FEDADAVR: Adaptive Variance Reduction for Robust Federated Learning under Limited Client Participation

S M Ruhul Kabir Howlader
School of Computing and Mathematical Sciences
University of Leicester
Leicester LE1 7RH, United Kingdom
smrkh1@leicester.ac.uk

Xiao Chen*
School of Artificial Intelligence
Nanchang University
Nanchang 330031, China
xiao.chen@ncu.edu.cn

Yifei Xie
School of Management
Zhejiang University
Hangzhou 310027, China
xieyfzju@outlook.com

Lu Liu
Department of Computer Science
University of Exeter
Exeter EX4 4QF, United Kingdom
l.liu3@exeter.ac.uk

Abstract

Federated learning (FL) encounters substantial challenges due to heterogeneity, leading to gradient noise, client drift, and partial client participation errors, the last of which is the most pervasive but remains insufficiently addressed in current literature. In this paper, we propose FEDADAVR, a novel FL algorithm aimed at solving heterogeneity issues caused by partial client participation by incorporating an adaptive optimiser with a variance reduction technique. This method takes advantage of the most recent stored updates from clients, even when they are absent from the current training round, thereby emulating their presence. Furthermore, we propose FEDADAVR-QUANT, a server-side memory compression variant that stores historical client updates in quantised form (while keeping client-server communication in full precision), reducing the server memory footprint of FEDADAVR by 50% (FP16), 75% (Int8), and 87.5% (Int4) while maintaining highly competitive, and occasionally superior model performance. We analyse the non-convex convergence behaviour of FEDADAVR with Adagrad and prove that our proposed framework eliminates the first-order partial client participation bias while improving monotonically with cohort size. Extensive experiments conducted on multiple vision and NLP datasets demonstrate that FEDADAVR consistently outperforms state-of-the-art baseline methods.

1 Introduction

The successful deployment of machine learning models hinges upon access to large-scale, high-quality datasets; however, assembling such datasets poses considerable challenges. In healthcare contexts, for instance, the centralised collection of patient data and the training of machine learning models are constrained by regulations such as the Health Insurance Portability and Accountability Act (HIPAA) and the General Data Protection Regulation (GDPR) (Warnat-Herresthal et al., 2021). Federated Learning (FL) has emerged as an effective paradigm to address these constraints, enabling multiple clients to collaboratively train a shared model without exposing raw data beyond the owner's device.

*Corresponding Author

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

The cross-device FL setting (Kairouz et al., 2019) encompasses vast numbers of clients, only a subset of which are active in any given round, resulting in increased heterogeneity in both data distributions and device capabilities, thereby complicating FL training. In practical deployments like mobile predictive text keyboards and healthcare wearables, this active fraction is often exceedingly small, as devices typically participate only when simultaneously idle, charged, and on unmetered Wi-Fi. Moreover, such heterogeneous (non-IID) data give rise to client-drift errors. Convergence analyses of FedAvg (McMahan et al., 2017) have shown that data heterogeneity can slow convergence and introduce bias into the global model (i.e., client drift from non-IID data) (Karimireddy et al., 2020; Li et al., 2020c; Khaled et al., 2020). To counteract these effects, FedProx incorporates a proximal term into the local objective to penalise deviation from the global model (Li et al., 2020b), whereas SCAFFOLD employs control variates to correct client drift error induced by heterogeneity (Karimireddy et al., 2020). Subsequent methods, including ADACOMM (Wang and Joshi, 2019), FedBug (Kao and Wang, 2023), AdaBest (Varno et al., 2022) and Def-KT (Li et al., 2022a), have further refined techniques for mitigating drift.

While client drift has been extensively studied, FL is also affected by the error introduced by partial client participation. This challenge arising from client unavailability due to device and communication variability (Gu et al., 2021; Jhunjhunwala et al., 2022; Su et al., 2023; Wang et al., 2023; Mansour et al., 2023). To mitigate this challenge, MIFA was proposed (Gu et al., 2021), which approximates full participation by retaining each client's most recent update on the server and reusing it in subsequent rounds when that client is unavailable. FedVARP (Jhunjhunwala et al., 2022) advances this approach by addressing two key limitations of MIFA and demonstrating improved empirical performance. Moreover, it provides a decomposition of the total heterogeneity error into three components: stochastic gradient error, client drift error, and partial participation error. The partial participation error, although significant, has received comparatively limited attention. Importantly, partial client participation skews the global model update towards the data distributions of the selected subset of clients in each round, thereby exacerbating overall heterogeneity. To reduce the substantial storage overhead associated with maintaining per-client updates at scale, the ClusterFedVARP variant, introduced in (Jhunjhunwala et al., 2022), groups client updates into clusters. This clustering strategy mitigates memory demands without sacrificing convergence performance. However, it may also introduce potential challenges, including incorrect cluster assignments (misgrouped updates), imbalanced cluster sizes, and temporal changes in cluster membership (cluster drift).

Despite the advantages of FedVARP, our preliminary experiments reveal that it converges slowly under severely label-quantity skew conditions (e.g., when each client's dataset contains only one (LQ-1) or two (LQ-2) classes). This limitation arises primarily from the fixed server learning rate, resulting in a total absence of adaptivity. *Why adaptivity matters in variance-reduced FL.* In FedVARP, the variance-reduced update is applied with a fixed server learning rate η_s. However, when client participation is sporadic and data distributions are severely skewed (e.g., LQ-1 partitioning, the most extreme heterogeneity setting), different model parameters receive updates at drastically different frequencies. Parameters associated with frequent classes experience dense, low-variance signals, whereas those tied to rare classes are updated rarely and with high variance. A fixed η_s cannot accommodate this imbalance: any single choice either inflates noise on active coordinates or freezes learning on stale ones. This explains our experimental observation that FedVARP converges slowly under extreme label skew, and more importantly, it reveals a fundamental *missing ingredient* in current variance-reduced FL methods, i.e. *coordinate-wise step adaptation*.

Simply stacking an existing variance-reduced method onto an off-the-shelf adaptive server optimiser (e.g., feeding FedVARP outputs directly into FedOpt (Reddi et al., 2021)) is ineffective. As we show in Section 3, a naive composition causes the adaptive moments to absorb the variance-reduction scaling factor, which distorts the numerical stabiliser ϵ. Furthermore, standard implementations lack the early-round bias correction required under extreme non-IID participation. We address this by decoupling the SAGA-style variance-reduced correction $\mathbf{r}^{(t)}$ in (6) from the adaptive moment accumulators and explicitly reintegrating exact bias correction; this unified design ensures that the adaptive preconditioner cleanly tracks the debiased pseudo-gradient $\mathbf{G}^{(t)}$.

These insights, alongside advances in SAGA-style variance reduction (VR) (Defazio et al., 2014), partial variance reduction and heterogeneous client sampling (Gu et al., 2021; Jhunjhunwala et al., 2022; Su et al., 2023; Wang et al., 2023), and adaptive optimisation, motivate our proposed framework. Specifically, this paper makes the following contributions:

- We propose FEDADAVR, a federated learning algorithm that unifies server-side adaptive optimisation with SAGA-style variance reduction by decoupling historical client-state maintenance from adaptive moment preconditioning, addressing the coordinate-wise learning-rate imbalance induced by partial client participation. We establish a formal non-convex convergence guarantee for FEDADAVR with Adagrad (Theorem 4.4 and Appendix B).
- We further introduce FEDADAVR-QUANT, a complementary server-side memory compression variant wherein the most recent client updates are retained in quantised form to reduce server-side memory usage (by 50% (FP16), 75% (Int8), and 87.5% (Int4)). Empirical evaluations show that this quantisation preserves convergence and frequently acts as a mild regulariser while substantially reducing server memory overhead.
- Comprehensive experiments are carried out under various IID and non-IID data partitioning schemes on multiple datasets, simulating scenarios of extreme client unavailability against several state-of-the-art baselines.

While the theoretical analysis (Appendix B) provides worst-case convergence bounds under standard assumptions, the empirical results confirm that the proposed combination substantially improves training efficiency and final accuracy, especially in challenging low-participation settings where existing variance-reduced methods stagnate.

Unlike prior research, which has leveraged quantisation primarily to address communication heterogeneity (Reisizadeh et al., 2020; Mao et al., 2022; Ren et al., 2023; Chen et al., 2024) and device constraints (Gupta et al., 2023; Abdelmoniem and Canini, 2021; Chen et al., 2024), FEDADAVR-QUANT is, to the best of our knowledge, the first FL strategy to explicitly mitigate server-side memory bottlenecks via the quantisation of stored client states. Uniquely, our approach tackles the challenge of partial client participation by unifying server-side adaptive optimisation, SAGA-style variance reduction, and quantised state maintenance, with a formal non-convex convergence guarantee for full-precision FEDADAVR under server-side Adagrad (Section 4). A comprehensive review of related literature, alongside a detailed theoretical and mechanistic comparison of FEDADAVR against key baselines, is provided in Appendix A.

2 Preliminaries

Federated learning involves a network of N distributed clients (e.g., mobile devices or edge sensors), each possessing a private dataset $\mathcal{D}_n$=$\{(x_i, y_i)\}$, with no exchange of raw data. The overarching objective is to learn a shared model $w \in \mathbb{R}^d$ by minimising the aggregate empirical loss:

$$\min_{w} \; F(w) = \sum_{n=1}^{N} p_n \, F_n(w), \quad \text{where} \quad F_n(w) = \frac{1}{|\mathcal{D}_n|} \sum_{(x_i, y_i) \in \mathcal{D}_n} \ell\big(w; x_i, y_i\big). \tag{1}$$

where p_n=$\frac{|\mathcal{D}_n|}{\sum_{j=1}^{N} |\mathcal{D}_j|}$ represents the weight assigned to each client's contribution, and $\ell(\cdot)$ denotes the loss function. Owing to communication and privacy constraints, clients perform several local updates (typically stochastic gradient steps) on $F_n(w)$, and transmit only model updates (rather than raw data) to a central server. The server then aggregates these updates (e.g., through weighted averaging) to produce an improved global model. This federated framework aims to balance statistical heterogeneity (i.e., non-IID data across clients), limited communication bandwidth, and stringent data privacy requirements.

Consider the FedAvg algorithm, in which the server randomly selects a subset $\mathcal{S}^{(t)}$ of clients and transmits the current global model $\mathbf{w}^{(t)}$ to them. Each selected client i performs K steps of local SGD initialised from the current global model $\mathbf{w}^{(t)}$, returning the gradient

$$\mathbf{g}_i^{(t)} = \frac{1}{\eta_c} \left(\mathbf{w}^{(t)} - \mathbf{w}_i^{(t,K)} \right), \tag{2}$$

where η_c denotes the client learning rate, to the server (see Appendix C.1 for the full procedure). The server aggregates these local updates to form the new global model according to:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \sum_{i \in \mathcal{S}^{(t)}} p_i \mathbf{g}_i^{(t)}. \tag{3}$$

Variance arises in this process due to the random selection of clients, particularly when their local datasets are heterogeneous. Classical federated learning algorithms like FedAvg (McMahan et al., 2017) struggle to manage this variance in scenarios where only a small number of clients participate per round and the data distribution is highly skewed.

3 FEDADAVR and FEDADAVR-QUANT

This section introduces the proposed algorithms, FEDADAVR and FEDADAVR-QUANT, both designed to address variance resulting from limited client participation and highly skewed data distributions across clients. The proposed method combines the variance reduction capability of SAGA with the adaptive characteristics of optimisers such as Adam, Adagrad, Adabelief, Yogi, and Lamb. Notably, the proposed FEDADAVR, as defined in Algorithm 1, operates using only the gradients exchanged between the server and the clients, imposing no additional computational or communication burden on client devices.

During each training round, a subset $S^{(t)}$ of clients is randomly selected under the FEDADAVR framework. The selected clients receive the global model $\mathbf{w}^{(t)}$ from the server and carry out local stochastic gradient descent (SGD). Following this, each participating client transmits its local update $\mathbf{g}_i^{(t)}$ to the server. Up to this stage, the procedure mirrors that of FedAvg. However, whereas FedAvg applies direct averaging of client updates (see eq. (3)), it is susceptible to high variance under limited client participation when data distributions are heterogeneous. This source of variance constitutes the dominant error component. To mitigate this, the proposed method incorporates a variance reduction mechanism in conjunction with an adaptive optimiser. The server retains the most recent update received from each client, thereby simulating full client presence in each round. In the quantised variant, referred to as FEDADAVR-QUANT, the stored updates are quantised to reduce memory usage, while all other components remain consistent with those of FEDADAVR.

The server maintains the most recent updates received from clients. Initially, each client $j \in [N]$ is associated with an update denoted by $\mathbf{y}_j^{(0)}$. Following each training round, the updates are revised according to the following rule:

$$\mathbf{y}_j^{(t+1)} = \begin{cases} \mathbf{g}_j^{(t)} & \text{if } j \in S^{(t)} \\ \mathbf{y}_j^{(t)} & \text{if } j \notin S^{(t)} \end{cases}, \text{ for all } j \in [N]. \tag{4}$$

For the quantised variant, equation (4) is modified as follows:

$$\mathbf{y}_j^{(t+1)} = \begin{cases} \text{QUANT}(\mathbf{g}_j^{(t)}) & \text{if } j \in S^{(t)} \\ \mathbf{y}_j^{(t)} & \text{if } j \notin S^{(t)} \end{cases}, \text{ for all } j \in [N]. \tag{5}$$

Equations (4) and (5), corresponding to FEDADAVR and FEDADAVR-QUANT, respectively, ensure that $\mathbf{y}_j^{(t)}$ consistently stores the most recent update for each client j, and require $\mathcal{O}(Nd)$ memory on the server.

To compute the variance-reduced model update $\mathbf{r}^{(t)}$, the received updates $\mathbf{g}_i^{(t)}$ from the $M = |S^{(t)}|$ participating clients are jointly utilised with the stored updates $\mathbf{y}_j^{(t)}$ (under uniform client weighting $p_j = \frac{1}{N}$):

$$\mathbf{r}^{(t)} = \frac{1}{M} \sum_{i \in S^{(t)}} \left(\mathbf{g}_i^{(t)} - \mathbf{y}_i^{(t)} \right) + \frac{1}{N} \sum_{j=1}^{N} \mathbf{y}_j^{(t)}. \tag{6}$$

For the quantised variant, both the active correction term and the historical average dequantise the stored states $\mathbf{y}_j^{(t)}$, modifying equation (6) as follows:

$$\mathbf{r}^{(t)} = \frac{1}{M} \sum_{i \in S^{(t)}} \left(\mathbf{g}_i^{(t)} - \text{DEQUANT}(\mathbf{y}_i^{(t)}) \right) + \frac{1}{N} \sum_{j=1}^{N} \text{DEQUANT}(\mathbf{y}_j^{(t)}). \tag{7}$$

Algorithm 1 FEDADAVR and FEDADAVR-QUANT

Initialisation: Set initial model parameters $\mathbf{w}^{(0)}$, server learning rate η_s, client learning rate η_c, rounds T, and initial states $\mathbf{y}_j^{(0)} = 0$ for all $j \in [N]$, with $m_{-1} = 0$, $v_{-1} = 0$, and $z_{-1} = 0$

for $t = 0, 1, \ldots, T-1$ **do**

  Send $\mathbf{w}^{(t)}$ to all active devices $i \in S^{(t)}$

  for $i \in S^{(t)}$ **do**

    $\mathbf{g}_i^{(t)} \leftarrow \textsc{DeviceUpdate}(i, \mathbf{w}^{(t)}, \eta_c)$ ▷ Compute local gradient

  end for

$$\mathbf{r}^{(t)} \leftarrow \begin{cases} \frac{1}{M}\sum_{i\in S^{(t)}}\left(\mathbf{g}_i^{(t)} - \mathbf{y}_i^{(t)}\right) + \frac{1}{N}\sum_{j=1}^{N}\mathbf{y}_j^{(t)} \\ \frac{1}{M}\sum_{i\in S^{(t)}}\left(\mathbf{g}_i^{(t)} - \textsc{Dequant}(\mathbf{y}_i^{(t)})\right) + \frac{1}{N}\sum_{j=1}^{N}\textsc{Dequant}(\mathbf{y}_j^{(t)}) \end{cases}$$ ▷ VR update

  $\mathbf{G}^{(t)} \leftarrow \mathbf{r}^{(t)}\,\eta_c$ ▷ Form server pseudo-gradient

$$\mathbf{w}^{(t+1)} \leftarrow \begin{cases} \textsc{AdagradOptimiser}(\mathbf{w}^{(t)}, \mathbf{G}^{(t)}, \eta_s, z_t) \\ \textsc{AdamOptimiser}(\mathbf{w}^{(t)}, \mathbf{G}^{(t)}, \eta_s, m_t, v_t) \\ \textsc{AdabeliefOptimiser}(\mathbf{w}^{(t)}, \mathbf{G}^{(t)}, \eta_s, m_t, z_t) \\ \textsc{YogiOptimiser}(\mathbf{w}^{(t)}, \mathbf{G}^{(t)}, \eta_s, m_t, v_t) \\ \textsc{LambOptimiser}(\mathbf{w}^{(t)}, \mathbf{G}^{(t)}, \eta_s, m_t, v_t) \end{cases}$$ ▷ Select one optimiser

  for $j \in [N]$ **do**

$$\mathbf{y}_j^{(t+1)} \leftarrow \begin{cases} \mathbf{g}_j^{(t)} & \text{if } j \in S^{(t)} \\ \mathbf{y}_j^{(t)} & \text{if } j \notin S^{(t)} \end{cases}$$ ▷ Update stored client states

$$\mathbf{y}_j^{(t+1)} \leftarrow \begin{cases} \textsc{Quant}(\mathbf{g}_j^{(t)}) & \text{if } j \in S^{(t)} \\ \mathbf{y}_j^{(t)} & \text{if } j \notin S^{(t)} \end{cases}$$ ▷ Update quantised stored client states

  end for

end for

Subsequently, $\mathbf{r}^{(t)}$ is employed to compute the pseudo-gradient for the adaptive optimiser using the client learning rate η_c:

$$\mathbf{G}^{(t)} = \mathbf{r}^{(t)}\,\eta_c. \tag{8}$$

The weight decay parameter λ is incorporated into the pseudo-gradient when $\lambda \neq 0$:

$$\mathbf{G}^{(t)} = \mathbf{G}^{(t)} + \lambda \mathbf{w}^{(t)}. \tag{9}$$

Finally, $\mathbf{G}^{(t)}$ is passed to the selected adaptive optimiser (one of Adagrad, Adam, Adabelief, Yogi, or Lamb) to produce the updated global model $\mathbf{w}^{(t+1)}$. The full update rules and pseudocode for each optimiser are provided in Appendix C.2. For instance, under Adagrad the accumulator is updated as $z_t = z_{t-1} + \mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}$, and the global model is updated as $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta_s \frac{\mathbf{G}^{(t)}}{\sqrt{z_t}+\epsilon}$, where ϵ is a small constant.

To summarise Algorithm 1, the server initially transmits the current global model $\mathbf{w}^{(t)}$ to the selected clients $S^{(t)}$, who subsequently perform parallel local updates using their private datasets $\mathcal{D}_n$. Each participating client then returns an update $\mathbf{g}_i^{(t)}$ to the server. Upon receiving the updates, the server first computes the variance-reduced update $\mathbf{r}^{(t)}$, which is then employed to determine the server-side pseudo-gradient $\mathbf{G}^{(t)}$. This $\mathbf{G}^{(t)}$ is then processed by an adaptive optimiser together with its associated parameters. The optimiser updates the global model to $\mathbf{w}^{(t+1)}$, and the server stores the most recent client updates $\mathbf{y}_j^{(t+1)}$ in memory for use in the next round. This procedure is iterated for T communication rounds.

The complete execution of Algorithm 1 relies on several subroutines: the client update routine (Appendix C.1), the adaptive optimiser implementations (Appendix C.2), and the quantisation

procedure for FEDADAVR-QUANT (Appendix D). For further details on the specific adaptive optimisers, the reader is referred to (Duchi et al., 2011; Kingma and Ba, 2014; Zhuang et al., 2020; Zaheer et al., 2018; You et al., 2020).

Synergy between variance reduction and adaptive optimisers. The proposed algorithm combines two mechanisms whose roles are complementary. Let $\mathbf{G}^{(t)} = \eta_c \mathbf{r}^{(t)}$ be the server-side pseudo-gradient obtained after SAGA-style correction. Under partial participation, $\mathbf{G}^{(t)}$ faces two distinct challenges: (i) sampling bias and variance relative to the full active average due to unvisited clients, and (ii) severe coordinate-wise variance heterogeneity stemming from skewed label distributions across clients. The SAGA-style step (6) controls the partial-participation sampling error (Lemmas B.6 and B.7), while the adaptive preconditioner (e.g., $\mathrm{diag}(\sqrt{z_t} + \epsilon)^{-1}$ in Adagrad) normalises the heterogeneous coordinate-wise scales. Neither component alone suffices: without VR, partial-participation variance causes late-stage instability; without coordinate-wise adaptivity, the server step size is bottlenecked by the most volatile coordinates, stalling progress on rare-class parameters. As our ablation study (Section 5.4, Fig. 2 left) and full benchmark comparisons against fixed-step FedVARP across Adagrad, Adam, AdaBelief, Yogi, and Lamb (Appendix F, Tables 8–11) confirm, the unified approach consistently outperforms using either technique in isolation.

Where a naive stack of FedVARP + FedOpt differs from FEDADAVR. Because FEDADAVR combines the SAGA-style correction direction $\mathbf{r}^{(t)}$ from FedVARP (Jhunjhunwala et al., 2022) with server-side adaptive preconditioning from FedOpt (Reddi et al., 2021), it represents the principled formulation of variance-reduced adaptive federated optimisation, whereas naively piping the output of FedVARP ($\mathbf{g}_t = \eta_{\text{varp}} \eta_c \mathbf{r}^{(t)}$) into standard FedOpt implementations differs in key mechanics, as summarised in Table 1 (Appendix A.2). First, because adaptive accumulators are scale-covariant ($m_t \propto c$ and $v_t, z_t \propto c^2$ for $c = \eta_{\text{varp}} \eta_c > 0$), the scaling factor cancels in $m_t / \sqrt{v_t}$ and affects only the effective numerical stabiliser $\tau / (\eta_{\text{varp}} \eta_c)$. Second, standard FedOpt implementations omit exact moment bias correction: standard federated adaptive implementations omit exact bias correction ($\hat{m}_t, \hat{v}_t$). FEDADAVR resolves this by feeding the unscaled pseudo-gradient $\mathbf{G}^{(t)} = \eta_c \mathbf{r}^{(t)}$ directly into the accumulators, applying a single server learning rate η_s after normalisation, and applying exact bias correction $\hat{m}_t, \hat{v}_t$. Further comparison against MIFA, FedVARP, and SCAFFOLD is provided in Appendix A.2.

4 Convergence Analysis

This section presents the formal non-convex convergence analysis of FEDADAVR when instantiated with the server-side Adagrad optimiser (Algorithm 3). While our empirical evaluation in Section 5 demonstrates that the decoupled variance-reduced update extends effectively to momentum-based adaptive optimisers (Adam, AdaBelief, Yogi, and Lamb), establishing formal bounds for those variants requires tracking coupled heavy-ball momentum dynamics alongside historical state staleness; we therefore restrict our formal theoretical guarantee strictly to Adagrad. The analysis further assumes full-precision stored states (FEDADAVR-QUANT is not covered), vanilla local SGD without client momentum, and no weight decay ($\lambda = 0$). Consistent with the uniform weighting in (6), the analysis considers $f := \frac{1}{N} \sum_{i=1}^{N} f_i$ with $f_i \equiv F_i$, i.e., objective (1) with $p_n = \frac{1}{N}$ (e.g., equally sized local datasets); non-uniform weights p_n are not covered.

Assumption 4.1 (L-smoothness)**.** Each local objective function $f_i : \mathbb{R}^d \to \mathbb{R}$ is differentiable and L-smooth: for all $i \in [N]$ and all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, $\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq L \|\mathbf{x} - \mathbf{y}\|$.

