FLORETS: WEAVING CROSS-CLIENT LOW-RANK ADAPTERS INTO AN ALIGNED SUBSPACE FOR FEDERATED LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated fine-tuning of Large Language Models with low-rank adapters faces a fundamental challenge: updates from client adapters trained on non-IID data lie in misaligned representational bases. Conventional schemes aggregate these adapters via weighted averaging without aligning their bases, leading to destructive parameter interference. We propose a novel federated framework, FLORETS (Federated LOw-Rank adaptation with Election and Truncation for Subspace alignment) for aligning client updates within a shared basis entirely inside the low-rank adapter subspace, without ever reconstructing full-dimensional matrices. FLORETS uses a block-diagonal stacking mechanism to uncover the shared basis through a distributed Householder QR decomposition that parallelizes construction of the shared subspace amongst multiple clients. In contrast to simple weighted averaging, FLORETS aggregates the aligned updates using a coordinate-wise, consensus-driven sign election followed by rank selection. Extensive evaluations across Llama-3.2 (1B and 3B) on Dolly and GSM8K under severe Dirichlet non-IID settings demonstrate that FLORETS consistently outperforms existing federated LoRA baselines in downstream accuracy and computational overhead.

1 INTRODUCTION

Federated fine-tuning of Large Language Models (LLMs) via Low-Rank Adaptation (LoRA) (Hu et al., 2022) has emerged as the leading paradigm for training specialized edge models while preserving data privacy. In this framework, each client device k keeps base model parameters $\mathbf{W}_0 \in \mathbb{R}^{m\times n}$ frozen and locally optimizes only low-rank adapter factors ($\mathbf{B}_k \in \mathbb{R}^{m\times r_k}, \mathbf{A}_k \in \mathbb{R}^{r_k\times n}$), $r_k \ll \min(m,n)$ in parallel. However, in non-IID data distributions, we observe (in Section 3.1) that client updates $\Delta\mathbf{W}_k$ drift persistently into misaligned representational basis. Consequently, when a server aggregates the corresponding LoRA factors, parameter updates destructively interfere with each another, corrupting the global representation and degrading downstream performance.

Existing federated LoRA aggregation frameworks struggle to resolve this misalignment, falling into two primary paradigms. **Averaging Methods:** FedIT (Zhang et al., 2024) compute element-wise averages directly over client factors ($\mathbf{B}_k, \mathbf{A}_k$). This operation introduces spurious cross-client interference terms that distort the aggregated representations, while strictly enforcing homogeneous adapter ranks ($r_k = r$) across all clients. A reconstruction-based averaging method, FlexLoRA (Bai et al., 2024), instead rebuilds the dense update $\Delta\mathbf{W} = \sum_k \mathbf{B}_k\mathbf{A}_k$ and re-truncates it to each client's rank, recovering the exact aggregate but at full-dimensional $\mathcal{O}(mn)$ cost. Modern variants such as FedRot-LoRA (Zhang et al., 2026b) attempts to minimize these cross-terms by rotating ($\mathbf{B}_k, \mathbf{A}_k$) separately toward a reference coordinate system, yet leave the underlying client updates, $\Delta\mathbf{W}_k = \mathbf{B}_k\mathbf{A}_k \in \mathbb{R}^{m\times n}$, unaligned followed by standard linear averaging. Attempts to align these $\Delta\mathbf{W}_k$ updates will incur prohibitive computational, memory and communication cost with large number of clients. **Stacking Methods:** To support accurate averaging of client updates without cross-term aggregation noise, recent works such as FLoRA (Wang et al., 2024), FLoRIST (Ramesh & Dass, 2026a), and SpecTraL (Ramesh & Dass, 2026b) concatenate client adapter factors into unified block structures, avoiding factor-averaging and natively supporting varying client ranks. However, these stacking schemes construct global LoRA adapters on the server without any subspace alignment of client updates. This reveals a fundamental limitation of existing federated LoRA ag-

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

gregation methods: *linear weighted averaging, whether applied directly to individual LoRA factors or to stacked LoRA matrices is mathematically incapable of repairing subspace misalignment in client updates*. We are thus driven by a fundamental research question:

How to efficiently perform subspace-aligned federated aggregation using low-rank adapters without reconstructing full client updates?

To address this challenge, we propose **FLORETS** (**F**ederated **LO**w-**R**ank adaptation with **E**lection and **T**runcation for **S**ubspace alignment). To the best of our knowledge, FLORETS is the first federated LoRA method to align cross-client updates within a shared orthogonal basis and aggregate them non-linearly while operating entirely inside the low-rank adapter subspace without ever reconstructing full client update matrices. In contrast to conventional approaches that perform factor-wise averaging or standard stacking of LoRA (stack $\mathbf{B}_k$ horizontally and stack $\mathbf{A}_k$ vertically), where individual updates sum together and vanish prior to alignment, FLORETS arranges cross-client adapter $\mathbf{A}_k$ block-diagonally (illustrated in Appendix Figure 6). This key structural modification preserves individual update identities and seamlessly handles different client ranks.

Contributions. Our main contributions are as follows:

- **Subspace misalignment, measured.** We isolate misalignment of the client *updates*, rather than of their factorizations, as the obstacle that persists once the factorization gauge is resolved, and show that client update subspaces are close to mutually orthogonal under severe non-IID splits (Sec. 3.1). A shared basis must therefore be constructed from all clients jointly, inside the low-rank adapter space and without reconstructing full-dimensional updates.
- **Distributed alignment in low-rank space.** FLORETS stacks the $\mathbf{A}_k$ block-diagonally rather than vertically, which preserves per-client update identity, generates the full-space concatenation exactly from the low-rank factors, and admits heterogeneous ranks. The same structure decouples the $\mathbf{A}$-side factorization: clients compute thin Householder QRs of their own adapters (*florets*) in parallel, leaving the server one cross-client QR and one $r \times r$ SVD. We prove that this reproduces the centralized alignment exactly.
- **Consensus merging in the aligned basis.** Because a weighted average factors a shared basis out of the sum, we replace server-side averaging with a coordinate-wise rule: a sign election discards the contributions whose sign opposes the elected direction, a closed-form norm correction restores the magnitude lost to that removal, and random-matrix denoising selects the retained rank.
- **Empirical validation.** On Llama-3.2 (1B and 3B) (Grattafiori et al., 2024) with Dolly (Databricks, 2023) and GSM8K (Cobbe et al., 2021) under Dirichlet non-IID splits, FLORETS attains the highest accuracy in seven of eight settings against seven federated LoRA baselines, and a component ablation isolates the contribution of each mechanism. Its server-side cost is the lowest among methods that recover a singular value spectrum.

2 Background and Related Work

Throughout, $[\,X \mid Y\,]$ denotes horizontal concatenation and $[\,X; Y\,]$ vertical concatenation. Appendix B gives an extended treatment of each method family and a detailed comparison.

2.1 LoRA and Federated Fine-Tuning

Low-Rank Adaptation (Hu et al., 2022) replaces a full update to a frozen pretrained weight $W_0 \in \mathbb{R}^{m\times n}$ with $\Delta W = BA$, where $B \in \mathbb{R}^{m\times r}$, $A \in \mathbb{R}^{r\times n}$, and $r \ll \min(m,n)$, reducing the trainable parameter count from mn to $r(m+n)$. In a federation of K clients, client k holds a local dataset $\mathcal{D}_k$ and trains (B_k, A_k) at a client-specific rank r_k. The server must produce a single global update $\Delta W^* = \sum_{k=1}^{K} p_k\, B_k A_k$, with data-proportional weights $p_k = |\mathcal{D}_k| / \sum_{j=1}^{K} |\mathcal{D}_j|$, forming it without corruption, representing it compactly, and broadcasting it efficiently. We write $r = \sum_{k=1}^{K} r_k$ for the total stacked rank and assume $m, n \geq r$.

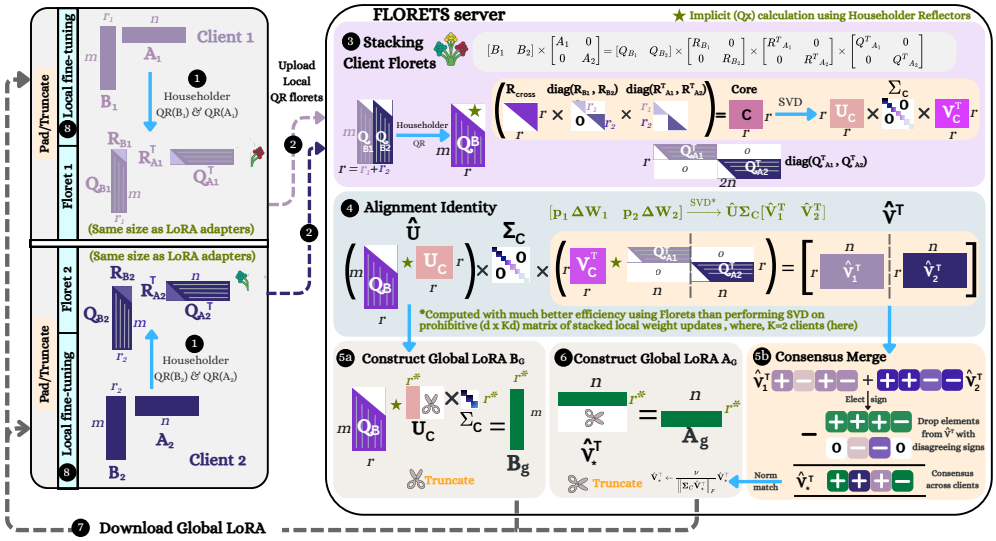


Figure 1: Overview of the FLORETS pipeline.

2.2 Three Obstacles to Aggregation

Obstacles 1 and 2: Cross-term noise and the rotational gauge. Two defects follow from aggregating B_k and A_k rather than the update ΔW_k they represent. *Cross-term noise*: averaging the factors independently, $\bar{B} = \sum_k p_k B_k$ and $\bar{A} = \sum_k p_k A_k$, gives $\bar{B}\bar{A} = \sum_k p_k^2 B_k A_k + \sum_{i \neq j} p_i p_j B_i A_j \neq \Delta W^*$, where the cross-terms $B_i A_j$ inject spurious directions that accumulate across rounds. *Rotational gauge*: the decomposition is not unique, since $(B_k R)(R^\top A_k) = B_k A_k$ for any orthogonal $R \in \mathbb{R}^{r_k \times r_k}$, so independently trained clients express their factors in arbitrarily rotated latent bases, and any operation that reads B_k or A_k separately is sensitive to a choice carrying no information. Both defects stem directly from the factorization and are resolved by aggregating the products $\Delta W_k = B_k A_k$, eliminating R and preventing cross-term formation.

Obstacle 3: Representational misalignment of the updates. Even given exact, gauge-free products, the ΔW_k share no coordinate system: each client's data drives its update along whatever directions that data selects, so the singular directions of ΔW_i and ΔW_j bear no relation to one another. Coordinate-wise aggregation therefore compares unrelated quantities until the updates are placed in a common frame. Resolving the gauge fails to align factorized updates because they address distinct aggregation targets (Sec. 2.3), an obstacle directly measured in Sec. 3.1.

2.3 Federated LoRA Aggregation

Averaging, freezing, and reconstruction. FedIT (Zhang et al., 2024) averages B and A independently and incurs the cross-term noise of Obstacle 1. FFA-LoRA (Sun et al., 2024) restores exactness by freezing A and averaging only B, and FedSA-LoRA (Guo et al., 2025) shares only A, producing a personalized scheme rather than a single global update; all three require homogeneous rank. FlexLoRA (Bai et al., 2024) reconstructs the dense $m \times n$ update and re-truncates it to each client's capacity at $O(\min(m,n)\, mn)$ cost.

Stacking. FLoRA (Wang et al., 2024) eliminates Obstacle 1 by stacking the factors, $B_{\text{stack}} = [\, B_1 \mid \cdots \mid B_K \,] \in \mathbb{R}^{m \times r}$ and $A_{\text{stack}} = [\, p_1 A_1; \cdots ; p_K A_K \,] \in \mathbb{R}^{r \times n}$, with the weights folded into one side, so that $B_{\text{stack}} A_{\text{stack}} = \Delta W^*$ exactly. Stacking is cross-term-free, but the stacked pair grows with $\sum_k r_k$. FLoRIST (Ramesh & Dass, 2026a) recovers the SVD of ΔW^* *without* forming the $m \times n$ product, composing thin SVDs of the stacked factors through a compact $r \times r$ interaction matrix. SpecTraL (Ramesh & Dass, 2026b) instead uses thin Householder QR, $B_{\text{stack}} = Q_B R_B$ and $A_{\text{stack}}^\top = Q_A R_A$, with $Q_B \in \mathbb{R}^{m \times r}$ and $Q_A \in \mathbb{R}^{n \times r}$ orthonormal-column and $R_B, R_A \in \mathbb{R}^{r \times r}$ upper triangular, and computes a single SVD of the $r \times r$ core $C = R_B R_A^\top$. Since $\Delta W^* =$

$Q_B\, C\, Q_A^\top$, the singular values of ΔW^* are exactly those of C, recovered at $O(r^3)$ cost independent of m and n. Our method builds directly on this construction. The vertical concatenation, however, sums the clients immediately: the individual ΔW_k are no longer recoverable once the product is formed, so no method built on it can address Obstacle 3, and none attempts to do so.

