edu). Removing both costs 0.377 ± 0.005 (0.361 ± 0.004). Removing capacity weighting while keeping ATM costs 0.002 ± 0.002 on C4 and 0.010 ± 0.001 on FineWeb-Edu. ATM therefore accounts for almost all of the gap, and capacity weighting matters mainly when batches are cut naively, where it recovers about two thirds of the loss.

Conditions. The 500M column contains three 30k-step MIXED-SPEED runs. They differ from the operating point only in the batch-handling rule, the averaging rule, or both; $p = 0.05$, $R = D = 5$, and the $4 \to 1$ coupling schedule are fixed. The naive-cutoff arm with capacity weighting was run twice from the same seed on different hardware. The table reports the first run; the second gives $+0.112 \pm 0.005$ on C4 ($+0.101 \pm 0.005$ on FineWeb-Edu). The two agree on the ordering and on the size of the effect relative to the other rows, but they differ by more than their paired standard errors, which bounds how finely single-run differences of this size should be read.

H EMPIRICAL CHECKS ON THE ANALYSIS

The 500M runs record consensus error, coupling, and optimizer-step norms. We use these measurements to evaluate Proposition 1 with the explicit constants derived in Appendix L.4:

$$\Xi_t \;\le\; 4\,\eta^2\,\frac{\sigma^2}{B}\,\frac{R}{p} \;+\; 8\,\eta^2 G^2 D^2. \tag{10}$$

We evaluate the level of the bound, its dependence on p and B, and the stability condition in Lemma 2. Table 17 summarizes these checks and lists the quantities that are absent from the current logs.

H.1 The level of the bound at the operating point

Calibration. Consider per-coordinate RMS deviation over the sampled index set. We convert it to the full squared norm as $\Xi_t = d \cdot (\text{rms})^2$ with $d = 490{,}710{,}528$ trainable parameters.

We estimate $\eta^2 G^2$ from the logged squared optimizer step. This avoids using the raw gradient norm, which remains almost constant under adaptive clipping even as the AdamW step decays with the learning rate. Since $\sigma^2/B \le G^2$, (10) is at most $(4 \cdot \frac{R}{p} + 8D^2)\,\mathbb{E}\|\delta\|^2 = 600\,\mathbb{E}\|\delta\|^2$ at the operating point, of which the delay term contributes $8D^2\,\mathbb{E}\|\delta\|^2 = 200\,\mathbb{E}\|\delta\|^2$.

This calibration is one-sided. The proposition assumes a uniform upper bound on $\mathbb{E}\|g_t\|^2$, while the logged mean step can underestimate it. Measurements below the calibrated ceiling are consistent with the bound, but measurements above it would not, on their own, contradict the proposition.

Bound level. Proposition 1 assumes a constant step size, so we use the largest logged step to form one horizontal ceiling. Taking $\eta^2 G^2 = \max_t \mathbb{E}\|\delta_t\|^2 = 5.64$ (attained at the learning-rate peak, step 4k) gives a ceiling of $3.38{\times}10^3$, against a measured peak of $\Xi_t = 2.06{\times}10^3$ at step 4.5k (Figure 11):

$$\max_t \Xi_t \;=\; 0.61 \times \text{bound},$$

and the measured curve stays below it at all 295 sampled steps. The smallest gap is a factor of 1.64. This uses the explicit Young-inequality parameter and the exact $1 - p/2$ contraction from the independent mask draw.

Contribution of the two terms. The delay term is the horizontal line at $200 \max_t \mathbb{E}\|\delta\|^2 = 1.13{\times}10^3$ in Figure 11a. The measured peak is 1.83 times larger, so the refresh-horizon term is necessary at this setting. The independent p-sweep in §H.2 gives the same result.

H.2 The measured p-scaling against Proposition 1

The sparse-fraction sweep changes p while holding η, B, R, D, the seed, and the data stream fixed. It therefore tests the p-dependence of Proposition 1:

$$\Xi_t \;=\; \tfrac{1}{N}\textstyle\sum_r \mathbb{E}\|\boldsymbol{\theta}_{r,t} - \bar{\boldsymbol{\theta}}_t\|^2 \;\le\; C\,\eta^2\Big(\tfrac{\sigma^2}{B}\,\tfrac{R}{p} \;+\; G^2 D^2\Big),$$

The noise term scales as $1/p$ and the delay term does not depend on p. At a matched step, the right-hand side has the form $A/p + B'$ with $A, B' \ge 0$, so a fit $\Xi \propto p^{-x}$ should have $x \in [0, 1]$.

The disparity logged in Figure 10a is proportional to $\sqrt{\Xi_t}$. Because the proportionality constant is shared across all arms, ratios and fitted exponents can be compared without estimating C, σ^2, G, or L. Table 16 reports five training windows.

Table 16: Measured dependence of consensus error on p. Each row compares matched training windows. The bound admits $\Xi \propto p^{-x}$ for $x \in [0, 1]$.

step window	$\Xi(p{=}0.01)/\Xi(p{=}0.10)$	measured x in $\Xi \propto p^{-x}$	bound admits
4k–8k	14.3	1.15	$[0, 1]$
8k–14k	17.9	1.24	$[0, 1]$
14k–20k	21.5	1.32	$[0, 1]$
20k–25k	24.6	1.39	$[0, 1]$
25k–30k	28.0	1.47	$[0, 1]$

Consensus error decreases with p in every window. The difference between $p = 0.01$ and $p = 0.10$ is a factor of 14–28, which rules out the p-independent delay term as the dominant term at the operating point.

The fitted exponent is steeper than the bound predicts: x rises from 1.15 to 1.47 across training. Setting the free constant from the $p = 0.01$ arm still satisfies the upper bound for every setting, but the simple $A/p + B'$ form does not fit the observed ratios tightly. One likely source is the norm-preserving assist, which is not exactly common across replicas and varies with p. Its measured magnitudes are 0.139, 0.111, and 0.167 for $p = 0.01, 0.05, 0.10$. The $p = 0.01$ run may also sit outside the proposition's step-size condition, although the $p = 0.05$ to $p = 0.10$ pair alone still gives $x \approx 1.45$.

Within the $p = 0.05$ run, disparity peaks near the learning-rate peak and then falls by a factor of 10.7, compared with the 100-fold decrease implied by the learning-rate schedule alone. Both comparisons indicate that consensus error does not separate cleanly into a $1/p$ term and a constant floor.

H.3 Scaling with the per-replica batch

At fixed η, R, p, D, (10) predicts $\Xi \propto \sigma^2/B$. The homogeneous runs use $B = 320/N = 40, 20, 10$ at $N = 8, 16, 32$, so this model predicts consensus-error ratios of $1 : 2 : 4$. Measured at matched steps, $\Xi^{(16)}/\Xi^{(8)} = 1.19$–$1.24$ and $\Xi^{(32)}/\Xi^{(8)} = 1.49$–$1.87$ across training (Figure 12b), corresponding to growth between $N^{0.25}$ and $N^{0.45}$ rather than N. The bound therefore predicts the direction of the change but overestimates its magnitude.

The optimizer-step norms explain much of the difference. Their maxima are 5.59, 5.56, and 5.44 across the three runs, within 3% even though B changes fourfold. AdamW normalizes much of the change in gradient magnitude, whereas assumption (A2) models an SGD step with variance proportional to $1/B$. The one-sided calibrated ceiling is exceeded by factors of 1.46–2.25 in these runs. This indicates that the mean-step proxy is below the uniform G in the proposition; it is not a violation of the bound. The fixed-global-batch setup also changes replica count and microbatch size together.

H.4 The estimator lemma's stability margin

Lemma 2 gives stability with exact deduction ($\gamma = 1$), while an estimator without deduction becomes unstable above $\lambda R = 1$. The run uses $\lambda R = 5$–18 after its initial ramp (Figure 13a), giving an undeducted growth factor of 3.0–9.6 per window. The corresponding counterfactual grows by more than 10^6 within 100 steps.

The measured deducted buffer remains bounded for all 6,000 windows. Its norm peaks at 0.445 near step 4.6k and then falls by a factor of 14.5 as the optimizer step anneals (Figure 13b). This behavior is consistent with the stability result. The logs do not contain the true window displacement u_w, so they cannot test unbiasedness directly.

H.5 Summary of empirical checks

Summary of empirical checks on theoretical predictions are shown in Table 17.

I 226M scale replication

Table 18 repeats the comparison with 226M parameters. The runs use eight replicas, a global batch of 320 sequences of length 1,024, the same data order, and 6k training steps, corresponding to approximately 20 tokens per parameter. MIXED-SPEED and MIXED-SPEED$^{\text{hw}}$ share the same 1:20 capacity profile; MIXED-SPEED$^{\text{hw}}$ repeats the experiment on identical hardware. The outage lasts either 200 or 1,000 steps.

