LACCO: LOCAL ACCUMULATION WITH COMPUTATION–COMMUNICATION OVERLAP

Anonymous authors
Paper under double-blind review

ABSTRACT

Training language models across low-bandwidth networks requires reducing synchronization overhead. Local optimization methods such as DiLoCo communicate only after several local steps, but outer gradient averaging and the outer update remain on the critical path. Existing computation–communication overlap methods avoid this stall through delayed or replica-local updates, which can degrade convergence or cause replicas to drift. We propose LACCO (**L**ocal **A**ccumulation with **C**omputation–**C**ommunication **O**verlap), a two-stage algorithm that overlaps global communication with local computation while applying globally aggregated updates shared by every replica, without introducing staleness in the outer update. We establish convergence guarantees for smooth non-convex objectives. Experiments on language-model pre-training with FineWeb show that LACCO consistently outperforms communication-overlapped baselines such as Eager and Delayed DiLoCo. These gains remain robust as model size scales up to 325M parameters and across a range of communication intervals and numbers of replicas up to 16 workers, while incurring only negligible—or fully overlappable—computational or memory overhead, which demonstrates the strength of LACCO.

1 INTRODUCTION

Training large language models (LLMs) at scale requires distributing computation across hundreds or thousands of accelerators. This is typically achieved through multi-dimensional parallelism (Shoeybi et al., 2019; Narayanan et al., 2021), which balances computation, communication, and memory requirements (Narayanan et al., 2021; Rajbhandari et al., 2020). For example, in distributed data-parallel training (DDP), multiple workers process different mini-batches concurrently and periodically synchronize their updates. At large scale, however, this synchronization can become a major bottleneck (Li et al., 2020; Rajbhandari et al., 2020), as communicating gradients or parameters across workers is costly, particularly when training spans multiple clusters or geographically distributed data centers (Douillard et al., 2023; Diskin et al., 2021). A broad range of techniques has been developed to reduce this communication overhead. Gradient compression methods, such as quantization (Alistarh et al., 2017) and sparsification (Aji & Heafield, 2017), reduce the volume of data exchanged at each synchronization step. An alternative and complementary approach is to communicate less frequently: Local SGD (Stich, 2019; Lin et al., 2020) follows this principle by allowing replicas to perform several local optimization steps before synchronizing their models. Besides reducing communication frequency, this approach can make distributed training more practical in heterogeneous or bandwidth-constrained environments (McMahan et al., 2017; Douillard et al., 2023; Cagnasso et al., 2026).

Yet, reducing the frequency of synchronization does not remove communication from the critical path. In conventional training, workers still remain idle while waiting for a global communication round to complete (Dean et al., 2012; Li et al., 2020). Modern distributed training systems partially mitigate this by overlapping communication with computation, transmitting each parameter or gradient shard (Rajbhandari et al., 2020; Douillard et al., 2025) as soon as it becomes available (Li et al., 2020; PyTorch Team, 2020). By exploiting this natural staggering, these approaches reduce synchronization stalls without altering the underlying optimization. Overlapping communication in Local SGD (Stich, 2019) raises a separate challenge: a communication round spans several local steps, during which local parameters keep evolving, so the communicated update may be stale by the time it is applied (Liu et al., 2024b; Kale et al., 2025). Streaming DiLoCo (Douillard et al., 2025), for instance, staggers updates across parameter fragments, but merges each averaged update

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.

into local parameters that have already advanced, which in practice limits the overlap to a few steps. More generally, naively applying delayed updates degrades convergence (Kale et al., 2025), while compensating for staleness generally requires additional synchronization (Kale et al., 2025), memory (Ren et al., 2021), or computation (Zheng et al., 2017). In particular, methods designed primarily to overlap optimizer computation seek to mitigate the interaction between delayed global updates and multiple intervening local steps, rather than eliminating staleness altogether.

In this work, we pursue this latter direction by combining Local SGD (Stich, 2019) with a computation–communication overlap mechanism inspired by Nabli et al. (2025). We introduce LACCO (**L**ocal **A**ccumulation with **C**omputation–**C**ommunication **O**verlap), an algorithm that overlaps global communication with local optimization without introducing parameter staleness. Existing read–write locking approaches execute communication and computation on separate streams and therefore require two buffers to maintain distinct parameter states (Nabli et al., 2025; Ren et al., 2021). Standard methods (Ren et al., 2021) manage these buffers independently, effectively decoupling communication from local computation, yet introducing staleness. In contrast, the key idea behind LACCO is to swap the communication and local-computation buffers, i.e. two parameter states, between two successive communication stages, while allowing each replica to make local progress. This design has the potential to hide communication latency behind computation while preserving a formulation that enables removal of staleness, and thereby good accuracy as well as simple convergence analysis.

Our main contributions are as follows: **(a) Replica consistency and no staleness.** We propose LACCO, an algorithm that fully overlaps the outer step with local computation. Thanks to our novel two-stage mechanism for pseudo-gradients, replicas remain consistent, without staleness from outer updates. **(b) Theoretical convergence analysis.** We provide a theoretical convergence analysis for smooth non-convex objectives in the Local SGD setting. **(c) Consistent scaling.** Across models up to 325M parameters, communication intervals, replica counts, and matched compute budgets, LACCO achieves the lowest evaluation loss among overlapping methods. It is the first method to fully overlap the outer step with local inner steps while remaining replica-consistent and staleness-free, improving significantly over DiLoCo variants designed for communication–computation overlap, while incurring a limited overhead. We will release our source code upon publication.

2 Related Work

Communication-Efficient Distributed Training. The standard approach to training LLMs is DDP (Li et al., 2020), in which each worker maintains a full copy of the model, computes gradients independently, and synchronizes them at every iteration, leading to communication overhead. One line of work mitigates this overhead by reducing the amount of information exchanged at each training step. Examples include compression (Seide et al., 2014; Lin et al., 2018; Janson et al., 2026; Team et al., 2026), quantization (Alistarh et al., 2017; Bernstein et al., 2018), sparsification (Aji & Heafield, 2017; Wangni et al., 2018), and decentralized synchronous (Koloskova et al., 2020) and asynchronous (Nabli & Oyallon, 2023) methods that communicate only with a subset of neighboring replicas. These techniques can significantly reduce bandwidth requirements, but they modify the underlying optimization or communication procedure and may introduce additional variance. Our method is orthogonal to such approaches: it requires no modification to their underlying mechanisms and can be combined with them to further reduce communication overhead.

Local updates. A complementary line of work, on which we build, reduces communication overhead by synchronizing less frequently (McMahan et al., 2017). In Local SGD (Stich, 2019), replicas perform multiple local optimization steps before averaging their model parameters. FedOpt (Reddi et al., 2021) generalizes this procedure through a bilevel optimization perspective: model averaging is interpreted as an outer server update, and the negative average displacement of the local models is treated as an outer gradient, or "pseudo-gradient". More recently, DiLoCo (Douillard et al., 2023) adapted this framework to LLM pre-training for low-bandwidth networks. It devotes significant design effort to the outer update, and in particular the choice of Nesterov SGD (Nesterov, 1983; Sutskever et al., 2013) and its hyperparameters, yielding stronger empirical convergence than vanilla Local SGD. Nevertheless, while these methods reduce communication frequency, replicas must wait for the outer gradients to be aggregated and the outer update to be applied before beginning the next sequence of local steps. Our method extends DiLoCo by fully overlapping this outer iteration with the subsequent local computation, thereby hiding communication latency without introducing staleness.

Asynchronous Training. One way to hide communication latency is asynchronous training, which removes global synchronization entirely, and thus the straggler effect (Dean et al., 2012; Lian et al., 2015). In parameter-server architectures (Agarwal & Duchi, 2011; Zhang et al., 2013; Mishchenko et al., 2022), each replica updates the global model on the server as soon as it completes a single step. However, this introduces gradient staleness: gradients computed on an older version of the model are applied to a model that has since been updated by other workers. As a consequence, extensive research has addressed this via delay compensation (Zheng et al., 2017), polynomial staleness decay (Xie et al., 2019), delay-dependent learning rates (Barkai et al., 2019), and weight-space lookahead correction (Hakimi et al., 2019). Recently, Liu et al. (2024b) extended asynchrony to the DiLoCo framework through delayed Nesterov updates and dynamic local updates, showing that some existing asynchronous approaches (Zheng et al., 2017; Xie et al., 2019) transfer poorly to this setting. In a similar vein, Ajanthan et al. (2025) alleviate staleness via a lookahead delay correction mechanism that extrapolates in the negative direction of momentum. All these methods apply delayed updates and then correct for them; in contrast, our two-stage mechanism never applies an outer gradient from a previous outer iteration, and thus avoids staleness by design. Beyond staleness, Decoupled DiLoCo (Decoupled DiLoCo Team, 2026) uses asynchrony for fault tolerance, through quorum-based fragment aggregation that relaxes replica consistency to tolerate stragglers and failures. Our method does not address stragglers or failures, which we leave for future work.

Table 1: **Comparison of outer computation–communication overlap methods among Local SGD methods. Full outer overlap** denotes overlapping a complete outer communication round with a local phase. **Replica consistency** means that replicas share identical outer model parameters after each outer update; **No "local" sync** means that the next outer update does not require outer gradients from the ongoing local phase.

Method	Comm quantity	Full outer overlap	Replica consistency	No "local" sync	Staleness free	Convergence guarantees
Overlap-Local SGD	Parameters	✗	✓	✗	✗	✓
CO2	Parameters	✓	✓	✗	✗	✓
DiLoCo	Outer grads	✗	✓	✓	✓	✓
Streaming DiLoCo	Outer grads	✗	✗	✗	✗	?
Delayed DiLoCo	Outer grads	✓	✗	✓	✗	?
Eager DiLoCo	Outer grads	✓	✗	✗	✗	?
LACCO	Outer grads	✓	✓	✓	✓	✓

Overlapping Communication and Computation. We do not discuss the part of the literature that focuses on overlapping communication and computation across stages associated with other forms of parallelism, such as pipeline parallelism (Liu et al., 2024a; Rivaud et al., 2025) or tensor parallelism (Zhang et al., 2025; Singh et al., 2025). Instead, we restrict our focus to DDP. Unlike asynchronous methods, computation–communication overlap methods preserve an explicit outer-step structure: replicas perform local steps while communication is in flight and, once it completes, all receive the same update before proceeding. Among them, an early framework is Overlap-Local SGD (Wang et al., 2020), where replicas perform local steps and are then pulled toward an anchor model, computed as the average of the replicas delayed by one communication round. Building on the same idea, CO2 (Sun et al., 2024) updates the global model with averaged parameters delayed by one outer iteration, but mitigates the resulting staleness through momentum clipping and a staleness-gap penalty. Crucially, both methods introduce delays, which hurts convergence. To retain DiLoCo's outer momentum, Streaming DiLoCo (Douillard et al., 2025) overlaps outer gradient averaging of parameter subsets with subsequent local steps, and its authors report negligible degradation up to five steps. By contrast, our method widens the overlap to all local steps and does not introduce staleness. A simple way to achieve a full overlap with all the local steps, discussed by Kale et al. (2025) as Delayed DiLoCo, is to use outer gradients delayed by one outer iteration. Nonetheless, applying only this delayed outer gradient comes at the cost of worse convergence. To address this, Eager DiLoCo (Kale et al., 2025) mixes each replica's fresh local outer gradient with the averaged outer gradient from the previous outer iteration. While this improves training convergence, relying on each

replica's own local outer gradient can introduce replica drift and requires a "local" synchronization step, as replicas must wait for the inner steps to finish before computing the outer iteration. Our method is compatible with this correction, but on its own it is more robust than Eager DiLoCo as the number of replicas increases and removes the need for this "local" synchronization step. Moreover, Factored Gossip DiLoCo (Hewa Koneputugodage et al., 2026) similarly overlaps communication, but by relaxing exact synchronization through a mixing/gossip step, a setting we do not address. Table 1 summarizes the differences between the above methods. Finally, our work builds on Nabli et al. (2025), but differs in two important respects: ACCO can be viewed as operating in the $H = 1$ regime, whereas our setting considers multiple local optimization steps. As a result, the standard ACCO formulation cannot directly rely on an inner–outer loop decomposition. Second, optimizer offloading is not feasible in our setting because the interval between consecutive gradient steps is too short to effectively hide the associated overhead. These differences require a substantially different treatment and constitute a significant extension beyond the original ACCO setting.