Alignment. A recent line of work treats alignment as an explicit design goal. FedRot-LoRA (Zhang et al., 2026b) rotates each client's factors toward a shared reference by orthogonal Procrustes before averaging them; since the rotation leaves $\Delta W_k = B_k A_k$ pointwise unchanged, its effect is confined to Obstacle 2 and the aggregation target remains the weighted average. LoRA-FAIR (Bian et al., 2025) likewise operates on the factors, correcting aggregation bias with a server-side residual rather than reconstructing the updates. Neither places the updates themselves in a shared basis. Methods that align activation subspaces or optimizer state instead of the updates (Zhang et al., 2026a; Peng et al., 2026; Mahla et al., 2025) are treated in Appendix B. The observation that a common basis is what renders coordinate-wise comparison across clients well-posed originates in model merging (Stoica et al., 2025), in a centralized, full-rank setting that we also review there.

An open question therefore remains: *how can exact shared-basis alignment and non-linear consensus merging be achieved directly from low-rank factors, without ever reconstructing full-dimensional update matrices?* We introduce FLORETS in Sec. 3 to address it.

3 FLORETS

We present FLORETS in two parts: a motivating empirical observation (Sec. 3.1) and the methodology itself (Sec. 3.2), which closes with a cost analysis. Pseudocode and the full complexity accounting are deferred to Appendix C.

3.1 Motivating Observation

Sec. 2.2 argued that the client updates $\{\Delta W_k\}$ share no coordinate system. We measure that claim directly, since its severity determines the design.

For a matrix X, let U_X^i denote its leading i left singular vectors, an orthonormal basis for the i-dimensional subspace of $\mathbb{R}^m$ that carries most of X's energy. We quantify the overlap of two such subspaces by $\phi(X, Y, i, j) = \|(U_X^i)^\top U_Y^j\|_F^2 / \min(i, j) \in [0, 1]$, the mean squared cosine of the principal angles between them and a subspace generalization of cosine similarity (Hu et al., 2022): $\phi = 1$ when the smaller subspace is contained in the larger, and $\phi = 0$ when the two are orthogonal. We evaluate ϕ on the products $\Delta W_k = B_k A_k$ rather than on the factors, since by Obstacle 2 only the products are invariant to the factorization gauge.

Observation: client updates occupy near-disjoint subspaces. Fig. 2 reports ϕ between per-client updates under severe heterogeneity. Overlap is near zero across the entire grid, including at the largest slices, where the full rank-r_k subspaces are compared and incidental intersection would be most likely. The clients are therefore not merely misaligned: their update subspaces are close to mutually orthogonal.

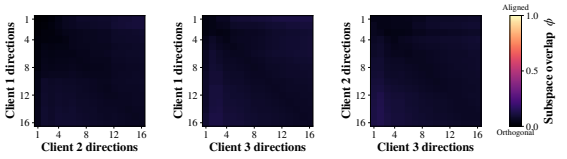


Figure 2: **Clients span near-disjoint subspaces.** Pairwise overlap $\phi(\Delta W_a, \Delta W_b, i, j)$ between per-client updates ($\alpha = 0.1$, Llama-3.2-1B / Dolly, `q_proj` layer 15, final round). Overlap is near zero across the whole grid. Full analysis in App. D.

This has two consequences for the aggregator. First, no client's basis can serve as the shared frame, since if the directions of client a are absent from the span of client b, the basis of b cannot represent the update of a, and symmetrically. A common basis must therefore be constructed from all clients jointly rather than borrowed from any one of them. Second, summation removes the structure a consensus rule requires: the vertical stack of the FLoRA family (Sec. 2) sums the clients on contact, superposing K near-disjoint subspaces into a single matrix in which no direction can be attributed to the client that produced it.

Together these requirements specify the object the aggregator needs: a representation that retains all K clients while exposing a single left factor common to them all, namely the concatenation $[\,\Delta W_1 \mid \cdots \mid \Delta W_K\,] \in \mathbb{R}^{m\times Kn}$, whose rank is $\approx \sum_k r_k = r$. Forming this matrix explicitly abandons the low-rank representation and scales with the ambient dimension m. Sec. 3.2 shows that it never has to be formed.

3.2 METHODOLOGY

FLORETS operates in five steps per communication round, illustrated in Fig. 1: (1) spectral transformation of each client's local adapters via Householder QR, (2) block-diagonal stacking of the transformed factors and distributed recovery of a compact core, (3) derivation of an alignment identity that expresses every client update in one shared basis, (4) consensus merging of the aligned factors, and (5) rank selection, construction, and broadcast of the compact global adapters.

Step 1: Local Spectral Transformation via Householder QR. Each client computes thin Householder QR factorizations (Golub & Van Loan, 2013) of its local adapters, $B_k = Q_{B_k} R_{B_k}$ and $A_k^\top = Q_{A_k} R_{A_k}$, where the transpose on the A-side is required because A_k is wide. The orthonormal factors are stored implicitly as reflectors and never formed. Client k uploads the resulting *floret* $\{Q_{B_k}, R_{B_k}, Q_{A_k}, R_{A_k}\}$, whose size matches that of the adapter it replaces. Because each factorization depends only on data that client k already holds, a single stacked decomposition is replaced by K independent ones that execute in parallel across the federation and overlap with local training, leaving the server no per-client orthogonalization to perform.

Step 2: Block-Diagonal Stacking and Distributed Recovery of the Core. The server arranges the A factors block-diagonally,

$$A_{\text{diag}} = \text{blkdiag}\left(p_1 A_1,\ p_2 A_2,\ \ldots,\ p_K A_K\right) \in \mathbb{R}^{r\times Kn}, \tag{1}$$

so that block k occupies rows $r_1+\cdots+r_{k-1}+1$ through $r_1+\cdots+r_k$ together with its own column band of width n, every off-diagonal block is zero, and the federated weights p_k are absorbed into the blocks. Paired with the horizontal B stack, block multiplication yields

$$B_{\text{stack}}\, A_{\text{diag}} \;=\; [\,p_1\Delta W_1 \mid \cdots \mid p_K \Delta W_K\,] \;\in\; \mathbb{R}^{m\times Kn}, \tag{2}$$

which is exactly the concatenation identified in Sec. 3.1, generated by the low-rank factors and never formed explicitly. The vertical stack and the block-diagonal stack differ only in the placement of the zero blocks: the former sums the clients into a single n-column block, whereas the latter places them side by side in K column bands, so that each client retains a distinct identity within one shared left factor B_{stack}.

The same block structure also decouples the decomposition, and the two sides behave differently. The A-side requires no server computation: because the blocks of A_{diag} share no rows or columns, the uploaded local factors assemble directly into a thin QR of $A_{\text{diag}}^\top$, with $Q_{A,\text{diag}} = \text{blkdiag}(Q_{A_1}, \ldots, Q_{A_K})$ and $R_{A,\text{diag}} = \text{blkdiag}(p_1 R_{A_1}, \ldots, p_K R_{A_K})$, which the server obtains by concatenation alone. The B-side requires one additional step, because the clients' column spaces overlap in $\mathbb{R}^m$: a single cross-client QR, $[\,Q_{B_1} \mid \cdots \mid Q_{B_K}\,] = Q_B R_{\text{cross}}$, resolves the overlap and yields $R_B = R_{\text{cross}}\,\text{blkdiag}(R_{B_1}, \ldots, R_{B_K})$. This is the only orthogonalization the server performs. Algorithm 1 gives the assembly step by step, and Appendix C verifies that both pairs are valid thin QRs. The server then forms the compact $r \times r$ core $C = R_B R_{A,\text{diag}}^\top$ and computes its SVD, $C = U_C \Sigma_C V_C^\top$, at $O(r^3)$ cost independent of m and n.

Proposition 1. *The server-side reconstruction of Step 2 reproduces exactly the alignment identity of equation 3 that a centralized decomposition with full access to the raw adapters would produce. No accuracy is traded for distribution* (proof in Appendix C).

Step 3: The Alignment Identity. Two objects follow from the decomposition of Step 2. The *shared consensus basis* is $\widehat{U} := Q_B U_C \in \mathbb{R}^{m\times r}$, common to all clients, and the *aligned right factor* of client k is $\widehat{V}_k^\top := (V_C^{(k)})^\top Q_{A_k}^\top \in \mathbb{R}^{r\times n}$, where $V_C^{(k)}$ denotes the k-th rank-block of rows of V_C induced by the client ranks. Because A_{diag} is block-diagonal, $Q_{A,\text{diag}}$ inherits the same structure, and block multiplication reduces the decomposition to a single identity,

$$p_k \Delta W_k \;=\; \widehat{U}\,\Sigma_C\,\widehat{V}_k^\top, \qquad k = 1, \ldots, K. \tag{3}$$

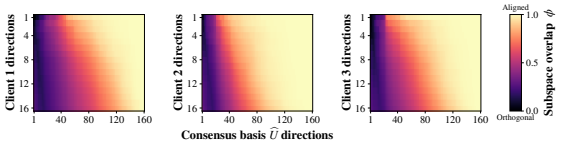


Figure 3: **The shared basis covers every client.** Containment $\phi(\Delta W_k, \widehat{U}, i, j)$ against the shared basis $\widehat{U} = Q_B U_C$; same setting as Fig. 2, with $r = 160$ (10 participating clients $\times$ rank 16). Every client is absorbed within the *leading* directions of $\widehat{U}$, those that truncation retains. Contrast Fig. 2.

Every client update is thus expressed in one shared basis $\widehat{U}\Sigma_C$, with all client-specific content confined to its own factor $\widehat{V}_k^\top$. This is the shared-basis alignment of the *updates* required by Sec. 3.1, obtained from a single $r \times r$ SVD at the server rather than from a decomposition of the $m \times Kn$ matrix. The identity places no constraint on the client ranks, since block k carries width r_k. Fig. 3 verifies the result empirically: the clients shown to be mutually near-orthogonal in Fig. 2 are each captured by the leading directions of $\widehat{U}$, the basis that Observation 1 established could not be borrowed from any single participant. Two properties of equation 3 are used in the steps that follow: the basis is identical across clients, so coordinate (i, j) of $\widehat{V}_k^\top$ denotes the same direction for every k; and the federated weight p_k is already contained in $\widehat{V}_k^\top$.

Step 4: Consensus Merging in the Shared Basis. Summing the per-client identities of equation 3 recovers the exact federated aggregate,

$$\Delta W^* = \sum_{k=1}^{K} p_k \Delta W_k = \widehat{U}\, \Sigma_C\, \widehat{V}^\top, \qquad \widehat{V}^\top := \sum_{k=1}^{K} \widehat{V}_k^\top, \tag{4}$$

so the exact aggregate is a special case of our formulation. The shared basis, however, factors back out of equation 4: a weighted sum compares no coordinate of one client against another and returns precisely what vertical stacking already delivers. A shared basis is therefore of value only under an aggregation rule that reads the shared coordinates. At each coordinate we elect the sign of the aggregate mass, $\gamma_{ij} = \operatorname{sign}\big(\sum_k (\widehat{V}_k^\top)_{ij}\big)$, and retain only the contributions consistent with it, $(\widehat{V}_\star^\top)_{ij} = \sum_{k \in \mathcal{A}_{ij}} (\widehat{V}_k^\top)_{ij}$, where $\mathcal{A}_{ij} = \{\, k : \operatorname{sign}(\widehat{V}_k^\top)_{ij} = \gamma_{ij} \,\}$ is the agreeing set. Equivalently, the conflicting contributions are subtracted from the weighted sum, $(\widehat{V}_\star^\top)_{ij} = (\widehat{V}^\top)_{ij} - \sum_{k \notin \mathcal{A}_{ij}} (\widehat{V}_k^\top)_{ij}$. Removing the conflicting terms, rather than allowing them to cancel within the average, prevents a dissenting minority from eroding a direction on which the federation agrees. The rule is mass-weighted, so a client holding a genuinely rare direction is outvoted only when opposed by larger agreeing mass; distinguishing that case from outlier noise is the intended behaviour for the latter and a genuine trade-off for the former. Excluding dissenters also removes mass, which we restore with a closed-form per-layer norm correction that introduces no tuned hyper-parameter (Appendix C).

The underlying sign-agreement principle is inspired by TIES-merging (Yadav et al., 2023), developed for combining independently fine-tuned checkpoints. Four differences follow from the federated low-rank setting. The election operates on the aligned right factors $\{\widehat{V}_k^\top\}$ within a basis recovered from the adapters rather than on full-size task vectors. It requires no per-client rescaling, since the shared Σ_C leaves every sign unchanged (Appendix C). It carries the federated weighting automatically, p_k being already absorbed into $\widehat{V}_k^\top$ by equation 1. Finally, it replaces the magnitude trim with the rank selection of Step 5.