Assumption 4.2 (Bounded local and global variance)**.** The stochastic gradient computed at each client is an unbiased estimator of the local gradient, $\mathbb{E}_{\xi_i \sim \mathcal{D}_i}[\nabla f_i(\mathbf{w}, \xi_i)] = \nabla f_i(\mathbf{w})$, with bounded local variance: $\mathbb{E}_{\xi_i \sim \mathcal{D}_i} \|\nabla f_i(\mathbf{w}, \xi_i) - \nabla f_i(\mathbf{w})\|^2 \leq \sigma^2$. Additionally, there exists a constant $\sigma_g \geq 0$ bounding the global gradient dissimilarity across clients: $\|\nabla f_i(\mathbf{w}) - \nabla f(\mathbf{w})\|^2 \leq \sigma_g^2$, for all $i \in [N]$.

Assumption 4.3 (Bounded stochastic gradients)**.** The stochastic gradients at each client are uniformly bounded: there exists a constant $G > 0$ such that for all $i \in [N]$ and $\mathbf{w} \in \mathbb{R}^d$, $\|\nabla f_i(\mathbf{w}, \xi_i)\| \leq G$ almost surely (which, by Jensen's inequality, implies $\|\nabla f_i(\mathbf{w})\| \leq G$, $\|\nabla f(\mathbf{w})\| \leq G$, and coordinate-wise $|[\nabla f_i(\mathbf{w}, \xi_i)]_j| \leq G$ for all $j \in [d]$).

Role and necessity of Assumption 4.3. Assumptions 4.1 and 4.2 are standard in federated optimisation (Karimireddy et al., 2020; Jhunjhunwala et al., 2022), and suffice on their own to establish our client-drift and variance-reduction lemmas (Lemmas B.5–B.7). Assumption 4.3 is invoked

exclusively on the adaptive server side, mirroring the stochastic gradient bound required by the adaptive FedOpt analyses of Reddi et al. (2021) (Assumption 3 therein). Specifically, it is required in three places: (i) bounding the monotone Adagrad accumulator $z_{t-1,j} \le 9T\eta_c^2K^2G^2$ to guarantee a deterministic lower bound $1/(\sqrt{z_{t-1,j}}+\epsilon) \ge 1/D_T$ on the effective coordinate-wise step size; (ii) bounding $|[\nabla f(\mathbf{w}^{(t)})]_j \tilde{G}_j^{(t)}| \le 3G^2$ to telescope the preconditioner-drift term across rounds; and (iii) bounding the one-step iterate drift $\|\tilde{\mathbf{G}}^{(t-1)}\|^2 \le 9G^2$ in the staleness recursion. Without Assumption 4.3, a single unbounded stochastic gradient early in training could permanently inflate the monotone accumulator $z_{t-1,j}$, preventing a deterministic lower bound on $\inf_{t\le T} 1/(\sqrt{z_{t-1,j}}+\epsilon)$ and obstructing the telescoping of the descent recursion. We note that the proof only invokes Assumption 4.3 at the global iterates $\mathbf{w}^{(t)}$ and the local iterates $\mathbf{w}_i^{(t,k)}$ visited by the algorithm, so the uniform statement over $\mathbb{R}^d$ above is a sufficient condition; relaxing it via affine-variance Adagrad techniques is a promising direction for future work.

Theorem 4.4 (Convergence of FEDADAVR with Adagrad)**.** *Suppose the local functions $\{f_i\}_{i=1}^N$ ($N \ge 2$) satisfy Assumptions 4.1, 4.2, and 4.3 with uniform client weights $p_i = \frac{1}{N}$. In each round $t \in \{0,\dots,T-1\}$, the server selects a subset $\mathcal{S}^{(t)}$ of $|\mathcal{S}^{(t)}| = M$ clients uniformly at random without replacement to perform $K \ge 1$ local SGD steps. Define the participation factors $A := 1 + \frac{13(N-M)}{2M(N-1)}$ and $B := 1 + \frac{4(N-M)}{N-1}$, the effective server step $\tilde{\eta}_s := \eta_s\eta_c K$, the accumulator upper bound $D_T := 3\eta_c KG\sqrt{T} + \epsilon$, and $\Delta_0 := f(\mathbf{w}^{(0)}) - f^*$ where $f^* = \min_{\mathbf{w}} f(\mathbf{w})$. If the client and server learning rates satisfy*

$$\eta_c \le \min\left\{\frac{1}{2LK},\ \frac{1}{8\sqrt{2A}\,L\sqrt{K(K-1)}},\ \left(\frac{\epsilon}{384AL^2K^2(K-1)G\sqrt{T}}\right)^{1/3}\right\},$$
$$\tilde{\eta}_s = \eta_s\eta_c K \le \min\left\{\frac{\epsilon}{4L},\ \frac{\epsilon^2}{16LD_T}\right\},$$

(for $K = 1$, the second and third bounds on η_c are void, i.e., read as $+\infty$), then the iterates $\{\mathbf{w}^{(t)}\}_{t=0}^{T-1}$ generated by FEDADAVR *satisfy*

$$\min_{0\le t\le T-1} \mathbb{E}\big\|\nabla f(\mathbf{w}^{(t)})\big\|^2 \le \frac{4D_T(\Delta_0+\Psi_0)}{\tilde{\eta}_s T} + \frac{12D_T dG^2}{\epsilon T} + \frac{4D_T}{\tilde{\eta}_s}\big(\bar{A}_1\sigma_g^2 + \bar{A}_2\sigma^2 + \bar{A}_3\big),$$

where d is the model dimension, $\Psi_0 := \frac{4\tilde{\eta}_s^2LN(N-M)}{\epsilon^2M^2(N-1)}G^2$ bounds the initial historical-state mismatch, and the per-round error constants are given by

$$\bar{A}_1 := \frac{8\tilde{\eta}_s A\eta_c^2L^2K(K-1)}{\epsilon}, \qquad \bar{A}_2 := \frac{2\tilde{\eta}_s A\eta_c^2L^2(K-1)}{\epsilon} + \frac{2\tilde{\eta}_s^2LB}{\epsilon^2MK},$$
$$\bar{A}_3 := \frac{36\tilde{\eta}_s^4L^3(N-M)}{\epsilon^4M(N-1)}\left(1 + \frac{2N(N-M)}{M^2}\right)G^2.$$

Remark 4.5 (Necessity of local steps and non-vanishing learning rates). The upper bounds on η_c and $\tilde{\eta}_s$ in Theorem 4.4 represent sufficient worst-case conditions for the non-convex smoothness analysis. They do not imply that $\eta_c \to 0$ is optimal: setting $\eta_c = 0$ yields $\tilde{\eta}_s = 0$, causing the leading $\Delta_0/(\tilde{\eta}_s T)$ term to diverge. In practice, a strictly positive $\eta_c > 0$ with $K > 1$ balances local progress against communication overhead, as confirmed by our hyperparameter studies (Tables 8–11).

The complete proof of Theorem 4.4, along with the supporting lemmas, is provided in Appendix B. Comparing Theorem 4.4 against the FedAvg error decomposition of Jhunjhunwala et al. (2022), which decomposes federated optimisation error into stochastic gradient variance, client drift, and partial participation error, shows that FEDADAVR eliminates the $\mathcal{O}(\sigma_g^2)$ partial participation bias: σ_g^2 enters solely through the higher-order client-drift term $\bar{A}_1\sigma_g^2$. Moreover, every partial-participation overhead ($A-1$, $B-1$, Ψ_0, and $\bar{A}_3$) decreases monotonically as the cohort size M increases and vanishes under full participation ($M = N$), where $A = B = 1$; the local stochastic-gradient noise and client-drift terms remain.

To obtain an explicit asymptotic rate in terms of T, K, and M, we specialise Theorem 4.4 under standard parameter choices.

Corollary 4.6. *Suppose Assumptions 4.1–4.3 hold and we set* $\eta_c = \Theta\big(\frac{1}{KL\sqrt{T}}\big)$, $\epsilon = \Theta\big(\frac{G}{L}\big)$, *and* $\eta_s = \Theta(1)$. *Then for fixed* N, M, *and* K *and sufficiently large* T, *the iterates of* FEDADAVR *satisfy*

$$\min_{0\le t\le T-1} \mathbb{E}\big\|\nabla f(\mathbf{w}^{(t)})\big\|^2 = \mathcal{O}\left(\frac{G\Delta_0}{\eta_s\sqrt{T}}\right) + \mathcal{O}\left(\frac{L\eta_s B\sigma^2}{GMK\sqrt{T}}\right) + \mathcal{O}\left(\frac{dG^2}{T}\right) + \mathcal{O}\left(\frac{A(K-1)\sigma^2}{K^2T} + \frac{A(K-1)\sigma_g^2}{KT}\right). \tag{10}$$

Corollary 4.6 establishes an $\mathcal{O}(1/\sqrt{T})$ convergence rate matching the canonical non-convex rate of federated Adagrad (Reddi et al., 2021), while ensuring that the dominant $\mathcal{O}(1/\sqrt{T})$ terms are completely free of the global heterogeneity variance σ_g^2 and that the sampling variance scales inversely with both M and K (since $1 \le B \le 5$). The omitted Ψ_0 and $\bar{A}_3$ contributions are of order $T^{-3/2}$ for fixed N and M, but carry participation factors of order N/M^2 and N^2/M^3 respectively; the $1/(MK)$ scaling of the sampling-variance term is therefore visible once $T \gtrsim KN^2/M^2$.

5 Experiments

The experimental evaluation comprises four primary components: an overview of the experimental setup across diverse configurations (further details in Appendix E); a comparative analysis of the proposed algorithm against state-of-the-art baselines (extended analysis in Appendix G.1); an evaluation of FEDADAVR and FEDADAVR-QUANT utilising various adaptive optimisers (additional results in Appendix H); and ablation and System Efficiency. Due to space constraints, detailed ablation study of FEDADAVR (Appendix F), details and results comparison of the quantisation methods (Appendix D and H.1), and the computational and communication comparisons against baselines (Appendix I) are provided exclusively in the Appendix. Unless otherwise specified, all experiments utilised FP16 quantisation for FEDADAVR-QUANT; due to computational constraints, a comprehensive ablation comparing FP32, FP16, Int8, and Int4 precisions was conducted exclusively on the CIFAR-10 dataset (LQ-1 partition) and is detailed in Appendix H.1.

5.1 Experimental Configuration

All experiments used the Flower framework (Beutel et al., 2020) with PyTorch (Paszke et al., 2019) for local model training. We evaluate on three vision datasets (MNIST, FMNIST, and CIFAR-10) and one NLP dataset (Shakespeare), across six data partitioning strategies: IID, IID-NonIID, Dirichlet, LQ-1, LQ-2, and LQ-3 (Li et al., 2022b) for vision, and the Natural ID strategy from the LEAF benchmark (Caldas et al., 2018) for Shakespeare. Client participation rates are deliberately extreme: 1% for MNIST and FMNIST, 2% for CIFAR-10, and below 1% for Shakespeare, with five clients selected per round in all cases. For vision, we adopt LeNet-5 (Lecun et al., 1998) for MNIST and FMNIST, and ResNet-18 (He et al., 2016) with group normalisation (Hsieh et al., 2020) for CIFAR-10. For Shakespeare we use a GRU-based RNN (Chung et al., 2014). Hyperparameters were selected via grid search for all methods to ensure a fair comparison, with most experiments conducted using one random seed, some using two, and a small subset using three seeds to support reproducibility. Full details of datasets, architectures, partitioning strategies, hyperparameter grids, and evaluation protocols are provided in Appendix E.

5.2 Comparison with State-of-the-Art Methods

To evaluate the proposed algorithms against state-of-the-art methods, we compared them with the most relevant baselines: FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020b), SCAFFOLD (Karimireddy et al., 2020), FedNova (Wang et al., 2020), FedAdam (Reddi et al., 2021), FedAdagrad (Reddi et al., 2021), FedYogi (Reddi et al., 2021), MIFA (Gu et al., 2021), and FedVARP (Jhunjhunwala et al., 2022). Accuracy and training loss served as the primary metrics for performance assessment. Figure 1 illustrates only the accuracy trajectories, while Figures 3–8 in Appendix G.1 present the accuracy and loss curves across varying data partitioning strategies and vision datasets. Figure 9 in Appendix G.1.7 illustrates the results on the Shakespeare dataset.

FEDADAVR consistently outperforms all baseline algorithms in nearly all scenarios, despite incurring no additional communication or computational overhead at the clients. The experimental setup posed

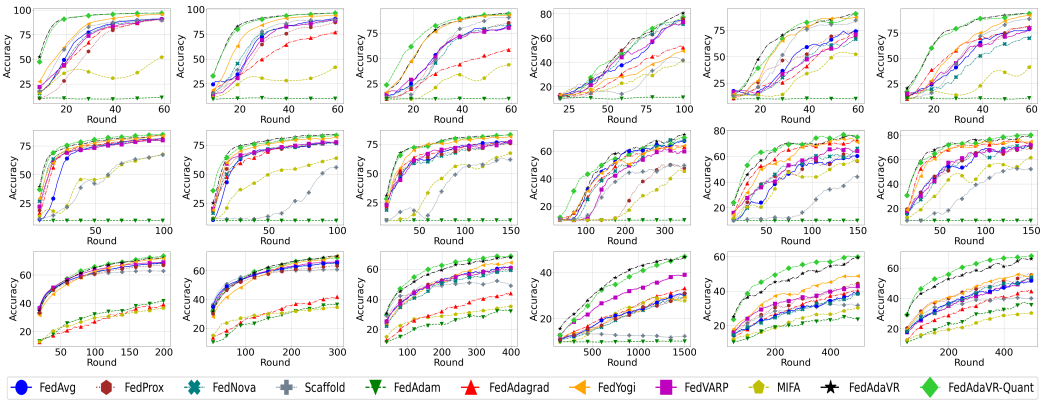


Figure 1: Comparison of accuracy across various data partitioning methods—IID, IID-NonIID, Dirichlet, LQ-1, LQ-2, and LQ-3 (left to right)—for the MNIST (top), FMNIST (middle), and CIFAR-10 (bottom) datasets. For larger versions of these plots alongside their corresponding loss metrics, please refer to Figures 3–8 in Appendix G.

significant challenges (variance and data heterogeneity), particularly under the LQ-1 partitioning scheme (where each client holds data from only one class) and with very limited client participation rates (1% for MNIST and FMNIST, 2% for CIFAR-10 and, $< 1\%$ for Shakespeare). This setting renders learning especially difficult for all algorithms. In LQ-1, data are sorted by label and partitioned into a single chunk (two chunks for LQ-2 and three for LQ-3), such that each partition contains data from only one (or two or three) class(es).

FedAdam failed to cope with this exceptionally demanding scenario on MNIST and FMNIST datasets and also underperformed in IID data distributions combined with extreme partial client participation. While the other baseline methods remained functional, both FEDADAVR and its quantised variant, FEDADAVR-QUANT, demonstrated superior performance. FedYogi exhibited strong results throughout the experiments, which was somewhat unexpected given the low client participation rates, indicating its capacity to handle partial client participation errors. Although FedVARP outperformed other baselines under the highly challenging LQ-1 data partitioning on the CIFAR-10 dataset (exhibiting improved robustness to partial client participation error), it still falls significantly short of the performance achieved by our proposed algorithms. MIFA also achieved reasonable performance but suffered from slower convergence. Similarly, FedAvg, FedAdagrad, FedProx, SCAFFOLD, and FedNova displayed sluggish convergence rates. While FedAdagrad remains a strong competitor in the NLP domain (Shakespeare), our method matches its performance while strictly outperforming it in vision, thereby establishing a broader cross-domain superiority.

A detailed analysis of system-side metrics, including communication cost, server memory, client computation overhead, per-round wall-clock time, end-to-end time to reach accuracy thresholds, and communication round efficiency, is provided in Appendix I. In summary, whilst FEDADAVR incurs a minor per-round latency increase over FedAvg (178.31s vs. 166.43s) due to server-side variance reduction, its end-to-end wall-clock cost is substantially lower: FEDADAVR reaches 45% accuracy on CIFAR-10 (LQ-1) in 163,689 seconds compared to 232,336 seconds for FedAvg, a reduction of approximately 30%. Furthermore, FEDADAVR requires significantly fewer communication rounds than all baselines to reach equivalent accuracy thresholds, confirming that the per-round overhead is more than compensated by faster convergence.

5.3 Comparison Between FEDADAVR and FEDADAVR-QUANT Across Adaptive Optimisers

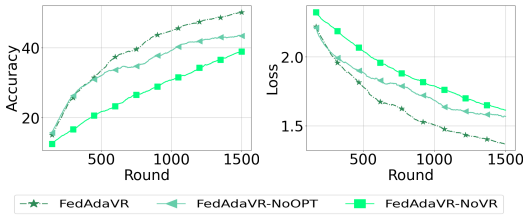
Both algorithms exhibit rapid convergence and comparable performance. Despite storing previous local model weights in a quantised format, FEDADAVR-QUANT maintains comparable accuracy while offering enhanced memory efficiency. This observation suggests that FL algorithms employing memory to retain historical updates for global aggregation are only marginally impacted by quantisation. This minimal change arises from the negligible difference between client weights pre- and post-quantisation. When computing $\mathbf{y}^{(t)}$, quantisation errors are randomly distributed, leading to an

inconsequential deviation that does not materially affect the performance of the algorithm (may act as a mild regulariser (Tang et al., 2026)). Therefore, our quantised version sometimes performs better.

Both FEDADAVR and FEDADAVR-QUANT employ adaptive optimisers. Tables 8, 9 10, and 11 detail the accuracy achieved under various optimiser and server learning rate. These tables present average accuracy over the final training rounds, as detailed in Table 3. The best accuracy values and configurations for both algorithms, as recorded in Tables 8, 9, 10, and 11, are illustrated in Figures 1, and 3–9. A full breakdown of accuracy under each optimiser and server learning rate configuration is provided in Appendix H, with discussion of optimiser selection in Appendix H.2.

5.4 Ablation and System Efficiency

To evaluate the contributions of our components, we conduct an ablation study on the CIFAR-10 dataset with LQ-1 partitioning. To assess the practical overhead of the algorithms, we perform a system-side efficiency analysis. As shown in Fig. 2 (left), removing the SAGA-style correction (FEDADAVR-NOVR) causes severe late-stage training instability and loss spikes, providing a direct empirical manifestation of uncorrected partial-participation estimator variance. Conversely, removing the adaptive preconditioner (FEDADAVR-NOOPT) eliminates these oscillations via global variance reduction, yet convergence severely stagnates; this confirms that under extreme label skew (LQ-1), resolving global participation bias alone is insufficient without coordinate-wise step-size adaptation to balance sparse, high-variance rare-class updates against dense frequent-class signals. Furthermore, while server-side variance reduction incurs a minor per-round latency overhead, it drastically reduces total communication rounds: as shown in Fig. 2 (right), FEDADAVR reduces end-to-end wall-clock time to reach 45% accuracy by $\sim 30\%$ compared to FedAvg (full details in Appendices F and I).



Algorithm	Time/Rnd	Time to 45%
FedAvg	166s	232,336s
FedNova	168s	251,457s
FedVARP	175s	242,749s
FEDADAVR	178s	**163,689s**

Figure 2: **Left:** Ablation study on CIFAR-10 (LQ-1) demonstrating the necessity of both variance reduction and adaptive optimisation. **Right:** End-to-end wall-clock time required to reach 45% accuracy on CIFAR-10, highlighting FEDADAVR's 30% speedup.

6 Conclusion and Future Work

This study introduces FEDADAVR, a novel FL algorithm integrating an adaptive optimiser with variance reduction mechanisms by decoupling historical client-state maintenance from adaptive moment preconditioning. Extensive experiments across vision and NLP benchmarks show this approach enhances convergence speed while mitigating variance from extreme partial client participation (1–2%) and severe label skew. The algorithm leverages server-side memory to retain the most recent client updates, thereby emulating continuous client availability during aggregation. We also present non-convex convergence results of FEDADAVR and prove that our proposed algorithm can asymptotically eliminate first-order partial client participation error while improving monotonically with participation. To further optimise memory efficiency, we propose a quantised variant, FEDADAVR-QUANT, wherein updates are stored in FP16, Int8, or Int4 formats, reducing server memory requirements by up to 87.5% while preserving generalisation accuracy.

Limitations and Future Work. While FEDADAVR and FEDADAVR-QUANT consistently outperform state-of-the-art baselines, our current study focuses on three deterministic linear quantisation schemes. Future work will explore advanced compression and hybrid caching to scale server memory to foundation-model regimes, alongside staleness-aware decay for temporal distribution shifts. Furthermore, while our formal analysis focuses on Adagrad and leaves the coupled quantisation error bounds of FEDADAVR-QUANT for future work, our experimental results confirm strong practical efficacy across adaptive optimisers and low-precision states (Tang et al., 2026). For an extended discussion of these limitations, we refer the reader to Appendix J.

Acknowledgments and Disclosure of Funding

The work has been supported by the SLAIDER project funded by the UK Research and Innovation Grant EP/Y018281/1. Xiao Chen is supported by the National Natural Science Foundation of China (NSFC) under Grant No. 62672224. This research used the ALICE High Performance Computing facility at the University of Leicester.

References

Ahmed M. Abdelmoniem and Marco Canini. Towards mitigating device heterogeneity in federated learning via adaptive model quantization. In *Proceedings of the 1st Workshop on Machine Learning and Systems*, EuroSys '21. ACM, April 2021. doi: 10.1145/3437984.3458839.

Bastien Batardière and Joon Kwon. Finite-sum optimization: Adaptivity to smoothness and loopless variance reduction, 2023.

Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Kwing Hei Li, Titouan Parcollet, Pedro Porto Buarque de Gusmão, and Nicholas D. Lane. Flower: A friendly federated learning research framework, 2020.

Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings, 2018.

Shengbo Chen, Le Li, Guanghui Wang, Meng Pang, and Cong Shen. Federated learning with heterogeneous quantization bit allocation and aggregation for internet of things. *IEEE Internet of Things Journal*, 11(2):3132–3143, January 2024. ISSN 2372-2541. doi: 10.1109/jiot.2023.3296493.

Xiangyi Chen, Xiaoyun Li, and Ping Li. Toward communication efficient adaptive gradient method. In *Proceedings of the 2020 ACM-IMS on Foundations of Data Science Conference*, FODS '20, page 119–128, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450381031. doi: 10.1145/3412815.3416891.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: a fast incremental gradient method with support for non-strongly convex composite objectives. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'14, page 1646–1654, Cambridge, MA, USA, 2014. MIT Press.

Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011.

Xinran Gu, Kaixuan Huang, Jingzhao Zhang, and Longbo Huang. Fast federated learning in the presence of arbitrary device unavailability. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

Badra Souhila Guendouzi, Samir Ouchani, Hiba EL Assaad, and Madeleine EL Zaher. A systematic review of federated learning: Challenges, aggregation methods, and development tools. *Journal of Network and Computer Applications*, 220:103714, November 2023. ISSN 1084-8045. doi: 10.1016/j.jnca.2023.103714.

Kartik Gupta, Marios Fournarakis, Matthias Reisser, Christos Louizos, and Markus Nagel. Quantization robust federated learning for efficient inference on heterogeneous devices. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Andrew Howard, Mark Sandler, Bo Chen, Weijun Wang, Liang-Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, Yukun Zhu, Ruoming Pang, Hartwig Adam, and Quoc Le. Searching for mobilenetv3. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324. IEEE, October 2019. doi: 10.1109/iccv.2019.00140.

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip B. Gibbons. The non-iid data quagmire of decentralized machine learning. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size, 2016.

Divyansh Jhunjhunwala, Pranay Sharma, Aushim Nagarkatti, and Gauri Joshi. Fedvarp: Tackling the variance due to partial client participation in federated learning. In James Cussens and Kun Zhang, editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 906–916. PMLR, 01–05 Aug 2022.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning, 2019.

Chia-Hsiang Kao and Yu-Chiang Frank Wang. Fedbug: A bottom-up gradual unfreezing framework for federated learning, 2023.

Belhal Karimi, Ping Li, and Xiaoyun Li. Fed-LAMB: Layer-wise and dimension-wise locally adaptive federated learning. In Robin J. Evans and Ilya Shpitser, editors, *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*, volume 216 of *Proceedings of Machine Learning Research*, pages 1037–1046. PMLR, 31 Jul–04 Aug 2023.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5132–5143. PMLR, 13–18 Jul 2020.

Ahmed Khaled, Konstantin Mishchenko, and Peter Richtarik. Tighter theory for local sgd on identical and heterogeneous data. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 4519–4529. PMLR, 26–28 Aug 2020.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

Raghuraman Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper, 2018.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Chengxi Li, Gang Li, and Pramod K. Varshney. Decentralized federated learning via mutual knowledge transfer. *IEEE Internet of Things Journal*, 9(2):1136–1147, 2022a. doi: 10.1109/JIOT.2021.3078543.

Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 965–978, 2022b. doi: 10.1109/ICDE53745.2022.00077.

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, may 2020a. doi: 10.1109/msp.2020.2975749.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In I. Dhillon, D. Papailiopoulos, and V. Sze, editors, *Proceedings of Machine Learning and Systems*, volume 2, pages 429–450, 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*, 2020c.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. *ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design*, pages 122–138. Springer International Publishing, 2018. ISBN 9783030012649. doi: 10.1007/978-3-030-01264-9_8.

Adnan Ben Mansour, Gaia Carenini, and Alexandre Duplessis. Tackling computational heterogeneity in fl: A few theoretical insights, 2023.

Yuzhu Mao, Zihao Zhao, Guangfeng Yan, Yang Liu, Tian Lan, Linqi Song, and Wenbo Ding. Communication-efficient federated learning with adaptive quantization. *ACM Transactions on Intelligent Systems and Technology*, 13(4):1–26, aug 2022. doi: 10.1145/3510587.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.

Lam M. Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. Sarah: a novel method for machine learning problems using stochastic recursive gradient. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 2613–2621. JMLR.org, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

K. M. Jawadur Rahman, Faisal Ahmed, Nazma Akhter, Mohammad Hasan, Ruhul Amin, Kazi Ehsan Aziz, A. K. M. Muzahidul Islam, Md. Saddam Hossain Mukta, and A. K. M. Najmul Islam. Challenges, applications and design aspects of federated learning: A survey. *IEEE Access*, 9:124682–124700, 2021. ISSN 2169-3536. doi: 10.1109/access.2021.3111118.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021.

Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *International conference on artificial intelligence and statistics*, pages 2021–2031. PMLR, 2020.

Yaoyao Ren, Yu Cao, Chengyin Ye, and Xu Cheng. Two-layer accumulated quantized compression for communication-efficient federated learning: TLAQC. *Scientific Reports*, 13(1), jul 2023. doi: 10.1038/s41598-023-38916-x.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520. IEEE, June 2018. doi: 10.1109/cvpr.2018.00474.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient, 2013.

Lili Su, Ming Xiang, Jiaming Xu, and Pengkun Yang. Federated Learning in the Presence of Adversarial Client Unavailability. *arXiv e-prints*, art. arXiv:2305.19971, May 2023. doi: 10.48550/arXiv.2305.19971.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9. IEEE, June 2015. doi: 10.1109/cvpr.2015.7298594.

Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019.

Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. Mnasnet: Platform-aware neural architecture search for mobile, 2018.

Xuan Tang, Jichu Li, and Difan Zou. A convergence analysis of adaptive optimizers under floating-point quantization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL `https://openreview.net/forum?id=wwP1SCACee`.

Farshid Varno, Marzie Saghayi, Laya Rafiee Sevyeri, Sharut Gupta, Stan Matwin, and Mohammad Havaei. Adabest: Minimizing client drift in federated learning via adaptive bias estimation. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIII*, page 710–726, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN 978-3-031-20049-6. doi: 10.1007/978-3-031-20050-2_41.

Jianyu Wang and Gauri Joshi. Adaptive communication strategies to achieve the best error-runtime trade-off in local-update sgd. In A. Talwalkar, V. Smith, and M. Zaharia, editors, *Proceedings of Machine Learning and Systems*, volume 1, pages 212–229, 2019.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Lin Wang, Yongxin Guo, Tao Lin, and Xiaoying Tang. Delta: diverse client sampling for fasting federated learning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Stefanie Warnat-Herresthal, Hartmut Schultze, Krishnaprasad Lingadahalli Shastry, Sathyanarayanan Manamohan, Saikat Mukherjee, Vishesh Garg, Ravi Sarveswara, Kristian Händler, Peter Pickkers, N Ahmad Aziz, et al. Swarm learning for decentralized and confidential clinical machine learning. *Nature*, 594(7862):265–270, 2021.

Jie Wen, Zhixia Zhang, Yang Lan, Zhihua Cui, Jianghui Cai, and Wensheng Zhang. A survey on federated learning: challenges and applications. *International Journal of Machine Learning and Cybernetics*, 14(2):513–535, November 2022. ISSN 1868-808X. doi: 10.1007/s13042-022-01647-y.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. In *International Conference on Learning Representations*, 2020.

Manzil Zaheer, Sashank Reddi, Devendra Sachan, and Satyen Kale. Adaptive methods for nonconvex optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients, 2016.

Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18795–18806. Curran Associates, Inc., 2020.

Contents

1 Introduction **1**

2 Preliminaries **3**

3 FEDADAVR and FEDADAVR-QUANT **4**

4 Convergence Analysis **6**

5 Experiments **8**

5.1 Experimental Configuration 8

5.2 Comparison with State-of-the-Art Methods 8

5.3 Comparison Between FEDADAVR and FEDADAVR-QUANT Across Adaptive Optimisers 9

5.4 Ablation and System Efficiency 10

6 Conclusion and Future Work **10**

Appendix **16**

A Extended Related Work and Algorithmic Differences **18**

A.1 Background and Related Literature 18

A.2 Detailed Differences from Key Baselines 18

B Convergence Analysis for FEDADAVR **20**

B.1 Basic Notations and Normalised Quantities 20

B.2 Auxiliary Lemmas and Full Proofs 21

B.3 Proof of Theorem 4.4 and Corollary 4.6 24

C Algorithms **26**

C.1 Client Device Update Procedure 26

C.2 Adaptive Optimisers 26

C.2.1 Adagrad (Algorithm 3). 27

C.2.2 Adam (Algorithm 4). 27

C.2.3 Adabelief (Algorithm 5). 28

C.2.4 Yogi (Algorithm 6). 28

C.2.5 Lamb (Algorithm 7). 29

D Model Quantisation **30**

E Experimental Setup **30**

E.1 Datasets 31

E.2 Dataset Partitioning Methods 31

E.3 Model Architectures 32
E.4 Client Configuration 32
E.5 Hyperparameter Configuration 32
E.6 Training Rounds and Evaluation Protocol 32
E.7 Multi-Seed Evaluation under Computational Constraints 33

F Impact of Variance Reduction and Adaptive Optimisation 33

G Experimental Results 34

G.1 Accuracy and Loss Comparison with State-of-the-Art Methods 34
G.1.1 Accuracy and loss comparison with the IID data partitioning method (Vision Datasets). 34
G.1.2 Accuracy and loss comparison with the IID-Non_IID data partitioning method (Vision Datasets). 35
G.1.3 Accuracy and loss comparison with the Dirichlet data partitioning method (Vision Datasets). 35
G.1.4 Accuracy and loss comparison with LQ-1 data partitioning method (Vision Datasets). 36
G.1.5 Accuracy and loss comparison with LQ-2 data partitioning method (Vision Datasets). 36
G.1.6 Accuracy and loss comparison with LQ-3 data partitioning method (Vision Datasets). 37
G.1.7 Accuracy and loss comparison on Shakespeare dataset (Natural ID Partitioner). 37

H Detailed Comparison of FEDADAVR and FEDADAVR-QUANT Across Adaptive Optimisers 38

H.1 Comparison Between Different Quantisation Methods 38
H.2 Server Learning Rate and Optimiser Selection 38

I Computational, Latency, and Communication Comparison Against Baselines 39

I.1 Algorithmic Complexity Comparison 39
I.2 Wall-Clock Time Comparison 40
I.3 Communication Round Efficiency 41

J Extended Limitations and Future Directions 41

A Extended Related Work and Algorithmic Differences

A.1 Background and Related Literature

Federated Learning. FedAvg (McMahan et al., 2017) is the foundational algorithm in federated learning, wherein a server initialises a global model and dispatches it to clients, who perform several steps of local training using their private data. Clients subsequently return their local updates, which the server aggregates to form an updated global model. This process iterates until convergence. Following FedAvg, numerous FL algorithms have been introduced to address its inherent limitations and the broader challenges within FL. For an overview of these challenges, we refer the reader to (Li et al., 2020a; Rahman et al., 2021; Wen et al., 2022; Guendouzi et al., 2023).

Adaptive Optimisers. Prior to the advent of federated learning, adaptive optimisers such as Adam (Kingma and Ba, 2014) and Adagrad (Duchi et al., 2011) were widely recognised for their effectiveness in guiding convergence behaviour. Building upon this success, the authors of FedOpt (Reddi et al., 2021) pioneered the integration of adaptive optimisers (e.g., Adagrad (Duchi et al., 2011), Adam (Kingma and Ba, 2014), and Yogi (Zaheer et al., 2018)) into federated settings. In the FedOpt family of algorithms, stochastic gradient descent (SGD) is retained on the client side to preserve communication parity with FedAvg, while three distinct adaptive methods (i.e. FedAdagrad, FedAdam, and FedYogi) are employed on the server side (Reddi et al., 2021). The Lamb optimiser (You et al., 2020) was subsequently utilised to develop FedLamb (Karimi et al., 2023), whereas the AmsGrad optimiser (Reddi et al., 2018) was adopted in a separate FL algorithm (Chen et al., 2020).

Variance Reduction Techniques. Variance reduction techniques have emerged as potent tools for accelerating the convergence of stochastic optimisation by mitigating gradient noise. In SAGA (Defazio et al., 2014), for instance, a table of past gradients is maintained and updated to construct an unbiased estimator at each iteration, yielding linear convergence for strongly convex functions without requiring full-batch gradients. Related methods include SAG (Schmidt et al., 2013), which averages stored gradients to reduce variance. SVRG (Johnson and Zhang, 2013) avoids storage overhead by alternating between full-gradient snapshots and inner stochastic loops. SARAH (Nguyen et al., 2017) further refines this approach by employing recursive gradient estimators without necessitating full-batch gradients. Numerous other VR methods have been proposed for centralised stochastic problems without requiring additional memory. In the context of FL, these methods are generally classified into two categories: *SVRG-style Variance Reduction*, which necessitates client participation in specific rounds, and *Momentum-based Variance Reduction*, which incurs both double communication and double computation costs (Jhunjhunwala et al., 2022). AdaLVR combines variance reduction with Adagrad optimiser to enhance finite-sum optimisation performance (Batardière and Kwon, 2023). Our proposed algorithms are influenced by AdaLVR but are designed to avoid the aforementioned drawbacks.

Quantisation. Quantisation refers to the process of reducing the numerical precision of model parameters to produce a more compact version of the same neural network. Various quantisation techniques have previously been employed to address diverse FL challenges, particularly communication heterogeneity (Reisizadeh et al., 2020; Mao et al., 2022; Ren et al., 2023; Chen et al., 2024), and device heterogeneity (Gupta et al., 2023; Abdelmoniem and Canini, 2021; Chen et al., 2024). The FEDADAVR-QUANT variant of our algorithm employs quantisation to store the most recent client updates, enabling their reuse in subsequent training rounds. To demonstrate the practical feasibility of our approach, we evaluated three fundamental and computationally efficient formats: FP16, Int8, and Int4. While diverse integer quantisation methods exist in the literature, FEDADAVR-QUANT specifically adopts the 8-bit symmetric per-tensor uniform quantiser proposed by (Krishnamoorthi, 2018) and the 4-bit methodology of (Zhou et al., 2016), which packs two 4-bit values per byte for storage.

A.2 Detailed Differences from Key Baselines

This work relates closely to several state-of-the-art strategies. A brief comparison of these methods with the proposed algorithm is provided below:

MIFA. Authors of MIFA were among the first to utilise memory for storing the latest observed client updates in FL to address the issue of unpredictable client unavailability (Gu et al., 2021). Their

Table 1: Mechanistic comparison of optimiser dynamics across FedVARP, FedOpt, a naive composition (FedVARP + FedOpt), and FEDADAVR. Here $\tilde{m}_t, \tilde{v}_t, \tilde{z}_t$ denote accumulators formed on the unscaled base signal $\mathbf{r}^{(t)}$, $\mathcal{T}$ is the round counter, and η_{opt} is the server rate of the stacked optimiser.

Step	**Standard FedVARP**	**Standard FedOpt**	**Naive Stack (FedVARP + FedOpt)**	**FEDADAVR**
Signal	$\mathbf{r}^{(t)} =$ $\text{VR}_{\text{Correction}}$	$\mathbf{g}_t =$ $\text{Client}_{\text{avg}}$	$\mathbf{g}_t = \eta_{\text{varp}} \cdot$ $\eta_c \cdot \mathbf{r}^{(t)}$	$\mathbf{G}^{(t)} =$ $\eta_c \cdot \mathbf{r}^{(t)}$
m_t **(Adam, Yogi)**	*(None)*	$\cdots +$ $(1-\beta_1)\mathbf{g}_t$	$\cdots + (1-\beta_1)\cdot$ $(\eta_{\text{varp}}\eta_c\mathbf{r}^{(t)})$	$\cdots +$ $(1-\beta_1)\mathbf{G}^{(t)}$
v_t **(Adam)**	*(None)*	$\cdots +$ $(1-\beta_2)\mathbf{g}_t^2$	$\cdots + (1-\beta_2)\cdot$ $(\eta_{\text{varp}}\eta_c\mathbf{r}^{(t)})^2$	$\cdots +$ $(1-\beta_2)(\mathbf{G}^{(t)})^2$
v_t **(Yogi)**	*(None)*	$v_{t-1} - (1-\beta_2)\mathbf{g}_t^2$ $\text{sgn}(v_{t-1} - \mathbf{g}_t^2)$	*same on* $\eta_{\text{varp}}\eta_c\mathbf{r}^{(t)}$	*same on* $\mathbf{G}^{(t)}$
z_t **(Adagrad)**	*(None)*	$z_{t-1}+$ $\mathbf{g}_t^2$	$z_{t-1}+$ $(\eta_{\text{varp}}\eta_c\mathbf{r}^{(t)})^2$	$z_{t-1}+$ $(\mathbf{G}^{(t)})^2$
Bias correction	—	none (all three)	none	$\hat{m}_t = \frac{m_t}{1-\beta_1^{\mathcal{T}}}$, $\hat{v}_t = \frac{v_t}{1-\beta_2^{\mathcal{T}}}$ (Adam, Yogi); none (Adagrad)
Update	$\mathbf{w}^{(t)} -$ $\eta_{\text{varp}}(\eta_c\mathbf{r}^{(t)})$	$\mathbf{w}^{(t)} -$ $\eta_{\text{opt}}\frac{m_t}{\sqrt{v_t}+\tau}$	$\mathbf{w}^{(t)} -$ $\eta_{\text{opt}}\frac{m_t}{\sqrt{v_t}+\tau}$	$\mathbf{w}^{(t)} -$ $\eta_s\frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$
Final — Adagrad	*(Direct Step)*	*(Std. Adagrad)*	$\approx \mathbf{w}^{(t)} -$ $\eta_{\text{opt}}\frac{\mathbf{r}^{(t)}}{\sqrt{\tilde{z}_t}+\frac{\tau}{\eta_{\text{varp}}\eta_c}}$	$\approx \mathbf{w}^{(t)} -$ $\eta_s\frac{\mathbf{r}^{(t)}}{\sqrt{\tilde{z}_t}+\frac{\epsilon}{\eta_c}}$
Final — Adam / Yogi	*(Direct Step)*	*(Std. Adam/Yogi)*	$\approx \mathbf{w}^{(t)} -$ $\eta_{\text{opt}}\frac{\tilde{m}_t}{\sqrt{\tilde{v}_t}+\frac{\tau}{\eta_{\text{varp}}\eta_c}}$	$\approx \mathbf{w}^{(t)} -$ $\eta_s\frac{\hat{\tilde{m}}_t}{\sqrt{\hat{\tilde{v}}_t}+\frac{\epsilon}{\eta_c}}$

aggregation procedure follows a SAG-like approach (Schmidt et al., 2013), which assigns equal weight to both the most recent and previous updates, thereby impeding convergence speed. Moreover, MIFA requires all clients to participate in the initial round, a condition rarely met in practical federated learning scenarios. In contrast, neither of the proposed algorithms relies on uniform weighting, nor do they require complete client participation at the outset.

FedVARP and ClusterFedVARP. Variance due to partial client participation is a major source of error in FL, first addressed by the authors of FedVARP (Jhunjhunwala et al., 2022). FedVARP applies a SAGA-like variance reduction technique on the server to mitigate errors from partial participation. Its variant, ClusterFedVARP, groups clients into clusters to improve memory efficiency by storing a single cluster update rather than individual updates. Although both variants outperform earlier state-of-the-art methods in highly non-IID settings, experiments show relatively slow convergence, potentially due to the fixed server learning rate (absence of server-side adaptivity). The proposed FEDADAVR adopts a similar SAGA-based reduction but replaces the fixed learning rate with an adaptive optimiser. According to our experiments, these refinements lead to improved performance in both IID and non-IID cases. To further enhance memory efficiency, FEDADAVR-QUANT is proposed, where the latest client updates are stored in a quantised representation.

SCAFFOLD. Designed to address client drift error, SCAFFOLD is among the earliest works to highlight this challenge (Karimireddy et al., 2020). It employs a SAGA-like technique (Defazio et al., 2014) enhanced with control variates, which necessitate communication between server and clients. While effective, these control variates increase communication overhead and slow down the overall federated learning process. In contrast, FEDADAVR requires clients to perform local SGD and transmit only local weight updates, without additional information for variance reduction. Consequently, FEDADAVR achieves lower communication costs.

FedOpt. FedOpt utilises adaptive optimisers such as Adam, Adagrad, and Yogi in federated settings (Reddi et al., 2021). Inspired by the success of adaptive optimisers in non-federated contexts, these algorithms have demonstrated substantial performance improvements. The current work also employs these optimisers following the application of SAGA-like variance reduction. Experimental findings indicate that integrating adaptive optimisation after variance reduction further accelerates convergence. Although Adabelief and Lamb optimisers are additionally considered here, other research applying these specific optimisers directly has not been included in this study. A detailed mechanistic comparison of how these baseline (FedOpt and FedVARP) update rules and naive compositions differ from our unified FEDADAVR framework is provided in Section 3 and illustrated in Table 1.

B Convergence Analysis for FEDADAVR

This appendix presents the complete, self-contained convergence proofs for Lemmas B.5–B.8, Theorem 4.4, and Corollary 4.6.

B.1 Basic Notations and Normalised Quantities

Following standard federated optimisation analyses (Jhunjhunwala et al., 2022; Reddi et al., 2021), we consider uniform client weights $p_i = \frac{1}{N}$ (yielding an active average of $\frac{1}{M}\sum_{i\in\mathcal{S}^{(t)}}$) for the theoretical convergence analysis. Note that while our code implementation weights clients proportionally to their local dataset sizes, this proportional weighting is mathematically identical to uniform weighting for balanced partitions (where local dataset sizes are equal) and serves as a standard practical approximation for theoretical analyses in unbalanced non-IID settings. Let $\mathcal{F}_t$ denote the σ-algebra generated by all stochasticity prior to communication round t. Within round t, for each local step $k \in \{0, \dots, K-1\}$, let $\mathcal{F}_{t,k}$ denote the σ-algebra generated by $\mathcal{F}_t$ and the local mini-batch draws $\{\xi_i^{(t,\tau)} : i \in [N], 0 \le \tau < k\}$, where $\xi_i^{(t,k)} \sim \mathcal{D}_i$ are sampled independently across clients $i \in [N]$ and steps k, and independently of the active client subset $\mathcal{S}^{(t)}$ (of size $|\mathcal{S}^{(t)}| = M$). We write $\mathbb{E}_{\xi^{(t)}}[\cdot] := \mathbb{E}_{\xi^{(t)}}[\cdot \mid \mathcal{F}_t]$ for the conditional expectation over all local mini-batch randomness $\xi^{(t)} = \{\xi_i^{(t,k)}\}$ at round t, and $\mathbb{E}_t[\cdot] := \mathbb{E}_{\mathcal{S}^{(t)},\xi^{(t)}}[\cdot \mid \mathcal{F}_t]$ for the joint conditional expectation over $\mathcal{S}^{(t)}$ and $\xi^{(t)}$ given $\mathcal{F}_t$.

Recall from Algorithm 2 that each participating client $i \in \mathcal{S}^{(t)}$ returns $\mathbf{g}_i^{(t)} = \frac{1}{\eta_c}(\mathbf{w}^{(t)} - \mathbf{w}_i^{(t,K)}) = \sum_{k=0}^{K-1} \nabla f_i(\mathbf{w}_i^{(t,k)}, \xi_i^{(t,k)})$, and the server forms $\mathbf{G}^{(t)} = \eta_c \mathbf{r}^{(t)}$. To align the scale of the server update with the average per-step gradient, we define the step-normalised quantities:

$$\Delta_i^{(t)} := \frac{1}{K}\mathbf{g}_i^{(t)} = \frac{1}{K}\sum_{k=0}^{K-1} \nabla f_i\big(\mathbf{w}_i^{(t,k)}, \xi_i^{(t,k)}\big),$$

$$\mathbf{h}_i^{(t)} := \frac{1}{K}\sum_{k=0}^{K-1} \nabla f_i\big(\mathbf{w}_i^{(t,k)}\big), \qquad \bar{\mathbf{h}}^{(t)} := \frac{1}{N}\sum_{i=1}^{N} \mathbf{h}_i^{(t)},$$

$$\tilde{\mathbf{y}}_j^{(t)} := \frac{1}{K}\mathbf{y}_j^{(t)}, \qquad \tilde{\mathbf{G}}^{(t)} := \frac{\mathbf{G}^{(t)}}{\eta_c K} = \frac{\mathbf{r}^{(t)}}{K} = \frac{1}{M}\sum_{i\in\mathcal{S}^{(t)}} \big(\Delta_i^{(t)} - \tilde{\mathbf{y}}_i^{(t)}\big) + \frac{1}{N}\sum_{j=1}^{N} \tilde{\mathbf{y}}_j^{(t)},$$

$$\tilde{\eta}_s := \eta_s\, \eta_c\, K.$$

Under the server-side Adagrad optimiser (Algorithm 3, with $\lambda = 0$), the accumulator at round $t \in \{0, \dots, T-1\}$ is updated via $z_t = z_{t-1} + \mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}$ from initial state $z_{-1} := \mathbf{0} \in \mathbb{R}^d$, and the global model update is given equivalently by

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta_s \frac{\mathbf{G}^{(t)}}{\sqrt{z_t} + \epsilon} = \mathbf{w}^{(t)} - \tilde{\eta}_s \frac{\tilde{\mathbf{G}}^{(t)}}{\sqrt{z_t} + \epsilon}, \tag{11}$$

where division and square roots are understood coordinate-wise. Because $\mathbf{w}_i^{(t,k)}$ is $\mathcal{F}_{t,k}$-measurable, Assumption 4.2 implies that $\delta_i^{(t,k)} := \nabla f_i(\mathbf{w}_i^{(t,k)}, \xi_i^{(t,k)}) - \nabla f_i(\mathbf{w}_i^{(t,k)})$ forms a martingale difference sequence with respect to $\{\mathcal{F}_{t,k}\}_{k=0}^{K-1}$, satisfying $\mathbb{E}[\delta_i^{(t,k)} \mid \mathcal{F}_{t,k}] = \mathbf{0}$ and $\mathbb{E}[\|\delta_i^{(t,k)}\|^2 \mid \mathcal{F}_{t,k}] \le \sigma^2$. By the tower property of conditional expectation, $\mathbb{E}_{\xi^{(t)}}[\Delta_i^{(t)} - \mathbf{h}_i^{(t)}] = \mathbf{0}$ and, by pairwise orthogonality of martingale differences, $\mathbb{E}_{\xi^{(t)}}\|\Delta_i^{(t)} - \mathbf{h}_i^{(t)}\|^2 = \frac{1}{K^2}\sum_{k=0}^{K-1} \mathbb{E}_{\xi^{(t)}}\|\delta_i^{(t,k)}\|^2 \le \frac{\sigma^2}{K}$. Crucially, because $\mathcal{S}^{(t)}$ is sampled uniformly at random independently of $\xi^{(t)}$ and $\mathcal{F}_t$, the normalised variance-reduced direction $\tilde{\mathbf{G}}^{(t)}$ is conditionally unbiased for $\mathbb{E}_{\xi^{(t)}}[\bar{\mathbf{h}}^{(t)}]$:

$$\mathbb{E}_t\big[\tilde{\mathbf{G}}^{(t)}\big] = \frac{1}{N}\sum_{i=1}^{N}\big(\mathbb{E}_{\xi^{(t)}}[\Delta_i^{(t)}] - \tilde{\mathbf{y}}_i^{(t)}\big) + \frac{1}{N}\sum_{j=1}^{N}\tilde{\mathbf{y}}_j^{(t)} = \mathbb{E}_{\xi^{(t)}}\big[\bar{\mathbf{h}}^{(t)}\big]. \tag{12}$$

