DECOUPLING FEDERATED LEARNING TO ADDRESS UTILITY-DRIVEN ASYNCHRONY

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated learning (FL) enables scalable collaborative training through a server that establishes a fixed communication structure and governs the aggregation of parameters sent by the clients. Efficient FL is marred by *exogenous* factors such as stragglers, network delays, and staleness, and *endogenous* factors such as local distribution shifts. These factors introduce arbitrary client availability and changes in the aggregation mechanism – unobservable to the server. In fact, the effect of such phenomenon is only observed in the clients' utility. Clients must be allowed to drive learning according to their local utility. This does not happen at present because client and the server depend on each other for updates leading to a coupling completely driven by the server. In this paper, we enable *utility-driven* FL, where the clients can communicate based on "how their local utility changes with time?" instead of a pre-determined communication structure. We leverage dynamic-programming to develop a novel framework that decouples client-server communication dependency and design UASYNCFL where client can drive learning based on their own utility. Empirically and theoretically, we show that convergence can be achieved in spite of no predetermined communication structure and in the presence of staleness, client churn, gradient magnitude variance, and angular misalignment. In particular, without prior knowledge of the endogenous/exogenous factors present in the dataset, UASYNCFL outperforms standard methods, carefully designed for specific (endogenous/exogenous) factors on standard benchmarks and realistic datasets by upto 46%.

1 INTRODUCTION

Federated learning (FL) seeks to train a model at a central location (server) using data distributed across multiple clients (McMahan et al., 2017). In typical FL, whether synchronous (McMahan et al., 2017) or asynchronous (Xie et al., 2019; Nguyen et al., 2022), training proceeds over communication rounds: the server aggregates client parameters using a predetermined rule, and clients update the latest server parameters on new data. As each side depends on the other's updates, there is a dependency between client and server updates, which we call "coupling." This coupling dictates the learning and the coupling depends on a rigid communication and coordination structure – predetermined by the server.

However, the server is poorly placed to make such decisions. For example, a hospital's model may degrade when a new scanner is added (Roschewitz et al., 2023; Rieke et al., 2020), and a grid load forecaster may face a feeder that gains an EV charging depot (Taik & Cherkaoui, 2020). The server cannot observe such issues as the clients are evolving with time. Moreover, "endogenous factors" such as data-distribution shifts change the utility (objective function) and "exogenous factors" such as network delays (Leconte et al., 2024; Lu et al., 2024), stragglers (Xie et al., 2019; Nguyen et al., 2022) and staleness Nguyen et al. (2022) alter communication structures. Both phenomenons may happen on the fly during the learning processm, developing patterns that the server cannot observe a priori. As such behavior is observed in "how the utility changes over time?," rigid communication structure can force the clients to communicate without taking the utility into account. *To alleviate this, two central challenges need to be addressed, first, the "coupling" must be disentangled such that clients can arbitrarily decide when to communicate. Second, a mechanism must be established to update server/client without prior knowledge of the type of asynchrony.*

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.

We decouple the server-client update dependency (coupling) by allowing the client/server updates proceed on independent clocks. Then, client/server updates are enabled by two buffers, a client pool that server reads and a server pool the clients read. We use a dynamic programming formalism and derive a discrete-time Hamilton-Jacobi-Bellman equation at both server and client to establish "how utility changed with time?" We then construct UAsyncFL algorithm to update the client/server leveraging the buffers that elucidate, "how the asynchrony is observed over time?"

Our *theoretical analysis* establishes that, UAsyncFL is guaranteed to converge irrespective of the type of asynchrony Thm. 3, Lem. 1 and evolution of asynchrony. *Empirically*, we show that, for non-IID setting with arbitrary endogenous factors (Table 2), UAsyncFL outperforms six baselines designed to handle specific factors of asynchrony on FL benchmarks (MNIST, CIFAR-10, CIFAR-100). We show the superiority of UAsyncFL on WILDS FMoW – a real world dataset, where a 29% improvement over FedAsync and a 10% improvement over FedBuff are observed. In particular, WILDS FMoW is a dataset with implicit distribution drift where a dedicated method such as FedDrift is expected to perform well. However, from Table 3, UAsyncFL outperforms FedDrift by 12%. Furthermore, we also show that in the presence of IID, no heterogeneity and asynchrony, UAsyncFL matches state of the art synchronous FL performance. The results from Fig. 2, Table 2 and Table 3 indicate by decoupling client server dependency and allowing client to control the learning, UAsyncFL captures different endogenous and exogenous factors holistically without knowing the exact form of asynchrony in prior at the server and is able to outperform methods that are specifically designed to outperform certain types of asynchrony.

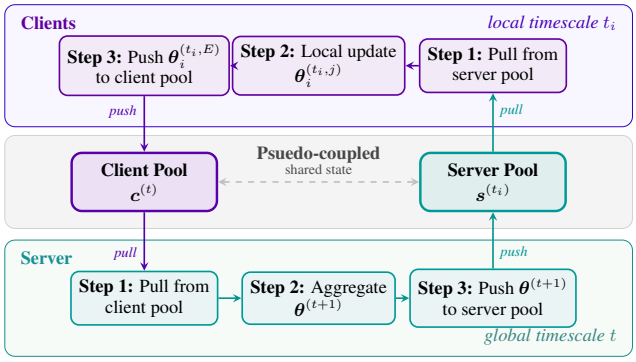

Figure 1: Architecture of the utility-driven asynchronous FL framework. Clients (blue) and server (teal) operate on independent timescales, communicating asynchronously via shared pools in the synchro layer.

- **Contributions:** We show that, in realistic application, lack of control at the client on communication patterns and aggregation structures as key bottle necks. We introduce a dynamic programming driven framework to address this.
- We instantiate this framework as UAsyncFL and develop an algorithm for FL in the presence of arbitrary asynchrony.
- We prove convergence of our algorithm in the non-convex smooth setting and show empirically through benchmarks and realistic datasets that UAsyncFL indeed captures asynchrony holistically.

2 Related Work

We organize prior work by the source of asynchrony and who decides when clients communicate.

Exogenous asynchrony. In most asynchronous FL methods treat asynchrony, the goal is to avoid waiting for the clients that may be slow and intermittently available. Solutions include staleness-aware aggregation (Xie et al., 2019), semi-asynchronous and tiered protocols (Wu et al., 2021; Chai et al., 2020; 2021), buffered aggregation (Nguyen et al., 2022; Toghani & Uribe, 2022), cached-update calibration (Wang et al., 2024), and queuing analyses of delay (Leconte et al., 2024); see Fraboni et al. (2023); Xu et al. (2023). A related line analyzes arbitrary, intermittent, or correlated participation (Ruan et al., 2021; Wang & Ji, 2022; Ribero et al., 2022; Rodio et al., 2023; Gu et al., 2021) and Yang et al. (2022) lets clients decide when to participate. Nevertheless, the timing of updates is external: participation is modeled independent of the client's data and of the model's utility, and the server dictates aggregation.

Endogenous asynchrony. A separate line studies client data that evolve over time: periodically shifting populations (Zhu et al., 2021a), federated continual learning (Yoon et al., 2021; Li et al., 2025),

distributed concept drift (Jothimurugesan et al., 2023; Chen et al., 2021), streaming data (Huynh et al., 2025), and asynchronous task arrival (Shenaj et al., 2023). These methods adapt the model to the shift but keep a server-driven schedule, so a client whose utility has degraded cannot initiate an update. The closest mechanisms for client-initiated communication come from communication-efficient optimization, where nodes transmit only when their local state changes sufficiently (Chen et al., 2018; Zehtabi et al., 2022); these reduce communication under a fixed objective.

Client drift. Endogenous asynchrony is distinct from, but interacts with, *client drift*. Drift is a cross-client effect: under heterogeneous but fixed objectives, local steps pull clients toward their own optima (Zhao et al., 2018; Karimireddy et al., 2020). It is corrected with control variates (Karimireddy et al., 2020; 2021), regularization (Li et al., 2020b; Acar et al., 2021), normalized aggregation (Wang et al., 2020), or server momentum and extrapolation (Hsu et al., 2019; Reddi et al., 2021; Jhunjhunwala et al., 2023). Endogenous asynchrony is instead a within-client-temporal effect that governs the timing of an update and the distribution behind it.

In summary, existing methods address exogenous or endogenous asynchrony separately, but, holistically, all present methods assume a server-controlled communication structure. As discussed earlier, such server controlled interaction structure requires asynchrony to be known in prior, which is an important issue with present approaches; an issue our approach as in (Fig. 1) is designed for.

3 UTILITY-DRIVEN ASYNCFL

Notations: We consider a FL system with K local models denoted $\boldsymbol{\theta}_i \in \Theta : i \in [K] = \{1, \ldots, K\}$, and a single server having a global model $\boldsymbol{\theta} \in \Theta$ operating on separate time scales. The server is indexed by $t \in \{1, \ldots, T\}$ and the client i is indexed by $t_i \in \{1, \ldots, T_i\}$ with T and T_i being their respective time horizons. At each t_i, the client updates for E epochs, indexed by j.

$$\min_{\boldsymbol{\theta} \in \Theta} \left\{ \frac{1}{K} \sum_{i=1}^{K} \left(f_i(\boldsymbol{\theta}) \equiv \mathbb{E}_{\xi \in \mathcal{D}_i} \left[\ell(\boldsymbol{\theta}; \xi) \right] \right) \right\} \tag{1}$$

The typical FL problem is that of determining $\boldsymbol{\theta}$ by solving Eq. (1), where ξ describes the random sample at any client $i \in \{1, \cdots, K\}$ through $\mathcal{D}_i$ which is the data at the client. Due to asynchronous arrival or departure of clients, the total number of clients is unknown at any t and a client time may not correspond to the server at t. Therefore wee regard the number of clients as time dependent.

$$\min_{\boldsymbol{\theta}^{(t)} \in \Theta} \left\{ \frac{1}{K^{(t)}} \sum_{i=1}^{K^{(t)}} \left(f_i(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}_i^{(t_i)}) \equiv \mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \left[\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}_i^{(t_i)}; \xi) \right] \right) \right\} \tag{2}$$

Dynamic programming (DP) (Bertsekas, 2017) is the standard method to model and solve a problem of the class in Eq. (2) where an optimization problem is defined for each $t \in [1, T]$ to get

$$\min_{\{\boldsymbol{\theta}^{(t)}\}_{t=1}^{T} \in \Theta} \sum_{t=1}^{T} \frac{1}{K(t)} \sum_{i=1}^{K(t)} f_i(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}_i^{(t_i)}) \tag{Utility-Fed}$$

In Eq. (Utility-Fed) describes the effect of arbitrary arrival and departure is modeled through its effect on utility f_i at the client. Solving Eq. (Utility-Fed) provides a sequence of server aggregations $\{\boldsymbol{\theta}^{(t)}\}_{t=1}^{T}$ at each t. However, we need to address the coupling between t and t_i.

Utility-driven asynchronous participation. Let $\mathcal{D}_i^t$ denote the local data distribution of client i at time t. After a client-specific elapsed time t_i, the local distribution may evolve to $\mathcal{D}_i^{(t+t_i)}$. We say that client i has an increased incentive to participate in FL when the utility of the current global model deteriorates under its evolved local distribution. In the simplest loss-based form, this can be expressed as $f_i(\boldsymbol{\theta}^{(t)}; \mathcal{D}_i^{(t+t_i)}) \geq f_i(\boldsymbol{\theta}^{(t)}; \mathcal{D}_i^{(t)}) + \varepsilon_i^{(t_i)}$, where f_i denotes the local loss of client i and $\varepsilon_i^{(t_i)}$ denotes incurred utility loss due to distribution change. Because t_i and the resulting degradation can differ across clients, clients become interested in rejoining federated training at different times. We refer to this as *utility-driven asynchrony* where different clients and server operate at different time scales calculating f t and t_i to coincide, hence "coupled."

Decoupling client and server time scales: Client and server operate at different time scales. Client i observes the server-side global model at t_i through a random variable – server memory ($\boldsymbol{s}$) and the server observes the client parameters at t through a random variable – client memory ($\boldsymbol{c}$). Therefore

$$\mathbf{S}: \min_{\{\boldsymbol{\theta}^{(t)}\}_{t=1}^{T}\in\Theta} \left\{ \sum_{t=1}^{T} \mathbb{E}_{\mathbb{P}}[f(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)})] \right\} \quad \mathbf{C}: \min_{\{\boldsymbol{\theta}_i^{(t_i)}\}_{t_i=1}^{T_i}\in\Theta} \left\{ \sum_{t_i=1}^{T_i} \mathbb{E}_{\xi\in\mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi) \right] \right\},$$

where we have replaced the effect of $K(t)$ with $\mathbb{E}_{\mathbb{P}}$ calculated over the probability space of all clients, i.e. $\mathbb{P}$. The server-side problem depends on how the server observes client parameters through $\boldsymbol{c}$, and the client-side problem depends on how it observes server parameters through $\boldsymbol{s}$. Due to this, client side and server side problems are decoupled and we obtain

$$\mathbf{S}: v_{\pi}^{\star}(t, \boldsymbol{c}^{(t)}) = \min_{\pi^{(t)}} v_{\pi}(t, \boldsymbol{c}^{(t)}), \quad \mathbf{C}: \quad v_{\pi_i}^{\star}(t_i, \boldsymbol{s}^{(t_i)}) = \min_{\pi_i^{(t_i)}} \; v_{\pi_i}(t_i, \boldsymbol{s}^{(t_i)}), \tag{Value}$$

where value at client $v_{\pi_i}(t_i, \boldsymbol{s}^{(t_i)})$ and value at server as $v_{\pi}(t, \boldsymbol{c}^{(t)})$ are the costs, $\{\boldsymbol{\theta}^{(t)}\}_{t=1}^{T}$ is the server policy $\pi^{(t)}$, $\{\boldsymbol{\theta}_i^{(t_i)}\}_{t=1}^{T_i}$ is client policy $\pi_i^{(t_i)}$. *The server memory and client memory act as the "pseudo-coupling" that explicit synchronization at immense benefits.* By standard DP rules, the problem is solvable given knowledge of which client arrives at each future instant – formally handling endogenous and exogenous factors in a unified manner. However, Eq. (Value) remains intractable because future client and server states are unobservable at t or t_i.

Federated Hamilton Jacobi Bellman: We decompose this problem through a recursive characterization known as the discrete-time Hamilton–Jacobi–Bellman (HJB) equation (Bertsekas, 2017; Ohsawa et al., 2011) to obtain the federated learning system as

$$\begin{aligned}
&\mathbf{S}: \quad \Delta v_{\pi}^{\star}(t, \boldsymbol{c}^{(t)}) = \min_{\pi} \left[f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) + \left\langle \nabla_{\boldsymbol{c}} v_{\pi}^{\star}(t+1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \right\rangle \right] \\
&\mathbf{C}: \quad \Delta v_{\pi_i}^{\star}(t_i, \boldsymbol{s}^{(t_i)}) = \min_{\pi_i} \left[\mathbb{E}_{\xi\in\mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi) \right] + \left\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^{\star}(t_i+1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \right\rangle \right]
\end{aligned} \tag{CFL}$$

where $\Delta \boldsymbol{s}^{(t_i)} = h_{\mathcal{S}}(\boldsymbol{s}^{(t_i)}, \boldsymbol{\theta}^{(t)}) - \boldsymbol{s}^{(t_i)}$, and $\Delta \boldsymbol{c}^{(t)} = h_{\mathcal{C}}\big(\boldsymbol{c}^{(t)}, \{\boldsymbol{\theta}_i^{(t_i)}\}\big) - \boldsymbol{c}^{(t)}$ with $h_{\mathcal{S}}$ and $h_{\mathcal{C}}$. referring to the update maps for the server and client pools. The terms $\Delta \boldsymbol{s}^{(t_i)}, \Delta \boldsymbol{c}^{(t)}$ refer to the change in the server/client pool. For the full derivation see Section A2.2.

4 UAsyncFL— An algorithm

To efficiently solve Eq. (CFL), we first define the terms inside the minimization as the Hamiltonian (following conventions) and write

$$\begin{aligned}
\mathrm{H}_{\pi}(\boldsymbol{c}, \boldsymbol{\theta}) &= f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) + \left\langle \nabla_{\boldsymbol{c}} v_{\pi}^{\star}(t+1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \right\rangle \\
\mathrm{H}_{\pi_i}(\boldsymbol{s}, \boldsymbol{\theta}_i) &= \mathbb{E}_{\xi\in\mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi) \right] + \left\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^{\star}(t_i+1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \right\rangle,
\end{aligned} \tag{3}$$

which implies $\mathbf{S}: \min_{\pi} \mathrm{H}_{\pi}(\boldsymbol{c}, \boldsymbol{\theta})$; $\mathbf{C}: \min_{\pi_i} \; \mathrm{H}_{\pi_i}(\boldsymbol{s}, \boldsymbol{\theta}_i)$, with updates to the policy given as $\mathbf{S}: \pi \leftarrow \nabla_{\pi} \mathrm{H}_{\pi}(\boldsymbol{c}, \boldsymbol{\theta})$; $\mathbf{C}: \pi_i \leftarrow \nabla_{\pi_i} \mathrm{H}_{\pi_i}(\boldsymbol{s}, \boldsymbol{\theta}_i)$. *Practical approximation for client server updates* are necessary to obtain UAsyncFL and provided in Prop. 3 in Section A2.4 such that

$$\boldsymbol{\theta}^{(t+1)} = \mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] + \alpha^{(t)} \left[\mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] - \mathbb{E}_{\mathbb{P}}[\boldsymbol{s}^{(t-1)}] \right], \tag{Server Update}$$

$$\boldsymbol{\theta}_i^{(t_i, j+1)} = \boldsymbol{\theta}_i^{(t_i, j)} - \eta^{(j)} \boldsymbol{g}_i^{(j)} - \beta^{(t_i, j)} \left\langle \boldsymbol{J}^{(t_i, j)}, \Delta \boldsymbol{s}^{(t_i)} \right\rangle, \tag{Client Update}$$

where we have added a superscript to indicate the local index at the j-th epoch and $\boldsymbol{g}_i^{(j)}$ is the stochastic gradient obtained at the client side and $\alpha^{(t)}, \eta^{(j)}, \beta^{(t_i, j)}$ representing the respective step sizes. See Algorithm 1, 2 with Fig. 1 for full practical implementation of UAsyncFL.

4.1 Analysis

In this analysis, we tackle the following questions: (i) does our method converge; (ii) how do the endogenous and exogenous factors affect convergence? Towards this end, we make some standard assumptions about the setting including A. 3, A. 1, and A. 2, and assume convergence on the client side (see (Krishnan & Balaprakash, 2021) for full proofs and Section A2.5 for more details). We assume that the client/server pool and the asynchrony (both endogenous and exogenous) introduces an error $\epsilon^{(t,t_i)}$ which is bounded as in A. 3.

Convergence Analysis — UAsyncFL:

Theorem 1. *Consider Eq.* (CFL) *and let A. 3, A. 1 and A. 2 hold. Define the server and client updates as in Eq.* (Server Update) *and Eq.* (Client Update)*.Then,*

$$\min_{t\in\{0,1,2,\cdots,T\}} \left(\mathbb{E}_{\mathbb{P}}\left[\left\|\nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)})\right\|^2\right]\right)$$
$$\leq \underbrace{\frac{2}{\mathrm{A}}\,\mathbb{E}_{\mathbb{P}}\left[f(\boldsymbol{\theta}^{(0)}, \boldsymbol{s}^{(0)}) - f^*\right]}_{\text{(I) Optimization}} + \underbrace{2L_{\epsilon_B}\,\mathrm{B}}_{\text{(II) Comm. bias}}$$
$$+ \underbrace{2(L_{\boldsymbol{\theta}} + L_{\boldsymbol{s}})\,\mathrm{AB}^2}_{\text{(III) Curvature}} + \underbrace{\mathcal{O}(\Delta_{\epsilon_B}/K)}_{\text{(IV) Residual}}. \tag{4}$$

with A *being a function of* $\alpha^{(t)}, \beta^{(t_i,j)}, \rho_{\max}$ *and* B *being a function of* $\epsilon_B, \Delta_{\epsilon_B}, \eta^{(j)}$ *as in Lem. 3. Choose* $\beta^{(t_i,j)}, \rho_{\max}, \eta^{(j)}, \alpha^{(t)}$ *as in in Thm. 3 to ensure A and B, then* $\epsilon_B, \Delta_{\epsilon_B} \to 0$.