Remark 1 (Consensus recovers the federated average under agreement). *At any coordinate where all clients agree, $\mathcal{A}_{ij} = \{1, \ldots, K\}$ and $\sum_k p_k = 1$, so the consensus returns $\sum_k (\widehat{V}_k^\top)_{ij}$, which is exactly the federated average of equation 4 at that coordinate. FLORETS departs from the weighted average only at coordinates where clients disagree.*

Step 5: Rank Selection and Construction of the Global Adapters. Because Q_B and $Q_{A,\mathrm{diag}}$ have orthonormal columns, the singular values $\{\sigma_i\}_{i=1}^r$ of the core C are exactly those of the weighted concatenation of equation 2. The spectrum that FLORETS truncates is therefore free of the cross-client interference terms that arise when the spectrum of a summed aggregate is thresholded, and it reflects genuine inter-client agreement. We select the retained rank from this spectrum by random-matrix denoising, setting $r^* = |\{i : \sigma_i > \hat{\theta}\}|$ for the MSE-optimal hard threshold

$\hat{\theta}$ of ScreeNOT (Donoho et al., 2023), which requires no tuned threshold and adapts per layer at $O(r \log r)$ cost. Unlike a magnitude trim, which discards individual entries and must be told what fraction to keep, this rule discards whole directions of the shared basis and derives its cutoff from the spectrum itself. Appendix C.7 compares it against a fixed energy threshold.

Truncating to r^* gives the global adapters in closed form, $B_g = \widehat{U}_{:,\,1:r^*}(\Sigma_C)_{1:r^*,\,1:r^*}$ and $A_g = (\widehat{V}_\star^\top)_{1:r^*,\,:}$, so that $B_g A_g$ is the consensus update expressed in the shared basis, exact at $r^* = r$ and otherwise its best rank-r^* approximation in that basis. By Remark 1 it coincides with ΔW^* wherever the clients agree, and by Fig. 3 the retained directions continue to cover every client. Both $\widehat{U} = Q_B U_C$ and the per-client Q_{A_k} inside $\widehat{V}_\star^\top$ are applied through the stored reflectors, so no orthonormal factor is materialized at any point. The server broadcasts the compact pair $(B_g, A_g) \in \mathbb{R}^{m \times r^*} \times \mathbb{R}^{r^* \times n}$, and each client pads or truncates it to its local rank r_k for the next round.

Cost. Algorithm 1 states the full round, and Appendices C.5–C.8 give the asymptotic and measured costs. The per-client QR of Step 1 is quadratic in that client's own rank and independent of the federation size: as Table 8 reports, it requires 146.5 MFLOP per round against 1.97 PFLOP for a single local epoch, and the florets add 7 KB, or 0.04%, to an upload otherwise identical to that of the stacking baselines. Distributing this factorization reduces server work rather than relocating it: a QR is quadratic in rank and $\sum_k r_k^2 = r^2/K$, so the K per-client factorizations require a factor of K fewer flops in total than the single $n \times r$ decomposition they replace, and they execute concurrently. The server is left with one $O(mr^2)$ cross-client QR, an $r \times r$ SVD, and a consensus in the compact $r \times n$ basis at $O(Krn)$, totalling 14.25 GFLOP, the lowest among methods that recover a singular value spectrum. The $m \times Kn$ concatenation is never formed, so the server's footprint remains essentially flat as the federation grows.

4 Experiments

We evaluate FLORETS against seven federated LoRA baselines on two models and two datasets spanning instruction following and mathematical reasoning, under data heterogeneity, adapter-rank variation, and the two rank-selection rules of Sec. 3.2. All experimental details omitted here for space are provided in Appendix A.

Models and datasets. We fine-tune Llama-3.2-1B and Llama-3.2-3B (Grattafiori et al., 2024), with two datasets under disjoint evaluation protocols. **Dolly** (Databricks, 2023) is an open-ended instruction-following corpus; following prior federated LoRA work (Ramesh & Dass, 2026a), Dolly-tuned models are evaluated by 5-shot **MMLU** (Hendrycks et al., 2021) macro-accuracy, which probes transferable knowledge rather than memorized instructions. **GSM8K** (Cobbe et al., 2021) is a grade-school math-reasoning benchmark; models are trained on the full chain-of-thought and evaluated by exact-match accuracy on the final answer.

Baselines. We compare against FedIT (Zhang et al., 2024), FFA-LoRA (Sun et al., 2024), FLoRA (Wang et al., 2024), FlexLoRA (Bai et al., 2024), FLoRIST (Ramesh & Dass, 2026a), SpecTraL (Ramesh & Dass, 2026b), and FedRot-LoRA (Zhang et al., 2026b). FLORETS selects rank with ScreeNOT (Sec. 3.2); Appendix C.7 compares that rule against a fixed energy threshold and motivates the choice.

Federated protocol and heterogeneity. All methods use $N = 50$ clients with 10 sampled per round, one local epoch per round, and LoRA rank $r_k = 16$ unless varied. We train for 100 rounds on GSM8K and 75 on Dolly, reporting accuracy at the final round; the round-by-round trajectory for a representative setting is shown in Sec. C.6. Each configuration is run once at a fixed seed, so every number reported here is a single-run point estimate. Clients are partitioned by a Dirichlet(α) split with $\alpha \in \{0.1, 0.5\}$, from severe to milder heterogeneity; because standard federated GSM8K treats the corpus as a single category, a Dirichlet split there induces only quantity skew, so we instead derive a genuine category label from each problem's chain-of-thought, pairing reasoning depth with the dominant arithmetic operator to obtain 18 balanced categories, so that α controls true category skew (Appendix A.3). Shared hyperparameters follow FLoRIST (Ramesh & Dass, 2026a)

			Federated LoRA Baselines (No Alignment)						With Alignment	
			Averaging			Stacking			Averaging	Stacking
α	Dataset	Scale	FedIT	FFA-LoRA	FlexLoRA	FLoRA	FLoRIST	SpecTraL	FedRot-LoRA	**FLORETS**
0.1	Dolly (MMLU)	1B	32.04	30.67	30.41	28.22	32.98	32.24	33.56	**34.36**
		3B	48.34	48.40	47.75	**50.20**	47.73	48.01	46.35	48.93
	GSM8K	1B	21.68	21.08	23.05	19.26	21.15	21.08	22.82	**24.03**
		3B	41.85	43.14	42.08	40.33	42.53	41.32	41.93	**43.82**
0.5	Dolly (MMLU)	1B	31.19	29.37	31.23	28.13	30.61	32.08	31.47	**33.27**
		3B	47.75	49.07	48.34	48.71	47.89	45.55	47.92	**50.94**
	GSM8K	1B	23.35	21.08	21.53	17.97	22.97	23.50	23.73	**24.18**
		3B	41.02	41.55	40.56	40.94	42.38	43.14	42.84	**43.75**

Table 1: Performance results at two heterogeneity levels (Dirichlet $\alpha \in \{0.1, 0.5\}$; smaller α is more severe). Dolly is evaluated by MMLU macro-accuracy (%) and GSM8K by exact-match accuracy (%); higher is better. The FLORETS column is shaded; the best value in each row is in **bold**. FLORETS is best in seven of the eight settings, the sole exception being Dolly-3B at $\alpha = 0.1$, where FLoRA's vertical stack leads.

and method-specific ones their originating papers (Appendix A.5); experiments run on the Delta AI cluster (National Center for Supercomputing Applications, 2024).

4.1 Experimental Results

Performance Results. Table 1 reports accuracy at two heterogeneity levels. FLORETS is the best method in seven of the eight settings, spanning both datasets, both model scales, and both heterogeneity levels rather than being confined to one regime. Its margin over the strongest baseline ranges from 0.5 to 1.9 points: for example $+0.8$ on Dolly-1B (34.36 vs. 33.56) and $+0.7$ on GSM8K-3B (43.82 vs. 43.14) at $\alpha = 0.1$, and $+1.9$ on Dolly-3B (50.94 vs. 49.07) at $\alpha = 0.5$. No baseline is a consistent runner-up, FedRot-LoRA, FLoRA, FlexLoRA, SpecTraL, and FFA-LoRA each lead the non-FLORETS field in some setting and trail in others, whereas FLORETS leads across the great majority. The one exception is Dolly-3B at $\alpha = 0.1$, where FLoRA's vertical stack reaches 50.20 against FLORETS's 48.93. FLoRA performs no compression, broadcasting the unprocessed stack at the full rank $r = \sum_k r_k$: at the reference configuration of Appendix C.8 this costs 175.0 MB per client per round against FLORETS's 18.3 MB, and leaves each client a rank-r merge, 770× more expensive than the factorization FLORETS requires (Table 8). Figure 5(Appendix C.6) traces accuracy over communication rounds on GSM8K / Llama-3.2-1B.

Effect of Client Participation. Figure 4 varies the number of clients sampled per round, $K \in \{10, 20\}$, at fixed federation size $N = 50$ on GSM8K / Llama-3.2-3B. Raising K forces more mutually misaligned client updates into each aggregation step, and accuracy falls for every method. FLORETS remains the best at both levels, and its lead over the strongest baseline *widens* as participation grows, from $+0.7$ at $K = 10$ (43.8 vs. FFA-LoRA 43.1) to $+1.2$ at $K = 20$ (42.2 vs. FLoRIST 41.0). The gap opens because the averaging baselines degrade most sharply under heavier aggregation, FFA-LoRA falling $43.1 \rightarrow 39.1$ and FedIT $41.9 \rightarrow 39.6$, while FLORETS drops only $43.8 \rightarrow 42.2$. Reconciling more heterogeneous updates in a single round is exactly the regime in which placing them in a shared frame before combining pays off, so FLORETS's advantage is largest where the aggregation task is hardest.

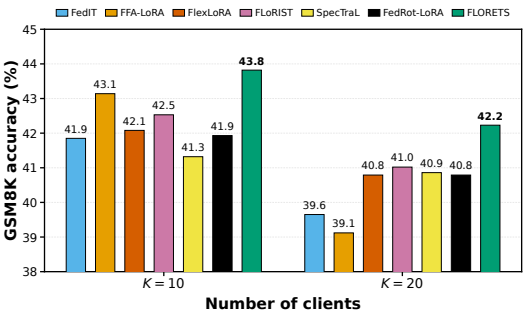


Figure 4: Performance versus number of clients selected per round on GSM8K / Llama-3.2-3B ($N = 50$, $\alpha = 0.1$). lead over the other baselines widens as K grows (10 to 20).

Effect of Rank. Table 2 varies the adapter rank on GSM8K / Llama-3.2-3B over $r \in \{8, 16, 32\}$. At $r = 8$ and $r = 16$ the field is tightly clustered and FLORETS leads only narrowly: at low rank each client's update is dominated by high-energy, task-general directions that the clients broadly share, and which any aggregator combines with little interference. Raising the rank adds no more of that shared signal. The directions it appends are the low-energy tail that a lower rank discards as noise (Hu et al., 2022), and across independently trained clients these are near-orthogonal, so the capacity gained is precisely the capacity on which the clients disagree.

Method	$r = 8$	$r = 16$	$r = 32$
FedIT	42.00	41.85	42.61
FFA-LoRA	41.47	43.14	43.37
FlexLoRA	42.68	42.08	42.30
FLoRIST	42.15	42.53	41.32
SpecTraL	41.32	41.32	42.38
FedRot-LoRA	41.39	41.93	43.52
FLORETS	**43.75**	**43.82**	**46.78**

Table 2: Adapter rank on GSM8K / Llama-3.2-3B (exact-match %, $\alpha = 0.1$). Best per column **bold**.

At $r = 32$ the field separates along this line. FedIT and FlexLoRA remain flat across the sweep; FLoRIST, which shares FLORETS's aggregation but neither aligns nor merges non-linearly, regresses ($42.53 \rightarrow 41.32$) as the tail enters its aggregate as interference rather than signal; FFA-LoRA and FedRot-LoRA improve but plateau in the 43s. FLORETS alone converts the added rank into accuracy, rising from 43.82 to 46.78, and its margin over the better of FLoRIST and SpecTraL widens from 1.6 points at $r = 8$ to 4.4 at $r = 32$. That the two methods identical to ours in aggregation, and lacking only alignment and consensus, gain least where the misalignment to resolve is greatest isolates those two mechanisms as the source of the gain.

Ablation. FLORETS changes four things relative to a stacking aggregator: shared-frame alignment, consensus sign election, the norm correction that election requires, and spectral rank selection. Table 3 removes each of the last three in turn, holding the rest fixed. Alignment has no row because switching it off changes the construction itself, an unaligned FLORETS *is* a stacking aggregator; Table 1 reads its effect instead, through FLoRIST and SpecTraL, which performs vertical stacking of A, which neither aligns nor merges nonlinearly.

Variant	**MMLU**	Δ
FLORETS	34.36	ref.
– Consensus merge	29.79	−4.57
– Norm matching	19.45	−14.91
– Rank truncation	31.65	−2.71

Table 3: **Component ablation of FLORETS.** Each row removes one component of FLORETS and holds the rest fixed, on Dolly / Llama-3.2-1B at $\alpha = 0.1$ (5-shot MMLU macro-accuracy, %).

