MODEL-AGNOSTIC PROJECTION OPTIMIZATION FOR COMMUNICATION-EFFICIENT FEDERATED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated learning enables collaborative model training across distributed clients without sharing sensitive data. However, communication overhead remains a significant bottleneck. Low-rank decomposition techniques address this by approximating each layer's weights or gradients as a product of low-rank matrices, thereby reducing communication costs in FL. While effective, these methods are typically layer-wise and constrained by the layer's architecture and shapes, limiting their flexibility and performance. We propose *Model-Agnostic Projection Optimization* (MAPO), a novel model-wide low-rank parametrization method that parameterizes the full model update using a random *reconstruction vector*, regenerated each round, and a *trainable projection vector*, thereby avoiding layer-wise decomposition and architectural constraints. MAPO directly optimizes the projection in a randomly sampled subspace, with all clients generating the reconstruction vector via a shared random seed, so the only overhead is one seed per round. By decoupling the gradient from architecture through reshaping and enabling communication-free exploration of subspaces via seed sharing, MAPO provides a flexible and efficient low-rank representation. Across seven FL benchmarks, MAPO reaches 92–99.7% of FedAvg's accuracy while using 0.13–3.1% of the cumulative uplink FedAvg needs to reach a common target accuracy, exceeding all evaluated baselines.

1 INTRODUCTION

Federated Learning (FL) is a training paradigm in which clients collaboratively optimize a model without centralizing their data. In each communication round, clients download the global model, perform local updates, and transmit the resulting model updates to a server for aggregation (FedAvg; McMahan et al., 2017). While this iterative process enables collaborative learning, frequent transmission of model updates incurs significant communication overhead, particularly with large models or resource-constrained clients.

The Communication-Efficient Federated Learning (CEFL) literature (Jia et al., 2025) proposes a wide range of strategies to reduce communication load. Prior work (Konečný et al., 2016) categorizes them into *sketched updates*, which compress the total model update after optimization (e.g., subsampling, quantization, random projection), and *structured updates*, which restrict the trainable parameters to a lower-dimensional subspace before optimization (e.g., random masks, weight-sharing, and low-rank decomposition). While sketched updates offer straightforward post-hoc solutions for communication overhead, they often have suboptimal convergence, as we discuss further in Section 2.

Low-rank decomposition is a widely used approximation technique that expresses model gradients or parameters as the product of low-rank matrices (Sainath et al., 2013; Denil et al., 2013; Jaderberg et al., 2014; Lebedev et al., 2015; Wang et al., 2018b). *Parameter decomposition* is particularly effective for parameter-efficient fine-tuning, where low-rank adaptation (LoRA) modules are added to reduce the computational and storage overhead of full-model fine-tuning (Hu et al., 2022). Although LoRA alleviates communication burdens in FL, constraining model parameters to a low-rank subspace degrades performance. In contrast, *gradient decomposition* preserves full-rank model representations during inference and restricts only the gradients to a low-rank form during backpropagation (Vogels et al., 2019; Zhao et al., 2024; Hao et al., 2024). A comparison is shown in Figure 1.

Challenges. While low-rank decomposition offers notable benefits, it faces several key challenges. In the case where both factors are trainable, we have inconsistent aggregation and coarse granularity:

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.

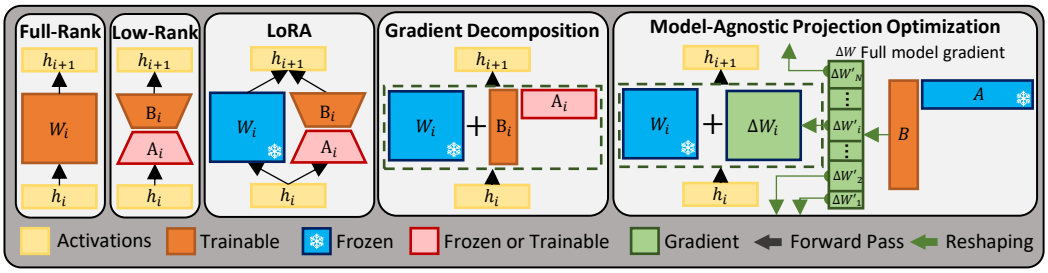


Figure 1: Comparison of various decomposition methods, from left to right: no decomposition, low-rank parameter decomposition, frozen model with low-rank adapter, low-rank gradient decomposition, and MAPO.

(i) Given M clients and (A_i^j, B_i^j) denoting the low-rank decomposition of layer i from client j, averaging low-rank matrices is *not equivalent to full-rank average* as:

$$\frac{1}{M}\sum_{j=1}^{M}(B_i^j A_i^j) \neq \Big(\frac{1}{M}\sum_{j=1}^{M} B_i^j\Big)\Big(\frac{1}{M}\sum_{j=1}^{M} A_i^j\Big)\,.$$

(ii) Given a decomposition $\Delta W_i \in \mathbb{R}^{d_1 \times d_2} \approx B_i A_i$, where $A_i \in \mathbb{R}^{r \times d_2}$ and $B_i \in \mathbb{R}^{d_1 \times r}$, the number of transmitted parameters is $\mathbf{C} = |A_i| + |B_i| = r(d_1 + d_2)$ for $r \in \mathbb{N}$, imposing a *communication granularity* as $C \in (d_1 + d_2)\mathbb{N}$, with the minimum floor of $d_1 + d_2$.

(iii) While fixing A factors and training only B can mitigate the aggregation error and improve the communication granularity to $C \in d_1\mathbb{N}$ (Rahimi et al., 2023; Sun et al., 2024), it restricts the exploration of A subspaces, leading to *suboptimal solutions* (Guo et al., 2025b).

Additionally, the layer-wise decomposition assumes that each module adapts independently, which contradicts the intrinsic dimensionality hypothesis. Model updates are known to lie in a low-dimensional subspace that spans all modules (Aghajanyan et al., 2021; Li et al., 2018). Recent work has begun to challenge this assumption. VeRA (Kopiczko et al., 2024) shares frozen random matrices across layers. VB-LoRA (Li et al., 2024) composes adapters from a vector bank. These methods show that sharing is feasible and beneficial, yet they retain a per-layer structure, which is limiting:

(iv) The layer-wise decomposition adheres to structural constraints, limiting parameter sharing to modules with the same shapes and architectures.

Given these challenges, we aim to answer the following key question: *How can we develop a model-wide decomposition that offers flexibility on parameter sharing, addresses the low-rank averaging problem and suboptimality of freezing A, and enables a fine-grained communication budget?*

Key Ideas. We propose a novel Model-Agnostic Projection Optimization (**MAPO**) that streamlines gradient projection and addresses its challenges while being computationally lighter than layer-wise methods. Our key ideas are described as follows:

(i) Firstly, MAPO reimagines gradient projection by treating the entire model gradient as a single matrix rather than decomposing it layer by layer. It eliminates architecture constraints by merging the flattened gradients across all parameter tensors to construct the *universal gradient vector* $\Delta W \in \mathbb{R}^d$.

(ii) Secondly, given any communication budget $k \in \{1, \ldots, d\}$, MAPO pads ΔW with zeros so the length becomes divisible by k. Afterwards, the padded ΔW will be reshaped to $\Delta W' \in \mathbb{R}^{k \times \lceil d/k \rceil}$ which can further be decomposed into $A \in \mathbb{R}^{1 \times \lceil d/k \rceil}$ and $B \in \mathbb{R}^{k \times 1}$ vectors, as $\Delta W' \approx BA$. Here, k is the fine-grained control over parameter sharing and the communication budget. A higher k results in a higher communication budget and less parameter sharing.

(iii) Lastly, instead of freezing or training A, MAPO regenerates it each round using a shared seed. This ensures that clients use the same A during training and aggregation, avoiding low-rank averaging error, and allows subspace exploration across rounds at the negligible cost of one seed per round.

Summary of Contributions. By integrating model-wide decomposition, fine-grained control, and subspace exploration, MAPO offers a flexible trade-off between communication and performance while remaining more efficient than layer-wise decomposition methods. Our main contributions are:

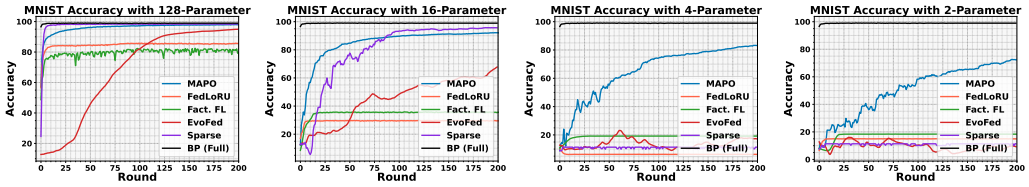


Figure 2: Baseline performance for varying numbers of trainable parameters in a centralized setting.

- We introduce model-agnostic projection optimization that enhances communication and computational efficiency, boosts performance through exploration, and offers fine-grained control.
- We provide a theoretical analysis of MAPO's convergence and computational efficiency over layer-wise factorization under the same communication and error rates.
- We conduct extensive experiments across diverse datasets, architectures, and baselines, demonstrating that MAPO surpasses existing methods in both training and fine-tuning scenarios.

2 Background and Related Work

In this section, we review key CEFL approaches in relation to MAPO. We begin with sketch-based techniques that project model updates into subspaces and outline their limitations. Then, we examine structured methods, particularly projection optimization, highlighting the opportunities and challenges that arise from operating within a fixed subspace.

2.1 Sketched update vs. Structured update

Communication-efficient FL methods can be separated by whether the communication constraint is imposed *after* or *during* local optimization. In sketched-update methods, clients first obtain a full-space local update and then compress it:

$$\Delta W^{\text{full}} \approx \arg\min_{\Delta \in \mathbb{R}^d} F_t(W^t + \Delta), \qquad \Delta W^{\text{sk}} \in \arg\min_{\Delta \in \mathcal{S}_t} \|\Delta W^{\text{full}} - \Delta\|_2^2,$$

where $\mathcal{S}_t$ is the communicated low-dimensional update family and F_t is the client's local objective at round t. Thus, sketching optimizes a reconstruction proxy for the full update, rather than the loss under the communication constraint. In contrast, structured update methods impose the constraint during training:

$$\Delta W^{\text{str}} \in \arg\min_{\Delta \in \mathcal{S}_t} F_t(W^t + \Delta).$$

For MAPO, the feasible set is induced by the vector A^t, and clients' objectives are written as:

$$\mathcal{S}_t = \left\{ \text{vec}(BA^t)_{[0:d]} : B \in \mathbb{R}^{k \times 1} \right\}, \quad B_\star^{t,j} \approx \arg\min_{B \in \mathbb{R}^{k \times 1}} \mathcal{L}^j\left(W^t + \text{vec}(BA^t)_{[0:d]}, \mathcal{D}^j\right).$$

This distinction becomes critical at low communication budgets: as $\mathcal{S}_t$ becomes small, the reconstruction proxy $\|\Delta W^{\text{full}} - \Delta\|_2^2$ can diverge from the true loss objective. Figure 2 illustrates the simplest case of centralized MNIST training, showing the performance degradation of sketched techniques, such as EvoFed and sparsification, compared to MAPO. As sparsity increases, MAPO continues to converge, even with only 2 or 4 trainable parameters out of 11,274. Nonetheless, sketching remains beneficial as a complementary step to structured updates, in which quantization or sparsification can be applied to the low-dimensional structured gradients to reduce communication overhead further.

Structured update techniques include pruning (Han et al., 2016; He et al., 2017; Luo et al., 2017; Zhang et al., 2018), weight-sharing (Chen et al., 2015; Ullrich et al., 2017), low-rank gradient (Vogels et al., 2019; Zhao et al., 2024; Park & Klabjan, 2024; Guo et al., 2025a; Zhao et al., 2025), and parameter decomposition (Yao et al., 2025; Hyeon-Woo et al., 2022; Jeong & Hwang, 2022; Hameed et al., 2024), including LoRA and its variants (Hu et al., 2022; Zhang et al., 2023; Sun et al., 2024; Zhu et al., 2024; Hao et al., 2024). Parameter decomposition techniques reduce model size by replacing weight matrices with low-rank representations. Reducing the parameter size results in subpar performance, as shown in Figure 2 for Factorized-FL (Jeong & Hwang, 2022). Therefore, communication-efficient methods generally adopt a gradient-decomposition approach. In particular, gradient decomposition methods with a fixed A are also known as random subspace training (Li et al., 2018; Aghajanyan et al., 2021) or *projection optimization*.

Gradient-subspace optimizers. GaLore (Zhao et al., 2024) projects layer gradients onto data-dependent SVD subspaces to save optimizer memory. Federated adaptations align these subspaces

and optimizer states across clients (Peng et al., 2026) or share a random projection per round while uploading full-dimensional updates (He et al., 2026). MAPO's seed-shared, model-wide basis needs no subspace transmission or alignment, so clients exchange only k coefficients with exact averaging. The comparison between MAPO and the federated GaLore is presented in Appendix D.1.

Prior methods decompose each layer separately, producing layer-specific factors A_i and B_i. While respecting tensor geometry, this approach makes the method architecture-dependent, prevents parameter sharing across layers with different shapes, and imposes coarse-grained communication tied to layer dimensions. MAPO overcomes these by flattening the model gradient into a universal vector $\Delta W \in \mathbb{R}^d$, padding it to a length divisible by the arbitrary communication budget k, and evenly partitioning it into k segments $\{\Delta W'_i\}_{i=1}^{k}$, where $\Delta W'_i \in \mathbb{R}^{\lceil d/k \rceil}$. Stacking these segments gives:

$$\Delta W' \in \mathbb{R}^{k \times \lceil d/k \rceil}, \qquad \Delta W' \approx BA, \quad B \in \mathbb{R}^{k \times 1}, \quad A \in \mathbb{R}^{1 \times \lceil d/k \rceil}.$$

This model-wide reshape allows the random reconstruction vector A to be shared across segments, and the size of B directly controls the trade-off between communication and the error rate.

2.2 Parameter Decomposition vs. Gradient Decomposition

Despite their apparent similarities, parameter and gradient decomposition differ fundamentally in their objectives. Parameter decomposition imposes a low-rank structure on model parameters, replacing the model with a compressed architecture. Its communication savings come only from shrinking the model, at the cost of model capacity, since every remaining parameter is still communicated each round. In contrast, gradient decomposition preserves the original model while reducing communication by transmitting compressed gradients, as shown in Figure 1. In parameter decomposition, the forward pass of a layer $W^i \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is modified by routing activations through low-rank matrices, directly constraining their shapes:

$$h^{i+1} = \sigma\big(W^i h^i + B^i (A^i h^i)\big), \qquad A^i \in \mathbb{R}^{r \times d_{\text{in}}},\ B^i \in \mathbb{R}^{d_{\text{out}} \times r},\ r \leq \min(d_{\text{in}}, d_{\text{out}}).$$

In gradient decomposition, the forward pass remains unchanged, and the reparameterization affects only the backward pass. Therefore, low-rank matrix shapes can be chosen more freely:

$$h^{i+1} = \sigma\left(\left(W^i + \text{mat}_{d_{\text{out}} \times d_{\text{in}}}\left(\text{vec}(B^i A^i)_{[0:d_{\text{out}} d_{\text{in}}]}\right)\right) h^i\right), B^i \in \mathbb{R}^{a \times b},\ A^i \in \mathbb{R}^{b \times c},\ ac \geq d_{\text{out}} d_{\text{in}}.$$

In this work, to ensure a fair assessment of communication efficiency, we evaluate MAPO against gradient-based compression baselines under consistent model architectures. Additional experiments with parameter decomposition and LoRA-based methods are provided in Appendices C and D for completeness. Key methodological distinctions among related works are summarized in Table 1.

Table 1: Summary of CEFL: "Comm. Flex" indicates arbitrary bitrates, and "Avg. Eq." denotes equality of low-rank and full-rank average.

Method	Scope	Target	Full-rank Inference	Avg. Eq.	PEFT	Fixed Subspace	Arch-Agnostic	Comm Flex	Personalized FL
Sparsification (Konečný et al., 2016)	Model	Update	✓	✓	✗	✗	✓	✓	✗
Quantization (Reisizadeh et al., 2020)	Model	Update	✓	✓	✗	✗	✓	✓	✗
EvoFed (Rahimi et al., 2023)	Model	Update	✓	✓	✗	✓	✓	✓	✗
Factorized-FL (Jeong & Hwang, 2022)	Layer	Parameter	✗	✗	✗	✗	✗	✗	✓
LoRA (Hu et al., 2022)	Layer	Adapter	✗	✗	✓	✗	✗	✗	✗
FFA-LoRA (Sun et al., 2024)	Layer	Adapter	✗	✓	✓	✓	✗	✗	✗
FedSA-LoRA (Guo et al., 2025b)	Layer	Adapter	✗	✗	✓	✗	✗	✗	✓
FedLoRU (Park & Klabjan, 2024)	Layer	Gradient	✓	✗	✗	✗	✗	✗	✗
MAPO (Ours)	Model	Gradient	✓	✓	✗	✓	✓	✓	✗

3 Model-Agnostic Projection Optimization

In this section, we introduce MAPO and its application in FL. We first present the MAPO factorization technique and discuss its key properties regarding communication efficiency and error rate. Subsequently, we detail how MAPO can be effectively integrated into the FL training process.