Algorithm 1: UAsyncFL: Server

```
Initialize client pool c
while t ∈ [1, ..., T] do
    if size of client pool > C then
        Sample c from the client pool
        Update server with Eq. (Server Update)
    end if
    Update s^(t_i).
end while
```

Algorithm 2: UAsyncFL: Client

```
Initialize server pool and s
while t_i ∈ [1, ..., T_i] do
    if size of server pool > S then
        Sample from the server pool
        Initialize θ_i^(t_i,0) with E_P[s^(t_i)]
        while For each batch j of data do
            Update client with Eq. (Client Update)
        end while
        Update the client pool c^(t).
    end if
end while
```

Effect of heterogeneity on UAsyncFL: We assume that $\epsilon^{(t,t_i)}$ decomposes into *exogenous* and *endogenous* effects and consider four factors: staleness, client churn, gradient diversity and magnitude variance (Li et al., 2020a; Karimireddy et al., 2020; Zhao et al., 2018; Zhu et al., 2021b) (see A. 4).

Lemma 1 (Communication Error Bound)**.** *Let A. 3, A. 1 and A. 4 hold with gradient bound* g_M *as in A. 1. Let* $D_{\boldsymbol{\theta}} = \max_{s\in[T]} \left\|\Delta\boldsymbol{\theta}^{(s)}\right\|$ *denote the maximum single-step server displacement. Then,*

$$\mathbb{E}_{\mathbb{P}}\left[\left\|\epsilon_i^{(t,t_i)}\right\|\right] \leq \overbrace{\frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K}}^{\text{exogenous}} + \overbrace{\left(\sum_{j=1}^{E}\eta^{(j)}\right)\left(\sigma_{\mathrm{mag}} + g_M\sqrt{2(1-\delta_{\min})}\right)}^{\text{endogenous}} \tag{Bound}$$

$$\mathbb{E}_{\mathbb{P}}\left[\left\|\Delta\epsilon_i^{(t,t_i)}\right\|\right] \leq \underbrace{\frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K}}_{\text{exogenous}} + \underbrace{\frac{2\gamma}{K}\left(\sum_{j=1}^{E}\eta^{(j)}\right) g_M\,\max\left(1,\,\sqrt{2(1-\delta_{\min})}\right)}_{\text{endogenous}}$$

where we assume $\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| \leq \frac{1}{K}\sum_i \mathbb{E}_{\mathbb{P}}[\left\|\epsilon_i^{(t,t_i)}\right\|]$ *and* $\left\|\mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]\right\| \leq \frac{1}{K}\sum_i \mathbb{E}_{\mathbb{P}}[\left\|\Delta\epsilon_i^{(t,t_i)}\right\|]$*,* γ *is the churn rate,* $\tau_{\max}$ *is the maximum allowable staleness. The proof is provided in Section A2.6.*

The exogenous component couples the maximum client staleness $\tau_{\max}$ with the server step size Xie et al. (2019) which depends on $\epsilon^{(t,t_i)}$, and $\epsilon^{(t,t_i)}$ depends on $D_{\boldsymbol{\theta}}$. This circularity can be resolved when specific learning rate schedules are substituted as in Thm. 3.

The endogenous component in Eq. (Bound) captures the combined effect of gradient magnitude variance (σ_{mag}) and angular misalignment ($\delta_{\min}$) coupled through g_M. This coupling reveals a variance component (as in the (G, B)-bounded gradient divergence of (Karimireddy et al., 2020)) and an angular component related to the gradient diversity measure of (Yin et al., 2018).

Full proofs, the well-posedness condition, and detailed rate characterization are deferred to Appendix Section A2.5, A2.7, A2.6.1 and A2.6. Source code is provided as a supplementary material.

5 EXPERIMENTS

We empirically test our insights from Section 4.1 under various degree of asynchrony through both endogenous and exogenous factors. Additional results are in the Section A1.2.

Experimental Design We evaluate UASYNCFL across four regimes: *exogenous IID*, *exogenous non-IID*, *endogenous IID*, and *endogenous non-IID* across three datasets MNIST (2-layer CNN), CIFAR-10 (ResNet-18), and CIFAR-100 (ResNet-18), with non-IID partitions generated via Dirichlet label splits. In the exogenous setting, clients participate through an arbitrary asynchronous schedule independent of model utility. In the endogenous setting, each client is assigned a local image rotation angle ρ_i that serves as a controlled proxy for distributional mismatch, and a participation interval $q_i = \max(1, \lceil |\rho_i|/\rho_0 \rceil)$ that couples participation timing to local utility loss—realizing the decoupled client-server clocks of Eq. (CFL). We compare against six baselines spanning synchronous (FedAvg (McMahan et al., 2017)), staleness-aware (FedAsync (Xie et al., 2019)), buffered (FedBuff (Nguyen et al., 2022)), drift-aware (FedDrift (Jothimurugesan et al., 2023)), context-aware (CA^{2}FL (Wang et al., 2024)), and time-prior (FedTKM (Zhu et al., 2021a)) aggregation. Results are reported as mean $\pm$ standard deviation over three seeds. Full protocol details are deferred to Section A1.1 and the experimental hyperparameters are listed in Table A1.

5.1 RESULTS AND DISCUSSION

The effect of **Exogenous Factors** is reported in Table 1 where client participation is arbitrary rather than utility-triggered for both IID and non-IID scenario. In the **IID setting**, it is expected that, the addition of new clients simply introduces redundancy. As such, a competitive performance is expected from UASYNCFL as observed in Table 1. We note that UASYNCFL improves over a standard synchronous FL method FedAvg by 0.40, 4.14, and 8.68 percentage points (pp) on MNIST, CIFAR-10, and CIFAR-100. Moreover, UASYNCFL matches or outperforms all asynchronous baselines across MNIST, CIFAR-10, and CIFAR-100. Gains over FedAsync (up to +12.12 pp / 24.5%), FedDrift (up to +5.72 pp / 15.0%), FedTKM (up to +11.94 pp / 37.6%), and CA²FL (up to +10.84 pp / 32.8%) are substantial, while improvements over FedBuff are modest (up to +2.32 pp / 3.6%) but consistent. The modest performance improvement compared to FedBuff is due to a similar buffer/pool driven performance in FedBuff. Note however, that the degree of asynchrony here is not significant due to the IID nature of the problem. Nonetheless, the largest relative gains appear on CIFAR-100, where baselines are stressed the most.

Table 1: Accuracy (%) under exogenous participation across IID and non-IID settings. UASYNCFL remains competitive under arbitrary asynchronous arrivals and achieves the strongest performance on CIFAR-10 and CIFAR-100 in both regimes.

Method	MNIST	CIFAR-10	CIFAR-100
IID			
FedAvg	98.50 ± 0.12	74.70 ± 1.17	35.05 ± 2.08
FedAsync	$\mathbf{99.04 \pm 0.15}$	66.72 ± 1.69	35.12 ± 2.04
FedBuff	98.90 ± 0.10	76.52 ± 0.97	42.22 ± 2.39
FedDrift	94.20 ± 0.60	75.32 ± 1.69	38.01 ± 2.72
FedTKM	98.76 ± 0.11	76.04 ± 1.93	31.79 ± 2.17
CA^{2}FL	92.64 ± 0.18	68.00 ± 1.92	32.93 ± 2.05
UASYNCFL	98.90 ± 0.14	$\mathbf{78.84 \pm 1.18}$	$\mathbf{43.73 \pm 2.31}$
Non-IID			
FedAvg	93.56 ± 0.57	70.70 ± 1.34	32.61 ± 2.31
FedAsync	92.22 ± 1.77	61.14 ± 3.26	33.16 ± 3.63
FedBuff	95.58 ± 1.36	72.06 ± 3.51	38.52 ± 1.87
FedDrift	88.08 ± 1.71	67.44 ± 3.70	33.83 ± 3.55
FedTKM	94.10 ± 1.82	66.68 ± 3.13	29.68 ± 3.40
CA^{2}FL	88.66 ± 1.13	67.74 ± 3.15	28.04 ± 2.03
UASYNCFL	$\mathbf{95.80 \pm 1.25}$	$\mathbf{73.30 \pm 2.83}$	$\mathbf{41.02 \pm 2.72}$

In the **non-IID setting**, UASYNCFL is expected to benefit from the server and client pool architecture. This is observed in Table 1, where UASYNCFL outperforms all baselines across MNIST, CIFAR-10, and CIFAR-100. Gains are largest over FedAsync (up to $+12.16$ pp / 23.7%), CA^{2}FL (up to $+12.98$ pp / 46.3%), and FedTKM (up to $+11.34$ pp / 38.2%), with consistent improvements over FedAvg (up to $+8.41$ pp / 25.8%) and FedDrift (up to $+7.72$ pp / 21.3%); gains over FedBuff are smaller but uniform (up to $+2.50$ pp / 6.5%). As expected, improvements scale with task difficulty—smallest on MNIST (saturated) and largest on CIFAR-100, where client drift most stresses baseline aggregation.

The effect of **endogenous factors** is reported in Table 2. To introduce such heterogeneity, we change the active rotation angle within the images over time. Therefore, participation timing, local data,

and gradient statistics are jointly non-stationary. This setting represents Eq. (Utility-Fed) and the endogenous component of Lem. 1. UASYNCFL performs as expected and is strongest in this regime as the extra correction term in Eq. (Server Update) explicitly uses the change in server memory to compensate for the oncoming heterogeneity.

Evidently, in the **IID setting**, UASYNCFL achieves significant improvements, reflecting baselines' inability to handle utility-driven asynchrony. The most striking gain is over FedAvg on CIFAR-100 ($+19.10$ pp / 77.7%), where data-heterogeneity-induced participation severely degrades synchronous aggregation. Gains over CA^{2}FL (up to $+11.13$ pp / 34.2%), FedTKM (up to $+8.27$ pp / 23.4%), and FedAsync (up to $+7.75$ pp / 21.6%) are also substantial, while improvements over FedDrift (up to $+5.56$ pp / 14.6%) and FedBuff (up to $+3.40$ pp / 8.4%) remain consistent. As in earlier settings, gains scale with task complexity, with CIFAR-100 showing the largest improvements.

Table 2: Accuracy (%) under endogenous participation across IID and non-IID settings. UASYNCFL achieves the highest accuracy on all datasets in both regimes, showing that pool-based aggregation and client-side correction are especially effective when participation frequency is coupled with local distribution shifts.

Method	MNIST	CIFAR-10	CIFAR-100
IID			
FedAvg	93.72 ± 0.27	65.48 ± 2.41	24.57 ± 1.23
FedAsync	92.84 ± 0.11	68.68 ± 2.96	35.92 ± 2.06
FedBuff	96.64 ± 0.19	73.68 ± 0.72	40.27 ± 0.51
FedDrift	95.90 ± 0.44	71.62 ± 1.13	38.11 ± 3.24
FedTKM	94.04 ± 0.12	72.50 ± 0.95	35.40 ± 2.08
CA^{2}FL	89.02 ± 0.20	66.00 ± 1.16	32.54 ± 0.30
UASYNCFL	$\mathbf{98.10 \pm 0.14}$	$\mathbf{75.70 \pm 0.14}$	$\mathbf{43.67 \pm 0.11}$
Non-IID			
FedAvg	89.52 ± 1.61	64.12 ± 0.40	32.03 ± 1.30
FedAsync	89.60 ± 0.96	63.04 ± 0.33	32.58 ± 2.90
FedBuff	93.16 ± 2.62	68.70 ± 1.87	39.05 ± 0.62
FedDrift	89.06 ± 0.90	69.52 ± 1.47	36.49 ± 4.34
FedTKM	92.61 ± 0.12	69.40 ± 0.56	32.66 ± 1.35
CA^{2}FL	87.62 ± 0.22	60.98 ± 0.41	31.89 ± 0.60
UASYNCFL	$\mathbf{93.78 \pm 1.30}$	$\mathbf{73.14 \pm 0.51}$	$\mathbf{40.69 \pm 0.78}$

Under **endogenous non-IID data**, where utility-driven participation compounds with data heterogeneity, UASYNCFL maintains substantial gains across all baselines: largest over CA^{2}FL (up to $+12.16$ pp / 27.6%), FedAsync (up to $+10.10$ pp / 24.9%), and FedAvg (up to $+9.02$ pp / 27.0%); consistent over FedTKM (up to $+8.03$ pp / 24.6%) and FedDrift (up to $+4.72$ pp / 11.5%); and smallest but uniform over FedBuff (up to $+4.44$ pp / 6.5%). These results empirically support Thm. 1: by controlling communication error through memory-pool aggregation and client-side correction, UASYNCFL remains converges even when angular divergence and magnitude variance are simultaneously active (Lem. 1).

5.2 REAL-WORLD DATASET:

To analyze the performance of UASyncFL on a realistic dataset, we implemented WILDS FMoW (Functional Map of the world) dataset and the results are in Table 3. FMoW is appropriate because distribution shift is induced by real spatial and temporal factors. In particular, satellite images differ across regions, years, sensors, and land-use categories which introduce indistinguishable endogenous and exogenous shifts, making our approach suitable.

Experimental Setting: For this data, we follow FedDrift-style FMoW protocol: five clients formed from geographic regions, with data evolving across temporal partitions using the same partitioning as FedDrift. At time t, clients train on their current partition and evaluate on the next partition, measuring adaptation to gradual drift. We use ResNet-18 with ImageNet initialization, RGB images, full client partitions, and UAsyncFL aggregation.

Table 3: Comparison on WILDS FMoW under the FedDrift-aligned federated setting.

Method	Final Acc. (%)
FedAsync	56.42 ± 1.24
FedBuff	65.98 ± 0.66
FedDrift	64.84 ± 0.51
UAsyncFL	72.96 ± 0.74

Results: On FMoW, UAsyncFL improves over FedAsync by +16.54 pp (29.3% relative), over FedBuff by +6.98 pp (10.6% relative), and over FedDrift by +8.12 pp (12.5% relative). The gain over FedBuff again reflects that the improvement is not only from buffering but from the utility-aware corrections, and the gain over a drift-specialised baseline (FedDrift) shows that the method remains competitive against methods built specifically for drift.

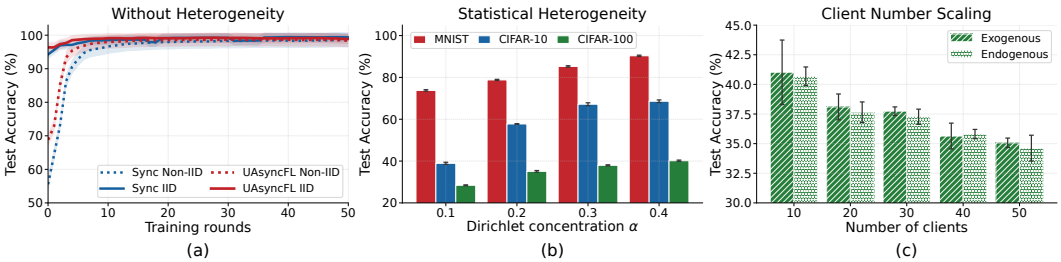

Figure 2: Ablation Studies of UAsyncFL. (a) UAsyncFL closely tracks synchronous FedAvg, showing that pool-triggered aggregation does not degrade convergence when the client heterogeneity is removed, (b) With the decrease in statistical heterogeneity (higher Dirichlet concentration α), accuracy improves across all datasets for UAsyncFL, and (c) Increasing the number of clients gradually lowers CIFAR-100 accuracy, but the degradation remains controlled under both exogenous and endogenous participation.

5.3 Why UAsyncFL outperforms baselines.

Each baseline captures a partial view of asynchrony. For instance, FedAvg suffices when local objectives align (MNIST) but cannot react between synchronization points. FedAsync handles staleness but ignores the coupling between update timing and shifting local distributions due to endogenous factors. FedBuff is the closest competitor—its buffered aggregation smooths noisy updates—but the same buffering delays adaptation when the active client domain changes. FedDrift assumes that the structure of the drift is known a priori but does not handle arbitrary shift in the data distribution. FedTKM's time prior helps with stable periodic participation but falters when label skew and covariate shift interact. CA^{2}FL's context-aware weighting loses calibration when the context signal must simultaneously track label skew and rotation-driven shifts. Baseline methods in the literature that seek to attack some form of asynchrony depend on prior knowledge of endogenous and exogenous factors, which is simple not available in any realistic setting. UAsyncFL avoids this necessaity by decoupling the need to know the heterogeneity or communication patterns in prior.

Computational Complexity of UAsyncFL. We compare wall-clock time against test accuracy on the endogenous non-IID CIFAR-10 and CIFAR-100 runs (Fig. A1, Section A1.2). UAsyncFL's cumulative time over 100 rounds is comparable to baselines, but its accuracy gains are substantial—up to 29% over baselines on CIFAR-100 at the cost of only 9% additional wall-clock time over FedDrift.

We also compare peak CPU and GPU memory allocation across baselines on non-IID CIFAR-10 with ResNet-18, averaged over five rounds (Fig. A2, Section A1.2). UAsyncFL incurs $\sim$6% higher CPU and $\sim$15% higher GPU overhead than baselines without persistent tensors (FedAvg, FedAsync)—expected, since the extra correction term in Eq. (Server Update) adds transient client-side autograd work. Against baselines maintaining additional algorithmic state (FedDrift, FedTKM), UAsyncFL is comparable on GPU and 1–5% better on CPU.

Table 4: Component-wise ablation of UAsyncFL on endogenous CIFAR-10. Accuracy is final overall test accuracy (%)

Setting	IID (%)	Non-IID(%)
UAsyncFL	75.70	73.14
w/o JVP	74.20	72.01
w/o server momentum	72.33	69.60
w/o JVP and momentum	71.88	68.45
FedBuff	73.68	68.70

5.4 Ablation Study

Component-level ablation. We report a component-wise ablation to isolate the roles of the client-side JVP correction, the server momentum correction by switching off different terms from Eq. (Server Update) and Eq. (Client Update). The results can be found in Table 4. Here, removing both corrections reduces the update to buffered pool aggregation but without the staleness correction FedBuff has. As observed from the results, the corresponding FedBuff scores are 73.68 (IID) / 68.70 (non-IID), and

the full UAsyncFL improves on it by +2.02 / +4.44 pp. In fact, just introducing the server momentum term without the JVP correction at the client, improves performance over the FedBuff scores.

Under no heterogeneity. Our analysis in Lem. 1 indicates that, under the presence of no endogeneous or exogeneous factors, our approach is comparable to a simple synchronous FL setting. To support this hypothesis, we test UASYNCFL under a fixed client population with no client replacement or dropout, thus introducing no heterogeneity; results are depicted in Fig. 2 (a). In this setting, all clients remain available throughout training and the synchronous baseline aggregates only after collecting updates from all clients, whereas UASYNCFL performs an update once the client pool reaches the prescribed pool size. UASYNCFL closely tracks synchronous FedAvg in both IID and non-IID settings: nearly indistinguishable after the first few rounds in the IID case, and rising more quickly from a higher starting accuracy in the non-IID case. This supports the no-churn implication of Lem. 1—without client replacement, angular disruption from churn vanishes and asynchrony reduces to staleness alone. *Decoupling the client and server clocks therefore does not materially degrade optimization, while avoiding the synchronization barrier that forces FedAvg to wait for every client.*

Effect of Statistical Heterogeneity. In our experiments, varying the Dirichlet concentration parameter changes the degree of statistical heterogeneity. We report results for UASYNCFL under varying degrees in Fig. 2 (b), smaller α produces more concentrated client label distributions, larger gradient disagreement, and therefore a larger communication-error neighborhood in Thm. 1. This is the standard difficulty of non-IID FL: when each client observes a narrower slice of the label space, local updates become more client-specific and the average update is a noisier proxy for the global descent direction. The results follow this prediction. As α increases from 0.1 to 0.4, MNIST accuracy improves by approximately 22.4%, CIFAR-10 improves by approximately 76.3%, and CIFAR-100 improves by approximately 41.4%. The improvement is especially sharp on CIFAR-10 because moderate label mixing quickly restores useful shared visual features. CIFAR-100 remains harder because the task has many more classes, fewer examples per class per client, and a more difficult representation problem, so even larger α leaves substantial client drift. This is consistent with the theory: the bound depends on induced gradient disagreement, not on the partition parameter alone.