B.2 Auxiliary Lemmas and Full Proofs

Lemma B.1 (Young's inequality)**.** *For any $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$ and any scalar $\gamma > 0$, $\langle \mathbf{u}, \mathbf{v}\rangle \le \frac{\|\mathbf{u}\|^2}{2\gamma} + \frac{\gamma\|\mathbf{v}\|^2}{2}$ and $\|\mathbf{u}+\mathbf{v}\|^2 \le (1+\frac{1}{\gamma})\|\mathbf{u}\|^2 + (1+\gamma)\|\mathbf{v}\|^2$.*

Lemma B.2 (Jensen's inequality)**.** *For any convex function $\phi : \mathbb{R}^d \to \mathbb{R}$ and integrable random vector X, $\phi\big(\mathbb{E}[X]\big) \le \mathbb{E}\big[\phi(X)\big]$.*

Lemma B.3 (Sum of squares)**.** *For any positive integer K and vectors $\mathbf{x}_1, \dots, \mathbf{x}_K \in \mathbb{R}^d$, $\big\|\sum_{k=1}^K \mathbf{x}_k\big\|^2 \le K\sum_{k=1}^K \|\mathbf{x}_k\|^2$.*

Lemma B.4 (Variance under uniform without-replacement sampling)**.** *Let $\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^N \mathbf{x}_i$. If $\mathcal{S}$ is a subset of size M sampled uniformly at random without replacement from $[N]$, then $\mathbb{E}_{\mathcal{S}}\big[\frac{1}{M}\sum_{i\in\mathcal{S}}\mathbf{x}_i\big] = \bar{\mathbf{x}}$ and*

$$\mathbb{E}_{\mathcal{S}}\Big\|\frac{1}{M}\sum_{i\in\mathcal{S}}\mathbf{x}_i - \bar{\mathbf{x}}\Big\|^2 = \frac{N-M}{M(N-1)}\,\frac{1}{N}\sum_{i=1}^N\|\mathbf{x}_i - \bar{\mathbf{x}}\|^2 \le \frac{N-M}{M(N-1)}\,\frac{1}{N}\sum_{i=1}^N\|\mathbf{x}_i\|^2.$$

Lemma B.5 (Client drift bound)**.** *Suppose the local functions $\{f_i\}$ satisfy Assumptions 4.1 and 4.2, and let the client learning rate satisfy $\eta_c \le \frac{1}{2LK}$. Then the iterates $\{\mathbf{w}^{(t)}\}$ generated by* FEDADAVR *satisfy*

$$\begin{aligned}
&\big\|\nabla f(\mathbf{w}^{(t)}) - \mathbb{E}_{\xi^{(t)}}[\bar{\mathbf{h}}^{(t)}]\big\|^2 \\
&\qquad\le \mathbb{E}_{\xi^{(t)}}\big\|\nabla f(\mathbf{w}^{(t)}) - \bar{\mathbf{h}}^{(t)}\big\|^2 \\
&\qquad\le C^{(t)} := \frac{1}{N}\sum_{i=1}^N \mathbb{E}_{\xi^{(t)}}\big\|\nabla f_i(\mathbf{w}^{(t)}) - \mathbf{h}_i^{(t)}\big\|^2 \\
&\qquad\le 2\eta_c^2 L^2(K-1)\sigma^2 + 8\eta_c^2 L^2 K(K-1)\Big[\sigma_g^2 + \|\nabla f(\mathbf{w}^{(t)})\|^2\Big].
\end{aligned}$$

Proof. By Lemma B.2 (convexity of $\|\cdot\|^2$) and $\nabla f(\mathbf{w}^{(t)}) = \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\mathbf{w}^{(t)})$, the first two inequalities hold immediately. For each client $i \in [N]$, by Lemma B.3 and Assumption 4.1,

$$\mathbb{E}_{\xi^{(t)}}\big\|\nabla f_i(\mathbf{w}^{(t)}) - \mathbf{h}_i^{(t)}\big\|^2 = \mathbb{E}_{\xi^{(t)}}\Big\|\frac{1}{K}\sum_{k=0}^{K-1}\big(\nabla f_i(\mathbf{w}^{(t)}) - \nabla f_i(\mathbf{w}_i^{(t,k)})\big)\Big\|^2$$
$$\le \frac{L^2}{K}\sum_{k=0}^{K-1}\mathbb{E}_{\xi^{(t)}}\big\|\mathbf{w}^{(t)} - \mathbf{w}_i^{(t,k)}\big\|^2. \tag{13}$$

For $k = 0$, $\mathbf{w}_i^{(t,0)} = \mathbf{w}^{(t)}$. For $k \in \{1, \dots, K-1\}$, unrolling the local SGD update $\mathbf{w}_i^{(t,k)} = \mathbf{w}^{(t)} - \eta_c\sum_{\tau=0}^{k-1}\nabla f_i(\mathbf{w}_i^{(t,\tau)}, \xi_i^{(t,\tau)})$, separating the zero-mean martingale difference noise $\delta_i^{(t,\tau)}$ from the full local gradient, and using conditional independence across steps τ together with Assumption 4.2 and Lemma B.3 gives

$$\mathbb{E}_{\xi^{(t)}}\big\|\mathbf{w}_i^{(t,k)} - \mathbf{w}^{(t)}\big\|^2 \le 2\eta_c^2\mathbb{E}_{\xi^{(t)}}\Big\|\sum_{\tau=0}^{k-1}\big(\nabla f_i(\mathbf{w}_i^{(t,\tau)}, \xi_i^{(t,\tau)}) - \nabla f_i(\mathbf{w}_i^{(t,\tau)})\big)\Big\|^2$$
$$+ 2\eta_c^2\mathbb{E}_{\xi^{(t)}}\Big\|\sum_{\tau=0}^{k-1}\nabla f_i(\mathbf{w}_i^{(t,\tau)})\Big\|^2$$
$$\le 2\eta_c^2 k\sigma^2 + 2\eta_c^2 k\sum_{\tau=0}^{k-1}\mathbb{E}_{\xi^{(t)}}\big\|\nabla f_i(\mathbf{w}_i^{(t,\tau)})\big\|^2.$$

By Lemma B.3 and Assumptions 4.1 and 4.2, for every $\tau \ge 0$,

$$\big\|\nabla f_i(\mathbf{w}_i^{(t,\tau)})\big\|^2 \le 2\big\|\nabla f_i(\mathbf{w}_i^{(t,\tau)}) - \nabla f_i(\mathbf{w}^{(t)})\big\|^2 + 2\big\|\nabla f_i(\mathbf{w}^{(t)})\big\|^2$$
$$\le 2L^2\big\|\mathbf{w}_i^{(t,\tau)} - \mathbf{w}^{(t)}\big\|^2 + 4\big(\sigma_g^2 + \|\nabla f(\mathbf{w}^{(t)})\|^2\big).$$

Substituting this bound back and averaging over $k = 0, \dots, K-1$ (using $\sum_{k=1}^{K-1} k = \frac{K(K-1)}{2}$ and $\sum_{k=1}^{K-1} k^2 \le \frac{K^2(K-1)}{2}$) yields

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}_{\xi^{(t)}}\big\|\mathbf{w}_i^{(t,k)} - \mathbf{w}^{(t)}\big\|^2 \le \eta_c^2(K-1)\sigma^2$$
$$+ 2\eta_c^2L^2K(K-1)\left(\frac{1}{K}\sum_{\tau=0}^{K-1}\mathbb{E}_{\xi^{(t)}}\big\|\mathbf{w}_i^{(t,\tau)} - \mathbf{w}^{(t)}\big\|^2\right)$$
$$+ 4\eta_c^2K(K-1)\big(\sigma_g^2 + \|\nabla f(\mathbf{w}^{(t)})\|^2\big).$$

Since $\eta_c \le \frac{1}{2LK}$, we have $2\eta_c^2L^2K(K-1) \le \frac{1}{2}$. Rearranging and multiplying both sides by $2L^2$, then substituting into (13) and averaging over $i \in [N]$ completes the proof. □

Lemma B.6 (Variance of the SAGA-style server estimator)**.** *Suppose the local functions $\{f_i\}$ satisfy Assumptions 4.1 and 4.2. With the boundary convention $\mathbf{w}^{(-1)} := \mathbf{w}^{(0)}$ and $\tilde{\mathbf{G}}^{(-1)} := \mathbf{0}$, for every round $t \ge 0$, the iterates of* FEDADAVR *satisfy*

$$\mathbb{E}_t\big\|\tilde{\mathbf{G}}^{(t)} - \bar{\mathbf{h}}^{(t)}\big\|^2 \le \frac{2\sigma^2}{MK} + \frac{5(N-M)}{M(N-1)}\,C^{(t)} + \frac{4(N-M)}{M(N-1)}\,\frac{\tilde{\eta}_s^2L^2}{\epsilon^2}\big\|\tilde{\mathbf{G}}^{(t-1)}\big\|^2$$
$$+ \frac{2(N-M)}{M(N-1)}\,\frac{1}{N}\sum_{i=1}^{N}\big\|\nabla f_i(\mathbf{w}^{(t-1)}) - \tilde{\mathbf{y}}_i^{(t)}\big\|^2.$$

Proof. By definition of $\tilde{\mathbf{G}}^{(t)}$ and $\bar{\mathbf{h}}^{(t)}$, we decompose the estimation error as

$$\tilde{\mathbf{G}}^{(t)} - \bar{\mathbf{h}}^{(t)} = \underbrace{\frac{1}{M}\sum_{i\in\mathcal{S}^{(t)}}\big(\Delta_i^{(t)} - \mathbf{h}_i^{(t)}\big)}_{U_1} + \underbrace{\frac{1}{M}\sum_{i\in\mathcal{S}^{(t)}}\big(\mathbf{h}_i^{(t)} - \tilde{\mathbf{y}}_i^{(t)}\big) - \frac{1}{N}\sum_{j=1}^{N}\big(\mathbf{h}_j^{(t)} - \tilde{\mathbf{y}}_j^{(t)}\big)}_{U_2}.$$

To decouple the local-trajectory randomness in U_2 from the $\mathcal{F}_t$-measurable historical states, we further split $U_2 = U_{2,a} + U_{2,b}$, where

$$U_{2,a} := \frac{1}{M}\sum_{i\in\mathcal{S}^{(t)}}\big(\mathbf{h}_i^{(t)} - \nabla f_i(\mathbf{w}^{(t)})\big) - \frac{1}{N}\sum_{j=1}^{N}\big(\mathbf{h}_j^{(t)} - \nabla f_j(\mathbf{w}^{(t)})\big),$$

$$U_{2,b} := \frac{1}{M}\sum_{i\in\mathcal{S}^{(t)}}\big(\nabla f_i(\mathbf{w}^{(t)}) - \tilde{\mathbf{y}}_i^{(t)}\big) - \frac{1}{N}\sum_{j=1}^{N}\big(\nabla f_j(\mathbf{w}^{(t)}) - \tilde{\mathbf{y}}_j^{(t)}\big).$$

Conditioned on $\mathcal{F}_t$ and $\mathcal{S}^{(t)}$, $U_{2,b}$ is deterministic and $\mathbb{E}_{\xi^{(t)}}[U_1 \mid \mathcal{F}_t, \mathcal{S}^{(t)}] = \mathbf{0}$, so $\mathbb{E}_t[\langle U_1, U_{2,b}\rangle] = 0$. For the remaining cross-term, Lemma B.1 ($\gamma = 1$) gives $2\mathbb{E}_t[\langle U_1, U_{2,a}\rangle] \le \mathbb{E}_t\|U_1\|^2 + \mathbb{E}_t\|U_{2,a}\|^2$, and thus

$$\begin{aligned}\mathbb{E}_t\big\|\tilde{\mathbf{G}}^{(t)} - \bar{\mathbf{h}}^{(t)}\big\|^2 &= \mathbb{E}_t\|U_1\|^2 + 2\mathbb{E}_t[\langle U_1, U_{2,b}\rangle] + 2\mathbb{E}_t[\langle U_1, U_{2,a}\rangle] + \mathbb{E}_t\|U_2\|^2\\ &\le 2\mathbb{E}_t\|U_1\|^2 + \mathbb{E}_t\|U_{2,a}\|^2 + \mathbb{E}_t\|U_2\|^2.\end{aligned}$$

By conditional independence of mini-batch sampling across clients and steps together with Assumption 4.2, $\mathbb{E}_t\|U_1\|^2 \le \frac{\sigma^2}{MK}$. Applying Lemma B.4 to $U_{2,a}$ and U_2 yields $\mathbb{E}_t\|U_{2,a}\|^2 \le \frac{N-M}{M(N-1)}\,C^{(t)}$ and

$$\mathbb{E}_t\|U_2\|^2 \le \frac{N-M}{M(N-1)}\,\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{\xi^{(t)}}\big\|\mathbf{h}_i^{(t)} - \tilde{\mathbf{y}}_i^{(t)}\big\|^2.$$

Writing $\mathbf{h}_i^{(t)} - \tilde{\mathbf{y}}_i^{(t)} = (\mathbf{h}_i^{(t)} - \nabla f_i(\mathbf{w}^{(t)})) + (\nabla f_i(\mathbf{w}^{(t)}) - \nabla f_i(\mathbf{w}^{(t-1)})) + (\nabla f_i(\mathbf{w}^{(t-1)}) - \tilde{\mathbf{y}}_i^{(t)})$, applying Lemma B.1 ($\gamma = 1$) twice ($\|\mathbf{a}+\mathbf{b}+\mathbf{c}\|^2 \le 4\|\mathbf{a}\|^2 + 4\|\mathbf{b}\|^2 + 2\|\mathbf{c}\|^2$), and using Assumption 4.1 together with (11) and $\sqrt{z_{t-1,j}} + \epsilon \ge \epsilon$ (or $\mathbf{w}^{(0)} - \mathbf{w}^{(-1)} = \mathbf{0}$ at $t = 0$) gives

$$\big\|\nabla f_i(\mathbf{w}^{(t)}) - \nabla f_i(\mathbf{w}^{(t-1)})\big\|^2 \le L^2\big\|\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)}\big\|^2 = \tilde{\eta}_s^2 L^2\Big\|\frac{\tilde{\mathbf{G}}^{(t-1)}}{\sqrt{z_{t-1}} + \epsilon}\Big\|^2 \le \frac{\tilde{\eta}_s^2 L^2}{\epsilon^2}\big\|\tilde{\mathbf{G}}^{(t-1)}\big\|^2.$$

Combining the bounds on $2\mathbb{E}_t\|U_1\|^2$, $\mathbb{E}_t\|U_{2,a}\|^2$, and $\mathbb{E}_t\|U_2\|^2$ establishes the claim. □

Lemma B.7 (Staleness recursion for stored client states)**.** *Suppose the local functions $\{f_i\}$ satisfy Assumptions 4.1 and 4.2. With the boundary convention $\mathbf{w}^{(-1)} := \mathbf{w}^{(0)}$ and $\tilde{\mathbf{G}}^{(-1)} := \mathbf{0}$, for any $\beta > 0$ and $t \ge 0$, the iterates of* FEDADAVR *satisfy*

$$\begin{aligned}\mathbb{E}_t\left[\frac{1}{N}\sum_{j=1}^{N}\big\|\nabla f_j(\mathbf{w}^{(t)}) - \tilde{\mathbf{y}}_j^{(t+1)}\big\|^2\right] \le{}& \frac{2M}{N}\left[\frac{\sigma^2}{K} + C^{(t)}\right]\\ &+ \Big(1 - \frac{M}{N}\Big)(1+\beta)\,\frac{1}{N}\sum_{j=1}^{N}\big\|\nabla f_j(\mathbf{w}^{(t-1)}) - \tilde{\mathbf{y}}_j^{(t)}\big\|^2\\ &+ \Big(1 - \frac{M}{N}\Big)\Big(1 + \frac{1}{\beta}\Big)\frac{\tilde{\eta}_s^2 L^2}{\epsilon^2}\big\|\tilde{\mathbf{G}}^{(t-1)}\big\|^2.\end{aligned}$$

Proof. By the memory update rule (4), each client $j \in [N]$ is included in $\mathcal{S}^{(t)}$ with probability $\frac{M}{N}$ (setting $\tilde{\mathbf{y}}_j^{(t+1)} = \Delta_j^{(t)}$) and excluded with probability $1 - \frac{M}{N}$ (retaining $\tilde{\mathbf{y}}_j^{(t+1)} = \tilde{\mathbf{y}}_j^{(t)}$). Hence,

$$\begin{aligned}\mathbb{E}_t\left[\frac{1}{N}\sum_{j=1}^{N}\big\|\nabla f_j(\mathbf{w}^{(t)}) - \tilde{\mathbf{y}}_j^{(t+1)}\big\|^2\right] ={}& \frac{M}{N}\,\frac{1}{N}\sum_{j=1}^{N}\mathbb{E}_{\xi^{(t)}}\big\|\nabla f_j(\mathbf{w}^{(t)}) - \Delta_j^{(t)}\big\|^2\\ &+ \Big(1 - \frac{M}{N}\Big)\frac{1}{N}\sum_{j=1}^{N}\big\|\nabla f_j(\mathbf{w}^{(t)}) - \tilde{\mathbf{y}}_j^{(t)}\big\|^2.\end{aligned}$$

For the first sum, writing $\nabla f_j(\mathbf{w}^{(t)}) - \Delta_j^{(t)} = (\nabla f_j(\mathbf{w}^{(t)}) - \mathbf{h}_j^{(t)}) + (\mathbf{h}_j^{(t)} - \Delta_j^{(t)})$ and applying Lemma B.1 ($\gamma = 1$) together with $\mathbb{E}_{\xi^{(t)}}\|\Delta_j^{(t)} - \mathbf{h}_j^{(t)}\|^2 \le \frac{\sigma^2}{K}$ gives $\mathbb{E}_{\xi^{(t)}}\|\nabla f_j(\mathbf{w}^{(t)}) - \Delta_j^{(t)}\|^2 \le$

$2\mathbb{E}_{\xi^{(t)}}\|\Delta_j^{(t)} - \mathbf{h}_j^{(t)}\|^2 + 2\mathbb{E}_{\xi^{(t)}}\|\nabla f_j(\mathbf{w}^{(t)}) - \mathbf{h}_j^{(t)}\|^2 \le \frac{2\sigma^2}{K} + 2\mathbb{E}_{\xi^{(t)}}\|\nabla f_j(\mathbf{w}^{(t)}) - \mathbf{h}_j^{(t)}\|^2$. For the second sum, applying Lemma B.1 with parameter $\beta > 0$ to $\nabla f_j(\mathbf{w}^{(t)}) - \tilde{\mathbf{y}}_j^{(t)} = (\nabla f_j(\mathbf{w}^{(t)}) - \nabla f_j(\mathbf{w}^{(t-1)})) + (\nabla f_j(\mathbf{w}^{(t-1)}) - \tilde{\mathbf{y}}_j^{(t)})$ and using $\|\nabla f_j(\mathbf{w}^{(t)}) - \nabla f_j(\mathbf{w}^{(t-1)})\|^2 \le \frac{\tilde{\eta}_s^2 L^2}{\epsilon^2}\|\tilde{\mathbf{G}}^{(t-1)}\|^2$ yields the result. □

Lemma B.8 (Uniform bound on the Adagrad accumulator)**.** *Suppose Assumption 4.3 holds. Then for all $t \in \{0, \dots, T-1\}$ and all coordinates $j \in [d]$,*

$$\big\|\tilde{\mathbf{G}}^{(t)}\big\| \le 3G, \qquad \big\|\mathbf{G}^{(t)}\big\| \le 3\eta_c K G, \qquad \frac{1}{D_T} \le \frac{1}{\sqrt{z_{t-1,j}} + \epsilon} \le \frac{1}{\epsilon},$$

where $D_T := 3\eta_c K G\sqrt{T} + \epsilon$.

Proof. Under Assumption 4.3, $\|\nabla f_i(\mathbf{w}_i^{(t,k)}, \xi_i^{(t,k)})\| \le G$, so by Lemma B.2 (convexity of $\|\cdot\|$), $\|\Delta_i^{(t)}\| \le \frac{1}{K}\sum_{k=0}^{K-1}\|\nabla f_i(\mathbf{w}_i^{(t,k)}, \xi_i^{(t,k)})\| \le G$. Since $\tilde{\mathbf{y}}_j^{(0)} = \mathbf{0}$ and $\tilde{\mathbf{y}}_j^{(t)}$ stores the most recent $\Delta_j^{(s)}$ ($s < t$), induction on t gives $\|\tilde{\mathbf{y}}_j^{(t)}\| \le G$ for all $j \in [N]$ and $t \ge 0$. Because $\frac{1}{M}\sum_{i\in\mathcal{S}^{(t)}}$ and $\frac{1}{N}\sum_{j=1}^{N}$ are convex combinations, the triangle inequality yields

$$\big\|\tilde{\mathbf{G}}^{(t)}\big\| \le \frac{1}{M}\sum_{i\in\mathcal{S}^{(t)}}\big(\|\Delta_i^{(t)}\| + \|\tilde{\mathbf{y}}_i^{(t)}\|\big) + \frac{1}{N}\sum_{j=1}^{N}\|\tilde{\mathbf{y}}_j^{(t)}\| \le 2G + G = 3G.$$

Consequently, $\|\mathbf{G}^{(t)}\| = \eta_c K\|\tilde{\mathbf{G}}^{(t)}\| \le 3\eta_c K G$, which implies $|G_j^{(s)}| \le \|\mathbf{G}^{(s)}\| \le 3\eta_c K G$ for every coordinate $j \in [d]$ and round s. Summing over $s = 0, \dots, t-1$ (with $t \le T$), the non-negative accumulator satisfies $0 \le z_{t-1,j} = \sum_{s=0}^{t-1}(G_j^{(s)})^2 \le 9t\eta_c^2 K^2 G^2 \le 9T\eta_c^2 K^2 G^2$. Taking square roots and adding $\epsilon > 0$ gives $\epsilon \le \sqrt{z_{t-1,j}} + \epsilon \le 3\eta_c K G\sqrt{T} + \epsilon = D_T$, which inverts to the stated inequality. □

B.3 Proof of Theorem 4.4 and Corollary 4.6

Proof of Theorem 4.4. By Assumption 4.1 (L-smoothness of $f = \frac{1}{N}\sum_{i=1}^{N} f_i$) and the server update (11), taking conditional expectation $\mathbb{E}_t[\cdot]$ gives

$$\mathbb{E}_t\big[f(\mathbf{w}^{(t+1)})\big] \le f(\mathbf{w}^{(t)}) + \underbrace{\tilde{\eta}_s\Big\langle \nabla f(\mathbf{w}^{(t)}),\, \mathbb{E}_t\Big[-\frac{\tilde{\mathbf{G}}^{(t)}}{\sqrt{z_{t-1}} + \epsilon}\Big]\Big\rangle}_{T_1^{(t)}}$$
$$+ \underbrace{\tilde{\eta}_s\Big\langle \nabla f(\mathbf{w}^{(t)}),\, \mathbb{E}_t\Big[\frac{\tilde{\mathbf{G}}^{(t)}}{\sqrt{z_{t-1}} + \epsilon} - \frac{\tilde{\mathbf{G}}^{(t)}}{\sqrt{z_t} + \epsilon}\Big]\Big\rangle}_{T_2^{(t)}} + \underbrace{\frac{\tilde{\eta}_s^2 L}{2}\,\mathbb{E}_t\Big\|\frac{\tilde{\mathbf{G}}^{(t)}}{\sqrt{z_t} + \epsilon}\Big\|^2}_{T_3^{(t)}}. \quad (14)$$

Step 1: Bounding the first-order term $T_1^{(t)}$. Because z_{t-1} is $\mathcal{F}_t$-measurable, property (12) yields $\mathbb{E}_t\big[\frac{\tilde{\mathbf{G}}^{(t)}}{\sqrt{z_{t-1}}+\epsilon}\big] = \frac{\mathbb{E}_{\xi^{(t)}}[\bar{\mathbf{h}}^{(t)}]}{\sqrt{z_{t-1}}+\epsilon}$. Writing $-\mathbb{E}_{\xi^{(t)}}[\bar{\mathbf{h}}^{(t)}] = -\nabla f(\mathbf{w}^{(t)}) + (\nabla f(\mathbf{w}^{(t)}) - \mathbb{E}_{\xi^{(t)}}[\bar{\mathbf{h}}^{(t)}])$ and applying Lemma B.1 (Young's inequality with $\gamma = 1$) coordinate-wise while retaining the positive weight $\frac{1}{\sqrt{z_{t-1,j}}+\epsilon}$ gives

$$\begin{aligned} T_1^{(t)} &= -\tilde{\eta}_s\sum_{j=1}^{d}\frac{([\nabla f(\mathbf{w}^{(t)})]_j)^2}{\sqrt{z_{t-1,j}} + \epsilon} + \tilde{\eta}_s\sum_{j=1}^{d}\frac{[\nabla f(\mathbf{w}^{(t)})]_j\,\big[\nabla f(\mathbf{w}^{(t)}) - \mathbb{E}_{\xi^{(t)}}[\bar{\mathbf{h}}^{(t)}]\big]_j}{\sqrt{z_{t-1,j}} + \epsilon} \\ &\le -\frac{\tilde{\eta}_s}{2}\sum_{j=1}^{d}\frac{([\nabla f(\mathbf{w}^{(t)})]_j)^2}{\sqrt{z_{t-1,j}} + \epsilon} + \frac{\tilde{\eta}_s}{2}\sum_{j=1}^{d}\frac{\big([\nabla f(\mathbf{w}^{(t)}) - \mathbb{E}_{\xi^{(t)}}[\bar{\mathbf{h}}^{(t)}]]_j\big)^2}{\sqrt{z_{t-1,j}} + \epsilon} \\ &\le -\frac{\tilde{\eta}_s}{2D_T}\big\|\nabla f(\mathbf{w}^{(t)})\big\|^2 + \frac{\tilde{\eta}_s}{2\epsilon}\,C^{(t)}, \end{aligned} \quad (15)$$

where the last step uses Lemma B.8 ($\frac{1}{D_T} \le \frac{1}{\sqrt{z_{t-1,j}}+\epsilon} \le \frac{1}{\epsilon}$) and Lemma B.5 ($\|\nabla f(\mathbf{w}^{(t)}) - \mathbb{E}_{\xi^{(t)}}[\bar{\mathbf{h}}^{(t)}]\|^2 \le C^{(t)}$).