3 METHOD

3.1 LOCAL SGD SETUP

We consider the minimization of a differentiable objective $f : \mathbb{R}^d \to \mathbb{R}$ of the form

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

where $\theta \in \mathbb{R}^d$ denotes the global model, $\xi \sim \mathcal{D}$ is a random sample from a dataset $\mathcal{D}$, and $F(\theta; \xi)$ is the sample loss. We consider a distributed, data-homogeneous setting with K replicas, each sampling data from the same distribution $\mathcal{D}$. Let $\theta_t \in \mathbb{R}^d$ denote the global model at outer iteration t, and $\theta^i_{t,h} \in \mathbb{R}^d$ denote the local model of replica i after h local optimization steps at outer iteration t. Initially, we assume that $\theta^i_{0,0} = \theta_0$ for all $i = 1, \dots, K$. Within each outer iteration t, replicas perform H local steps using an inner optimizer, denoted by $\mathrm{InnerOpt}$, typically AdamW (Loshchilov & Hutter, 2019). We denote these updates, for $\xi^i_{t,h} \sim \mathcal{D}$ independently sampled for each worker,

$$\theta^i_{t,h+1} = \mathrm{InnerOpt}\left(\theta^i_{t,h}, \nabla F\left(\theta^i_{t,h}; \xi^i_{t,h}\right)\right), \qquad h = 0, \dots, H-1, \tag{2}$$

where the initialization $\theta^i_{t,0}$ for $t > 0$ depends on the specific method under consideration.

3.2 BACKGROUND

DiLoCo. At each outer iteration t, replica i initializes $\theta^i_{t,0} = \theta_t$ and performs H inner steps. It then computes its outer gradient g^i_t, which is averaged across replicas to obtain g_t. The averaged gradient is applied to θ_t using an outer optimizer, denoted by $\mathrm{OuterOpt}$. The resulting update is

$$\begin{cases} g^i_t = \theta_t - \theta^i_{t,H}, \\ \theta_{t+1} = \mathrm{OuterOpt}(\theta_t, g_t), \end{cases} \qquad \text{with} \quad g_t = \frac{1}{K} \sum_{i=1}^{K} g^i_t. \tag{3}$$

Synchronizing the outer gradient g_t forces all replicas to be idle before updating θ_{t+1} and starting the next H inner steps. A simple way to avoid this waiting is to use the outer gradient g_{t-1} from the previous outer iteration, which we discuss below.

Delayed DiLoCo. Delayed DiLoCo (Kale et al., 2025) delays outer updates by one iteration to overlap communication with computation. Each replica initializes $\theta^i_{t,0} = \theta^i_t$ (with equal θ^i_0 across replicas), performs the inner steps, and asynchronously communicates its local outer gradient. At $t = 0$, replicas proceed directly to the next inner steps, while for $t > 0$, they first apply the previously communicated gradient g_{t-1} to $\theta^i_{t,0}$. The resulting update is

$$\begin{cases} g^i_t = \theta^i_t - \theta^i_{t,H}, \\ \theta^i_{t+1} = \begin{cases} \theta^i_{t,H}, & t = 0, \\ \mathrm{OuterOpt}(\theta^i_t, g_{t-1}), & t > 0, \end{cases} \end{cases} \qquad \text{with} \quad g_{t-1} = \frac{1}{K} \sum_{i=1}^{K} g^i_{t-1}. \tag{4}$$

This allows the communication of g^i_t to overlap with the next local steps, thereby hiding communication latency. However, applying the one-step-delayed outer gradient g_{t-1} to the current model introduces staleness, which can degrade convergence in both local and non-local training settings (Kale et al., 2025; Nabli et al., 2025; Nabli & Oyallon, 2025), as we will also see in Section 4.

Eager DiLoCo. To correct the staleness introduced by g_{t-1}, Eager DiLoCo (Kale et al., 2025) replaces each replica's stale contribution with its fresh outer gradient g_t^i, available at the end of the H inner steps on worker i. The resulting update is:

$$\begin{cases} g_t^i = \theta_t^i - \theta_{t,H}^i, \\ \theta_{t+1}^i = \begin{cases} \theta_{t,H}^i, & t = 0, \\ \mathrm{OuterOpt}(\theta_t^i, \hat{g}_t^i), & t > 0, \end{cases} \end{cases} \quad \text{with} \quad g_{t-1} = \frac{1}{K}\sum_{i=1}^{K} g_{t-1}^i, \tag{5}$$

where for $t > 0$, we define

$$\hat{g}_t^i = g_{t-1} + \frac{1}{K}\left(g_t^i - g_{t-1}^i\right) = \frac{1}{K}\left(g_t^i + \sum_{j \neq i} g_{t-1}^j\right). \tag{6}$$

Because the correction depends on the replica, replicas apply different outer updates and can drift apart over time. Our method, by contrast, preserves consistency across replicas and does not require the computation of local steps to be finished before applying the outer step.

3.3 LACCO: Local Accumulation with Computation–Communication Overlap

Following Nabli et al. (2025), and in contrast to DiLoCo variants, we maintain a pair of parameters $(\theta_t, \tilde{\theta}_t)$, for both communication and computation. At each time step t, (*i*) the roles of the two parameters are swapped, after which (*ii*) a lookahead stage and (*iii*) a main stage are performed, each comprising H inner steps. This yields the update sequence $(\theta_t, \tilde{\theta}_t) \to (\theta_t, \tilde{\theta}_{t+1}) \to (\theta_{t+1}, \tilde{\theta}_{t+1})$. As in DiLoCo (Douillard et al., 2023), LACCO communicates an outer gradient every H inner steps, preserving the number of samples processed between communication rounds. We adjust the total number of steps accordingly so that the overall number of communications (and thus computations) remains unchanged. Figure 1 highlights the resulting difference with DiLoCo. Accordingly, each replica maintains two inner optimizers, $\mathrm{InnerOpt}_{\mathrm{M}}$ and $\mathrm{InnerOpt}_{\mathrm{L}}$, and two outer optimizers, $\mathrm{OuterOpt}_{\mathrm{M}}$ and $\mathrm{OuterOpt}_{\mathrm{L}}$, associated with θ_t and $\tilde{\theta}_t$, respectively.

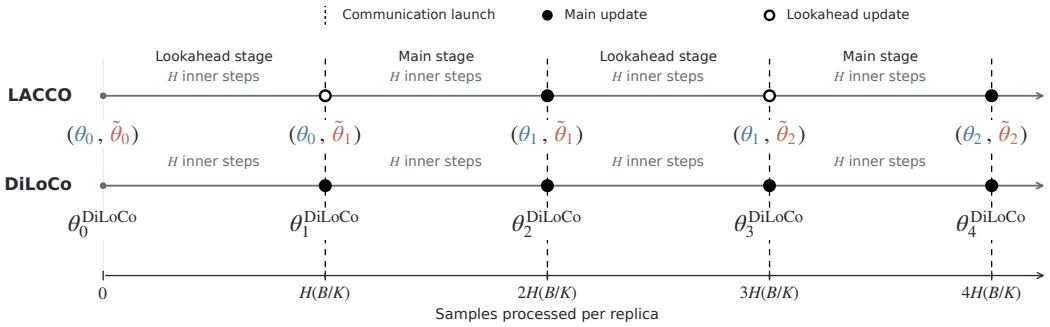


Figure 1: **Timeline of outer updates in LACCO and DiLoCo.** Vertical marks indicate communication launches of outer gradients. B is the global batch size and K is the number of replicas.

Lookahead stage. Each replica i initializes $\theta_{t,0}^i = \theta_t$ and performs H inner steps, defined as

$$\theta_{t,h+1}^i = \mathrm{InnerOpt}_{\mathrm{M}}\left(\theta_{t,h}^i, \nabla F\left(\theta_{t,h}^i; \xi_{t,h}^i\right)\right), \quad h = 0, \ldots, H-1, \tag{7}$$

where $\xi_{t,h}^i \sim \mathcal{D}$. It then computes its local outer gradient g_t^i and communicates it asynchronously to obtain g_t. During these steps, communication of $\tilde{g}_t$, computed from $\tilde{\theta}_t$, completes. We apply this gradient to θ_t to construct the $\tilde{\theta}_{t+1}$. Formally, the resulting update is thus

$$\begin{cases} g_t^i = \theta_t - \theta_{t,H}^i, \\ \tilde{\theta}_{t+1} = \mathrm{OuterOpt}_{\mathrm{L}}(\theta_t, \tilde{g}_t), \end{cases} \quad \text{with} \quad g_t = \frac{1}{K}\sum_{i=1}^{K} g_t^i. \tag{8}$$

Main stage. Each replica i initializes $\tilde{\theta}_{t+1,0}^i = \tilde{\theta}_{t+1}$ and performs H inner steps, defined as

$$\tilde{\theta}_{t+1,h+1}^i = \mathrm{InnerOpt}_{\mathrm{L}}\left(\tilde{\theta}_{t+1,h}^i, \nabla F\left(\tilde{\theta}_{t+1,h}^i; \tilde{\xi}_{t+1,h}^i\right)\right), \quad h = 0, \ldots, H-1, \tag{9}$$

where $\tilde{\xi}^i_{t+1,h} \sim \mathcal{D}$. It then computes its local outer gradient $\tilde{g}^i_{t+1}$ and launches its asynchronous communication. Once g_t is available, we mix it with $\tilde{g}_t$ to update the θ_{t+1}, so that

$$\begin{cases} \tilde{g}^i_{t+1} = \tilde{\theta}_{t+1} - \tilde{\theta}^i_{t+1,H}, \\ \theta_{t+1} = \mathrm{OuterOpt}_{\mathrm{M}}(\theta_t, \frac{1}{2}(g_t + \tilde{g}_t)), \end{cases} \qquad \text{with} \quad \tilde{g}_{t+1} = \frac{1}{K}\sum_{i=1}^{K} \tilde{g}^i_{t+1}. \tag{10}$$

Implementation. Each stage launches asynchronous communication of the outer gradient, overlapping it with local computation in the subsequent stage, as illustrated in Figure 2. Each stage maintains separate outer and inner optimizer states; for the inner optimizers, we found it beneficial to average their momentum states at the end of each iteration. While maintaining two sets of parameters and optimizer states introduces additional memory overhead relative to DiLoCo and its variants, only one inner trajectory is active at a time. The inactive AdamW state can therefore be offloaded to CPU memory and prefetched before the next stage. The outer Nesterov momentum is similarly well suited for offloading, as it is accessed only during outer updates rather than at every local step. This allows most of the additional memory cost to be shifted from GPU to CPU memory, with only a small timing overhead, quantified in Section 4.1. Complete pseudocode is provided in Appendix 1.

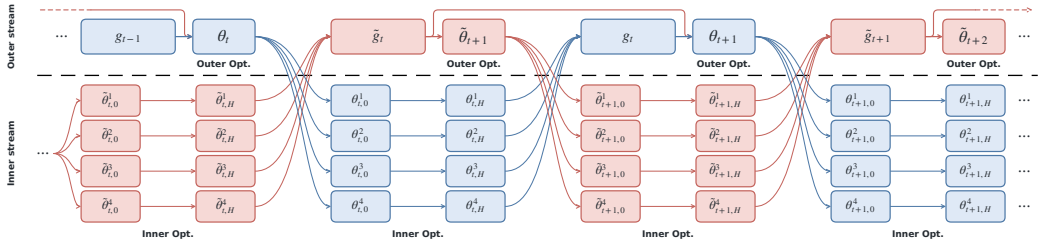


Figure 2: **Overview of LACCO.** Each outer iteration alternates a lookahead stage (red) and a main stage (blue), each consisting of H inner steps on $K = 4$ replicas. While one stage performs its local steps, communication of the previous stage's outer gradient is overlapped with this computation.