Replacing the election with the weighted sum of equation 4 costs 4.57 points ($34.36 \rightarrow 29.79$), confirming the algebra of Sec. 3.2: a linear rule reads no coordinate of one client against another, so the frame it is handed does no work. Broadcasting at the full stacked rank $r = 160$ rather than the selected r^* costs 2.71 ($34.36 \rightarrow 31.65$), the untruncated tail entering the adapter as noise. The largest single effect is norm matching, at 14.91 points ($34.36 \rightarrow 19.45$), and its size follows from what the election does: discarding every contribution that opposes the elected sign removes mass at each contested coordinate, and under severe heterogeneity most coordinates are contested, so the surviving factor is systematically under-scaled. The correction is closed-form and untuned, but a consensus rule that removes conflict must restore the scale that removal costs.

5 Conclusion

We introduce FLORETS, a federated LoRA framework that resolves subspace misalignment under non-IID data entirely inside the low-rank adapter space. By combining block-diagonal stacking with a distributed Householder QR decomposition (Proposition 1), FLORETS extracts an exact shared basis to enable non-linear sign consensus merging and interference-free rank truncation, overcoming the fundamental limits of traditional linear averaging. Benchmarked across two model scales and tasks under severe Dirichlet heterogeneity, FLORETS consistently achieves state-of-the-art accuracy, with performance margins widening alongside data imbalance and adapter rank. Future work will focus on fully decentralizing the server-side cross-client QR step and expanding to larger foundation model architectures.

AI USE STATEMENT

In this work, we used generative AI tools only for recommended-disclosure tasks: editing the manuscript to improve readability, formatting LaTeX tables and figures, and suggesting keywords. We did not use generative AI tools for any required-disclosure task; the method, theory, experiments, results, and their interpretation are the authors' own work. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

FLORETS is a server-side aggregation rule for federated LoRA fine-tuning; it introduces no new data collection and is evaluated only on public benchmarks (Dolly, GSM8K, MMLU) and released model weights (Llama-3.2), with no human subjects involved. Two points nonetheless warrant note. **Privacy:** FLORETS targets the federated setting, where raw data stays on-device, and each client uploads only the QR factors of the adapter it already trains, no activations or auxiliary statistics beyond what standard federated LoRA transmits. It provides no formal guarantee, however: as with any method that communicates model updates, the factors can in principle leak local information, so FLORETS should be composed with differential privacy or secure aggregation where formal guarantees are required. **Fairness:** the consensus merge elects a per-coordinate sign from the aggregate mass and discards opposing contributions (Sec. 3.2); this removes cross-client interference but, like any majority rule, can outvote a client holding a rare but correct direction, so deployments in which minority clients represent underserved groups should monitor per-group rather than aggregate performance. Finally, FLORETS reduces the compute and communication of federated LLM fine-tuning and neither creates nor amplifies capabilities beyond existing federated LoRA methods.

REPRODUCIBILITY STATEMENT

The method is specified in full in the main text and appendix. Sec. 3.2 gives the construction step by step and Algorithm 1 the complete per-round procedure; the assumptions and a full proof of Proposition 1 (exactness of the distributed assembly) are in Appendix C, and the complexity and communication analysis, with worked reference-configuration costs, in Appendices C.5 and C.8. All experiments use publicly available datasets (Dolly, GSM8K, MMLU) and released model weights (Llama-3.2); the full protocol, the federated configuration, the two rank-selection rules, and both shared and method-specific hyperparameters, is reported in Sec. 4 and Appendices A and A.5, and the construction of the category-level non-IID split for GSM8K in Appendix A.3. Each configuration is run once at a fixed seed, so reported numbers are single-run point estimates. We are happy to release the source code reproducing all results in the camera-ready version upon acceptance.

REFERENCES

Jiamu Bai, Daoyuan Chen, Bingchen Qian, Liuyi Yao, and Yaliang Li. Federated fine-tuning of large language models under heterogeneous tasks and client resources. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Jieming Bian, Lei Wang, Letian Zhang, and Jie Xu. LoRA-FAIR: Federated loRA fine-tuning with aggregation and initialization refinement. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

Yae Jee Cho, Luyang Liu, Zheng Xu, Aldi Fahrezi, and Gauri Joshi. Heterogeneous loRA for federated fine-tuning of on-device foundation models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Databricks. Free Dolly: Introducing the world's first truly open instruction-tuned LLM. https://www.databricks.com/blog/2023/04/12/

dolly-first-open-commercially-viable-instruction-tuned-llm, 2023. databricks-dolly-15k dataset.

David Donoho, Matan Gavish, and Elad Romanov. ScreeNOT: Exact MSE-optimal singular value thresholding in correlated noise. *Annals of Statistics*, 51(1):122–148, 2023.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 4th edition, 2013.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. Llama-3.2-1B and Llama-3.2-3B model cards; Meta AI.

Pengxin Guo, Shuang Zeng, Yanran Wang, Huijie Fan, Feifei Wang, and Liangqiong Qu. Selective aggregation for low-rank adaptation in federated learning. In *International Conference on Learning Representations (ICLR)*, 2025.

Zhengbao He, Ruiqi Ding, Zhehao Huang, Ruikai Yang, Tao Li, and Xiaolin Huang. Compress then merge: From multiple loRAs into one low-rank adapter. In *International Conference on Machine Learning (ICML)*, 2026.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Weirui Kuang, Bingchen Qian, Zitao Li, Daoyuan Chen, Dawei Gao, Xuchen Pan, Yuexiang Xie, Yaliang Li, Bolin Ding, and Jingren Zhou. FederatedScope-LLM: A comprehensive package for fine-tuning large language models in federated learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 5260–5271, 2024.

Navyansh Mahla, Kshitij Sharad Jadhav, and Ganesh Ramakrishnan. Exploring gradient subspaces: Addressing and overcoming LoRA's limitations in federated fine-tuning of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Daniel Marczak, Simone Magistri, Sebastian Cygert, Bartłomiej Twardowski, Andrew D. Bagdanov, and Joost van de Weijer. No task left behind: Isotropic model merging with common and task-specific subspaces. In *International Conference on Machine Learning (ICML)*, 2025.

National Center for Supercomputing Applications. DeltaAI: NCSA advanced computing resource. https://www.ncsa.illinois.edu/research/project-highlights/deltaai/, 2024. Accessed via NSF ACCESS allocation.

Aniello Panariello, Daniel Marczak, Simone Magistri, Angelo Porrello, Bartłomiej Twardowski, Andrew D. Bagdanov, Simone Calderara, and Joost van de Weijer. Accurate and efficient low-rank model merging in core space. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Hongyi Peng, Han Yu, Xiaoxiao Li, and Qiang Yang. Rethinking lora for data heterogeneous federated learning: Subspace and state alignment, 2026. URL https://arxiv.org/abs/2602.01746.

Hariharan Ramesh and Jyotikrishna Dass. FLoRIST: Singular value thresholding for efficient and accurate federated fine-tuning of large language models. In *Ninth Conference on Machine Learning and Systems*, 2026a. URL https://openreview.net/forum?id=GTZRs756YJ.

Hariharan Ramesh and Jyotikrishna Dass. Spectral transformation for layer-wise global rank discovery in federated lora for vision transformers, 2026b. URL https://arxiv.org/abs/2607.21074.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with SVD to tie the KnOTS. In *International Conference on Learning Representations (ICLR)*, 2025.

Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving loRA in privacy-preserving federated learning. In *International Conference on Learning Representations (ICLR)*, 2024.

Ziyao Wang, Zheyu Shen, Yexiao He, Guoheng Sun, Hongyi Wang, Lingjuan Lyu, and Ang Li. FLoRA: Federated fine-tuning large language models with heterogeneous low-rank adaptations. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 22513–22533, 2024.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=xtaX3WyCj1.

Haobo Zhang, Jiankun Wang, Suraj Rajendran, Weishen Pan, Lam Tsoi, Yong Chen, Fei Wang, and Jiayu Zhou. Dysco: Dynamic subspace boosting to mitigate LoRA interference in federated learning. *arXiv preprint arXiv:2607.14367*, 2026a.

Haoran Zhang, Dongjun Kim, Seohyeon Cha, and Haris Vikalo. Fedrot-lora: Mitigating rotational misalignment in federated lora, 2026b. URL https://arxiv.org/abs/2602.23638.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Guoyin Wang, and Yiran Chen. Towards building the federated GPT: Federated instruction tuning. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6915–6919, 2024.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. GaLore: Memory-efficient LLM training by gradient low-rank projection. In *International Conference on Machine Learning*, 2024.

A Detailed Experimental Setup

This appendix expands Sec. 4 with the full configuration needed to reproduce our results: models, datasets and evaluation, the non-IID construction, the federated protocol, per-method hyperparameters, and hardware.

A.1 Models

We use two instruction-tunable decoder-only language models from the Llama-3.2 family (Grattafiori et al., 2024): **Llama-3.2-1B** and **Llama-3.2-3B**. LoRA adapters (Hu et al., 2022) are attached to the query and value projections (W_Q, W_V) of every attention block; the pretrained weights remain frozen throughout, and only the adapter factors (B_k, A_k) are trained and communicated. Unless a rank ablation states otherwise, every adapter uses rank $r_k = 16$ with the LoRA scaling and dropout defaults inherited from FLoRIST (Ramesh & Dass, 2026a).

A.2 Datasets and Evaluation

Dolly / MMLU. Databricks-Dolly-15k (Databricks, 2023) is an open-ended instruction-following dataset of roughly 15k human-authored prompt–response pairs across categories such as brainstorming, classification, closed and open question answering, generation, information extraction, and summarization. Because Dolly is a free-form generation corpus with no held-out label to score directly, we follow the evaluation protocol of prior federated LoRA work (Ramesh & Dass, 2026a) and assess the Dolly-tuned model on **MMLU** (Hendrycks et al., 2021), a 57-subject multiple-choice benchmark of world knowledge and reasoning. We report macro-accuracy over subjects in the 5-shot setting, decoding up to 32 new tokens and parsing the selected option. MMLU measures whether federated instruction tuning yields transferable competence rather than surface imitation of Dolly's style.

Scheme	Categories	Cats/client ($\alpha = 0.1$)	Dominant share
`depth_op` (default)	18	6.7	0.55
`depth`	4	2.0	0.89
`operation`	5	2.4	0.86
`none` (literature)	1	1.0	1.00
IID reference	18	16.7	0.17

Table 4: GSM8K non-IID labeling schemes. `none` reproduces the standard quantity-skew-only baseline of prior work (Kuang et al., 2024); `depth_op` is our default and induces genuine category skew. "Dominant share" is the fraction of data in the largest category.

GSM8K. **GSM8K** (Cobbe et al., 2021) contains $7{,}473$ training and $1{,}319$ test grade-school math word problems, each with a natural-language chain-of-thought terminating in a delimited final answer. We train on the full chain-of-thought so the model learns to emit the delimited answer, and at evaluation we greedy-decode up to 256 new tokens and take the number following the answer delimiter, falling back to the last number in the generation when the delimiter is absent (standard practice, so that correct reasoning is not penalized for a missing marker). We normalize surface forms before comparison (stripping thousands separators and trailing zeros, e.g. $1{,}234 \to 1234$ and $72.0 \to 72$) and report exact-match accuracy on the final answer. We verified that gold-answer extraction succeeds on 100% of the $7{,}473$ training targets.

A.3 Non-IID Construction

We partition each dataset across clients with a symmetric Dirichlet(α) distribution over categories, sweeping $\alpha \in \{0.1, 0.5\}$ so that smaller α concentrates each client on fewer categories (severe heterogeneity) and larger α approaches a more uniform split.

For Dolly, the eight authored task categories serve directly as the Dirichlet support. GSM8K, however, ships as a single undifferentiated pool, so the standard federated recipe (Kuang et al., 2024) treats the whole dataset as one pseudo-category and a Dirichlet split then induces only *quantity* skew, never *category* skew. Since FLORETS targets heterogeneity-induced misalignment, quantity skew alone understates the regime of interest. We therefore derive a genuine category label from each problem's chain-of-thought, combining *reasoning depth* (the number of calculator-annotated arithmetic steps in the solution) with the *dominant operator* (multiplication, addition, division, or subtraction). The default labeling (`depth_op`) yields 18 well-balanced categories in which no single category exceeds $\sim 15\%$ of the data, giving strong category skew without a dominant class. Under $\alpha = 0.1$ this places on average ~ 6.7 categories per client, against ~ 16.7 for an IID reference. Table 4 summarizes the labeling schemes.

The `none` scheme reproduces the literature's quantity-skew-only baseline, letting us ablate genuine category skew against pure quantity skew.

A.4 Federated Protocol

Every configuration uses $N = 50$ clients with a client-participation fraction of 0.2, so 10 clients are sampled uniformly at random each round. Each sampled client performs one local epoch of training per round with AdamW (learning rate 1×10^{-4}, weight decay 0.01), batch size 8, and maximum sequence length 512. We train for 100 communication rounds on GSM8K and 75 on Dolly. Accuracy is reported at the final round; the round-by-round trajectory for a representative setting is shown in the convergence analysis (Sec. C.6). Data-proportional aggregation weights $p_k = |\mathcal{D}_k| / \sum_j |\mathcal{D}_j|$ are used by every method that accepts weights. Each configuration is run once with a fixed random seed of 42; reported numbers are therefore single-run point estimates.