MAPO parameterizes the model update $\Delta W \in \mathbb{R}^d$ in a model-agnostic manner, removing architectural constraints and allowing subspace exploration. As shown in Figure 1, it reshapes $\Delta W \in \mathbb{R}^{d \times 1}$ into $\Delta W' \in \mathbb{R}^{k \times \lceil d/k \rceil}$ and decomposes it into a reconstruction vector $A \in \mathbb{R}^{1 \times \lceil d/k \rceil}$ and a projection vector $B \in \mathbb{R}^{k \times 1}$. An illustration is provided in Figure 3, corresponding to Propositions 3.3 to 3.5.

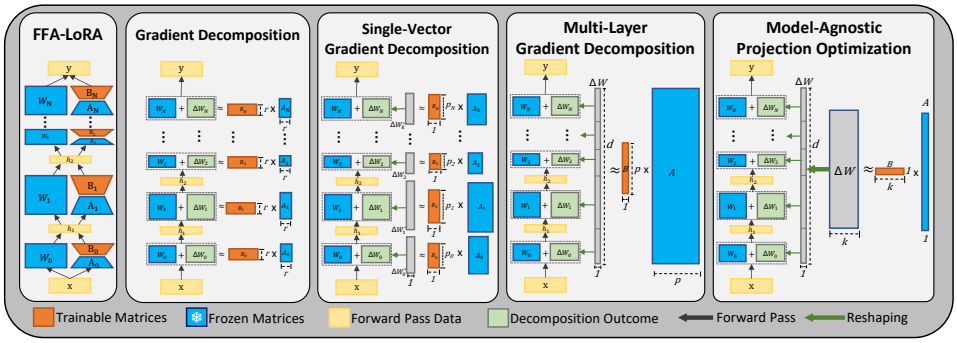


Figure 3: Step-by-step illustration of the methodology based on propositions, demonstrating how each step contributes to the design of the MAPO factorization and how it differs from the LoRA architecture.

MAPO Propositions. We provide a theoretical basis showing that, under the same communication budget, the MAPO factorization achieves the same size-weighted expected error rate as layer-wise factorization with random reconstruction factors, while reducing the total number of parameters. Specifically, since only B is communicated, the parameter reduction is achieved by reducing the size of A, which is the number of parameters required to reconstruct the gradient.

For clarity, we present the argument in three steps. We show that vectorizing a single layer preserves the same expected reconstruction error rate under a matched communication budget. Then, Proposition 3.4 extends this result to the model-wide vector obtained by concatenating all layers, in the size-weighted sense, and states exactly when it also holds energy-weighted. Finally, Proposition 3.5 shows that reshaping the model-wide vector into $\Delta W' \in \mathbb{R}^{k \times \lceil d/k \rceil}$, followed by rank-1 factorization, preserves the reconstruction error and communication overhead while using around *half* of the layer-wise parameters. The formal proofs are located in Appendix H.

Definition 3.1. *Let $\Delta W_i \in \mathbb{R}^{d_1 \times d_2}$ and $\Delta W_i \approx B_i A_i$, where $A_i \in \mathbb{R}^{q \times d_2}$ is fixed, $B_i \in \mathbb{R}^{d_1 \times q}$ is trainable, and $q \leq d_2$, where $B_i^\star \in \arg\min_{B \in \mathbb{R}^{d_1 \times q}} \|\Delta W_i - B A_i\|_F^2$. We define the **communication overhead rate** and **reconstruction error rate** as:*

$$\mathrm{CO}_{\mathrm{rate}} := \frac{|B_i|}{|\Delta W_i|} = \frac{d_1 q}{d_1 d_2} = \frac{q}{d_2}, \qquad \mathrm{RE}_{\mathrm{rate}}(A_i) := \frac{\|\Delta W_i - B_i^\star A_i\|_F^2}{\|\Delta W_i\|_F^2},$$

Lemma 3.2. *Under Definition 3.1, let A_i be a random matrix with entries drawn independently from a Gaussian $\mathcal{N}(0, \sigma^2)$ with $\sigma > 0$. Then, A_i is almost surely of full rank, as the probability of A_i being rank deficient is zero (Vershynin, 2018; Tao, 2012), and for any fixed nonzero ΔW_i we have:*

$$\mathbb{E}_{A_i}[\mathrm{RE}_{\mathrm{rate}}(A_i)] = 1 - \frac{q}{d_2}.$$

The result follows since $P_{A_i} = A_i^\top (A_i A_i^\top)^{-1} A_i$ is the orthogonal projector onto a uniformly random q-dimensional subspace of $\mathbb{R}^{d_2}$, so $\mathbb{E}[P_{A_i}] = (q/d_2) I_{d_2}$. The formal proof is in Appendix H.1.

Proposition 3.3 (Single-Vector Factorization). *Let ΔW_i, A_i, and B_i be factorizations of a single layer of the network as in Definition 3.1. By reshaping ΔW_i into $\Delta W_i' \in \mathbb{R}^{1 \times d_1 d_2}$, the factorization $\Delta W_i' \approx B_i' A_i'$ where $A_i' \in \mathbb{R}^{p \times d_1 d_2}$ and $B_i' \in \mathbb{R}^{1 \times p}$ achieves the same expected **reconstruction error** rate and **communication overhead** as the conventional factorization of ΔW_i when $p = q d_1$.*

Proposition 3.4 (Multi-Layer Factorization). *Let ΔW_i, A_i, and B_i be **single-vector factorization** of i-th layer of the N-layer network as in Proposition 3.3. By concatenating the reshaped updates ΔW_i into $\Delta W' \in \mathbb{R}^{1 \times d}$, where $d = \sum_{i=1}^N d_1^i d_2^i$, the factorization $\Delta W' \approx B' A'$ with $A' \in \mathbb{R}^{p \times d}$ and $B' \in \mathbb{R}^{1 \times p}$ achieves the same size-weighted **aggregate reconstruction error** rate, $\sum_i (d_1^i d_2^i / d)(1 - q/d_2^i) = 1 - p/d$, and **communication overhead** as the single-vector factorization applied to each ΔW_i when $p = \sum_{i=1}^N q\, d_1^i$. The energy-weighted layer-wise error $\sum_i (1 - q/d_2^i) \|\Delta W_i\|_F^2 / \|\Delta W\|_2^2$ also equals $1 - p/d$ if all d_2^i are equal or $\|\Delta W_i\|_F^2 \propto d_1^i d_2^i$, but not in general.*

Proposition 3.5 (MAPO Factorization). *Let ΔW, A, B, and rank p be a **multi-layer factorization** as in Proposition 3.4. Assume $k \mid d$. By reshaping $\Delta W \in \mathbb{R}^{1 \times d}$ into $\Delta W' \in \mathbb{R}^{k \times \lceil d/k \rceil}$ and using the factorization $\Delta W' \approx B' A'$ with $A' \in \mathbb{R}^{1 \times \lceil d/k \rceil}$ and $B' \in \mathbb{R}^{k \times 1}$, we achieve the same expected **reconstruction error** rate, $1 - p/d$, and **communication overhead**, p/d, when $k = p$, with around **half** of the layer-wise parameters.*

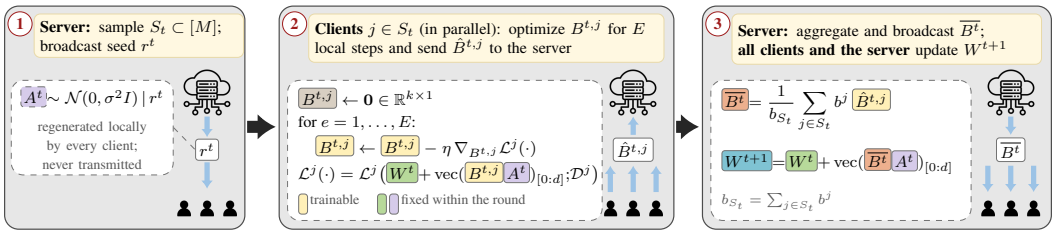


Figure 4: MAPO application to communication-efficient federated learning

In the common equal-width case, where $d_1^i = d_2^i = h$ and $q_i = q$ for all $i \in [N]$, the reconstruction factor A shrinks by a factor of Nq^2 compared to layer-wise factorization. Since the communicated B size is matched by construction, the total $A + B$ count for MAPO is approximately *half* of the layer-wise. The exact ratio for arbitrary widths under a common rank q, $(k + \lceil d/k \rceil)/(k + q \sum_i d_2^i)$, is provided in Appendix H.3.

3.1 Application to Communication-Efficient FL

This subsection explains how MAPO is utilized in FL. The procedure pseudocode is provided in Algorithm 1 and visualized in Figure 4.

Round structure. The server and all clients initialize the same model W^0 from a shared seed. In round $t \in \{0, \ldots, T-1\}$, the server samples the participating clients S_t and broadcasts a fresh seed r^t, from which each client regenerates the same reconstruction vector $A^t \sim \mathcal{N}(0, \sigma^2 I)$ and resets $B^{t,j} \leftarrow \mathbf{0}$. Hence W^t and A^t are identical across the network, and A^t is never transmitted. Here, **vec**$(\cdot)$ and $(\cdot)_{[0:d]}$ denote vectorization and truncation to the first d elements.

Algorithm 1 Federated Learning with MAPO

Input: shared $W^0 \in \mathbb{R}^d$; k, σ, η, E, T; $n = \lceil d/k \rceil$
for $t = 0, \ldots, T-1$ **do**
  Server: sample $S_t \subset [M]$; broadcast seed r^t
  Clients $j \in S_t$ *(in parallel):*
  $A^t \leftarrow \mathcal{N}(0, \sigma^2 I_n) \mid r^t$; $B^{t,j} \leftarrow \mathbf{0} \in \mathbb{R}^{k \times 1}$
  for $e = 1, \ldots, E$ **do**
    $\nabla B^{t,j} \leftarrow \nabla_{B^{t,j}} \mathcal{L}^j\left(W^t + \mathrm{vec}(B^{t,j} A^t)[0:d], \mathcal{D}^j\right)$
    $B^{t,j} \leftarrow B^{t,j} - \eta \nabla B^{t,j}$
  end for
  Send $\hat{B}^{t,j} \leftarrow B^{t,j}$ to the server
  Server: $\overline{B^t} \leftarrow \frac{1}{b_{S_t}} \sum_{j \in S_t} b^j \hat{B}^{t,j}$ equation 2
  Server & clients: $W^{t+1} \leftarrow W^t + \mathrm{vec}(\overline{B^t} A^t)[0:d]$
end for
Return W^T

Local Projection Optimization. After initializing A^t and $B^{t,j}$, clients perform local training to optimize $B^{t,j}$ using their local data $\mathcal{D}^j$. This step optimizes the projection $\hat{B}^{t,j}$ to minimize the client loss $\mathcal{L}(W^t + \mathbf{vec}(B^{t,j} A^t)_{[0:d]}, \mathcal{D}^j)$. The gradient of the projection vector is computed as:

$$\nabla B^{t,j} = \nabla_{B^{t,j}} \mathcal{L}^j\left(W^t + \mathbf{vec}(B^{t,j} A^t)_{[0:d]};\ \mathcal{D}^j\right), \quad \text{where } \mathcal{L}^j(W) = \frac{1}{|\mathcal{D}^j|} \sum_{x \in \mathcal{D}^j} \ell(W, x). \tag{1}$$

where $\ell(W, x)$ is the loss function given model W and data point x. After E steps of $B^{t,j} \leftarrow B^{t,j} - \eta \nabla B^{t,j}$ with learning rate η, clients send the optimized projection vector $\hat{B}^{t,j}$ to the server.

Server-Side Aggregation and Global Model Update. Upon receiving the projection vectors $\hat{B}^{t,j}$ and their corresponding weights $b^j = |D^j|$ (the number of local samples) from the participating clients $j \in S_t$, the server aggregates them to form the global projection vector:

$$\overline{B^t} = \frac{1}{b_{S_t}} \sum_{j \in S_t} b^j \hat{B}^{t,j}, \quad \text{where} \quad b_{S_t} = \sum_{j \in S_t} b^j. \tag{2}$$

This weighted averaging captures the collective contribution of participating clients, proportional to their data sizes. Section 4 analysis assumes a uniform weights, $b^j \equiv 1$. Since all clients share A^t, averaging the coefficients is exactly averaging the reconstructed updates, so no low-rank averaging error arises. The server broadcasts $\overline{B^t}$, and the server and all clients update $W^{t+1} = W^t + \mathbf{vec}(\overline{B^t} A^t)_{[0:d]}$. This process is repeated for $t = 0, \ldots, T-1$, and Algorithm 1 returns W^T.

4 CONVERGENCE ANALYSIS

We analyze MAPO directly over the optimized variable B. Let $n := \lceil d/k \rceil$ and let $\mathcal{R} : \mathbb{R}^d \to \mathbb{R}^{k\times n}$ denote MAPO's pad-and-reshape operation, such that $\mathrm{vec}(BA)_{[0:d]} = \mathcal{R}^*(BA)$ and $\mathcal{R}^*\mathcal{R} = I_d$. We consider one local SGD step ($E = 1$) from $B^{t,j} = \mathbf{0}$ with uniform aggregation weights, where a fresh $A^t \sim \mathcal{N}(0, \sigma^2 I_n)$ is sampled at each round independently of all other randomness, and minibatches are drawn independently across clients; this simplifies Algorithm 1, whose data-size weights, $E > 1$, momentum, and AdamW are not covered (Remark I.9). If $\tilde{g}^{t,j}$ denotes client j's stochastic gradient at W^t, the resulting MAPO update can be written as

$$W^{t+1} = W^t - \eta\, U_{A^t}(\tilde{g}^t), \qquad U_A(g) := \mathcal{R}^*\big((\mathcal{R}g)A^\top A\big), \qquad \tilde{g}^t := \tfrac{1}{m}\textstyle\sum_{j\in S_t} \tilde{g}^{t,j}. \tag{3}$$

For any fixed g, this update satisfies $\mathbb{E}_A[U_A(g)] = \sigma^2 g$ and $\mathbb{E}_A\|U_A(g)\|^2 \le \sigma^4(n+2)\|g\|^2$.