Step 2: Telescoping the preconditioner-drift term $T_2^{(t)}$. Since $z_{t,j} = z_{t-1,j} + (G_j^{(t)})^2 \geq z_{t-1,j}$, the difference $\frac{1}{\sqrt{z_{t-1,j}}+\epsilon} - \frac{1}{\sqrt{z_{t,j}}+\epsilon}$ is non-negative. By Assumption 4.3 and Lemma B.8, $|[\nabla f(\mathbf{w}^{(t)})]_j| \leq G$ and $|\tilde{G}_j^{(t)}| \leq 3G$, so

$$T_2^{(t)} = \tilde{\eta}_s \sum_{j=1}^{d} \mathbb{E}_t\left[[\nabla f(\mathbf{w}^{(t)})]_j \tilde{G}_j^{(t)} \Big(\frac{1}{\sqrt{z_{t-1,j}}+\epsilon} - \frac{1}{\sqrt{z_{t,j}}+\epsilon} \Big)\right]$$
$$\leq 3\tilde{\eta}_s G^2 \sum_{j=1}^{d} \mathbb{E}_t\Big[\frac{1}{\sqrt{z_{t-1,j}}+\epsilon} - \frac{1}{\sqrt{z_{t,j}}+\epsilon}\Big].$$

Taking full expectation and summing over $t = 0, \ldots, T-1$ telescopes the series (since $z_{-1,j} = 0$):

$$\sum_{t=0}^{T-1} \mathbb{E}\big[T_2^{(t)}\big] \leq 3\tilde{\eta}_s G^2 \sum_{j=1}^{d} \mathbb{E}\Big[\frac{1}{\epsilon} - \frac{1}{\sqrt{z_{T-1,j}}+\epsilon}\Big] \leq \frac{3\tilde{\eta}_s d G^2}{\epsilon}. \tag{16}$$

Step 3: Bounding the second-order term $T_3^{(t)}$ via Lyapunov analysis. Since $\sqrt{z_{t,j}} + \epsilon \geq \epsilon$, using Lemma B.1 ($\gamma = 1$) twice to expand $\tilde{\mathbf{G}}^{(t)} = (\tilde{\mathbf{G}}^{(t)} - \bar{\mathbf{h}}^{(t)}) + (\bar{\mathbf{h}}^{(t)} - \nabla f(\mathbf{w}^{(t)})) + \nabla f(\mathbf{w}^{(t)})$ gives

$$T_3^{(t)} \leq \frac{\tilde{\eta}_s^2 L}{2\epsilon^2} \mathbb{E}_t\big\|\tilde{\mathbf{G}}^{(t)}\big\|^2 \leq \frac{\tilde{\eta}_s^2 L}{\epsilon^2} \mathbb{E}_t\big\|\tilde{\mathbf{G}}^{(t)} - \bar{\mathbf{h}}^{(t)}\big\|^2 + \frac{2\tilde{\eta}_s^2 L}{\epsilon^2} C^{(t)} + \frac{2\tilde{\eta}_s^2 L}{\epsilon^2} \big\|\nabla f(\mathbf{w}^{(t)})\big\|^2. \tag{17}$$

Define the historical staleness functional $Y^{(t)} := \frac{1}{N}\sum_{j=1}^{N} \|\nabla f_j(\mathbf{w}^{(t-1)}) - \tilde{\mathbf{y}}_j^{(t)}\|^2$ (with $\mathbf{w}^{(-1)} := \mathbf{w}^{(0)}$ and $\tilde{\mathbf{G}}^{(-1)} := \mathbf{0}$, so $Y^{(0)} \leq G^2$). In Lemma B.7, choose $\beta = \frac{M}{2(N-M)}$ (for $M < N$; if $M = N$, $Y^{(t)}$ vanishes from Lemma B.6 and we set $c_y := 0$), which gives $(1 - \frac{M}{N})(1+\beta) = 1 - \frac{M}{2N}$ and $(1 - \frac{M}{N})(1 + \frac{1}{\beta}) = \frac{N-M}{N}\frac{2N-M}{M} \leq \frac{2(N-M)}{M}$. Defining the Lyapunov weight $c_y := \frac{\tilde{\eta}_s^2 L}{\epsilon^2}\frac{4N(N-M)}{M^2(N-1)}$ so that $c_y \frac{M}{2N} = \frac{\tilde{\eta}_s^2 L}{\epsilon^2}\frac{2(N-M)}{M(N-1)}$, combining (17) with Lemma B.6 and $c_y(\mathbb{E}_t[Y^{(t+1)}] - Y^{(t)})$ from Lemma B.7 cancels the $Y^{(t)}$ term and yields

$$T_3^{(t)} + c_y\big(\mathbb{E}_t[Y^{(t+1)}] - Y^{(t)}\big) \leq \frac{\tilde{\eta}_s^2 L}{\epsilon^2}\Big(2 + \frac{13(N-M)}{M(N-1)}\Big) C^{(t)} + \frac{2\tilde{\eta}_s^2 L}{\epsilon^2}\big\|\nabla f(\mathbf{w}^{(t)})\big\|^2 + \frac{2\tilde{\eta}_s^2 LB}{\epsilon^2 MK}\sigma^2$$
$$+ \frac{4\tilde{\eta}_s^4 L^3 (N-M)}{\epsilon^4 M(N-1)}\Big(1 + \frac{2N(N-M)}{M^2}\Big)\big\|\tilde{\mathbf{G}}^{(t-1)}\big\|^2,$$

where $A = 1 + \frac{13(N-M)}{2M(N-1)}$ collects the $C^{(t)}$ coefficients ($2 + \frac{5(N-M)}{M(N-1)} + c_y \frac{\epsilon^2}{\tilde{\eta}_s^2 L}\frac{2M}{N} = 2 + \frac{13(N-M)}{M(N-1)} = 2A$), and $B = 1 + \frac{4(N-M)}{N-1}$ collects the direct noise terms, since $\frac{\tilde{\eta}_s^2 L}{\epsilon^2}\frac{2\sigma^2}{MK} + c_y \frac{2M}{N}\frac{\sigma^2}{K} = \frac{2\tilde{\eta}_s^2 L}{\epsilon^2 MK}\big(1 + \frac{4(N-M)}{N-1}\big)\sigma^2$. We also used $\frac{4N(N-M)}{M^2(N-1)} \cdot \frac{2(N-M)}{M} = \frac{4(N-M)}{M(N-1)} \cdot \frac{2N(N-M)}{M^2}$. By Lemma B.8, $\|\tilde{\mathbf{G}}^{(t-1)}\|^2 \leq 9G^2$, so the last term is bounded by $\bar{A}_3$. Moreover, since $\tilde{\eta}_s \leq \frac{\epsilon}{4L}$, we have $\frac{2\tilde{\eta}_s^2 L}{\epsilon^2}(1 + \frac{13(N-M)}{2M(N-1)}) \leq \frac{\tilde{\eta}_s}{2\epsilon} A$.

Step 4: Combining terms and telescoping over $t = 0, \ldots, T-1$. Defining the Lyapunov potential $\Phi^{(t)} := f(\mathbf{w}^{(t)}) + c_y Y^{(t)}$ and combining (14), (15) (using $\frac{\tilde{\eta}_s}{2\epsilon} C^{(t)} \leq \frac{\tilde{\eta}_s A}{2\epsilon} C^{(t)}$), and Step 3 gives

$$\mathbb{E}_t\big[\Phi^{(t+1)}\big] - \Phi^{(t)} \leq -\frac{\tilde{\eta}_s}{2D_T}\big\|\nabla f(\mathbf{w}^{(t)})\big\|^2 + \frac{2\tilde{\eta}_s^2 L}{\epsilon^2}\big\|\nabla f(\mathbf{w}^{(t)})\big\|^2$$
$$+ \frac{\tilde{\eta}_s A}{\epsilon} C^{(t)} + \frac{2\tilde{\eta}_s^2 LB}{\epsilon^2 MK}\sigma^2 + \bar{A}_3 + T_2^{(t)}.$$

Substituting the bound on $C^{(t)}$ from Lemma B.5 produces

$$\mathbb{E}_t\big[\Phi^{(t+1)}\big] - \Phi^{(t)} \leq -\tilde{\eta}_s\left[\frac{1}{2D_T} - \frac{2\tilde{\eta}_s L}{\epsilon^2} - \frac{8A\eta_c^2 L^2 K(K-1)}{\epsilon}\right]\big\|\nabla f(\mathbf{w}^{(t)})\big\|^2$$
$$+ \bar{A}_1\sigma_g^2 + \bar{A}_2\sigma^2 + \bar{A}_3 + T_2^{(t)}.$$

Under the learning-rate conditions of Theorem 4.4, $\frac{2\tilde{\eta}_s L}{\epsilon^2} \le \frac{1}{8D_T}$ and $\frac{8A\eta_c^2 L^2 K(K-1)}{\epsilon} \le \frac{1}{8D_T}$ (since $64A\eta_c^2 L^2 K(K-1)\epsilon \le \frac{\epsilon}{2}$ and $64A\eta_c^2 L^2 K(K-1)\cdot 3\eta_c KG\sqrt{T} \le \frac{\epsilon}{2}$). Hence the bracketed coefficient is at least $\frac{1}{4D_T}$. Setting $\Gamma := \frac{\tilde{\eta}_s}{4D_T}$, taking full expectations, summing from $t=0$ to $T-1$, and using (16) along with $\Phi^{(0)} - \mathbb{E}[\Phi^{(T)}] \le \Delta_0 + c_y Y^{(0)} \le \Delta_0 + \Psi_0$ gives

$$\Gamma \sum_{t=0}^{T-1} \mathbb{E}\big\|\nabla f(\mathbf{w}^{(t)})\big\|^2 \le \Delta_0 + \Psi_0 + \frac{3\tilde{\eta}_s dG^2}{\epsilon} + T\big(\bar{A}_1\sigma_g^2 + \bar{A}_2\sigma^2 + \bar{A}_3\big).$$

Dividing both sides by $\Gamma T = \frac{\tilde{\eta}_s T}{4D_T}$ completes the proof of Theorem 4.4. □

Proof of Corollary 4.6. With $\eta_c = \Theta(\frac{1}{KL\sqrt{T}})$, $\epsilon = \Theta(\frac{G}{L})$, and $\eta_s = \Theta(1)$, we have $\tilde{\eta}_s = \eta_s\eta_c K = \Theta(\frac{\eta_s}{L\sqrt{T}})$ and $D_T = 3\eta_c KG\sqrt{T} + \epsilon = \Theta(\frac{G}{L}) = \Theta(\epsilon)$. Thus $\frac{D_T}{\tilde{\eta}_s} = \Theta(\frac{G\sqrt{T}}{\eta_s})$. Substituting these scalings into each term of Theorem 4.4: (i) $\frac{4D_T\Delta_0}{\tilde{\eta}_s T} = \mathcal{O}(\frac{G\Delta_0}{\eta_s\sqrt{T}})$ and $\frac{4D_T\Psi_0}{\tilde{\eta}_s T} = \mathcal{O}\big(\frac{\eta_s GL}{T^{3/2}}\frac{N(N-M)}{M^2(N-1)}\big) = o(\frac{1}{T})$ for fixed N, M; (ii) $\frac{12D_T dG^2}{\epsilon T} = \mathcal{O}(\frac{dG^2}{T})$; (iii) $\frac{4D_T}{\tilde{\eta}_s}\bar{A}_1\sigma_g^2 = \Theta(A\eta_c^2 L^2 K(K-1)\sigma_g^2) = \mathcal{O}(\frac{A(K-1)\sigma_g^2}{KT})$; (iv) $\frac{4D_T}{\tilde{\eta}_s}\bar{A}_2\sigma^2 = \mathcal{O}(\frac{A(K-1)\sigma^2}{K^2T}) + \mathcal{O}(\frac{L\eta_s B\sigma^2}{GMK\sqrt{T}})$; and (v) $\frac{4D_T}{\tilde{\eta}_s}\bar{A}_3 = \mathcal{O}\big(\frac{\eta_s^3 L^3}{G\,T^{3/2}}\frac{N-M}{M(N-1)}\big(1 + \frac{2N(N-M)}{M^2}\big)\big) = o(\frac{1}{T})$ for fixed N, M. Combining (i)–(v) yields (10). □

C Algorithms

C.1 Client Device Update Procedure

Each participating client i executes the DEVICEUPDATE procedure described in Algorithm 2. The client initialises its local model from the current global model $\mathbf{w}^{(t)}$ and performs K stochastic gradient descent steps using its private dataset $\mathcal{D}_i$ at learning rate η_c. The resulting local update, computed as the scaled difference between the initial and final model states, is then returned to the server.

Algorithm 2 DEVICEUPDATE(i, $\mathbf{w}^{(t)}$, η_c)

1: $\mathbf{w}_i^{(t,0)} \leftarrow \mathbf{w}^{(t)}$
2: **for** local step $k = 0, 1, \ldots, K-1$ **do**
3: Compute stochastic gradient $\nabla f_i\left(\mathbf{w}_i^{(t,k)}\right)$
4: $\mathbf{w}_i^{(t,k+1)} \leftarrow \mathbf{w}_i^{(t,k)} - \eta_c \nabla f_i\left(\mathbf{w}_i^{(t,k)}\right)$
5: **end for**
6: **return** $\frac{1}{\eta_c}\left(\mathbf{w}^{(t)} - \mathbf{w}_i^{(t,K)}\right)$

C.2 Adaptive Optimisers

This appendix provides the full pseudocode and update rules for each of the five server-side adaptive optimisers supported by FEDADAVR, as referenced in Section 3. In each case, the optimiser receives the current global model $\mathbf{w}^{(t)}$, the pseudo-gradient $\mathbf{G}^{(t)}$ (computed via equations (8) and (9)), the server learning rate η_s, and optimiser-specific state variables. Only one optimiser is selected per experimental run; the five options are mutually exclusive.

C.2.1 Adagrad (Algorithm 3).

Algorithm 3 ADAGRADOPTIMISER (Latest model parameters $\mathbf{w}^{(t)}$, pseudo-gradient $\mathbf{G}^{(t)}$, server learning rate η_s, accumulator z_t)

Require: ϵ (small constant), λ (weight decay)
1: **if** $\lambda \neq 0$ **then**
2: $\quad \mathbf{G}^{(t)} \leftarrow \mathbf{G}^{(t)} + \lambda \mathbf{w}^{(t)}$;
3: **end if**
4: $z_t \leftarrow z_{t-1} + \mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}$;
5: $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s \frac{\mathbf{G}^{(t)}}{\sqrt{z_t}+\epsilon}$;
6: **return** $\mathbf{w}^{(t+1)}$.

The value $\mathbf{G}^{(t)}$, as computed in equation (9), is utilised in the calculation of z_t (accumulator):

$$z_t \leftarrow z_{t-1} + \mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}. \tag{18}$$

Then z_t is used for the global update:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s \frac{\mathbf{G}^{(t)}}{\sqrt{z_t} + \epsilon}, \tag{19}$$

where ϵ is a small constant.

C.2.2 Adam (Algorithm 4).

Algorithm 4 ADAMOPTIMISER (Latest model parameters $\mathbf{w}^{(t)}$, pseudo-gradient $\mathbf{G}^{(t)}$, server learning rate η_s, moments m_t, v_t)

Require: Decay rates β_1, β_2, small constant ϵ, weight decay λ
1: **if** $\lambda \neq 0$ **then**
2: $\quad \mathbf{G}^{(t)} \leftarrow \mathbf{G}^{(t)} + \lambda\, \mathbf{w}^{(t)}$;
3: **end if**
4: $m_t \leftarrow \beta_1\, m_{t-1} + (1-\beta_1)\, \mathbf{G}^{(t)}$;
5: $v_t \leftarrow \beta_2\, v_{t-1} + (1-\beta_2)\, \big(\mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}\big)$;
6: $\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^{\mathcal{T}}}$;
7: $\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^{\mathcal{T}}}$;
8: $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s\, \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$;
9: **return** $\mathbf{w}^{(t+1)}$.

The $\mathbf{G}^{(t)}$, obtained from the equation (9), is employed to compute the moments m_t and v_t, both of which are initially set to 0. The hyperparameters β_1 and β_2 are also required for the computation of m_t and v_t:

$$m_t \leftarrow \beta_1\, m_{t-1} + (1-\beta_1)\, \mathbf{G}^{(t)}; \tag{20}$$

$$v_t \leftarrow \beta_2\, v_{t-1} + (1-\beta_2)\, \big(\mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}\big). \tag{21}$$

The moments m_t and v_t are subsequently utilised to compute $\hat{m}_t$ and $\hat{v}_t$:

$$\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^{\mathcal{T}}}; \tag{22}$$

$$\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^{\mathcal{T}}}. \tag{23}$$

Here, $\mathcal{T}$ denotes the iteration counter (the server round, with $\mathcal{T} = t+1$ at round t in the proposed algorithms). The values $\hat{m}_t$ and $\hat{v}_t$ are then employed to compute the global update:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s\, \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \tag{24}$$

where ϵ is a small constant.

C.2.3 Adabelief (Algorithm 5).

Algorithm 5 ADABELIEFOPTIMISER (Latest model parameters $\mathbf{w}^{(t)}$, pseudo-gradient $\mathbf{G}^{(t)}$, server learning rate η_s, moments m_t, z_t)

Require: Decay rates β_1, β_2, small constant ϵ, weight decay λ
1: **if** $\lambda \neq 0$ **then**
2: $\quad \mathbf{G}^{(t)} \leftarrow \mathbf{G}^{(t)} + \lambda\, \mathbf{w}^{(t)}$;
3: **end if**
4: $m_t \leftarrow \beta_1\, m_{t-1} + (1-\beta_1)\, \mathbf{G}^{(t)}$;
5: $z_t \leftarrow \beta_2\, z_{t-1} + (1-\beta_2)\, \big(\mathbf{G}^{(t)} - m_t\big)^2$;
6: $\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^{\mathcal{T}}}$;
7: $\hat{z}_t \leftarrow \frac{z_t}{1-\beta_2^{\mathcal{T}}}$;
8: $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s\, \frac{\hat{m}_t}{\sqrt{\hat{z}_t}+\epsilon}$;
9: **return** $\mathbf{w}^{(t+1)}$.

The $\mathbf{G}^{(t)}$, obtained in the equation (9), is used to compute the moments m_t and z_t, both of which are initially set to 0. The hyperparameters β_1 and β_2 are also required for the computation of m_t and z_t.

$$m_t \leftarrow \beta_1\, m_{t-1} + (1-\beta_1)\, \mathbf{G}^{(t)}; \tag{25}$$

$$z_t \leftarrow \beta_2\, z_{t-1} + (1-\beta_2)\, \big(\mathbf{G}^{(t)} - m_t\big)^2. \tag{26}$$

The moments m_t and z_t are subsequently used to compute $\hat{m}_t$ and $\hat{z}_t$:

$$\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^{\mathcal{T}}}; \tag{27}$$

$$\hat{z}_t \leftarrow \frac{z_t}{1-\beta_2^{\mathcal{T}}}. \tag{28}$$

Here, $\mathcal{T}$ denotes the iteration counter (the server round, with $\mathcal{T} = t+1$ at round t in the proposed algorithms). The values $\hat{m}_t$ and $\hat{z}_t$ are then employed to compute the global update:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s\, \frac{\hat{m}_t}{\sqrt{\hat{z}_t}+\epsilon}, \tag{29}$$

where ϵ is a small constant.

C.2.4 Yogi (Algorithm 6).

Algorithm 6 YOGIOPTIMISER (Latest model parameters $\mathbf{w}^{(t)}$, pseudo-gradient $\mathbf{G}^{(t)}$, server learning rate η_s, moments m_t, v_t)

Require: Decay rates β_1, β_2, small constant ϵ, and weight decay λ
1: **if** $\lambda \neq 0$ **then**
2: $\quad \mathbf{G}^{(t)} \leftarrow \mathbf{G}^{(t)} + \lambda\, \mathbf{w}^{(t)}$;
3: **end if**
4: $m_t \leftarrow \beta_1\, m_{t-1} + (1-\beta_1)\, \mathbf{G}^{(t)}$;
5: $v_t \leftarrow v_{t-1} - (1-\beta_2)\, \mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}\, \mathrm{sgn}(v_{t-1} - \mathbf{G}^{(t)} \odot \mathbf{G}^{(t)})$;
6: $\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^{\mathcal{T}}}$;
7: $\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^{\mathcal{T}}}$;
8: $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s\, \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$;
9: **return** $\mathbf{w}^{(t+1)}$.

The $\mathbf{G}^{(t)}$ calculated in the equation (9) is used to compute the moments m_t and v_t, which are initially set to 0. The hyperparameters β_1 and β_2 are also required for calculating m_t and v_t:

$$m_t \leftarrow \beta_1\, m_{t-1} + (1-\beta_1)\, \mathbf{G}^{(t)}; \tag{30}$$

$$v_t \leftarrow v_{t-1} - (1-\beta_2)\,\mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}\,\mathrm{sgn}(v_{t-1} - \mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}). \tag{31}$$

The m_t and v_t are then used to compute $\hat{m}_t$ and $\hat{v}_t$:

$$\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^{\mathcal{T}}}; \tag{32}$$

$$\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^{\mathcal{T}}}. \tag{33}$$

Here, $\mathcal{T}$ denotes the iteration counter (the server round, with $\mathcal{T} = t+1$ at round t in the proposed algorithms). Subsequently, $\hat{m}_t$ and $\hat{v}_t$ are utilised for the global update:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s\,\frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}, \tag{34}$$

where ϵ is a small constant.

C.2.5 Lamb (Algorithm 7).