3.4 Convergence Analysis

In this section, for the sake of simplicity and without losing in generality, we establish the convergence of LACCO when both $\mathrm{InnerOpt}_{\mathrm{M}}$, $\mathrm{InnerOpt}_{\mathrm{L}}$ and $\mathrm{OuterOpt}_{\mathrm{M}}$, $\mathrm{OuterOpt}_{\mathrm{L}}$ are standard SGD. Let K denote the number of replicas. For each $h = 0, \ldots, H-1$ of iteration t, each replica i draws samples $\xi^i_{t,h}, \tilde{\xi}^i_{t,h} \sim \mathcal{D}$ and performs the following local SGD updates for $\beta > 0$

$$\begin{aligned} \theta^i_{t,0} &= \theta_t, & \theta^i_{t,h+1} &= \theta^i_{t,h} - \frac{\beta}{H}\nabla F(\theta^i_{t,h}; \xi^i_{t,h}), \\ \tilde{\theta}^i_{t,0} &= \tilde{\theta}_t, & \tilde{\theta}^i_{t,h+1} &= \tilde{\theta}^i_{t,h} - \frac{\beta}{H}\nabla F(\tilde{\theta}^i_{t,h}; \tilde{\xi}^i_{t,h}), \end{aligned} \qquad h = 0, \ldots, H-1. \tag{11}$$

Let η denote the outer learning rate. Then, the pair $\tilde{\theta}_t, \theta_t$ is updated as

$$\text{(i)} \quad \begin{cases} \tilde{g}^i_t = \tilde{\theta}_t - \tilde{\theta}^i_{t,H}, \\ \tilde{g}_t = \dfrac{1}{K}\displaystyle\sum_{i=1}^{K} \tilde{g}^i_t, \\ \tilde{\theta}_{t+1} = \theta_t - \eta \tilde{g}_t, \end{cases} \qquad \text{(ii)} \quad \begin{cases} g^i_t = \theta_t - \theta^i_{t,H}, \\ g_t = \dfrac{1}{K}\displaystyle\sum_{i=1}^{K} g^i_t, \\ \theta_{t+1} = \theta_t - \dfrac{\eta}{2}(g_t + \tilde{g}_t). \end{cases} \tag{12}$$

We make the following assumptions.

Assumption 1. *The objective $f : \mathbb{R}^d \to \mathbb{R}$ is bounded below by $f^\star$ and L-smooth, namely $f^\star := \inf_{\theta \in \mathbb{R}^d} f(\theta) > -\infty$ and $\|\nabla f(\theta) - \nabla f(\theta')\| \le L\|\theta - \theta'\|$ for all $\theta, \theta' \in \mathbb{R}^d$.*

Assumption 2. *Let $\mathcal{F}_t$ denote the filtration generated by all randomness preceding iteration t. Conditionally on $\mathcal{F}_t$, the local sample sequences $(\xi^i_{t,0}, \ldots, \xi^i_{t,H-1})$, $i = 1, \ldots, K$, are i.i.d.*

Assumption 3. *Let $\mathcal{F}_{t,h}$ be defined as in Assumption 2. The stochastic gradients satisfy $\mathbb{E}[\nabla F(\theta^i_{t,h}; \xi^i_{t,h}) \mid \mathcal{F}_{t,h}] = \nabla f(\theta^i_{t,h})$ and $\mathbb{E}[\|\nabla F(\theta^i_{t,h}; \xi^i_{t,h}) - \nabla f(\theta^i_{t,h})\|^2 \mid \mathcal{F}_{t,h}] \le \sigma^2$.*

Altogether, using standard Local SGD lemmas proved in Appendix C, we can prove:

Theorem 1. *Suppose Assumptions 1, 2 and 3 hold. Let $H \geq 1$ be the number of local SGD steps performed by K replicas. Let $\beta, \eta > 0$ be as in equation 11 and equation 12. Assume that*

$$0 < \beta L \leq \frac{1}{8\sqrt{2}}, \qquad \eta\beta L \leq \frac{1}{2}. \tag{13}$$

If $\theta_0 = \tilde{\theta}_0$, then, for every $T \geq 1$, we have

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\Big[\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\Big] \leq \frac{16\mathcal{V}_0}{3\eta\beta T} + \frac{512}{3}\frac{\beta L\sigma^2}{H} + \frac{512}{3}\frac{\eta\beta L\sigma^2}{KH}. \tag{14}$$

The key idea underlying the proof is that the following function serves as a Lyapunov function for our dynamics adapted to Local SGD, by chosing κ small enough,

$$\mathcal{V}_t = f(\theta_t) - f^\star + f(\tilde{\theta}_t) - f^\star + \kappa\|\theta_t - \tilde{\theta}_t\|^2 \geq 0. \tag{15}$$

4 Experiments

Our experiments use Llama 3-style (Dubey et al., 2024) decoder-only Transformers (Vaswani et al., 2017) with 134M and 325M parameters. We use the Llama 2 (7B) tokenizer (Touvron et al., 2023) with a vocabulary size of 32,000 and a sequence length of 2,048 tokens. Following standard practice (Hoffmann et al., 2022), each model is trained on a token budget equal to 20× its parameter count. We train on FineWeb (Penedo et al., 2024) and validate on a disjoint held-out split of 65.5M tokens. For methods where replicas are not synchronized, we average them before validation, while for LACCO we validate on θ_t. Moreover, we perform zero-shot downstream tasks on classic benchmarks. Whereas Ajanthan et al. (2026) use $K = 2$ replicas, we use $K \geq 4$ replicas with a global batch size of 131,072 tokens per inner step. Architectural details, hyperparameter choices, and implementation details, all built on TorchFT (Meta PyTorch, 2026), are provided in Appendices B.1, B.2, and B.3, respectively. We include DDP and DiLoCo as synchronous references and overlapping baselines use the same amount of computation and communication.

4.1 Main Results

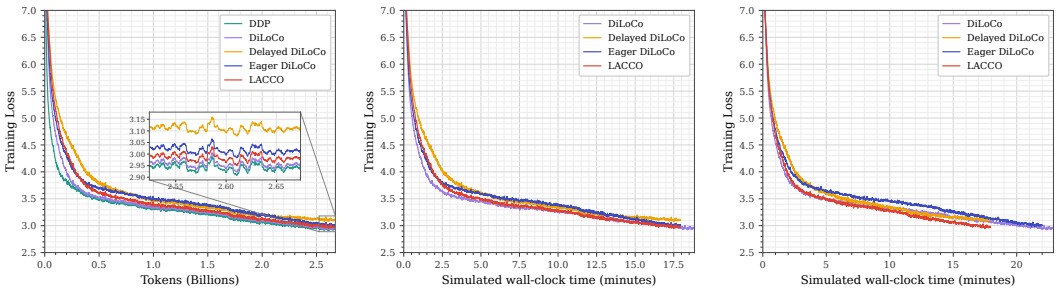
LACCO evaluation on 134M model size with $K = 4$ and $H = 30$. Table 2 compares LACCO with both synchronous baselines and methods that overlap communication with local computation. Thanks to its two-stage mechanism, LACCO achieves the lowest evaluation loss among communication-overlapping methods, and furthermore, LACCO remains within 1.77% of DDP and 0.91% of DiLoCo. It also outperforms the overlapping baselines on HellaSwag and PIQA, and is close on ARC-Easy.

Table 2: **134M evaluation, $K = 4$ and $H = 30$.** Comparison of potentially overlapping baselines.

Method	Overlap	Eval loss ↓	HellaSwag ↑	PIQA ↑	ARC-Easy ↑
DDP	✗	2.936	28.76	59.79	38.80
DiLoCo	✗	2.961	29.06	60.50	36.91
LACCO	✓	**2.988**	**28.87**	**61.10**	37.04
Eager DiLoCo	✓	3.020	28.77	59.74	**37.50**
Delayed DiLoCo	✓	3.117	27.81	58.98	35.98

Simulated wall-clock time. Figure 3 reports training loss against tokens and simulated wall-clock time using the timing model detailed in Appendix D. We assume a pessimistic inter-replica bandwidth of $b_{\text{net}} = 10\,\text{GB/s}$ and CPU–GPU bandwidth of $b_{\text{CPU/GPU}} = 3\,\text{GB/s}$. Let T_{inner}, T_{comm}, and T_{outer} denote the time spent on local computation, outer-gradient communication, and the outer update, respectively. Without overlap, the time per round is $T_{\text{serial}} = T_{\text{inner}} + T_{\text{comm}} + T_{\text{outer}}$. If communication is overlapped with local computation but the outer update remains serialized,

$$T_{\text{partial}} = \max\{T_{\text{inner}}, T_{\text{comm}}\} + T_{\text{outer}}.$$



(a) Measured loss versus tokens. (b) OuterOpt state on GPU. (c) OuterOpt state offloaded.

Figure 3: **Training loss versus tokens and simulated wall-clock time.** (a) Measured loss versus tokens. (b)–(c) The same losses against simulated time (H100, 50% MFU, $b_{\text{net}} = 10\,\text{GB/s}$, $b_{\text{CPU/GPU}} = 3\,\text{GB/s}$), with the OuterOpt state kept on GPU (b) or offloaded to CPU (c).

If the complete outer path is overlapped with local computation,

$$T_{\text{overlap}} = \max\{T_{\text{inner}}, T_{\text{comm}} + T_{\text{outer}}\}.$$

Hence, communication and outer-update costs are fully hidden whenever $T_{\text{inner}} \geq T_{\text{comm}} + T_{\text{outer}}$. For the 134M model with $K = 4$, $H = 30$, and 131,072 tokens per inner step, we consider an H100 (details in Appendix D). This gives $T_{\text{inner}} \simeq 1.58\,\text{s}$, and $T_{\text{comm}} \simeq 0.08\,\text{s}$. For LACCO, the additional optimizer-state transfers amount to an estimated $T_{\text{outer}} \simeq 0.89\,\text{s}$, whereas it is $T_{\text{outer}} \simeq 0.36\,\text{s}$ for the other methods. The complete outer path is therefore hidden by the local computation at $H = 30$: under these assumptions, full overlap requires only $H \geq 19$, as detailed in Appendix D. This threshold is independent of the parameter count P under our simplified model: both the local-computation cost and the communicated/offloaded volumes scale linearly with P. This cancellation assumes fixed MFU and effective bandwidths and neglects the comparatively small optimizer kernel time. All model computation is assumed to execute on the GPU.

4.2 Ablations

4.2.1 Varying the model size

Table 3: **Model size evaluation, $K = 4$ and $H = 30$.** Comparison across $134M$ and $325M$ models. LACCO consistently outperforms other overlapping methods and is on par with DiLoCo.

Scale	Method	Overlap	Eval loss ↓	HellaSwag ↑	PIQA ↑	ARC-Easy ↑
134M	DDP	✗	2.936	28.76	59.79	38.80
	DiLoCo	✗	2.961	29.06	60.50	36.91
	LACCO	✓	**2.988**	**28.87**	**61.10**	37.04
	Eager DiLoCo	✓	3.020	28.77	59.74	**37.50**
	Delayed DiLoCo	✓	3.117	27.81	58.98	35.98
325M	DDP	✗	2.673	33.90	64.80	42.76
	DiLoCo	✗	2.692	34.27	64.25	41.20
	LACCO	✓	**2.696**	**34.10**	**64.36**	40.57
	Eager DiLoCo	✓	2.721	33.74	63.98	**41.20**
	Delayed DiLoCo	✓	2.795	31.39	63.11	39.44

We train and evaluate all methods at 134M and 325M parameters using $H = 30$ and $K = 4$, and we tune the baselines according to Appendix B.2. In Table 3, we observe that LACCO achieves the lowest evaluation loss among communication-overlapping methods at every model size, demonstrating consistent scaling. At the larger 325M scale, LACCO achieves a validation loss only approximately 0.86% higher than DDP while communicating (i) $30\times$ less frequently and (ii) fully hiding communication latency, with (iii) comparable downstream accuracy.

Table 4: **Evaluation of the inner step size H, with model size 134M and $K = 4$.** DiLoCo, LACCO, and Eager DiLoCo all show a U-shaped trend in H, as observed in (Kale et al., 2025). Only Delayed DiLoCo does not exhibit this trend, which is consistent with its delayed update dynamics.

Method	$H = 5$	$H = 10$	$H = 15$	$H = 30$	$H = 45$	$H = 60$	$H = 75$
DiLoCo	3.034	2.959	2.960	2.961	2.984	3.000	3.016
LACCO	3.077	**2.960**	**2.959**	**3.001**	**3.039**	**3.071**	**3.103**
Eager DiLoCo	4.370	3.170	3.068	3.020	3.053	3.079	3.114
Delayed DiLoCo	**3.040**	3.061	3.074	3.117	3.150	3.176	3.201

4.2.2 Varying the inner steps H

We vary H from 5 to 75 on the 134M model with $K = 4$ (Table 4) and keep the other hyperparameters constant. LACCO achieves the lowest loss among overlapping methods for every $H > 5$, and performs comparably to synchronous DiLoCo for $H = 10$ and $H = 15$. At larger H, the gap to DiLoCo widens, possibly because θ_t and $\tilde{\theta}_t$ drift further apart over longer local phases.