A.5 Baselines and Hyperparameters

We compare against seven federated LoRA aggregators grouped by the two families of Sec. 2. The stacking family: **FedIT** (Zhang et al., 2024) (independent FedAvg of B and A; homogeneous rank only, requiring zero-padding via HetLoRA (Cho et al., 2024) for the heterogeneous setting, hence omitted there), **FFA-LoRA** (Sun et al., 2024) (freeze A, average B; homogeneous only), **FLoRA** (Wang et al., 2024) (exact stacking), **FlexLoRA** (Bai et al., 2024) (reconstruct-

and-redistribute via SVD), **FLoRIST** (Ramesh & Dass, 2026a) (stacked thin SVDs with energy thresholding), and **SpecTraL** (Ramesh & Dass, 2026b) (stacked QR with a compact core SVD and ScreeNOT rank selection). The alignment family: **FedRot-LoRA** (Zhang et al., 2026b), which each round rotates client factors toward a shared reference via orthogonal Procrustes before averaging; because its rotation must be square it is restricted to homogeneous rank.

Shared optimization hyperparameters (optimizer, learning rate, batch size, sequence length, local epochs) are held identical across all methods and follow FLoRIST (Ramesh & Dass, 2026a). Method-specific hyperparameters follow each method's originating paper. For FedRot-LoRA we use the alignment configuration of its released setup, alternating the aligned factor between rounds with alignment strength $\lambda = 1.0$ and applying rotation only after the first round to avoid an ill-conditioned rotation at the zero initialization (Zhang et al., 2026b). For our method, FLORETS-τ uses the energy threshold $\tau = 0.9$ (the robust default established by FLoRIST (Ramesh & Dass, 2026a)), and FLORETS-SN uses ScreeNOT (Donoho et al., 2023) with its MSE-optimal threshold and requires no manual tuning; the consensus merge keeps, at each coordinate, only the client contributions agreeing with the elected sign, and applies norm matching as described in sign election (Step 4), with no magnitude trim.

A.6 Hardware

All experiments were conducted on the Delta AI cluster (National Center for Supercomputing Applications, 2024), using the H100 GPUs. Each run trains on a single GPU with clients simulated sequentially per round.

B Extended Background and Related Work

This appendix expands Sec. 2. We state the requirements a federated LoRA aggregator should satisfy and compare the literature against them (Sec. B.1), then treat in detail the two families whose relationship to FLORETS is most easily mistaken: methods that align in the federated setting without aligning the updates (Sec. B.3), and methods that respond to federated LoRA's difficulties by abandoning adapters altogether (Sec. B.4).

B.1 Requirements

An ideal federated LoRA aggregator should satisfy:

- **(R1) Exactness.** Cross-term-free aggregation, injecting no spurious $B_i A_j$ directions.
- **(R2) Heterogeneous rank.** Native support for client-specific ranks r_k, without zero-padding or truncation to a common width.
- **(R3) Compactness.** A global adapter whose size does not grow with the number of clients or with the number of communication rounds.
- **(R4) Tractability.** Server-side cost that does not scale with the ambient dimensions m, n beyond a single pass over the uploaded factors.
- **(R5) Update alignment.** Shared-basis alignment of the client *updates* $\{\Delta W_k\}$ (Obstacle 3), not merely resolution of the factorization gauge (Obstacle 2).
- **(R6) An informed merge.** An aggregation rule that the shared frame actually informs, since a weighted sum in an aligned frame returns the unaligned aggregate.

Table 5 places the literature against these. Two patterns are visible. Every federated method satisfying R5 and R6 is absent, and every method satisfying R5 and R6 is centralized. The federated column and the alignment columns have never been checked together.

B.2 Federated LoRA Aggregation, Extended

Averaging, freezing, and reconstruction. FedIT (Zhang et al., 2024) integrates LoRA with FedAvg by averaging B and A independently, incurring the cross-term noise of Obstacle 1, and sup-

Method	Fed.	R1	R2	R3	R4	R5	R6
FedIT (Zhang et al., 2024)	✓	×	×	✓	✓	×	×
FFA-LoRA (Sun et al., 2024)	✓	✓	×	✓	✓	×	×
FedSA-LoRA (Guo et al., 2025)	✓	✓	×	✓	✓	×	×
FlexLoRA (Bai et al., 2024)	✓	✓	✓	✓	×	×	×
LoRA-FAIR (Bian et al., 2025)	✓	∼	×	✓	×	×	×
FedRot-LoRA (Zhang et al., 2026b)	✓	✓	×	✓	✓	×	×
Dysco (Zhang et al., 2026a)	✓	✓	×	×	✓	×	×
FedGaLore (Peng et al., 2026)	✓	✓	×	×	✓	×	×
FedFTG (Mahla et al., 2025)	✓	✓	×	×	✓	×	×
FLoRA (Wang et al., 2024)	✓	✓	✓	×	✓	×	×
FLoRIST (Ramesh & Dass, 2026a)	✓	✓	✓	✓	✓	×	×
SpecTraL (Ramesh & Dass, 2026b)	✓	✓	✓	✓	✓	×	×
KnOTS (Stoica et al., 2025)	×	✓	×	×	×	✓	✓
Core Space (Panariello et al., 2025)	×	✓	✓	✓	✓	✓	✓
CtM (He et al., 2026)	×	✓	✓	✓	✓	✓	✓
FLORETS (ours)	✓	✓	✓	✓	✓	✓	✓

Table 5: Federated LoRA aggregation and model merging against the requirements of Sec. B.1. "Fed." marks methods that operate as a federated aggregator rather than a one-shot centralized merge. $\sim$ denotes partial satisfaction. R3 fails for Dysco because the adapter accumulates one rank-r block per communication round, and for FedGaLore and FedFTG because they transmit second-moment state and full weight blocks respectively.

ports only homogeneous ranks; heterogeneity requires zero-padding the narrower clients, as in HetLoRA (Cho et al., 2024). FFA-LoRA (Sun et al., 2024) restores exact aggregation by freezing A at its initialization so that the aggregate is $\big(\sum_k p_k B_k\big) A_0$, at the cost of halving the trainable parameters and forgoing rank heterogeneity. FedSA-LoRA (Guo et al., 2025) keeps both factors trainable and shares only A, on the finding that A captures general transferable structure while B encodes client-specific knowledge; the result is a personalized scheme rather than a single global update. FlexLoRA (Bai et al., 2024) reconstructs the dense update and applies SVD to redistribute components to each client, but truncates to client capacity rather than to the intrinsic dimensionality of the aggregate, which is the behavior FedFTG (Mahla et al., 2025) criticizes as information-destroying (see Sec. B.4).

Stacking and its cost. Stacking is the only route to R1 and R2 simultaneously, because block k of the stack occupies its own r_k columns and never mixes with block j. The price is that the stacked pair has width $r = \sum_k r_k$, so broadcasting it unprocessed inflates download cost linearly in the number of participating clients and forces every client to re-merge a wider update into W_0. FLoRIST and SpecTraL both answer this by recovering the spectrum of ΔW^* from compact factors and truncating, differing in whether the compact object is obtained by two thin SVDs composed through an interaction matrix or by two thin QR decompositions composed into an $r \times r$ core. SpecTraL additionally selects per-layer ranks from the recovered spectrum using random-matrix theory rather than a fixed energy threshold.

Why the vertical stack forecloses alignment. The observation in Sec. 2 deserves restating precisely. In §B, vertical stack the right factor is a *vertical* concatenation, so the product $B_{\text{stack}} A_{\text{stack}}$ sums the per-client contributions immediately: the object entering the subsequent SVD or QR is ΔW^* itself, a single $m \times n$ matrix in which the individual ΔW_k are no longer recoverable. Whatever basis the decomposition produces is therefore a basis for the aggregate, not a common frame in which the clients can be compared. Alignment in the sense of Obstacle 3 is not merely absent from FLoRA, FLoRIST, and SpecTraL; it is unreachable from their stacking geometry.

B.3 The Federated Alignment Family, Extended

FedRot-LoRA. Each round, FedRot-LoRA solves an orthogonal Procrustes problem per client, alternating between aligning the A side and the B side, and averages the rotated factors. Since the

rotation is orthogonal, $\tilde{B}_k\tilde{A}_k = B_kA_k$ exactly: the update is invariant and only its factorization changes. The benefit is entirely to the bilinear average, whose cross-terms are least destructive when the clients' latent bases agree. The aggregation target is unchanged, and remains the weighted average. Two structural consequences follow. The rotation matrix is square, forcing homogeneous rank across clients, and the reference pair is the previous round's average, making the procedure iterative and dependent on round-to-round stability.

LoRA-FAIR. LoRA-FAIR aggregates the adapters and then learns a server-side correction intended to reduce the discrepancy between the factor-wise aggregate and the ideal update, trading exactness for the cost of never reconstructing ΔW^*. The design point is explicit: correction in factor space is cheaper than reconstruction, and the residual it leaves is controlled rather than eliminated. FLORETS inverts this trade. Because alignment is obtained from the stacking geometry rather than purchased with an approximation, the updates are placed in a shared frame exactly, and at lower server cost than the iterative correction LoRA-FAIR requires.

Dysco. Dysco identifies *data-parameter interference*: once adapters are merged, evaluating $W_0+B_1A_1+B_2A_2$ on an activation h_1 drawn from client 1 produces a cross-client term $B_2A_2h_1$, so client 2's adapter perturbs client 1's predictions. The term vanishes when $A_2h_1 \approx 0$, which recasts aggregation as a subspace-allocation problem over the *row spaces* of the A_k. Each client computes an activation-insensitive basis by minimizing $\|AH_k^\top\|_F^2$ subject to $AA^\top = I$, whose closed-form solution is the trailing eigenvectors of the activation Gram matrix $H_k^\top H_k$, and uploads only that basis. The server allocates to each client a merged subspace maximizing overlap with the *other* clients' insensitive directions, again in closed form, and broadcasts it back. Clients then freeze A at the allocated subspace and train only B. Because representations drift across rounds, Dysco freezes the completed pair each round and stacks a fresh block, so the adapter accumulates as $\sum_t B_k^t A_k^{\mathrm{merge},t}$.

Six differences separate this from FLORETS.

1. **Constrained optimization versus aggregation.** Dysco dictates the update subspace *before* local training and thereby changes what clients are permitted to learn. FLORETS leaves local training untouched and operates post-hoc on whatever the clients produced. The two are composable rather than competing.

2. **A different object, from a different source.** Dysco aligns the LoRA row space against client *activation* statistics and never examines the spectrum of ΔW. FLORETS aligns the reconstructed updates in a basis derived from the updates themselves, requiring no activations, no auxiliary data, and no uplink beyond the factors clients already send.

3. **The merge remains linear.** After the subspace machinery, B is aggregated by whatever federated algorithm sits underneath. There is no sign election, no conflict resolution, and no rank selection. Dysco *prevents* cross-client overlap; FLORETS *resolves* it where it occurs. These are opposite positions on whether shared structure across clients is signal to be pooled or interference to be suppressed.

4. **Frozen A and round-wise rank growth.** Freezing the right factor places Dysco structurally in the FFA-LoRA lineage, and the multi-round stacking exists to recover the expressivity that freezing costs. The consequence is an adapter whose rank grows by r every round, and whose right factors are *client-specific*, so no single global (B_g, A_g) is produced. FLORETS returns one global adapter at a rank selected from the spectrum.

5. **A regime requirement.** The allocation is effective when the ambient dimension can be partitioned into near-orthogonal client slices, that is, when $N \cdot r \lesssim d$. The authors report that at large client counts the strict partition becomes infeasible and the method degenerates toward FedAvg by construction. Our configuration uses $N = 50$ clients with partial participation.

6. **Stated limitations.** Dysco assumes a common architecture and synchronous aggregation, since the server must collect every client's basis before merging; partial participation is outside its specification. It also introduces a per-round exchange of activation-derived bases, whose privacy implications the authors leave to future work.

B.4 Beyond Adapters

A separate line responds to federated LoRA's difficulties by removing the adapters. FedFTG (Mahla et al., 2025) argues that low-rank adapters are the source of the problem: aggregation inflates the rank of the combined update, and truncating back to the fixed adapter rank discards information, so the paper recommends direct weight averaging with GaLore (Zhao et al., 2024) as the local optimizer, restricting fine-tuning to the lower MLP projections. FedGaLore (Peng et al., 2026) extends this with a high-probability convergence analysis and adds server-side synchronization of the projected second-moment optimizer state, extracting a shared component across clients before broadcasting it as the next round's initializer.