Client Scaling. is studied in Fig. 2 (c) (additional results are in Fig. A3) where we vary client counts from 10 to 50. The theory contains two competing effects. Increasing K can average the pool error in A. 3 and Lem. 1 leading to the faster convergence rate as well, but with the same training horizon it also gives each client fewer effective opportunities to influence the global model and can increase the diversity of local objectives. This is a familiar scaling tradeoff in cross-device FL: more clients improve population coverage, but non-IID partitioning across more clients can make each local update more biased. The results reflect this tradeoff. Under exogenous non-IID participation, increasing the number of clients from 10 to 30 reduces MNIST accuracy by approximately 12.5% and CIFAR-10 accuracy by approximately 8.0%. CIFAR-100, which is evaluated up to 50 clients, decreases by approximately 14.5% from 10 to 50 clients. Under endogenous participation, the same trend appears: CIFAR-100 decreases by approximately 14.9% from 10 to 50 clients, while CIFAR-10 decreases by approximately 8.9% from 10 to 30 clients. These drops are noticeable but not catastrophic, which supports the well-posedness discussion after Lem. 1: larger client pools can absorb bounded staleness, but only when the induced heterogeneity remains controlled.

6 CONCLUSION

We introduced *utility-driven asynchrony* that seeks to decouple the client and server communication requirements. The key advantage was to ensure that change in utility can drive federated training even when the source of asynchrony is unknown in prior. Theoretically, we established convergence of our method to a small neighborhood governed by the degree of asynchrony and the convergence neighborhood vanished as a function of number of clients. Empirically, UASYNCFL matches or outperforms six baselines across IID, non-IID, exogenous and endogenous settings. Our results indicate that UASYNCFL captures different endogenous and exogenous factors holistically and outperform methods that are specifically designed for certain types of asynchrony. For instance, the performance improvement over FedDrift on the realistic dataset. We note that, decoupling the client server dependency allowed UASYNCFL to capture the evolving nature of asynchrony by looking at the evolution of the utility function– hence, utility-driven asynchrony.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish writing, find related works, boilerplate generation of the project code structure, integration of baseline codes to our project code-base, code generation to draw the experimental figures using our experimental data, and latex formatting of graphics/tables. We have not used generative AI tools for research ideation, mathematical proof, methodological implementation of our proposed and baseline methods. We have reviewed all our AI-assisted work. For example, the related works suggested by AI were manually checked for relevance and the bibtex files were verified. Also, the code generated by AI was verified and tested by two independent reviewers via multiple test cases. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work involves no human subjects and no data collection, and every environment is simulated. This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

REPRODUCIBILITY STATEMENT

To reproduce the reported results, we provide the full architecture details, hyperparameters, dataset partitioning (Dirichlet splits), client participation schedules, and training protocol in Section A. Results are reported as mean and standard deviation over three random seeds. Code is released as part of the submission package. Instructions for running the code are detailed in the 'readme' file within the code package. The code-base includes scripts to reproduce all experimental results for the proposed method and baselines. We also provide the complete mathematical derivation in Section B.

REFERENCES

Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N. Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*, 2021.

Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control, Vol. I*. Athena Scientific, Belmont, MA, 4th edition, 2017. ISBN 978-1-886529-43-4.

Zheng Chai, Ahsan Ali, Syed Zawad, Stacey Truex, Ali Anwar, Nathalie Baracaldo, Yi Zhou, Heiko Ludwig, Feng Yan, and Yue Cheng. TiFL: A tier-based federated learning system. In *ACM International Symposium on High-Performance Parallel and Distributed Computing (HPDC)*, 2020.

Zheng Chai, Yujing Chen, Ali Anwar, Liang Zhao, Yue Cheng, and Huzefa Rangwala. FedAT: A high-performance and communication-efficient federated learning system with asynchronous tiers. In *International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2021.

Tianyi Chen, Georgios B. Giannakis, Tao Sun, and Wotao Yin. LAG: Lazily aggregated gradient for communication-efficient distributed learning. In *Advances in Neural Information Processing Systems*, 2018.

Yujing Chen, Zheng Chai, Yue Cheng, and Huzefa Rangwala. Asynchronous federated learning for sensor data with concept drift. In *2021 IEEE International Conference on Big Data (Big Data)*, pp. 4822–4831. IEEE, 2021. doi: 10.1109/BigData52589.2021.9671924.

Yann Fraboni, Richard Vidal, Laetitia Kameni, and Marco Lorenzi. A general theory for federated optimization with asynchronous and heterogeneous clients updates. *Journal of Machine Learning Research*, 24(110), 2023.

Xinran Gu, Kaixuan Huang, Jingzhao Zhang, and Longbo Huang. Fast federated learning in the presence of arbitrary device unavailability. In *Advances in Neural Information Processing Systems*, volume 34, pp. 12052–12064, 2021.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

Tan-Khiem Huynh, Malcolm Egan, Giovanni Neglia, and Jean-Marie Gorce. Streaming federated learning with markovian data. *arXiv preprint arXiv:2503.18807*, 2025.

Divyansh Jhunjhunwala, Shiqiang Wang, and Gauri Joshi. FedExP: Speeding up federated averaging via extrapolation. In *International Conference on Learning Representations*, 2023.

Ellango Jothimurugesan, Kevin Hsieh, Jianyu Wang, Gauri Joshi, and Phillip B Gibbons. Federated learning under distributed concept drift. In *International Conference on Artificial Intelligence and Statistics*, pp. 5834–5853. PMLR, 2023.

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 (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5132–5143. PMLR, 2020.

Sai Praneeth Karimireddy, Martin Jaggi, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2021.

Raghavan Krishnan and Prasanna Balaprakash. Formalizing the generalization-forgetting trade-off in continual learning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 13498–13509. Curran Associates, Inc., 2021. URL https://arxiv.org/abs/2109.14035.

Louis Leconte, Matthieu Jonckheere, Sergey Samsonov, and Eric Moulines. Queuing dynamics of asynchronous federated learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 1711–1719. PMLR, 2024.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020a.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems (MLSys)*, 2020b.

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

Yichen Li, Yuying Wang, Haozhao Wang, Yining Qi, Tianzhe Xiao, and Ruixuan Li. FedSSI: Rehearsal-free continual federated learning with synergistic synaptic intelligence. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 36151–36169. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/li25cq.html.

Renhao Lu, Weizhe Zhang, Qiong Li, Hui He, Xiaoxiong Zhong, Hongwei Yang, Desheng Wang, Zenglin Xu, and Mamoun Alazab. Adaptive asynchronous federated learning. *Future Generation Computer Systems*, 152:193–206, 2024.

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 (AISTATS)*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017.

John Nguyen, Kshitiz Malik, Hongyuan Zhan, Ashkan Yousefpour, Michael Rabbat, Mani Malek, and Dzmitry Huba. Federated learning with buffered asynchronous aggregation. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 151 of *Proceedings of Machine Learning Research*. PMLR, 2022.

Tomoki Ohsawa, Anthony M. Bloch, and Melvin Leok. Discrete Hamilton–Jacobi theory. *SIAM Journal on Control and Optimization*, 49(4):1829–1856, 2011. doi: 10.1137/090776822.

Sashank J. 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.

Mónica Ribero, Haris Vikalo, and Gustavo de Veciana. Federated learning under intermittent client availability and time-varying communication constraints. *IEEE Journal of Selected Topics in Signal Processing*, 17(1):98–111, 2022.

Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletarì, Holger R. Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N. Galtier, Bennett A. Landman, Klaus Maier-Hein, Sébastien Ourselin, Micah Sheller, Ronald M. Summers, Andrew Trask, Daguang Xu, Maximilian Baust, and M. Jorge Cardoso. The future of digital health with federated learning. *npj Digital Medicine*, 3(1):119, 2020. doi: 10.1038/s41746-020-00323-1.

Angelo Rodio, Francescomaria Faticanti, Othmane Marfoq, Giovanni Neglia, and Emilio Leonardi. Federated learning under heterogeneous and correlated client availability. In *IEEE/ACM Transactions on Networking*, volume 32, pp. 1451–1460, 2023.

Mélanie Roschewitz, Galvin Khara, Joe Yearsley, Nisha Sharma, Jonathan J James, Éva Ambrózay, Adam Heroux, Peter Kecskemethy, Tobias Rijken, and Ben Glocker. Automatic correction of performance drift under acquisition shift in medical image classification. *Nature Communications*, 14(1):6608, 2023.

Yichen Ruan, Xiaoxi Zhang, Shu-Che Liang, and Carlee Joe-Wong. Towards flexible device participation in federated learning. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Minseok Ryu, Youngdae Kim, Kibaek Kim, and Ravi K Madduri. APPFL: open-source software framework for privacy-preserving federated learning. In *2022 IEEE international parallel and distributed processing symposium workshops (IPDPSW)*, pp. 1074–1083. IEEE, 2022.

Donald Shenaj, Marco Toldo, Alberto Rigon, and Pietro Zanuttigh. Asynchronous federated continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 5055–5063, June 2023. doi: 10.1109/CVPRW59228.2023.00534. URL https://openaccess.thecvf.com/content/CVPR2023W/FedVision/papers/Shenaj_Asynchronous_Federated_Continual_Learning_CVPRW_2023_paper.pdf.

Afaf Taik and Soumaya Cherkaoui. Electrical load forecasting using edge computing and federated learning. In *ICC 2020 – 2020 IEEE International Conference on Communications (ICC)*, pp. 1–6. IEEE, 2020. doi: 10.1109/ICC40277.2020.9148937.

Mohammad Taha Toghani and César A. Uribe. Unbounded gradients in federated learning with buffered asynchronous aggregation. In *Allerton Conference on Communication, Control, and Computing*, 2022.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *arXiv preprint arXiv:2007.07481*, 2020.

Shiqiang Wang and Mingyue Ji. A unified analysis of federated learning with arbitrary client participation. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Yujia Wang, Yuanpu Cao, Jingcheng Wu, Ruoyu Chen, and Jinghui Chen. Tackling the data heterogeneity in asynchronous federated learning with cached update calibration. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=4aywmeb97I.

Wentai Wu, Ligang He, Weiwei Lin, Rui Mao, Carsten Maple, and Stephen Jarvis. SAFA: A semi-asynchronous protocol for fast federated learning with low overhead. *IEEE Trans. Comput.*, 70(5):655–668, May 2021. ISSN 0018-9340. doi: 10.1109/TC.2020.2994391. URL https://doi.org/10.1109/TC.2020.2994391.

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

Chenhao Xu, Youyang Qu, Yong Xiang, and Longxiang Gao. Asynchronous federated learning on heterogeneous devices: A survey. *Computer Science Review*, 50, 2023.

Haibo Yang, Xin Zhang, Prashant Khanduri, and Jia Liu. Anarchic federated learning. In *International Conference on Machine Learning*, pp. 25331–25363. PMLR, 2022.

Dong Yin, Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett. Gradient diversity: a key ingredient for scalable distributed learning. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 84 of *Proceedings of Machine Learning Research*, pp. 1998–2007. PMLR, 2018.

Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12073–12086. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/yoon21b.html.

Shahryar Zehtabi, Seyyedali Hosseinalipour, and Christopher G. Brinton. Decentralized event-triggered federated learning with heterogeneous communication thresholds. In *IEEE Conference on Decision and Control (CDC)*, 2022.

Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-IID data. *arXiv preprint arXiv:1806.00582*, 2018.

Chen Zhu, Zheng Xu, Mingqing Chen, Jakub Konečnỳ, Andrew Hard, and Tom Goldstein. Diurnal or nocturnal? federated learning of multi-branch networks from periodically shifting distributions. In *International Conference on Learning Representations*, 2021a.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International conference on machine learning*, pp. 12878–12889. PMLR, 2021b.

APPENDIX

Contents

A1 Empirical Analysis . . . 14
A1.1 Experimental Design . . . 14
A1.2 Additional Experiment Results . . . 17
A2 Mathematical Analysis . . . 20
A2.1 Notation Table . . . 20
A2.2 Federated Discrete Bellman Equations . . . 20
A2.3 Server and Client Updates . . . 21
A2.4 Algorithmic Approximation . . . 22
A2.5 Communication Error Analysis . . . 30
A2.6 Convergence Analysis . . . 24
A2.6.1 First Difference on the Server Side . . . 25
A2.6.2 Upper Bound on the First Difference . . . 26
A2.6.3 Minimum Gradient Value . . . 28
A2.6.4 Convergence Rate . . . 33
A2.7 Discussion on the Bound and Rate . . . 35

A EMPIRICAL ANALYSIS

A1.1 EXPERIMENTAL DESIGN

Participation protocols. We evaluate two forms of asynchrony. In the *exogenous* setting, clients participate according to an arbitrary asynchronous schedule that is not determined by the utility of the current model. This setting isolates the effect of irregular arrivals, stale updates, and partial participation. In the *endogenous* setting, each client is assigned a local image rotation angle. The rotation angle is a controlled proxy for local distributional mismatch: larger rotations create a larger gap between the current global model and the client's active local domain, which corresponds to a larger utility loss for that client. The scheduler converts this utility proxy into a participation interval through the following

$$q_i = \max\left(1, \left\lceil \frac{|\rho_i|}{\rho_0} \right\rceil\right), \tag{A1}$$

where ρ_i is the active rotation angle of client i and ρ_0 is the rotation threshold. Client i participates in rounds t satisfying $t \bmod q_i = 0$. Thus, clients with stronger local shifts are assigned longer, client-specific update intervals, while clients with smaller shifts remain close to synchronous participation. In the endogenous setting, each client alternates between a primary and alternate rotation angle every five rounds, and the interval q_i is recomputed from the active angle. This realizes the setting of Eqs. (Utility-Fed) and (CFL), where client time t_i and server time t are decoupled and the arrival process is tied to the client data-generating process.

Server and client updates. UAsyncFL follows the pool-based protocol in Algorithms 1 and 2. Clients pull the current server state, perform local training, and push their update into a client pool. The server aggregates once the pool reaches threshold $C = 5$. The server update uses the momentum-corrected average in Eq. (Server Update), while each UAsyncFL client augments the local cross-entropy objective with the Jacobian-vector product correction in Eq. (Client Update). This term approximates the sensitivity of the local objective to the change in the server memory, which is precisely the role of the future-value gradient term in the HJB system Eq. (CFL). Synchronous FedAvg is recovered by using a full client threshold and zero momentum, matching the recovery statement following Lem. 1.

Regimes. We evaluate four regimes: exogenous IID, exogenous non-IID, endogenous IID, and endogenous non-IID. The exogenous regimes test whether a method can handle arbitrary asynchronous participation when the arrival process is not itself a utility signal. The endogenous regimes test the harder case where the client distribution, local utility, and participation interval evolve together. This

distinction directly probes the decomposition in A. 4: the exogenous results emphasize staleness and participation irregularity, while the endogenous results additionally activate the angular-divergence and magnitude-variance components.

Datasets and models. We use MNIST, CIFAR-10, CIFAR-100, and FMoW. MNIST is trained with a 2-Layer convolutional network, while CIFAR-10, CIFAR-100, and FMoW use a ResNet-18 model with a CIFAR-compatible stem and no pretrained weights. IID partitions assign examples uniformly across clients. Non-IID partitions use a Dirichlet split with concentration parameter $\alpha = 0.5$ unless otherwise specified. The non-IIDness ablation varies $\alpha \in \{0.1, 0.2, 0.3, 0.4\}$, where smaller values induce stronger label skew and therefore larger angular misalignment in the sense of A. 4. For FMoW dataset, we used the aligned settings as reported in Jothimurugesan et al. (2023). All the results are reported in a mean $\pm$ standard format iterating over three different seeds.

Baselines. We compare against synchronous FedAvg (McMahan et al., 2017), FedAsync (Xie et al., 2019), FedBuff (Nguyen et al., 2022), FedDrift (Jothimurugesan et al., 2023), CA^{2}FL (Wang et al., 2024), and a time-kernel baseline denoted FedTKM (Zhu et al., 2021a). These baselines cover synchronous aggregation, staleness-aware asynchronous aggregation, buffered asynchronous aggregation, drift-aware expert selection, context-aware asynchronous weighting, and time-prior adaptation. All baselines use the same dataset split, client schedule, model family, and training horizon as UAsyncFL within each regime.

Compute Details. All our experiments, measurement of the training time and memory overhead are conducted on a compute cluster node equipped with four NVIDIA A100 GPUs with 40 GB HBM per GPU and one AMD Milan CPU with 64 physical cores and 256 GB of DDR4 host memory. We have used open-source APPFL (Ryu et al., 2022) package as the FL orchestrator framework and imported the available baselines like FedAsync and FedBuff from APPFL code base. The implementation supports CPU, single-GPU, and distributed multi-GPU execution. In the distributed setting used for the timing runs, client local training is assigned across the four GPUs, while server-side pool storage and aggregation are performed on the CPU. Thus, the measured GPU cost mainly reflects client training and the additional client-side operations such as the JVP correction in UAsyncFL, whereas the pool memory and aggregation state are accounted for in host memory. We report elapsed wall-clock time over 100 training rounds for each baseline in Fig. A1.

Table A1: Hyperparameters used in the experiments.