Algorithm 7 LAMBOPTIMISER (Latest model parameters $\mathbf{w}^{(t)}$, pseudo-gradient $\mathbf{G}^{(t)}$, server learning rate η_s, moments m_t, v_t)

Require: Decay rates β_1, β_2, small constant ϵ, and weight decay λ

if $\lambda \neq 0$ **then**
  $\mathbf{G}^{(t)} \leftarrow \mathbf{G}^{(t)} + \lambda\,\mathbf{w}^{(t)}$;
end if
$m_t \leftarrow \beta_1\,m_{t-1} + (1-\beta_1)\,\mathbf{G}^{(t)}$;
$v_t \leftarrow \beta_2\,v_{t-1} + (1-\beta_2)\,\big(\mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}\big)$;
$\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^{\mathcal{T}}}$;
$\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^{\mathcal{T}}}$;
$\hat{r}_t \leftarrow \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$;
$weight_norm \leftarrow \|\mathbf{w}^{(t)}\|$;
$update_norm \leftarrow \|\hat{r}_t\|$;
if $weight_norm > 0$ **and** $update_norm > 0$ **then**
  $r_t \leftarrow \frac{weight_norm}{update_norm}$;
else
  $r_t \leftarrow 1.0$;
end if
$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s\,r_t\,\hat{r}_t$;
return $\mathbf{w}^{(t+1)}$.

$\mathbf{G}^{(t)}$ calculated in the equation (9) is used to compute the moments m_t and v_t, both initially set to 0. The hyperparameters β_1 and β_2 are also required to calculate m_t and v_t:

$$m_t \leftarrow \beta_1\,m_{t-1} + (1-\beta_1)\,\mathbf{G}^{(t)}; \tag{35}$$

$$v_t \leftarrow \beta_2\,v_{t-1} + (1-\beta_2)\,\big(\mathbf{G}^{(t)} \odot \mathbf{G}^{(t)}\big). \tag{36}$$

The m_t and v_t are then used to compute $\hat{m}_t$ and $\hat{v}_t$:

$$\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^{\mathcal{T}}}; \tag{37}$$

$$\hat{r}_t \leftarrow \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}; \tag{38}$$

$$\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^{\mathcal{T}}}; \tag{39}$$

Here, $\mathcal{T}$ denotes the iteration counter (the server round, with $\mathcal{T} = t+1$ at round t in the proposed algorithms). Then, $\mathbf{w}^{(t)}$ and $\hat{r}_t$ are used to calculate:

$$weight_norm \leftarrow \|\mathbf{w}^{(t)}\|; \tag{40}$$

$$update_norm \leftarrow \|\hat{r}_t\|; \tag{41}$$

$$r_t = \begin{cases} \frac{weight_norm}{update_norm} & weight_norm > 0 \text{ and } update_norm > 0 \\ r_t \leftarrow 1.0 \end{cases}. \tag{42}$$

Then, r_t and $\hat{r}_t$ are used for the global update:

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta_s \, r_t \, \hat{r}_t. \tag{43}$$

D Model Quantisation

While various quantisation paradigms exist, we focus on three distinct strategies for compressing stored client updates. Algorithm 8 details the use of FP16 (half-precision), Int8, and Int4 quantisation, complemented by the corresponding dequantisation procedures in Algorithm 9. These methods achieve server-side memory reductions of 50%, 75%, and 87.5%, respectively. We provide experimental validation of these integer-based formats; however, a comprehensive evaluation of alternative quantisation schemes is beyond the scope of this work due to computational constraints. For 8-bit quantisation we use the symmetric per-tensor uniform quantiser of (Krishnamoorthi, 2018), and for 4-bit quantisation we follow (Zhou et al., 2016), packing two 4-bit values per byte for storage. Values are scaled by the tensor maximum; per-channel/asymmetric alternatives exist but are out of scope.

Algorithm 8 QUANT($\mathbf{g}_j^{(t)}$)

Initialisation: Set $mode$ ▷ Initialise quantisation $mode$ (eg: FP16, Int8, Int4).
$\mathcal{Q} \leftarrow \varnothing$; ▷ Initialise list to store compressed layers
for all $\mathbf{W}$ in $\mathbf{g}_j^{(t)}$ **do**
  if $mode =$ FP16 **then**
    $\mathbf{W}_{\text{q}} \leftarrow \text{cast}(\mathbf{W}, \text{float16})$;
    Append $\mathbf{W}_{\text{q}}$ to $\mathcal{Q}$;
  else if $mode =$ Int8 **then**
    $\alpha \leftarrow \max(|\mathbf{W}|)/127$; ▷ Scale for signed 8-bit
    $\alpha \leftarrow 1.0$ **if** $\alpha = 0$;
    $\mathbf{W}_{\text{int}} \leftarrow \text{cast}(\text{clip}(\text{round}(\mathbf{W}/\alpha), -127, 127), \text{int8})$;
    Append $(\mathbf{W}_{\text{int}}, \alpha)$ to $\mathcal{Q}$; ▷ Store Int8 tensor and scale
  else if $mode =$ Int4 **then**
    $S \leftarrow \text{shape}(\mathbf{W})$;
    $\alpha \leftarrow \max(|\mathbf{W}|)/7$; ▷ Scale for signed 4-bit
    $\alpha \leftarrow 1.0$ **if** $\alpha = 0$;
    $\mathbf{W}_{\text{int}} \leftarrow \text{clip}(\text{round}(\mathbf{W}/\alpha), -7, 7)$;
    $\mathbf{W}_{\text{flat}} \leftarrow \text{flatten}(\mathbf{W}_{\text{int}} + 8)$; ▷ Shift to $[1, 15]$ and flatten
    $\mathbf{P} \leftarrow \varnothing$;
    for $k = 0, 2, \ldots, \text{length}(\mathbf{W}_{\text{flat}}) - 1$ **do**
      $h \leftarrow \mathbf{W}_{\text{flat}}[k] \ll 4$; ▷ High 4 bits
      $l \leftarrow \mathbf{W}_{\text{flat}}[k+1]$; ▷ Low 4 bits
      Append $(h \vee l)$ to $\mathbf{P}$; ▷ Pack into Uint8
    end for
    Append $(\mathbf{P}, \alpha, S)$ to $\mathcal{Q}$; ▷ Store packed data, scale, and shape
  end if
end for
return $\mathcal{Q}$.

E Experimental Setup

This appendix provides a complete description of all experimental configurations used in Section 5 of the main paper, supplementing the concise summary provided there.

Algorithm 9 DEQUANT($\mathcal{Q}$)

Initialisation: Set $mode$ ▷ Initialise dequantisation $mode$ (eg: FP16, Int8, Int4).
$\mathcal{W}_{\text{restored}} \leftarrow \varnothing$;
for all item **in** $\mathcal{Q}$ **do**
  if mode $=$ FP16 **then**
    $\mathbf{W}_{\text{float}} \leftarrow \text{cast}(\text{item}, \text{float32})$;
    Append $\mathbf{W}_{\text{float}}$ to $\mathcal{W}_{\text{restored}}$;
  else if mode $=$ Int8 **then**
    $(\mathbf{W}_{\text{int}}, \alpha) \leftarrow$ item;
    $\mathbf{W}_{\text{float}} \leftarrow \text{cast}(\mathbf{W}_{\text{int}}, \text{float32}) \times \alpha$; ▷ Restore scale
    Append $\mathbf{W}_{\text{float}}$ to $\mathcal{W}_{\text{restored}}$;
  else if mode $=$ Int4 **then**
    $(\mathbf{P}, \alpha, S) \leftarrow$ item; ▷ Unpack tuple: Packed data, scale, shape
    $N \leftarrow \prod_{d \in S} d$; ▷ Total elements
    $\mathbf{U}_{\text{even}} \leftarrow (\mathbf{P} \gg 4)$; ▷ Extract high 4 bits
    $\mathbf{U}_{\text{odd}} \leftarrow (\mathbf{P}$ & 0x0F$)$; ▷ Extract low 4 bits
    $\mathbf{U} \leftarrow \text{Interleave}(\mathbf{U}_{\text{even}}, \mathbf{U}_{\text{odd}})$; ▷ Vectorised merge
    $\mathbf{U} \leftarrow \mathbf{U}[0 : N]$; ▷ Remove padding if any
    $\mathbf{W}_{\text{signed}} \leftarrow \text{cast}(\mathbf{U}, \text{int8}) - 8$; ▷ Reverse shift offset
    $\mathbf{W}_{\text{float}} \leftarrow \text{reshape}(\text{cast}(\mathbf{W}_{\text{signed}}, \text{float32}) \times \alpha, S)$;
    Append $\mathbf{W}_{\text{float}}$ to $\mathcal{W}_{\text{restored}}$;
  end if
end for
return $\mathcal{W}_{\text{restored}}$.

E.1 Datasets

To conduct the experiments, we utilised three publicly available vision classification datasets and one next-character prediction text dataset. Brief descriptions of these datasets are provided below.

- **MNIST (Deng, 2012).** The MNIST dataset contains 60,000 handwritten digit images (50,000 for training and 10,000 for testing) across 10 classes. Each image is a 28×28 grayscale image.
- **FMNIST (Xiao et al., 2017).** The FMNIST dataset consists of 60,000 training and 10,000 test images of fashion items, each as a 28×28 grayscale image across 10 classes.
- **CIFAR-10 (Krizhevsky, 2009).** The CIFAR-10 dataset comprises 60,000 colour images (50,000 for training and 10,000 for testing) of size 32×32 across 10 classes.
- **Shakespeare (Caldas et al., 2018).** This dataset is built from The Complete Works of William Shakespeare. We used the Flower (Beutel et al., 2020) Hugging Face repository to download this dataset, which contains a total of 4,226,158 samples. Each speaking role in each play is treated as a distinct device, yielding 1,129 clients in total.

E.2 Dataset Partitioning Methods

In our experiments, data were partitioned into an Independent and Identically Distributed (IID) setting and five distinct Non-IID settings. Vision datasets were partitioned based on label skewness, whereas the Shakespeare dataset utilised a Natural ID partitioning strategy. These settings include:

- **IID:** Data are uniformly distributed across labels with no skewness. Each client receives an equal amount of data from every class.
- **IID-NonIID:** A hybrid setting in which 50% of the data follows an IID distribution, while the remainder follows a non-IID distribution. The notation Non_IID is used instead of IID-NonIID for clarity in the tables.
- **Dirichlet:** A non-IID scenario where data points are assigned to clients according to a Dirichlet distribution. The parameter β governs the degree of skewness; smaller β values indicate greater skew, whereas larger values approximate an IID distribution. In all experiments, $\beta = 0.5$ was employed.

- **Sort and Partition:** Data are initially sorted by label and then partitioned into C chunks, where $C = 1$ implies each partition contains data from only one class, representing the most extreme non-IID setting. LQ-1, LQ-2, and LQ-3 refer to $C = 1, 2, 3$ chunks respectively.
- **Natural ID Partitioner:** Unlike synthetic splitting strategies, this method preserves the inherent user-based segmentation present in the raw data, capturing realistic non-IID statistical heterogeneity. This strategy is applied exclusively to the Shakespeare dataset.

Further details regarding these partitioning methods are provided in (Li et al., 2022b) and the official Flower documentation (Beutel et al., 2020).

E.3 Model Architectures

For MNIST and FMNIST, the LeNet-5 architecture (Lecun et al., 1998) was adopted, comprising seven layers. For CIFAR-10, the ResNet-18 architecture (He et al., 2016) was employed, with batch normalisation replaced by group normalisation (Hsieh et al., 2020). For the Shakespeare dataset, we utilise a recurrent neural network (RNN) based on Gated Recurrent Units (GRU) (Chung et al., 2014). The GRU architecture consists of an 8-dimensional character embedding layer, followed by a single GRU layer with 128 hidden units, and concludes with a fully connected layer mapping the final hidden state to the character vocabulary size.

E.4 Client Configuration

The MNIST and FMNIST datasets were partitioned across 500 clients, CIFAR-10 across 250 clients, and Shakespeare into 904 training clients. In each federated learning round, five clients were selected uniformly at random without replacement from the complete client pool, yielding participation rates of 1% for MNIST and FMNIST, 2% for CIFAR-10, and below 1% for Shakespeare. Test datasets were partitioned identically to the training distributions. For the vision datasets, 250 clients were sampled for evaluation after each round, whereas for Shakespeare the model was evaluated on a held-out set of 225 clients from the full 1,129 (904 used for training).

E.5 Hyperparameter Configuration

Hyperparameter configurations were tailored to each specific algorithm to ensure fair benchmarking. Most experiments were conducted using a single fixed random seed, while some were repeated with two seeds and a small subset with three seeds to assess reproducibility. For client-side optimisation, we performed a grid search over the learning rate $\eta_c \in \{0.001, 0.01, 0.1\}$ and report the results of the best-performing configuration. For methods incorporating server-side optimisation (FedVARP, FEDADAVR, and FEDADAVR-QUANT), we conducted an additional grid search over $\eta_s \in \{0.01, 0.1, 1.0\}$ for FedVARP and $\eta_s \in \{0.001, 0.005, 0.01\}$ for the FEDADAVR variants. The algorithms MIFA, FedVARP, FEDADAVR, and FEDADAVR-QUANT were implemented directly using Flower and PyTorch. For baseline algorithms already available within Flower (FedAvg, FedAdagrad, FedAdam, FedYogi, FedProx, SCAFFOLD, and FedNova), default hyperparameter values provided by the official Flower repositories were used. All experiments were executed on high-performance computing systems equipped with CPUs and GPUs. These default server hyperparameters in Flower (for FedAdagrad, FedAdam, and FedYogi) have been extensively tuned and validated by the framework maintainers to reproduce the peak performance of their original publications. Conversely, because FedVARP, FEDADAVR, and FEDADAVR-QUANT do not have pre-existing framework defaults, we restricted their server-side tuning to a minimal three-value grid over η_s while keeping all other optimiser hyperparameters ($\beta_1, \beta_2, \epsilon$) completely untuned at standard defaults, ensuring a fair comparison across all methods.

Table 2 provides a complete summary of all parameters and hyperparameters employed across datasets. The best-performing configurations for FedVARP, FEDADAVR, and FEDADAVR-QUANT across datasets and partitioning strategies are summarised in Tables 8, 9, 10, and 11 in Appendix H.

E.6 Training Rounds and Evaluation Protocol

Convergence speed varies according to the dataset and the complexity of the data partitioning method. Consequently, the number of federated learning rounds was adjusted per setting, with more rounds

Table 2: Experimental and hyperparameter setup. Client momentum is used in all experiments, whereas the convergence analysis in Section 4 considers vanilla local SGD without momentum.

	Dataset			
Parameter	**MNIST**	**F-MNIST**	**CIFAR-10**	**Shakespeare**
# of clients	500	500	250	904
# of participating clients	5	5	5	5
% of participating clients	1%	1%	2%	$< 1\%$
# of clients for evaluation	250	250	250	225
Local model	LeNet-5	LeNet-5	ResNet-18	GRU
Batch size	20	20	20	64
Local epochs	1	3	5	5
Client learning rate	0.1	0.01	0.01	0.1, 0.01
Client momentum	0.9			
Decay rate β_1	0.9			
Decay rate β_2	0.999			
Small constant ϵ	1×10^{-8}			
Weight decay λ	0			

allocated to more challenging scenarios. Due to notable performance fluctuations between rounds, average accuracy is reported over the final rounds (typically the last 10% of total rounds) in Tables 8, 9, 10, and 11. Table 3 summarises the total rounds and the subset of final rounds used for averaging per partitioning strategy. For instance, in the LQ-1 partitioning of CIFAR-10, 1,500 FL rounds were conducted with results from the last 150 rounds averaged.

Table 3: Number of training rounds and selected rounds for average accuracy reporting, by dataset and partition strategy.

		Partition						
Type	**Dataset**	**IID**	**Non-IID**	**Dirichlet**	**LQ-1**	**LQ-2**	**LQ-3**	**Natural ID**
Training rounds	MNIST	60	60	60	100	60	60	x
	F-MNIST	100	100	150	350	150	150	x
	CIFAR-10	200	300	400	1500	500	500	x
	Shakespeare	x	x	x	x	x	x	200
Selected rounds	MNIST	10	10	10	10	10	10	x
	F-MNIST	10	10	15	35	25	15	x
	CIFAR-10	20	30	40	150	50	50	x
	Shakespeare	x	x	x	x	x	x	20

E.7 Multi-Seed Evaluation under Computational Constraints

Due to the computational cost of simulating thousands/hundreds of federated communication rounds, the number of random seeds varied across experiments. Most experiments were conducted using a single random seed, while some were repeated with two seeds and a small subset with three seeds to assess reproducibility. Additional runs were prioritised for the more challenging data partitioning strategies (Dirichlet, LQ-1, LQ-2, and LQ-3) on CIFAR-10, where variations in client data distributions and partial client participation can affect training outcomes. Accordingly, mean $\pm$ standard deviation is reported for experiments conducted with multiple seeds, while single-seed results are reported without estimates of variability across runs.

F Impact of Variance Reduction and Adaptive Optimisation

To rigorously evaluate the individual contributions of our proposed components, we conduct an ablation study comparing the full FEDADAVR algorithm against two dismantled variants: (i) FEDADAVR-

NoVR, which removes the variance reduction mechanism while retaining the optimiser (Adagrad for this case); and (ii) FedAdaVR-NoOPT, which removes the server-side adaptive optimiser while retaining variance reduction. All ablation experiments are performed on the CIFAR-10 dataset under the LQ-1 partitioning scheme, strictly adhering to the experimental protocols and hyperparameter settings detailed in Table 2 (with $\eta_s = \eta_c = 0.01$).

Figure 2 (left) in the main text illustrates the convergence trajectories of these variants. FedAdaVR demonstrates clear superiority, achieving the lowest loss and highest accuracy. Neither component strictly dominates; rather, they serve complementary and necessary roles. Removing variance reduction (FedAdaVR-NoVR) leads to severe instability and high variance in later rounds, confirming that the SAGA-style variance reduction technique is crucial for mitigating the sampling variance induced by partial client participation. Conversely, removing the adaptive optimiser (FedAdaVR-NoOPT) yields stable but exceedingly slow convergence, confirming that coordinate-wise step adaptation is essential for handling the severe parameter update imbalance caused by heterogeneous data distributions. In summary, variance reduction provides the necessary stability whilst the adaptive optimiser drives convergence speed, and the synergistic combination of both is required to achieve the performance gains reported in Section 5.2.

G Experimental Results

G.1 Accuracy and Loss Comparison with State-of-the-Art Methods

This section presents a comprehensive comparative evaluation of FedAdaVR and FedAdaVR-Quant against state-of-the-art methods.

G.1.1 Accuracy and loss comparison with the IID data partitioning method (Vision Datasets).

Figure 3 illustrate the accuracy and loss of various state-of-the-art methods alongside the two variants of FedAdaVR. In every instance, FedAdaVR surpasses the existing state-of-the-art algorithms. Although FedYogi performs comparatively well, it remains inferior to FedAdaVR under this straightforward data partitioning.

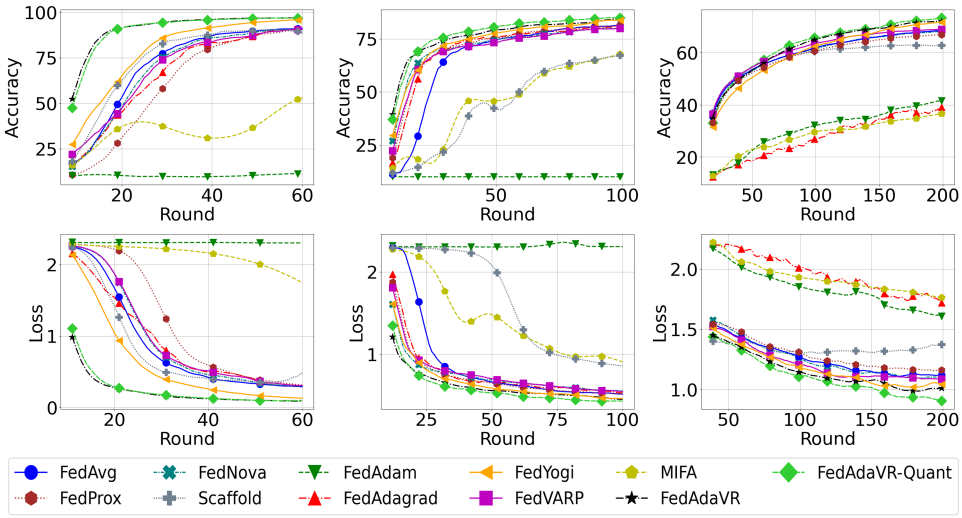


Figure 3: Accuracy and loss metrics under the IID data partitioning setup are presented in the top and bottom rows, respectively, for the MNIST (left), FMNIST (center), and CIFAR-10 (right) datasets.

G.1.2 Accuracy and loss comparison with the IID-Non_IID data partitioning method (Vision Datasets).

From Figure 4, the accuracy and loss of various state-of-the-art methods alongside the two variants of FEDADAVR under the IID-Non_IID data partitioning method are presented. FEDADAVR and FEDADAVR-QUANT consistently outperforms existing state-of-the-art approaches.

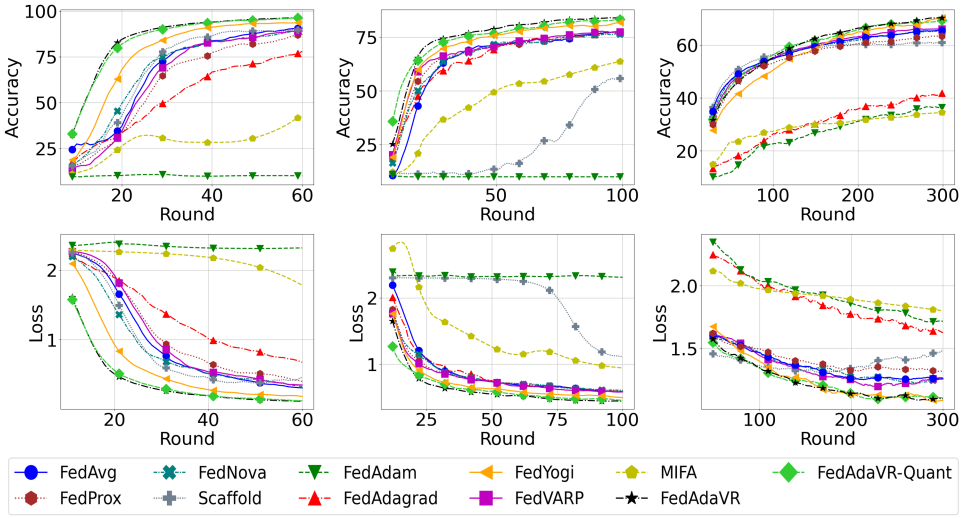


Figure 4: Accuracy and loss metrics under the IID-Non_IID data partitioning setup are presented in the top and bottom rows, respectively, for the MNIST (left), FMNIST (center), and CIFAR-10 (right) datasets.

G.1.3 Accuracy and loss comparison with the Dirichlet data partitioning method (Vision Datasets).

Figure 5 demonstrate that the two variants of FEDADAVR outperform existing state-of-the-art methods. While FedYogi performs close to FEDADAVR and FEDADAVR-QUANT on the MNIST and FMNIST datasets, it falls behind on the CIFAR-10 dataset.

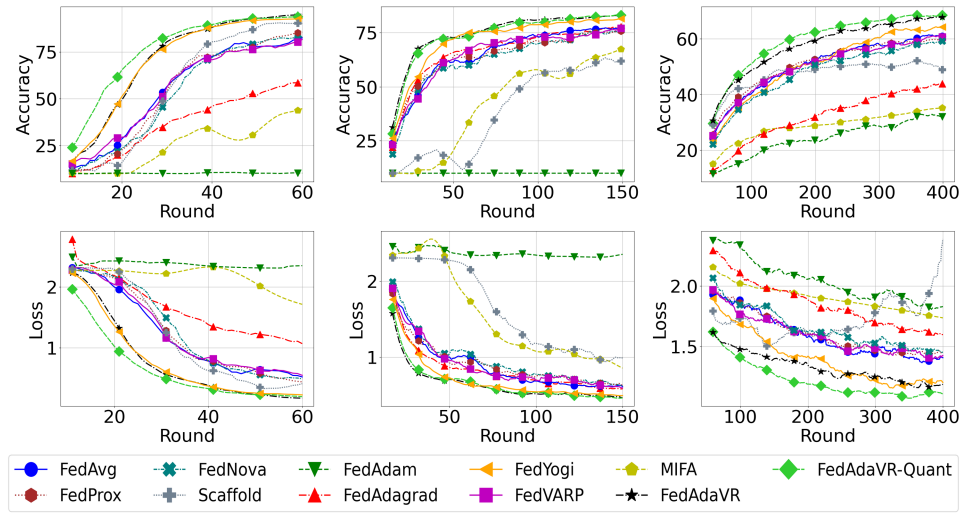


Figure 5: Accuracy and loss metrics under the Dirichlet data partitioning setup are presented in the top and bottom rows, respectively, for the MNIST (left), FMNIST (center), and CIFAR-10 (right) datasets.