Neither is a LoRA aggregation method, and neither can be compared against FLORETS on its own terms: there are no factors (B_k, A_k) to align, no factorization gauge, and no reconstructed ΔW_k. The word alignment in this line refers to matching the space in which clients optimize to the space in which the server aggregates, and to synchronizing optimizer state; the model update itself is aggregated by unmodified FedAvg. The practical costs are correspondingly different: FedFTG communicates full weight blocks rather than low-rank factors, while FedGaLore adds a second-moment buffer to every uplink and requires all clients to reconstruct a common seeded projector, which precludes heterogeneous rank.

Their diagnosis supports our design. Two arguments from this line are worth adopting: First, FedGaLore observes that lifting client adapters to full space and summing them produces an aggregate whose rank can grow to $|\mathcal{P}|\, r$, placing it off the rank-r manifold, with the distance to that manifold given by the tail singular mass beyond rank r. This is a precise statement of why an unprocessed stacked aggregate is unsatisfactory, and it is exactly the quantity that the spectral truncation step of FLORETS controls: rather than accepting the inflated rank or truncating to a fixed adapter width, we select the retained rank from the recovered spectrum. Second, FedFTG's criticism of rank inflation lands on methods that collapse the aggregate back to the original LoRA rank, such as FlexLoRA and FFA-LoRA. It does not apply to the FLoRIST, SpecTraL, and FLORETS line, which select rank from the spectrum of the aggregate, and in fact motivates doing so.

B.5 Model Merging

The idea that a common basis is what makes coordinate-wise merge operations well-defined originates in model merging, where Obstacle 3 was first isolated. KnOTS (Stoica et al., 2025) concatenates the full-size task matrices $\Delta W_k \in \mathbb{R}^{m\times n}$ and takes a single SVD, $[\,\Delta W_1 \mid \cdots \mid \Delta W_K\,] = U\Sigma V^\top$ with $V^\top = [\,V_1^\top \mid \cdots \mid V_K^\top\,]$, thereby expressing every update in one shared left frame, $\Delta W_k = U\Sigma V_k^\top$, with all task-specific structure confined to its own right factor. Two features of this construction matter for our purposes. First, it operates on the reconstructed updates rather than on any factorization, which is what makes it an answer to Obstacle 3 rather than Obstacle 2. Second, the shared object is the *basis*, not a separation into mutually orthogonal subspaces: the $V_k^\top$ are not orthogonal to one another, and it is precisely the common frame, rather than any enforced disjointness, that makes coordinate-wise comparison across tasks meaningful. This is the distinction that separates the shared-basis line from methods that instead try to push client subspaces apart.

The cost is the concatenation itself. KnOTS decomposes an $m \times Kn$ matrix with every task adapter present simultaneously, at $O(m \cdot Kn)$ storage, and is centralized, full-rank, and applied once. Core Space (Panariello et al., 2025) avoids forming the wide matrix by projecting each adapter into a pair of shared reference bases, obtaining compact per-task core matrices; CtM (He et al., 2026) compresses adapters into a shared core before merging and therefore returns a single low-rank adapter by construction. Both reach R5 and R6 at low-rank cost, which is why their rows in Table 5 are otherwise full. What separates all three from a federated aggregator is the federated column alone: they assume simultaneous access to every adapter, apply no data-dependent weighting p_k, and merge once rather than aggregating round after round under partial participation and drifting clients.

Having supplied a frame, KnOTS defers the combination itself to a merge function applied to the aligned $\{V_k^\top\}$, and the two are not independent. This is the argument made in Sec. 3.2 from the side of what a linear rule cannot use: a weighted sum of $\Delta W_k = U\Sigma V_k^\top$ factors the shared frame straight back out, so the frame it is handed does no work. TIES-merging (Yadav et al., 2023) is the canonical rule that does use it, electing at each coordinate a sign from the aggregate mass across

Algorithm 1 FLORETS: One Communication Round

Require: Clients $\{1, \dots, K\}$ with adapters $\{(B_k, A_k)\}$ and weights $\{p_k\}$; rank rule (threshold τ or ScreeNOT bound κ)
Ensure: Global adapters (B_g, A_g) per adapted layer
// Client side (per adapted layer, in parallel)
for each client k **do**
  Local QR: $B_k = Q_{B_k} R_{B_k}$, $A_k^\top = Q_{A_k} R_{A_k}$ ▷ reflectors stored implicitly
  Upload $\{Q_{B_k}, R_{B_k}, Q_{A_k}, R_{A_k}\}$
end for
// Server side (per adapted layer)
for each adapted layer **do**
  A-side: $R_{A,\mathrm{diag}} \leftarrow \mathrm{blkdiag}(p_1 R_{A_1}, \dots, p_K R_{A_K})$ ▷ no recomputation
  B-side: $[Q_{B_1} | \cdots | Q_{B_K}] = Q_B R_{\mathrm{cross}}$; $R_B \leftarrow R_{\mathrm{cross}}\, \mathrm{blkdiag}(R_{B_1}, \dots, R_{B_K})$
  Core: $C \leftarrow R_B R_{A,\mathrm{diag}}^\top$
  SVD: $C = U_C \Sigma_C V_C^\top$, $\sigma_1 \geq \cdots \geq \sigma_r$
  Aligned factors: $\widehat{V}_k^\top \leftarrow \left(V_C^{(k)}\right)^\top Q_{A_k}^\top$, $k = 1, \dots, K$ ▷ apply A_k reflectors
  // Consensus merge (conflict removal)
  Elect signs: $\gamma \leftarrow \mathrm{sign}\left(\sum_k \widehat{V}_k^\top\right)$ ▷ mass-weighted; p_k already inside
  Keep agreers: $\widehat{V}_\star^\top \leftarrow$ sign election (Step 4) ▷ drop contributions opposing γ
  Match norm: $\widehat{V}_\star^\top \leftarrow$ equation 5
  // Truncate and reconstruct
  Rank: $r^* \leftarrow$ energy-τ **or** ScreeNOT$(\sigma_1, \dots, \sigma_r; r, \kappa)$
  Consensus basis: $\widehat{U} \leftarrow Q_B (U_C)_{:,1:r^*}$ ▷ apply B reflectors
  Reconstruct: $B_g \leftarrow \widehat{U}\, (\Sigma_C)_{1:r^*,1:r^*}$, $A_g \leftarrow \left(\widehat{V}_\star^\top\right)_{1:r^*,:}$
end for
Broadcast (B_g, A_g) per layer to all clients
// Client side
for each client k **do**
  Pad to local rank: $B_k \leftarrow [B_g \mid 0]$, $A_k \leftarrow [A_g; A_{\mathrm{new}}]$, $A_{\mathrm{new}} \sim \mathcal{N}(0, \sigma_A^2)$
  $(B_k, A_k) \leftarrow$ LocalUpdate$(W_0, B_k, A_k, \mathcal{D}_k, E)$
end for

tasks and averaging only those tasks that agree with it, discarding the dissenters rather than letting them cancel. Both the election and the averaging are coordinate-wise comparisons across tasks, and both are meaningless outside a shared frame.

The same coupling is corroborated from the opposite side by Iso-CTS (Marczak et al., 2025), which shows that the alignment between task-specific and merged singular subspaces predicts merge quality: tasks poorly covered by the merged subspace are the ones that lose accuracy. Read together, the two statements bracket the design space. Alignment without a coordinate-wise merge is inert, and a coordinate-wise merge without alignment is incoherent. FLORETS is placed to satisfy both, and Appendix D measures the alignment side of that claim directly in the federated setting.

C Method Details

This appendix collects the material deferred from Sec. 3: the proof of Proposition 1, the explicit norm-matching correction, the full per-round algorithm, and the complexity analysis.

C.1 Proof of Proposition 1 (Lossless Distribution)

We show that assembling the decomposition from the per-client factors of step 1 yields the same core C, and hence the same $\widehat{U}, \Sigma_C, V_C$ and aligned factors $\widehat{V}_k^\top$, as a centralized thin QR computed from the raw adapters.

Thin QR is unique up to the signs of the diagonal entries of R: if $M = QR = Q'R'$ are two thin QRs of the same full-column-rank matrix, then $Q' = QS$ and $R' = SR$ for a diagonal sign matrix S with $S^2 = I$ (Golub & Van Loan, 2013). Any such S cancels in the products that determine our output, as shown below.

A-side (exact by block-diagonality). $A_{\text{diag}}^\top = \text{blkdiag}\left((p_1 A_1)^\top, \ldots, (p_K A_K)^\top\right)$, and $(p_k A_k)^\top = Q_{A_k}(p_k R_{A_k})$ is exactly the (scaled) local QR uploaded by client k. Because the blocks share no rows or columns, $Q_{A,\text{diag}} = \text{blkdiag}(Q_{A_1}, \ldots, Q_{A_K})$ has orthonormal columns ($Q_{A_i}^\top Q_{A_j} = 0$ for $i \neq j$ by disjoint support, and $= I$ for $i = j$), and $R_{A,\text{diag}} = \text{blkdiag}(p_1 R_{A_1}, \ldots, p_K R_{A_K})$ is upper triangular. Hence $(Q_{A,\text{diag}}, R_{A,\text{diag}})$ is a valid thin QR of $A_{\text{diag}}^\top$, assembled from the local factors without any server-side orthogonalization.

B-side (one cross-client fusion). $B_{\text{stack}} = [\, Q_{B_1} \mid \cdots \mid Q_{B_K} \,]\, \text{blkdiag}(R_{B_1}, \ldots, R_{B_K})$ from the local QRs. The left factor $[\, Q_{B_1} \mid \cdots \mid Q_{B_K} \,]$ need not have orthonormal columns, since the clients' column spaces overlap in $\mathbb{R}^m$; one thin QR $[\, Q_{B_1} \mid \cdots \mid Q_{B_K} \,] = Q_B R_{\text{cross}}$ restores orthonormality, and $R_B = R_{\text{cross}}\, \text{blkdiag}(R_{B_1}, \ldots, R_{B_K})$ is upper triangular as a product of upper-triangular factors, so (Q_B, R_B) is a thin QR of B_{stack} identical to the centralized one up to diagonal signs.

Sign invariance of the output. The core is $C = R_B R_{A,\text{diag}}^\top$. Under a change of QR sign convention $Q_B \mapsto Q_B S_B$, $R_B \mapsto S_B R_B$ and likewise S_A on the A-side, the singular values of C and the reconstructed update $\widehat{U}\Sigma_C \widehat{V}_k^\top = Q_B C$ (A-side factors) are unchanged, because S_B, S_A are orthogonal and cancel between the Q and R factors in every product that reaches the output. The distributed route therefore produces the identical C, $\widehat{U} = Q_B U_C$, Σ_C, V_C, and $\widehat{V}_k^\top = (V_C^{(k)})^\top Q_{A_k}^\top$ as the centralized decomposition. □

C.2 The Shared Spectrum Leaves the Consensus Invariant

A merge that compares coordinates across clients must ordinarily reinstate each client's scale before the comparison is meaningful, since scale and direction live in separate factors of its decomposition. Our construction removes that step.

Because Σ_C is shared across all clients by the alignment identity of equation 3 and is positive on its support, $\text{sign}(\sigma_i x) = \text{sign}(x)$ for every $\sigma_i > 0$. Scaling row i of every $\widehat{V}_k^\top$ by the same σ_i therefore changes no entry sign, and, because the scaling is common to all k, no cross-client sum sign either: $\text{sign}\left(\sum_k \sigma_i (\widehat{V}_k^\top)_{ij}\right) = \text{sign}\left(\sum_k (\widehat{V}_k^\top)_{ij}\right)$. Both the elected sign γ_{ij} and the agreeing set $\mathcal{A}_{ij}$ are consequently unchanged by the presence or absence of Σ_C.

The consensus is therefore computed directly on $\{\widehat{V}_k^\top\}$, with Σ_C reinstated once at reconstruction, and since p_k is already folded into $\widehat{V}_k^\top$ through A_{diag}, the federated weighting carries through automatically. This is what removes the per-task rescaling step that a centralized coordinate-wise merge requires.

C.3 Norm Matching

Excluding dissenting clients at each coordinate removes mass, so the consensus factor $\widehat{V}_\star^\top$ is systematically smaller in Frobenius norm than the aggregate it replaces. We restore the scale with a single global factor per adapted layer,

$$\widehat{V}_\star^\top \;\leftarrow\; \frac{\nu}{\left\|\Sigma_C \widehat{V}_\star^\top\right\|_F}\, \widehat{V}_\star^\top, \qquad \nu \;=\; \sum_{k=1}^{K} p_k \left\|\Sigma_C \widehat{V}_k^\top\right\|_F, \tag{5}$$

where ν is the mass-weighted mean magnitude of a client update. Since $\widehat{U}$ has orthonormal columns, $\|\widehat{U}\Sigma_C \widehat{V}_\star^\top\|_F = \|\Sigma_C \widehat{V}_\star^\top\|_F$, so equation 5 matches the norm of the reconstructed global update itself and requires no additional decomposition. The correction carries no tuned hyper-parameter: the target scale is read directly off the clients.

Table 6: Per-round computational complexity. $\mathcal{T}(\cdot)$ is the per-epoch local training cost. FLORETS is the only method with a client term beyond training, and the only one whose server cost carries mr^2 rather than $(m+n)r^2$, because the A-side assembles by concatenation from the client QRs.