The ordering is similar to the 500M results. DiLoCo performs best in HOMOGENEOUS, while PACE performs best in the heterogeneous settings and the 1,000-step outage on both validation sets. SparseLoCo gives the lowest loss for the 200-step outage but degrades under the longer outage, after its error-feedback state has had more time to drift.

Table 17: Empirical checks of the analysis. "Conservative" means that the measured quantity grows more slowly than the bound allows. The final column points to the supporting measurement.

claim	measurement	verdict	where
Prop. 1, level of (10)	Ξ_t vs explicit-constant ceiling, G from realized step (one-sided)	holds at every step; peak ratio 0.61 — tight to $1.6\times$	§H.1
noise term $\propto 1/p$	three full runs, only p varied	sign and dominance confirmed; exponent 1.15–1.47 vs admitted $[0, 1]$ — form not tight	§H.2
noise term $\propto \sigma^2/B$	HOMOGENEOUS N=8/16/32, only B varied	conservative: measured $N^{0.25-0.45}$ vs allowed N; AdamW step-norm invariance identified as cause	§H.3
delay term G^2D^2 subdominant at the operating point	two calibration-independent routes	confirmed: peak $\Xi = 1.83\times$ delay-alone; 14–$28\times$ variation in p	§H.1, H.2
Lem. 2, stability at $\gamma = 1$ for arbitrary λ	run sits 5–$18\times$ past the undeducted threshold	confirmed as a dichotomy: bounded $6{,}000$ windows vs counterfactual $10^6\times$ per 100 steps	§H.4
Lem. 2, unbiasedness $\mathbb{E}\mathbf{M} \to u$	u_w not logged	proxy only: buffer tracks the annealed step ($14.5\times$ decay)	§H.4
Thm. 1, assist charged at $\cos = -1$	measured $\cos(\mathbf{M}$, descent)	deliberately loose: measured 0.62–0.85, never negative	Fig. 9b
Prop. 2(ii), perturbation $\propto \eta$	disparity against the LR schedule	direction confirmed at 1B; decay sub-linear in η at 500M	Fig. 9a, §H.2
Lem. 1	mean-preservation is an identity; residual not separately observable	enters only through the delay term, which both routes find subdominant	§H.1

Table 18: Held-out loss for the 226M runs. Values use the uniformly averaged replicas after 6k steps. MIXED-SPEED$^{\text{hw}}$ repeats the MIXED-SPEED profile on identical hardware. § marks outage runs that grant barriered methods absent-contributor semantics. In the blocking row, A means that progress needs a quorum, B means communication blocks compute, and * means partial overlap.

	ours	communication-efficient baselines						ablation
setting	PACE	SparseLoCo	DiLoCo (DN+DyLU)	NoLoCo	Streaming DiLoCo	AD-PSGD	Moshpit	async SPARTA
blocks on	**neither**	A + B	B	B	A	B*	B*	**neither**
(a) C4 held-out (primary metric)								
HOMOGENEOUS	4.242	—	**4.156**^{H5}	4.352	4.261	4.416	4.512	4.408
MIXED-SPEED	**4.458**	—	4.485	4.951	4.841	4.997	5.021	4.617
MIXED-SPEED$^{\text{hw}}$	**4.460**	4.617	4.627	4.963	4.869	5.050	5.034	4.656
OUTAGE [2k,3k]	**4.258**	6.245§	4.287§	4.367	4.281	4.417	4.405	4.411
(b) FineWeb-Edu held-out (disjoint shards)								
HOMOGENEOUS	3.616	—	**3.526**^{H5}	3.719	3.631	3.783	3.778	3.780
MIXED-SPEED	**3.819**	—	3.840	4.315	4.207	4.375	4.390	3.980
MIXED-SPEED$^{\text{hw}}$	**3.813**	3.942	3.990	4.332	4.234	4.427	4.408	4.012
OUTAGE [2k,3k]	**3.630**	5.755§	3.651§	3.735	3.647	3.790	3.777	3.788

MIXED-SPEED$^{\text{hw}}$, DiLoCo-naive (batch scaling instead of DyLU): **diverged**; loss peaks at 3.4×10^3, still 774 when killed at step 5.6k.

J THE DETECTION LEDGER: SIX TRUSTLESS ESTIMATORS THAT FAIL

Before developing ATM, we evaluated six statistics for detecting and down-weighting noisy replicas. None improved upon uniform averaging, which gives a C4 loss of 4.840 for the 226M MIXED-SPEED run:

estimator	signal	C4
inverse-variance v1	per-coordinate squared deviation	5.010
inverse-variance v2	deviation trend / age	(no-op under bf16)
inverse-variance v2.1	fp32 momentum-trend regression	(trend fraction 0.00)
inverse-variance v3	lag-1 autocorrelation of the lead	5.405 (at 3k)
gradient-variance proxy (v4)	Adam second-moment ratios	5.178 (at 5.8k)
per-rank coupling (λ_r)	per-replica assist scaling	4.858 (at 4.8k)

The statistics either treated fast progress as noise or produced little signal. For example, v1 reduced the fast worker's weight toward zero, the v2.1 trend fraction was identically zero, and the v3 autocorrelation was 0.02. ATM does not require this classification because it equalizes the sample count per optimizer step.

K Staleness fixed point

At averaging-window granularity, let Γ_t be the gap between a slow replica and the fleet average. The fleet moves by Δ between windows, and the correction in (2) contracts the existing gap by $1-\alpha$:

$$\Gamma_{t+1} = (1-\alpha)\,\Gamma_t + \Delta \quad \Longrightarrow \quad \Gamma^\star = \Delta/\alpha,$$

The recursion converges geometrically to the fixed point Δ/α. In the experiments, disparity reaches its plateau within approximately 20 steps of a capacity change. Because Δ scales with the learning rate, this plateau decreases during learning-rate decay; Figure 9 shows a 15-fold reduction. Batch-induced noise is instead added at every step. Weighting reduces it only by reducing the slow replica's contribution, which explains why the estimators in Appendix J do not recover the loss from naive batch scaling.

L Proofs for Section 5

Expectations are over microbatch noise and mask draws. The symbols c, c', C denote absolute constants and may change between lines. Recall $\bar{\boldsymbol{\theta}}_t = \frac{1}{N}\sum_r \boldsymbol{\theta}_{r,t}$ and

$$\Xi_t \;=\; \tfrac{1}{N}\sum_r \mathbb{E}\|\boldsymbol{\theta}_{r,t} - \bar{\boldsymbol{\theta}}_t\|^2,$$

the consensus error. We write $e_{r,t} = \boldsymbol{\theta}_{r,t} - \bar{\boldsymbol{\theta}}_t$, so $\Xi_t = \frac{1}{N}\sum_r \mathbb{E}\|e_{r,t}\|^2$ and $\sum_r e_{r,t} = 0$. A vector is *common* if it is identical on every replica; such vectors cancel from pairwise differences.

L.1 Elementary facts used repeatedly

We will use the following standard facts.

(F1) Young's inequality. For vectors a, b and any $\rho > 0$,

$$2\langle a, b\rangle \;\le\; \rho\|a\|^2 + \rho^{-1}\|b\|^2.$$

Proof. A squared norm is nonnegative, so $0 \le \|\sqrt{\rho}\,a - \rho^{-1/2}b\|^2 = \rho\|a\|^2 - 2\langle a, b\rangle + \rho^{-1}\|b\|^2$; and rearranging gives the result.

(F2) Cauchy–Schwarz for a sum of n terms. For vectors $v_1, \dots, v_n$,

$$\Big\|\sum_{j=1}^n v_j\Big\|^2 \;\le\; n\sum_{j=1}^n \|v_j\|^2.$$

Proof. Expand the left side as $\sum_j\sum_k\langle v_j, v_k\rangle$, bound each inner product by $\frac{1}{2}(\|v_j\|^2 + \|v_k\|^2)$ using (F1) with $\rho = 1$, and count: each $\|v_j\|^2$ appears n times. Equivalently, $\|\frac{1}{n}\sum_j v_j\|^2 \le \frac{1}{n}\sum_j \|v_j\|^2$ — averaging cannot increase a squared norm (Jensen's inequality); we use both forms.

We use this inequality for the D post-snapshot increments. Because these increments are dependent, their cross terms cannot be discarded, resulting in the D^2 delay factor.

(F3) Two-term and parametrized versions of (F2). Taking $n = 2$,

$$\|a+b\|^2 \;\le\; 2\|a\|^2 + 2\|b\|^2,$$

and for any $\alpha > 0$, applying (F1) to the cross term of $\|a+b\|^2 = \|a\|^2 + 2\langle a, b\rangle + \|b\|^2$ with $\rho = \alpha$,

$$\|a+b\|^2 \;\le\; (1+\alpha)\|a\|^2 + \big(1+\alpha^{-1}\big)\|b\|^2.$$

In §L.4, we choose α so that the resulting per-step growth factor is $(1+2\eta L)^2$.