Assumption 4.1. *For each j, $\mathcal{L}^j$ is β-smooth, i.e., $\|\nabla\mathcal{L}^j(u) - \nabla\mathcal{L}^j(v)\| \le \beta\|u - v\|$ for any u, v, and $\mathcal{L} := \frac{1}{M}\sum_j \mathcal{L}^j$ is bounded below by $\mathcal{L}^*$.*

Assumption 4.2. *Variance of the SGD is bounded for client j: the minibatch gradient $\widetilde{\nabla}\mathcal{L}^j(W)$ is unbiased and $\mathbb{E}[\|\nabla\mathcal{L}^j(W) - \widetilde{\nabla}\mathcal{L}^j(W)\|^2] \le \sigma_l^2$.*

Assumption 4.3. *Bounded clients' gradient dissimilarity: $\frac{1}{M}\sum_{j=1}^M \big\|\nabla\mathcal{L}^j(W) - \nabla\mathcal{L}(W)\big\|^2 \le \sigma_g^2$.*

Assumption 4.4. *At each round t, the server selects a subset $S_t \subset [M]$ with $|S_t| = m < M$ uniformly without replacement, with sampling variance $\sigma_{\mathrm{het}}^2 = \rho\sigma_g^2$, where $\rho = \frac{M-m}{m(M-1)}$.*

Theorem 4.5. *Let Assumptions 4.1 to 4.4 hold in the setting above; define $\Sigma^2 := \sigma_l^2/m + \sigma_{\mathrm{het}}^2$, and let $\gamma := \eta\sigma^2$. If $\gamma \le \frac{1}{\beta(n+2)}$, where $n = \lceil d/k \rceil$, then for every $T \ge 1$,*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\big[\|\nabla\mathcal{L}(W^t)\|^2\big] \le \frac{2(\mathcal{L}(W^0) - \mathcal{L}^*)}{\gamma T} + \beta(n+2)\gamma\Sigma^2.$$

In the noiseless case, $\sigma_l = \sigma_g = 0$, choosing $\gamma = \frac{1}{\beta(n+2)}$ gives the rate $2\beta(n+2)(\mathcal{L}(W^0) - \mathcal{L}^*)/T$. With stochastic gradients and partial participation, the best constant step size for this bound gives $O\big(\frac{n}{T} + \sqrt{\frac{n}{T}}\big)$. Moreover, for deterministic steps $\gamma_t = \eta_t\sigma^2 \in (0, \frac{1}{\beta(n+2)}]$ satisfying $\sum_t \gamma_t = \infty$ and $\sum_t \gamma_t^2 < \infty$, we obtain

$$\frac{\sum_{t<T} \gamma_t\, \mathbb{E}\big[\|\nabla\mathcal{L}(W^t)\|^2\big]}{\sum_{t<T}\gamma_t} \to 0,$$

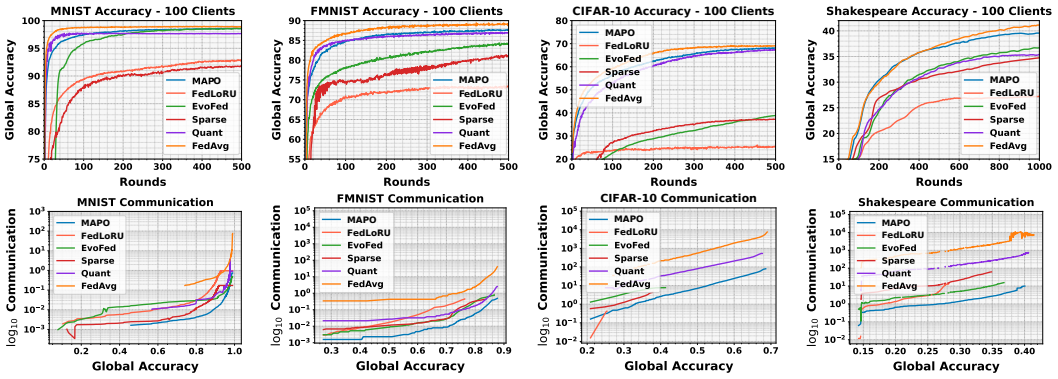
so an iterate W^τ drawn with $\Pr(\tau = t) \propto \gamma_t$ is stationary in the limit: $\mathbb{E}\|\nabla\mathcal{L}(W^\tau)\|^2 \to 0$.

Importantly, Theorem 4.5 holds for *every* $k \in \{1, \dots, d\}$ without requiring a small per-round reconstruction error. Compression enters only through $n + 2$, where $n = \lceil d/k \rceil \approx d/k$, as a variance and round-complexity factor. Although each reconstruction captures only a $1/n$ fraction of the gradient energy in expectation, independently resampling A^t keeps the update unbiased up to the factor σ^2. Thus, a large residual $\epsilon_t \approx 1 - 1/n$ is expected under strong compression and does not prevent convergence (Proposition I.11). In contrast, freezing A confines the iterates to a fixed affine subspace and can produce a nonzero gradient floor (Proposition I.10). Proofs are in Appendix I.

5 EXPERIMENTAL SETUP

We evaluate MAPO across diverse architectures, tasks, and baselines. Our benchmarks include five image classification datasets, MNIST (LeCun et al., 1998), FMNIST (Xiao et al., 2017), CIFAR-10, CIFAR-100 (Krizhevsky, 2009), and TinyImageNet (Le & Yang, 2015), as well as Shakespeare next-character prediction and Sentiment140 sentiment analysis from the FL-oriented LEAF suite (Caldas et al., 2018). We also evaluate MAPO for fine-tuning on GLUE (Wang et al., 2018a) against LoRA-based methods, with communication and computational results reported in Appendix C. Dataset specifications and model architectures are summarized in Table 9 (Appendix E).

Non-IID Distribution. We partition the training datasets in a non-IID manner across 100 clients. For image classification and GLUE tasks, each client holds samples from only a few classes (Table 9). For LEAF tasks, we follow the natural user-based partitioning, where individual Shakespearean roles and Twitter users correspond to separate clients.



(a) MNIST (b) FMNIST (c) CIFAR-10 (d) Shakespeare

Figure 5: **Performance comparison** on MNIST, FMNIST, CIFAR-10, and Shakespeare datasets. The top row shows accuracy per round, while the bottom shows the communication required to achieve each accuracy level.

Table 2: Maximum accuracy (%, mean ± std over 3 runs) and communication (Com.): cumulative uplink needed to reach the worst-performing baseline's accuracy, as % of FedAvg's.

	MNIST		FMNIST		CIFAR-10		CIFAR-100		Shakespeare		Sentiment140		TinyImageNet	
Method	Com.	Acc.	Com.	Acc.	Com.	Acc.	Com.	Acc.	Com.	Acc.	Com.	Acc.	Com.	Acc.
FedAvg	100	98.9 (±0.1)	100	89.2 (±0.2)	100	69.0 (±0.2)	100	43.47 (±0.3)	100	41.86 (±0.3)	100	74.90 (±0.3)	100	36.48 (±0.4)
Sparse	15.3	92.1 (±0.4)	24.1	81.1 (±0.4)	2.7	37.15 (±0.5)	1.20	33.72 (±0.5)	1.73	34.86 (±0.4)	1.93	74.21 (±0.3)	1.32	25.34 (±0.5)
Quantize	31.3	97.6 (±0.2)	24.1	87.1 (±0.3)	15.2	67.40 (±0.3)	6.10	40.05 (±0.4)	10.11	35.45 (±0.4)	13.85	73.70 (±0.3)	8.75	34.47 (±0.4)
EvoFed	9.40	98.5 (±0.2)	7.60	84.7 (±0.3)	3.4	39.50 (±0.4)	20.4	37.62 (±0.4)	0.23	36.76 (±0.3)	0.40	70.50 (±0.3)	1.85	15.40 (±0.5)
FedLoRU	30.2	93.8 (±0.4)	17.9	74.1 (±0.5)	1.7	23.52 (±0.5)	1.20	19.10 (±0.5)	1.67	28.07 (±0.5)	1.30	66.61 (±0.4)	1.27	7.31 (±0.5)
MAPO	**2.95**	**98.6** (±0.1)	**3.10**	**88.0** (±0.2)	**1.20**	**68.3** (±0.2)	**0.91**	**40.16** (±0.3)	**0.13**	**39.96** (±0.3)	**0.19**	**74.50** (±0.2)	**0.97**	**35.22** (±0.3)

Model Architectures. We evaluate MAPO across diverse architectures of varying complexity, including CNNs for MNIST, FMNIST, and CIFAR-10; WideResNet for CIFAR-100 and TinyImageNet; LSTM for next-character prediction; Transformer for sentiment analysis; and RoBERTa for GLUE tasks. Detailed architecture specifications and hyperparameters are in Appendix E.

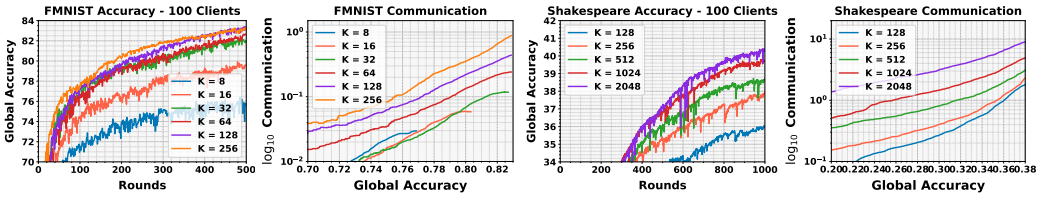
Baselines. We compare MAPO with standard compression methods based on subsampling (Sparse) (Konečný et al., 2016) and quantization (Quantize) (Reisizadeh et al., 2020), as well as EvoFed (Rahimi et al., 2023) and FedLoRU (Park & Klabjan, 2024). Sparse and Quantize represent conventional compression, while EvoFed evaluates MAPO against post-optimization compression and FedLoRU against layer-wise low-rank updates with trainable factors. Comparisons with parameter factorization (Jeong & Hwang, 2022) and adapter-based methods (LoRA (Hu et al., 2022), FFA-LoRA (Sun et al., 2024), and FedSA-LoRA (Guo et al., 2025b)) are in Appendices C and D.

Federated Learning Setting. In each training round, 10% of the clients are randomly selected to participate. Selected clients train locally in parallel and transmit their updates to the central server, which aggregates these updates and redistributes the resulting global model back to the clients.

6 Results and Discussion

This section reports the maximum accuracy per communication budget and the communication required to reach a target accuracy. Figure 5 (top) shows that MAPO outperforms other communication-efficient methods and approaches FedAvg accuracy while transmitting only a fraction of the parameters. This gain comes from MAPO's dynamic subspace optimization, which improves exploration and communication efficiency. Figure 5 (bottom) further shows that MAPO requires substantially less communication to reach any given accuracy. Additional results are provided in Appendix B.

Table 2 summarizes experimental results by comparing each baseline's maximum accuracy and communication cost relative to FedAvg. To ensure fair comparison, communication costs are reported as the percentage of FedAvg's cumulative uplink required to reach the accuracy of the worst-performing baseline. MAPO consistently achieves competitive accuracy with significantly lower communication overhead. Specifically, on MNIST and FMNIST, MAPO achieves 99.7% and 98.6% of FedAvg's accuracy, respectively, using only 3% of FedAvg's communication cost. For CIFAR-10, CIFAR-100, and TinyImageNet, MAPO attains 98.9%, 92.4%, and 96.5% of FedAvg's accuracy, respectively, while incurring approximately 1% of FedAvg's communication cost. Finally, in sequential tasks (Shakespeare and Sentiment140), MAPO retains up to 95.5% and 99.5% of FedAvg's accuracy, respectively, while dramatically reducing communication to less than 0.2%.

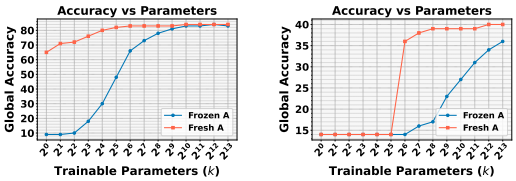


(a) FMNIST Accuracy (b) FMNIST Communication (c) Shakespeare Accuracy (d) Shakespeare Communication

Figure 6: Accuracy and communication cost per accuracy level for the FMNIST and Shakespeare datasets, demonstrating the effect of the number of trainable parameters (k) on MAPO's communication efficiency.

MAPO Hyperparameter. MAPO simplifies gradient projection by applying a single factorization across all model parameters, thereby replacing per-layer rank with a single hyperparameter, k, that directly controls communication cost and model accuracy. Figure 6 illustrates the effect of k on performance and communication efficiency for the FMNIST and Shakespeare datasets. While a smaller k significantly reduces communication overhead, it slows convergence, requiring more training rounds. Conversely, increasing k improves convergence speed and accuracy but rapidly raises communication costs, often with diminishing returns. Therefore, the optimal k achieves a target accuracy with minimal total communication. Figure 6(b) and (d) show communication costs associated with specific accuracy levels, guiding the selection of optimal k. We use the same guidelines for all baselines to ensure fair hyperparameter tuning.

Fresh reconstruction vector. A key factor in MAPO's performance is using a dynamically generated reconstruction vector A rather than a fixed one. This approach promotes the exploration of new subspaces throughout training. Figure 7 illustrates the benefits of using a fresh A on the FMNIST and Shakespeare datasets. We evaluate MAPO across varying numbers of trainable parameters, ranging from 2^0 to 2^{13}.



(A) FMNIST (B) Shakespeare

Figure 7: Comparison between fresh and frozen A.

For FMNIST, this corresponds to 0.009% to 72.66% of the total model parameters, while for Shakespeare, it ranges from 0.0001% to nearly 1%. In both cases, MAPO with a fresh A achieves superior convergence with fewer parameters, thereby leveraging the search space more effectively. In contrast, when A is frozen, performance scales logarithmically with the number of trainable parameters, requiring an exponentially larger number of parameters to match a fresh A. This matches Proposition I.10 as the frozen A confines training to a subspace with a nonzero gradient floor.

Additional Results. Comparisons with LoRA-based methods and Factorized-FL are provided in Appendices C and D. The ablation study on the effect of the exploration of A on training stability is presented in Appendix G. Finally, Appendix J presents a detailed analysis of memory complexity, emphasizing computational efficiency and flexibility compared to layer-wise low-rank factorization. Furthermore, FedAvg is the uncompressed reference, not a competitor; Appendices C.1, D.1 and F add ViT/Llama models, GaLore-FL, and extreme label skew.

Limitations. MAPO's improved communication efficiency comes at the cost of additional computational overhead from projection optimization. While significantly reduced compared to prior layer-wise methods, MAPO still requires $\lceil d/k \rceil + k$ extra factor storage plus an $O(d)$ update buffer, with the computational overhead remaining $O(d)$ per step (see Appendix J). Our convergence guarantee covers one local SGD step with uniform aggregation (Remark I.9).

7 Conclusion

We introduced *Model-Agnostic Projection Optimization* (MAPO), a novel approach for CEFL. Unlike layer-wise decomposition, MAPO parameterizes the entire model update using a projection vector and a random reconstruction vector, regenerated at each round. MAPO balances communication efficiency and accuracy without imposing architecture-specific constraints or fixed-subspace limitations. Our theoretical analysis establishes a one-local-step convergence guarantee, and empirical results demonstrate superior performance and scalability across diverse datasets and model architectures.

AI USE STATEMENT

We used generative AI tools to improve the text's readability. We did not use generative AI tools to generate data, run experiments, or produce results. All AI-assisted content was verified by the authors, who take full responsibility for the content of this paper.

REFERENCES

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 7319–7328. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.568.

Kazuoki Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal, Second Series*, 19(3):357–367, 1967. doi: 10.2748/tmj/1178243286.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: Composable transformations of Python+NumPy programs. `http://github.com/jax-ml/jax`, 2018. Version 0.3.13.

Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Wenlin Chen, James Wilson, Stephen Tyree, Kilian Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In *International Conference on Machine Learning (ICML)*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2285–2294. PMLR, 2015.

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.

Misha Denil, Babak Shakibi, Laurent Dinh, Marc'Aurelio Ranzato, and Nando de Freitas. Predicting parameters in deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 26, pp. 2148–2156, 2013.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

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

Mingzhao Guo, Dongzhu Liu, Osvaldo Simeone, and Dingzhu Wen. Efficient wireless federated learning via low-rank gradient factorization. *IEEE Transactions on Vehicular Technology*, 74(4): 6811–6816, 2025a. doi: 10.1109/TVT.2024.3506950.

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)*, 2025b.

Marawan Gamal Abdel Hameed, Aristides Milios, Siva Reddy, and Guillaume Rabusseau. ROSA: Random subspace adaptation for efficient fine-tuning. *arXiv preprint arXiv:2407.07802*, 2024.

Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In *International Conference on Learning Representations (ICLR)*, 2016.

Yongchang Hao, Yanshuai Cao, and Lili Mou. Flora: Low-rank adapters are secretly gradient compressors. In *International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17554–17571. PMLR, 2024.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1398–1406, 2017. doi: 10.1109/ICCV.2017.155.

Yutong He, Zhengyang Huang, Jiahe Geng, and Kun Yuan. FedSLoP: Memory-efficient federated learning with low-rank gradient projection. *arXiv preprint arXiv:2604.24012*, 2026.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. doi: 10.1080/01621459.1963.10500830.

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

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.

Nam Hyeon-Woo, Moon Ye-Bin, and Tae-Hyun Oh. FedPara: Low-rank Hadamard product for communication-efficient federated learning. In *International Conference on Learning Representations (ICLR)*, 2022.

Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Speeding up convolutional neural networks with low rank expansions. In *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA Press, 2014. doi: 10.5244/C.28.88.

Wonyong Jeong and Sung Ju Hwang. Factorized-FL: Personalized federated learning with parameter factorization & similarity matching. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 35684–35695, 2022.

Ninghui Jia, Zhihao Qu, Baoliu Ye, Yanyan Wang, Shihong Hu, and Song Guo. A comprehensive survey on communication-efficient federated learning in mobile edge environments. *IEEE Communications Surveys & Tutorials*, 27(6):3710–3741, 2025. doi: 10.1109/COMST.2025.3535957.

Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Dawid J. Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. VeRA: Vector-based random matrix adaptation. In *International Conference on Learning Representations (ICLR)*, 2024.

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

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge. Stanford CS231N course project report, 2015.

Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan Oseledets, and Victor Lempitsky. Speeding-up convolutional neural networks using fine-tuned CP-decomposition. In *International Conference on Learning Representations (ICLR)*, 2015.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations (ICLR)*, 2018.

Yang Li, Shaobo Han, and Shihao Ji. VB-LoRA: Extreme parameter efficient fine-tuning with vector banks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 16724–16751, 2024.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. ThiNet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 5068–5076, 2017. doi: 10.1109/ICCV.2017.541.

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Haemin Park and Diego Klabjan. Communication-efficient federated low-rank update algorithm and its connection to implicit regularization. *arXiv preprint arXiv:2409.12371*, 2024.

Hongyi Peng, Han Yu, Xiaoxiao Li, and Qiang Yang. Rethinking LoRA for data heterogeneous federated learning: Subspace and state alignment. *arXiv preprint arXiv:2602.01746*, 2026.

Mohammad Mahdi Rahimi, Hasnain Irshad Bhatti, Younghyun Park, Humaira Kousar, and Jaekyun Moon. EvoFed: Leveraging evolutionary strategies for communication-efficient federated learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 62428–62441, 2023.

Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2021–2031. PMLR, 2020.

Tara N. Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6655–6659, 2013.

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

Terence Tao. *Topics in Random Matrix Theory*, volume 132 of *Graduate Studies in Mathematics*. American Mathematical Society, 2012.

Karen Ullrich, Edward Meeds, and Max Welling. Soft weight-sharing for neural network compression. In *International Conference on Learning Representations (ICLR)*, 2017.

Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018.

Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. PowerSGD: Practical low-rank gradient compression for distributed optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, 2018a.

Wenqi Wang, Yifan Sun, Brian Eriksson, Wenlin Wang, and Vaneet Aggarwal. Wide compression: Tensor ring nets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9329–9338, 2018b.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-demos.6.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Yuexiang Xie, Zhen Wang, Dawei Gao, Daoyuan Chen, Liuyi Yao, Weirui Kuang, Yaliang Li, Bolin Ding, and Jingren Zhou. FederatedScope: A flexible federated learning platform for heterogeneity. *Proceedings of the VLDB Endowment*, 16(5):1059–1072, 2023.

Dezhong Yao, Wanning Pan, Yuexin Shi, Michael J. O'Neill, Yutong Dai, Yao Wan, Peilin Zhao, Hai Jin, and Lichao Sun. FedHM: Efficient federated learning for heterogeneous models via low-rank factorization. *Artificial Intelligence*, 344:104333, 2025. doi: 10.1016/j.artint.2025.104333.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, pp. 87.1–87.12. BMVA Press, 2016. doi: 10.5244/C.30.87.

Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. LoRA-FA: Memory-efficient low-rank adaptation for large language models fine-tuning. *arXiv preprint arXiv:2308.03303v1*, 2023.

Tianyun Zhang, Shaokai Ye, Kaiqi Zhang, Jian Tang, Wujie Wen, Makan Fardad, and Yanzhi Wang. A systematic DNN weight pruning framework using alternating direction method of multipliers. In *European Conference on Computer Vision (ECCV)*, volume 11212 of *Lecture Notes in Computer Science*, pp. 191–207. Springer, 2018. doi: 10.1007/978-3-030-01237-3_12.

Hanzhen Zhao, Xingyu Xie, Cong Fang, and Zhouchen Lin. SEPARATE: A simple low-rank projection for gradient compression in modern large-scale model training process. In *International Conference on Learning Representations (ICLR)*, 2025.

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 (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61121–61143. PMLR, 2024.

Jiacheng Zhu, Kristjan Greenewald, Kimia Nadjahi, Haitz Sáez de Ocáriz Borde, Rickard Brüel Gabrielsson, Leshem Choshen, Marzyeh Ghassemi, Mikhail Yurochkin, and Justin Solomon. Asymmetry in low-rank adapters of foundation models. In *International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62369–62385. PMLR, 2024.

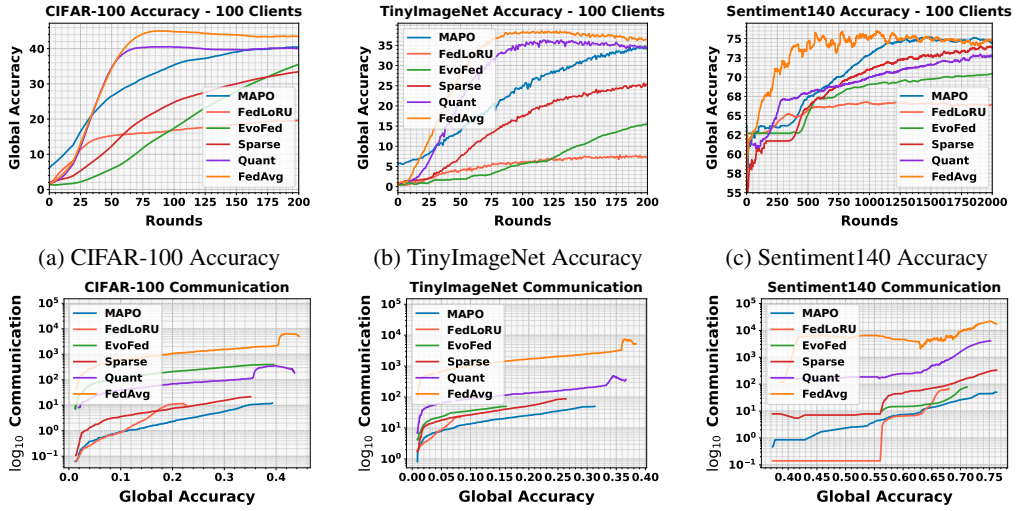
A NOTATIONS

Table 3: Notation and Definitions

Symbol	Meaning / Definition
N	Number of layers in a model.
i	Indexing notation for the layers of the model. ($1 \leq i \leq N$)
M	Number of clients in FL.
j	Indexing notation for clients. ($1 \leq j \leq M$)
T	Total number of communication rounds in FL.
t	Indexing notation for rounds. ($0 \leq t \leq T-1$)
$\mathcal{D}^j$	Local dataset for client j.
b^j	Weight for client j, usually set as the number of local samples $\lvert\mathcal{D}^j\rvert$.
ΔW	Model update, treated as a single vector, $\in \mathbb{R}^{d\times 1}$.
W^t	Model parameters at communication round t.
$\overline{B^t}$	Aggregated projection vector at round t, broadcast by the server.
r^t	Random seed used to synchronize vector generation across clients and the server.
A^t	Reconstruction vector at round t, regenerated using r^t.
$B^{t,j}$	Trainable projection vector for client j at round t.
$\hat{B}^{t,j}$	Locally optimized projection vector for client j at round t.
η	Learning rate for local optimization.
d	Total number of model parameters, $d = \sum_\ell \lvert\Theta_\ell\rvert$ over all parameter tensors Θ_ℓ.
d_1^i, d_2^i	Row and column dimensions of the weight matrix for layer i; viewing each tensor as a matrix, $d = \sum_i d_1^i d_2^i$.
p	Factorization rank after reshaping.
q	LoRA Factorization rank before reshaping.
k	Design parameter controlling reshape dimension ($\Delta W'$ reshaped into $\mathbb{R}^{k\times\lceil d/k\rceil}$).
$A \in \mathbb{R}^{\cdot\times\cdot}$, $B \in \mathbb{R}^{\cdot\times\cdot}$	Reconstruction and projection factors (vectors in MAPO) in factorization.
$\mathcal{L}(W)$	Global loss function.
$\mathcal{L}^j(W)$	Local loss function for client j.
$\nabla\mathcal{L}(W)$	Gradient of the global loss function.
$\widetilde{\nabla}\mathcal{L}^j(W)$	Stochastic (mini-batch) gradient of $\mathcal{L}^j$ at W.
$\nabla B^{t,j}$	Gradient of local loss for the projection vector.
E	Number of local optimization steps per round.
S_t	Set of clients selected by the server at round t, $\lvert S_t\rvert = m$.
m, ρ	Number of sampled clients and sampling-variance factor $\rho = \frac{M-m}{m(M-1)}$.
σ_l^2	Bounded variance of stochastic gradients.
σ_g^2	Bound on clients' gradient dissimilarity.
σ_{het}^2	Sampling-induced variance, $\sigma_{\text{het}}^2 = \rho\sigma_g^2$.
Σ^2	Variance bound of the aggregated stochastic gradient, $\Sigma^2 = \sigma_l^2/m + \sigma_{\text{het}}^2$.
σ^2, γ	Variance of the entries of A^t; effective step size $\gamma = \eta\sigma^2$.
n, n_i	Segment length $n = \lceil d/k\rceil$; number of real (non-padded) coordinates of segment i.
$\mathcal{R}$, $U_A(g)$	Padding/reshape map with $\mathrm{vec}(BA)_{[0:d]} = \mathcal{R}^*(BA)$; MAPO step $U_A(g) = \mathcal{R}^*((\mathcal{R}g)A^\top A)$.
β	Smoothness constant of the loss function.
ϵ_t	relative reconstruction residual of the round-t gradient along A^t.

B ACCURACY AND COMMUNICATION LEARNING CURVES

This appendix provides extended experimental results that complement the main findings discussed in Section 6. We include detailed evaluations of MAPO and baseline methods on CIFAR-100, TinyImageNet, and Sentiment140 datasets. Similar to the main results, Figure 8 reports both maximum test accuracy and the communication cost required to reach a given accuracy threshold. These additional experiments further demonstrate MAPO's superior communication efficiency and consistent performance gains on more challenging, large-scale tasks.



(a) CIFAR-100 Accuracy (b) TinyImageNet Accuracy (c) Sentiment140 Accuracy

(d) CIFAR-100 Comm. cost (e) TinyImageNet Comm. cost (f) Sentiment140 Comm. cost

Figure 8: **Performance comparison** of MAPO and baseline methods on CIFAR-100, TinyImageNet, and Sentiment140 datasets. The top row shows the accuracy achieved by each method on its respective dataset, while the bottom row illustrates the communication cost for each method.

C COMPARISON WITH LOW-RANK ADAPTATION IN FINE-TUNING

We conduct fine-tuning experiments using RoBERTa-large (Liu et al., 2019) on five GLUE tasks to evaluate MAPO alongside LoRA, FFA-LoRA, and FedSA-LoRA. Table 4 compares the number of updated and optimized parameters and the communication load per round for each method. Table 5 summarizes fine-tuning results under federated settings, reporting communication efficiency based on the number of rounds and total communication required to reach 80% accuracy. Overall, the results indicate that MAPO improves communication efficiency without compromising performance.

Table 4: Parameters updated, variables optimized per round, and parameters communicated per round. MAPO updates all weights through k optimized coefficients.

Method	Parameters updated	Optimized per round	Communicated per round
LoRA	1.83M	1.83M	0.78M
FFA-LoRA	1.44M	1.44M	0.39M
FedSA-LoRA	1.83M	1.83M	0.39M
MAPO$_{d/1k}$	357M	0.36M	0.36M
MAPO$_{d/10k}$	357M	35.70K	35.70K
MAPO$_{d/100k}$	357M	3.57K	3.57K
MAPO$_{d/1m}$	357M	357	357

C.1 SCALING TO LARGER MODELS

To test MAPO beyond RoBERTa-large, we fine-tune pretrained ViT-B/16 and ViT-L/16 (Dosovitskiy et al., 2021) (85.9M and 303.3M parameters) on CIFAR-100, and Llama-3.2-3B-Instruct (Grattafiori et al., 2024) (3.21B parameters) on GSM8K (Cobbe et al., 2021), all with full (non-adapter) projected training and fp16 communication accounting. ViT-B/16 is trained for 100 communication rounds

Table 5: Comparison of model accuracies, communication rounds, and total communication cost.

Model	SST2			QNLI			RTE			MNLIm			MNLImm		
	Acc	Round	Total	Acc	Round	Total	Acc	Round	Total	Acc	Round	Total	Acc	Round	Total
LoRA	84.86	36	28.08M	91.72	85	66.30M	86.62	180	140.40M	87.41	86	67.08M	87.34	82	63.96M
FFA-LoRA	94.15	44	17.16M	91.63	76	29.64M	57.28	—	—	85.92	76	29.64M	86.46	213	83.07M
FedSA-LoRA	95.41	19	7.41M	91.04	55	21.45M	70.01	—	—	89.44	29	11.31M	85.49	126	49.14M
MAPO$_{d/1k}$	96.79	5	1.78M	93.14	11	3.93M	87.91	23	8.21M	88.90	17	6.07M	88.26	22	7.85M
MAPO$_{d/10k}$	96.10	5	178.50K	92.57	8	285.60K	89.57	23	821.10K	88.81	18	642.60K	87.43	25	892.50K
MAPO$_{d/100k}$	95.53	5	17.85K	89.24	7	24.99K	84.38	24	85.68K	85.04	20	71.40K	84.60	29	103.53K
MAPO$_{d/1m}$	90.37	7	2.50K	80.09	34	12.14K	57.04	—	—	72.46	—	—	37.76	—	—

over 100 clients under shard-based non-IID partitioning (5 classes and 500 samples per client), sampling 3 clients per round, with one local epoch, batch size 8, and SGD (lr 0.03, momentum 0.9); ViT-L/16 for 300 rounds over 10 IID clients with full participation, 10 local steps per round, batch size 32, and SGD (lr 0.03, gradient clipping 1.0); and Llama-3.2-3B for 100 rounds over 3 IID clients with full participation, 40 local steps per round, batch size 1 at sequence length 512, and SGD (lr 0.01). GSM8K accuracy is exact-match on the final numeric answer under greedy, zero-shot (no chain-of-thought) decoding with at most 96 generated tokens, evaluated on a fixed held-out subset of 128 problems. Table 6 reports the per-round uplink relative to FedAvg (Com.) and the final accuracy for a sweep of k.

Table 6: MAPO on larger models: per-round uplink relative to FedAvg (Com., %) and final accuracy (Acc., %).

ViT-B/16 · CIFAR-100			ViT-L/16 · CIFAR-100			Llama-3.2-3B · GSM8K		
k	Com.	Acc.	k	Com.	Acc.	k	Com.	Acc.
2^{16}	0.08	61.1	2^{19}	0.17	70.6	2^{22}	0.13	19.6
2^{17}	0.15	72.8	2^{20}	0.34	80.7	2^{23}	0.26	22.7
2^{18}	0.31	77.4	2^{21}	0.68	85.5	2^{24}	0.52	27.9
2^{19}	0.61	81.1	2^{22}	1.36	89.8	2^{25}	1.04	29.2
2^{20}	1.22	83.1	2^{23}	2.72	90.2	2^{26}	2.08	30.2
2^{21}	2.44	85.8	2^{24}	5.44	91.4	2^{27}	4.16	31.7
FedAvg	100	86.2	FedAvg	100	93.9	FedAvg	100	32.6

With about 2–5% of FedAvg's uplink, MAPO comes within 0.4 (ViT-B), 2.5 (ViT-L), and 0.9 (Llama) points of FedAvg, and accuracy increases monotonically with k in all three sweeps, in the direction suggested by the d/k factor in Theorem 4.5, although that bound concerns gradient norms rather than accuracy.

D Comparison with Factorized-FL

In this section, we present a detailed comparison between MAPO and Factorized-FL as a representative parameter-decomposition method. Factorized-FL can be interpreted as a variant of rank-1 LoRA, in which a sparse bias matrix replaces LoRA's frozen, fine-tuned weights, initialized to zero. Table 7 reports the communication efficiency of MAPO and Factorized-FL on CIFAR-10 and SVHN (Netzer et al., 2011) datasets, evaluated under both IID and non-IID partitions. Each column denotes the total communication in GB required to reach X% of FedAvg's final test accuracy. Results show that MAPO achieves significantly lower communication costs compared to Factorized-FL while maintaining competitive performance across both datasets and data distributions.

D.1 Comparison with a federated GaLore variant under matched uplink

GaLore (Zhao et al., 2024) builds data-dependent, per-layer SVD subspaces and primarily targets optimizer-state memory. A federated adaptation must additionally specify how client-specific subspaces are represented, transmitted, or aligned before aggregation. We implement *GaLore-FL*, a federated realization of GaLore in which each client transmits its data-dependent rank-r representa-

Table 7: Communication cost comparison across different methods on SVHN and CIFAR-10 under IID and Non-IID settings.