4.2.3 Varying the number of replicas K

Table 5: **Evaluation of K, for 134M scale and $H = 30$.** Loss degrades with K for all methods, with $K > 8$ known to be a critical regime for DiLoCo (Douillard et al., 2023). Nonetheless, LACCO remains within 1% of DiLoCo at every K, degrading more gracefully than other overlapped methods.

Method	$K = 2$	$K = 4$	$K = 8$	$K = 16$
DiLoCo	2.941	2.961	3.029	3.143
LACCO	2.953	**2.988**	**3.054**	**3.154**
Eager DiLoCo	**2.946**	3.020	3.183	3.445
Delayed DiLoCo	3.027	3.117	3.209	3.353

We vary K from 2 to 16 on the 134M model (by keeping the global batch size fixed) with $H = 30$ (Table 5) and keep the other hyperparameters constant. LACCO remains within 1% of DiLoCo for all K and achieves the lowest loss among overlapping methods for $K \geq 4$. In contrast, Eager DiLoCo degrades markedly as K grows, from 3.020 at $K = 4$ to 3.445 at $K = 16$, compared with 2.988 to 3.154 for LACCO. Eager DiLoCo is slightly better at $K = 2$: with a single replica, local and averaged outer gradients coincide, and with two replicas each local outer gradient still accounts for half of the average. As K increases, the discrepancy between replica-local and global updates grows, whereas the shared updates of LACCO keep all replicas consistent. Consistently with Equation 6, the gap between Eager and Delayed DiLoCo vanishes as K grows.

5 Conclusion

We introduced LACCO, a two-stage local optimization algorithm that fully overlaps outer communication and the outer update with local computation. Unlike existing overlap methods, LACCO applies the same globally averaged update to every replica and computes each outer update from the current pair of models, thereby preserving replica consistency without introducing staleness or local synchronization. We established convergence guarantees for smooth objectives and showed that, among communication-overlapping methods, LACCO achieves the lowest evaluation loss across model sizes, communication intervals, and replica counts, while remaining close to synchronous DiLoCo and scaling markedly better than Eager DiLoCo as the number of replicas grows.

6 AI disclosure

In this work, we used generative AI tools to polish the manuscript and assist with code review and debugging. All AI-assisted edits and code changes were reviewed by the authors. We take responsibility for the final content of this work, including all text, claims, and code produced with the assistance of generative AI.

REFERENCES

Alekh Agarwal and John C. Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems*, volume 24, pp. 873–881, 2011.

Thalaiyasingam Ajanthan, Sameera Ramasinghe, Gil Avraham, Yan Zuo, and Alexander Long. Momentum look-ahead for asynchronous distributed low-communication training. In *ICLR Workshop on Modularity for Collaborative, Decentralized, and Continual Deep Learning*, 2025. URL https://openreview.net/forum?id=4O8nzTkHPI.

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, 2026. URL https://arxiv.org/abs/2601.22442.

Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. In *Conference on Empirical Methods in Natural Language Processing*, pp. 440–445, 2017. doi: 10.18653/v1/D17-1045.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, 2017.

Saar Barkai, Ido Hakimi, and Assaf Schuster. Gap aware mitigation of gradient staleness. *arXiv preprint arXiv:1909.10802*, 2019.

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Anima Anandkumar. signSGD: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, 2018.

Pietro Cagnasso, Eugene Belilovsky, and Edouard Oyallon. Unifying local communications and local updates for llm pretraining, 2026. URL https://arxiv.org/abs/2606.11081.

Zachary Charles, Gabriel Teston, Lucio Dery, Keith Rush, Nova Fallen, Zachary Garrett, Arthur Szlam, and Arthur Douillard. Communication-efficient language model training scales reliably and robustly: Scaling laws for diloco. In *Advances in Neural Information Processing Systems*, 2025.

Jeffrey Dean, Greg S. Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, Quoc V. Le, and Andrew Y. Ng. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

Decoupled DiLoCo Team. Decoupled DiLoCo for resilient distributed pre-training. *arXiv preprint arXiv:2604.21428*, 2026.

Michael Diskin, Alexey Bukhtiyarov, Max Ryabinin, Lucile Saulnier, Quentin Lhoest, Anton Sinitsin, Dmitry Popov, Dmitry Pyrkin, Maxim Kashirin, Alexander Borzunov, et al. Distributed deep learning in open collaborations. *arXiv preprint arXiv:2106.10207*, 2021.

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: Towards a distributed free lunch. In *Conference on Language Modeling*, 2025. URL https://arxiv.org/abs/2501.18512.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Ido Hakimi, Saar Barkai, Moshe Gabel, and Assaf Schuster. Taming momentum in a distributed asynchronous environment. *arXiv preprint arXiv:1907.11612*, 2019.

Chamin Hewa Koneputugodage et al. Factored gossip DiLoCo: Reducing blocking communication in DiLoCo. *arXiv preprint arXiv:2606.22768*, 2026. URL https://arxiv.org/abs/2606.22768.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Paul Janson, Edouard Oyallon, and Eugene Belilovsky. Stabilizing native low-rank llm pretraining. *arXiv preprint arXiv:2602.12429*, 2026.

Sylvain Jeaugey. NCCL 2.0. GPU Technology Conference, 2017. URL https://developer.nvidia.com/nccl.

Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, Dipankar Das, Kunal Banerjee, Sasikanth Avancha, Dharma Teja Vooturi, Nataraj Jammalamadaka, Jianyu Huang, Hector Yuen, et al. A study of BFLOAT16 for deep learning training. *arXiv preprint arXiv:1905.12322*, 2019. URL https://arxiv.org/abs/1905.12322.

Satyen Kale, Arthur Douillard, and Yanislav Donchev. Eager updates for overlapped communication and computation in DiLoCo. *arXiv preprint arXiv:2502.12996*, 2025. URL https://arxiv.org/abs/2502.12996.

Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian U. Stich. A unified theory of decentralized SGD with changing topology and local updates. In *International Conference on Machine Learning*, 2020.

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, 2020. doi: 10.14778/3415478.3415530.

Xiangru Lian, Yijun Huang, Yuncheng Li, and Ji Liu. Asynchronous parallel stochastic gradient for nonconvex optimization. In *Advances in Neural Information Processing Systems*, 2015.

Wanchao Liang, Tianyu Liu, Less Wright, Will Constable, Andrew Gu, Chien-Chin Huang, Iris Zhang, Wei Feng, Howard Huang, Junjie Wang, et al. TorchTitan: One-stop PyTorch native solution for production-ready LLM pre-training. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=WuQtmIkiUL.

Tao Lin, Sebastian U. Stich, Kumar Kshitij Patel, and Martin Jaggi. Don't use large mini-batches, use local SGD. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=B1eyO1BFPr.

Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SkhQHMW0W.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

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*, 2024b.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

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

H. 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. PMLR, 2017. URL https://proceedings.mlr.press/v54/mcmahan17a.html.

Meta PyTorch. TorchFT: Fault tolerance for PyTorch, 2026. URL https://github.com/meta-pytorch/torchft. Software repository; accessed 22 July 2026.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1gs9JgRZ.

Konstantin Mishchenko, Francis Bach, Mathieu Even, and Blake Woodworth. Asynchronous SGD beats minibatch SGD under arbitrary delays. In *Advances in Neural Information Processing Systems*, 2022.

Adel Nabli and Edouard Oyallon. DADAO: Decoupled accelerated decentralized asynchronous optimization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25604–25626. PMLR, 2023. URL https://proceedings.mlr.press/v202/nabli23a.html.

Adel Nabli and Edouard Oyallon. Decentralized asynchronous optimization with dadao allows decoupling and acceleration. *Journal of Machine Learning Research*, 26(207):1–48, 2025.

Adel Nabli, Louis Fournier, Pierre Erbacher, Louis Serrano, Eugene Belilovsky, and Edouard Oyallon. Acco: Accumulate while you communicate for communication-overlapped sharded llm training, 2025. URL https://arxiv.org/abs/2406.02613.

Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Anand Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. Efficient large-scale language model training on GPU clusters using Megatron-LM. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–15, 2021. doi: 10.1145/3458817.3476209.

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

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 *The Thirty-eighth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=n6SCkn2QaG.

PyTorch Team. Pytorch distributed data parallel: Gradient bucketing and communication–computation overlap. PyTorch Blog, 2020. URL https://pytorch.org/blog/efficient-pytorch-ddp/. Accessed 2026-07-22.

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

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H. Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=LkFG3lB13U5.

Jie Ren, Samyam Rajbhandari, Reza Yazdani Aminabadi, Olatunji Ruwase, Shuangyan Yang, Minjia Zhang, Dong Li, and Yuxiong He. Zero-offload: Democratizing billion-scale model training. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pp. 551–564. USENIX Association, 2021. URL https://www.usenix.org/conference/atc21/presentation/ren-jie.

Stéphane Rivaud, Louis Fournier, Thomas Pumir, Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Petra: Parallel end-to-end training with reversible architectures. In *International Conference on Learning Representations*, volume 2025, pp. 80162–80178, 2025.

Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs. In *Interspeech*, pp. 1058–1062, 2014. doi: 10.21437/Interspeech.2014-274.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-LM: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019. URL https://arxiv.org/abs/1909.08053.

Vaibhav Singh, Zafir Khalid, Pietro Cagnasso, Edouard Oyallon, and Eugene Belilovsky. Model parallelism with subnetwork data parallelism. *arXiv preprint arXiv:2507.09029*, 2025.

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

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, and Bo Wen. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

Weigao Sun, Zhen Qin, Weixuan Sun, Shidi Li, Dong Li, Xuyang Shen, Yu Qiao, and Yiran Zhong. CO2: Efficient distributed training with full communication–computation overlap. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ZO5cn4IfaN.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1139–1147. PMLR, 2013. URL https://proceedings.mlr.press/v28/sutskever13.html.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Jianyu Wang, Hao Liang, and Gauri Joshi. Overlap local-SGD: An algorithmic approach to hide communication delays in distributed SGD. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8871–8875, 2020. doi: 10.1109/ICASSP40776.2020.9053552.

Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. Gradient sparsification for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems*, 2018.

Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*, 2019.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, 2019.

Muru Zhang, Mayank Mishra, Zhongzhu Zhou, William Brandon, Jue Wang, Yoon Kim, Jonathan Ragan-Kelley, Shuaiwen Leon Song, Ben Athiwaratkun, and Tri Dao. Ladder-residual: parallelism-aware architecture for accelerating large model inference with communication overlapping. *arXiv preprint arXiv:2501.06589*, 2025.

Sixin Zhang, Cong Zhang, Zhao You, Rong Zheng, and Bo Xu. Asynchronous stochastic gradient descent for DNN training. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6660–6663. IEEE, 2013.

Shuxin Zheng, Qi Meng, Taifeng Wang, Wei Chen, Nenghai Yu, Zhi-Ming Ma, and Tie-Yan Liu. Asynchronous stochastic gradient descent with delay compensation. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 4120–4129. PMLR, 2017. URL https://proceedings.mlr.press/v70/zheng17b.html.