Parameter	**Value**
Datasets	MNIST, CIFAR-10, CIFAR-100, FMoW
Models	CNN for MNIST, ResNet-18 for CIFAR-10, CIFAR-100, and FMoW
Clients	10 in baseline and non-IIDness studies, 10, 20, 30, 40, 50 in scaling studies; 5 for FMoW
Rounds	100 global rounds
Seed	4, 8, 12
Optimizer	Adam
Local epochs	2
Batch size	64 for training and validation
Learning rate	10^{-3}
Loss and metric	Cross-entropy loss, sample-weighted accuracy
Validation support	100 validation samples per class
IID partition	Uniform split across clients; not applied to the FMoW CSV
Non-IID partition	Dirichlet label split, $\alpha = 0.5$ unless varied; predefined client/time partitions for FMoW
Non-IIDness ablation	$\alpha \in \{0.1, 0.2, 0.3, 0.4\}$; not applied to FMoW
Rotation angles	$5^\circ, 10^\circ, \ldots, 50^\circ$, repeated for larger client counts; disabled for FMoW
Endogenous alternate angles	$10^\circ, 5^\circ, 20^\circ, 15^\circ, 30^\circ, 25^\circ, 40^\circ, 35^\circ, 50^\circ, 45^\circ$, repeated for larger client counts; disabled for FMoW
Rotation threshold ρ_0	5°; not applied to FMoW
Participation interval	$q_i = \max(1, \lceil \lvert\rho_i\rvert / \rho_0 \rceil)$; not applied to FMoW
Endogenous switch period	5 rounds; no synthetic rotation switching for FMoW
UAsyncFL aggregation threshold	5 client updates
UAsyncFL server momentum	0.5
UAsyncFL JVP weight	0.01
FedAvg synchronous subset	5 clients per round
FedBuff buffer size and mixing	$K = 4$, $\alpha = 0.9$, polynomial staleness with $a = 0.5$, $b = 4.0$
CA^{2}FL buffer and step size	Buffer size 5, $\eta = 0.01$
FedDrift parameters	Drift threshold 0.06, isolation 1 round, at most 10 experts
FedTKM parameters	Period 10, geometric step 0.2, label-smoothing weight 0.1
FMoW dataset version	Labeled WILDS FMoW v1.1
FMoW partition	FedDrift partition A
FMoW image input	224×224 RGB

A1.2 Additional Experiment Results

Computation Complexity

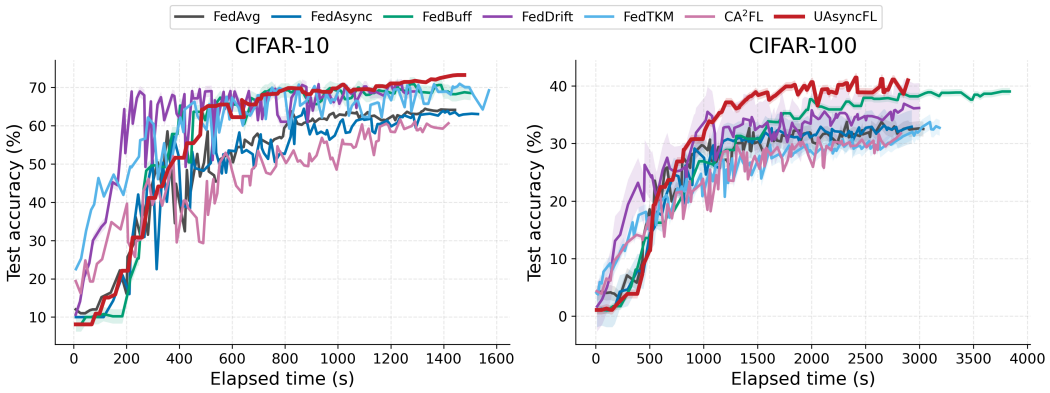

Figure A1: Elapsed wall-clock time versus test accuracy on endogenous non-IID CIFAR-10 and CIFAR-100. UAsyncFL demonstrates gradual increase to an intermediate target on CIFAR-10, but it achieves the strongest final accuracy. On CIFAR-100, where label space and client heterogeneity are larger, UAsyncFL reaches the target faster than FedBuff and obtains the best final accuracy, showing that the extra client-side correction becomes more valuable as the task becomes harder. Shaded regions indicate method-level standard deviations from the corresponding non-IID accuracy table.

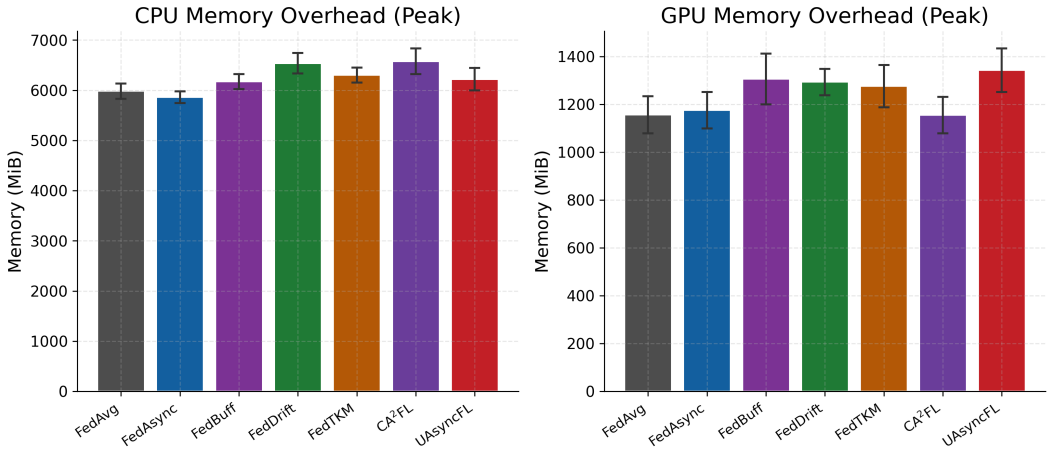

Figure A2: Peak CPU and GPU memory overhead across methods on the non-IID CIFAR-10 setting. UAsyncFL has a modest CPU-memory increase over stateless baselines because the server stores a pool of client updates before aggregation, but its CPU footprint remains below methods that maintain richer persistent state, such as FedDrift, FedTKM, and CA^{2}FL. The GPU peak is slightly higher for UAsyncFL due to the client-side JVP correction, which adds transient autograd work rather than persistent GPU storage.

SCALING PERFORMANCE OF UASYNCFL

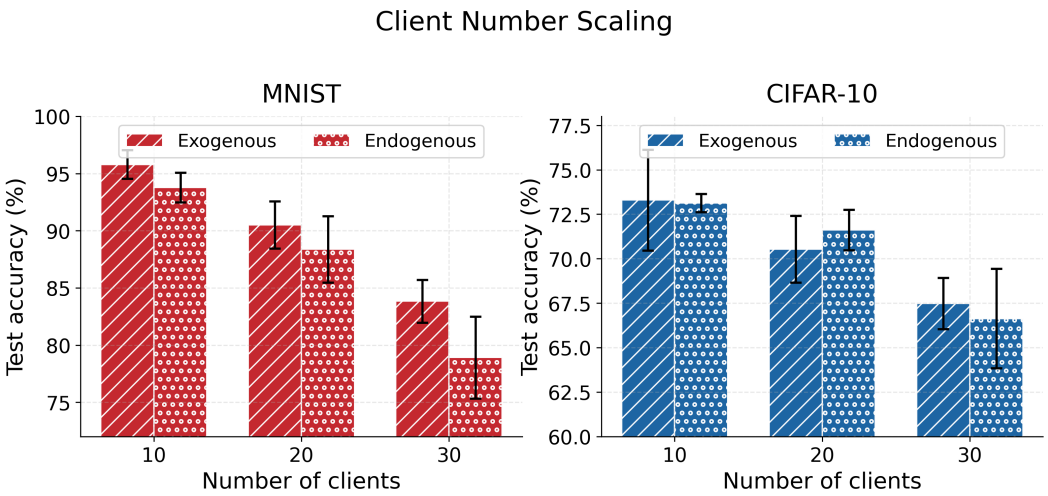

Figure A3: Performance under Client Number Scaling for MNIST-CNN and CIFAR10-ResNet18 use-cases. UAsyncFL remains stable as the client population increases from 10 to 30, but accuracy decreases as each client receives a narrower and more heterogeneous local data slice.

A1.3 SENSITIVITY ANALYSIS

Pool Size. Increasing the client pool threshold C from 2 to 7 improves accuracy by +7.19 pp (IID) and +13.60 pp (non-IID): a small pool is dominated by a few biased updates, while a larger pool averages across more client directions and reduces variance, which confirms that C trades memory for a tighter convergence neighborhood. However, this increase in pools size comes with a cost of larger time between aggregation, large memory and more staleness. Note from Table R2, one could improve the CiFar10 scores with a value of $C > 5$, however, this comes at the expense of extra time to aggregation which invariably results in staleness for the clients that do not participate in this particular aggregation. Despite the presence of such staleness, UASyncFL proves improved performance over FedBuff which has a dedicated staleness correction term, demonstrating the strength of our method. The detailed results are included in Table A2.

Table A2: Sensitivity to client pool threshold C (endogenous CIFAR-10).

Pool size B	**IID Acc. (%)**	**Non-IID Acc. (%)**	**Time between aggregations (s)- staleness metric**
2	68.91	60.70	6.27
3	72.88	66.10	9.43
4	73.55	70.40	12.44
5	75.70	73.14	15.74
6	75.60	73.10	18.81
7	76.10	74.30	21.53

JVP correction. Relative to no JVP correction (), the default improves accuracy both for (IID) and (non-IID), detail results are in Table A3. Increasing beyond this range consistently degrades performance: the correction is useful as a bounded adjustment but should not dominate the local update direction, which matches its theoretical role of reducing update mismatch.

Table A3: Sensitivity to JVP weight (endogenous CIFAR-10).

JVP Weight	**IID Acc. (%)**	**Non-IID Acc. (%)**
0	74.20	72.01
0.01	75.70	73.14
0.02	75.60	73.10
0.03	75.11	72.40
0.04	71.60	68.60
0.05	69.80	64.70
0.07	67.44	62.50
0.10	65.60	59.20
0.30	64.00	59.60

Server momentum. As shown in Table A4, moderate server momentum improves stability, whereas excessive momentum is harmful. Relative to $\mu = 0$, the default $\mu = 0.5$ improves accuracy by +1.80 pp (IID) and +2.33 pp (non-IID); momentum of 0.7 degrades accuracy and $\mu = 1.0$ collapses training, consistent with stale asynchronous updates being smoothed only within a moderate range.

Table A4: Sensitivity to server momentum (endogenous CIFAR-10).

Momentum	**IID Acc. (%)**	**Non-IID Acc. (%)**
0	72.33	69.60
0.3	75.10	71.40
0.5	75.70	73.14
0.7	69.40	66.70
1.0	10.00	10.00

B MATHEMATICAL ANALYSIS

A2.1 NOTATIONS

Table A5: Main notation used in the paper.

Symbol	Meaning
Sets, indices, and time	
$[K]$	Set of clients, $\{1, \dots, K\}$.
i	Client index.
t	Server-side asynchronous update index (server time).
t_i	Client-side asynchronous update index (client i's time).
j	Local optimization step index at client i.
E	Number of local epochs.
$T,\ T_i$	Server and client time horizons, respectively.
Θ	Parameter space.
Models, memory variables, and data	
$\boldsymbol{\theta}^{(t)}$	Global model at server time t.
$\boldsymbol{\theta}_i^{(t_i,j)}$	Local model at client i and local time t_i after j local updates from $j = 1, \dots, E$.
$\boldsymbol{s}^{(t_i)}$	Memory from the server pool available to client i at time t_i.
$\boldsymbol{c}^{(t)}$	Memory from the client pool available at server time t.
$\mathcal{D}_i^{(t_i)}$	Data observed by client i at local time t_i.
ξ	Random sample or mini-batch drawn from $\mathcal{D}_i^{(t_i)}$.
Objectives and value functions	
$\ell(t_i, \boldsymbol{\theta}_i, \boldsymbol{s}; \xi)$	Instantaneous local loss of client i at local time t_i.
$f(\boldsymbol{\theta}, \boldsymbol{c})$	Instantaneous server/global objective.
$\pi_i = \pi_i$	Policy of client i, i.e., the sequence of local iterates over its horizon.
$\pi = \pi$	Server policy, i.e., the sequence of server iterates over time.
$v_{\pi_i}(t_i, \boldsymbol{s}^{(t_i)})$	Client-side value function / cost-to-go starting from local time t_i.
$v_{\pi}(t, \boldsymbol{c}^{(t)})$	Server-side value function / cost-to-go starting from time t.
$v_{\pi_i}^\star$	Optimal client-side value function.
$v_{\pi}^\star$	Optimal server-side value function.
Updates and algorithmic quantities	
$\boldsymbol{g}_i^{(j)}$	Stochastic gradient used by client i at local epoch j.
$\eta^{(j)}$	Client-side learning rate at local epoch j.
$\alpha^{(t)}$	Server-side correction coefficient at server time t.
$\beta^{(t_i,j)}$	Client-side correction coefficient for the Jacobian-vector product term.
$p_i^{(t)}$	Aggregation weight of client i at server time t, with $\sum_{i=1}^K p_i^{(t)} = 1$.
$h_{\mathcal{C}}$	Update map for the server-side client memory $\boldsymbol{c}$.
$h_{\mathcal{S}}$	Update map for the client-side server memory $\boldsymbol{s}$.
$\boldsymbol{J}^{(t_i,j)}$	Jacobian of the local loss with respect to the server snapshot.
$\rho_{\max}$	Upper bound on the spectral norm, i.e., the largest singular value.
$\epsilon^{(t,t_i)}$	Communication mismatch, i.e., staleness gap between server time t and client time t_i.
ϵ_B	Uniform upper bound on the communication mismatch.
Regularity constants	
L	Generic smoothness constant.
g_M	Uniform upper bound on gradient norms.
M_i	Client-dependent bound used in the analysis.

A2.2 DERIVATION OF THE FEDERATED DISCRETE BELLMAN EQUATIONS

Client-Side Discrete Bellman Equation: At the client, the server state is observed only through the memory estimate $\boldsymbol{s}^{(t_i)}$. The client seeks the optimal value

$$v_{\pi_i}^\star(t_i, \boldsymbol{s}^{(t_i)}) = \min_{\pi_i}\ v_{\pi_i}(t_i, \boldsymbol{s}^{(t_i)}) \quad \text{such that} \quad \boldsymbol{s}^{(t_i+1)} = h_{\mathcal{S}}(\boldsymbol{s}^{(t_i)}, \boldsymbol{\theta}^{(t)}). \tag{A2}$$

where the server memory evolves according to the discrete update map

Proposition 1 (Client-Side Discrete HJB). *Let $v_{\pi_i}^\star(t_i, \boldsymbol{s}^{(t_i)})$ denote the solution to* (A2) *and define the discrete map $\boldsymbol{s}^{(t_i+1)} = h_{\mathcal{S}}(\boldsymbol{s}^{(t_i)}, \boldsymbol{\theta}^{(t)})$ to imply transition of the server memory. Assume $v_{\pi_i}^\star(t_i+1, \cdot)$ is differentiable with respect to $\boldsymbol{s}$ and define the memory increment $\Delta\boldsymbol{s}^{(t_i)} \coloneqq h_{\mathcal{S}}(\boldsymbol{s}^{(t_i)}, \boldsymbol{\theta}^{(t)}) - \boldsymbol{s}^{(t_i)}$. Then, $v_{\pi_i}^\star$ satisfies*

$$v_{\pi_i}^\star(t_i, \boldsymbol{s}^{(t_i)}) = \min_{\pi_i} \Big[\mathbb{E}_{\xi\in\mathcal{D}_i^{(t_i)}} \Big[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\Big] + v_{\pi_i}^\star(t_i+1, \boldsymbol{s}^{(t_i)}) + \Big\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^\star(t_i+1, \boldsymbol{s}^{(t_i)}),\ \Delta\boldsymbol{s}^{(t_i)} \Big\rangle \Big]. \tag{A3}$$

Proof. By Bellman's principle of optimality , the optimal cost-to-go decomposes as (note the future term uses $v^\star_{\pi_i}$, the optimal value, consistent with $v^\star_{\pi_i} = \min_{\pi_i} v_{\pi_i}$ from (A2))

$$v^\star_{\pi_i}(t_i, \boldsymbol{s}^{(t_i)}) = \min_{\pi_i} \Big[\mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \Big[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi) \Big] + v^\star_{\pi_i}(t_i + 1, \boldsymbol{s}^{(t_i+1)}) \Big]. \tag{A4}$$

Substituting $\boldsymbol{s}^{(t_i+1)} = h_{\mathcal{S}}(\boldsymbol{s}^{(t_i)}, \boldsymbol{\theta}^{(t)})$ and applying a first-order Taylor expansion of $v^\star_{\pi_i}(t_i + 1, \boldsymbol{s}^{(t_i+1)})$ around $\boldsymbol{s}^{(t_i)}$, we obtain

$$\begin{aligned} v^\star_{\pi_i}\big(t_i + 1,\ h_{\mathcal{S}}(\boldsymbol{s}^{(t_i)}, \boldsymbol{\theta}^{(t)})\big) &= v^\star_{\pi_i}(t_i + 1, \boldsymbol{s}^{(t_i)}) \\ &\quad + \Big\langle \nabla_{\boldsymbol{s}} v^\star_{\pi_i}(t_i + 1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \Big\rangle + \mathcal{O}\big(\|\Delta \boldsymbol{s}^{(t_i)}\|^2\big). \end{aligned} \tag{A5}$$

Dropping the higher-order remainder (valid when $\|\Delta \boldsymbol{s}^{(t_i)}\|$ is small, i.e., the memory update is a near-identity map), we obtain the result. □

Server-Side Discrete Bellman Equation: Similarly, the server observes client information only through the client memory $\boldsymbol{c}^{(t)}$ and seeks

$$v^\star_\pi(t, \boldsymbol{c}^{(t)}) = \min_\pi v_\pi(t, \boldsymbol{c}^{(t)}), \quad \text{such that} \quad \boldsymbol{c}^{(t+1)} = h_{\mathcal{C}}\big(\boldsymbol{c}^{(t)}, \{\boldsymbol{\theta}_i^{(t_i)} : i = 1, \cdots, K\}\big). \tag{A6}$$

Proposition 2 (Server-Side Discrete HJB)**.** *Let $v^\star_\pi(t, \boldsymbol{c}^{(t)})$ denote the solution to* (A6) *and define the discrete map $\boldsymbol{c}^{(t+1)} = h_{\mathcal{C}}\big(\boldsymbol{c}^{(t)}, \{\boldsymbol{\theta}_i^{(t_i)} : i = 1, \cdots, K\}\big)$ to imply transition of the client memory. Assume $v^\star_\pi(t + 1, \cdot)$ is differentiable with respect to $\mathbf{c}$ and define the memory increment $\Delta \boldsymbol{c}^{(t)} \coloneqq h_{\mathcal{C}}\big(\boldsymbol{c}^{(t)}, \{\boldsymbol{\theta}_i^{(t_i)} : i = 1, \cdots, K\}\big) - \boldsymbol{c}^{(t)}$. Then, $v^\star_\pi$ satisfies*

$$v^\star_\pi(t, \boldsymbol{c}^{(t)}) = \min_\pi \Big[f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) + v^\star_\pi(t + 1, \boldsymbol{c}^{(t)}) + \Big\langle \nabla_{\boldsymbol{c}} v^\star_\pi(t + 1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \Big\rangle \Big]. \tag{A7}$$

Proof. By Bellman's principle of optimality, the optimal cost-to-go decomposes as (note the future term uses $v^\star_\pi$, the optimal value, consistent with $v^\star_\pi = \min_\pi v_\pi$ from (A6))

$$v^\star_\pi(t, \boldsymbol{c}^{(t)}) = \min_\pi \Big[f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) + v^\star_\pi(t + 1, \boldsymbol{c}^{(t+1)}) \Big]. \tag{A8}$$

Substituting $\boldsymbol{c}^{(t+1)} = h_{\mathcal{C}}\big(\boldsymbol{c}^{(t)}, \{\boldsymbol{\theta}_i^{(t_i)} : i = 1, \cdots, K\}\big)$ and applying a first-order Taylor expansion of $v^\star_\pi(t + 1, \boldsymbol{c}^{(t+1)})$ around $\boldsymbol{c}^{(t)}$, we obtain

$$\begin{aligned} v^\star_\pi\big(t + 1,\ h_{\mathcal{C}}(\cdot)\big) &= v^\star_\pi(t + 1, \boldsymbol{c}^{(t)}) \\ &\quad + \Big\langle \nabla_{\boldsymbol{c}} v^\star_\pi(t + 1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \Big\rangle + \mathcal{O}\big(\|\Delta \boldsymbol{c}^{(t)}\|^2\big). \end{aligned} \tag{A9}$$

Dropping the higher-order remainder (valid when $\|\Delta \boldsymbol{c}^{(t)}\|$ is small, i.e., the memory update is a near-identity map) and substituting yields (A7). □