Method	SVHN				CIFAR-10				Com/Round
	IID@80%	IID@90%	Non-IID@80%	Non-IID@90%	IID@80%	IID@90%	Non-IID@80%	Non-IID@90%	
FedAvg	183.51	244.68	285.46	509.75	305.85	407.80	326.24	652.48	20.39GB
Factorized-FL	127.75	182.50	146.00	219.00	182.50	292.00	200.75	310.25	18.25GB
MAPO_{2k}	0.32	0.79	0.56	–	0.32	–	0.94	–	**0.78MB**
MAPO_{16k}	**0.08**	**0.18**	**0.12**	**0.27**	**0.08**	**0.18**	**0.23**	**0.45**	6.25MB
MAPO_{40k}	3.84	8.64	5.76	13.12	3.84	8.64	10.88	21.12	0.32GB

tion (projector and coefficients), costing $r(d_1 + d_2)$ scalars per $d_1 \times d_2$ weight matrix, with r set to the largest value that fits the uplink cap. Even at $r = 1$, this construction transmits $\sum_i (d_1^i + d_2^i) = 17{,}811$ scalars per client and round on the CIFAR-100 WideResNet-16-4 (Zagoruyko & Komodakis, 2016) model ($d \approx 2.85$M), i.e., $2.17\times$ MAPO's $k = 8{,}192$ and about 0.6% of d; this is a construction-specific payload, not a lower bound for every GaLore-based FL algorithm.

Controlled setting. This ablation uses 10 IID clients with full participation and the same model, split, optimizer, rounds, local steps, and aggregation for both methods. It is lighter than the main 100-client, non-IID, 10%-participation setting, so absolute accuracies are not comparable to Table 2. WideResNet-16-4 values are the final global validation top-1 accuracy after 200 rounds; ViT-B/16 is pretrained, and we report its best validation top-1 accuracy after saturation.

Table 8: MAPO vs. GaLore-FL at matched uplink caps (% of FedAvg's per-round uplink). "Infeasible": even $r = 1$ exceeds the cap.

Dataset / model (FedAvg acc.)	Uplink cap	MAPO	GaLore-FL	Δ
CIFAR-100 / WRN-16-4 (67.0)	10%	**66.3**	64.5	+1.8
	2%	**55.1**	54.7	+0.4
	1%	**49.9**	48.6	+1.3
	0.2%	**48.6**	infeasible	–
	0.1%	**41.5**	infeasible	–
CIFAR-10 / WRN-16-4 (93.5)	10%	**91.9**	89.2	+2.7
	2%	**86.1**	85.5	+0.6
	1%	**83.8**	82.6	+1.2
	0.2%	**82.6**	infeasible	–
	0.1%	**80.6**	infeasible	–
CIFAR-100 / ViT-B/16 (94.7)	1%	**94.2**	92.1	+2.1
	0.1%	**93.6**	infeasible	–

At every budget GaLore-FL can reach, MAPO is higher by 0.4–2.7 points ($+1.44$ on average). GaLore-FL also has an integer rank floor because even $r = 1$ requires transmitting its data-dependent projector: about 0.6% of the uplink on WideResNet and 0.2% on ViT-B/16, whereas MAPO operates well below these floors.

E Implementation details and Hyperparameters

All experiments were conducted on a single NVIDIA RTX 3090 with 24 GB of memory. The main experiments and baselines were implemented with JAX (Bradbury et al., 2018). The GLUE tasks and LLM fine-tuning implementation use Hugging Face libraries and models (Wolf et al., 2020), implemented in FederatedScope (Xie et al., 2023), and operate in half-precision (i.e., 16-bit floats). The model configuration and training used in this work are provided in Tables 10 and 11.

Communication accounting. All communication values count uplink payload bits per participating client. Per-round values (e.g., Tables 6 and 12) are relative to FedAvg's per-round uplink, whereas the Com. column of Table 2 is the cumulative uplink needed to reach the worst-performing baseline's accuracy, relative to FedAvg's cumulative uplink to that accuracy. MAPO transmits its k coefficients in fp32; the reconstruction vector is regenerated from a shared 64-bit seed, so no basis is transmitted, and the server's downlink consists of the k values plus the seed. Sparse transmits the selected values

Table 9: Summary of datasets and models used in our experiments.

Dataset	Client Distribution	Train/Test	# Classes	Model	# Parameters
MNIST (LeCun et al., 1998)	Non-IID (2 classes)	60K / 10K	10	CNN - 2 Layers	11,274
FMNIST (Xiao et al., 2017)	Non-IID (2 classes)	60K / 10K	10	CNN - 2 Layers	11,274
CIFAR-10 (Krizhevsky, 2009)	Non-IID (2 classes)	50K / 10K	10	CNN - 4 Layers	1,146,634
CIFAR-100 (Krizhevsky, 2009)	Non-IID (10 classes)	50K / 10K	100	WideResNet-16-4	2,854,420
TinyImageNet (Le & Yang, 2015)	Non-IID (10 classes)	100K / 10K	200	WideResNet-16-4	2,880,120
Shakespeare (Caldas et al., 2018)	Distributed by Roles	14K / 2K	65	LSTM	814,957
Sentiment140 (Caldas et al., 2018)	Distributed by Users	1.4M / 200K	2	Transformer	2,221,570
GLUE Tasks (Wang et al., 2018a)	Non-IID	*differ per task*	*differ per task*	RoBERTa-Large	357,199,876

in fp32 together with their coordinate indices ($\lceil \log_2 d \rceil$ bits per index). Quantize transmits the chosen bit width plus fp32 scales. EvoFed and FedLoRU transmit their communicated parameters in fp32 (number of parameters $\times$ 32 bits). For RoBERTa, ViT, and Llama, all methods use fp16.

Table 10: Neural network configurations for different datasets.

Dataset	**Model type**	**# Conv**	**Kernel**	**Hidden features**	**# Linear**	**# Output**	**# Parameters**
MNIST	CNN	2	5×5	8, 16	1	10	11.3K
FMNIST	CNN	2	5×5	8, 16	1	10	11.3K
CIFAR-10	CNN	4	5×5	64, 64, 128, 128	2	10	1.1M
CIFAR-100	WideResNet	16	3×3	64×4, 128×4	2	100	2.8M
TinyImageNet	WideResNet	16	3×3	64×4, 128×4	2	200	2.88M
Shakespeare	LSTM	-	-	256, 8 (embed)	2	65	814K
Sentiment140	Transformer	-	-	512, 96 (embed)	2	2	2.2M
SVHN	CNN	4	5×5	64, 64, 128, 128	2	10	1.1M
GLUE	RoBERTa-large	-	-	1024 (hidden)	2	Varies	357M

Table 11: Training hyperparameters for FedAvg and variants.

Hyperparameter	**MNIST**	**FMNIST**	**CIFAR-10**	**CIFAR-100**	**TinyImageNet**	**Sentiment140**	**Shakespeare**	**SVHN**	**GLUE**
Batch size	32	32	32	32	32	32	32	32	128
Optimizer	SGD	SGD	SGD	AdamW	AdamW	SGD	SGD	SGD	SGD
Learning rate	0.2	0.2	0.03	0.1	0.2	0.001	0.2	0.03	0.02
Momentum / β_1 (AdamW)	0.9	0.9	0.4	0.9	0.9	0.9	0.9	0.4	0.0
L1 regularization	0.0	0.0	1e-4	0.0	1e-5	0.0	5e-6	1e-4	0.0
L2 reg. / weight decay (AdamW)	0.0	0.0	1e-5	3e-3	1e-4	0.0	5e-5	1e-5	0.0

E.1 Guideline for tuning the MAPO's communication budget

To select an optimal value of k, we begin by evaluating MAPO under a high compression setting, typically starting around $k \approx \sqrt{d}$. We then increase k progressively by doubling its value at each step. After each evaluation, we compare both the final accuracy and the convergence rate. The search continues until we observe that increases in k no longer yield meaningful performance gains, and that the faster convergence no longer compensates for the higher communication cost. In practice, this means choosing the smallest k that achieves near-optimal accuracy while also minimizing the total number of communication rounds. For example, if $k = 128$ reaches 90 percent accuracy in 100 rounds and $k = 256$ reaches the same accuracy in 60 rounds, the choice of $k = 128$ remains preferable because the overall communication load is lower despite the slower convergence.

F Extreme Label Skew

The main non-IID setting assigns two classes per client on MNIST, FMNIST, and CIFAR-10, which is already severe. We further sweep Dirichlet label skew (Hsu et al., 2019) with concentration $\alpha \in \{0.5, 0.3, 0.1, 0.05\}$ and add a one-class-per-client endpoint, using the same partition for all methods at each level and reporting the final accuracy after 500 rounds (Table 12).

MAPO is the strongest communication-efficient method at every Dirichlet level and at the one-class endpoint (where, on CIFAR-10, all methods are close to chance). Across the 12 Dirichlet settings, MAPO's average deficit to FedAvg is 0.74 points and its largest deficit is 2.2 points (CIFAR-10,

Table 12: Final accuracy (%) under Dirichlet label skew and one class per client. The last column shows the per-round uplink as a percentage of FedAvg's uplink.

Dataset	Method	$\alpha = 0.5$	0.3	0.1	0.05	1 class	Uplink (%)
MNIST	FedAvg	98.8	98.8	98.5	98.0	10.0	100
	MAPO	**98.7**	**98.5**	**97.3**	**96.8**	**67.6**	≈2.3
	FedLoRU	84.6	82.0	32.1	9.8	9.8	
	EvoFed	87.3	85.9	78.6	65.5	67.3	
FMNIST	FedAvg	89.1	87.3	83.9	82.3	10.0	100
	MAPO	**87.5**	**86.5**	**82.3**	**81.9**	**55.4**	≈2.3
	FedLoRU	68.5	67.0	60.2	52.5	10.0	
	EvoFed	68.7	67.4	58.7	58.1	49.8	
CIFAR-10	FedAvg	60.3	56.5	48.7	40.5	13.3	100
	MAPO	**59.6**	**54.3**	**47.9**	**42.5**	**10.7**	≈0.7
	FedLoRU	31.4	27.8	10.0	10.0	10.0	
	EvoFed	10.2	10.1	10.6	10.2	10.1	

$\alpha = 0.3$), while it uses about 0.7–2.3% of FedAvg's per-round uplink; on CIFAR-10 with $\alpha = 0.05$ it exceeds FedAvg by 2.0 points. We do not claim higher raw accuracy than the full-communication FedAvg reference: MAPO remains close to it while substantially reducing uplink communication, and it outperforms the evaluated communication-efficient baselines at every tested heterogeneity level.

G Ablation Study: Subspace Exploration and Stability

Design. We compare three variants of MAPO based on how the reconstruction vector A is handled:
(1) FRESH-A, which regenerates a new random A_t every communication round;
(2) FROZEN-A, which keeps A fixed throughout training; and
(3) TRAINABLE-A, which optimizes A via gradient descent.

Subspace exploration. In theory, refreshing A_t expands the number of explored low-rank subspaces, which can accelerate optimization under a tight communication budget. Empirically, we observe the same behavior across all six benchmarks: FRESH-A may begin slightly slower but consistently surpasses FROZEN-A after the initial rounds and converges to substantially higher accuracy. This validates that exploration dominates early-round variance.

Trainable-A: communication limitations and instability. Optimizing A (Trainable-A) introduces a fundamental limitation: it removes the possibility of *communication-less synchronization* of the reconstruction vector. Unlike FRESH-A and FROZEN-A, which synchronize A using shared random seeds, TRAINABLE-A requires every client to communicate both a projected update and the trainable vector, leading to the following communication rate:

$$\text{COM}_{\text{Trainable-A}} = \frac{d}{k'} + k',$$

which is minimized when $k' = \sqrt{d}$, yielding a lower bound of $2\sqrt{d}$ per round.

In large models (e.g., GLUE with $d = 357\text{M}$ parameters), this corresponds to $2\sqrt{d} \approx 37{,}789$ scalars per round, far exceeding the communication cost of MAPO with $k \in \{d/10{,}000,\ d/100{,}000\} = \{35{,}700,\ 3{,}570\}$, both of which outperform other baselines. A similar gap appears in lightweight vision and text tasks. For instance:

$$\text{COM}^{\text{Trainable-A}}_{\text{FMNIST}} = 213, \quad \text{COM}^{\text{Trainable-A}}_{\text{Shakespeare}} = 1806,$$

while MAPO requires less to match FedAvg's performance.

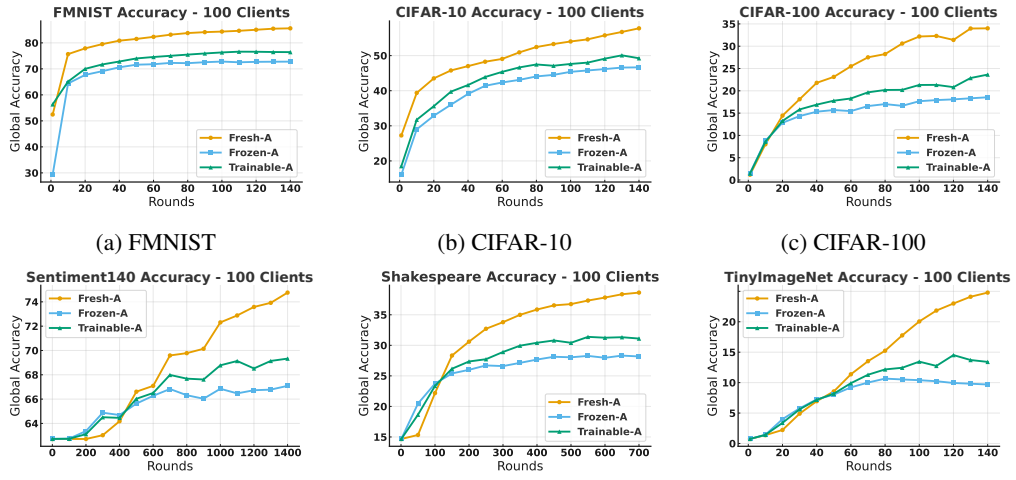
$$\text{COM}_{\text{FMNIST}} = 32, \quad \text{COM}_{\text{Shakespeare}} = 256.$$

Although this additional communication could, in principle, improve convergence, our experiments show the opposite. Trainable-A often converges faster *locally* but fails to converge *globally* due to the **low-rank averaging problem**: clients' optimized A_t vectors diverge in incompatible directions,

and the global aggregation collapses into unstable or oscillatory updates. Even when stabilizing aggregation using *full-rank* averaging at the server and transmitting the full model back to clients, the final accuracy remains far below the FRESH-A variant, and it is typically close to FROZEN-A, despite requiring much higher communication, as shown in Figure 9.

Interpretation. We hypothesize that regenerating A_t promotes more robust exploration of diverse low-rank subspaces, especially under tight communication budgets, than gradient-based optimization of A. While this behavior makes MAPO effective, it also highlights a limitation of parameter-factorization methods such as LoRA, where the underlying model parameters are fixed and do not permit A-regeneration. The broader implications for both gradient and parameter factorization remain open for future investigation.

A fixed, small subspace quickly becomes capacity-limited. Allowing A to refresh each round enhances subspace exploration, resulting in reliably higher accuracies after the initial transient. In our implementation, A is (re)initialized *once per communication round*. Each round comprises many local optimization steps (multiple batches/epochs), executed *within a fixed subspace*. The subspace changes only after server aggregation. This design avoids the instability that could arise from re-drawing A at *every* local step.



(a) FMNIST (b) CIFAR-10 (c) CIFAR-100

(d) Sentiment140 (e) Shakespeare (f) TinyImageNet

Figure 9: Accuracy over communication rounds for six benchmarks under Fresh-A, Frozen-A, and Trainable-A. Trainable-A consistently stays between the two, closer to Frozen-A, and fails to reach Fresh-A's performance.

G.1 SENSITIVITY TO RANDOM SEEDS.

Across all of our experiments, MAPO shows very low sensitivity to the choice of random seed, as reflected in the standard deviations reported in Table 2. Since MAPO regenerates a new reconstruction basis A_t at every communication round, the effect of any individual "lucky" or "unlucky" random subspace is averaged out over the course of training. Across repeated trials with different seeds, the observed variance remains small and is comparable to the inherent stochasticity of FedAvg. This aligns with our intuition that exploring many random subspaces prevents the training trajectory from being dominated by a single reconstruction vector.

To further isolate the role of the random seed, we conducted an expanded study on *Frozen-A*. This condition exposes the direct influence of the random seed on optimization dynamics. In this setting, we observe a seed-dependent pattern:

- For **large subspaces** (for example, $k \geq d/2$), a fixed A behaves similarly to full-rank training. Most seeds converge to comparable accuracies, although a small fraction of seeds underperform due to the omission of useful directions.
- For **small subspaces** (higher compression ratios), Frozen-A MAPO reliably converges to a suboptimal local minimum. The seed-to-seed variance becomes noticeably larger than in the Fresh-A case.
- In every such setting, **Fresh-A MAPO removes the majority of this variance** and delivers stable performance across seeds.