G.1.4 Accuracy and loss comparison with LQ-1 data partitioning method (Vision Datasets).

Figure 6 illustrate that the two variants of FEDADAVR significantly outperform existing state-of-the-art methods on the CIFAR-10 dataset, which represents the most challenging scenario. Although FedVARP performs well, it remains notably behind both FEDADAVR-QUANT and FEDADAVR. This clearly demonstrates the superiority of the adaptivity to variance reduction approach in extreme cases. Furthermore, FEDADAVR and FEDADAVR-QUANT exhibit faster convergence than other state-of-the-art methods on the MNIST and FMNIST datasets.

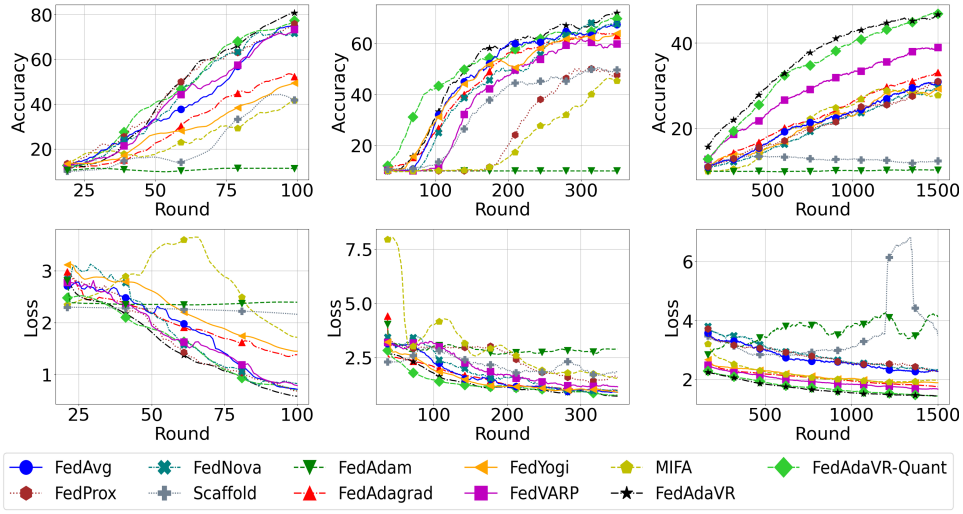


Figure 6: Accuracy and loss metrics under the LQ-1 data partitioning setup are presented in the top and bottom rows, respectively, for the MNIST (left), FMNIST (center), and CIFAR-10 (right) datasets.

G.1.5 Accuracy and loss comparison with LQ-2 data partitioning method (Vision Datasets).

Figure 7 demonstrate that the two variants of FEDADAVR outperform existing state-of-the-art methods. While FedYogi performs comparably to FEDADAVR and FEDADAVR-QUANT on the MNIST dataset, all state-of-the-art models lag significantly behind on the CIFAR-10 dataset.

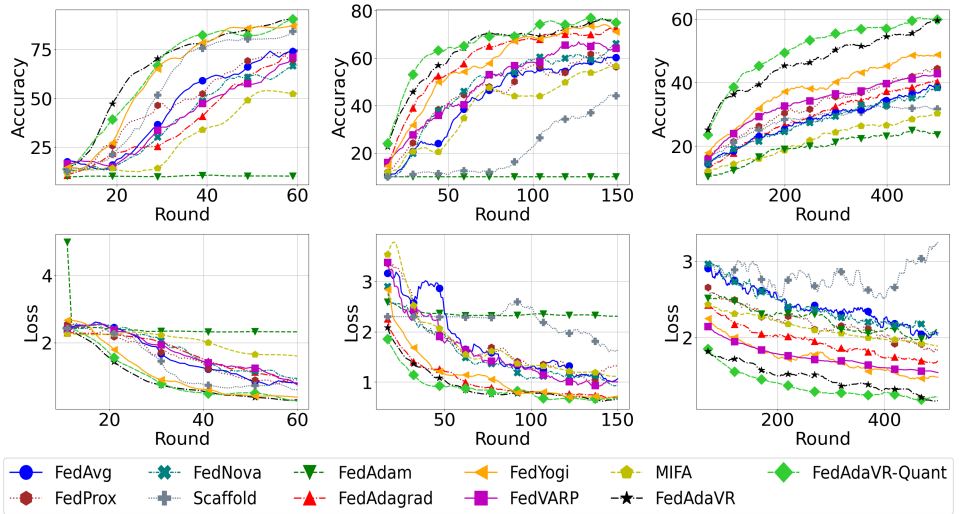


Figure 7: Accuracy and loss metrics under the LQ-2 data partitioning setup are presented in the top and bottom rows, respectively, for the MNIST (left), FMNIST (center), and CIFAR-10 (right) datasets.

G.1.6 Accuracy and loss comparison with LQ-3 data partitioning method (Vision Datasets).

Figures 8 show that FEDADAVR and FEDADAVR-QUANT consistently outperform existing state-of-the-art methods. Although FedAdagrad and FedYogi performs comparably on the FMNIST dataset, a clear performance gap is evident in other experiments.

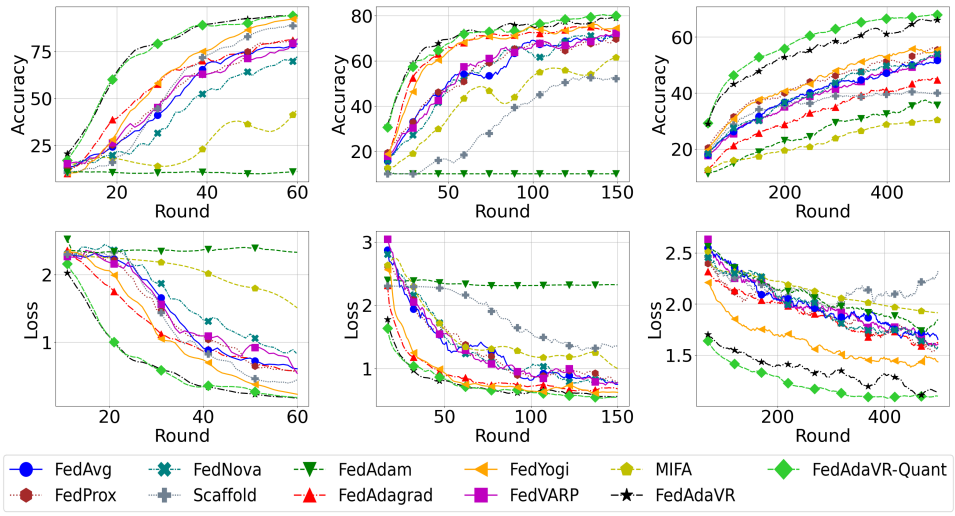


Figure 8: Accuracy and loss metrics under the LQ-3 data partitioning setup are presented in the top and bottom rows, respectively, for the MNIST (left), FMNIST (center), and CIFAR-10 (right) datasets.

G.1.7 Accuracy and loss comparison on Shakespeare dataset (Natural ID Partitioner).

As illustrated in Figure 9, both FEDADAVR and its quantised variant, FEDADAVR-QUANT, outperform existing state-of-the-art methods on the Shakespeare dataset, with the exception of FedAdagrad. While the margin is narrower here compared to vision experiments, FEDADAVR remains robust; it matches FedAdagrad on this specific task while strictly outperforming it across all other benchmarks.

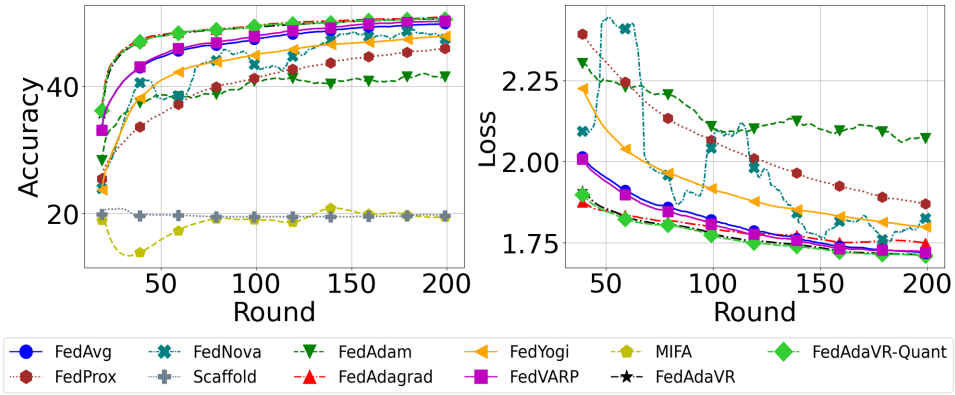


Figure 9: Accuracy and loss comparison on Shakespeare dataset (Natural ID Partitioner). Note that Scaffold and MIFA are omitted from the loss plots due to excessive out-of-range fluctuations.

In summary, FEDADAVR and FEDADAVR-QUANT consistently outperform existing state-of-the-art methods. While FedYogi demonstrates comparable performance on the MNIST and FMNIST datasets, and FedAdagrad achieves similar results on the Shakespeare dataset, a pronounced performance gap emerges under the LQ data partitioning scheme on the more complex CIFAR-10 dataset. Given the greater complexity of CIFAR-10 relative to MNIST, FMNIST, and Shakespeare, convergence rates

are slower across all algorithms. Real-world datasets are typically even more complex, featuring highly heterogeneous data distributions among clients. Therefore, federated learning algorithms must be designed to address such challenges effectively. Crucially, FEDADAVR does not discard client updates due to staleness; instead, it adheres to a SAGA-like structure that assumes virtual full participation by retaining all historical updates. While implementing a mechanism to prune excessively stale updates could theoretically enhance efficiency, this investigation remains outside the scope of our current research. Both proposed algorithms, FEDADAVR and FEDADAVR-QUANT, are deemed well-suited to real-world federated learning scenarios.

Numerical Instability of SCAFFOLD and MIFA. During hyperparameter tuning, we observed that SCAFFOLD and MIFA were sensitive to learning-rate changes under the extremely low client participation rates (1–2%) considered in our experiments. Attempts to adjust their learning rates did not consistently improve convergence and, in some cases, led to numerical instability, including exploding gradients and `NaN` training losses. This behaviour may be related to the infrequent client updates, which can make the control variates in SCAFFOLD and the stored historical gradients in MIFA increasingly outdated. Therefore, we report the best stable configurations obtained for these baselines under the evaluated settings.

H Detailed Comparison of FEDADAVR and FEDADAVR-QUANT Across Adaptive Optimisers

This appendix provides the complete experimental results summarised in Section 5.3 of the main paper. Tables 8, 9, 10, and 11 present a comparative analysis of FEDADAVR and FEDADAVR-QUANT across various adaptive optimisers and datasets. FedVARP is also included for reference, as it similarly utilises a server learning rate η_s. Bold text denotes the optimiser achieving the highest accuracy for a given configuration. The best-performing configurations from these tables are used to generate the curves shown in Figures 1, and 3–9.

Lamb consistently attains the best results on the CIFAR-10 dataset, whereas no single optimiser emerges as a definitive frontrunner across the other datasets. Consequently, Lamb appears to be the preferred choice for more complex tasks such as CIFAR-10. While Adagrad achieves the strongest performance on the Shakespeare dataset, it demonstrates competitive but largely sub-optimal performance across the vision datasets, with the exception of a single setting where it secured the highest accuracy.

H.1 Comparison Between Different Quantisation Methods

In our experiments, client–server communication retained the original FP32 format. However, for server-side storage of client updates, we evaluated four precision levels: FP32 (full precision), FP16 (half-precision), Int8, and Int4. While storing client states requires $\mathcal{O}(Nd)$ memory at full precision, employing FP16, Int8, or Int4 quantisation reduces this footprint by 50%, 75%, and 87.5%, respectively.

As detailed in the main text, the quantisation of stored updates has a negligible impact on convergence. Consequently, the quantised variants of FEDADAVR achieve performance comparable to the full-precision baseline, as illustrated in Figure 10. These findings were validated on the CIFAR-10 dataset under the LQ-1 partition, with both server and client learning rates fixed at $\eta_s = \eta_c = 0.001$.

Given that cross-device settings may involve thousands of resource-constrained clients, the $\mathcal{O}(Nd)$ storage complexity is a critical practical consideration. Table 4 details the estimated server-side memory requirements for each precision variant across a range of representative model architectures, demonstrating the deployment feasibility of FEDADAVR-QUANT in large-scale cross-device settings.

H.2 Server Learning Rate and Optimiser Selection

Experimental findings indicate that for a given optimiser, the performance difference across server learning rates is generally marginal. On the MNIST and FMNIST datasets, the LeNet-5 model achieves superior results with a higher server learning rate in most scenarios. Conversely, for ResNet-18 on CIFAR-10, a lower server learning rate facilitates faster convergence, particularly under

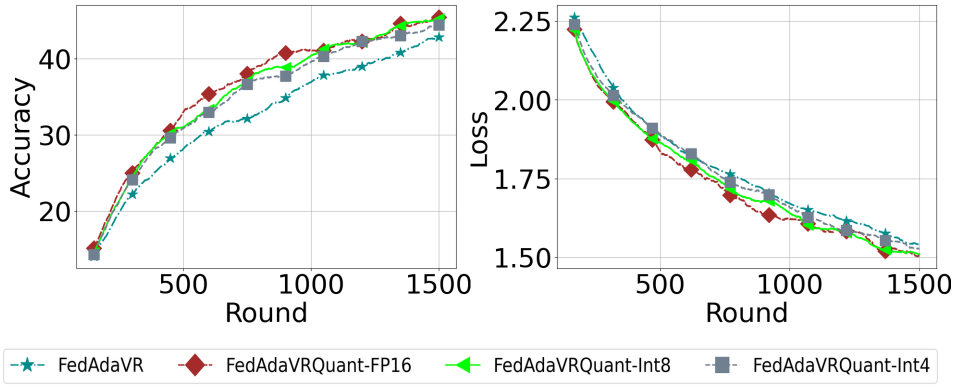


Figure 10: Comparison of model accuracy and loss across four quantisation precisions (FP32, FP16, Int8, Int4) for FEDADAVR on the CIFAR-10 dataset under LQ-1 partitioning.

the challenging LQ-1 partition. These observations underscore the critical sensitivity of algorithm performance to the choice of server learning rate.

All optimisers integrated into the proposed algorithms show competitive performance across tasks. The Lamb optimiser achieves the highest accuracy on the more complex CIFAR-10 dataset but converges slowly on MNIST and FMNIST. Given the black-box characteristics of federated learning, selecting an optimal optimiser a priori remains non-trivial. Nevertheless, all optimisers employed in FEDADAVR and FEDADAVR-QUANT consistently outperform existing baselines across all evaluated settings. Although five commonly used optimisers were evaluated in this work, other contemporary or related variants could be readily incorporated into the proposed framework.

I Computational, Latency, and Communication Comparison Against Baselines

This appendix provides the full system-side analysis summarised in Section 5.2 of the main paper. Computational latency in federated learning is governed by diverse factors, including hardware constraints, algorithmic design, and client participation density. Although all primary experiments were conducted on a High-Performance Computing (HPC) cluster, a workstation equipped with a 13th Gen Intel Core i9-13900H CPU (2.60 GHz), 64 GB of RAM, and an NVIDIA GeForce RTX 4080 Laptop GPU (12 GB VRAM) was utilised to measure the average time per round presented in Table 6. This standardised approach was necessary because the HPC cluster consists of heterogeneous nodes with varying capabilities, and the algorithms may not be allocated identical hardware configurations during execution. We organise the analysis into three parts: algorithmic complexity (Table 5), empirical wall-clock times (Table 6), and communication round efficiency (Table 7).

FEDADAVR is designed to impose no additional computational burden on clients; consequently, the client-side execution path is identical to that of FedAvg. However, the server incurs additional overhead to mitigate variance: the calculation of the correction term $\mathbf{r}^{(t)}$ requires aggregation of historical updates from all clients. As our algorithm explicitly accommodates dynamic client participation by allowing clients to join or depart arbitrarily, managing these historical states inevitably increases per-round computation time. It is worth noting, however, that the per-round server computation is strictly $\mathcal{O}(|\mathcal{S}^{(t)}|d)$ rather than $\mathcal{O}(Nd)$, as the variance reduction correction is maintained as a running sum and only the active clients' contributions are updated each round.

I.1 Algorithmic Complexity Comparison

Table 5 summarises the theoretical complexity of each algorithm in terms of communication cost per round, server-side memory requirements, and client-side computation overhead. d denotes model size, N total clients, K local epochs, and d_{quant} the quantised model size. FEDADAVR maintains the same lightweight client footprint as FedAvg, offloading all variance reduction and adaptive optimisation

Table 4: Estimated server-side memory requirements (GB) for storing client states in FEDADAVR across representative model architectures and client population sizes N (in thousands).

Local Model	Parameters (Millions)	# of Clients (Thousands)	Approximate Memory (GB)			
			FP32	FP16	Int8	Int4
SqueezeNet 1.1 (Iandola et al., 2016)	**1.2**	1	4.6	2.3	1.15	0.58
		10	46.0	23.0	11.5	5.8
		100	460	230	115	58
ShuffleNet V2 (Ma et al., 2018)	**2.3**	1	8.8	4.4	2.2	1.1
		10	88.0	44.0	22.0	11.0
		100	880	440	220	110
MobileNetV3-Small (Howard et al., 2019)	**2.5**	1	9.5	4.8	2.4	1.2
		10	95.0	48.0	24.0	12.0
		100	950	480	240	120
MobileNetV2 (Sandler et al., 2018)	**3.4**	1	13.0	6.5	3.25	1.63
		10	130	65.0	32.5	16.3
		100	1300	650	325	163
MNASNet 1.0 (Tan et al., 2018)	**3.9**	1	14.9	7.45	3.73	1.86
		10	149	74.5	37.3	18.6
		100	1490	745	373	186
MobileNetV1 (Howard et al., 2017)	**4.2**	1	16.0	8.0	4.0	2.0
		10	160	80.0	40.0	20.0
		100	1600	800	400	200
EffNet-Lite0 (Tan and Le, 2019)	**4.7**	1	17.9	8.95	4.48	2.24
		10	179	89.5	44.8	22.4
		100	1790	895	448	224
GoogLeNet (Szegedy et al., 2015)	**6.6**	1	25.2	12.6	6.3	3.15
		10	252	126	63.0	31.5
		100	2520	1260	630	315
ResNet-18 (He et al., 2016)	**11.7**	1	44.6	22.3	11.2	5.58
		10	446	223	112	55.8
		100	4460	2230	1115	558
ResNet-50 (He et al., 2016)	**25.6**	1	97.7	48.9	24.4	12.2
		10	977	489	244	122
		100	9770	4890	2440	1220
ConvNeXt-Tiny (Liu et al., 2022)	**28.6**	1	109	54.5	27.3	13.6
		10	1090	545	273	136
		100	10900	5450	2730	1360

entirely to the server. Compared to SCAFFOLD, FedProx, and FedNova, which introduce additional client-side computation in the form of control variates, proximal terms, and gradient normalisation respectively, FEDADAVR imposes no extra burden on resource-constrained client devices. Among all evaluated methods, only SCAFFOLD incurs additional communication overhead by transmitting dual parameters (model weights and control variates) during both uplink and downlink.

I.2 Wall-Clock Time Comparison

Table 6 reports empirical timing results across all evaluated algorithms. For FEDADAVR, we additionally break down per-round latency by server-side optimiser to illustrate the marginal overhead difference between optimiser choices. Whilst FEDADAVR incurs a minor per-round latency increase over FedAvg, this overhead is offset by a drastically reduced number of rounds required to reach target accuracy. FEDADAVR (Adagrad) reaches 45% accuracy in 163,689 seconds compared to

Table 5: Theoretical complexity comparison of all evaluated algorithms in terms of communication cost per round, server memory, and client computation. d denotes model size, N total clients, K local epochs, and d_{quant} the quantised model size.

Algorithm	**Communication Cost**	**Server Memory**	**Client Computation**
FedAvg	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(K)$
FedAdam	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(K)$
FedAdagrad	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(K)$
FedYogi	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(K)$
FedProx	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(K)$ + Proximal Term
SCAFFOLD	$\mathcal{O}(2d)$	$\mathcal{O}(d)$	$\mathcal{O}(K)$ + Control Variates
FedNova	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(K)$ + Gradient Normalisation
MIFA	$\mathcal{O}(d)$	$\mathcal{O}(Nd)$	$\mathcal{O}(K)$
FedVARP	$\mathcal{O}(d)$	$\mathcal{O}(Nd)$	$\mathcal{O}(K)$
FEDADAVR-QUANT	$\mathcal{O}(d)$	$\mathcal{O}(Nd_{\text{quant}})$	$\mathcal{O}(K)$
FEDADAVR	$\mathcal{O}(d)$	$\mathcal{O}(Nd)$	$\mathcal{O}(K)$

232,336 seconds for FedAvg, a reduction of approximately 30%. Hyphens indicate the algorithm did not reach the target accuracy within the allocated maximum rounds. While the server-side latency could be further optimised for scenarios with fixed client sets, such implementation details are beyond the scope of this work.

Table 6: Empirical wall-clock time comparison on CIFAR-10 under identical hardware. Per-round time is measured on the IID partition. End-to-end time to reach accuracy thresholds is measured on the LQ-1 partition at a 2% client participation rate. Times are reported in seconds. Hyphens indicate the algorithm did not reach the target accuracy within the allocated maximum rounds.

Algorithm	**Time/Round (s)**	**Time to Reach Target Accuracy (s)**			
		30%	**35%**	**40%**	**45%**
FedAvg	166.43	103,020	136,972	220,353	232,336
FedAdam	168.32	-	-	-	-
FedAdagrad	167.89	-	229,673	-	-
FedYogi	168.54	-	249,945	-	-
FedProx	170.57	123,834	157,095	203,490	-
SCAFFOLD	184.89	-	-	-	-
FedNova	168.65	123,789	182,479	228,015	251,457
MIFA	172.58	174,651	-	-	-
FedVARP	175.27	74,840	99,203	153,887	242,749
FEDADAVR	178.31	**52,780**	**75,069**	**117,863**	**163,689**
FEDADAVR-QUANT	178.31	61,160	98,784	143,183	199,351

I.3 Communication Round Efficiency

Due to their superior convergence rates, FEDADAVR and FEDADAVR-QUANT require significantly fewer total communication rounds compared to all baselines. Table 7 details the number of federated rounds required by each algorithm to achieve specific target accuracy thresholds on CIFAR-10 (LQ-1), along with the normalised communication cost (Norm. Rounds) relative to FEDADAVR. A value of 2.0 indicates the method required twice as many rounds as FEDADAVR to reach the same threshold. The symbol 'x' indicates the target accuracy was not reached within the allocated maximum rounds.

J Extended Limitations and Future Directions

Below we expand on the theoretical and system-level limitations summarised in Section 6 and outline directions for future work:

Table 7: Communication efficiency analysis on CIFAR-10 (LQ-1 partition). The table reports the total communication rounds required to reach specific accuracy thresholds, and the normalised communication cost relative to FEDADAVR (Norm. Rounds). A value of 2.0 indicates twice as many rounds as FEDADAVR. 'x' indicates the target accuracy was not reached within the allocated maximum rounds.