A2.3 Updates to the server and the client

Approximation to the updates: Here, we seek to approximate H_π and H_{π_i} to obtain practical updates to the server and client parameters. On the server side, the update at each t involves fetching the client pool for $\boldsymbol{c}^{(t)}$ and constructing the approximation for H_π is constructed using $\boldsymbol{c}^{(t)}$ and $\boldsymbol{\theta}^{(t)}$. This approximation at both client and server leads to

$$\begin{aligned} &\boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}^{(t)}} \mathrm{H}_\pi = \boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}^{(t)}} f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) - \alpha^{(t)} \nabla_{\boldsymbol{\theta}^{(t)}} \Big\langle \nabla_{\boldsymbol{c}} v^\star_\pi(t + 1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \Big\rangle \\ &\approx \mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] + \alpha^{(t)} \Big(\mathbb{E}_{\mathbb{P}}[\Delta \boldsymbol{c}^{(t)}] \Big). \qquad \text{(Server Update)} \\ &\boldsymbol{\theta}_i^{(t_i, j+1)} = \boldsymbol{\theta}_i^{(t_i, j)} - \eta^{(j)} \boldsymbol{g}_i^{(j)} - \underbrace{\beta^{(t_i, j)} \Big\langle \boldsymbol{J}^{(t_i, j)}, \Delta \boldsymbol{s}^{(t_i)} \Big\rangle}_{\text{Jacobian vector product}} \qquad \text{(Client Update)} \end{aligned}$$

$$\nabla_{\boldsymbol{\theta}^{(t)}} \mathrm{H}_\pi = \nabla_{\boldsymbol{\theta}^{(t)}} f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) + \nabla_{\boldsymbol{\theta}^{(t)}} \left\langle \nabla_{\boldsymbol{c}} v_\pi^\star(t+1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \right\rangle. \tag{A10}$$

With attention to the first term, the function $f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big)$ depends on $\boldsymbol{\theta}^{(t)}$ and therefore $\boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}^{(t)}} f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) \simeq \mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}]$, since the gradient of the total cost f is communicated to the server through $\boldsymbol{c}^{(t)}$. Next, the term $\nabla_{\boldsymbol{\theta}^{(t)}} \left\langle \nabla_{\boldsymbol{c}} v_\pi^\star(t+1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \right\rangle$ captures the change introduced into the policy by the client, with $\nabla_{\boldsymbol{c}} v_\pi^\star(t+1, \boldsymbol{c}^{(t)})$ playing the role of a Lagrange multiplier. We approximate this term as $\boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}^{(t)}} \left\langle \nabla_{\boldsymbol{c}} v_\pi^\star(t+1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \right\rangle \approx \alpha^{(t)} \left[\mathbb{E}_{\mathbb{P}}[\Delta \boldsymbol{c}^{(t)}]\right]$, where $\alpha^{(t)}$ takes the form of the Lagrange multiplier. In practice, one can set it with a hyperparameter of choice. Within these constructs, we obtain

$$\begin{aligned} \boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}^{(t)}} \mathrm{H}_\pi &= \boldsymbol{\theta}^{(t)} - \alpha^{(t)} \nabla_{\boldsymbol{\theta}^{(t)}} f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) - \alpha^{(t)} \nabla_{\boldsymbol{\theta}^{(t)}} \left\langle \nabla_{\boldsymbol{c}} v_\pi^\star(t+1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \right\rangle \\ &\approx \mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] + \alpha^{(t)} \left(\mathbb{E}_{\mathbb{P}}[\Delta \boldsymbol{c}^{(t)}]\right). \end{aligned} \tag{A11}$$

Similarly, the update on the client side follows as

$$\nabla_{\boldsymbol{\theta}_i^{(t_i)}} \mathrm{H}_{\pi_i} = \nabla_{\boldsymbol{\theta}_i^{(t_i)}} \left[\mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\right] + \left\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^\star(t_i+1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \right\rangle\right]. \tag{A12}$$

Similar to the server side, the client side polls the current server parameters at t_i through $\boldsymbol{s}^{(t_i)}$, uses the client data $\mathcal{D}_i^{(t_i)}$ to construct the approximation for H_{π_i}. Therefore, given the server pool, the first term in the bracket is provided through a gradient of $\mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\right]$ which we denote as $\boldsymbol{g}_i^{(j)}$. Now, the second term is provided through a Jacobian-vector product on $\mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\right]$ such that $\left\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^\star(t_i+1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \right\rangle \leq \left\langle \nabla_{\boldsymbol{s}^{(t_i)}} \mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\right],\ \Delta \boldsymbol{s}^{(t_i)} \right\rangle$, which provides the client update for the jth batch of data as

$$\nabla_{\boldsymbol{\theta}_i^{(t_i)}} \mathrm{H}_{\pi_i} \approx \eta^{(j)} \boldsymbol{g}_i^{(j)} + \beta^{(t_i,j)} \underbrace{\left\langle \nabla_{\boldsymbol{s}^{(t_i)}} \mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\right],\ \Delta \boldsymbol{s}^{(t_i)} \right\rangle}_{\text{Jacobian vector product}}$$

Notably, the data at the client side is consumed in batches and using $\boldsymbol{J}^{(t_i,j)}$ to denote $\nabla_{\boldsymbol{s}^{(t_i)}} \mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\right]$, we achieve the client update as

$$\boldsymbol{\theta}_i^{(t_i,j+1)} = \boldsymbol{\theta}_i^{(t_i,j)} - \eta^{(j)} \boldsymbol{g}_i^{(j)} - \underbrace{\beta^{(t_i,j)} \left\langle \boldsymbol{J}^{(t_i,j)}, \Delta \boldsymbol{s}^{(t_i)} \right\rangle}_{\text{Jacobian vector product}} \tag{A13}$$

A2.4 Approximation to the updates and algorithmic implementation

Proposition 3. *Given*

$$\begin{aligned} \mathrm{H}_\pi(\boldsymbol{c}, \boldsymbol{\theta}) &= f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) + \left\langle \nabla_{\boldsymbol{c}} v_\pi^\star(t+1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \right\rangle \\ \mathrm{H}_{\pi_i}(\boldsymbol{s}, \boldsymbol{\theta}_i) &= \mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\right] + \left\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^\star(t_i+1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \right\rangle. \end{aligned} \tag{A14}$$

and the following update rules for the server and client as $\nabla_{\boldsymbol{\theta}} \mathrm{H}_\pi(\boldsymbol{c}, \boldsymbol{\theta}), \nabla_{\boldsymbol{\theta}_i} \mathrm{H}_{\pi_i}(\boldsymbol{s}, \boldsymbol{\theta}_i)$ respectively. Then, the following is true for the client and server updates

$$\boldsymbol{\theta}^{(t+1)} = \mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] + \alpha^{(t)} \left[\mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] - \mathbb{E}_{\mathbb{P}}[\boldsymbol{s}^{(t-1)}]\right], \tag{Server Update}$$

$$\boldsymbol{\theta}_i^{(t_i,j+1)} = \boldsymbol{\theta}_i^{(t_i,j)} - \eta^{(j)} \boldsymbol{g}_i^{(j)} - \beta^{(t_i,j)} \left\langle \boldsymbol{J}^{(t_i,j)}, \Delta \boldsymbol{s}^{(t_i)} \right\rangle, \tag{Client Update}$$

where $\boldsymbol{g}_i^{(j)} = \mathbb{E}_{\xi \in \mathcal{D}_i^{(tt_i)}} \left[\ell(tt_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}; \xi)\right]$.

Proof. This proof is done in two parts. First we will show that the client update is a first-order approximation to the client Hamiltonian and then we will show that the server update is a first-order approximation to the server Hamiltonian.

(a) Client Side: Since,

$$\mathrm{H}_{\pi_i}(\boldsymbol{s}, \boldsymbol{\theta}_i) = \Big[\mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \Big[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\Big] + \Big\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^{\star}(t_i + 1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \Big\rangle \Big],$$

we have

$$\nabla_{\boldsymbol{\theta}_i} \mathrm{H}_{\pi_i}(\boldsymbol{s}, \boldsymbol{\theta}_i) = \nabla_{\boldsymbol{\theta}_i} \Big[\mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \Big[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\Big] + \Big\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^{\star}(t_i + 1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \Big\rangle \Big],$$

which provides

$$\nabla_{\boldsymbol{\theta}_i} \mathrm{H}_{\pi_i}(\boldsymbol{s}, \boldsymbol{\theta}_i) = \nabla_{\boldsymbol{\theta}_i} \Big[\mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \Big[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\Big] + \Big\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^{\star}(t_i + 1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \Big\rangle \Big] \tag{A15}$$

$$= \nabla_{\boldsymbol{\theta}_i} \Big[\mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \Big[\ell(t_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}^{(t_i)}; \xi)\Big] \Big] + \nabla_{\boldsymbol{\theta}_i} \Big[\Big\langle \nabla_{\boldsymbol{s}} v_{\pi_i}^{\star}(t_i + 1, \boldsymbol{s}^{(t_i)}),\ \Delta \boldsymbol{s}^{(t_i)} \Big\rangle \Big] \tag{A16}$$

Since, the problem is a minimization problem, the optimal value function is the lower bound of the current cost function. Therefore, for a feasible choice $\boldsymbol{\theta}_i^{(tt_i)} = \boldsymbol{\theta}_i^{(t_i)}$ for all $tt_i \geq t_i + 1$ yields, for every $\boldsymbol{s}$,

$$v_{\pi_i}^{\star}(t_i + 1, \boldsymbol{s}) \ \leq \ \sum_{tt_i = t_i + 1}^{T_i} \mathbb{E}_{\xi \in \mathcal{D}_i^{(tt_i)}} \big[\ell(tt_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}; \xi)\big]. \tag{A17}$$

Evaluated at $\boldsymbol{s} = \boldsymbol{s}^{(t_i)} + \Delta \boldsymbol{s}^{(t_i)}$ we obtain through finite differences that

$$\begin{aligned}
&\nabla_{\boldsymbol{s}} v_{\pi_i}^{\star}(t_i + 1, \boldsymbol{s}^{(t_i)}) \leq v_{\pi_i}^{\star}(t_i + 1, \boldsymbol{s}^{(t_i)} + \Delta \boldsymbol{s}^{(t_i)}) - v_{\pi_i}^{\star}(t_i + 1, \boldsymbol{s}^{(t_i)}) \\
&\leq \left[\sum_{tt_i = t_i + 1}^{T_i} \mathbb{E}_{\xi \in \mathcal{D}_i^{(tt_i)}} \big[\ell(tt_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s} + \Delta \boldsymbol{s}^{(t_i)}; \xi)\big] - \sum_{tt_i = t_i + 1}^{T_i} \mathbb{E}_{\xi \in \mathcal{D}_i^{(tt_i)}} \big[\ell(tt_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}; \xi)\big] \right] \\
&\approx \boldsymbol{J}^{(t_i, j)},
\end{aligned} \tag{A18}$$

where we have assume that the change between T_i and t_i can be approximated by the change between t_i and $t_i + 1$ and $\boldsymbol{J}^{(t_i, j)}$ is this one step approximation given by the Jacobian of the cost function calculated over an history at the local client with respect to the change in the server parameters. Therefore,

$$\Delta \boldsymbol{\theta}_i = -\eta^{(j)} \boldsymbol{g}_i^{(j)} - \beta^{(t_i, j)} \Big\langle \boldsymbol{J}^{(t_i, j)}, \Delta \boldsymbol{s}^{(t_i)} \Big\rangle \tag{A19}$$

where $\boldsymbol{g}_i^{(j)} = \mathbb{E}_{\xi \in \mathcal{D}_i^{(tt_i)}} \big[\ell(tt_i, \boldsymbol{\theta}_i^{(t_i)}, \boldsymbol{s}; \xi)\big]$.

(b) Server Side: We are given

$$\nabla_{\boldsymbol{\theta}} \Big[f\big(\boldsymbol{\theta}^{(t)}, \boldsymbol{c}^{(t)}\big) + \Big\langle \nabla_{\boldsymbol{c}} v_{\pi}^{\star}(t + 1, \boldsymbol{c}^{(t)}),\ \Delta \boldsymbol{c}^{(t)} \Big\rangle \Big] \tag{A20}$$

The first term is the approximation of the gradient of the cost function with respect to the server parameters. From traditional federated learning, this approximation is given by aggregating the client parameters which is equivalent to $\mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}]$. The second term is the approximation of the gradient of the optimal value function with respect to the change in the client memory. This provides with $v := \nabla_{\boldsymbol{c}} v_{\pi}^{\star}(t + 1, \boldsymbol{c}^{(t)})$ through the product rule as

$$\nabla_{\boldsymbol{\theta}^{(t)}} \langle v,\ \Delta \boldsymbol{c}^{(t)} \rangle \ = \ \big\langle \nabla_{\boldsymbol{\theta}^{(t)}} v,\ \Delta \boldsymbol{c}^{(t)} \big\rangle \ + \ \big\langle v,\ \nabla_{\boldsymbol{\theta}^{(t)}} \Delta \boldsymbol{c}^{(t)} \big\rangle. \tag{A21}$$

Both these factors are implicitly functions of $\boldsymbol{\theta}^{(t)}$ through $\boldsymbol{c}^{(t)}$, but, they are not explicit functions of $\boldsymbol{\theta}^{(t)}$. In particular, the second terms is the change in the client memory with respect to the server parameters. This is zero because the client memory is independent of the server parameters.

On the other hand, the first term is the change in the optimal value function with respect to the server parameters. This is equal to the jacobian vector product between gradient of the value function and the change in the client client memory. We therefore obtain

$$\nabla_{\boldsymbol{\theta}^{(t)}} \langle v,\, \Delta \boldsymbol{c}^{(t)} \rangle \;=\; \left\langle \nabla_{\boldsymbol{\theta}^{(t)}} v,\, \Delta \boldsymbol{c}^{(t)} \right\rangle \tag{A22}$$

$$\leq \sum_{tt=t}^{T} \left[f\big(\boldsymbol{\theta}^{(tt)} + \Delta \boldsymbol{\theta}^{(tt)}, \boldsymbol{c}^{(tt)}\big) - f\big(\boldsymbol{\theta}^{(tt)}, \boldsymbol{c}^{(tt)}\big) \right] \tag{A23}$$

The second term is given by the previous aggregation of the client parameters on the server side. The first terms is the change in the server parameter after the current aggregation to the client. These two facts provides by cauchy-Schwarz inequality that

$$\nabla_{\boldsymbol{\theta}^{(t)}} \langle v,\, \Delta \boldsymbol{c}^{(t)} \rangle \;=\; \left\langle \nabla_{\boldsymbol{\theta}^{(t)}} v,\, \Delta \boldsymbol{c}^{(t)} \right\rangle \tag{A24}$$

$$\leq \sum_{tt=t}^{T} \left[f\big(\boldsymbol{\theta}^{(tt)} + \Delta \boldsymbol{\theta}^{(tt)}, \boldsymbol{c}^{(tt)}\big) - f\big(\boldsymbol{\theta}^{(tt)}, \boldsymbol{c}^{(tt)}\big) \right] \qquad \leq\; \alpha^{(t)} \left[\mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] - \mathbb{E}_{\mathbb{P}}[\boldsymbol{s}^{(t)}] \right] \tag{A25}$$

$$\leq\; \alpha^{(t)} \left[\mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] - \mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] \right] \tag{A26}$$

$$\leq\; \alpha^{(t)} \left[\mathbb{E}_{\mathbb{P}}[\Delta \boldsymbol{c}^{(t)}] \right] \tag{A27}$$

where $\alpha^{(t)}$ is the step size for the server update and acts as a proxy to the Lipschitz constant. Moreover, the sum goes away due to telescopic sum and only the first and the last terms remain. Upon assuming that the change between T and tt can be approximated by the change between t and $t+1$, we obtain the server update as

$$\boldsymbol{\theta}(t+1) = \mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] + \alpha^{(t)} \left[\mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] - \mathbb{E}_{\mathbb{P}}[\boldsymbol{s}^{(t-1)}] \right] \tag{A28}$$

□

A2.5 Convergence Analysis

The main assumptions of the paper are

Assumption 1. *(i) For any $i \in [K]$, the local loss function ℓ_i is L-smooth but possibly nonconvex, with $\ell_i^* = \inf_{\boldsymbol{\theta}_i} \ell_i > -\infty$. (ii) For all $i \in [K]$, the local stochastic gradient oracle $\boldsymbol{g}_i^{(j)}$ is an unbiased estimator of $\nabla_{\boldsymbol{\theta}_i} \ell_i$, i.e., $\boldsymbol{g}_i^{(j)} = \nabla_{\boldsymbol{\theta}_i} \mathbb{E}_{\xi \in \mathcal{D}_i^{(t_i)}} \left[\ell(t_i, \boldsymbol{\theta}_i, \boldsymbol{s}^{(t_i)}; \xi) \right] = \nabla_{\boldsymbol{\theta}_i} \ell_i$. (iii) The gradients are uniformly bounded: $\left\| \boldsymbol{g}_i^{(j)} \right\| \leq g_M$ for all $i \in [K]$ and $j \in \{1, \ldots, E\}$.*

One could argue that the convergence of client in the presence of Eq. (Client Update) may not be guaranteed. However, note that, this convergence proof is trivial as the result is essentially that of stochastic gradient with regularization obtained under simple Lipchitz assumption. A precise convergence analysis for this can be found in the continual learning literature; see Krishnan et al. (Krishnan & Balaprakash, 2021). Given the assumptions in A. 1, we have the following assumptions on the server.

Assumption 2. *Let $f^* = \inf_{x \in \mathbb{R}^n} f(x) > -\infty$, where f denotes the server objective function. Then, the (ii) client contribution to the server is defined through the client probability space $\mathbb{P}$ with the contribution probabilities $p_i^{(t)}$ uniformly distributed. The serverpool size is one and $\mathbb{E}_{\mathbb{P}} \left[\boldsymbol{\theta}^{(t-1)} \right] = \boldsymbol{\theta}^{(t-1)}$ and and (iv) $\boldsymbol{c}$ and $\boldsymbol{s}$ satisfies L-smoothness with respect to f.*

Then, we will also assume that the communication required to transfer the client/server parameters is bounded by a constant ϵ_B as in A. 3.

Assumption 3. *Consider Eq.* (CFL)*, Algorithm 1, and Algorithm 2, and assume, (i) there is a client pool $\boldsymbol{c}$ that satisfies $\mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] = \mathbb{E}_{\mathbb{P}}[\boldsymbol{\theta}_i^{(t)} + \epsilon^{(t,t_i)}]$. (ii) There is a server pool $\boldsymbol{s}$ that satisfies $\mathbb{E}_{\mathbb{P}}[\boldsymbol{s}^{(t)}] = \mathbb{E}_{\mathbb{P}}[\boldsymbol{\theta}^{(t)} + \epsilon^{(t,t_i)}]$ where $\epsilon^{(t,t_i)}$ is the worst case error. (iii) We assume that $\left\| \epsilon^{(t,t_i)} \right\| \leq \epsilon_B, \forall i$.*

A2.5.1 First difference on the server side

We summarize this change with the next lemma.