These findings align with the theoretical picture: large subspaces induce small projection error, so seeds matter mainly when the sampled subspace excludes important directions. Smaller subspaces induce larger projection error, so a fixed subspace can trap optimization in seed-dependent minima. MAPO mitigates this issue by randomizing the projection basis across rounds, thereby averaging out the variability and stabilizing convergence. Table 13 illustrates the observed pattern for 10 random seeds on the FMNIST dataset.

Table 13: Seed sensitivity of Fresh-A versus Frozen-A MAPO on FMNIST.

Method	k/d	Std	Min	Max
Fresh-A	$1/2$	0.0021	0.8838	0.8900
Frozen-A	$1/2$	0.0898	0.6720	0.8992
Fresh-A	$1/100$	0.0035	0.8733	0.8836
Frozen-A	$1/100$	0.0115	0.7389	0.7711

H Proof of Definitions and Propositions

H.1 Proof of Lemma

For completeness, we restate Definition 3.1 and Lemma 3.2 from the main paper before giving the proof.

Definition H.1 (Communication Overhead and Reconstruction Error Rates). *Let $\Delta W_i \in \mathbb{R}^{d_1 \times d_2}$ and $\Delta W_i \approx B_i A_i$, where $A_i \in \mathbb{R}^{q \times d_2}$ is fixed, $B_i \in \mathbb{R}^{d_1 \times q}$ is trainable, and $q \leq d_2$, with $B_i^\star \in \arg\min_{B \in \mathbb{R}^{d_1 \times q}} \|\Delta W_i - B A_i\|_F^2$. The **communication overhead rate** and **reconstruction error rate** are*

$$\mathrm{CO}_{\mathrm{rate}} := \frac{|B_i|}{|\Delta W_i|} = \frac{d_1 q}{d_1 d_2} = \frac{q}{d_2}, \qquad \mathrm{RE}_{\mathrm{rate}}(A_i) := \frac{\|\Delta W_i - B_i^\star A_i\|_F^2}{\|\Delta W_i\|_F^2}.$$

Lemma H.2 (Expected Reconstruction Error of a Gaussian Projection). *Let A_i be a random matrix with entries drawn independently from $\mathcal{N}(0, \sigma^2)$ with $\sigma > 0$. Then A_i is almost surely of full rank, and for any fixed nonzero ΔW_i,*

$$\mathbb{E}_{A_i}[\mathrm{RE}_{\mathrm{rate}}(A_i)] = 1 - \frac{q}{d_2}.$$

Proof. Since $\sigma > 0$, A_i has a density on $\mathbb{R}^{q \times d_2}$, so it has full row rank q almost surely. Write ΔW_i row-wise with rows $\Delta w_r \in \mathbb{R}^{1 \times d_2}$, $r = 1, \ldots, d_1$, and B with rows $b_r \in \mathbb{R}^{1 \times q}$, so that

$$\|\Delta W_i - B A_i\|_F^2 = \sum_{r=1}^{d_1} \|\Delta w_r - b_r A_i\|_2^2.$$

The rows decouple. Row r is minimized by $b_r^\star = \Delta w_r A_i^\top (A_i A_i^\top)^{-1}$, so $b_r^\star A_i = \Delta w_r P_{A_i}$, where $P_{A_i} = A_i^\top (A_i A_i^\top)^{-1} A_i$ is the orthogonal projector onto the row space of A_i. By the Pythagorean theorem,

$$\|\Delta W_i - B_i^\star A_i\|_F^2 = \sum_{r=1}^{d_1} \big(\|\Delta w_r\|_2^2 - \Delta w_r P_{A_i} \Delta w_r^\top \big).$$

The law of A_i is invariant under $A_i \mapsto A_i Q$ for every orthogonal Q, and $P_{A_i Q} = Q^\top P_{A_i} Q$. Hence $\mathbb{E}[P_{A_i}]$ commutes with every orthogonal matrix and is a multiple of I_{d_2}; since $\operatorname{tr} P_{A_i} = q$ almost surely, $\mathbb{E}[P_{A_i}] = \frac{q}{d_2} I_{d_2}$. Therefore $\mathbb{E}_{A_i}[\Delta w_r P_{A_i} \Delta w_r^\top] = \frac{q}{d_2} \|\Delta w_r\|_2^2$ for every row, including zero rows, and

$$\mathbb{E}_{A_i}\big[\|\Delta W_i - B_i^\star A_i\|_F^2\big] = \Big(1 - \frac{q}{d_2}\Big) \|\Delta W_i\|_F^2.$$

Dividing by $\|\Delta W_i\|_F^2 > 0$ gives the claim. No per-row normalization is needed.

□

H.2 Proof of Propositions

Proposition H.3 (Single-Vector Factorization). *Let ΔW_i, A_i, and B_i be factorizations of a single layer of the network as in Definition 3.1. By reshaping ΔW_i into $\Delta W_i' \in \mathbb{R}^{1 \times d_1 d_2}$, the factorization $\Delta W_i' \approx B_i' A_i'$ where $A_i' \in \mathbb{R}^{p \times d_1 d_2}$ and $B_i' \in \mathbb{R}^{1 \times p}$ achieves the same expected **reconstruction error** rate and **communication overhead** as the conventional factorization of ΔW_i when $p = q d_1$.*

Proof of Error Preservation. In the single-vector setup, $\Delta W_i' \in \mathbb{R}^{1 \times d_1 d_2}$ is projected onto a uniformly random subspace of dimension p spanned by the rows of A_i'. By Lemma 3.2 applied with ambient dimension $d_1 d_2$ and rank p,

$$\mathbb{E}_{A_i'}\left[\frac{\|\Delta W_i' - B_i'^\star A_i'\|_2^2}{\|\Delta W_i'\|_2^2} \right] \;=\; 1 - \frac{p}{d_1 d_2}.$$

Substituting $p = q d_1$ gives

$$1 - \frac{q d_1}{d_1 d_2} \;=\; 1 - \frac{q}{d_2},$$

so the expected reconstruction error satisfies

$$\mathbb{E}_{A'_i}\left[\|\Delta W'_i - B'^{\star}_i A'_i\|_2^2\right] \;=\; \left(1 - \frac{q}{d_2}\right)\|\Delta W'_i\|_2^2,$$

which matches the per-layer error rate of the original factorization. □

Proof of Communication Preservation. The reshaped update has $\mathrm{size}(\Delta W'_i) = d_1 d_2$ and the trainable factor satisfies

$$\mathrm{size}(B'_i) \;=\; p \;=\; q d_1,$$

so the communication overhead rate of the single-vector factorization is

$$\mathrm{CO}'_{\mathrm{rate}} \;=\; \frac{\mathrm{size}(B'_i)}{\mathrm{size}(\Delta W'_i)} \;=\; \frac{q d_1}{d_1 d_2} \;=\; \frac{q}{d_2},$$

which matches the original overhead in Definition 3.1.

Since both the expected reconstruction error and the communication overhead are unchanged, the single-vector factorization with $p = q d_1$ is equivalent to the conventional factorization in terms of efficiency. □

Proposition H.4 (Multi-Layer Factorization). *Let ΔW_i, A_i, and B_i be **single-vector factorization** of i-th layer of the N-layer network as in Proposition 3.3. By concatenating the reshaped updates ΔW_i into $\Delta W' \in \mathbb{R}^{1\times d}$, where $d = \sum_{i=1}^N d_1^i d_2^i$, the factorization $\Delta W' \approx B'A'$ with $A' \in \mathbb{R}^{p\times d}$ and $B' \in \mathbb{R}^{1\times p}$ achieves the same size-weighted **aggregate reconstruction error rate**, $\sum_i (d_1^i d_2^i/d)(1 - q/d_2^i) = 1 - p/d$, and **aggregate communication overhead** as the single-vector factorization applied to each ΔW_i when $p = \sum_{i=1}^N q\, d_1^i$. The energy-weighted layer-wise error $\sum_i (1 - q/d_2^i)\|\Delta W_i\|_F^2/\|\Delta W\|_2^2$ also equals $1 - p/d$ if all d_2^i are equal or $\|\Delta W_i\|_F^2 \propto d_1^i d_2^i$, but not in general.*

Proof of Error Preservation. For each layer i, a random full-rank matrix $A_i \in \mathbb{R}^{q\times d_2^i}$ yields an expected squared reconstruction error

$$\mathbb{E}\Big[\|\Delta W_i - B_i^{\star} A_i\|_F^2\Big] \;=\; \Big(1 \;-\; \frac{q}{d_2^i}\Big)\|\Delta W_i\|_F^2.$$

Flattening ΔW_i into $\Delta W'_i \in \mathbb{R}^{(d_1^i d_2^i)\times 1}$, a single-vector projection of dimension $q\, d_1^i$ preserves this same error ratio (cf. Proposition 3.3).

When we concatenate all $\Delta W'_i$ into $\Delta W' \in \mathbb{R}^{1\times d}$, we form a block-structured vector. Let $p := \sum_{i=1}^N q\, d_1^i$ and let $A' \in \mathbb{R}^{p\times d}$ be drawn from a Gaussian distribution. By the standard random-projection argument in dimension d with subspace size p,

$$\frac{\mathbb{E}\Big[\|\Delta W' - B'^{\star} A'\|_2^2\Big]}{\|\Delta W'\|_2^2} \;=\; 1 - \frac{p}{d} \;=\; 1 - \frac{\sum_{i=1}^N q\, d_1^i}{\sum_{i=1}^N d_1^i d_2^i}.$$

This aggregate ratio matches the (size-weighted) average of the per-layer ratios $1 - q/d_2^i$ obtained from applying single-vector factorizations of rank q to each $\Delta W'_i$ individually, since $\sum_i \frac{d_1^i d_2^i}{d}\big(1 - \frac{q}{d_2^i}\big) = 1 - \frac{p}{d}$; it equals the energy-weighted aggregate error $\sum_i (1 - \frac{q}{d_2^i})\|\Delta W_i\|_F^2/\|\Delta W'\|_2^2$ of the layer-wise scheme whenever all layers share the same d_2^i (e.g., the equal-width case of Proposition 3.5) or $\|\Delta W_i\|_F^2 \propto d_1^i d_2^i$. □

Proof of Communication Preservation. For each layer i, the single-vector factorization of ΔW_i introduces

$$\mathrm{size}(B_i) \;=\; q\, d_1^i, \quad \mathrm{size}(\Delta W_i) \;=\; d_1^i\, d_2^i, \quad \text{hence} \quad \frac{\mathrm{size}(B_i)}{\mathrm{size}(\Delta W_i)} \;=\; \frac{q}{d_2^i}.$$

Concatenating all $\Delta W_i'$ into $\Delta W' \in \mathbb{R}^{1\times d}$ gives $\text{size}(\Delta W') = d = \sum_{i=1}^{N} d_1^i d_2^i$. Meanwhile, the new trainable vector $B' \in \mathbb{R}^{1\times p}$ has

$$\text{size}(B') \;=\; p \;=\; \sum_{i=1}^{N} q\, d_1^i.$$

Thus

$$\frac{\text{size}(B')}{\text{size}(\Delta W')} \;=\; \frac{\sum_{i=1}^{N} q\, d_1^i}{\sum_{i=1}^{N} d_1^i\, d_2^i},$$

which matches the *aggregate* overhead of N individual rank-q factorizations (one per layer): the total number of transmitted parameters $\sum_i q d_1^i$ is identical, so the aggregate communication overhead rate is preserved.

Since both the aggregate expected reconstruction error and aggregate communication overhead remain the same, choosing $p = \sum_i q\, d_1^i$ for $\Delta W'$ is equivalent in aggregate to applying single-vector factorization of rank q separately to each layer. □

Proposition H.5 (**MAPO Factorization**). *Let ΔW, A, B, and rank p be a **multi-layer factorization** as in Proposition 3.4. Assume $k \mid d$. By reshaping $\Delta W \in \mathbb{R}^{1\times d}$ into $\Delta W' \in \mathbb{R}^{k\times \lceil d/k\rceil}$, and the factorization $\Delta W' \approx B'A'$ where $A' \in \mathbb{R}^{1\times \lceil d/k\rceil}$ and $B' \in \mathbb{R}^{k\times 1}$, we can achieve the same expected **reconstruction error** rate, $1 - p/d$, and **communication overhead**, p/d, when $k = p$, with around **half** of the layer-wise parameters.*

Proof of Error Preservation. Since $k \mid d$, there is no padding and $n := \lceil d/k\rceil = d/k$. Reshaping is an isometry, so $\|\Delta W'\|_F^2 = \|\Delta W\|_2^2$. Row r of $\Delta W'$, $\Delta w_r' \in \mathbb{R}^{1\times n}$ ($r = 1, \ldots, k$), is approximated by $b_r' A'$, so the rows decouple, and the optimal $B'^{\star}$ projects every row onto the same randomly sampled one-dimensional subspace $\text{span}(A')$. These projections are dependent across rows, but expectations add. By Lemma 3.2 with $q = 1$ and ambient dimension n, applied to each row,

$$\mathbb{E}\big[\|\Delta w_r' - b_r'^{\star} A'\|_2^2\big] = \Big(1 - \frac{1}{n}\Big)\|\Delta w_r'\|_2^2.$$

Summing over the k rows gives

$$\frac{\mathbb{E}\big[\|\Delta W' - B'^{\star} A'\|_F^2\big]}{\|\Delta W'\|_F^2} = 1 - \frac{1}{n} = 1 - \frac{k}{d} = 1 - \frac{p}{d},$$

where the last equality uses $k = p$. This is exactly the multi-layer error rate.

□

Proof of Communication Preservation. The matrix $B' \in \mathbb{R}^{k\times 1}$ has size k in total. Meanwhile, $\Delta W' \in \mathbb{R}^{k\times \lceil d/k\rceil}$ has size $k \times \lceil d/k\rceil = d$ (since $k \mid d$). Thus

$$\frac{\text{size}(B')}{\text{size}(\Delta W')} \;=\; \frac{k}{\lceil d/k\rceil\, k} \;=\; \frac{k}{d} = \frac{p}{d}.$$

Setting $k = p$ matches the original ratio $\frac{p}{d}$ from $B \in \mathbb{R}^{p\times 1}$ in the multi-layer factorization. □

Remark H.6 (**When $k \nmid d$**). *If $k \nmid d$, the $kn - d$ padded entries are discarded by $\text{vec}(\cdot)_{[0:d]}$, so row r holds only $n_r \le n$ real coordinates, with $\sum_r n_r = d$; with a row-major fill, at most one row is partial and trailing rows may be empty. The optimal coefficient of row r uses only the corresponding n_r entries of A', and the argument above in dimension n_r gives*

$$\mathbb{E}\big[\|\Delta W - \text{vec}(B'^{\star} A')_{[0:d]}\|_2^2\big] = \sum_{r:\, n_r \ge 1} \Big(1 - \frac{1}{n_r}\Big)\|\Delta w_r'\|_2^2 \le \Big(1 - \frac{1}{n}\Big)\|\Delta W\|_2^2.$$

Thus the expected error rate is at most $1 - 1/\lceil d/k\rceil \approx 1 - k/d$, the communication rate is k/d, and Proposition 3.5 holds up to this rounding.

H.3 PARAMETER REDUCTION RATE

Proof of Parameter Reduction (around half of the layer-wise total). We compare the total number of factorization parameters $|A| + |B|$ between layer-wise factorization and MAPO under the matched communication budget $k = p = \sum_{i=1}^{N} q\, d_1^i$.

Layer-wise total. For each layer i with $A_i \in I\!R^{q \times d_2^i}$ and $B_i \in I\!R^{d_1^i \times q}$,

$$|A_i| + |B_i| \;=\; q\, d_2^i + q\, d_1^i.$$

Summing across all N layers,

$$|A_{\text{layer}}| + |B_{\text{layer}}| \;=\; q \sum_{i=1}^{N} d_2^i \;+\; q \sum_{i=1}^{N} d_1^i \;=\; q \sum_{i=1}^{N} d_2^i \;+\; k,$$

where the last equality uses $k = \sum_i q\, d_1^i$ from Proposition 3.4.