(F4) Descent lemma. If f is L-smooth (A1), then for all x, y,

$$f(y) \;\le\; f(x) + \langle \nabla f(x), y - x\rangle + \tfrac{L}{2}\|y-x\|^2.$$

Proof. Integrate the gradient along the line segment from x to y and apply L-smoothness.

(F5) Variance of a weighted mean. If $\hat{g}_1, \dots, \hat{g}_N$ are conditionally independent and unbiased for $\nabla f_1, \dots, \nabla f_N$ with $\mathrm{Var}(\hat{g}_r) \le s_r^2$, then for weights w with $\sum_r w_r = 1$,

$$\mathbb{E}\Big[\sum_r w_r \hat{g}_r\Big] = \sum_r w_r \nabla f_r, \qquad \mathrm{Var}\Big(\sum_r w_r \hat{g}_r\Big) \;\le\; \sum_r w_r^2 s_r^2.$$

Proof. Linearity of expectation gives the first identity. For the second, the variance of a sum of independent terms is the sum of their variances, and scaling a term by w_r scales its variance by w_r^2. Conditional independence *across replicas* is what supplies the $1/N$ variance reduction; unbiasedness alone does not.

(F6) Fixed point of a contracting linear recursion. If $x_{w+1} \le a\, x_w + b$ with $a \in [0, 1)$, $b \ge 0$ and $x_0 = 0$, then $x_w \le b/(1-a)$ for every w. *Proof.* Unrolling gives $x_w \le a^w x_0 + b(1 + a + \cdots + a^{w-1}) = b\,\frac{1-a^w}{1-a} \le \frac{b}{1-a}$, since $x_0 = 0$ and $a^w \ge 0$. If $a \ge 1$, the same recursion need not have a uniform bound.

(F7) Spread about the mean equals half the mean pairwise spread. For any vectors $x_1, \dots, x_N$ with mean $\bar{x}$,

$$\frac{1}{N}\sum_r \|x_r - \bar{x}\|^2 \;=\; \frac{1}{2N^2}\sum_r\sum_{r'} \|x_r - x_{r'}\|^2.$$

Proof. Expanding both sides, the left is $\frac{1}{N}\sum_r \|x_r\|^2 - \|\bar{x}\|^2$. The right is $\frac{1}{2N^2}\big(2N\sum_r \|x_r\|^2 - 2\|\sum_r x_r\|^2\big)$, which is the same after substituting $\sum_r x_r = N\bar{x}$. We use this identity in §L.4 to convert the pairwise disagreement controlled by Lemma 1 into the spread about the mean measured by Ξ_t.

(F8) Two exponential bounds. For all real x, $1 + x \le e^x$; and for $0 \le u \le \frac{1}{2}$, $e^u \le 1 + 2u$. *Proof.* The first is the tangent-line bound on the convex function e^x at $x = 0$. For the second, $e^u - 1 = \int_0^u e^s\, ds \le u\, e^u \le 2u$ whenever $e^u \le 2$, which holds for $u \le \ln 2 \approx 0.693$. These bounds convert a product of R per-step growth factors into $1 + O(p)$, allowing the window recursion to contract.

L.2 Proof of Lemma 1 (averaging invariance)

Proof. By (3), the weighted mean of the corrections at an apply event is

$$\bar{W} - \sum_r w_r W_r^{\mathrm{snap}} \;=\; 0.$$

Thus the correction leaves $\sum_r w_r \boldsymbol{\theta}_r$ unchanged for any weights that sum to one.

For disagreement, fix a sampled coordinate i and decompose replica r's current value into its snapshot and post-snapshot increment:

$$\boldsymbol{\theta}_r[i] \;=\; W_r^{\mathrm{snap}}[i] \;+\; \delta_r[i],$$

which is what defines $\delta_r[i]$: the increment accumulated over the D steps since the snapshot. The correction replaces the snapshot part by the population value, sending $\boldsymbol{\theta}_r[i] \mapsto \bar{W}[i]+\delta_r[i]$. Subtract two replicas and the term $\bar{W}[i]$, being common, cancels:

$$\boldsymbol{\theta}_r[i] - \boldsymbol{\theta}_{r'}[i] \;=\; \big(\bar{W}[i] + \delta_r[i]\big) - \big(\bar{W}[i] + \delta_{r'}[i]\big) \;=\; \delta_r[i] - \delta_{r'}[i].$$

The snapshot disagreement cancels, leaving only the difference between the two D-step increments.

Assumption (A4) gives $D \le R$, so at most one correction is in flight, and Algorithm 1 applies a completed correction before taking a new snapshot. Hence δ_r contains only D local optimizer steps and their common assists.

It remains to bound the residual. Expanding δ_r, the common assist cancels from the difference, leaving only the gradients:

$$\delta_r - \delta_{r'} \;=\; -\eta \sum_{j<D} \big(g_{r,j} - g_{r',j}\big).$$

The D gradient differences are dependent, so we apply (F2) rather than discarding their cross terms:

$$\mathbb{E}\big\|\delta_r - \delta_{r'}\big\|^2 \;\le\; D \sum_{j<D} \eta^2\, \mathbb{E}\big\|g_{r,j} - g_{r',j}\big\|^2.$$

For each term, apply (F3) with $a = g_{r,j}$ and $b = -g_{r',j}$, followed by the second-moment bound $\mathbb{E}\|g\|^2 \le G^2$ in (A2):

$$\mathbb{E}\|g_{r,j} - g_{r',j}\|^2 \;\le\; 2\mathbb{E}\|g_{r,j}\|^2 + 2\mathbb{E}\|g_{r',j}\|^2 \;\le\; 2G^2 + 2G^2 \;=\; 4G^2,$$

so substituting, and remembering the sum has D terms,

$$\mathbb{E}\big\|\delta_r - \delta_{r'}\big\|^2 \;\le\; D \cdot D \cdot \eta^2 \cdot 4G^2 \;=\; 4D^2\eta^2G^2,$$

which proves the claim. One factor of D counts summands; the other comes from (F2). □

Remark 1. The result follows from Eq. (3) and a single subtraction. Averaging preserves the mean and leaves only a bounded delay residual, so the fleet mean moves through the local optimizer steps and the assist. The latter is analyzed in Lemma 2 and Theorem 1.

L.3 Proof of Lemma 2 (estimator stability)

Proof. Fix a coordinate and index averaging windows by w. Within a window, $\mathbf{M}$ is fixed and the additive assist contributes the same $\lambda\mathbf{M}_w$ at each of the window's R steps, contributing $\lambda R\mathbf{M}_w$ in total. So the displacement observed across the window is not the genuine optimizer motion alone but that motion plus the buffer's own footprint,

$$\underbrace{u_w}_{\text{genuine}} \;+\; \underbrace{\lambda R\,\mathbf{M}_w}_{\text{self-injection}},$$

where the second term is self-injection.

Suppose a fraction $\gamma \in [0,1]$ of the self-injection is subtracted before the EMA sees it. The quantity entering (4) becomes $u_w + (1-\gamma)\lambda R\mathbf{M}_w$, since the deduction removes $\gamma\lambda R\mathbf{M}_w$. Substituting into $\mathbf{M}_{w+1} = \mu\mathbf{M}_w + (1-\mu)(\text{observation})$ gives

$$\begin{aligned}
\mathbf{M}_{w+1} &= \mu\mathbf{M}_w + (1-\mu)\big(u_w + (1-\gamma)\lambda R\mathbf{M}_w\big) \\
&= \mu\mathbf{M}_w + (1-\mu)(1-\gamma)\lambda R\,\mathbf{M}_w + (1-\mu)\,u_w \\
&= \underbrace{\big(\mu + (1-\mu)(1-\gamma)\lambda R\big)}_{=:\,a_\gamma}\,\mathbf{M}_w \;+\; (1-\mu)\,u_w,
\end{aligned}$$

This is the stated recursion. Deduction changes a_γ but not the drive term $(1-\mu)u_w$.

Unrolling the recursion gives $\mathbf{M}_{w+1} = a_\gamma\big(a_\gamma \mathbf{M}_{w-1} + (1-\mu)u_{w-1}\big) + (1-\mu)u_w$, and continuing back to the start,

$$\mathbf{M}_w \;=\; a_\gamma^{\,w}\,\mathbf{M}_0 \;+\; (1-\mu)\sum_{j<w} a_\gamma^{\,j}\,u_{w-1-j}.$$

By (F6), if $a_\gamma < 1$, the initial condition decays geometrically and the drive has convergent geometric weights. Hence, $\mathbf{M}_w$ has a uniformly bounded second moment. If $a_\gamma = 1$, the sum is undamped: a constant nonzero-mean drive produces linear growth, while an i.i.d. zero-mean non-degenerate drive produces linearly growing variance. This case is marginal. If $a_\gamma > 1$, the recursion grows geometrically.