Method	Client	Server
Full FT	$\mathcal{O}(E \cdot \mathcal{T}_F(m, n, \lvert\mathcal{D}_k\rvert))$	$\mathcal{O}(LKmn)$
FedIT (Zhang et al., 2024)	$\mathcal{O}(E \cdot \mathcal{T}(\cdot))$	$\mathcal{O}(L(m+n)r)$
FFA-LoRA (Sun et al., 2024)	$\mathcal{O}(E \cdot \mathcal{T}(\cdot))$	$\mathcal{O}(Lmr)$
FlexLoRA (Bai et al., 2024)	$\mathcal{O}(E \cdot \mathcal{T}(\cdot)) + \mathcal{O}(Lmnr)$	$\mathcal{O}(L \min(m,n)\, mn)$
FLoRA (Wang et al., 2024)	$\mathcal{O}(E \cdot \mathcal{T}(\cdot)) + \mathcal{O}(Lmnr)$	None
FedRot-LoRA (Zhang et al., 2026b)	$\mathcal{O}(E \cdot \mathcal{T}(\cdot))$	$\mathcal{O}(L(m+n)r\bar{r}) + \mathcal{O}(LK\bar{r}^3)$
FLoRIST (Ramesh & Dass, 2026a)	$\mathcal{O}(E \cdot \mathcal{T}(\cdot))$	$\mathcal{O}(L(m+n)r^2) + \mathcal{O}(Lr^3)$
SpecTraL (Ramesh & Dass, 2026b)	$\mathcal{O}(E \cdot \mathcal{T}(\cdot))$	$\mathcal{O}(L(m+n)r^2) + \mathcal{O}(Lr^3) + \mathcal{O}\big(\sum_\ell (m+n) r\, r_\ell^*\big)$
FLORETS (ours)	$\mathcal{O}(E \cdot \mathcal{T}(\cdot)) + \mathcal{O}(L(m+n)r_k^2)$	$\mathcal{O}(Lmr^2) + \mathcal{O}(Lr^3) + \mathcal{O}\big(\sum_\ell (m+n) r\, r_\ell^*\big) + \mathcal{O}\big(\sum_\ell Kn\, r_\ell^*\big)$

C.4 Per-Round Algorithm

Algorithm 1 gives one communication round, making the client/server split explicit: the per-client QRs run in parallel on the clients, the A-side is assembled with no recomputation, the server performs a single cross-client QR, and the consensus runs entirely in the compact $r \times n$ frame.

C.5 Complexity and Communication Analysis

Notation. $m \times n$ is the adapted weight shape, L the number of adapted matrices, K the clients sampled per round, r_k client k's rank, $r = \sum_{k=1}^{K} r_k$, $\bar{r} = \max_k r_k$, and $r_\ell^* \le r$ the rank retained at layer ℓ. E is the number of local epochs and $\mathcal{T}(m, n, r_k, |\mathcal{D}_k|)$ the per-epoch local training cost. A thin Householder QR of a $q \times s$ matrix, with Q left in factored form, costs $2qs^2 - \frac{2}{3}s^3$ flops (Golub & Van Loan, 2013).

Client-side computation. FLORETS is the only method here that asks the client for work beyond training: two thin QRs of its own factors, B_k ($m \times r_k$) and $A_k^\top$ ($n \times r_k$), costing

$$2(m+n)r_k^2 - \tfrac{4}{3}r_k^3 \quad \text{flops per adapted matrix,} \tag{6}$$

with no allocation beyond the adapter itself, since `geqrf` overwrites its input in place. This is quadratic in the client's own rank and independent of the federation size, so it does not grow as clients join. For the configuration of Sec. 4 ($m = n = 2048$, $L = 32$, $r_k = 16$) it is 2.09 MFLOP per matrix and 66.9 MFLOP per round, against roughly 1.14 PFLOP for a single local epoch: a ratio below 10^{-7}. The orthogonalization is therefore not a client bottleneck in any regime where local training is affordable at all.

Distributing the A-side QR removes work rather than relocating it. The comparison that matters is not client-versus-server but total. A thin QR is *quadratic* in rank, and the block-diagonal structure of A_{diag} factors the A-side decomposition into K independent per-client QRs (Sec. 3.2). Because $\sum_k r_k^2 \ll \big(\sum_k r_k\big)^2 = r^2$, with equal ranks, $\sum_k r_k^2 = r^2/K$, the K small factorizations cost a factor of K *fewer* flops in total than the single $n \times r$ QR they replace, and, running in parallel, a factor of K^2 less wall-clock. At $K = 10$, $r_k = 16$: 102.1 MFLOP centralized against 10.5 MFLOP summed across the federation, a $9.8\times$ reduction in total work. The server correspondingly loses the $O(nr^2)$ term that FLoRIST and SpecTraL pay, and retains only the B-side cross-client QR, whose columns genuinely overlap in $\mathbb{R}^m$ and so cannot be decoupled. Distribution here is a consequence of the algebra, not a scheduling decision, and Proposition 1 establishes it is exact.

Upload overhead of the florets. Uploading a floret instead of the raw adapter costs exactly $2r_k$ additional floats per adapted matrix. A client that would have sent B_k and A_k ($r_k(m+n)$ floats) instead sends the two packed factorizations ($r_k(m+n) + 2r_k$ floats): `geqrf` leaves R in the upper triangle of the array that held the factor and the Householder vectors below it, so the triangular factors are transmitted at no cost and only the two length-r_k scalar vectors are new. The relative overhead is therefore

$$\frac{2r_k}{r_k(m+n)} = \frac{2}{m+n}, \tag{7}$$

Table 7: Per-round communication, summed over the K participating clients. FLORETS's upload matches the stacking baselines to leading order: the floret $\{Q_{B_k}, R_{B_k}, Q_{A_k}, R_{A_k}\}$ is stored in the packed `geqrf` form, in which R and the Householder vectors share the array that held the factor, so the only additional payload is the two length-r_k scalar vectors.

Method	Upload	Download
Full FT	$\mathcal{O}(LKmn)$	$\mathcal{O}(LKmn)$
FedIT (Zhang et al., 2024)	$\mathcal{O}(L(m+n)r)$	$\mathcal{O}(LK(m+n)\bar{r})$
FFA-LoRA (Sun et al., 2024)	$\mathcal{O}(Lmr)$	$\mathcal{O}(LKm\bar{r})$
FlexLoRA (Bai et al., 2024)	$\mathcal{O}(L(m+n)r)$	$\mathcal{O}(LK(m+n)\bar{r})$
FLoRA (Wang et al., 2024)	$\mathcal{O}(L(m+n)r)$	$\mathcal{O}(LK(m+n)r)$
FedRot-LoRA (Zhang et al., 2026b)	$\mathcal{O}(L(m+n)r)$	$\mathcal{O}(LK(m+n)\bar{r})$
FLoRIST (Ramesh & Dass, 2026a)	$\mathcal{O}(L(m+n)r)$	$\mathcal{O}\big(K(m+n)\sum_\ell r_\ell^*\big)$
SpecTraL (Ramesh & Dass, 2026b)	$\mathcal{O}(L(m+n)r)$	$\mathcal{O}\big(K(m+n)\sum_\ell r_\ell^*\big)$
`FLORETS` (ours)	$\mathcal{O}(L(m+n)r) + \mathcal{O}(Lr)$	$\mathcal{O}\big(K(m+n)\sum_\ell r_\ell^*\big)$

independent of the client's rank and shrinking as models widen. For the configuration of Sec. 4 this is 0.05%: 4 KB against an 8 MB adapter payload per client per round. The uplink is thus unchanged from FLoRIST and SpecTraL to leading order, while the downlink inherits their compression, carrying the truncated pair (B_g, A_g) at rank r_ℓ^* rather than the width-r stack that FLoRA broadcasts.

Server-side summary. Per adapted matrix the server performs a cross-client QR of the $m \times r$ matrix $[\,Q_{B_1} \mid \cdots \mid Q_{B_K}\,]$ at $O(mr^2)$, forms and decomposes the $r \times r$ core at $O(r^3)$, applies the stored reflectors to obtain the aligned and truncated factors at $O((m+n)r\,r^*)$, and runs the consensus over K clients in the compact frame at $O(Kn\,r^*)$. No term involves the $m \times Kn$ concatenation that full-space alignment would require, and no term scales with the ambient dimension beyond the single cross-client QR, so the server's footprint stays essentially flat as the federation grows.

C.6 Convergence

Figure 5 traces accuracy over communication rounds on GSM8K / Llama-3.2-1B. Early in training all methods rise together and remain tightly bunched: while the models are still far from convergence, the choice of aggregator has little room to matter. The methods separate only once training approaches convergence. Past roughly the mid-point, FLORETS pulls clear and sustains a visible lead for the rest of training, while the seven baselines collapse into a single low band, closely grouped and well below it. The final-round gap of Table 1 is therefore not a favorable stopping point but the endpoint of a clean separation: FLORETS and the baselines are indistinguishable until the aggregation quality begins to bind, after which alignment and consensus carry FLORETS above a field that otherwise converges to the same place.

C.7 Rank Selection: ScreeNOT vs. a Fixed Energy Threshold

The spectral truncation of Sec. 3.2 needs a rule for choosing the retained rank r^*, and two are available. A fixed energy threshold (FLORETS-τ) keeps the smallest r^* with $\sum_{i\le r^*}\sigma_i^2 / \sum_{i\le r}\sigma_i^2 \ge \tau$ at a preset τ, the rule inherited from FLoRIST (Ramesh & Dass, 2026a). The random-matrix-theory estimator ScreeNOT (FLORETS-SN) instead sets $r^* = |\{i : \sigma_i > \hat{\theta}\}|$ for the MSE-optimal hard threshold $\hat{\theta}$ (Donoho et al., 2023). We use ScreeNOT throughout the main paper; this appendix reports the comparison behind that choice.

Figure 7 compares the two rules across three settings and FLORETS-SN wins every one, by 2.8 to 3.3 points: 34.4 against 31.1 on Dolly-MMLU (1B, r=16), 43.8 against 40.8 on GSM8K (3B, r=16), and 46.8 against 44.0 on GSM8K (3B, r=32).

We attribute the gap to how alignment reshapes the spectrum being truncated. A fixed threshold such as $\tau = 0.9$ is a reasonable default for the spectrum of an unaligned aggregate, which is what FLoRIST and SpecTraL threshold. Our construction instead produces an interference-free spectrum (Sec. 3.2), from which the cross-client terms that basis misalignment produces have been removed, so its energy concentrates differently across singular directions and a threshold tuned for the contam-

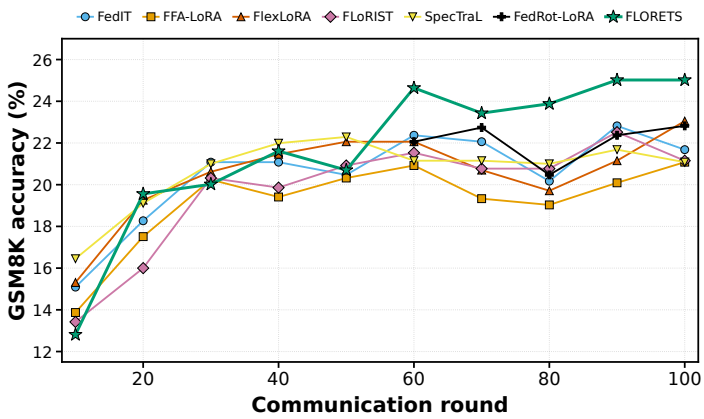


Figure 5: GSM8K accuracy over communication rounds (Llama-3.2-1B, $\alpha = 0.1$). All methods track together early in training; past convergence FLORETS separates and holds a clear lead while the baselines remain grouped below.