MAPO total. With $A' \in I\!R^{1 \times \lceil d/k \rceil}$ and $B' \in I\!R^{k \times 1}$,

$$|A_{\text{MAPO}}| + |B_{\text{MAPO}}| \;=\; \lceil d/k \rceil + k.$$

Reduction in the reconstruction factor A. The ratio of the two reconstruction-factor sizes is

$$\frac{|A_{\text{layer}}|}{|A_{\text{MAPO}}|} \;=\; \frac{q \sum_{i=1}^{N} d_2^i}{\lceil d/k \rceil} \;\approx\; \frac{k \sum_{i=1}^{N} q\, d_2^i}{d}.$$

In the common equal-width case where $d_1^i = d_2^i = h$ and $q_i = q$ for all i, we have $k = Nqh$ and $d = Nh^2$, so $\sum_i q\, d_2^i = Nqh = k$, and

$$\frac{|A_{\text{layer}}|}{|A_{\text{MAPO}}|} \;\approx\; \frac{k^2}{d} \;=\; Nq^2.$$

That is, MAPO's reconstruction factor A is approximately Nq^2 times smaller than its layer-wise counterpart.

Total parameter count. Since $|B|$ is matched by construction ($|B_{\text{layer}}| = q \sum_i d_1^i = k = |B_{\text{MAPO}}|$), the parameter saving comes entirely from A. For arbitrary widths under a common rank q, the exact ratio is

$$\frac{|A_{\text{MAPO}}| + |B_{\text{MAPO}}|}{|A_{\text{layer}}| + |B_{\text{layer}}|} \;=\; \frac{k + \lceil d/k \rceil}{k + q \sum_{i=1}^{N} d_2^i}.$$

In the equal-width case, $|A_{\text{layer}}| = k$ and $|A_{\text{MAPO}}| = \lceil d/k \rceil = h/q$, hence

$$\frac{|A_{\text{MAPO}}| + |B_{\text{MAPO}}|}{|A_{\text{layer}}| + |B_{\text{layer}}|} \;=\; \frac{h/q + k}{k + k} \;=\; \frac{1}{2}\left(1 + \frac{1}{Nq^2}\right) \;\xrightarrow[Nq^2 \to \infty]{}\; \frac{1}{2}.$$

Thus, MAPO retains the reconstruction error and communication overhead of the layer-wise factorization while using approximately **half** of its parameters, with the saving concentrated in the reconstruction factor A (an $Nq^2 \approx k^2/d$ reduction). □

I CONVERGENCE ANALYSIS PROOF

This appendix proves Theorem 4.5, states and proves Corollary I.8 (Appendices I.1 and I.2), characterizes the effect of frozen versus fresh reconstruction vectors (Appendix I.3), and establishes explicit relations between the budget k and the per-round reconstruction residual ϵ_t (Appendix I.4).

Let $\{\mathcal{L}^j\}_{j=1}^{M}$ denote the client objectives and $\mathcal{L}(W) := \frac{1}{M} \sum_{j=1}^{M} \mathcal{L}^j(W)$ the global objective. Let W^t be the global model at the beginning of communication round $t \in \{0, 1, \dots\}$, and let $E \in \mathbb{N}$ denote the number of local steps per round (the analysis below takes $E = 1$).

For completeness, we restate Assumptions 4.1 to 4.4 using the same labels.

Assumption I.1 (restatement of Assumption 4.1)**.** *For each j, $\mathcal{L}^j$ is β-smooth, i.e., $\|\nabla\mathcal{L}^j(u) - \nabla\mathcal{L}^j(v)\| \le \beta\|u-v\|$ for any u, v, and $\mathcal{L}$ is bounded below by $\mathcal{L}^*$.*

Assumption I.2 (restatement of Assumption 4.2)**.** *For each client j, the minibatch gradient $\widetilde{\nabla}\mathcal{L}^j(W)$ is unbiased and has bounded variance:*

$$\mathbb{E}\Big[\|\nabla\mathcal{L}^j(W) - \widetilde{\nabla}\mathcal{L}^j(W)\|^2\Big] \le \sigma_l^2.$$

Assumption I.3 (restatement of Assumption 4.3)**.** *Client gradients have bounded dissimilarity:*

$$\frac{1}{M}\sum_{j=1}^{M}\|\nabla\mathcal{L}^j(W) - \nabla\mathcal{L}(W)\|^2 \le \sigma_g^2.$$

Assumption I.4 (restatement of Assumption 4.4)**.** *At each round t, the server samples a subset $S_t \subset [M]$ of $|S_t| = m < M$ clients uniformly without replacement. The resulting sampling variance is $\sigma_{het}^2 = \rho\sigma_g^2$, where $\rho = \frac{M-m}{m(M-1)}$.*

I.1 Setting and notation

Padding and reshaping. Let $n := \lceil d/k \rceil$. For $v \in \mathbb{R}^d$, $\mathcal{R}(v) \in \mathbb{R}^{k\times n}$ zero-pads v to length kn and arranges its entries in the same order used by $\mathrm{vec}(\cdot)_{[0:d]}$ in Algorithm 1. Thus, for every $Z \in \mathbb{R}^{k\times n}$, $\mathrm{vec}(Z)_{[0:d]} = \mathcal{R}^*(Z)$, where $\mathcal{R}^*$ is the adjoint of $\mathcal{R}$ under the Euclidean and Frobenius inner products: $\langle \mathcal{R}v, Z\rangle_F = \langle v, \mathcal{R}^*Z\rangle$. The adjoint reshapes back and removes padded entries, so $\mathcal{R}^*\mathcal{R} = I_d$. Row $i \in [k]$ of $\mathcal{R}(v)$ defines the i-th *segment*, containing $n_i \le n$ non-padded coordinates. The segments have disjoint supports and satisfy $\sum_{i=1}^{k} n_i = d$. When $k \mid d$, all segments have size $n_i = n$. We write $v_i \in \mathbb{R}^{n_i}$ for segment i, so $\|v\|^2 = \sum_i \|v_i\|^2$. For $A \in \mathbb{R}^{1\times n}$, let $a := A^\top \in \mathbb{R}^n$ and let $a_{(i)} \in \mathbb{R}^{n_i}$ denote its entries corresponding to the non-padded positions of segment i. The results below hold for any fixed assignment of coordinates to segments satisfying $n_i \le n$.

Randomness. Let $\mathcal{F}_t$ be the σ-algebra generated by all randomness before round t, so W^t is $\mathcal{F}_t$-measurable, and define $\mathbb{E}_t[\cdot] := \mathbb{E}[\cdot \mid \mathcal{F}_t]$. During round t, the server samples S_t, clients draw minibatches, and a reconstruction vector A^t is generated from a fresh seed. Let $\mathcal{F}_t^-$ additionally include S_t and the round-t minibatches. We assume $A^t \sim \mathcal{N}(0, \sigma^2 I_n)$ is independent of $\mathcal{F}_t^-$ (idealizing the seeded pseudo-random generator as an exact, independent Gaussian draw), and that minibatches are freshly sampled, independently across clients, independently of S_t, and independently of $\mathcal{F}_t$.

One MAPO round. We analyze one local step ($E = 1$) of plain SGD with step size η, initialized at $B^{t,j} = \mathbf{0}$, without clipping or regularization. We use uniform aggregation weights $b^j \equiv 1$ in Equation (2), matching the uniform objective $\mathcal{L} = \frac{1}{M}\sum_j \mathcal{L}^j$. Client $j \in S_t$ uses the minibatch gradient $\tilde{g}^{t,j} := \widetilde{\nabla}\mathcal{L}^j(W^t)$, and we define $\tilde{g}^t := \frac{1}{m}\sum_{j\in S_t}\tilde{g}^{t,j}$.

Lemma I.5 (**Exact MAPO step**)**.** *In the setting above,*

$$W^{t+1} = W^t - \eta\, U_{A^t}(\tilde{g}^t), \qquad U_A(g) := \mathcal{R}^*\big((\mathcal{R}g)A^\top A\big).$$

Moreover, segment i of $U_A(g)$ is $(g_i^\top a_{(i)})a_{(i)}$ for every $g \in \mathbb{R}^d$.

Proof. Let $\Phi^j(B) := \widetilde{\mathcal{L}}^j(W^t + \mathcal{R}^*(BA^t))$, where $\widetilde{\mathcal{L}}^j$ is client j's minibatch loss in round t. For any direction $\Theta \in \mathbb{R}^{k\times 1}$, the chain rule and adjoint relation give

$$D\Phi^j(B)[\Theta] = \langle \nabla\widetilde{\mathcal{L}}^j(\cdot), \mathcal{R}^*(\Theta A^t)\rangle = \langle (\mathcal{R}\nabla\widetilde{\mathcal{L}}^j(\cdot))(A^t)^\top, \Theta\rangle.$$

Hence, $\nabla\Phi^j(\mathbf{0}) = (\mathcal{R}\tilde{g}^{t,j})(A^t)^\top$, and one SGD step gives $\hat{B}^{t,j} = -\eta(\mathcal{R}\tilde{g}^{t,j})(A^t)^\top$. Using $b^j \equiv 1$ and linearity of $\mathcal{R}$,

$$\overline{B}^t = \frac{1}{m}\sum_{j\in S_t}\hat{B}^{t,j} = -\eta(\mathcal{R}\tilde{g}^t)(A^t)^\top.$$

Substituting this into $W^{t+1} = W^t + \mathcal{R}^*(\overline{B}^t A^t)$ proves the first claim. For segment i, entry i of $(\mathcal{R}g)A^\top$ equals $g_i^\top a_{(i)}$, since padded entries are zero. Therefore, row i of $(\mathcal{R}g)A^\top A$ is $(g_i^\top a_{(i)})a^\top$, and $\mathcal{R}^*$ retains its non-padded coordinates, yielding $(g_i^\top a_{(i)})a_{(i)}$. □

Lemma I.6 (Aggregated stochastic gradient). *Under Assumptions 4.2 to 4.4,*

$$\mathbb{E}_t[\tilde{g}^t] = \nabla\mathcal{L}(W^t), \qquad \mathbb{E}_t\|\tilde{g}^t - \nabla\mathcal{L}(W^t)\|^2 \le \Sigma^2 := \sigma_l^2/m + \sigma_{het}^2.$$

If minibatch noise is correlated across clients, the bound instead holds with $\sigma_l^2 + \sigma_{het}^2$ by Jensen's inequality.

Proof. Write $\tilde{g}^t - \nabla\mathcal{L}(W^t) = \xi + \zeta$, where

$$\xi := \frac{1}{m}\sum_{j\in S_t}\big(\tilde{g}^{t,j} - \nabla\mathcal{L}^j(W^t)\big), \qquad \zeta := \frac{1}{m}\sum_{j\in S_t}\nabla\mathcal{L}^j(W^t) - \nabla\mathcal{L}(W^t).$$

Conditioned on $\mathcal{F}_t$ and S_t, the terms in ξ are independent and zero-mean. Thus, $\mathbb{E}_t[\xi] = 0$, $\mathbb{E}_t\langle\xi,\zeta\rangle = 0$, and $\mathbb{E}_t\|\xi\|^2 \le \sigma_l^2/m$ by Assumption 4.2. Uniform sampling without replacement gives $\mathbb{E}_t[\zeta] = 0$ and

$$\mathbb{E}_t\|\zeta\|^2 = \rho\frac{1}{M}\sum_{j=1}^{M}\|\nabla\mathcal{L}^j(W^t) - \nabla\mathcal{L}(W^t)\|^2 \le \rho\sigma_g^2 = \sigma_{\text{het}}^2$$

by Assumptions 4.3 and 4.4. Combining the two bounds proves the claim. □

Lemma I.7 (Gaussian moments). *(a) For fixed $x \in \mathbb{R}^{n'}$ and $z \sim \mathcal{N}(0, \sigma^2 I_{n'})$,*

$$\mathbb{E}\big[(x^\top z)z\big] = \sigma^2 x, \qquad \mathbb{E}\big\|(x^\top z)z\big\|^2 = \sigma^4(n'+2)\|x\|^2.$$

(b) Consequently, for fixed $g \in \mathbb{R}^d$ and $A^\top \sim \mathcal{N}(0, \sigma^2 I_n)$,

$$\mathbb{E}_A[U_A(g)] = \sigma^2 g, \qquad \mathbb{E}_A\|U_A(g)\|^2 = \sigma^4\sum_{i=1}^{k}(n_i+2)\|g_i\|^2 \le \sigma^4(n+2)\|g\|^2.$$

Proof. (a) Since $\mathbb{E}[zz^\top] = \sigma^2 I$, $\mathbb{E}[(x^\top z)z] = \sigma^2 x$. For the second moment, $\|(x^\top z)z\|^2 = (x^\top z)^2\|z\|^2$. By rotational invariance, take $x = \|x\|e_1$. Then

$$\mathbb{E}[(x^\top z)^2\|z\|^2] = \|x\|^2\Big(\mathbb{E}[z_1^4] + \sum_{c\neq 1}\mathbb{E}[z_1^2]\mathbb{E}[z_c^2]\Big) = \|x\|^2\big(3\sigma^4 + (n'-1)\sigma^4\big) = \sigma^4(n'+2)\|x\|^2.$$

(b) By Lemma I.5, segment i of $U_A(g)$ is $(g_i^\top a_{(i)})a_{(i)}$, where $a_{(i)} \sim \mathcal{N}(0, \sigma^2 I_{n_i})$. Applying part (a) with $x = g_i$ to each segment gives the mean. Since the segments have disjoint supports, $\|U_A(g)\|^2$ is the sum of their squared norms, and linearity of expectation gives the second identity (no independence between segments is needed). Finally, $n_i \le n$ yields the stated bound. Empty segments contribute nothing. □

I.2 Proof of Theorem 4.5 and Corollary I.8

We restate Theorem 4.5 for convenience.

> *Let Assumptions 4.1 to 4.4 hold in the setting of Appendix I.1, and let $\Sigma^2 := \sigma_l^2/m + \sigma_{het}^2$ and $\gamma := \eta\sigma^2$. If $\gamma \le \frac{1}{\beta(n+2)}$ with $n = \lceil d/k\rceil$, then for every $T \ge 1$,*
>
> $$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big[\|\nabla\mathcal{L}(W^t)\|^2\big] \le \frac{2(\mathcal{L}(W^0) - \mathcal{L}^*)}{\gamma T} + \beta(n+2)\gamma\Sigma^2.$$

Proof of Theorem 4.5. Let $U^t := U_{A^t}(\tilde{g}^t)$, so $W^{t+1} = W^t - \eta U^t$ by Lemma I.5.

Step 1 (conditional moments). Since $\tilde{g}^t$ is $\mathcal{F}_t^-$-measurable and A^t is independent of $\mathcal{F}_t^-$, conditioning first on $\mathcal{F}_t^-$ and applying Lemma I.7(b), followed by Lemma I.6, gives

$$\mathbb{E}_t[U^t] = \sigma^2\mathbb{E}_t[\tilde{g}^t] = \sigma^2\nabla\mathcal{L}(W^t), \tag{4}$$

$$\begin{aligned}\mathbb{E}_t\|U^t\|^2 &\le \sigma^4(n+2)\mathbb{E}_t\|\tilde{g}^t\|^2 \\ &\le \sigma^4(n+2)\big(\|\nabla\mathcal{L}(W^t)\|^2 + \Sigma^2\big).\end{aligned} \tag{5}$$

Thus, conditioned on the current iterate, the MAPO update is a scaled-unbiased stochastic gradient, even though a single A^t captures only a $1/n$ fraction of the gradient energy in expectation (Proposition I.11).

Step 2 (descent). Since $\mathcal{L}$ is β-smooth as the average of β-smooth functions, Equations (4) and (5) and $\gamma = \eta\sigma^2$ yield

$$\begin{aligned}\mathbb{E}_t[\mathcal{L}(W^{t+1})] &\le \mathcal{L}(W^t) - \eta\left\langle \nabla\mathcal{L}(W^t), \mathbb{E}_t[U^t]\right\rangle + \frac{\beta\eta^2}{2}\mathbb{E}_t\|U^t\|^2 \\ &\le \mathcal{L}(W^t) - \left(\gamma - \frac{\beta(n+2)\gamma^2}{2}\right)\|\nabla\mathcal{L}(W^t)\|^2 + \frac{\beta(n+2)\gamma^2}{2}\Sigma^2.\end{aligned}$$

Step 3 (step size). If $\gamma \le \frac{1}{\beta(n+2)}$, then $\frac{\beta(n+2)\gamma^2}{2} \le \frac{\gamma}{2}$. Therefore,

$$\mathbb{E}_t[\mathcal{L}(W^{t+1})] \le \mathcal{L}(W^t) - \frac{\gamma}{2}\|\nabla\mathcal{L}(W^t)\|^2 + \frac{\beta(n+2)\gamma^2}{2}\Sigma^2. \tag{6}$$

This step-size condition is feasible for every $k \ge 1$ and $\sigma > 0$. Budgets $k > d$ only add padding ($n = 1$) and give no compression, so we consider $k \in \{1, \dots, d\}$.