Since $a_\gamma = \mu + (1-\mu)(1-\gamma)\lambda R$,

$$a_\gamma < 1 \iff (1-\mu)(1-\gamma)\lambda R < 1-\mu \iff (1-\gamma)\,\lambda R < 1,$$

where division by $1-\mu$ is valid because $\mu < 1$. Without deduction ($\gamma = 0$), the condition is $\lambda < 1/R$, with $\lambda = 1/R$ as the marginal case. For any partial deduction $\gamma < 1$, instability occurs once $\lambda \geq 1/((1-\gamma)R)$. At $\gamma = 1$, the self-injection term vanishes and $a_1 = \mu < 1$ for any λ. Exact deduction is therefore the only case that remains stable for arbitrary coupling.

With exact deduction, we remove the accumulated injection $Q = \lambda R\,\mathbf{M}_w$ before the EMA, leaving the plain recursion $\mathbf{M}_{w+1} = \mu\mathbf{M}_w + (1-\mu)u_w$. Suppose the drive has constant mean, $\mathbb{E}u_w \equiv u$. Taking expectations in the unrolled form with $a_\gamma = \mu$, and using the geometric sum $\sum_{j<w}\mu^j \to 1/(1-\mu)$,

$$\mathbb{E}\mathbf{M}_w \;=\; \mu^w\,\mathbb{E}\mathbf{M}_0 + (1-\mu)\sum_{j<w}\mu^j u \;\longrightarrow\; (1-\mu)\cdot\frac{u}{1-\mu} \;=\; u \quad \text{geometrically.}$$

Thus, the buffer tracks the expected consensus velocity for any coupling. □

Remark 2 (Numerical values). At PACE's $\mu = 0.5$, $R = 5$, $\gamma = 0$ the coefficient is $a = 0.5 + 2.5\lambda$, giving $a \in [3, 10.5]$ for $\lambda \in [1, 4]$ — the divergence example in §4.2. The undeducted threshold satisfies $a = 1 \iff \lambda R(1-\mu) = 1-\mu$, i.e. $\mu_{\text{eff}} = 1-\mu$ in the units of Eq. (7), whereas the NAG point is $\mu_{\text{eff}} = \mu$. These thresholds coincide at the paper's setting $\mu = 1/2$. This agreement is specific to $\mu = 1/2$ and is not a general identity.

Remark 3 (scope). The lemma is *open loop*: it assumes bounded optimizer-driven displacements u_w and establishes stability of the estimator, not of the coupled optimization process. Theorem 1 treats the latter. The coupled process can degrade for reasons outside this lemma; for example, a sustained $\lambda \equiv 4$ costs $+2.030$ in Table 3 with no estimator instability at all. The accumulator here is snapshot-aligned; §4.2 describes the implementation's apply-aligned accumulator and its boundary residual.

L.4 Proof of Proposition 1 (consensus error)

Proof. Let Ξ_w be the consensus error just before averaging window w. The error grows during the R local steps and contracts when an averaging event lands.

Between averaging events, replica r takes $\boldsymbol{\theta}_{r,t+1} = \boldsymbol{\theta}_{r,t} - \eta g_{r,t} + (\text{common assist})$. Averaging over the N replicas, and writing $\bar{g}_t = \frac{1}{N}\sum_r g_{r,t}$,

$$\bar{\boldsymbol{\theta}}_{t+1} \;=\; \bar{\boldsymbol{\theta}}_t - \eta\,\bar{g}_t + (\text{common assist}),$$

the assist unchanged by averaging because it is the same vector for each replica. Subtracting the second display from the first, the assist cancels and we are left with a recursion for the deviation alone:

$$e_{r,t+1} \;=\; e_{r,t} \;-\; \eta\big(g_{r,t} - \bar{g}_t\big). \tag{$*$}$$

Thus, only the difference between a replica's gradient and the fleet mean can change $e_{r,t}$.

Split each stochastic gradient into its true gradient and noise, $g_{r,t} = \nabla f(\boldsymbol{\theta}_{r,t}) + \xi_{r,t}$ with $\mathbb{E}\xi_{r,t} = 0$. Substituting into $(*)$ sorts the motion accordingly,

$$e_{r,t+1} \;=\; \underbrace{e_{r,t} - \eta\,d_{r,t}}_{=:\,a_r} \;+\; \underbrace{\big(-\eta\,\nu_{r,t}\big)}_{=:\,b_r}, \qquad d_{r,t} = \nabla f(\boldsymbol{\theta}_{r,t}) - \tfrac{1}{N}\textstyle\sum_s \nabla f(\boldsymbol{\theta}_{s,t}), \quad \nu_{r,t} = \xi_{r,t} - \bar{\xi}_t,$$

where d is the *drift* caused by evaluating true gradients at different parameters, and ν is the *noise*. Taking squared norms and expectations,

$$\mathbb{E}\|e_{r,t+1}\|^2 \;=\; \mathbb{E}\|a_r\|^2 \;+\; 2\,\mathbb{E}\langle a_r, b_r\rangle \;+\; \mathbb{E}\|b_r\|^2,$$

The cross term is zero because a_r is measurable from the history at step t, while b_r is proportional to fresh zero-mean noise. Thus $\mathbb{E}[\langle a_r, b_r\rangle] = \mathbb{E}[\langle a_r, \mathbb{E}[b_r \mid \text{past}]\rangle] = 0$ (the martingale-difference property), and

$$\mathbb{E}\|e_{r,t+1}\|^2 \;=\; \mathbb{E}\|a_r\|^2 \;+\; \eta^2\,\mathbb{E}\|\nu_{r,t}\|^2. \tag{$**$}$$

For the noise term,

$$\mathbb{E}\|\xi_r - \bar{\xi}\|^2 = \mathbb{E}\|\xi_r\|^2 - 2\,\mathbb{E}\langle \xi_r, \bar{\xi}\rangle + \mathbb{E}\|\bar{\xi}\|^2,$$

and because the replicas' noises are independent and mean-zero, every cross term $\mathbb{E}\langle \xi_r, \xi_s\rangle$ with $s \neq r$ vanishes. That leaves $\mathbb{E}\langle \xi_r, \bar{\xi}\rangle = \frac{1}{N}\mathbb{E}\|\xi_r\|^2$ and $\mathbb{E}\|\bar{\xi}\|^2 = \frac{1}{N^2}\sum_s \mathbb{E}\|\xi_s\|^2$, so substituting and averaging over r the three pieces collect into

$$\frac{1}{N}\sum_r \mathbb{E}\|\xi_r - \bar{\xi}\|^2 \;=\; \Big(1 - \frac{1}{N}\Big)\cdot\frac{1}{N}\sum_r \mathbb{E}\|\xi_r\|^2 \;\le\; \Big(1 - \frac{1}{N}\Big)\frac{\sigma^2}{B} \;\le\; \frac{\sigma^2}{B},$$

using (A2) and a local batch of B samples, whose mean has variance σ^2/B. Each step therefore injects at most $\eta^2\sigma^2/B$ into Ξ; the factor $(1 - 1/N)$ is a mild bonus from subtracting the fleet mean, which we discard.

For the drift term, rewrite it as an average of pairwise gradient differences, $d_{r,t} = \frac{1}{N}\sum_s \big(\nabla f(\boldsymbol{\theta}_{r,t}) - \nabla f(\boldsymbol{\theta}_{s,t})\big)$, and apply (F2) in its averaging form followed by L-smoothness (A1):

$$\|d_{r,t}\|^2 \;\le\; \frac{1}{N}\sum_s \big\|\nabla f(\boldsymbol{\theta}_{r,t}) - \nabla f(\boldsymbol{\theta}_{s,t})\big\|^2 \;\le\; \frac{L^2}{N}\sum_s \|\boldsymbol{\theta}_{r,t} - \boldsymbol{\theta}_{s,t}\|^2 \;=\; \frac{L^2}{N}\sum_s \|e_{r,t} - e_{s,t}\|^2,$$

the last equality because the two copies of $\bar{\boldsymbol{\theta}}_t$ cancel in the difference. Now bound each summand by (F3), $\|e_r - e_s\|^2 \le 2\|e_r\|^2 + 2\|e_s\|^2$, and average over r:

$$\frac{1}{N}\sum_r \|d_{r,t}\|^2 \;\le\; \frac{L^2}{N^2}\sum_r\sum_s \big(2\|e_r\|^2 + 2\|e_s\|^2\big) \;=\; \frac{L^2}{N^2}\cdot 4N\sum_r \|e_r\|^2 \;=\; 4L^2\,\Xi_t,$$

because each squared deviation appears $4N$ times. The RMS drift is therefore at most $2L\sqrt{\Xi_t}$, which produces the multiplicative term in the recursion.