Lemma 2. *Consider Eq.* (CFL) *and let A. 3, A. 1 and A. 2 be true. Let the server updates be given as in* (Server Update) *and the client update be given as in* (Client Update)*. Then, it follows that the first difference of the server update with respect to server time* t *is given as*

$$\mathbb{E}_{\mathbb{P}}\left[\Delta\,\boldsymbol{\theta}^{(t)}\right] = \frac{-1}{1-\alpha^{(t)}}\left(\sum_{j=1}^{E}\eta^{(j)}\mathbb{E}_{\mathbb{P}}\left[\boldsymbol{g}_i^{(j)}\right] + \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta\boldsymbol{s}^{(t_i)}\right\rangle\right] - 2\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] - \alpha^{(t)}\Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right) \tag{A29}$$

where $\boldsymbol{J}^{(t_i,j)} = \nabla_{\boldsymbol{s}^{(t_i)}}\mathbb{E}_{\xi\in\mathcal{D}_i^{(t_i)}}\left[\ell(t_i,\boldsymbol{\theta}_i^{(t_i)},\boldsymbol{s}^{(t_i)};\xi)\right]$ *is the Jacobian,* $\beta^{(t_i,j)}$ *refers to some regularization factor at the client with* $\alpha^{(t)}$ *and* $\eta^{(j)}$*being the learning rate at the server and the client respectively.*

Proof. From (Server Update) and A. 3 we have

$$\boldsymbol{\theta}^{(t)} = \mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] + \alpha^{(t)}\left(\mathbb{E}_{\mathbb{P}}[\Delta\boldsymbol{c}^{(t)}]\right) \quad \text{where} \quad \mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] = \mathbb{E}_{\mathbb{P}}[\boldsymbol{\theta}_i^{(t)} + \epsilon^{(t,t_i)}] \tag{A30}$$

From the client updates as in (Client Update) we have

$$\boldsymbol{\theta}_i^{(t_i,E)} = \boldsymbol{\theta}_i^{(t_i,0)} - \sum_{j=1}^{E}\left[\eta^{(j)}\boldsymbol{g}_i^{(j)} + \beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta\boldsymbol{s}^{(t_i)}\right\rangle\right], \tag{A31}$$

where E is the total number of local client updates. Substitute into (A30) to achieve

$$\begin{aligned}\mathbb{E}_{\mathbb{P}}[\boldsymbol{c}^{(t)}] &= \mathbb{E}_{\mathbb{P}}\left[\boldsymbol{\theta}_i^{(t_i,0)} - \sum_{j=1}^{E}\left[\eta^{(j)}\boldsymbol{g}_i^{(j)} + \beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta\boldsymbol{s}^{(t_i)}\right\rangle\right]\right] + \mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \\ &= \mathbb{E}_{\mathbb{P}}\left[\boldsymbol{\theta}_i^{(t_i,0)}\right] - \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\eta^{(j)}\boldsymbol{g}_i^{(j)}\right] - \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta\boldsymbol{s}^{(t_i)}\right\rangle\right] + \mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\end{aligned}$$

Therefore,

$$\boldsymbol{\theta}^{(t)} = \alpha^{(t)}\left(\mathbb{E}_{\mathbb{P}}[\Delta\boldsymbol{c}^{(t)}]\right) + \mathbb{E}_{\mathbb{P}}\left[\boldsymbol{\theta}_i^{(t_i,0)}\right] - \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\eta^{(j)}\boldsymbol{g}_i^{(j)}\right] - \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta\boldsymbol{s}^{(t_i)}\right\rangle\right] + \mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \tag{A32}$$

Each client i initializes from the server pool: $\boldsymbol{\theta}_i^{(t_i,0)} \leftarrow \boldsymbol{s}_i^{(t-1)}$. Since the server pool size is one (A. 2) and the server updates every communication round, the server parameter is deterministic: $\mathbb{E}_{\mathbb{P}}[\boldsymbol{\theta}^{(t-1)}] = \boldsymbol{\theta}^{(t-1)}$. Taking expectations over the client probability space $\mathbb{P}$ with uniform weights $p_i^{(t)} = 1/K$ and applying A. 3, we obtain $\mathbb{E}_{\mathbb{P}}[\boldsymbol{\theta}_i^{(t_i,0)}] = \boldsymbol{\theta}^{(t-1)} + \mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]$, which gives

$$\boldsymbol{\theta}^{(t)} = \alpha^{(t)}\left(\mathbb{E}_{\mathbb{P}}[\Delta\boldsymbol{c}^{(t)}]\right) + \mathbb{E}_{\mathbb{P}}\left[\boldsymbol{\theta}^{(t-1)} + \mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right] - \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\eta^{(j)}\boldsymbol{g}_i^{(j)}\right] - \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta\boldsymbol{s}^{(t_i)}\right\rangle\right] + \mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \tag{A33}$$

Since, $\mathbb{E}_{\mathbb{P}}\left[\boldsymbol{\theta}^{(t-1)}\right] = \boldsymbol{\theta}^{(t-1)}$ by the fact that the serverpool size is one, we obtain

$$\boldsymbol{\theta}^{(t)} = \alpha^{(t)}\left(\mathbb{E}_{\mathbb{P}}[\Delta \boldsymbol{c}^{(t)}]\right) + \boldsymbol{\theta}^{(t-1)} + \mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] - \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\eta^{(j)}\boldsymbol{g}_i^{(j)}\right] - \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta \boldsymbol{s}^{(t_i)}\right\rangle\right] + \mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}], \tag{A34}$$

where rearrangement provides

$$\begin{aligned}\Delta\,\boldsymbol{\theta}^{(t)} &= \boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^{(t-1)} \\ &= \alpha^{(t)}\left(\mathbb{E}_{\mathbb{P}}[\Delta \boldsymbol{c}^{(t)}]\right) + 2\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] - \sum_{j=1}^{E}\eta^{(j)}\mathbb{E}_{\mathbb{P}}\left[\boldsymbol{g}_i^{(j)}\right] \\ &\quad - \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta \boldsymbol{s}^{(t_i)}\right\rangle\right]\end{aligned} \tag{A35}$$

where A. 3 provides $\mathbb{E}_{\mathbb{P}}[\Delta \boldsymbol{c}^{(t)}] = \mathbb{E}_{\mathbb{P}}[\Delta \boldsymbol{\theta}_i^{(t)}] = \Delta\boldsymbol{\theta} + \mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]$ and leads to the result as

$$\begin{aligned}\mathbb{E}_{\mathbb{P}}\left[\Delta\,\boldsymbol{\theta}^{(t)}\right] &= \Delta\,\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^{(t-1)} \\ &= \frac{-1}{1-\alpha^{(t)}}\Big(- 2\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] - \alpha^{(t)}\Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] + \sum_{j=1}^{E}\eta^{(j)}\mathbb{E}_{\mathbb{P}}\left[\boldsymbol{g}_i^{(j)}\right] \\ &\quad + \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta \boldsymbol{s}^{(t_i)}\right\rangle\right]\Big)\end{aligned} \tag{A36}$$

□

A2.5.2 Upper Bound on the first difference

Lemma 3. *Consider Eq.* (CFL) *and let A. 1, A. 2, and A. 3 hold. Define the server and client updates as in* (Server Update) *and* (Client Update) *with the first difference given by Lem. 2. Assume the well-posedness condition*

$$\mathrm{A} \coloneqq \frac{1}{(1-\alpha^{(t)})K - \sum_{j=1}^{E}\beta^{(t_i,j)}\rho_{\max}}, \tag{A37}$$

where $0 < \rho_{\max} < 1$ *is the spectral norm bound on the Jacobian* $\boldsymbol{J}^{(t_i,j)}$*. Define*

$$\mathrm{B} \coloneqq \frac{2\epsilon_B}{K} + \frac{g_M}{K}\left\|\sum_{j=1}^{E}\eta^{(j)}\right\| + \frac{\Delta_{\epsilon_B}}{K}\left[\alpha^{(t)} + \sum_{j=1}^{E}\beta^{(t_i,j)}\rho_{\max}\right], \tag{A38}$$

where $\epsilon_B \geq K\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\|$ *and* $\Delta_{\epsilon_B} \geq K\left\|\mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]\right\|$ *are the* K*-aggregate communication error bounds,* $g_M \geq \left\|\boldsymbol{g}_i^{(j)}\right\|$ *for all* i, j*. Then,*

$$\left\|\Delta\,\boldsymbol{\theta}^{(t)}\right\| \leq \mathrm{A}\cdot\mathrm{B}. \tag{A39}$$

Proof. From Lem. 2, applying the triangle inequality to split the norm of the sum, Jensen's inequality ($\|\mathbb{E}_{\mathbb{P}}[X]\| \leq \mathbb{E}_{\mathbb{P}}[\|X\|]$) to move norms inside expectations, and Cauchy–Schwarz to bound the

Jacobian-vector product,

$$\begin{aligned}\left\|\Delta\,\boldsymbol{\theta}^{(t)}\right\| &\le \frac{1}{1-\alpha^{(t)}}\Big(2\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| + \alpha^{(t)}\left\|\Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| + \left\|\sum_{j=1}^{E}\eta^{(j)}\mathbb{E}_{\mathbb{P}}\left[\boldsymbol{g}_i^{(j)}\right]\right\| \\ &\quad + \left\|\sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta\boldsymbol{s}^{(t_i)}\right\rangle\right]\right\|\Big) \\ &\le \frac{1}{1-\alpha^{(t)}}\Big(2\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| + \alpha^{(t)}\left\|\Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| + \left\|\sum_{j=1}^{E}\eta^{(j)}\mathbb{E}_{\mathbb{P}}\left[\boldsymbol{g}_i^{(j)}\right]\right\| \\ &\quad + \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\rho_{\max}\left\|\Delta\boldsymbol{s}^{(t_i)}\right\|\right]\Big)\end{aligned}$$

where the second inequality uses $\left\|\left\langle \boldsymbol{J}^{(t_i,j)}, \Delta\boldsymbol{s}^{(t_i)}\right\rangle\right\| \le \left\|\boldsymbol{J}^{(t_i,j)}\right\|_{\mathrm{op}}\left\|\Delta\boldsymbol{s}^{(t_i)}\right\| \le \rho_{\max}\left\|\Delta\boldsymbol{s}^{(t_i)}\right\|$ via the Cauchy–Schwarz inequality and the spectral norm bound $\left\|\boldsymbol{J}^{(t_i,j)}\right\|_{\mathrm{op}} \le \rho_{\max}$. Therefore,

$$\begin{aligned}\left\|\Delta\,\boldsymbol{\theta}^{(t)}\right\| \le \frac{1}{1-\alpha^{(t)}}\Big(&2\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| + \alpha^{(t)}\left\|\Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| + \left\|\sum_{j=1}^{E}\eta^{(j)}\mathbb{E}_{\mathbb{P}}\left[\boldsymbol{g}_i^{(j)}\right]\right\| \\ &+ \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\rho_{\max}\left\|\Delta\boldsymbol{\theta}^{(t)}\right\|\right] + \sum_{j=1}^{E}\mathbb{E}_{\mathbb{P}}\left[\beta^{(t_i,j)}\rho_{\max}\left\|\Delta\epsilon^{(t,t_i)}\right\|\right]\Big)\end{aligned} \tag{A40}$$

Since the server pool has size one (A. 2), $\boldsymbol{\theta}^{(t)}$ is deterministic. Consequently, A. 3(ii) holds almost surely: $\boldsymbol{s}^{(t_i)} = \boldsymbol{\theta}^{(t)} + \epsilon^{(t,t_i)}$ a.s. for each client i, so $\Delta\boldsymbol{s}^{(t_i)} = \Delta\boldsymbol{\theta}^{(t)} + \Delta\epsilon^{(t,t_i)}$ a.s. Applying the triangle inequality gives $\left\|\Delta\boldsymbol{s}^{(t_i)}\right\| \le \left\|\Delta\boldsymbol{\theta}^{(t)}\right\| + \left\|\Delta\epsilon^{(t,t_i)}\right\|$ a.s., and since $\boldsymbol{\theta}^{(t)}$ is deterministic, $\mathbb{E}_{\mathbb{P}}[\left\|\Delta\boldsymbol{\theta}^{(t)}\right\|] = \left\|\Delta\boldsymbol{\theta}^{(t)}\right\|$.

We introduce two communication bounds, both K-aggregates by the same convention.

(i') **Aggregate error bound:** $\epsilon_B \ge K\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\|$, equivalently $\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| \le \epsilon_B/K$. Here $\epsilon_B = \sum_{i=1}^{K}\epsilon_B^{(i)}$ is a K-aggregate of per-client bounds; the averaged quantity $\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| = \left\|(1/K)\sum_i \epsilon_i^{(t,t_i)}\right\|$ picks up the $1/K$ factor.

(ii') **Change bound:** $\Delta_{\epsilon_B} \ge K\left\|\mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]\right\|$, equivalently $\left\|\mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]\right\| \le \Delta_{\epsilon_B}/K$. Here $\Delta_{\epsilon_B} = \sum_{i=1}^{K}\Delta_{\epsilon_B}^{(i)}$ is likewise a K-aggregate; the averaged change $\left\|\mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]\right\| = \left\|(1/K)\sum_i \Delta\epsilon_i^{(t,t_i)}\right\|$ carries the same $1/K$.

Both bounds share the same K-aggregate structure because both $\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]$ and $\mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]$ are averages over K clients. Applying A. 1(iii), $\left\|\mathbb{E}_{\mathbb{P}}[\boldsymbol{g}_i^{(j)}]\right\| \le g_M$, we obtain

$$\begin{aligned}\left\|\Delta\,\boldsymbol{\theta}^{(t)}\right\| \le \frac{1}{1-\alpha^{(t)}}\Big(&\frac{2\epsilon_B}{K} + \frac{\alpha^{(t)}\Delta_{\epsilon_B}}{K} + \frac{g_M}{K}\left\|\sum_{j=1}^{E}\eta^{(j)}\right\| + \frac{1}{K}\sum_{j=1}^{E}\beta^{(t_i,j)}\rho_{\max}\left\|\Delta\boldsymbol{\theta}^{(t)}\right\| \\ &+ \frac{\Delta_{\epsilon_B}}{K}\sum_{j=1}^{E}\beta^{(t_i,j)}\rho_{\max}\Big)\end{aligned} \tag{A41}$$

which simplifies on rearrangement to

$$\left\|\Delta\,\boldsymbol{\theta}^{(t)}\right\|\left(1-\frac{1}{1-\alpha^{(t)}}\frac{1}{K}\sum_{j=1}^{E}\beta^{(t_i,j)}\rho_{\max}\right)\leq\frac{1}{1-\alpha^{(t)}}$$
$$\left(\frac{2\epsilon_B}{K}+\frac{\alpha^{(t)}\Delta_{\epsilon_B}}{K}+\frac{g_M}{K}\left\|\sum_{j=1}^{E}\eta^{(j)}\right\|+\frac{\Delta_{\epsilon_B}}{K}\sum_{j=1}^{E}\beta^{(t_i,j)}\rho_{\max}\right)\tag{A42}$$

which provides upon simplification

$$\left\|\Delta\,\boldsymbol{\theta}^{(t)}\right\|\times\frac{\left((1-\alpha^{(t)})K-\sum_{j=1}^{E}\beta^{(t_i,j)}\rho_{\max}\right)}{1-\alpha^{(t)}}\leq\frac{1}{1-\alpha^{(t)}}$$
$$\left(\frac{2\epsilon_B}{K}+\frac{\alpha^{(t)}\Delta_{\epsilon_B}}{K}+\frac{g_M}{K}\left\|\sum_{j=1}^{E}\eta^{(j)}\right\|+\frac{\Delta_{\epsilon_B}}{K}\sum_{j=1}^{E}\beta^{(t_i,j)}\rho_{\max}\right)\tag{A43}$$

Since $1/A>0$ by hypothesis (A37), we may divide both sides by $1/A(1-\alpha^{(t)})$ to obtain

$$\left\|\Delta\,\boldsymbol{\theta}^{(t)}\right\|\leq A\Big(\frac{2\epsilon_B}{K}+\frac{g_M}{K}\left\|\sum_{j=1}^{E}\eta^{(j)}\right\|+\frac{\Delta_{\epsilon_B}}{K}\left[\alpha^{(t)}+\sum_{j=1}^{E}\beta^{(t_i,j)}\rho_{\max}\right]\Big).\tag{A44}$$

Remark (well-posedness under the learning rate schedule). Under the schedule $\beta^{(t_i,j)}=1/(E^2T)$ and $(1-\alpha^{(t)})=1/(KET)$, we have $1/A=(1-\rho_{\max})/(ET)>0$ precisely when $\rho_{\max}<1$, which is assumed throughout.

□

A2.5.3 Minimum value of the Grad

Theorem 2. *Consider Eq.* (CFL) *and let A. 1, A. 3 and A. 2 hold. Define the server and client updates as in* (Server Update) *and* (Client Update)*, with* A *and* B *as in Lem. 3. Then,*

$$\min_{t\in\{0,1,2,\cdots,T\}}\left(\mathbb{E}_{\mathbb{P}}\left[\left\|\nabla_{\boldsymbol{\theta}^{(t)}}f(\boldsymbol{\theta}^{(t)},\boldsymbol{s}^{(t-1)})\right\|^2\right]\right)\tag{A45}$$
$$\leq\frac{2}{\mathrm{A}}\left(\mathbb{E}_{\mathbb{P}}\left[f(\boldsymbol{\theta}^{(0)},\boldsymbol{s}^{(0)})-f^*\right]+L_{\epsilon_B}\left(\frac{\Delta_{\epsilon_B}}{K}\right)+L_{\boldsymbol{s}}\left(\frac{\Delta_{\epsilon_B}}{K}\right)^2\right)$$
$$+2\left(L_{\epsilon_B}+2L_{\boldsymbol{s}}\left(\frac{\Delta_{\epsilon_B}}{K}\right)\right)\mathrm{B}$$
$$+2\left(L_{\boldsymbol{\theta}}+L_{\boldsymbol{s}}\right)\mathrm{AB}^2\tag{A46}$$
$$+\mathrm{B}^2,$$

which in simplified form is given as

$$\min_{t\in\{0,1,2,\cdots,T\}}\left(\mathbb{E}_{\mathbb{P}}\left[\left\|\nabla_{\boldsymbol{\theta}^{(t)}}f(\boldsymbol{\theta}^{(t)},\boldsymbol{s}^{(t-1)})\right\|^2\right]\right)\leq\underbrace{\frac{2}{\mathrm{A}}\,\mathbb{E}_{\mathbb{P}}\left[f(\boldsymbol{\theta}^{(0)},\boldsymbol{s}^{(0)})-f^*\right]}_{\text{(I) Optimization}}$$
$$+\underbrace{2L_{\epsilon_B}\mathrm{B}}_{\text{(II) Comm. bias}}+\underbrace{2(L_{\boldsymbol{\theta}}+L_{\boldsymbol{s}})\,\mathrm{AB}^2}_{\text{(III) Curvature}}+\underbrace{\mathcal{O}(\Delta_{\epsilon_B}/K)}_{\text{(IV) Residual}}.$$

with A *being a function of* $\alpha^{(t)},\beta^{(t_i,j)},\rho_{\max})$ *and* B *being a function of* $\epsilon_B,\Delta_{\epsilon_B},\eta^{(j)}$.