A ALGORITHMIC VIEW

Algorithm 1 LACCO

Inputs: number of outer iterations T, inner steps H, number of replicas K
Inputs: initial models $\theta_0 = \tilde{\theta}_0$, initial $\tilde{g}_0 = 0$
Inputs: optimizers $\mathrm{InnerOpt}_{\mathrm{M}}$, $\mathrm{InnerOpt}_{\mathrm{L}}$, $\mathrm{OuterOpt}_{\mathrm{M}}$, $\mathrm{OuterOpt}_{\mathrm{L}}$
Inputs: data shards $\mathcal{D}_1, \ldots, \mathcal{D}_K$
function INNERSTEPS($\vartheta_0^i, \mathcal{D}_i, H, \mathrm{Opt}$)
  for $h = 0, \ldots, H-1$ **do**
    $\xi_h^i \sim \mathcal{D}_i$
    $\vartheta_{h+1}^i \leftarrow \mathrm{Opt}\big(\vartheta_h^i, \nabla F(\vartheta_h^i; \xi_h^i)\big)$
  end for
  return ϑ_H^i
end function
for $t = 0, \ldots, T-1$ **do**
  for all $i \in \{1, \ldots, K\}$ **in parallel do**
  Lookahead stage
    $\theta_{t,H}^i \leftarrow$ INNERSTEPS$(\theta_t, \mathcal{D}_i, H, \mathrm{InnerOpt}_{\mathrm{M}})$
    $g_t^i \leftarrow \theta_t - \theta_{t,H}^i$
    $g_t \leftarrow$ ASYNC-ALLREDUCE(g_t^i)
    WAIT$(\tilde{g}_t)$
    $\tilde{\theta}_{t+1} \leftarrow \mathrm{OuterOpt}_{\mathrm{L}}(\theta_t, \tilde{g}_t)$
  Main stage
    $\tilde{\theta}_{t+1,H}^i \leftarrow$ INNERSTEPS$(\tilde{\theta}_{t+1}, \mathcal{D}_i, H, \mathrm{InnerOpt}_{\mathrm{L}})$
    $\tilde{g}_{t+1}^i \leftarrow \tilde{\theta}_{t+1} - \tilde{\theta}_{t+1,H}^i$
    $\tilde{g}_{t+1} \leftarrow$ ASYNC-ALLREDUCE$(\tilde{g}_{t+1}^i)$
    WAIT(g_t)
    $\theta_{t+1} \leftarrow \mathrm{OuterOpt}_{\mathrm{M}}\big(\theta_t, \frac{1}{2}(g_t + \tilde{g}_t)\big)$
  end for
end for

B EXPERIMENTAL DETAILS

B.1 MODEL ARCHITECTURE

Our experiments use Llama 3-style (Dubey et al., 2024) decoder-only Transformers (Vaswani et al., 2017) with RMSNorm (Zhang & Sennrich, 2019) pre-normalization, SwiGLU feed-forward networks (Shazeer, 2020), and rotary positional embeddings (RoPE) (Su et al., 2021) with base $\theta = 10{,}000$. Attention uses standard multi-head attention with head dimension 64. We use the Llama 2 (7B) tokenizer (Touvron et al., 2023) with a vocabulary size of 32,000 and a sequence length of 2,048 tokens. Following the Chinchilla scaling laws (Hoffmann et al., 2022), each model is trained on a token budget equal to $20\times$ its parameter count. All models are trained from scratch, with weights initialized following Dubey et al. (2024). Details of the architectures used at each scale are reported in Table 6.

Table 6: **Model architecture.** We report, for each scale, the total parameter count, the number of Transformer layers, the number of attention heads, the embedding dimension, the per-head dimension, the SwiGLU feed-forward hidden dimension, and the Chinchilla token budget.

Model parameters	Transformer layers	Attention heads	Embedding dimension	Head dimension	FFN hidden dimension	Token budget
134M	12	12	768	64	2048	2.7B
325M	20	16	1024	64	2816	6.5B

B.2 Hyperparameter Tuning

We use AdamW (Loshchilov & Hutter, 2019) as the inner optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.95$. The inner learning rate follows a cosine decay schedule (Loshchilov & Hutter, 2017) with a 10% linear warm-up (Goyal et al., 2017). Weight decay is set to 0.1. The outer optimizer is Nesterov SGD (Nesterov, 1983; Sutskever et al., 2013). For each method, all hyperparameters were tuned using $H = 30$ communication interval, $K = 4$ replicas, and a global batch size of $B = 64$ on the 134M model. When scaling to the 325M, the outer learning rate and momentum are kept fixed for each method (Charles et al., 2025). For DDP, the inner learning rate η_{in} was swept over $\eta_{\text{in}} \in \{1\text{e}{-3}, 2\text{e}{-3}, 3\text{e}{-3}, 4\text{e}{-3}, 5\text{e}{-3}, 6\text{e}{-3}\}$. For all the other methods, the inner learning rate η_{in} was swept over $\{6\text{e}{-4}, 7\text{e}{-4}, 8\text{e}{-4}, 1\text{e}{-3}, 2\text{e}{-3}, 3\text{e}{-3}\}$, the outer learning rate η_{out} over $\{0.2, 0.3, 0.4, 0.5, 0.6, 0.7\}$, and the momentum over $\mu \in \{0.7, 0.8, 0.9\}$. For Delayed DiLoCo, the outer learning rate was selected by lowering by $\times 4$ the optimal outer learning found for Eager DiLoCo (Kale et al., 2025). Given the two-stage step mechanism of LACCO, two inner learning rates η_{in} and $\tilde{\eta}_{in}$ as well as two outer learning rates η_{out} and $\tilde{\eta}_{out}$ should be all tuned. For simplicity, we set $\eta_{in} = \tilde{\eta}_{in}$ and $\eta_{out} = \tilde{\eta}_{out}$. Table 7 summarizes the values that were selected for each method.

Table 7: **Selected hyperparameters across model sizes.**

Method	(a) 134M model			(b) 325M model		
	η_{in}	η_{out}	μ	η_{in}	η_{out}	μ
DDP	$5\text{e}{-3}$	—	—	$2\text{e}{-3}$	—	—
DiLoCo	$1\text{e}{-3}$	0.7	0.9	$7\text{e}{-4}$	0.7	0.9
LACCO	$2\text{e}{-3}$	0.7	0.9	$8\text{e}{-4}$	0.7	0.9
Eager DiLoCo	$2\text{e}{-3}$	0.4	0.9	$8\text{e}{-4}$	0.4	0.9
Delayed DiLoCo	$2\text{e}{-3}$	0.1	0.9	$8\text{e}{-4}$	0.1	0.9

B.3 Implementation

All experiments were conducted using PyTorch 2.13.0 (Paszke et al., 2019) with CUDA 12.6, cuDNN 9.10.2, building on the TorchTitan (Liang et al., 2025) and TorchFT codebases. Replica communication relies on NCCL 2.29.3 collectives (Jeaugey, 2017), launched asynchronously to overlap communication with local computation. Models are trained in mixed precision (Micikevicius et al., 2018; Kalamkar et al., 2019) on NVIDIA A100 and H100 GPUs.

C Proofs

This section collects the proofs of the main technical results.

C.1 Auxiliary lemmas

Lemma 1 (Gradient Mismatch)**.** *Let f be L-smooth. Fix $H \geq 1$ and $0 < \beta L \leq 1$. Define, for every replica $i = 1, \ldots, K$, the local deterministic trajectory*

$$\theta^i_{t,0} = \theta_t, \qquad \theta^i_{t,h+1} = \theta^i_{t,h} - \frac{\beta}{H} \nabla f(\theta^i_{t,h}), \qquad h = 0, \ldots, H-1,$$

and the accumulated local updates

$$g^i_t := \theta_t - \theta^i_{t,H}.$$

Then

$$\begin{aligned} \left\| g^i_t - \beta \nabla f(\theta_t) \right\| &\leq \frac{\left(1 + \frac{\beta L}{H}\right)^H - 1 - \beta L}{L} \|\nabla f(\theta_t)\| \\ &\leq \beta^2 L \|\nabla f(\theta_t)\|. \end{aligned}$$

Proof. First note that

$$g_t^i = \frac{\beta}{H} \sum_{h=0}^{H-1} \nabla f(\theta_{t,h}^i).$$

Hence

$$g_t^i - \beta \nabla f(\theta_t) = \frac{\beta}{H} \sum_{h=0}^{H-1} \big(\nabla f(\theta_{t,h}^i) - \nabla f(\theta_t)\big).$$

By L-smoothness,

$$\big\|g_t^i - \beta \nabla f(\theta_t)\big\| \le \frac{\beta L}{H} \sum_{h=0}^{H-1} \|\theta_{t,h}^i - \theta_t\|.$$

We claim that

$$\|\theta_{t,h}^i - \theta_t\| \le \frac{\left(1 + \frac{\beta L}{H}\right)^h - 1}{L} \|\nabla f(\theta_t)\|.$$

Indeed, using the update rule and L-smoothness,

$$\|\theta_{t,h+1}^i - \theta_t\| \le \left(1 + \frac{\beta L}{H}\right) \|\theta_{t,h}^i - \theta_t\| + \frac{\beta}{H} \|\nabla f(\theta_t)\|.$$

Since $\theta_{t,0}^i = \theta_t$, iterating this recursion gives the claim. Therefore,

$$\begin{aligned} \big\|g_t^i - \beta \nabla f(\theta_t)\big\| &\le \frac{\beta}{H} \sum_{h=0}^{H-1} \left[\left(1 + \frac{\beta L}{H}\right)^h - 1\right] \|\nabla f(\theta_t)\| \\ &= \frac{\left(1 + \frac{\beta L}{H}\right)^H - 1 - \beta L}{L} \|\nabla f(\theta_t)\|. \end{aligned}$$

Finally, since $0 < \beta L \le 1$,

$$\left(1 + \frac{\beta L}{H}\right)^H \le e^{\beta L}$$

and

$$e^x - 1 - x \le x^2, \qquad 0 \le x \le 1.$$

Thus

$$\left(1 + \frac{\beta L}{H}\right)^H - 1 - \beta L \le (\beta L)^2,$$

which gives

$$\big\|g_t^i - \beta \nabla f(\theta_t)\big\| \le \beta^2 L \|\nabla f(\theta_t)\|.$$

□

Lemma 2 (Expected Stochastic Gradient Mismatch)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be L-smooth. Let $\mathcal{F}_{t,h}$ be the filtration generated by the randomness up to the h-th local step of outer iteration t. Assume that:*

(i) The stochastic gradient is unbiased:

$$\mathbb{E}\left[\nabla F(\theta_{t,h}^i; \xi_{t,h}^i) \mid \mathcal{F}_{t,h}\right] = \nabla f(\theta_{t,h}^i).$$

(ii) The conditional variance is uniformly bounded by σ^2:

$$\mathbb{E}\left[\big\|\nabla F(\theta_{t,h}^i; \xi_{t,h}^i) - \nabla f(\theta_{t,h}^i)\big\|^2 \mid \mathcal{F}_{t,h}\right] \le \sigma^2.$$

Fix $H \ge 1$ and $0 < \beta L \le 1$. For every replica $i = 1, \dots, K$, define the local stochastic trajectory by

$$\theta_{t,0}^i = \theta_t, \qquad \theta_{t,h+1}^i = \theta_{t,h}^i - \frac{\beta}{H} \nabla F(\theta_{t,h}^i; \xi_{t,h}^i), \qquad h = 0, \dots, H-1.$$

Define the accumulated local update

$$g_t^i := \theta_t - \theta_{t,H}^i = \frac{\beta}{H} \sum_{h=0}^{H-1} \nabla F(\theta_{t,h}^i; \xi_{t,h}^i).$$

Then, with

$$q := 1 + \frac{\beta L}{H},$$

we have

$$\left\| \mathbb{E}[g_t^i] - \beta \nabla f(\theta_t) \right\|^2 \leq \frac{2\beta^2}{H} \sum_{h=0}^{H-1} (q^h - 1)^2 \|\nabla f(\theta_t)\|^2 + \frac{2\beta^3 L \sigma^2}{H^2} \sum_{h=0}^{H-1} \frac{q^{2h} - 1}{2 + \frac{\beta L}{H}}.$$

Consequently,

$$\left\| \mathbb{E}[g_t^i] - \beta \nabla f(\theta_t) \right\|^2 \leq 8\beta^4 L^2 \|\nabla f(\theta_t)\|^2 + 8\frac{\beta^3 L \sigma^2}{H}.$$

Proof. Throughout the proof, all expectations are taken with respect to the local stochastic-gradient randomness, conditional on the current iterate θ_t.