Putting the noise and drift bounds into $(**)$ gives one step of growth. For $\|a_r\|^2 = \|e_{r,t} - \eta d_{r,t}\|^2$ use the parametrized bound of (F3) with a free $\alpha > 0$:

$$\frac{1}{N}\sum_r \mathbb{E}\|a_r\|^2 \;\le\; (1+\alpha)\,\Xi_t + \big(1 + \alpha^{-1}\big)\eta^2\cdot 4L^2\Xi_t \;=\; \Xi_t\Big[(1+\alpha) + 4\big(1+\alpha^{-1}\big)\eta^2L^2\Big].$$

Any α is legitimate here; the useful choice is $\alpha = 2\eta L$, which makes the bracket a perfect square,

$$(1 + 2\eta L) + 4\Big(1 + \tfrac{1}{2\eta L}\Big)\eta^2L^2 = 1 + 2\eta L + 4\eta^2L^2 + 2\eta L = (1 + 2\eta L)^2,$$

so that per step

$$\Xi_{t+1} \;\le\; (1 + 2\eta L)^2\,\Xi_t \;+\; \eta^2\frac{\sigma^2}{B}.$$

Iterating that across the R steps of a window — the situation of (F6), but with $a = (1 + 2\eta L)^2$ possibly exceeding 1 — and writing Ξ_w^+ for the value at the window's end,

$$\Xi_w^+ \;\le\; (1 + 2\eta L)^{2R}\,\Xi_w \;+\; \eta^2\frac{\sigma^2}{B}\sum_{j=0}^{R-1}(1 + 2\eta L)^{2j} \;\le\; (1 + 2\eta L)^{2R}\Big(\Xi_w + R\,\eta^2\frac{\sigma^2}{B}\Big),$$

bounding every term of the geometric sum by its largest. The step-size condition now controls the growth factor. By (F8), $(1+2\eta L)^{2R} \le e^{4\eta LR}$, and $\eta \le c_0 p/(LR)$ makes that exponent at most $4c_0 p$, which for $c_0 \le \frac{1}{16}$ and $p \le 1$ is at most $\frac{1}{4}$; the second bound of (F8) then applies and gives

$$(1+2\eta L)^{2R} \;\le\; 1 + 8c_0\, p \;=:\; 1 + c\, p, \qquad c := 8c_0 \le \tfrac{1}{2}.$$

Since $c\,p \le 1$, the factor multiplying the noise term is at most 2, and the window obeys

$$\Xi_w^{+} \;\le\; (1 + c\,p)\,\Xi_w \;+\; 2R\eta^2\sigma^2/B,$$

This gives a multiplicative $1 + O(p)$ term and the noise accumulated over R steps. The argument up to this point follows the standard local-SGD analysis (e.g. Koloskova et al., 2020) localized to a single window; what is specific to PACE is the reset that follows, and the fact that the step-size condition ties η to the sparsity p rather than to a synchronization period.

We now account for the sparse reset. By (A4), each coordinate is sampled independently with probability p. For a sampled coordinate, Lemma 1 replaces its disagreement with the difference between two D-step post-snapshot increments. An unsampled coordinate retains its previous disagreement. In expectation, a fraction p of the error is replaced by the residual and a fraction $1-p$ is retained.

To express the residual in terms of Ξ, we convert the pairwise disagreement in Lemma 1 to spread about the mean using (F7):

$$\frac{1}{N}\sum_r \mathbb{E}\|\delta_r - \bar\delta\|^2 = \frac{1}{2N^2}\sum_r\sum_{r'} \mathbb{E}\|\delta_r - \delta_{r'}\|^2 \;\le\; \frac{1}{2N^2}\cdot N^2 \cdot 4D^2\eta^2G^2 \;=\; 2D^2\eta^2G^2.$$

Combining with the growth bound, and keeping the looser constant 4 in place of 2 for slack,

$$\begin{aligned}\mathbb{E}\,\Xi_{w+1} \;&\le\; (1-p)\Big[(1+c\,p)\,\Xi_w + 2R\eta^2\sigma^2/B\Big] \;+\; 4p\,D^2\eta^2G^2 \\ &\le\; (1-p)(1+c\,p)\,\Xi_w \;+\; 2R\eta^2\sigma^2/B \;+\; 4p\,D^2\eta^2G^2,\end{aligned}$$

the last step using $1-p \le 1$ on the noise term.

The recursion contracts if its coefficient is smaller than one. Expanding the coefficient and using $c \le \frac{1}{2}$ gives

$$(1-p)(1+c\,p) \;=\; 1 - p + c\,p - c\,p^2 \;\le\; 1 - p + \tfrac{p}{2} \;=\; 1 - \tfrac{p}{2},$$

after discarding the negative term $-cp^2$. We can therefore apply (F6) with contraction factor $a = 1 - p/2 \in [0,1)$ and drive $b = 2R\eta^2\sigma^2/B + 4pD^2\eta^2G^2$. Sparsity has bought contraction: drawing a p fraction of coordinates shrinks the error by a $p/2$ fraction per window, which beats the $1 + cp$ growth precisely because the step-size condition ties η to p.

Since (A4) supplies the common initialization $\Xi_0 = 0$, (F6) applies directly and finishes the argument:

$$\Xi_w \;\le\; \frac{b}{1-a} \;=\; \frac{2R\eta^2\sigma^2/B + 4p\,D^2\eta^2G^2}{p/2} \;=\; 4\eta^2\frac{\sigma^2}{B}\frac{R}{p} + 8\,\eta^2G^2D^2 \;\le\; C\,\eta^2\Big(\tfrac{\sigma^2}{B}\,\tfrac{R}{p} + G^2D^2\Big)$$

for every w, the division by $p/2$ being what turns the noise term's R into R/p and cancels the p in front of the delay term. Values strictly inside a window exceed boundary values by at most one window's growth, absorbed into C.

The same argument holds with the common additive assist in (A5). It cancels in the first subtraction, and (A1)–(A2) apply at the commonly shifted evaluation points. Hence, the assist does not appear in the consensus bound. □

Remark 4 (why D enters quadratically). The delay residual enters as D^2 because the D post-snapshot increments are dependent, so (F2) applies across them rather than orthogonality — in contrast to the noise, where independence across steps makes the contributions accumulate only linearly. A refined martingale treatment of the *noise* component of the residual would improve that component's σ^2 share to linear in D; we do not need it, since at the operating point $D = 5$ against $R/p = 100$ the residual is subdominant.

Remark 5 (two readings of the bound). First, R/p is the expected refresh horizon of a coordinate. The noise term therefore matches the consensus error of *dense periodic averaging with period* $H = R/p$ — DiLoCo's consensus geometry — at the same order, while never performing a dense synchronization and communicating the same amortized volume. Second, the delay enters only through the snapshot residual G^2D^2, so the cost of overlap is small when D is short but grows quadratically with delay. Appendix H.2 evaluates the dependence on p and identifies where this bound is not tight.

L.5 Proof of Theorem 1 (convergence)

Proof. By Lemma 1, only the local SGD steps and the assist change $\bar{\boldsymbol{\theta}}$. Under look-ahead, each replica evaluates its stochastic gradient at $\boldsymbol{\theta}_{r,t}+q_t$, where q_t is the *clipped* common assist of (A5) — used both for the look-ahead and for the post-update assist — so that $\|q_t\| \le \kappa_t \eta G$ by construction. Averaging replica r's update $\boldsymbol{\theta}_{r,t+1} = \boldsymbol{\theta}_{r,t} - \eta g_r(\boldsymbol{\theta}_{r,t}+q_t) + q_t + (\text{correction})$ over r, the corrections average to zero and the common q_t survives unchanged, its average over r being itself:

$$\bar{\boldsymbol{\theta}}_{t+1} \;=\; \bar{\boldsymbol{\theta}}_t - \frac{\eta}{N}\sum_r g_r(\boldsymbol{\theta}_{r,t}+q_t) \;+\; q_t.$$

The assist therefore changes the fleet average and does not appear as replica-specific noise.

To convert this parameter motion into a statement about the loss, apply the descent lemma (F4) with $x = \bar{\boldsymbol{\theta}}_t$, $y = \bar{\boldsymbol{\theta}}_{t+1}$, substitute the display above for $y - x$, and take expectations, splitting the linear term into its two pieces:

$$\mathbb{E} f(\bar{\boldsymbol{\theta}}_{t+1}) \le f(\bar{\boldsymbol{\theta}}_t) \underbrace{- \eta \Big\langle \nabla f(\bar{\boldsymbol{\theta}}_t), \tfrac{1}{N}\textstyle\sum_r \nabla f(\boldsymbol{\theta}_{r,t}+q_t) \Big\rangle}_{\text{(T1) descent}} + \underbrace{\big\langle \nabla f(\bar{\boldsymbol{\theta}}_t), q_t \big\rangle}_{\text{(T2) assist}} + \underbrace{\tfrac{L}{2}\mathbb{E}\|\bar{\boldsymbol{\theta}}_{t+1} - \bar{\boldsymbol{\theta}}_t\|^2}_{\text{(T3) quadratic}},$$

where unbiasedness (A2) replaces the stochastic gradients by true gradients in (T1).