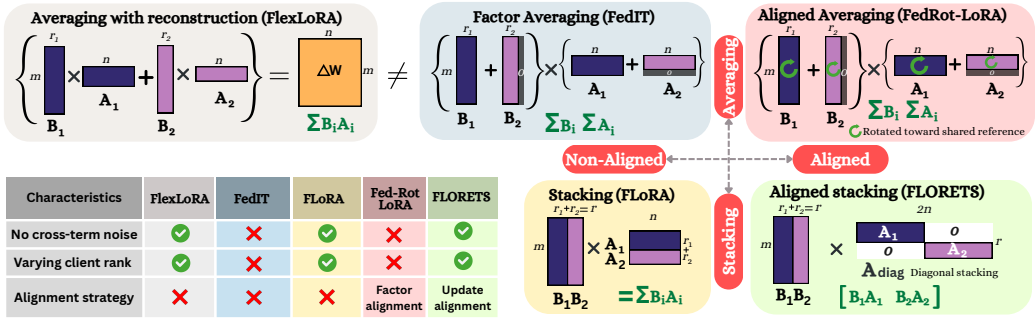


Figure 6: A taxonomy of federated LoRA aggregation, organized by *family* (averaging vs. stacking) and by whether the method *aligns*. Averaging methods reconstruct or combine the factors directly: reconstruction (FlexLoRA, ①) forms the exact aggregate $\Delta W = \sum_i B_i A_i$ but at full-dimensional cost, while factor averaging (FedIT ②) is inexact, the cross-client terms of $(\sum_i B_i)(\sum_i A_i)$ corrupt the aggregate, and admits only homogeneous ranks. Stacking methods (FLoRA, ③) recover the exact aggregate and support heterogeneous ranks, but the vertical stack sums the clients on contact, so the individual updates vanish before any alignment is possible. The aligned column adds a shared reference: FedRot-LoRA (④) rotates each client's factors toward it, but the rotation leaves $\Delta W_k = B_k A_k$ unchanged, so it aligns the *factorization* while still averaging the updates. FLORETS (⑤) is the aligned member of the stacking family: it keeps the shared B-stack and arranges the A_k factors block-diagonally, so each client remains in its own block, preserving per-client update identity in a separable, heterogeneous-rank form and exposing a shared basis that aligns the *updates* themselves. The embedded table summarizes the three properties that separate the families: only FLORETS attains cross-term-free aggregation, supports varying client rank, and update alignment together.

inated case can retain too few or too many components. ScreeNOT sidesteps the guess: it estimates the noise floor of the spectrum and keeps the components that provably rise above it, adapting per layer to the aligned spectrum's shape without a tuned parameter. This is a hypothesis consistent with the three settings shown rather than an isolated measurement of spectrum shape, but it is the reading we find most plausible, and the practical consequence is unambiguous: ScreeNOT removes a hyper-parameter and wins wherever we have compared the two.

C.8 Cost at a Reference Configuration

Table 8 evaluates the expressions of Sec. C.5 at a single fixed setting, so that the asymptotics can be read as concrete quantities. We take Llama-3.2-3B on Dolly with LoRA on the query and value

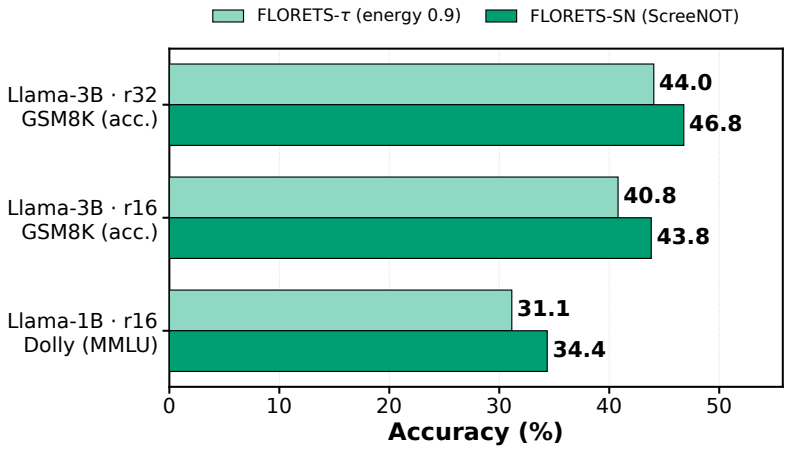


Figure 7: Rank selection in FLORETS: ScreeNOT (FLORETS-SN) against the energy threshold (FLORETS-τ) at $\alpha = 0.1$. FLORETS-SN wins in all three settings.

Table 8: Per-round cost at the reference configuration of Sec. C.8. Communication is per client per round; compute is per round. "Client" counts work *beyond* local training, which is 1.97 PFLOP per epoch and identical for every method. r^* is the mean retained rank, measured for the three rank-selecting methods and equal to the training rank otherwise. "None" means no arithmetic is performed at that stage: the methods without a client entry ask nothing of the client beyond local training, and FLoRA's server only concatenates the uploaded factors. FLORETS matches the stacking baselines on upload to within 7 KB, adds 146.5 MFLOP of client work (7×10^{-8} of one local epoch), and still runs the cheapest server of any method that recovers a spectrum.

Method	r^*	Communication (MB / client / round) Upload	Download	Computation (per round) Client (MFLOP)	Server (GFLOP)
Full FT	n/a	1344.00	1344.00	None	3.52
FedIT (Zhang et al., 2024)	16	17.50	17.50	None	0.05
FFA-LoRA (Sun et al., 2024)	16	7.00	7.00	None	0.02
FlexLoRA (Bai et al., 2024)	16	17.50	17.50	None	15265.39
FLoRA (Wang et al., 2024)	160	17.50	175.00	112743	None
FedRot-LoRA (Zhang et al., 2026b)	16	17.50	17.50	None	2.94
FLoRIST (Ramesh & Dass, 2026a)	6.9	17.50	7.55	None	111.88
SpecTraL (Ramesh & Dass, 2026b)	32.0	17.50	35.00	None	25.06
FLORETS-SN (ours)	16.7	17.51	18.27	146.5	14.25

projections of all 28 blocks, giving $L = 56$ adapted matrices of two shapes (W_Q is 3072×3072, W_V is 1024×3072 under grouped-query attention); $N = 50$ clients with $K = 10$ sampled per round, $r_k = 16$, $r = 160$, $|\mathcal{D}_k| = 300$ samples at sequence length 512, one local epoch, and `fp32` on the wire. For the three methods that select rank per layer we use the mean retained rank measured over the 75 rounds of the run: $r^* = 16.7$ for FLORETS-SN, 32.0 for SpecTraL, and 6.9 for FLoRIST. We count a thin Householder QR of a $q \times s$ matrix at $2qs^2 - \frac{2}{3}s^3$ flops, an $s \times s$ SVD with both factors at $21s^3$, the application of n_r reflectors of length q to c columns at $4qn_rc$, and a dense product at $2abc$. A local epoch is charged at $4ND$ flops, the LoRA case of the usual $6ND$: the forward and the activation gradients cost $2ND$ each, while weight gradients are formed only for the adapters. These are analytic counts, not wall-clock measurements.

Reading the table. Four comparisons carry the answer to the cost objections.

Upload. FLORETS sends 17.51 MB per client per round against 17.50 MB for every other stacking method: an overhead of 7.0 KB, or 0.039%. The florets are not a heavier payload than the adapters they replace.

Client compute. The two thin QRs cost 146.5 MFLOP per round, against 1.97 PFLOP for a single local epoch, a ratio of 7.4×10^{-8}. For contrast, FLoRA asks its clients to merge a width-r stack into

W_0 at 112.7 GFLOP per round, which is $770\times$ more client work than FLORETS requires. Moving the orthogonalization to the edge does not make the edge the bottleneck; the one method here that genuinely burdens the client does so through its *downlink*, not its uplink.

Server compute. At 14.25 GFLOP, FLORETS runs a cheaper server than every method that recovers a spectrum: $7.9\times$ under FLoRIST and $1.8\times$ under SpecTraL, whose construction is otherwise closest to ours. The saving is structural rather than incidental. SpecTraL performs two thin QRs at the server, on the $m \times r$ and $n \times r$ stacks; FLORETS performs only the first, because A_{diag} is block-diagonal and its factorization assembles from the client QRs by concatenation. At this configuration the A-side costs 8.66 GFLOP as one centralized QR but only 0.88 GFLOP summed across the federation, a $9.8\times$ reduction in total flops matching the predicted factor of K, and a $98\times$ reduction in wall-clock since the pieces run concurrently. The work is not relocated to the clients; most of it ceases to exist.

Download. FLORETS broadcasts 18.27 MB, against 175.00 MB for FLoRA, which ships the unprocessed width-r stack, and 35.00 MB for SpecTraL, whose ScreeNOT threshold retains roughly twice the rank on this run. FLoRIST is lighter still at 7.55 MB, but reaches that by truncating to a mean rank of 6.9, and Table 1 shows the accuracy that costs: the comparison to draw is between retained rank and downstream accuracy, not on bytes alone.

D Subspace Alignment Analysis

This appendix expands the motivating observation of Sec. 3.1 and the verification of Sec. 3.2. Both figures referenced here appear in the main text: Fig. 2 measures the overlap between clients, and Fig. 3 measures the coverage of the shared frame. We give the measure in full and the reading conventions behind each panel.

D.1 The Measure

For two matrices with column-orthonormal left-singular bases U_X and U_Y, let U_X^i denote the leading i columns of U_X, i.e. the top-i singular directions. The overlap between the top-i subspace of X and the top-j subspace of Y is

$$\phi(U_X, U_Y, i, j) \;=\; \frac{\left\|(U_X^i)^\top U_Y^j\right\|_F^2}{\min(i,j)} \;=\; \frac{1}{\min(i,j)} \sum_{\ell} \cos^2\theta_\ell \;\in\; [0,1], \tag{8}$$

where $\{\theta_\ell\}$ are the principal angles between the two subspaces. The singular values of $(U_X^i)^\top U_Y^j$ are exactly the cosines of those angles, so ϕ is their mean square: a subspace generalization of cosine similarity, reducing to $\cos^2\theta$ for one-dimensional subspaces. It is invariant to any rotation within either subspace, so it reads the *spans* and not the particular bases that represent them. Equivalently ϕ is the complement of the squared projection metric on the Grassmann manifold; it is a Grassmannian *similarity*, not the geodesic distance, with which it is sometimes conflated.

We compute every basis from the low-rank factors directly, without forming any $m \times n$ update: the left basis of $\Delta W_k = B_k A_k$ is obtained by composing thin decompositions of B_k and A_k, and the shared frame is the consensus basis $\widehat{U} = Q_B U_C$ of Sec. 3.2 that FLORETS already produces, read from the aggregation state rather than recomputed.

D.2 Reading Fig. 2: Clients Against Each Other

Why equal-dimension slices. The $\min(i,j)$ normalization in equation 8 gives ϕ two distinct readings. When $i \neq j$ it measures *containment*: whether the smaller subspace lies inside the larger. When $i = j$ it measures *symmetric mutual overlap*. Two peer clients do not contain one another and neither is a reference for the other, so the containment reading is not meaningful for them and we read equal-dimension slices throughout Fig. 2. The same measure is read the other way in Fig. 3, where one argument is a reference frame.

The pattern, and why it matters. Overlap is near zero across the whole grid. It remains near zero at the largest slices, where the full rank-r_k subspaces of two clients are compared and incidental intersection would be most likely, so the near-orthogonality is not an artifact of restricting the

comparison to a few directions. This is consequential because the leading singular directions carry the largest share of each update's energy and are the ones any rank truncation retains: the clients disagree not on a negligible tail but on the part of the update that any aggregator will keep.

Contrast with the within-model case. The behavior reported for LoRA (Hu et al., 2022), where the top singular directions of a rank-8 and a rank-64 adapter coincide while the tails do not, is the opposite of what we observe, and the difference is instructive. There the two adapters are trained on the same data from the same initialization, so their dominant directions are determined by a shared objective and recur across ranks. Here the clients share only the frozen initialization: each one's data selects its own dominant directions, and those directions recur in no other client. Agreement on leading directions is therefore a property of shared training, not a property of low-rank adapters, and it cannot be assumed in a federation.

D.3 Reading Fig. 3: Clients Against the Shared Frame

Containment, and what is and is not guaranteed. Here the two arguments have different dimension, so ϕ reads as containment: a bright cell means client k's top-i subspace lies inside the frame's top-j subspace. One part of the figure is guaranteed and one part is not, and the distinction is what the figure is for. The alignment identity $p_k \Delta W_k = \widehat{U} \Sigma_C \widehat{V}_k^\top$ of equation 3 places the column space of every ΔW_k inside the span of $\widehat{U}$ exactly, so $\phi \to 1$ at the right edge of each panel is a property of the construction rather than an empirical finding.

What is *not* guaranteed is *where* in the frame each client is absorbed. The columns of $\widehat{U}$ are ordered by the shared spectrum Σ_C, and nothing in equation 3 forces a given client's dominant directions to coincide with the dominant directions of that shared order. A client whose energy happened to sit in the tail of Σ_C would produce a panel that stayed dark until large j, and truncating the frame would then discard that client's contribution while leaving the others intact. This is the under-representation failure mode identified for centralized merging by Iso-CTS (Marczak et al., 2025), where tasks poorly aligned with the merged subspace are the ones that lose accuracy. The informative content of Fig. 3 is therefore the *rate* at which each panel brightens, not the saturated right edge: every client is absorbed within the leading directions of $\widehat{U}$, so the rank selection of Step 5 retains all of them and no client is systematically under-served.

Why the two figures must be read together. Taken alone, neither figure carries the argument. Fig. 2 establishes that the clients share no basis; Fig. 3 establishes that they are nonetheless all captured by $\widehat{U}$. It is the conjunction that matters: the frame holds leading structure that no individual client's basis could supply, which is why it must be constructed from the stack rather than borrowed from any single client or assumed to exist in advance.

D.4 Scope

These measurements substantiate the alignment property of the construction (R5, Sec. B.1): the updates are placed in a shared frame, and that frame covers every client at the ranks truncation retains. They do not, on their own, establish that the aggregation exploits it (R6). As Sec. 3.2 argues, a weighted sum in an aligned frame returns the unaligned aggregate, so a high-coverage frame is necessary but not sufficient; the consensus rule is what converts alignment into accuracy, and its effect is isolated by the experiments of the main text rather than by ϕ.