Proof. By A. 2, $f(\boldsymbol{\theta}^{(t)},\boldsymbol{s}^{(t)})$ is separately $L_{\boldsymbol{\theta}}$-smooth in $\boldsymbol{\theta}$ and $L_{\boldsymbol{s}}$-smooth in $\boldsymbol{s}$. Applying the descent lemma to each argument,

$$f(\boldsymbol{\theta}^{(t)},\boldsymbol{s}^{(t)})\leq f(\boldsymbol{\theta}^{(t-1)},\boldsymbol{s}^{(t-1)})+\left\langle\nabla_{\boldsymbol{\theta}}f,\Delta\boldsymbol{\theta}^{(t)}\right\rangle$$
$$+\left\langle\nabla_{\boldsymbol{s}}f,\Delta\boldsymbol{s}^{(t)}\right\rangle+\frac{L_{\boldsymbol{\theta}}}{2}\left\|\Delta\boldsymbol{\theta}^{(t)}\right\|^2+\frac{L_{\boldsymbol{s}}}{2}\left\|\Delta\boldsymbol{s}^{(t)}\right\|^2,\tag{A47}$$

where we write $\nabla_{\boldsymbol{\theta}} f$ for $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t-1)})$ and similarly for $\nabla_{\boldsymbol{s}} f$. Here $L_{\boldsymbol{s}}$ and $L_{\boldsymbol{\theta}}$ are the Lipschitz gradient constants with respect to $\boldsymbol{s}$ and $\boldsymbol{\theta}$ respectively (A. 2). Rearranging,

$$f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t)}) - f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t-1)}) \leq \left\langle \nabla_{\boldsymbol{\theta}} f, \Delta\boldsymbol{\theta}^{(t)} \right\rangle + \left\langle \nabla_{\boldsymbol{s}} f, \Delta\boldsymbol{s}^{(t)} \right\rangle + \frac{L_{\boldsymbol{\theta}}}{2} \left\| \Delta\boldsymbol{\theta}^{(t)} \right\|^2 + \frac{L_{\boldsymbol{s}}}{2} \left\| \Delta\boldsymbol{s}^{(t)} \right\|^2. \tag{A48}$$

Note however that, as a consequence of A. 3, we have $\mathbb{E}_{\mathbb{P}}[\Delta\boldsymbol{s}^{(t)}] = \Delta\boldsymbol{\theta}^{(t)} + \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]$ and $\mathbb{E}_{\mathbb{P}}[\boldsymbol{s}^{(t)}] = \boldsymbol{\theta}^{(t)} + \mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]$ These facts provide

$$\begin{aligned} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t)}) - f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t-1)}) &\leq \left\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)}), \Delta\boldsymbol{\theta}^{(t)} \right\rangle \\ &+ \left\langle \nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\boldsymbol{\theta}^{(t)} \right\rangle + \left\langle \nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\rangle \\ &+ \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\boldsymbol{\theta}^{(t)} \right\rangle + \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\rangle \\ &+ L_{\boldsymbol{\theta}} \left\| \Delta\boldsymbol{\theta}^{(t)} \right\|^2 + L_{\boldsymbol{s}} \left\| \Delta\boldsymbol{\theta}^{(t)} + \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\|^2, \end{aligned}$$

Notably, $\left\langle \nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\boldsymbol{\theta}^{(t)} \right\rangle$ and $\left\langle \nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\rangle$ vanish because $f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)})$ is evaluated at $\boldsymbol{\theta}^{(t-1)}$, which does not depend on $\boldsymbol{\theta}^{(t)}$, hence $\nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}) = \mathbf{0}$. This provides

$$\begin{aligned} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t)}) - f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t-1)}) &\leq \left\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)}), \Delta\boldsymbol{\theta}^{(t)} \right\rangle \\ &+ \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\boldsymbol{\theta}^{(t)} \right\rangle + \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\rangle \\ &+ L_{\boldsymbol{\theta}} \left\| \Delta\boldsymbol{\theta}^{(t)} \right\|^2 + L_{\boldsymbol{s}} \left\| \Delta\boldsymbol{\theta}^{(t)} + \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\|^2, \end{aligned}$$

$$\Delta\boldsymbol{\theta}^{(t)} = -\big(\mathcal{G}^{(t)} + \mathcal{Q}^{(t)}\big) \tag{A49}$$

where $\mathcal{G}^{(t)} := \frac{1}{1-\alpha^{(t)}} \sum_j \eta^{(j)}\, \mathbb{E}_{\mathbb{P}}[\boldsymbol{g}_i^{(j)}]$ and $\mathcal{Q}^{(t)} := \frac{1}{1-\alpha^{(t)}} \Big(\sum_j \mathbb{E}_{\mathbb{P}}[\beta^{(t_i,j)} \left\langle \boldsymbol{J}^{(t_i,j)}, \Delta\boldsymbol{s}^{(t_i)} \right\rangle] - 2\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] - \alpha^{(t)} \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \Big)$. Substituting from (A49) provides

$$\begin{aligned} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t)}) - f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t-1)}) &\leq \left\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)}), -\big(\mathcal{G}^{(t)} + \mathcal{Q}^{(t)}\big) \right\rangle \\ &+ \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\boldsymbol{\theta}^{(t)} \right\rangle + \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\rangle \\ &+ L_{\boldsymbol{\theta}} \left\| \Delta\boldsymbol{\theta}^{(t)} \right\|^2 + L_{\boldsymbol{s}} \left\| \Delta\boldsymbol{\theta}^{(t)} + \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\|^2, \end{aligned}$$

which gives

$$\begin{aligned} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t)}) - f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t-1)}) &\leq \left\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)}), -\mathcal{G}^{(t)} \right\rangle + \left\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)}), -\mathcal{Q}^{(t)} \right\rangle \\ &+ \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\boldsymbol{\theta}^{(t)} \right\rangle + \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\rangle \\ &+ L_{\boldsymbol{\theta}} \left\| \Delta\boldsymbol{\theta}^{(t)} \right\|^2 + L_{\boldsymbol{s}} \left\| \Delta\boldsymbol{\theta}^{(t)} + \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\|^2, \end{aligned}$$

Rearranging then gives

$$\begin{aligned} \left\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)}), \mathcal{G}^{(t)} \right\rangle &\leq \Big[f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t-1)}) - f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t)}) \Big] - \left\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)}), \mathcal{Q}^{(t)} \right\rangle \\ &+ \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\boldsymbol{\theta}^{(t)} \right\rangle + \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)}), \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\rangle \\ &+ L_{\boldsymbol{\theta}} \left\| \Delta\boldsymbol{\theta}^{(t)} \right\|^2 + L_{\boldsymbol{s}} \left\| \Delta\boldsymbol{\theta}^{(t)} + \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\|^2. \end{aligned}$$

Therefore, By A. 2(v), $\langle \nabla_{\boldsymbol{\theta}} f, \mathcal{G}^{(t)} \rangle = \eta_{\text{eff}} \left\langle \nabla_{\boldsymbol{\theta}} f, \mathbb{E}_{\mathbb{P}}[\boldsymbol{g}_i^{(j)}] \right\rangle = \eta_{\text{eff}} \|\nabla_{\boldsymbol{\theta}} f\|^2$. Bounding $-\langle \nabla_{\boldsymbol{\theta}} f, \mathcal{Q}^{(t)} \rangle \leq \|\nabla_{\boldsymbol{\theta}} f\| \|\mathcal{Q}^{(t)}\|$ and applying Young, $\|\nabla_{\boldsymbol{\theta}} f\| \|\mathcal{Q}^{(t)}\| \leq \frac{\eta_{\text{eff}}}{2} \|\nabla_{\boldsymbol{\theta}} f\|^2 + \frac{1}{2\eta_{\text{eff}}} \|\mathcal{Q}^{(t)}\|^2$, then moving $\frac{\eta_{\text{eff}}}{2} \|\nabla_{\boldsymbol{\theta}} f\|^2$ left,

$$\begin{aligned}\frac{\eta_{\text{eff}}}{2} \left\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)})\right\|^2 \leq \left[f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t-1)}) - f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t)})\right] + \frac{1}{2\eta_{\text{eff}}} \left\|\mathcal{Q}^{(t)}\right\|^2 \\ + \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f, \Delta\boldsymbol{\theta}^{(t)} \right\rangle + \left\langle \nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f, \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}] \right\rangle \\ + L_{\boldsymbol{\theta}} \left\|\Delta\boldsymbol{\theta}^{(t)}\right\|^2 + L_{\boldsymbol{s}} \left\|\Delta\boldsymbol{\theta}^{(t)} + \Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\|^2.\end{aligned}$$

Use Lem. 3 to bound the magnitudes $\left\|\Delta\boldsymbol{\theta}^{(t)}\right\| \leq \text{AB}$, $\left\|\mathcal{Q}^{(t)}\right\| \leq \text{AB}$, $\left\|\Delta\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| \leq \Delta_{\epsilon_B}/K$, together with the L_{ϵ_B}-Lipschitz bound $\left\|\nabla_{\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]} f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t)})\right\| \leq L_{\epsilon_B}$ (A. 2) and collecting like terms to obtain

$$\begin{aligned}\frac{\eta_{\text{eff}}}{2} \left\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)})\right\|^2 \leq \left[f(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{s}^{(t-1)}) - f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t)})\right] + L_{\epsilon_B} \text{AB} + L_{\epsilon_B} \frac{\Delta_{\epsilon_B}}{K} \\ + \left(L_{\boldsymbol{\theta}} + L_{\boldsymbol{s}} + \frac{1}{2\eta_{\text{eff}}}\right)\text{A}^2\text{B}^2 + 2L_{\boldsymbol{s}} \text{AB} \frac{\Delta_{\epsilon_B}}{K} + L_{\boldsymbol{s}} \left(\frac{\Delta_{\epsilon_B}}{K}\right)^2.\end{aligned}$$

Taking expectation over the client probability space and summing from $0 \to T$ (the differences telescope once, so the leading term carries coefficient 1), using $\min_t \leq \frac{1}{T}\sum_t$, and dividing by $\frac{\eta_{\text{eff}} T}{2}$,

$$\begin{aligned}&\frac{\eta_{\text{eff}} T}{2} \min_{t \in \{0,1,2,\cdots,T\}} \left(\mathbb{E}_{\mathbb{P}}\left[\left\|\nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)})\right\|^2\right]\right) \\ &\leq \mathbb{E}_{\mathbb{P}}\left[f(\boldsymbol{\theta}^{(0)}, \boldsymbol{s}^{(0)}) - f(\boldsymbol{\theta}^{(T)}, \boldsymbol{s}^{(t)})\right] + T\left(L_{\epsilon_B} \text{AB} + L_{\epsilon_B} \tfrac{\Delta_{\epsilon_B}}{K}\right) \\ &\quad + T\left(L_{\boldsymbol{\theta}} + L_{\boldsymbol{s}} + \tfrac{1}{2\eta_{\text{eff}}}\right)\text{A}^2\text{B}^2 + 2TL_{\boldsymbol{s}} \text{AB} \tfrac{\Delta_{\epsilon_B}}{K} + TL_{\boldsymbol{s}} \left(\tfrac{\Delta_{\epsilon_B}}{K}\right)^2.\end{aligned}$$

Combining with $f^* = \inf_{\boldsymbol{\theta} \in \Theta} f(\boldsymbol{\theta}, \boldsymbol{s}) > -\infty$ (A. 2) and collecting terms provides the result as

$$\begin{aligned}&\min_{t \in \{0,1,2,\cdots,T\}} \left(\mathbb{E}_{\mathbb{P}}\left[\left\|\nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)})\right\|^2\right]\right) \\ &\leq \frac{2}{\eta_{\text{eff}} T} \mathbb{E}_{\mathbb{P}}\left[f(\boldsymbol{\theta}^{(0)}, \boldsymbol{s}^{(0)}) - f^*\right] + \frac{2}{\eta_{\text{eff}}}\left(L_{\epsilon_B} \text{AB} + L_{\epsilon_B} \tfrac{\Delta_{\epsilon_B}}{K} + 2L_{\boldsymbol{s}} \text{AB} \tfrac{\Delta_{\epsilon_B}}{K} + L_{\boldsymbol{s}} \left(\tfrac{\Delta_{\epsilon_B}}{K}\right)^2\right) \\ &\quad + \frac{2}{\eta_{\text{eff}}}\left(L_{\boldsymbol{\theta}} + L_{\boldsymbol{s}} + \tfrac{1}{2\eta_{\text{eff}}}\right)\text{A}^2\text{B}^2,\end{aligned}$$

where we assume that the infimum is achieved by repeated updates. The explicit convergence rate follows from instantiating A, B, and η_{eff} via the schedule in the convergence proof. □

A2.6 Communication error due to endogenous and exogenous heterogeneity

Assumption 4 (Communication Error Structure). *The communication error decomposes as $\epsilon^{(t,t_i)} = \epsilon^{\text{exo},(t,t_i)} + \epsilon^{\text{endo},(t,t_i)}$, where ϵ^{exo} and ϵ^{endo} represent the exogenous and endogenous components respectively, satisfying: (i) **Bounded staleness:** Each client i has delay $\tau_i \leq \tau_{\max}$; (ii) **Angular divergence:** $\langle \nabla_{\boldsymbol{\theta}_i} \ell_i, \nabla_{\boldsymbol{\theta}} f \rangle \geq \delta_{\min} \|\nabla_{\boldsymbol{\theta}_i} \ell_i\| \|\nabla_{\boldsymbol{\theta}} f\|, \forall\, i \in [K]$ with $\delta_{\min} \in (-1, 1]$; (iii) **Magnitude variance:** $\mathbb{E}_{\mathbb{P}}\left[\left(\|\nabla_{\boldsymbol{\theta}_i} \ell_i\| - \|\nabla_{\boldsymbol{\theta}} f\|\right)^2\right] \leq \sigma_{\text{mag}}^2$ and (iv) **Client churn:** At most γ clients are replaced per round, $\gamma \in [0, 1, 2, \cdots, K]$.*

Lemma 4 (Communication Error Bound (Restated)). *Let A. 3, A. 4 and A. 1 hold with gradient bound g_M as in A. 1. Let $D_{\boldsymbol{\theta}} = \max_{s \in [T]} \left\|\Delta\boldsymbol{\theta}^{(s)}\right\|$ denote the maximum single-step server displacement. Then, for each client $i \in [K]$,*

$$\mathbb{E}_{\mathbb{P}}\left[\left\|\epsilon_i^{(t,t_i)}\right\|\right] \leq \frac{\tau_{\max} \cdot D_{\boldsymbol{\theta}}}{K} + \left(\sum_{j=1}^{E} \eta^{(j)}\right)\left(\sigma_{\text{mag}} + g_M \sqrt{2(1 - \delta_{\min})}\right) \tag{A50}$$

and the per-client change in communication error satisfies

$$\mathbb{E}_{\mathbb{P}}\left[\left\|\Delta\epsilon_i^{(t,t_i)}\right\|\right] \leq \frac{1}{K}\left[\tau_{\max}\cdot D_{\boldsymbol{\theta}} + 2\gamma\left(\sum_{j=1}^{E}\eta^{(j)}\right) g_M\ \max\left(1,\ \sqrt{2(1-\delta_{\min})}\right)\right] \tag{A51}$$

The pool-averaged bounds follow by Jensen's inequality: $\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| \leq \frac{1}{K}\sum_i \mathbb{E}_{\mathbb{P}}[\left\|\epsilon_i^{(t,t_i)}\right\|]$ *and* $\left\|\mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]\right\| \leq \frac{1}{K}\sum_i \mathbb{E}_{\mathbb{P}}[\left\|\Delta\epsilon_i^{(t,t_i)}\right\|]$*; the K-aggregate convention $\epsilon_B \geq K\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\|$ is applied in Lem. 3.*

Proof. Fix an arbitrary client $i \in [K]$. By the triangle inequality and the decomposition in A. 4, $\left\|\epsilon_i^{(t,t_i)}\right\| \leq \|\epsilon_i^{\text{exo}}\| + \left\|\epsilon_i^{\text{endo}}\right\|$.

Exogenous bound. By A. 4(i), client i has delay $\tau_i \leq \tau_{\max}$. Since the pool is a K-aggregate of uniform client contributions (A. 2), the per-client exogenous contribution is $\|\epsilon_i^{\text{exo}}\| \leq \frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K}$.

Endogenous bound. The endogenous error is a combination of magnitude effect of the gradient and the divergence in the angle between a client gradient and the server gradient. To obtain the bound, we decompose the gradient difference between client and server into its magnitude and angular components such that

$$\begin{aligned}\|\nabla_{\boldsymbol{\theta}_i}\ell_i - \nabla_{\boldsymbol{\theta}} f\|^2 &= \|\nabla_{\boldsymbol{\theta}_i}\ell_i\|^2 + \|\nabla_{\boldsymbol{\theta}} f\|^2 - 2\left\langle\nabla_{\boldsymbol{\theta}_i}\ell_i, \nabla_{\boldsymbol{\theta}} f\right\rangle \\ &\leq \|\nabla_{\boldsymbol{\theta}_i}\ell_i\|^2 + \|\nabla_{\boldsymbol{\theta}} f\|^2 - 2\delta_{\min}\|\nabla_{\boldsymbol{\theta}_i}\ell_i\|\,\|\nabla_{\boldsymbol{\theta}} f\|\end{aligned} \tag{A52}$$

where we have used A. 4(ii). Adding and subtracting yields

$$\|\nabla_{\boldsymbol{\theta}_i}\ell_i - \nabla_{\boldsymbol{\theta}} f\|^2 = \left(\|\nabla_{\boldsymbol{\theta}_i}\ell_i\| - \|\nabla_{\boldsymbol{\theta}} f\|\right)^2 + 2\|\nabla_{\boldsymbol{\theta}_i}\ell_i\|\,\|\nabla_{\boldsymbol{\theta}} f\|\,(1-\delta_{\min}) \tag{A53}$$

Taking expectations over the client probability space and applying A. 4(iii) with g_M from A. 1,

$$\mathbb{E}_{\mathbb{P}}\left[\|\nabla_{\boldsymbol{\theta}_i}\ell_i - \nabla_{\boldsymbol{\theta}} f\|^2\right] \leq \sigma_{\text{mag}}^2 + 2g_M^2(1-\delta_{\min}) \tag{A54}$$

By Jensen's inequality, $\|\mathbb{E}_{\mathbb{P}}[\nabla_{\boldsymbol{\theta}_i}\ell_i - \nabla_{\boldsymbol{\theta}} f]\| \leq \sqrt{\sigma_{\text{mag}}^2 + 2g_M^2(1-\delta_{\min})}$, and applying $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$,

$$\mathbb{E}_{\mathbb{P}}\left[\|\nabla_{\boldsymbol{\theta}_i}\ell_i - \nabla_{\boldsymbol{\theta}} f\|\right] \leq \sigma_{\text{mag}} + g_M\sqrt{2(1-\delta_{\min})} \tag{A55}$$

The endogenous communication error for client i accumulates over E local update steps. From (Client Update), at each local step j the client drifts from the server trajectory by $\eta^{(j)}(\nabla_{\boldsymbol{\theta}_i}\ell_i - \nabla_{\boldsymbol{\theta}} f)$. Summing over E steps and applying the triangle inequality,

$$\left\|\epsilon_i^{\text{endo}}\right\| \leq \sum_{j=1}^{E}\eta^{(j)}\|\nabla_{\boldsymbol{\theta}_i}\ell_i - \nabla_{\boldsymbol{\theta}} f\| \tag{A56}$$

Applying (A55) to bound the expectation of the per-step drift and summing over E steps,

$$\mathbb{E}_{\mathbb{P}}\left[\left\|\epsilon_i^{\text{endo}}\right\|\right] \leq \left(\sum_{j=1}^{E}\eta^{(j)}\right)\left(\sigma_{\text{mag}} + g_M\sqrt{2(1-\delta_{\min})}\right) \tag{A57}$$

Since $\|\epsilon_i^{\text{exo}}\| \leq \frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K}$ is deterministic, applying $\mathbb{E}_{\mathbb{P}}[\left\|\epsilon_i^{(t,t_i)}\right\|] \leq \|\epsilon_i^{\text{exo}}\| + \mathbb{E}_{\mathbb{P}}[\left\|\epsilon_i^{\text{endo}}\right\|]$ provides the average bound on $\mathbb{E}_{\mathbb{P}}[\left\|\epsilon_i^{(t,t_i)}\right\|]$.

Next, we bound the average change in communication error $\left\|\Delta\epsilon_i^{(t,t_i)}\right\|$ between consecutive server rounds. By the triangle inequality and the decomposition as $\left\|\Delta\epsilon_i^{(t,t_i)}\right\| \leq \|\Delta\epsilon_i^{\text{exo}}\| + \left\|\Delta\epsilon_i^{\text{endo}}\right\|$. Again, for the exogenous component, the per-client staleness error is bounded by $\frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K}$ at every round (A. 4(i)), so its round-to-round change satisfies $\mathbb{E}_{\mathbb{P}}[\|\Delta\epsilon_i^{\text{exo}}\|] \leq \frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K}$. At any given communication round, new clients may join or leave the system. These clients could bring gradients that may

or may not align with with the prior average gradient. To model this, we introduce the replacement indicator $\mathbf{1}_i \in \{0, 1\}$, where $\mathbf{1}_i = 1$ if client i is replaced at the current round and $\mathbf{1}_i = 0$ otherwise. Then, due to this change, we have

$$\Delta\epsilon_i^{\text{endo}} = \mathbf{1}_i \left(\epsilon_{i_{\text{new}}}^{\text{endo}} - \epsilon_{i_{\text{old}}}^{\text{endo}}\right), \tag{A58}$$

so $\left\|\Delta\epsilon_i^{\text{endo}}\right\| \leq \mathbf{1}_i \left\|\epsilon_{i_{\text{new}}}^{\text{endo}} - \epsilon_{i_{\text{old}}}^{\text{endo}}\right\|$. By A. 4(iv), at most γ out of K clients are replaced per round. Since each client is equally likely to be replaced,

$$\mathbb{E}_{\mathbb{P}}[\mathbf{1}_i] = P(\mathbf{1}_i = 1) \leq \frac{\gamma}{K}. \tag{A59}$$

Now, for the change in the endogeneous effect, we note that the magnitude effect is upper bounded by the maximum value of the gradient. However, the essential component is the angular divergence between the incoming and the outgoing client gradients. We bound $\|\nabla_{\boldsymbol{\theta}_i}\ell_{\text{new}} - \nabla_{\boldsymbol{\theta}_i}\ell_{\text{old}}\|$, the gradient shift when client i is replaced. Let $\hat{n} = \nabla_{\boldsymbol{\theta}} f / \|\nabla_{\boldsymbol{\theta}} f\|$ denote the unit server gradient direction.