For (T1), write $\frac{1}{N}\sum_r \nabla f(\boldsymbol{\theta}_{r,t}+q_t) = \nabla f(\bar{\boldsymbol{\theta}}_t) + \Delta_t$ and applying (F2) in averaging form, then L-smoothness, then (F3):

$$\|\Delta_t\|^2 \;\le\; \frac{1}{N}\sum_r \big\|\nabla f(\boldsymbol{\theta}_{r,t}+q_t) - \nabla f(\bar{\boldsymbol{\theta}}_t)\big\|^2 \;\le\; \frac{L^2}{N}\sum_r \|e_{r,t}+q_t\|^2 \;\le\; 2L^2\big(\Xi_t + \|q_t\|^2\big),$$

where the two terms correspond to replica spread and the common shift. Applying (F1) to (T1) with $\rho = 1$ gives

$$\begin{aligned}\text{(T1)} &= -\eta\|\nabla f(\bar{\boldsymbol{\theta}}_t)\|^2 - \eta\langle \nabla f(\bar{\boldsymbol{\theta}}_t), \Delta_t\rangle \\ &\le\; -\eta\|\nabla f\|^2 + \tfrac{\eta}{2}\|\nabla f\|^2 + \tfrac{\eta}{2}\|\Delta_t\|^2 \\ &\le\; -\tfrac{\eta}{2}\|\nabla f\|^2 + \eta L^2\big(\Xi_t + \|q_t\|^2\big).\end{aligned}$$

The look-ahead contribution is $\eta L^2\kappa_t^2\eta^2 G^2 = (\eta L)^2\eta G^2\kappa_t^2$. For $\eta \le 1/(8L)$, it is dominated by the guard term below and absorbed into that term.

For (T2), apply Cauchy–Schwarz, the clip, and (F1) with ρ chosen to leave a $\frac{\eta}{8}$ share of the gradient norm:

$$\text{(T2)} \;=\; \langle \nabla f, q_t\rangle \;\le\; \|\nabla f\|\,\|q_t\| \;\le\; \kappa_t\eta G\|\nabla f\| \;\le\; \tfrac{\eta}{8}\|\nabla f\|^2 + 2\eta\kappa_t^2 G^2,$$

the last step being ρ-weighted Young with $a = \sqrt{\eta/4}\,\|\nabla f\|$ and $b = \sqrt{4/\eta}\,\kappa_t\eta G$, for which $ab \le \frac{1}{2}a^2 + \frac{1}{2}b^2$ gives this split. The bound uses only the norm of q_t and allows the assist to point in the worst possible direction. It therefore does not establish acceleration.

Term (T3) contains three parts. Substituting the consensus update and applying (F3), $\|\bar{\boldsymbol{\theta}}_{t+1} - \bar{\boldsymbol{\theta}}_t\|^2 \le 2\eta^2\|\frac{1}{N}\sum_r g_r\|^2 + 2\|q_t\|^2$. The mean-gradient part is $O(\eta^2 L)$ against the $O(\eta)$ descent retained above, so we absorb it for $\eta \le 1/(8L)$. The variance part is, by (F5) with $w_r = 1/N$ and $s_r^2 = \sigma^2/B$,

$$\sum_r w_r^2 s_r^2 \;=\; N\cdot\frac{1}{N^2}\cdot\frac{\sigma^2}{B} \;=\; \frac{\sigma^2}{NB}, \qquad \text{contributing} \quad \eta^2\sigma^2/(NB),$$

which uses conditional independence across replicas from (A2). The remaining piece, $\|q_t\|^2 \le \kappa_t^2\eta^2G^2$, is higher order.

After substituting these bounds into the descent inequality, the coefficient of the squared gradient norm is at most $-\frac{\eta}{2}+\frac{\eta}{8}\le-\frac{\eta}{4}$. Rearranging gives

$$\tfrac{\eta}{4}\,\mathbb{E}\|\nabla f(\bar{\boldsymbol{\theta}}_t)\|^2 \;\le\; f(\bar{\boldsymbol{\theta}}_t)-\mathbb{E}f(\bar{\boldsymbol{\theta}}_{t+1}) \;+\; c\,\eta L^2\Xi_t \;+\; c\,\eta\kappa_t^2G^2 \;+\; c\,L\eta^2\tfrac{\sigma^2}{NB}.$$

Summing over $t=1,\dots,T$, the function-value differences telescope to $f(\bar{\boldsymbol{\theta}}_0)-\mathbb{E}f(\bar{\boldsymbol{\theta}}_T)\le f(\bar{\boldsymbol{\theta}}_0)-f^\star$ since $f^\star$ is the infimum, and dividing by $\eta T/4$,

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla f(\bar{\boldsymbol{\theta}}_t)\|^2 \;\le\; C\left(\frac{f(\bar{\boldsymbol{\theta}}_0)-f^\star}{\eta T} \;+\; \frac{L\eta\sigma^2}{NB} \;+\; L^2\bar{\Xi} \;+\; G^2\bar{\kappa}_T^2\right),$$

with $\bar{\Xi}$ the time-averaged consensus error of Proposition 1 and $\bar{\kappa}_T^2=\frac{1}{T}\sum_t\kappa_t^2$; the absolute constant C collects the Young constants above. □

Remark 6 (Rate). With $\eta=\Theta\big(\min\{\sqrt{NB/T},\,1/L,\,p/(LR)\}\big)$, in the regime where $\sqrt{NB/T}$ binds, the first two terms give the $O(1/\sqrt{NBT})$ decentralized-SGD rate with linear speedup; the consensus term is then $O(1/T)$ for fixed N,B and communication parameters, hence higher order in T, though it grows with system size, so the linear-speedup regime persists only while it remains subordinate. With the assist off ($\kappa_t=0$) the substrate matches dense periodic averaging. With it on, the last term is a *bias floor*: a stationarity neighborhood of gradient-norm radius $O(G\bar{\kappa}_T)$. At fixed N,B a guard schedule $\kappa_t=O(t^{-\alpha})$ with $\alpha\ge1/4$ gives $\bar{\kappa}_T^2=O(1/\sqrt{T})$ and restores $T^{-1/2}$; preserving the full $1/\sqrt{NBT}$ dependence requires the stronger $\bar{\kappa}_T^2=O(1/\sqrt{NBT})$, since the guard term carries no $1/(NB)$ of its own. A schedule with $\kappa_t\to0$ alone removes the floor asymptotically but preserves the rate only if it decays fast enough (e.g. $\kappa_t=1/\log t$ does not).

Remark 7 (The residual term). The proof bounds the assist with worst-case Cauchy–Schwarz. The theorem therefore gives a finite stationarity guarantee with a guard-controlled residual, but does not show that the assist improves optimization. Empirically, its direction is an EMA of the fleet's descent motion, and removing it costs 0.06–0.17 at 500M (0.20 at 226M) in every regime (§6.3). The residual does not shrink automatically late in training. Both $\|\lambda_t\mathbf{M}\|$ and the optimizer step $\|\delta\|$ decay as the LR anneals, but the step decays faster: over the last third of the 500M HOMOGENEOUS run $\|\lambda_t\mathbf{M}\|$ falls to 0.22 of its mid-training value while $\|\delta\|$ falls to 0.08, with λ_t flat at 1 (Fig. 8b). Since κ_t is defined by $\|q_t\|\le\kappa_t\eta G$, it grows by approximately 7× over that interval. The clip in (A5) keeps the bound finite, but the stated rate requires an explicitly decaying guard schedule.

L.6 Proof of Proposition 2 (ATM)

Proof of part (i): variance-bound equalization. Begin with a single rank under naive batch scaling, where rank r's optimizer step averages ρ_rB i.i.d. samples. The mean of n i.i.d. samples has variance $1/n$ times that of one, so by (A2) the step is unbiased with variance at most

$$s_r^2 \;=\; \sigma^2/(\rho_rB).$$

This is an upper bound rather than an identity. A slower rank changes the variance of its contribution, not its mean, so the error is heteroscedastic rather than biased.

For the full fleet, consider fixed weights w with $\sum_r w_r=1$, (F5) together with the single-rank bound gives

$$\mathrm{Var}\Big(\sum_r w_r\hat{g}_r\Big) \;\le\; \frac{\sigma^2}{B}\sum_r\frac{w_r^2}{\rho_r},$$

while the conditional mean remains the exact weighted-mean gradient. With uniform weights $w_r=1/N$, pulling out $1/N^2$ and then one further factor of $1/N$,

$$\frac{\sigma^2}{B}\cdot\frac{1}{N^2}\sum_r\frac{1}{\rho_r} \;=\; \frac{\sigma^2}{BN}\cdot\underbrace{\frac{1}{N}\sum_r\frac{1}{\rho_r}}_{\ge1},$$

where the inequality follows from $1/\rho_r \geq 1$. Uniform averaging therefore inflates the homogeneous bound $\sigma^2/(BN)$ by the mean inverse speed. This factor is unbounded as $\rho_{\min} \to 0$.