Define the stochastic gradient noise

$$\varepsilon_{t,h}^i := \nabla F(\theta_{t,h}^i; \xi_{t,h}^i) - \nabla f(\theta_{t,h}^i).$$

By assumptions (i)–(ii),

$$\mathbb{E}[\varepsilon_{t,h}^i \mid \mathcal{F}_{t,h}] = 0, \qquad \mathbb{E}[\|\varepsilon_{t,h}^i\|^2 \mid \mathcal{F}_{t,h}] \leq \sigma^2.$$

Also define

$$\Delta_{t,h}^i := \theta_{t,h}^i - \theta_t.$$

Since $\theta_{t,0}^i = \theta_t$, we have

$$\Delta_{t,0}^i = 0.$$

Moreover, using the local update rule,

$$\Delta_{t,h+1}^i = \Delta_{t,h}^i - \frac{\beta}{H} \nabla f(\theta_{t,h}^i) - \frac{\beta}{H} \varepsilon_{t,h}^i.$$

Adding and subtracting $\nabla f(\theta_t)$ gives

$$\Delta_{t,h+1}^i = \Delta_{t,h}^i - \frac{\beta}{H} \big(\nabla f(\theta_{t,h}^i) - \nabla f(\theta_t) \big) - \frac{\beta}{H} \nabla f(\theta_t) - \frac{\beta}{H} \varepsilon_{t,h}^i.$$

Let

$$q := 1 + \frac{\beta L}{H}.$$

Introduce the noiseless comparison sequence $\bar{\Delta}_{t,h}^i$, defined by

$$\bar{\Delta}_{t,0}^i = 0,$$

and

$$\bar{\Delta}_{t,h+1}^i = \bar{\Delta}_{t,h}^i - \frac{\beta}{H} \big(\nabla f(\theta_t + \bar{\Delta}_{t,h}^i) - \nabla f(\theta_t) \big) - \frac{\beta}{H} \nabla f(\theta_t).$$

By L-smoothness,

$$\|\bar{\Delta}_{t,h+1}^i\| \leq q \|\bar{\Delta}_{t,h}^i\| + \frac{\beta}{H} \|\nabla f(\theta_t)\|.$$

Iterating this inequality yields

$$\|\bar{\Delta}_{t,h}^i\| \leq \frac{\|\nabla f(\theta_t)\|}{L} (q^h - 1).$$

Now consider

$$\Delta_{t,h}^i - \bar{\Delta}_{t,h}^i.$$

Using the two recursions above and L-smoothness,

$$\mathbb{E}\big[\|\Delta_{t,h+1}^i - \bar{\Delta}_{t,h+1}^i\|^2 \mid \mathcal{F}_{t,h}\big] \le q^2 \|\Delta_{t,h}^i - \bar{\Delta}_{t,h}^i\|^2 + \frac{\beta^2\sigma^2}{H^2}.$$

Iterating from $\Delta_{t,0}^i - \bar{\Delta}_{t,0}^i = 0$ gives

$$\mathbb{E}\big[\|\Delta_{t,h}^i - \bar{\Delta}_{t,h}^i\|^2\big] \le \frac{\beta^2\sigma^2}{H^2} \sum_{j=0}^{h-1} q^{2j}.$$

Since

$$\sum_{j=0}^{h-1} q^{2j} = \frac{q^{2h}-1}{q^2-1}$$

and

$$q^2 - 1 = \frac{\beta L}{H}\left(2 + \frac{\beta L}{H}\right),$$

we obtain

$$\mathbb{E}\big[\|\Delta_{t,h}^i - \bar{\Delta}_{t,h}^i\|^2\big] \le \frac{\beta\sigma^2}{HL} \frac{q^{2h}-1}{2+\frac{\beta L}{H}}.$$

Using

$$\Delta_{t,h}^i = \bar{\Delta}_{t,h}^i + \big(\Delta_{t,h}^i - \bar{\Delta}_{t,h}^i\big)$$

and

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

we get

$$\mathbb{E}\big[\|\Delta_{t,h}^i\|^2\big] \le \frac{2}{L^2}(q^h-1)^2\|\nabla f(\theta_t)\|^2 + \frac{2\beta\sigma^2}{HL}\frac{q^{2h}-1}{2+\frac{\beta L}{H}}.$$

Now, by definition,

$$g_t^i = \frac{\beta}{H}\sum_{h=0}^{H-1} \nabla F(\theta_{t,h}^i; \xi_{t,h}^i).$$

By the tower property and unbiasedness of the stochastic gradient,

$$\mathbb{E}[g_t^i] = \frac{\beta}{H}\sum_{h=0}^{H-1} \mathbb{E}[\nabla f(\theta_{t,h}^i)].$$

Therefore,

$$\mathbb{E}[g_t^i] - \beta\nabla f(\theta_t) = \frac{\beta}{H}\sum_{h=0}^{H-1} \mathbb{E}\big[\nabla f(\theta_{t,h}^i) - \nabla f(\theta_t)\big].$$

By Jensen's inequality, Cauchy–Schwarz, and L-smoothness,

$$\begin{aligned}
\big\|\mathbb{E}[g_t^i] - \beta\nabla f(\theta_t)\big\|^2 &= \left\|\frac{\beta}{H}\sum_{h=0}^{H-1}\mathbb{E}\big[\nabla f(\theta_{t,h}^i) - \nabla f(\theta_t)\big]\right\|^2 \\
&\le \frac{\beta^2}{H}\sum_{h=0}^{H-1}\big\|\mathbb{E}\big[\nabla f(\theta_{t,h}^i) - \nabla f(\theta_t)\big]\big\|^2 \\
&\le \frac{\beta^2}{H}\sum_{h=0}^{H-1}\mathbb{E}\big[\|\nabla f(\theta_{t,h}^i) - \nabla f(\theta_t)\|^2\big] \\
&\le \frac{\beta^2 L^2}{H}\sum_{h=0}^{H-1}\mathbb{E}\big[\|\Delta_{t,h}^i\|^2\big].
\end{aligned}$$

Substituting the previous estimate gives

$$\left\|\mathbb{E}[g_t^i] - \beta\nabla f(\theta_t)\right\|^2 \le \frac{2\beta^2}{H}\sum_{h=0}^{H-1}(q^h-1)^2\|\nabla f(\theta_t)\|^2 + \frac{2\beta^3 L\sigma^2}{H^2}\sum_{h=0}^{H-1}\frac{q^{2h}-1}{2+\frac{\beta L}{H}}.$$

Since $h \le H$ and $\beta L \le 1$,

$$q^h = \left(1+\frac{\beta L}{H}\right)^h \le e^{\beta L}.$$

Moreover,

$$e^{\beta L} - 1 \le (e-1)\beta L,$$

so

$$q^h - 1 \le (e-1)\beta L.$$

Also,

$$q^{2h} \le e^{2\beta L} \le e^2,$$

and therefore

$$\frac{q^{2h}-1}{2+\frac{\beta L}{H}} \le \frac{e^2-1}{2}.$$

Consequently,

$$\left\|\mathbb{E}[g_t^i] - \beta\nabla f(\theta_t)\right\|^2 \le 2(e-1)^2\beta^4 L^2\|\nabla f(\theta_t)\|^2 + (e^2-1)\frac{\beta^3 L\sigma^2}{H}.$$

Thus,

$$\left\|\mathbb{E}[g_t^i] - \beta\nabla f(\theta_t)\right\|^2 \le 2(e-1)^2\beta^4 L^2\|\nabla f(\theta_t)\|^2 + (e^2-1)\frac{\beta^3 L\sigma^2}{H}.$$

□

Lemma 3 (Averaged Stochastic Gradient Mismatch)**.** *Suppose that the assumptions and notation of Lemma 2 hold. Set $\mathcal{F}_t := \mathcal{F}_{t,0}$. Moreover, assume that, conditionally on $\mathcal{F}_t$, the local sample sequences*

$$(\xi_{t,0}^i, \dots, \xi_{t,H-1}^i), \qquad i = 1, \dots, K,$$

are i.i.d. across replicas.

Let

$$g_t := \frac{1}{K}\sum_{i=1}^K g_t^i.$$

Then

$$\|\mathbb{E}[g_t|\mathcal{F}_t] - \beta\nabla f(\theta_t)\|^2 \le 8\beta^4 L^2\|\nabla f(\theta_t)\|^2 + 8\frac{\beta^3 L\sigma^2}{H}.$$

Proof. By linearity of conditional expectation,

$$\mathbb{E}[g_t|\mathcal{F}_t] = \frac{1}{K}\sum_{i=1}^K \mathbb{E}\left[g_t^i\middle|\mathcal{F}_t\right].$$

Hence,

$$\mathbb{E}[g_t|\mathcal{F}_t] - \beta\nabla f(\theta_t) = \frac{1}{K}\sum_{i=1}^K \left(\mathbb{E}\left[g_t^i\middle|\mathcal{F}_t\right] - \beta\nabla f(\theta_t)\right).$$

Therefore, by Jensen's inequality,

$$\|\mathbb{E}[g_t|\mathcal{F}_t] - \beta\nabla f(\theta_t)\|^2 \le \frac{1}{K}\sum_{i=1}^{K}\left\|\mathbb{E}\big[g_t^i\big|\mathcal{F}_t\big] - \beta\nabla f(\theta_t)\right\|^2.$$

Since the replicas are conditionally identically distributed given $\mathcal{F}_t$, the terms in the sum are identical. Thus,

$$\|\mathbb{E}[g_t|\mathcal{F}_t] - \beta\nabla f(\theta_t)\|^2 = \left\|\mathbb{E}\big[g_t^i\big|\mathcal{F}_t\big] - \beta\nabla f(\theta_t)\right\|^2.$$

Applying Lemma 2 yields

$$\|\mathbb{E}[g_t|\mathcal{F}_t] - \beta\nabla f(\theta_t)\|^2 \le 8\beta^4L^2\|\nabla f(\theta_t)\|^2 + 8\frac{\beta^3L\sigma^2}{H}.$$

□

C.2 Main theorems

Theorem A (Deterministic Lyapunov Descent)**.** *Let $f:\mathbb{R}^d\to\mathbb{R}$ be L-smooth and bounded below by $f^\star$. Fix $H\ge 1$ and $\eta,\beta>0$ such that*

$$0<\beta L\le\frac{1}{4},\qquad \eta\beta L\le\frac{1}{2}.$$

For every replica $i=1,\dots,K$, define the deterministic local trajectories

$$\theta_{t,0}^i=\theta_t,\qquad \theta_{t,h+1}^i=\theta_{t,h}^i-\frac{\beta}{H}\nabla f(\theta_{t,h}^i),\qquad h=0,\dots,H-1,$$

and

$$\tilde\theta_{t,0}^i=\tilde\theta_t,\qquad \tilde\theta_{t,h+1}^i=\tilde\theta_{t,h}^i-\frac{\beta}{H}\nabla f(\tilde\theta_{t,h}^i),\qquad h=0,\dots,H-1.$$

Define the accumulated local updates

$$g_t^i:=\theta_t-\theta_{t,H}^i,\qquad \tilde g_t^i:=\tilde\theta_t-\tilde\theta_{t,H}^i,$$

and their replica averages

$$g_t=\frac{1}{K}\sum_{i=1}^{K}g_t^i,\qquad \tilde g_t=\frac{1}{K}\sum_{i=1}^{K}\tilde g_t^i.$$

Consider the outer updates

$$\theta_{t+1}=\theta_t-\frac{\eta}{2}(g_t+\tilde g_t),\qquad \tilde\theta_{t+1}=\theta_t-\eta\tilde g_t.$$

Set

$$\kappa:=\frac{3}{2\eta\beta},$$

and define

$$\mathcal{V}_t:=f(\theta_t)-f^\star+f(\tilde\theta_t)-f^\star+\kappa\|\theta_t-\tilde\theta_t\|^2.$$

Then

$$\mathcal{V}_{t+1}-\mathcal{V}_t\le-\frac{\eta\beta}{3}\left(\|\nabla f(\theta_t)\|^2+\|\nabla f(\tilde\theta_t)\|^2\right).$$

Moreover

$$\frac{1}{T}\sum_{t=0}^{T-1}\left(\|\nabla f(\theta_t)\|^2+\|\nabla f(\tilde\theta_t)\|^2\right)\le\frac{3\mathcal{V}_0}{\eta\beta T}.$$

Additionally, if $\theta_0=\tilde\theta_0$, then

$$\frac{1}{T}\sum_{t=0}^{T-1}\left(\|\nabla f(\theta_t)\|^2+\|\nabla f(\tilde\theta_t)\|^2\right)\le\frac{6\big(f(\theta_0)-f^\star\big)}{\eta\beta T}.$$

Proof. By L-smoothness and the update

$$\theta_{t+1} = \theta_t - \frac{\eta}{2}(g_t + \tilde{g}_t),$$

we have

$$f(\theta_{t+1}) - f(\theta_t) \le -\eta \left\langle \nabla f(\theta_t), \frac{g_t + \tilde{g}_t}{2} \right\rangle + \frac{L\eta^2}{2} \left\| \frac{g_t + \tilde{g}_t}{2} \right\|^2.$$