Algorithm	**Criteria**	**Accuracy Threshold**					
		20%	**25%**	**30%**	**35%**	**40%**	**45%**
FedAvg	Rounds	335	488	619	823	1324	1396
	Norm. Rounds	3.3	2.3	2.1	2.0	2.0	1.5
FedAdagrad	Rounds	355	646	988	1368	x	x
	Norm. Rounds	3.5	3.1	3.3	3.2	x	x
FedAdam	Rounds	x	x	x	x	x	x
	Norm. Rounds	x	x	x	x	x	x
FedYogi	Rounds	478	727	1066	1483	x	x
	Norm. Rounds	4.7	3.5	3.6	3.5	x	x
FedProx	Rounds	338	484	726	921	1193	x
	Norm. Rounds	3.3	2.3	2.5	2.2	1.8	x
SCAFFOLD	Rounds	x	x	x	x	x	x
	Norm. Rounds	x	x	x	x	x	x
FedNova	Rounds	338	612	734	1082	1352	1491
	Norm. Rounds	3.3	2.9	2.5	2.6	2.0	1.6
MIFA	Rounds	444	596	1012	x	x	x
	Norm. Rounds	4.4	2.9	3.4	x	x	x
FedVARP	Rounds	116	204	427	566	878	1385
	Norm. Rounds	1.1	1.0	1.4	1.3	1.3	1.5
FEDADAVR-QUANT	Rounds	189	284	343	554	803	1118
	Norm. Rounds	1.9	1.4	1.2	1.3	1.2	1.2
FEDADAVR	Rounds	**101**	**209**	**296**	**421**	**661**	**918**
	Norm. Rounds	1.0	1.0	1.0	1.0	1.0	1.0

Theoretical scope and assumptions

Our convergence analysis (Theorem 4.4) focuses on Adagrad as the server optimiser and relies on uniformly bounded stochastic gradients (Assumption 4.3), which the proof invokes only at the visited global and local iterates. Empirically, our decoupled variance reduction setup works equally well with Adam, AdaBelief, Yogi, and Lamb, though extending the formal proof to these optimisers is non-trivial since one must jointly bound heavy-ball momentum and stale historical states. Relaxing Assumption 4.3 using recent affine-variance techniques for Adagrad is another natural next step.

Convergence theory for FEDADAVR-QUANT

We have not derived a formal convergence bound for FEDADAVR-QUANT. If one uses unbiased stochastic quantisation, the compression noise has zero mean and can be folded directly into the variance bound σ^2 (Assumption 4.2) without changing the asymptotic rate, much like recent analyses of low-precision adaptive optimisers (Tang et al., 2026). However, our experiments use deterministic linear quantisation, where rounding errors accumulate and correlate across historical client states $\mathbf{y}_j^{(t)}$ over time; tightly bounding this coupled error remains open, even though empirically these rounding perturbations do not induce material bias and frequently act as a mild regulariser under extreme label skew.

Table 8: Accuracy comparison of FedVARP, FEDADAVR and FEDADAVR-QUANT algorithms on the Shakespeare dataset (Server learning rate: η_s).

Algorithm	**Optimiser**	η_s				
		0.001	**0.005**	**0.01**	**0.1**	**1.0**
FedVARP	x	x	x	43.038	49.434	**50.288**
FEDADAVR	Adabelief	45.271	49.453	49.514	x	x
	Adam	35.733	48.541	49.343	x	x
	Adagrad	38.379	50.048	**50.577**	x	x
	Yogi	27.921	48.440	49.549	x	x
	Lamb	42.057	41.034	31.383	x	x
FEDADAVR-QUANT	Adabelief	45.295	49.273	49.441	x	x
	Adam	28.589	48.604	49.468	x	x
	Adagrad	41.492	50.065	**50.611**	x	x
	Yogi	43.501	48.470	49.507	x	x
	Lamb	41.799	41.201	41.556	x	x

Server memory scaling at extreme model and population sizes

FEDADAVR-QUANT only compresses states on the server without modifying client-to-server communication. Its $\mathcal{O}(Nd_{\text{quant}})$ footprint is manageable for medium-scale cross-device settings (e.g., 1,360 GB for ConvNeXt-Tiny with $N = 10^5$ clients in Int4; Table 4). At foundation-model scales ($d \geq 10^8$ parameters and $N = 10^6$ clients), however, even an 87.5% savings from Int4 still leaves roughly 25 TB of server state. Cluster-based alternatives like ClusterFedVARP (Jhunjhunwala et al., 2022) cut this cost to $\mathcal{O}$(number of clusters x d), but they introduce their own failure modes when clients are misgrouped, clusters become imbalanced, or client representations drift during training. Scaling stateful variance reduction to billion-parameter models will likely require combining low-bit quantisation with partial client caching or low-rank state approximations.

Temporal distribution shifts

SAGA-style variance reduction assumes that a client's stored update $\mathbf{y}_j^{(t)}$ remains a reasonable proxy until that client is sampled again. If a client's local data distribution shifts rapidly over time (concept drift), those cached updates can become outdated and pull the aggregate estimate $\mathbf{r}^{(t)}$ in stale directions. Adding staleness-based decay or an exponential forgetting factor to downweight older $\mathbf{y}_j^{(t)}$ entries would help adapt the method to non-stationary settings.

Table 9: Accuracy comparison of FedVARP, FEDADAVR and FEDADAVR-QUANT algorithms on the MNIST dataset (Server learning rate: η_s).

			Partition					
Algorithm	**Optimiser**	η_s	**IID**	**Non_IID**	**Dir.**	**LQ-1**	**LQ-2**	**LQ-3**
FedAvg	-	-	91.06	90.44	81.75	75.26	74.04	78.78
FedProx	-	-	90.34	86.74	85.24	78.81	69.61	80.32
SCAFFOLD	-	-	89.63	89.59	90.31	38.45	84.40	88.79
FedNova	-	-	90.85	89.15	82.44	70.85	66.69	69.79
FedAdam	-	-	11.24	10.10	10.39	11.04	10.21	10.89
FedAdagrad	-	-	90.85	76.79	58.5	55.38	74.36	81.15
FedYogi	-	-	95.89	93.58	92.59	54.04	87.15	92.45
MIFA	-	-	52.26	41.77	43.89	42.21	52.43	41.25
FedVARP	-	0.01	11.94	12.37	10.89	7.11	13.15	8.67
		0.1	33.71	10.09	11.20	18.55	11.38	19.19
		1.0	**90.60**	**89.33**	**80.28**	**73.24**	**70.98**	**78.18**
FEDADAVR	Adabelief	0.001	93.10	90.91	84.93	69.18	77.92	83.23
		0.005	96.53	**96.27**	90.65	80.15	**90.30**	92.31
		0.01	96.47	94.31	88.22	29.60	86.85	92.16
	Adam	0.001	89.94	88.96	79.93	59.16	74.58	82.44
		0.005	95.82	95.52	**94.71**	75.65	84.95	85.93
		0.01	**97.05**	95.76	94.09	50.32	78.89	**94.38**
	Adagrad	0.001	80.72	73.43	58.21	33.54	55.08	46.12
		0.005	91.59	90.68	89.27	53.17	78.55	84.84
		0.01	92.78	93.17	92.54	61.94	83.39	87.94
	Lamb	0.001	38.89	16.90	14.17	19.57	19.27	12.74
		0.005	77.12	65.15	62.86	57.07	50.16	64.27
		0.01	90.10	85.64	82.00	81.88	77.99	75.29
	Yogi	0.001	89.73	89.10	78.24	49.33	71.81	83.51
		0.005	95.36	95.47	92.92	**82.80**	88.21	92.98
		0.01	96.65	94.92	93.64	60.52	86.69	93.48
FEDADAVR-QUANT	Adabelief	0.001	93.65	91.21	85.57	70.89	67.06	84.32
		0.005	96.28	**96.25**	92.16	68.55	87.63	92.68
		0.01	96.65	95.65	93.69	52.37	**90.60**	92.99
	Adam	0.001	89.49	88.65	80.97	71.77	68.05	82.38
		0.005	95.79	96.12	**93.77**	67.32	81.98	93.72
		0.01	**96.90**	95.74	85.74	23.23	89.08	**94.13**
	Adagrad	0.001	80.63	65.57	49.60	37.27	39.02	48.22
		0.005	91.78	91.74	83.91	52.49	75.97	86.97
		0.01	94.54	93.18	90.08	69.31	81.98	78.66
	Lamb	0.001	21.75	21.45	17.39	19.58	16.89	18.87
		0.005	78.68	72.76	65.16	49.36	58.69	54.94
		0.01	88.68	85.35	85.29	**79.12**	77.51	82.17
	Yogi	0.001	91.65	89.87	74.87	45.56	74.13	84.27
		0.005	96.28	95.49	91.64	73.52	83.34	91.72
		0.01	96.02	83.08	90.47	52.47	84.42	92.48

Table 10: Accuracy comparison of FedVARP, FEDADAVR and FEDADAVR-QUANT algorithms on the FMNIST dataset (Server learning rate: η_s).

			Partition					
Algorithm	**Optimiser**	η_s	**IID**	**Non_IID**	**Dir.**	**LQ-1**	**LQ-2**	**LQ-3**
FedAvg	-	-	81.35	77.43	77.08	63.53	59.70	71.92
FedProx	-	-	81.33	76.92	75.8	48.12	59.12	69.66
SCAFFOLD	-	-	67.18	55.81	62.0	48.49	41.57	52.24
FedNova	-	-	80.10	76.46	76.23	66.23	63.57	71.15
FedAdam	-	-	10.00	10.00	10.00	10.00	10.00	10.00
FedAdagrad	-	-	80.99	77.40	77.83	62.63	71.35	73.01
FedYogi	-	-	83.68	81.89	81.50	62.14	71.91	74.56
MIFA	-	-	67.84	63.74	67.42	35.81	55.36	61.51
FedVARP	-	0.01	10.00	10.37	10.00	14.46	30.65	16.86
		0.1	60.21	60.50	63.44	44.84	56.68	58.47
		1.0	**79.86**	**77.53**	**77.15**	**59.83**	**66.60**	**72.40**
FEDADAVR	Adabelief	0.001	77.20	75.72	75.47	60.53	70.33	71.19
		0.005	83.06	80.80	82.50	69.56	73.31	77.43
		0.01	84.08	**84.06**	**83.20**	**71.97**	74.12	77.96
	Adam	0.001	75.10	74.04	83.09	63.02	68.74	66.98
		0.005	82.96	81.79	79.05	70.45	71.88	77.60
		0.01	84.13	83.31	73.93	69.60	73.89	78.18
	Adagrad	0.001	61.77	55.18	51.17	32.83	48.90	56.65
		0.005	75.53	74.30	73.28	52.83	69.27	71.13
		0.01	80.63	77.40	77.08	66.62	72.21	73.21
	Lamb	0.001	29.95	18.78	20.71	25.69	10.83	31.06
		0.005	64.06	62.50	68.60	59.87	57.88	60.63
		0.01	74.48	74.90	76.42	66.52	70.55	74.94
	Yogi	0.001	73.73	72.90	73.57	58.43	69.22	71.97
		0.005	82.27	80.92	81.84	68.52	75.17	77.55
		0.01	**84.31**	83.48	82.66	70.35	**75.262**	**78.97**
FEDADAVR-QUANT	Adabelief	0.001	76.77	75.06	75.92	61.08	70.57	69.91
		0.005	84.31	80.53	82.48	64.23	71.66	78.02
		0.01	**84.84**	82.47	**83.25**	67.75	**76.29**	79.77
	Adam	0.001	83.62	71.55	73.35	64.84	65.53	71.40
		0.005	83.62	80.22	81.66	66.37	72.87	77.43
		0.01	83.28	**83.37**	82.75	67.42	74.90	**80.08**
	Adagrad	0.001	58.68	57.48	48.56	30.27	33.81	46.89
		0.005	75.56	73.36	74.48	55.51	66.89	70.55
		0.01	80.32	78.17	77.64	64.22	71.44	75.40
	Lamb	0.001	33.00	18.69	22.77	27.41	15.89	24.67
		0.005	62.42	65.23	69.66	60.41	59.47	69.68
		0.01	74.99	72.67	76.82	66.03	70.27	73.93
	Yogi	0.001	73.88	73.90	72.35	61.02	69.32	71.14
		0.005	83.06	80.67	80.39	**69.80**	72.44	77.48
		0.01	84.50	82.26	83.01	67.61	72.14	78.20

Table 11: Accuracy comparison of FedVARP, FEDADAVR and FEDADAVR-QUANT algorithms on the CIFAR-10 dataset (Server learning rate: η_s).

			Partition					
Algorithm	**Optimiser**	η_s	**IID**	**Non_IID**	**Dirichlet**	**LQ-1**	**LQ-2**	**LQ-3**
FedAvg	-	-	68.4	65.6	35.4 ± 25.4	30.6 ± 0.4	38.6 ± 2.6	48.9 ± 1.9
FedProx	-	-	66.9	63.4	60.5 ± 0.6	29.7 ± 1.4	43.0 ± 2.5	54.2 ± 2.7
SCAFFOLD	-	-	62.8	60.9	48.89	13.09	32.35	39.64
FedNova	-	-	68.2	65.1	59.2 ± 0.0	29.7 ± 0.1	38.0 ± 3.1	51.1 ± 1.1
FedAdam	-	-	41.6	36.4	32.6 ± 0.6	10.3 ± 0.0	23.4 ± 4.3	29.8 ± 3.0
FedAdagrad	-	-	39.1	41.8	43.1 ± 0.7	32.6 ± 0.5	39.4 ± 2.4	42.7 ± 2.1
FedYogi	-	-	71.7	70.2	63.1 ± 1.2	30.5 ± 1.2	47.0 ± 1.5	56.4 ± 3.4
MIFA	-	-	36.6	34.5	35.16	27.79	27.20	28.95
FedVARP	-	0.01	33.3	32.3	26.5 ± 1.2	34.9 ± 1.4	27.6 ± 1.6	42.4 ± 9.0
		0.1	51.9	51.6	48.1 ± 0.9	$\mathbf{38.4 \pm 0.5}$	$\mathbf{41.8 \pm 0.9}$	49.8 ± 0.4
		1.0	**69.0**	**65.4**	$\mathbf{61.0 \pm 0.3}$	32.8 ± 1.3	39.7 ± 2.4	$\mathbf{53.3 \pm 0.5}$
FEDADAVR	Adabelief	0.001	70.1	**70.3**	67.0 ± 2.0	41.0 ± 1.8	58.1 ± 4.1	66.0 ± 2.5
		0.005	66.9	67.4	62.1 ± 1.8	33.8 ± 0.6	49.8 ± 3.3	59.1 ± 5.2
		0.01	61.7	55.7	60.2 ± 1.9	21.6 ± 8.9	45.4 ± 1.7	57.1 ± 3.0
	Adam	0.001	67.7	69.6	67.2 ± 1.5	40.0 ± 2.0	57.2 ± 2.9	66.4 ± 1.4
		0.005	70.9	65.6	63.3 ± 2.2	34.4 ± 2.3	51.0 ± 1.7	59.9 ± 3.7
		0.01	61.2	61.6	58.6 ± 1.7	35.3 ± 0.3	48.2 ± 0.8	56.9 ± 2.2
	Adagrad	0.001	59.8	58.6	54.4 ± 0.2	46.1 ± 3.4	51.1 ± 0.5	59.8 ± 3.1
		0.005	68.2	67.4	65.3 ± 0.1	40.7 ± 4.4	52.8 ± 3.5	59.8 ± 3.1
		0.01	54.2	60.2	58.5 ± 0.9	39.8 ± 4.1	48.2 ± 3.2	55.0 ± 2.0
	Lamb	0.001	54.7	58.5	55.5 ± 0.3	$\mathbf{48.7 \pm 2.0}$	52.8 ± 1.7	57.2 ± 2.0
		0.005	67.4	68.0	65.3 ± 3.3	41.0 ± 1.2	57.4 ± 1.5	66.9 ± 1.1
		0.01	71.2	68.2	$\mathbf{69.0 \pm 1.2}$	33.2 ± 1.0	55.5 ± 2.4	64.5 ± 1.9
	Yogi	0.001	70.1	68.1	66.2 ± 2.1	41.3 ± 4.1	$\mathbf{60.8 \pm 1.2}$	$\mathbf{67.2 \pm 1.0}$
		0.005	65.0	68.0	62.3 ± 4.3	33.9 ± 3.0	50.9 ± 4.4	61.1 ± 3.0
		0.01	**72.0**	59.2	65.2 ± 2.0	40.0 ± 3.2	52.9 ± 3.2	61.5 ± 3.7
FEDADAVR-QUANT	Adabelief	0.001	69.8	68.0	68.1 ± 1.2	42.9 ± 1.2	58.7 ± 1.4	67.0 ± 1.1
		0.005	63.3	64.4	61.4 ± 2.1	33.4 ± 0.9	48.4 ± 2.6	60.1 ± 4.3
		0.01	61.7	59.6	58.9 ± 0.8	29.6 ± 0.2	45.3 ± 3.0	59.8 ± 0.6
	Adam	0.001	68.0	**69.6**	66.7 ± 1.8	41.6 ± 0.6	59.8 ± 0.4	$\mathbf{68.0 \pm 0.4}$
		0.005	64.2	65.3	62.9 ± 2.8	34.2 ± 1.4	50.9 ± 0.3	63.6 ± 2.1
		0.01	63.2	64.1	57.2 ± 1.2	34.4 ± 0.7	48.6 ± 0.5	55.2 ± 3.6
	Adagrad	0.001	59.4	59.3	54.7 ± 0.6	46.4 ± 1.0	51.0 ± 1.6	55.4 ± 1.8
		0.005	68.2	68.3	63.9 ± 1.0	44.8 ± 2.1	55.6 ± 0.8	63.8 ± 0.9
		0.01	55.5	61.4	59.5 ± 0.2	42.1 ± 2.4	49.7 ± 0.9	56.0 ± 0.6
	Lamb	0.001	55.0	57.8	55.4 ± 0.5	$\mathbf{46.9 \pm 0.1}$	53.7 ± 0.2	58.7 ± 0.2
		0.005	69.6	67.5	66.8 ± 0.4	39.8 ± 2.0	$\mathbf{59.9 \pm 0.1}$	67.9 ± 0.0
		0.01	**73.4**	69.2	$\mathbf{69.4 \pm 0.8}$	35.3 ± 0.9	58.4 ± 0.5	65.1 ± 1.4
	Yogi	0.001	70.3	67.6	66.9 ± 1.1	44.0 ± 1.1	59.7 ± 1.1	66.8 ± 1.1
		0.005	65.7	63.4	64.3 ± 2.5	37.9 ± 0.8	56.2 ± 0.2	63.5 ± 0.1
		0.01	60.0	63.3	65.2 ± 2.0	39.2 ± 7.6	53.5 ± 5.1	61.4 ± 2.6

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction accurately state that (i) FEDADAVR decouples SAGA-style variance reduction from server-side adaptive moment preconditioning and eliminates first-order partial client participation bias under Adagrad while improving monotonically with cohort size, (ii) FEDADAVR-QUANT acts as a server-side memory compression method (preserving full-precision client-server communication) that reduces server state memory by 50% (FP16), 75% (Int8), and 87.5% (Int4) while maintaining competitive (and occasionally superior) accuracy, and (iii) both methods consistently outperform state-of-the-art baselines under extreme (1–2%) participation. These claims are supported by Theorem 4.4, Corollary 4.6, and Section 5.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Section 6 and Appendix J discuss the theoretical and practical limitations of our work in detail: (i) the formal convergence guarantee (Theorem 4.4) is established specifically for server-side Adagrad under bounded stochastic gradients along visited iterates (Assumption 4.3) and uniform client weights, whereas momentum-based variants (Adam, AdaBelief, Yogi, and Lamb) are evaluated as empirical extensions; (ii) a formal coupled-error convergence bound for FEDADAVR-QUANT under deterministic linear quantisation is left for future work (with simplified bounding intuition provided via unbiased stochastic quantisation (Tang et al., 2026)); (iii) while Int4 quantisation reduces server state memory by 87.5%, the $\mathcal{O}(Nd_{\text{quant}})$ footprint remains a hardware bottleneck at extreme foundation-model scales ($d \geq 10^8, N = 10^6$, requiring $\sim$ 25 TB); and (iv) cached historical client states can become stale under rapid temporal distribution shifts (concept drift).

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be

used reliably to provide closed captions for online lectures because it fails to handle technical jargon.

- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: Assumptions 4.1 (L-smoothness), 4.2 (bounded local and global variance), and 4.3 (bounded stochastic gradients), along with an explicit account of the analytical scope and where Assumption 4.3 is required, are stated in Section 4. Complete, self-contained proofs of all auxiliary lemmas (Lemmas B.5–B.8), Theorem 4.4, and Corollary 4.6 are provided in Appendix B.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Section 5.1 and Appendix E provide complete experimental details: datasets (MNIST, FMNIST, CIFAR-10, Shakespeare), model architectures (LeNet-5, ResNet-18 with group normalisation, GRU), data partitioning strategies, client participation rates, batch sizes, local epochs, hyperparameter grids, and training rounds. Full results across hyperparameter configurations are reported in Tables 8–11, and the Flower and PyTorch frameworks are cited.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may

be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: Source code with instructions to faithfully reproduce the main experimental results is included in the supplementary material zip file. All four datasets used are publicly available (MNIST, FMNIST, CIFAR-10 via standard repositories; Shakespeare via the Flower/Hugging Face repository). Sufficient algorithmic and hyperparameter details are provided in Appendix.

   Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: Section 5.1 summarises the experimental configuration, and Appendix E provides the full specification including dataset splits, model architectures, client counts and participation rates, local epochs, batch sizes, optimiser hyperparameters ($\beta_1, \beta_2, \epsilon$, weight decay), server and client learning rate grids, number of training rounds, and evaluation protocols (Table 2 and Table 3). Hyperparameters were selected via grid search as detailed in Appendix E.5.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

Justification: As detailed in Section 5.1 and Appendix E (Subsection E.7), we report mean accuracy $\pm$ standard deviation across multiple random seeds for the high-variance skewed non-IID settings (Dirichlet, LQ-1, LQ-2, and LQ-3) on our primary CIFAR-10 benchmark in Table 11. Due to the substantial computational cost of simulating, milder IID/Non-IID splits and secondary benchmarks are reported from a single representative seed averaged over the final 10% of training rounds (Table 3) to mitigate inter-round fluctuations.

Guidelines:

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

8. **Experiments compute resources**

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

Answer: [Yes]

Justification: Appendix I specifies the hardware used: primary experiments were conducted on a High-Performance Computing (HPC) cluster, whilst a dedicated workstation (13th Gen Intel Core i9-13900H CPU (2.60 GHz), 64 GB RAM, and an NVIDIA GeForce RTX 4080

Laptop GPU (12 GB VRAM)) was utilised to ensure standardised timing. Per-round wall-clock times for all algorithms are reported in Table 6, and end-to-end times to reach accuracy thresholds are reported for the most computationally demanding setting (CIFAR-10, LQ-1).

Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The paper proposes a privacy-preserving federated learning algorithm. No human subjects data are collected; all datasets are publicly available benchmarks. The work does not involve crowdsourcing, sensitive personal data, or dual-use risks. The authors have reviewed the NeurIPS Code of Ethics and confirm compliance.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [No]

    Justification: The paper does not include a dedicated broader impacts section. Positive impacts include enabling privacy-preserving collaborative learning in regulated domains (e.g., healthcare under HIPAA/GDPR, as noted in the introduction), and improved efficiency for resource-constrained edge devices through quantisation. Potential negative impacts are limited but include the risk that more efficient FL may lower barriers to deploying models on sensitive personal data; however, the algorithm itself does not introduce new privacy vulnerabilities beyond the federated learning framework it extends.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper proposes an optimisation algorithm evaluated on standard public benchmarks. No pre-trained models, scraped datasets, or generative models with misuse potential are released. No specific safeguards beyond those already inherent in the federated learning paradigm (raw data never leaves client devices) are required.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Datasets are cited with their original papers and used in accordance with the licences or usage terms stated by their sources. Fashion-MNIST is distributed under the MIT License. The Shakespeare dataset accessed via Flower is distributed under the BSD 2-Clause License. The software frameworks used, PyTorch and Flower, are open source under BSD-style and Apache 2.0 licences, respectively.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: The paper does not release a new dataset or pre-trained model. The new contribution is an algorithm (FEDADAVR and its quantised variant FEDADAVR-QUANT), fully specified in Algorithms 1–9 and Appendices C–D. Code is attached to supplementary materials.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects. All experiments use publicly available machine learning benchmark datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [N/A]

    Justification: No human subjects are involved in this research. IRB approval is not required.

    Guidelines:

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

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: LLMs are not used as part of the core methodology. The research involves federated optimisation algorithms evaluated on standard vision and NLP benchmarks. Any LLM use was limited to writing or editing assistance and does not affect the scientific content or originality of the work.

    Guidelines:

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