The optimal fixed weighting minimizes $\sum_r w_r^2/\rho_r$ subject to $\sum_r w_r = 1$. Applying Cauchy–Schwarz gives

$$1 = \Big(\sum_r w_r\Big)^2 = \Big(\sum_r \frac{w_r}{\sqrt{\rho_r}} \cdot \sqrt{\rho_r}\Big)^2 \leq \Big(\sum_r \frac{w_r^2}{\rho_r}\Big)\Big(\sum_r \rho_r\Big),$$

so $\sum_r w_r^2/\rho_r \geq 1/\sum_r \rho_r$ for *every* feasible w. Cauchy–Schwarz holds with equality exactly when the two sequences are proportional, i.e. when $w_r/\sqrt{\rho_r} = \text{const} \cdot \sqrt{\rho_r}$, which says $w_r \propto \rho_r$. Normalizing,

$$w_r^\star = \frac{\rho_r}{\sum_j \rho_j}, \qquad \text{minimum value } \frac{\sigma^2}{B\sum_r \rho_r} \geq \frac{\sigma^2}{BN},$$

where the final inequality follows from $\sum_r \rho_r \leq N$, with equality only when every $\rho_r = 1$. Thus, capacity weighting minimizes this upper bound, but cannot reach the homogeneous bound when any rank is slow. It also reduces the contribution of slow ranks.

ATM instead equalizes the variance bound. Each rank accumulates the same target of B samples before taking an optimizer step; only the time required to reach that target depends on ρ_r. By (A2), the step variance is at most σ^2/B, independent of the rank's speed.

In the reported runs, ranks follow a shared clock and can accumulate only an integer number of ticks. A rank processing through $\rho_r B$ samples per tick must settle for the nearest integer $k_r = \mathrm{round}(1/\rho_r)$ of them, and its step averages $k_r\rho_r B$ samples with variance at most $\sigma^2/(k_r\rho_r B)$. Rounding to the nearest integer means $|k_r\rho_r - 1| \leq \rho_r/2$, hence $k_r\rho_r = 1 + O(\rho_r)$ and

$$\frac{\sigma^2}{k_r\rho_r B} = \frac{\sigma^2}{B}\big(1 + O(\rho_r)\big),$$

so the remaining dependence on speed comes only from rounding. ATM equalizes the sample counts without reducing a slow rank's contribution. □

Proof of part (ii): staleness in the stylized model. In the stylized model every rank contributes one full-batch-equivalent gradient at every shared counter tick, but evaluated at a parameter point up to $k_{\max} - 1$ ticks old. The gap between the evaluation point and the current one is an accumulation of at most $k_{\max}$ optimizer steps, each of norm at most ηG by (A2). Those steps are dependent, so (F2) with $n = k_{\max}$ applies rather than orthogonality, and the second moment of the gap is at most

$$k_{\max} \cdot k_{\max} \cdot \eta^2 G^2 = k_{\max}^2 \eta^2 G^2.$$

Substituting this bound into the disparity recursion $\Gamma_{m+1} = (1-\alpha)\Gamma_m + \Delta_m$ of Appendix K, whose per-round injection is therefore $\sup_m|\Delta_m| = O(\eta k_{\max} G)$ — the square root of the displacement bound just obtained — puts us in (F6)'s situation with contraction $a = 1 - \alpha$, so

$$|\Gamma^\star| \leq \frac{\sup_m|\Delta_m|}{1-(1-\alpha)} = \frac{\sup_m|\Delta_m|}{\alpha} = O\left(\frac{\eta\, k_{\max} G}{\alpha}\right).$$

The consensus error is quadratic in the disparity, so squaring gives

$$O\left(\frac{\eta^2 k_{\max}^2 G^2}{\alpha^2}\right),$$

corresponding to $O(L^2\eta^2 k_{\max}^2 G^2/\alpha^2)$ in Theorem 1. The contraction modulus appears squared, so the bound degrades as $\alpha \to 0$. The perturbation remains bounded by the fixed point $\Gamma^\star = \Delta/\alpha$ and decreases with the learning-rate schedule because $\Delta \propto \eta$ (Fig. 9). □

Proof of the corollary. Apply parts (i) and (ii) to Theorem 1 within the stylized model. Every rank contributes one gradient per tick, so the leading descent term is the same for both schemes; only the variance and consensus terms differ. From part (i), naive scaling inflates the variance term to

$$\frac{L\eta\sigma^2}{NB} \cdot \frac{1}{N}\sum_r \frac{1}{\rho_r} \qquad \text{(first order in } \eta\text{)},$$

while by the consensus perturbation of part (ii) the bounded-delay effect enters the consensus term at

$$O\left(\frac{L^2\eta^2 k_{\max}^2 G^2}{\alpha^2}\right) \qquad \text{(second order in } \eta\text{)}.$$

At fixed $\rho_{\min} < 1$, $\sigma^2 > 0$ and α bounded away from 0, every other factor is a constant, so the ratio of the staleness side to the variance side is $O(\eta)$: there exists $\eta_0 > 0$ below which the staleness side is strictly smaller, and the ratio vanishes as $\eta \to 0$. □

Remark 8 (scope of part (ii) and the corollary). Part (ii) bounds *parameter* staleness in the assumed mixing model; it is not the implemented ATM clock, under which rank r takes one optimizer step per k_r ticks and its update *count* falls behind without bound, while its parameter disparity is kept bounded only through the averaging contraction. The result does not cover the implemented skipped-update clock. Its participation indicators $I_{r,t}$ change the leading descent term because a slow rank contributes signal less often, with expected participation per tick approximately ρ_r. Part of the per-update variance reduction is therefore offset by lower update frequency. An event-time asynchronous analysis would be needed to cover this case. Table 15 and §6.3 provide the matched wall-clock comparison.

M 1B-SCALE HETEROGENEITY DECOMPOSITION

For the 1B model (width 2,560), we compare full PACE, PACE with weighting but without ATM, and SparseLoCo in MIXED-SPEED. The runs use a global batch of 1,024, 8k training steps, a peak learning rate of 4×10^{-4}, and 4k warmup steps. Each cell reports C4 / FineWeb-Edu loss.

step	PACE (ATM+W)	PACE (W only)	SparseLoCo
2000	**4.458** / 3.822	4.560 / 3.919	4.658 / 3.997
4000	**4.030** / 3.357	4.154 / 3.477	4.296 / 3.579
6000	**3.804** / 3.196	3.895 / 3.267	3.899 / 3.368
8000	**3.658** / 3.085	3.718 / 3.132	3.807 / 3.193

ATM and weighting make similar contributions to the improvement over SparseLoCo at each checkpoint. The C4 gap decreases from 0.27 at step 4k to 0.15 at step 8k. Table 2 gives the longer 500M comparison.