Using the identity

$$-\langle a, b\rangle = -\frac{\beta}{2}\|a\|^2 - \frac{1}{2\beta}\|b\|^2 + \frac{1}{2\beta}\|b - \beta a\|^2,$$

with

$$a = \nabla f(\theta_t), \qquad b = \frac{g_t + \tilde{g}_t}{2},$$

we obtain

$$\begin{aligned} f(\theta_{t+1}) - f(\theta_t) \le -\frac{\eta\beta}{2}\|\nabla f(\theta_t)\|^2 - \frac{\eta}{2\beta}\left(1 - \eta\beta L\right) \left\| \frac{g_t + \tilde{g}_t}{2} \right\|^2 \\ + \frac{\eta}{2\beta} \left\| \frac{g_t + \tilde{g}_t}{2} - \beta \nabla f(\theta_t) \right\|^2. \end{aligned}$$

Since $\eta\beta L \le 1$, the middle term is nonpositive and may be dropped. Moreover,

$$\begin{aligned} \frac{g_t + \tilde{g}_t}{2} - \beta\nabla f(\theta_t) = \frac{1}{2}\left(g_t - \beta\nabla f(\theta_t)\right) + \frac{1}{2}\left(\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right) \\ + \frac{\beta}{2}\left(\nabla f(\tilde{\theta}_t) - \nabla f(\theta_t)\right). \end{aligned}$$

Therefore, using $\|a + b + c\|^2 \le 3\|a\|^2 + 3\|b\|^2 + 3\|c\|^2$ and L-smoothness,

$$\begin{aligned} \left\| \frac{g_t + \tilde{g}_t}{2} - \beta\nabla f(\theta_t) \right\|^2 \le \frac{3}{4}\left\|g_t - \beta\nabla f(\theta_t)\right\|^2 + \frac{3}{4}\left\|\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right\|^2 \\ + \frac{3\beta^2 L^2}{4}\|\theta_t - \tilde{\theta}_t\|^2. \end{aligned}$$

Hence

$$\begin{aligned} f(\theta_{t+1}) - f(\theta_t) \le -\frac{\eta\beta}{2}\|\nabla f(\theta_t)\|^2 + \frac{3\eta}{8\beta}\left\|g_t - \beta\nabla f(\theta_t)\right\|^2 \\ + \frac{3\eta}{8\beta}\left\|\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right\|^2 + \frac{3\eta\beta L^2}{8}\|\theta_t - \tilde{\theta}_t\|^2. \end{aligned}$$

Next, by L-smoothness and the update

$$\tilde{\theta}_{t+1} = \theta_t - \eta\tilde{g}_t,$$

we have

$$\begin{aligned} f(\tilde{\theta}_{t+1}) - f(\tilde{\theta}_t) &\le \left\langle \nabla f(\tilde{\theta}_t), \theta_t - \tilde{\theta}_t - \eta\tilde{g}_t \right\rangle + \frac{L}{2}\left\|\theta_t - \tilde{\theta}_t - \eta\tilde{g}_t\right\|^2 \\ &= -\left\langle \nabla f(\tilde{\theta}_t), \eta\tilde{g}_t - \theta_t + \tilde{\theta}_t \right\rangle + \frac{L}{2}\left\|\eta\tilde{g}_t - \theta_t + \tilde{\theta}_t\right\|^2. \end{aligned}$$

Using the identity

$$-\langle a, b\rangle = -\frac{\gamma}{2}\|a\|^2 - \frac{1}{2\gamma}\|b\|^2 + \frac{1}{2\gamma}\|b - \gamma a\|^2,$$

with

$$a = \nabla f(\tilde{\theta}_t), \qquad b = \eta\tilde{g}_t - \theta_t + \tilde{\theta}_t, \qquad \gamma = \eta\beta,$$

gives

$$f(\tilde{\theta}_{t+1}) - f(\tilde{\theta}_t) \le -\frac{\eta\beta}{2}\|\nabla f(\tilde{\theta}_t)\|^2 - \frac{1}{2\eta\beta}\left(1-\eta\beta L\right)\left\|\eta\tilde{g}_t - \theta_t + \tilde{\theta}_t\right\|^2 + \frac{1}{2\eta\beta}\left\|\eta\tilde{g}_t - \theta_t + \tilde{\theta}_t - \eta\beta\nabla f(\tilde{\theta}_t)\right\|^2.$$

Since $\eta\beta L \le 1$, the middle term is nonpositive and may be dropped. Furthermore,

$$\eta\tilde{g}_t - \theta_t + \tilde{\theta}_t - \eta\beta\nabla f(\tilde{\theta}_t) = \eta\left(\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right) - (\theta_t - \tilde{\theta}_t).$$

Thus, by $\|a-b\|^2 \le 2\|a\|^2 + 2\|b\|^2$,

$$\left\|\eta\tilde{g}_t - \theta_t + \tilde{\theta}_t - \eta\beta\nabla f(\tilde{\theta}_t)\right\|^2 \le 2\eta^2\left\|\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right\|^2 + 2\|\theta_t - \tilde{\theta}_t\|^2.$$

Therefore,

$$f(\tilde{\theta}_{t+1}) - f(\tilde{\theta}_t) \le -\frac{\eta\beta}{2}\|\nabla f(\tilde{\theta}_t)\|^2 + \frac{\eta}{\beta}\left\|\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right\|^2 + \frac{1}{\eta\beta}\|\theta_t - \tilde{\theta}_t\|^2.$$

Finally, using the two outer updates,

$$\theta_{t+1} - \tilde{\theta}_{t+1} = \theta_t - \frac{\eta}{2}(g_t + \tilde{g}_t) - (\theta_t - \eta\tilde{g}_t) = \frac{\eta}{2}(\tilde{g}_t - g_t).$$

Hence

$$\|\theta_{t+1} - \tilde{\theta}_{t+1}\|^2 - \|\theta_t - \tilde{\theta}_t\|^2 = -\|\theta_t - \tilde{\theta}_t\|^2 + \frac{\eta^2}{4}\|g_t - \tilde{g}_t\|^2.$$

Moreover,

$$g_t - \tilde{g}_t = (g_t - \beta\nabla f(\theta_t)) - \left(\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right) + \beta\left(\nabla f(\theta_t) - \nabla f(\tilde{\theta}_t)\right).$$

Therefore, using $\|a+b+c\|^2 \le 3\|a\|^2 + 3\|b\|^2 + 3\|c\|^2$ and L-smoothness,

$$\|g_t - \tilde{g}_t\|^2 \le 3\left\|g_t - \beta\nabla f(\theta_t)\right\|^2 + 3\left\|\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right\|^2 + 3\beta^2L^2\|\theta_t - \tilde{\theta}_t\|^2.$$

Thus

$$\kappa\left(\|\theta_{t+1} - \tilde{\theta}_{t+1}\|^2 - \|\theta_t - \tilde{\theta}_t\|^2\right) \le -\kappa\left(1 - \frac{3\eta^2\beta^2L^2}{4}\right)\|\theta_t - \tilde{\theta}_t\|^2 + \frac{3\kappa\eta^2}{4}\left\|g_t - \beta\nabla f(\theta_t)\right\|^2 + \frac{3\kappa\eta^2}{4}\left\|\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right\|^2.$$

Adding the three estimates gives

$$\mathcal{V}_{t+1} - \mathcal{V}_t \le -\frac{\eta\beta}{2}\|\nabla f(\theta_t)\|^2 - \frac{\eta\beta}{2}\|\nabla f(\tilde{\theta}_t)\|^2 + \left(\frac{3\eta}{8\beta} + \frac{3\kappa\eta^2}{4}\right)\left\|g_t - \beta\nabla f(\theta_t)\right\|^2 + \left(\frac{11\eta}{8\beta} + \frac{3\kappa\eta^2}{4}\right)\left\|\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right\|^2 + \left[\frac{3\eta\beta L^2}{8} + \frac{1}{\eta\beta} - \kappa\left(1 - \frac{3\eta^2\beta^2L^2}{4}\right)\right]\|\theta_t - \tilde{\theta}_t\|^2.$$

For $\kappa = 3/(2\eta\beta)$, the coefficient of $\|\theta_t - \tilde{\theta}_t\|^2$ satisfies

$$\frac{3\eta\beta L^2}{8} + \frac{1}{\eta\beta} - \kappa\left(1 - \frac{3\eta^2\beta^2L^2}{4}\right) = \frac{1}{\eta\beta}\left[-\frac{1}{2} + \frac{3}{2}(\eta\beta L)^2\right] \le 0,$$

where the last inequality follows from $\eta\beta L \le 1/2$. Moreover,

$$\frac{3\eta}{8\beta} + \frac{3\kappa\eta^2}{4} = \frac{3\eta}{2\beta}, \qquad \frac{11\eta}{8\beta} + \frac{3\kappa\eta^2}{4} = \frac{5\eta}{2\beta}.$$

Consequently,

$$\mathcal{V}_{t+1} - \mathcal{V}_t \le -\frac{\eta\beta}{2}\|\nabla f(\theta_t)\|^2 - \frac{\eta\beta}{2}\|\nabla f(\tilde{\theta}_t)\|^2 + \frac{3\eta}{2\beta}\left\|g_t - \beta\nabla f(\theta_t)\right\|^2 + \frac{5\eta}{2\beta}\left\|\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\right\|^2.$$

By Lemma 1, for every replica i,

$$\|g_t^i - \beta\nabla f(\theta_t)\| \le \beta^2 L\|\nabla f(\theta_t)\|,$$

and

$$\|\tilde{g}_t^i - \beta\nabla f(\tilde{\theta}_t)\| \le \beta^2 L\|\nabla f(\tilde{\theta}_t)\|.$$

Jensen's inequality for the replica averages then gives

$$\|g_t - \beta\nabla f(\theta_t)\|^2 \le \beta^4 L^2\|\nabla f(\theta_t)\|^2$$

and

$$\|\tilde{g}_t - \beta\nabla f(\tilde{\theta}_t)\|^2 \le \beta^4 L^2\|\nabla f(\tilde{\theta}_t)\|^2.$$

Substituting these estimates yields

$$\mathcal{V}_{t+1} - \mathcal{V}_t \le -\eta\beta\left(\frac{1}{2} - \frac{3}{2}\beta^2L^2\right)\|\nabla f(\theta_t)\|^2 - \eta\beta\left(\frac{1}{2} - \frac{5}{2}\beta^2L^2\right)\|\nabla f(\tilde{\theta}_t)\|^2.$$

Finally, since $\beta L \le 1/4$,

$$\frac{1}{2} - \frac{3}{2}\beta^2L^2 \ge \frac{13}{32}, \qquad \frac{1}{2} - \frac{5}{2}\beta^2L^2 \ge \frac{11}{32} \ge \frac{1}{3}.$$

Therefore,

$$\mathcal{V}_{t+1} - \mathcal{V}_t \le -\frac{\eta\beta}{3}\left(\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\right).$$

Summing this inequality from $t = 0$ to $T - 1$ and using $\mathcal{V}_T \ge 0$, we obtain

$$\frac{\eta\beta}{3}\sum_{t=0}^{T-1}\left(\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\right) \le \mathcal{V}_0.$$

Dividing by $\eta\beta T/3$ gives

$$\frac{1}{T}\sum_{t=0}^{T-1}\left(\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\right) \le \frac{3\mathcal{V}_0}{\eta\beta T}$$

Moreover, if $\theta_0 = \tilde{\theta}_0$, then

$$\mathcal{V}_0 = 2\big(f(\theta_0) - f^\star\big),$$

and hence

$$\frac{1}{T}\sum_{t=0}^{T-1}\left(\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\right) \le \frac{6\big(f(\theta_0) - f^\star\big)}{\eta\beta T}.$$