For any client gradient $\nabla_{\boldsymbol{\theta}_i}\ell$ satisfying $\|\nabla_{\boldsymbol{\theta}_i}\ell\| \leq g_M$ (A. 1(iii)) and $\langle \nabla_{\boldsymbol{\theta}_i}\ell, \hat{n}\rangle \geq \delta_{\min} \|\nabla_{\boldsymbol{\theta}_i}\ell\|$ (A. 4(ii)), expand the squared deviation from $g_M\hat{n}$:

$$\|\nabla_{\boldsymbol{\theta}_i}\ell - g_M\hat{n}\|^2 = \|\nabla_{\boldsymbol{\theta}_i}\ell\|^2 + g_M^2 - 2g_M \langle \nabla_{\boldsymbol{\theta}_i}\ell, \hat{n}\rangle \tag{A60}$$

Applying the angular lower bound $\langle \nabla_{\boldsymbol{\theta}_i}\ell, \hat{n}\rangle \geq \delta_{\min} \|\nabla_{\boldsymbol{\theta}_i}\ell\|$:

$$\|\nabla_{\boldsymbol{\theta}_i}\ell - g_M\hat{n}\|^2 \leq \|\nabla_{\boldsymbol{\theta}_i}\ell\|^2 + g_M^2 - 2\delta_{\min}\, g_M \|\nabla_{\boldsymbol{\theta}_i}\ell\| \tag{A61}$$

The right-hand side is quadratic in $\|\nabla_{\boldsymbol{\theta}_i}\ell\|$ on $[0, g_M]$ with interior minimum at $\|\nabla_{\boldsymbol{\theta}_i}\ell\| = \delta_{\min}\, g_M$. The maximum therefore occurs at one of the two boundaries:

$$\begin{aligned} \|\nabla_{\boldsymbol{\theta}_i}\ell\| = 0 &\implies g_M^2, \\ \|\nabla_{\boldsymbol{\theta}_i}\ell\| = g_M &\implies 2g_M^2(1 - \delta_{\min}). \end{aligned}$$

Since $2(1 - \delta_{\min}) \geq 1$ if and only if $\delta_{\min} \leq 1/2$, every valid client gradient satisfies

$$\|\nabla_{\boldsymbol{\theta}_i}\ell - g_M\hat{n}\| \leq g_M \max\left(1,\ \sqrt{2(1 - \delta_{\min})}\right). \tag{A62}$$

Applying (A62) to both the incoming and outgoing client gradients and invoking the triangle inequality,

$$\|\nabla_{\boldsymbol{\theta}_i}\ell_{\text{new}} - \nabla_{\boldsymbol{\theta}_i}\ell_{\text{old}}\| \leq \|\nabla_{\boldsymbol{\theta}_i}\ell_{\text{new}} - g_M\hat{n}\| + \|g_M\hat{n} - \nabla_{\boldsymbol{\theta}_i}\ell_{\text{old}}\| \leq 2g_M \max\left(1,\ \sqrt{2(1 - \delta_{\min})}\right). \tag{A63}$$

From (Client Update), at each local step j the gradient disruption is scaled by $\eta^{(j)}$. Summing over E steps,

$$\left\|\epsilon_{i_{\text{new}}}^{\text{endo}} - \epsilon_{i_{\text{old}}}^{\text{endo}}\right\| \leq \left(\sum_{j=1}^{E} \eta^{(j)}\right) 2g_M \max\left(1,\ \sqrt{2(1 - \delta_{\min})}\right) \tag{A64}$$

Since $\mathbf{1}_i$ and $\left\|\epsilon_{i_{\text{new}}}^{\text{endo}} - \epsilon_{i_{\text{old}}}^{\text{endo}}\right\|$ are independent (the gradient bound (A64) is a worst-case bound that holds regardless of whether client i is replaced),

$$\mathbb{E}_{\mathbb{P}}\left[\left\|\Delta\epsilon_i^{\text{endo}}\right\|\right] \leq \mathbb{E}_{\mathbb{P}}[\mathbf{1}_i] \cdot \left\|\epsilon_{i_{\text{new}}}^{\text{endo}} - \epsilon_{i_{\text{old}}}^{\text{endo}}\right\| \leq \frac{\gamma}{K} \cdot \left(\sum_{j=1}^{E} \eta^{(j)}\right) 2g_M \max\left(1,\ \sqrt{2(1 - \delta_{\min})}\right) \tag{A65}$$

Since $\|\Delta\epsilon_i^{\text{exo}}\| \leq \frac{\tau_{\max} \cdot D_{\boldsymbol{\theta}}}{K}$ holds deterministically and $\mathbb{E}_{\mathbb{P}}[\left\|\Delta\epsilon_i^{\text{endo}}\right\|] \leq \frac{\gamma}{K}\left(\sum_{j=1}^{E} \eta^{(j)}\right) 2g_M \max\left(1,\ \sqrt{2(1 - \delta_{\min})}\right)$, applying $\mathbb{E}_{\mathbb{P}}[\left\|\Delta\epsilon_i^{(t,t_i)}\right\|] \leq \|\Delta\epsilon_i^{\text{exo}}\| +$

$\mathbb{E}_{\mathbb{P}}[\|\Delta\epsilon_i^{\text{endo}}\|]$ gives

$$\mathbb{E}_{\mathbb{P}}\Big[\Big\|\Delta\epsilon_i^{(t,t_i)}\Big\|\Big] \le \frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K} + \frac{\gamma}{K}\left(\sum_{j=1}^{E}\eta^{(j)}\right) 2g_M\ \max\big(1,\ \sqrt{2(1-\delta_{\min})}\big)$$
$$= \frac{1}{K}\left[\tau_{\max}\cdot D_{\boldsymbol{\theta}} + 2\gamma\left(\sum_{j=1}^{E}\eta^{(j)}\right) g_M\ \max\big(1,\ \sqrt{2(1-\delta_{\min})}\big)\right] \tag{A66}$$

which provides the average bound on $\mathbb{E}_{\mathbb{P}}[\Big\|\Delta\epsilon_i^{(t,t_i)}\Big\|]$. □

A2.6.1 Convergence Rate

Theorem 3 (Convergence Rate)**.** *Consider Eq.* (CFL) *and let A. 3, A. 1, A. 2 and A. 4 hold. Define the server and client updates as in* (Server Update) *and* (Client Update) *with learning rates* $\eta^{(j)} = \beta^{(t_i,j)} = 1/(E^2T)$ *and* $(1-\alpha^{(t)}) = 1/(KET)$. *Define* $\sigma_e := \sigma_{\text{mag}} + g_M\sqrt{2(1-\delta_{\min})}$ *and* $\sigma_\delta := g_M\ \max\big(1,\ \sqrt{2(1-\delta_{\min})}\big)$. *If the well-posedness condition*

$$\tau_{\max} < \frac{(1-\rho_{\max})K}{3ET+\rho_{\max}} \tag{A67}$$

holds, then the communication error resolves to

$$\epsilon_B = \frac{\tau_{\max}g_M + (1-\rho_{\max})K^2\sigma_e/(ET)}{(1-\rho_{\max})K - \tau_{\max}(3ET+\rho_{\max})}, \tag{A68}$$

and

$$\min_{t\in\{0,1,2,\cdots,T\}}\left(\mathbb{E}_{\mathbb{P}}\left[\left\|\nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)})\right\|^2\right]\right)$$
$$\le \frac{2\big[f_0 - f^*\big]}{KT} + \frac{C_{\text{b}}}{ET\,\psi} + \frac{C_{\text{c}}}{(ET)^2\,\psi^2}. \tag{A69}$$

yielding the convergence rate $\mathcal{O}(1/(ET))$ *as* K *grows large.*

Proof. Define the endogenous divergence $\sigma_e := \sigma_{\text{mag}} + g_M\sqrt{2(1-\delta_{\min})}$ and the angular disruption $\sigma_\delta := g_M\ \max\big(1,\ \sqrt{2(1-\delta_{\min})}\big)$.
Using the learning rate schedules that is $\beta^{(t_i,j)} = 1/(E^2T)$, $(1-\alpha^{(t)}) = 1/(KET)$, and $\eta^{(j)} = 1/(E^2T)$ as in Lem. 3, we have $\sum_{j=1}^{E}\eta^{(j)} = 1/(ET)$. Substituting into Lem. 1,

$$\Big\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\Big\| \le \frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K} + \frac{\sigma_e}{ET}, \qquad \Big\|\mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]\Big\| \le \frac{1}{K}\left[\tau_{\max}\cdot D_{\boldsymbol{\theta}} + \frac{2\gamma\sigma_\delta}{ET}\right]. \tag{A70}$$

From Lem. 1, we have the following bounds on ϵ_B/K and Δ_{ϵ_B}/K

$$\frac{\epsilon_B}{K} \le \frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K} + \frac{\sigma_e}{ET}, \qquad \frac{\Delta_{\epsilon_B}}{K} \le \frac{\tau_{\max}\cdot D_{\boldsymbol{\theta}}}{K} + \frac{2\gamma\sigma_\delta}{KET}. \tag{A71}$$

Both bounds share the leading term $\tau_{\max}D_{\boldsymbol{\theta}}/K$. Substituting $D_{\boldsymbol{\theta}} \le \mathrm{B}\cdot ET/(1-\rho_{\max})$ into each term in Eq. (A71) with $\mathrm{B} = 2\epsilon_B/K + \Delta_{\epsilon_B}/K$:

$$\mathrm{B} \le \underbrace{\frac{2\tau_{\max}\mathrm{B}\cdot ET}{K(1-\rho_{\max})}}_{\text{from } 2\epsilon_B/K} + \frac{2\sigma_e}{ET} + \underbrace{\frac{\tau_{\max}\mathrm{B}\cdot ET}{K(1-\rho_{\max})}}_{\text{from } \Delta_{\epsilon_B}/K} + \frac{2\gamma\sigma_\delta}{KET}. \tag{A72}$$

Collecting B on the left:

$$\mathrm{B}\left[1 - \frac{3\tau_{\max}ET}{K(1-\rho_{\max})}\right] \le \frac{2\sigma_e}{ET} + \frac{2\gamma\sigma_\delta}{KET}. \tag{A73}$$

Define $\Psi := (1-\rho_{\max})K - 3\tau_{\max}ET$. Then,

$$\mathrm{B} \le \frac{K(1-\rho_{\max})}{\Psi} \cdot \frac{2\sigma_e + 2\gamma\sigma_\delta/K}{ET}, \tag{A74}$$

which provides the well posedness condition $\Psi > 0$ for the bound to be meaningful, i.e.,

$$\tau_{\max} < \frac{(1-\rho_{\max})K}{3ET}. \tag{A75}$$

Under (A75), back-substituting (A74) into (A71) yields:

$$\mathrm{B} = \mathcal{O}\Big(\frac{\sigma_e}{ET}\Big), \qquad \frac{\epsilon_B}{K} = \mathcal{O}\Big(\frac{\sigma_e}{ET}\Big), \qquad \frac{\Delta_{\epsilon_B}}{K} = \mathcal{O}\Big(\frac{\sigma_e}{ET}\Big). \tag{A76}$$

Substituting the schedule $\beta^{(t_i,j)} = \eta^{(j)} = 1/(E^2T)$ and $(1-\alpha^{(t)}) = 1/(KET)$ into the constants of Lem. 3 and Thm. 2 yields the effective step size and the amplification factor

$$\eta_{\text{eff}} \;:=\; \frac{1}{1-\alpha^{(t)}} \sum_{j=1}^{E} \eta^{(j)} \;=\; K, \qquad \mathrm{A} \;=\; \frac{ET}{1-\rho_{\max}}. \tag{A77}$$

Retaining the well-posedness margin $\Psi := (1-\rho_{\max})K - 3\tau_{\max}ET > 0$ of (A75), the closed-form bound (A74) gives

$$\begin{aligned} \mathrm{AB} \;&=\; \frac{ET}{1-\rho_{\max}} \cdot \frac{K(1-\rho_{\max})}{\Psi} \cdot \frac{2\sigma_e + 2\gamma\sigma_\delta/K}{ET} \;=\; \frac{2(K\sigma_e + \gamma\sigma_\delta)}{\Psi}, \\ \mathrm{A}^2\mathrm{B}^2 \;&=\; \frac{4(K\sigma_e + \gamma\sigma_\delta)^2}{\Psi^2}. \end{aligned} \tag{A78}$$

Substituting (A77)–(A78) into the three leading contributions of Thm. 2 gives

$$\textit{(I) Optimization:} \quad \frac{2}{\eta_{\text{eff}} T}\, \mathbb{E}_{\mathbb{P}}\big[f(\boldsymbol{\theta}^{(0)}, \boldsymbol{s}^{(0)}) - f^*\big] \;=\; \frac{2\big[f_0 - f^*\big]}{KT}, \tag{A79}$$

$$\textit{(II) Communication bias:} \quad \frac{2}{\eta_{\text{eff}}}\, L_{\epsilon_B}\, \mathrm{AB} \;=\; \frac{4L_{\epsilon_B}\,(K\sigma_e + \gamma\sigma_\delta)}{K\,\Psi}, \tag{A80}$$

$$\textit{(III) Curvature:} \quad \frac{2}{\eta_{\text{eff}}}\,(L_{\boldsymbol{\theta}} + L_{\boldsymbol{s}})\,\mathrm{A}^2\mathrm{B}^2 \;=\; \frac{8(L_{\boldsymbol{\theta}} + L_{\boldsymbol{s}})\,(K\sigma_e + \gamma\sigma_\delta)^2}{K\,\Psi^2}. \tag{A81}$$

The residual contribution of Thm. 2 (the terms weighted by Δ_{ϵ_B}/K and by $1/(2\eta_{\text{eff}})$) is of strictly lower order, since each such term carries an additional factor $\Delta_{\epsilon_B}/K = \mathcal{O}(\sigma_e/(ET))$ or $1/\eta_{\text{eff}} = 1/K$ relative to (A80)–(A81). Combining (A79)–(A81) therefore yields, for every $\Psi > 0$,

$$\begin{aligned} &\min_{t\in\{0,1,2,\cdots,T\}} \left(\mathbb{E}_{\mathbb{P}}\left[\left\|\nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)})\right\|^2\right]\right) \\ &\quad\le\; \frac{2\big[f_0 - f^*\big]}{KT} + \frac{4L_{\epsilon_B}\,(K\sigma_e + \gamma\sigma_\delta)}{K\,\Psi} + \frac{8(L_{\boldsymbol{\theta}} + L_{\boldsymbol{s}})\,(K\sigma_e + \gamma\sigma_\delta)^2}{K\,\Psi^2}, \end{aligned} \tag{A82}$$

where $\Psi \;=\; (1-\rho_{\max})K - 3\tau_{\max}ET$. To exhibit the horizon dependence, factor ET out of the margin. Writing $\Psi = ET\,\psi$ with the normalized margin

$$\psi \;:=\; \frac{(1-\rho_{\max})K}{ET} - 3\tau_{\max} \;=\; \underbrace{\frac{(1-\rho_{\max})K}{ET}}_{\text{vanishing term}} \;-\; \underbrace{3\tau_{\max}}_{\text{staleness term}}, \tag{A83}$$

and collecting the horizon-independent constants

$$C_{\mathrm{b}} \;:=\; \frac{4L_{\epsilon_B}(K\sigma_e + \gamma\sigma_\delta)}{K}, \qquad C_{\mathrm{c}} \;:=\; \frac{8(L_{\boldsymbol{\theta}} + L_{\boldsymbol{s}})(K\sigma_e + \gamma\sigma_\delta)^2}{K}, \tag{A84}$$

the bound (A82) takes the equivalent form

$$\begin{aligned} &\min_{t\in\{0,1,2,\cdots,T\}} \left(\mathbb{E}_{\mathbb{P}}\left[\left\|\nabla_{\boldsymbol{\theta}^{(t)}} f(\boldsymbol{\theta}^{(t)}, \boldsymbol{s}^{(t-1)})\right\|^2\right]\right) \\ &\quad\le\; \frac{2\big[f_0 - f^*\big]}{KT} \;+\; \frac{C_{\mathrm{b}}}{ET\,\psi} \;+\; \frac{C_{\mathrm{c}}}{(ET)^2\,\psi^2}. \end{aligned} \tag{A85}$$

Thus the optimization term decays as $1/(KT)$, the communication-bias and curvature terms carry explicit $1/(ET)$ and $1/(ET)^2$ factors. The margin ψ which models the effect of staleness has a terms that balances the number K with the number of local epochs and global server epochs. This is well posed when the number of clients relative to number of epoch dominates the staleness as in $(1-\rho_{\max})K/(ET) > 3\tau_{\max}$. Next, given the neighborhood constants (for large K) as $C_{\mathrm{b}} \approx 4L_{\epsilon_B}\sigma_e$ and $C_{\mathrm{c}} \approx 8(L_{\boldsymbol{\theta}} + L_{\boldsymbol{s}})K\sigma_e^2$, we then have $C_{\mathrm{b}}/\Psi = \Theta(1/(ET))$, $C_{\mathrm{c}}/\Psi^2 = \Theta(1/(K^2E^2T^2))$, and the optimization term $2[f_0 - f^*]/(KT) = \Theta(1/(KT))$. Consequently

$$\boxed{\mathcal{O}\Big(\tfrac{1}{KT}\Big)}.$$

□

A2.7 Discussion on the bound and the rate

Effect of client churn on $\Delta\epsilon_i^{(t,t_i)}$: The endogenous contribution to $\left\|\Delta\epsilon_i^{(t,t_i)}\right\|$ in bound couples the replacement probability γ/K with the angular disruption from client replacement. When $\delta_{\min} = 1$ (all clients aligned), $\max(1, \sqrt{2(1-\delta_{\min})}) = 1$ and client replacement introduces at most a gradient-magnitude disruption. When $\delta_{\min} \leq 1/2$, the angular term $\sqrt{2(1-\delta_{\min})}$ dominates, reflecting the increased gradient conflict between incoming and departing clients. In the standard FL setting with fixed client populations ($\gamma = 0$), the endogenous contribution vanishes entirely, and the bound reduces to the pure staleness-driven per-client bound $\left\|\Delta\epsilon_i^{(t,t_i)}\right\| \leq \tau_{\max} \cdot D_{\boldsymbol{\theta}}/K$, as in Xie et al. (2019) and Li et al. (2020b).

These remarks imply that the communication bound, we derived here, well approximates the effect of exogeneous and endogeneous and align well with standard intuition about these distinct effects in asynchronous FL. Notably, when these factors are absent, we shall recover the standard synchronous FL setting. But, the key to note here is that, our approach unifies these effects into a standard umbrella allowing us to design approaches and guarantee convergence under all of these effects together as we show below.

Well-posedness Condition: One of the key conditions under which the rate is valid is given by the condition that the maximum staleness $\tau_{\max}$ is small relative to the pool size K divided by the update epochs.

Recovery of synchronous FL When there is no staleness ($\tau_{\max} = 0$), no client churn ($\gamma = 0$), and all clients are homogeneous ($\delta_{\min} = 1$, $\sigma_{\mathrm{mag}} = 0$), A. 4 yields $\sigma_e = 0$ and Lem. 1 recovers $\left\|\mathbb{E}_{\mathbb{P}}[\epsilon^{(t,t_i)}]\right\| = 0$ and $\left\|\mathbb{E}_{\mathbb{P}}[\Delta\epsilon^{(t,t_i)}]\right\| = 0$. This is precisely the synchronous, homogeneous FL setting of McMahan et al. (2017) and Li et al. (2020c).