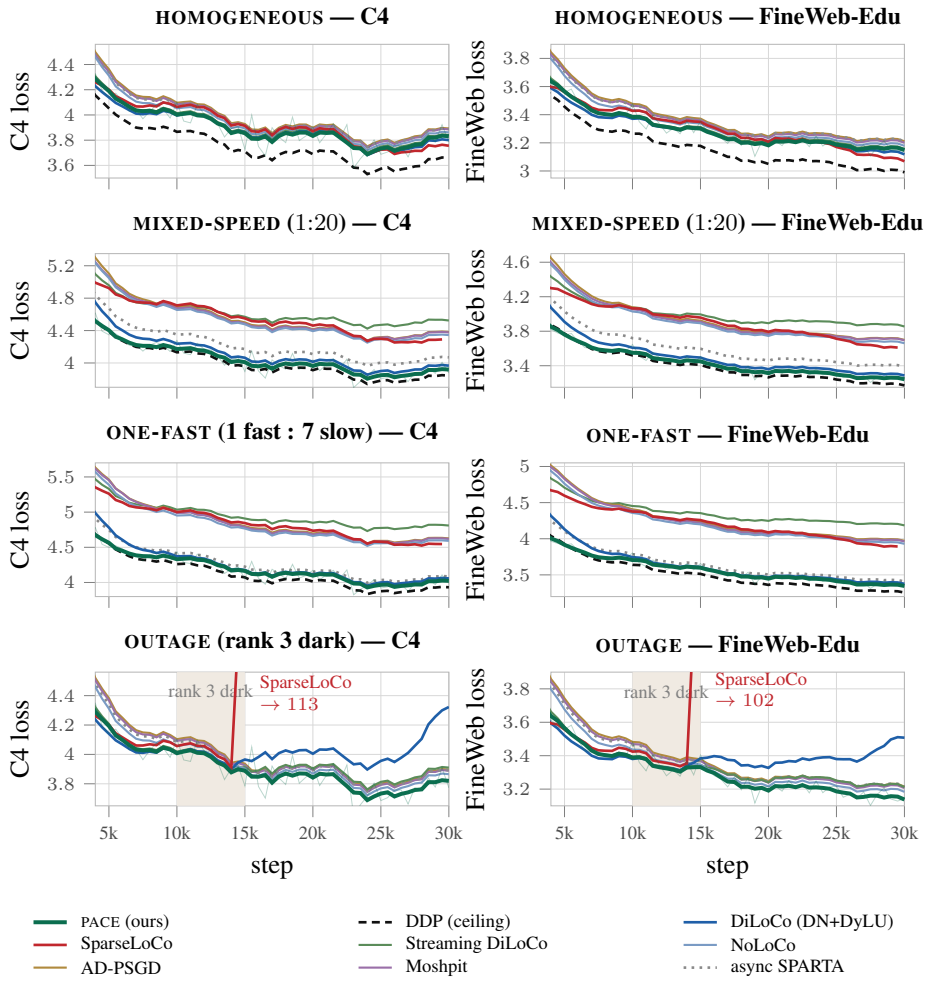


Figure 6: Validation histories for the 500M runs. C4 is on the left and FineWeb-Edu on the right. Lines are centred five-point moving averages of evaluations on the uniformly averaged replicas. Warmup is omitted. In OUTAGE, the shaded interval marks the 5,000 steps during which rank 3 is absent; SparseLoCo leaves the plotted range.

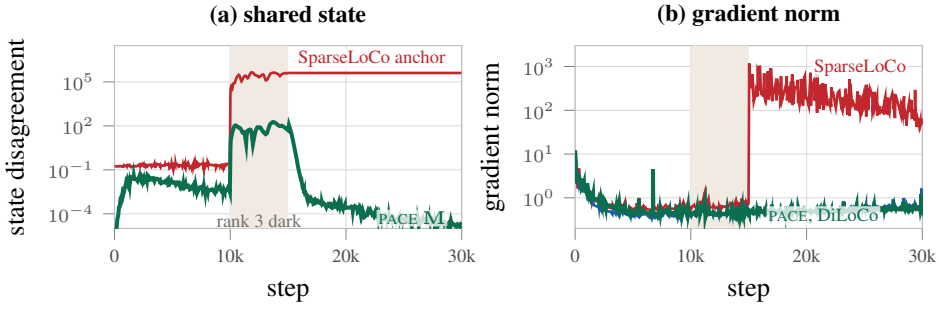


Figure 7: Method state during the 500M outage. The shaded interval marks rank 3's absence. PACE's buffer re-synchronizes after rejoin; SparseLoCo's anchor does not, and its gradient norm remains elevated.

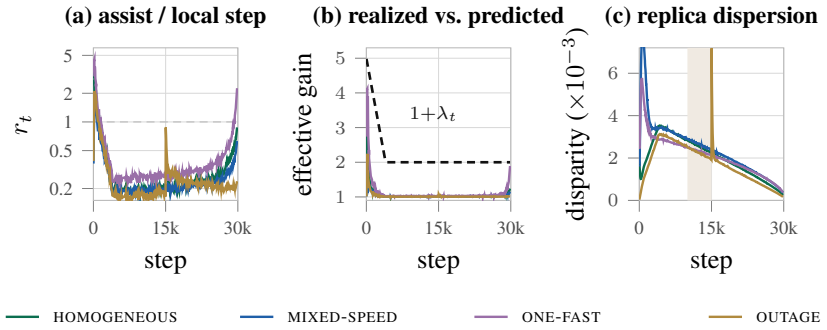


Figure 8: Assist magnitude and replica dispersion at 500M. r_t rises late because the optimizer step decays faster than the assist. The realized gain remains near 1.02. Dispersion falls throughout the second half of training and is lowest in the outage run at the end.

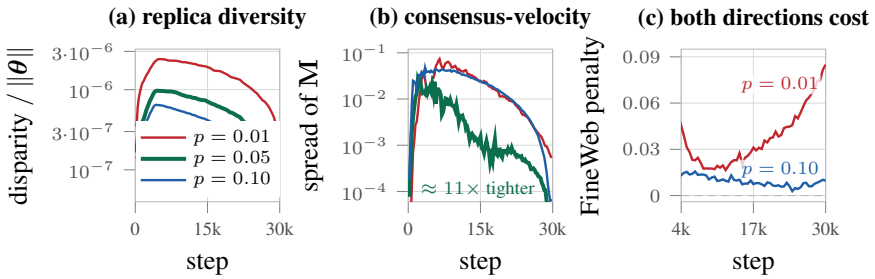


Figure 10: Diagnostics for the sparse-fraction sweep. *(a)* Parameter disparity decreases with p. *(b)* Agreement in $\mathbf{M}$ is best at $p = 0.05$. *(c)* Both neighboring settings have higher FineWeb-Edu loss throughout post-warmup training. See Table 11.

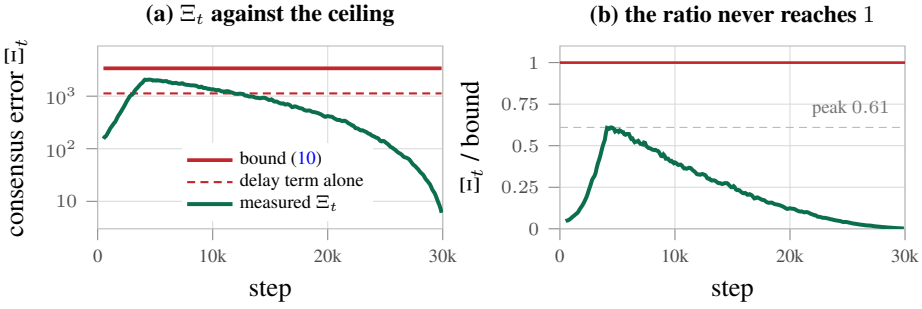


Figure 11: Measured consensus error and the calibrated bound. The 500M MIXED-SPEED run uses $p = 0.05$ and $R = D = 5$. *(a)* The measured Ξ_t remains below the constant-step ceiling, while its peak is 1.83 times the delay term alone. *(b)* The ratio to the full bound peaks at 0.61.

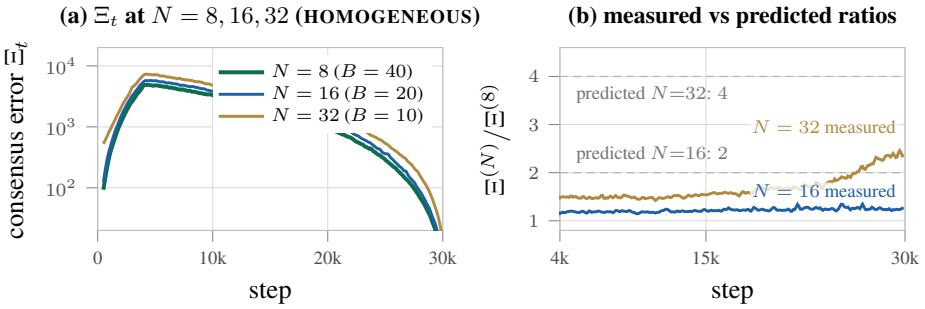


Figure 12: Consensus error as the per-replica batch shrinks. The bound predicts ratios $1 : 2 : 4$ for $N = 8, 16, 32$. The measured ratios are about 1.2 and 1.5–1.9. AdamW keeps the logged step norm within 3% across the runs, making the $1/B$ dependence conservative.

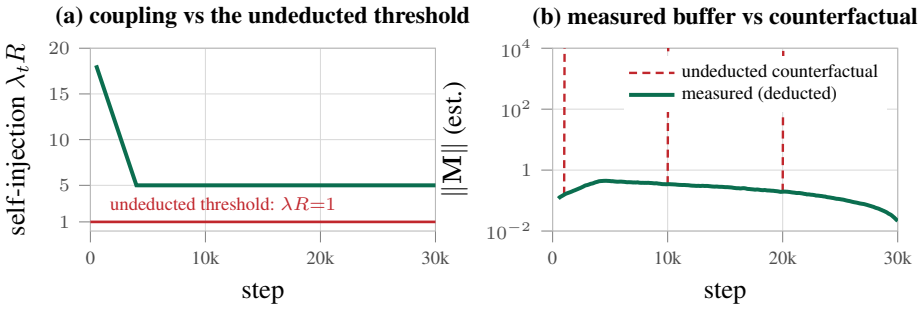


Figure 13: Measured buffer stability with exact deduction. *(a)* The run operates at $\lambda R = 5$–18, above the undeducted threshold of 1. *(b)* The undeducted counterfactual grows by more than 10^6 within 100 steps, while the measured buffer remains bounded.