□

Theorem B (Stochastic Lyapunov Descent)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be L-smooth and bounded below by $f^\star$. Suppose that the assumptions of Lemma 2 and Lemma 3 hold. Set $\mathcal{F}_t := \mathcal{F}_{t,0}$. Fix $H \geq 1$ and $\eta, \beta > 0$ such that*

$$0 < \beta L \leq \frac{1}{8\sqrt{2}}, \qquad \eta\beta L \leq \frac{1}{2}.$$

For every replica $i = 1, \dots, K$, define the stochastic local trajectories

$$\theta^i_{t,0} = \theta_t, \qquad \theta^i_{t,h+1} = \theta^i_{t,h} - \frac{\beta}{H}\nabla F(\theta^i_{t,h}; \xi^i_{t,h}), \qquad h = 0, \dots, H-1,$$

and

$$\tilde{\theta}^i_{t,0} = \tilde{\theta}_t, \qquad \tilde{\theta}^i_{t,h+1} = \tilde{\theta}^i_{t,h} - \frac{\beta}{H}\nabla F(\tilde{\theta}^i_{t,h}; \tilde{\xi}^i_{t,h}), \qquad h = 0, \dots, H-1.$$

Define

$$g^i_t := \theta_t - \theta^i_{t,H}, \qquad \tilde{g}^i_t := \tilde{\theta}_t - \tilde{\theta}^i_{t,H},$$

and

$$g_t = \frac{1}{K}\sum_{i=1}^K g^i_t, \qquad \tilde{g}_t = \frac{1}{K}\sum_{i=1}^K \tilde{g}^i_t.$$

Consider the outer updates

$$\theta_{t+1} = \theta_t - \frac{\eta}{2}(g_t + \tilde{g}_t), \qquad \tilde{\theta}_{t+1} = \theta_t - \eta\tilde{g}_t.$$

Set

$$\kappa := \frac{3}{2\eta\beta},$$

and define

$$\mathcal{V}_t := f(\theta_t) - f^\star + f(\tilde{\theta}_t) - f^\star + \kappa\|\theta_t - \tilde{\theta}_t\|^2.$$

Then

$$\mathbb{E}[\mathcal{V}_{t+1} - \mathcal{V}_t | \mathcal{F}_t] \leq -\frac{3\eta\beta}{16}\left(\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\right) + 32\frac{\eta\beta^2 L\sigma^2}{H} + 32\frac{L\eta^2\beta^2\sigma^2}{KH}.$$

Moreover, for every $T \geq 1$, if $\theta_0 = \tilde{\theta}_0$,

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\Big[\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\Big] \leq \frac{32\big(f(\theta_0) - f^\star\big)}{3\eta\beta T} + \frac{512}{3}\frac{\beta L\sigma^2}{H} + \frac{512}{3}\frac{\eta\beta L\sigma^2}{KH}.$$

Proof. Following the proof of Theorem A, and decomposing each stochastic update into its conditional mean and its centered fluctuation, we have

$$\begin{aligned}\mathbb{E}[\mathcal{V}_{t+1} - \mathcal{V}_t | \mathcal{F}_t] \leq\ & -\frac{\eta\beta}{2}\|\nabla f(\theta_t)\|^2 - \frac{\eta\beta}{2}\|\nabla f(\tilde{\theta}_t)\|^2 \\ & + \frac{3\eta}{2\beta}\left\|\mathbb{E}[g_t \mid \mathcal{F}_t] - \beta\nabla f(\theta_t)\right\|^2 \\ & + \frac{5\eta}{2\beta}\left\|\mathbb{E}[\tilde{g}_t \mid \mathcal{F}_t] - \beta\nabla f(\tilde{\theta}_t)\right\|^2 \\ & + 4L\eta^2\left(\mathbb{V}[g_t \mid \mathcal{F}_t] + \mathbb{V}[\tilde{g}_t \mid \mathcal{F}_t]\right).\end{aligned}$$

By Lemma 3,

$$\left\|\mathbb{E}[g_t \mid \mathcal{F}_t] - \beta\nabla f(\theta_t)\right\|^2 \leq 2(e-1)^2\beta^4 L^2\|\nabla f(\theta_t)\|^2 + (e^2-1)\frac{\beta^3 L\sigma^2}{H},$$

and similarly,

$$\left\|\mathbb{E}[\tilde{g}_t \mid \mathcal{F}_t] - \beta\nabla f(\tilde{\theta}_t)\right\|^2 \leq 2(e-1)^2\beta^4 L^2\|\nabla f(\tilde{\theta}_t)\|^2 + (e^2-1)\frac{\beta^3 L\sigma^2}{H}.$$

Using

$$(e-1)^2 \leq 8, \qquad e^2 - 1 \leq 8,$$

we obtain

$$\|\mathbb{E}[g_t \mid \mathcal{F}_t] - \beta\nabla f(\theta_t)\|^2 \le 16\beta^4 L^2\|\nabla f(\theta_t)\|^2 + 8\frac{\beta^3 L\sigma^2}{H},$$

and

$$\left\|\mathbb{E}[\tilde{g}_t \mid \mathcal{F}_t] - \beta\nabla f(\tilde{\theta}_t)\right\|^2 \le 16\beta^4 L^2\|\nabla f(\tilde{\theta}_t)\|^2 + 8\frac{\beta^3 L\sigma^2}{H}.$$

We next bound the conditional variance. For each replica, the stability estimate from Lemma 2 gives

$$\mathbb{V}[g_t^i \mid \mathcal{F}_t] \le \frac{e^2-1}{2}\frac{\beta^2\sigma^2}{H} \le 4\frac{\beta^2\sigma^2}{H}.$$

By conditional independence across replicas,

$$\mathbb{V}[g_t \mid \mathcal{F}_t] = \frac{1}{K^2}\sum_{i=1}^{K}\mathbb{V}[g_t^i \mid \mathcal{F}_t] \le 4\frac{\beta^2\sigma^2}{KH}.$$

Similarly,

$$\mathbb{V}[\tilde{g}_t \mid \mathcal{F}_t] \le 4\frac{\beta^2\sigma^2}{KH}.$$

Substituting these estimates gives

$$\begin{aligned}\mathbb{E}[\mathcal{V}_{t+1} - \mathcal{V}_t|\mathcal{F}_t] \le & -\eta\beta\left(\frac{1}{2} - 24\beta^2L^2\right)\|\nabla f(\theta_t)\|^2 \\ & -\eta\beta\left(\frac{1}{2} - 40\beta^2L^2\right)\|\nabla f(\tilde{\theta}_t)\|^2 \\ & + 32\frac{\eta\beta^2 L\sigma^2}{H} + 32\frac{L\eta^2\beta^2\sigma^2}{KH}.\end{aligned}$$

Since

$$\beta L \le \frac{1}{8\sqrt{2}},$$

we have

$$40\beta^2L^2 \le \frac{5}{16},$$

and therefore

$$\frac{1}{2} - 40\beta^2L^2 \ge \frac{3}{16}.$$

Moreover,

$$\frac{1}{2} - 24\beta^2L^2 \ge \frac{1}{2} - 40\beta^2L^2 \ge \frac{3}{16}.$$

Thus,

$$\mathbb{E}[\mathcal{V}_{t+1} - \mathcal{V}_t|\mathcal{F}_t] \le -\frac{3\eta\beta}{16}\left(\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\right) + 32\frac{\eta\beta^2 L\sigma^2}{H} + 32\frac{L\eta^2\beta^2\sigma^2}{KH}.$$

Equivalently,

$$\mathbb{E}[\mathcal{V}_{t+1} - \mathcal{V}_t|\mathcal{F}_t] \le -\frac{3\eta\beta}{16}\left(\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\right) + 32(\beta L)\frac{\eta\beta\sigma^2}{H} + 32(\eta\beta L)\frac{\eta\beta\sigma^2}{KH}.$$

Taking total expectation and summing from $t = 0$ to $T-1$, and using $\mathcal{V}_T \ge 0$, yields

$$\frac{3\eta\beta}{16}\sum_{t=0}^{T-1}\mathbb{E}\Big[\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\Big] \le \mathcal{V}_0 + 32T\frac{\eta\beta^2 L\sigma^2}{H} + 32T\frac{L\eta^2\beta^2\sigma^2}{KH}.$$

Dividing by $3\eta\beta T/16$, we obtain

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\Big[\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\Big] \le \frac{16\mathcal{V}_0}{3\eta\beta T} + \frac{512}{3}\frac{\beta L\sigma^2}{H} + \frac{512}{3}\frac{\eta\beta L\sigma^2}{KH}.$$

Finally, if $\theta_0 = \tilde{\theta}_0$, then

$$\mathcal{V}_0 = 2\big(f(\theta_0) - f^\star\big),$$

and therefore

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\Big[\|\nabla f(\theta_t)\|^2 + \|\nabla f(\tilde{\theta}_t)\|^2\Big] \le \frac{32\big(f(\theta_0) - f^\star\big)}{3\eta\beta T} + \frac{512}{3}\frac{\beta L\sigma^2}{H} + \frac{512}{3}\frac{\eta\beta L\sigma^2}{KH}.$$

□

Table 8: Timing estimates for the 134M model under the representative H100 setting described above. H_{hide} is the minimum number of inner steps required to hide the complete outer path when the method permits full overlap.

Method	c_m	$T_{\text{outer}}^{(m)}$	T_{round} at $H = 30$	H_{hide}
DiLoCo	2	0.36 s	2.02 s	–
Eager DiLoCo	2	0.36 s	1.94 s	–
Delayed DiLoCo	2	0.36 s	1.58 s	9
LACCO	5	0.89 s	1.58 s	19

D TIMING AND BANDWIDTH CONSIDERATIONS

Let P denote the number of model parameters, K the number of replicas, and N_{tok} the global number of tokens processed per inner step. Using the standard estimate of $6P$ FLOPs per token, the duration of one inner step is

$$T_{\text{step}} \simeq \frac{6PN_{\text{tok}}}{K\rho_{\text{MFU}}F_{\text{peak}}}, \qquad T_{\text{inner}} = HT_{\text{step}},$$

where F_{peak} is the peak accelerator throughput and ρ_{MFU} is the model FLOPs utilization.

Assuming an FP32 outer gradient and a ring all-reduce, the communication time is approximated by

$$T_{\text{comm}} \simeq \frac{2(K-1)}{K}\frac{4P}{b_{\text{net}}},$$

where b_{net} is the effective inter-replica bandwidth.

The CPU–GPU transfer cost depends on the method. Let c_m denote the number of FP32 parameter-sized buffers transferred by method m per outer step. We use

$$c_{\text{DiLoCo}} = c_{\text{Eager}} = c_{\text{Delayed}} = 2, \qquad c_{LACCO} = 5,$$

where the additional three buffers for LACCO correspond to the inactive AdamW state. The resulting outer-update cost is

$$T_{\text{outer}}^{(m)} \simeq T_{\text{opt}}^{(m)} + c_m\frac{4P}{b_{\text{CPU/GPU}}}.$$

In the estimates below, we neglect the optimizer kernel time relative to the transfer time.

The wall-clock time per communication round is therefore

$$\begin{aligned}
T_{\text{DiLoCo}} &= T_{\text{inner}} + T_{\text{comm}} + T_{\text{outer}}^{(\text{DiLoCo})},\\
T_{\text{Eager}} &= \max\{T_{\text{inner}}, T_{\text{comm}}\} + T_{\text{outer}}^{(\text{Eager})},\\
T_{\text{Delayed}} &= \max\{T_{\text{inner}}, T_{\text{comm}} + T_{\text{outer}}^{(\text{Delayed})}\},\\
T_{LACCO} &= \max\{T_{\text{inner}}, T_{\text{comm}} + T_{\text{outer}}^{(LACCO)}\}.
\end{aligned}$$

Representative H100 setting. For the 134M model with $K = 4$ and $N_{\text{tok}} = 131{,}072$, we take $F_{\text{peak}} = 1\,\text{PFLOP/s}$, $\rho_{\text{MFU}} = 0.5$, $b_{\text{net}} = 10\,\text{GB/s}$, and $b_{\text{CPU/GPU}} = 3\,\text{GB/s}$. This gives

$$T_{\text{step}} \simeq 52.7\,\text{ms}, \qquad T_{\text{comm}} \simeq 80\,\text{ms}, \qquad \frac{4P}{b_{\text{CPU/GPU}}} \simeq 179\,\text{ms}.$$

For $H = 30$, $T_{\text{inner}} \simeq 1.58\,\text{s}$.

For methods that overlap the complete outer path,

$$H_{\text{hide}}^{(m)} = \left\lceil \frac{T_{\text{comm}} + T_{\text{outer}}^{(m)}}{T_{\text{step}}} \right\rceil.$$

Thus, Delayed DiLoCo requires $H \geq 9$, while the larger optimizer state of LACCO requires $H \geq 19$. Both are below the $H = 30$ used in our main experiments, for which the outer path is fully hidden.