Step 4 (telescoping). Taking total expectations in Equation (6), summing over $t = 0, \dots, T-1$, and using $\mathbb{E}[\mathcal{L}(W^T)] \ge \mathcal{L}^*$ and the deterministic initialization W^0 gives

$$\frac{\gamma}{2}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla\mathcal{L}(W^t)\|^2 \le \mathcal{L}(W^0) - \mathcal{L}^* + \frac{T\beta(n+2)\gamma^2}{2}\Sigma^2.$$

Dividing by $\gamma T/2$ proves the theorem. □

We now state precisely the rates discussed after Theorem 4.5 in Section 4.

Corollary I.8 (Rates and stationarity). *Let the conditions of Theorem 4.5 hold and let $\Delta_0 := \mathcal{L}(W^0) - \mathcal{L}^*$. (i) If $\sigma_l = \sigma_g = 0$, then $\gamma = \frac{1}{\beta(n+2)}$ gives the bound $2\beta(n+2)\Delta_0/T$. (ii) With $\gamma_\star := \sqrt{2\Delta_0/(T\beta(n+2)\Sigma^2)}$ and $\gamma := \min\{\gamma_\star, \frac{1}{\beta(n+2)}\}$, the bound is at most $2\sqrt{2\beta(n+2)\Delta_0\Sigma^2/T} + 4\beta(n+2)\Delta_0/T = O\big(\frac{n}{T} + \sqrt{\frac{n}{T}}\big)$. (iii) For deterministic steps $\gamma_t = \eta_t\sigma^2 \in (0, \frac{1}{\beta(n+2)}]$ with $\sum_t \gamma_t = \infty$ and $\sum_t \gamma_t^2 < \infty$, $\sum_{t<T}\gamma_t\,\mathbb{E}\|\nabla\mathcal{L}(W^t)\|^2 \big/ \sum_{t<T}\gamma_t \to 0$; equivalently, an iterate W^τ drawn with $\Pr(\tau = t) \propto \gamma_t$, $t < T$, satisfies $\mathbb{E}\|\nabla\mathcal{L}(W^\tau)\|^2 \to 0$.*

Proof of Corollary I.8. Let $\Delta_0 := \mathcal{L}(W^0) - \mathcal{L}^*$. (i) If $\Sigma = 0$ and $\gamma = \frac{1}{\beta(n+2)}$, Theorem 4.5 gives $2\beta(n+2)\Delta_0/T$, with $n + 2 \le d/k + 3$. (ii) Let

$$\gamma_\star := \sqrt{\frac{2\Delta_0}{T\beta(n+2)\Sigma^2}}, \qquad \gamma := \min\left\{\gamma_\star, \frac{1}{\beta(n+2)}\right\}.$$

If $\gamma = \gamma_\star$, the bound becomes $2\sqrt{2\beta(n+2)\Delta_0\Sigma^2/T}$. Otherwise, $\frac{1}{\beta(n+2)} < \gamma_\star$, which implies $\Sigma^2 < 2\beta(n+2)\Delta_0/T$, and the bound is

$$2\beta(n+2)\Delta_0/T + \Sigma^2 < 4\beta(n+2)\Delta_0/T.$$

Thus, the stated rate holds in both cases. (iii) Applying Equation (6) round by round with deterministic $\gamma_t = \eta_t\sigma^2 \le \frac{1}{\beta(n+2)}$ gives

$$\frac{\sum_{t<T}\gamma_t\,\mathbb{E}\|\nabla\mathcal{L}(W^t)\|^2}{\sum_{t<T}\gamma_t} \le \frac{2\Delta_0 + \beta(n+2)\Sigma^2\sum_{t<T}\gamma_t^2}{\sum_{t<T}\gamma_t} \xrightarrow[T\to\infty]{} 0,$$

because $\sum_t \gamma_t^2 < \infty$ while $\sum_t \gamma_t = \infty$. The left-hand side equals $\mathbb{E}\|\nabla\mathcal{L}(W^\tau)\|^2$ when τ is sampled independently with $\Pr(\tau = t) \propto \gamma_t$ for $t < T$. □

Remark I.9 (**Scope of the guarantee**). *(a)* Aggregation weights. *The analysis assumes uniform weights, matching the uniform objective. Data-size weights $b^j = |\mathcal{D}^j|$ under uniform client sampling are not covered because the resulting weighted average is a biased ratio estimator. If clients are instead sampled i.i.d. with probabilities $p_j \propto |\mathcal{D}^j|$ and averaged uniformly, the same argument applies to $\mathcal{L}_p = \sum_j p_j \mathcal{L}^j$, with Assumption 4.3 stated for $\mathcal{L}_p$ and ρ replaced by $1/m$. (b)* Several local steps. *For $E > 1$, all local iterates remain in the round-specific subspace $W^t + \mathrm{range}(B \mapsto \mathcal{R}^*(BA^t))$, and the first step is exactly the one analyzed above. Controlling the drift of subsequent local steps requires standard client-drift arguments, which we leave for future work, together with momentum, clipping, and regularization. (c)* Scale of A. *The bound depends on A only through $\gamma = \eta\sigma^2$. For example, choosing $\sigma^2 = 1/n$ gives $\mathbb{E}\|A^t\|^2 = 1$. (d)* Interpretation. *The operator $U_A(\cdot)/\sigma^2$ is an unbiased compressor whose second moment is inflated by at most $n + 2$. Thus, Theorem 4.5 recovers the standard behavior of SGD with unbiased compression and guarantees convergence for every budget k. MAPO's communication advantage over alternative compressors is established empirically rather than by this bound.*

I.3 Frozen versus fresh reconstruction vectors

For a fixed $A \in \mathbb{R}^{1\times n}$, define $J_A : \mathbb{R}^k \to \mathbb{R}^d$ by $J_A b := \mathcal{R}^*(bA)$, and let $V := \mathrm{range}(J_A)$. Denote the orthogonal projector onto V by P_V. From the segment form in Lemma I.5, $J_A b$ contains $b_i a_{(i)}$ in segment i, while $J_A^\top x = (a_{(i)}^\top x_i)_{i=1}^k$.

Proposition I.10 (**Frozen-subspace floor and fresh coverage**). *Let $A \sim \mathcal{N}(0, \sigma^2 I_n)$ be sampled once and fixed across rounds (*FROZEN-A*). Almost surely, $\dim V$ equals the number of nonempty segments and therefore equals k whenever every segment contains at least one real coordinate, including when $k \mid d$.*

(i) ***Trapping.*** *For any update rule of the form $W^{t+1} = W^t + J_A \Delta b^t$, every iterate remains in $W^0 + V$, regardless of how Δb^t is generated. First-order stationarity of $b \mapsto \mathcal{L}(W^0 + J_A b)$ is equivalent to $P_V \nabla \mathcal{L} = 0$, which does not require the full gradient to vanish.*

(ii) ***Exact floor.*** *Let $k \mid d$ (so $n = d/k$), $\mathcal{L}(W) = \frac{1}{2}\|W - W^\star\|^2$, and $\Delta := W^0 - W^\star$. For every frozen-basis rule as in (i) and every t,*

$$\|\nabla \mathcal{L}(W^t)\|^2 \geq \|(I - P_V)\Delta\|^2 = \sum_{i=1}^{k} \Big(\|\Delta_i\|^2 - (\Delta_i^\top \hat{a})^2 \Big), \qquad \hat{a} := a/\|a\|.$$

Therefore,

$$\mathbb{E}_A \|(I - P_V)\Delta\|^2 = \left(1 - \frac{1}{n}\right) \|\Delta\|^2 = \left(1 - \frac{k}{d}\right) \|W^0 - W^\star\|^2.$$

This floor is attained asymptotically by gradient descent on b with step size $\alpha/\|a\|^2$ for any $\alpha \in (0, 2)$.

(iii) ***Fresh directions remove the obstruction.*** *If $A^0, \ldots, A^{T-1}$ are i.i.d. $\mathcal{N}(0, \sigma^2 I_n)$ (*FRESH-A*) with subspaces $V_t := \mathrm{range}(J_{A^t})$, then almost surely*

$$\dim\Big(\sum_{t<T} V_t \Big) = \sum_{i=1}^{k} \min(n_i, T),$$

which equals d once $T \geq \max_i n_i$ (at most $\lceil d/k \rceil$ rounds). When $k \mid d$, this dimension is $\min(d, kT)$, and moreover

$$\mathbb{E}[P_{V_t}] = \frac{1}{n} I_d = \frac{k}{d} I_d,$$

so every fixed direction has an expected captured fraction of k/d in each round.

Part (iii) characterizes the directions made available by fresh reconstruction vectors; it does not imply that all directions contribute equally to optimization. Convergence under fresh directions is established by Theorem 4.5.

Proof. The vectors $a_{(i)}$ corresponding to nonempty segments are nonzero almost surely and have disjoint supports. They are therefore linearly independent and span V, establishing the dimension statement.

(i) By induction, $W^t - W^0 = \sum_{s<t} J_A \Delta b^s \in V$. For $\varphi(b) := \mathcal{L}(W^0 + J_A b)$,

$$\nabla\varphi(b) = J_A^\top \nabla\mathcal{L}(W^0 + J_A b).$$

Thus, $\nabla\varphi(b) = 0$ if and only if $\nabla\mathcal{L} \perp V$, equivalently $P_V \nabla\mathcal{L} = 0$.

(ii) Since $\nabla\mathcal{L}(W^t) = W^t - W^\star = (W^t - W^0) + \Delta$ and $W^t - W^0 \in V$,

$$(I - P_V)\nabla\mathcal{L}(W^t) = (I - P_V)\Delta.$$

Pythagoras gives the lower bound. When $k \mid d$, the vectors containing $\hat{a}$ in one segment and zeros elsewhere form an orthonormal basis of V. Thus, P_V maps segment i of x to $(x_i^\top \hat{a})\hat{a}$, yielding the displayed expression. Because $\hat{a}$ is uniform on the unit sphere of $\mathbb{R}^n$, $\mathbb{E}[\hat{a}\hat{a}^\top] = I_n/n$, so $\mathbb{E}(\Delta_i^\top \hat{a})^2 = \|\Delta_i\|^2/n$. Summing over segments gives the expected floor. For attainability, run gradient descent $b^{s+1} = b^s - \frac{\alpha}{\|a\|^2} J_A^\top \nabla\mathcal{L}(W^s)$ with $W^s = W^0 + J_A b^s$. Since all segments are full, $J_A J_A^\top = \|a\|^2 P_V$, and therefore

$$W^{s+1} - W^\star = (I - \alpha P_V)(W^s - W^\star).$$

The V component contracts by $|1 - \alpha| < 1$, while the $V^\perp$ component remains unchanged. Therefore, $\|\nabla\mathcal{L}(W^s)\|^2 \to \|(I - P_V)\Delta\|^2$.

(iii) The space $\sum_{t<T} V_t$ is spanned by vectors containing $a_{(i)}^t$ in segment i, for all $t < T$ and $i \in [k]$. Within segment i, $a_{(i)}^0, \ldots, a_{(i)}^{T-1}$ are i.i.d. nondegenerate Gaussian vectors in $\mathbb{R}^{n_i}$ and therefore span $\min(n_i, T)$ dimensions almost surely. Since segments have disjoint supports, these dimensions add:

$$\dim\Big(\sum_{t<T} V_t\Big) = \sum_i \min(n_i, T).$$

This equals d once $T \geq \max_i n_i$, with $\max_i n_i \leq n = \lceil d/k \rceil$. If $k \mid d$, then $n_i = n$ and the dimension becomes $\min(d, kT)$; moreover, P_{V_t} then acts on each segment as $\hat{a}^t(\hat{a}^t)^\top$, whose expectation is I_n/n. □

I.4 Explicit relation between k and the reconstruction residual

For a nonzero gradient g and reconstruction vector A, with $a = A^\top$, the best reconstruction of segment i along a is its orthogonal projection onto $\mathrm{span}(a)$. For a full segment ($n_i = n$), define the captured fraction and residual as

$$Z_i := \frac{(g_i^\top a)^2}{\|g_i\|^2 \|a\|^2}, \qquad \epsilon_i := 1 - Z_i.$$

For the full model, let $G := \mathcal{R}g$ and define

$$\bar{Z} := \frac{a^\top H a}{\|a\|^2 \operatorname{tr} H}, \qquad H := G^\top G \in \mathbb{R}^{n\times n},$$

with $\bar{Z} := 1/n$ when $G = 0$. Here, ϵ_i is the reconstruction error rate of Definition 3.1 with $q = 1$ and $d_2 = n$, while $\bar{Z}$ is the fraction of $\|G\|_F^2$ captured by the best rank-one reconstruction BA for the sampled A. For $H \neq 0$, define its effective rank as

$$r_{\text{eff}}(H) := \frac{(\operatorname{tr} H)^2}{\operatorname{tr}(H^2)} \in [1, \min(k, n)].$$

Proposition I.11 (**k-to-ϵ relation**). *Let $n = \lceil d/k \rceil$ and $A \sim \mathcal{N}(0, \sigma^2 I_n)$ be independent of g.*

(i) ***Per segment.*** *For a full segment with $g_i \neq 0$ and $n \geq 2$, conditionally on g, $Z_i \sim \mathrm{Beta}(\frac{1}{2}, \frac{n-1}{2})$, and hence*

$$\mathbb{E}[\epsilon_i \mid g] = 1 - \frac{1}{n} \approx 1 - \frac{k}{d}.$$

For $\delta \in (0,1)$, let κ_δ be the δ-quantile of $\mathrm{Beta}(\frac{1}{2}, \frac{n-1}{2})$. *Then*

$$\Pr(\epsilon_i \le 1 - \kappa_\delta \mid g) = 1 - \delta.$$

If $n = 1$, then $Z_i = 1$ and $\epsilon_i = 0$. For a partial segment ($1 \le n_i < n$), the same holds with a replaced by $a_{(i)}$ and n by n_i, so $\mathbb{E}[\epsilon_i \mid g] = 1 - 1/n_i$.

(ii) ***Whole model.*** *Conditionally on g, $\mathbb{E}[\bar{Z} \mid g] = \frac{1}{n}$ and, for $G \neq 0$,*

$$\mathrm{Var}(\bar{Z} \mid g) = \frac{2}{n(n+2)} \left(\frac{1}{r_{\mathrm{eff}}(H)} - \frac{1}{n} \right).$$

Hence, for every $s > 0$,

$$\Pr\left(\left| \bar{Z} - \frac{1}{n} \right| \ge s \,\middle|\, g \right) \le \frac{2}{n(n+2)s^2} \left(\frac{1}{r_{\mathrm{eff}}(H)} - \frac{1}{n} \right).$$

(iii) ***Across rounds.*** *Let $\bar{Z}_t$ be computed using $g = \tilde{g}^t$ and $A = A^t$, and define the whole-model residual $\epsilon_t := 1 - \bar{Z}_t$. Then $\mathbb{E}[\bar{Z}_t \mid \mathcal{F}_t^-] = 1/n$, and $(\bar{Z}_t - \frac{1}{n})_t$ forms a martingale-difference sequence with values in an interval of length 1. Therefore, for every $s > 0$ (Hoeffding, 1963; Azuma, 1967),*

$$\Pr\left(\left| \frac{1}{T} \sum_{t<T} \left(\epsilon_t - \left(1 - \frac{1}{n} \right) \right) \right| \ge s \right) \le 2e^{-2Ts^2}.$$

Although the gradients adapt to the optimization trajectory, and the residuals are therefore neither constant nor i.i.d. across rounds, their running average remains within s of $1 - 1/n$ with probability at least $1 - 2e^{-2Ts^2}$. These results are diagnostic only: Theorem 4.5 relies on the exact moments in Lemma I.7 and does not require ϵ_t to be small. A residual close to 1 is therefore an expected consequence of strong compression, rather than a bias or feasibility failure.

Proof. All ratios are invariant to the scale of a, so $\hat{a} := a/\|a\|$ is uniform on the unit sphere of $\mathbb{R}^n$ and independent of g.

(i) Rotating g_i to $\|g_i\| e_1$ gives

$$Z_i = \hat{a}_1^2 = \frac{z_1^2}{\|z\|^2}, \qquad z \sim \mathcal{N}(0, I_n).$$

Since $z_1^2 \sim \chi_1^2$ and $\sum_{c \ge 2} z_c^2 \sim \chi_{n-1}^2$ are independent,

$$Z_i \sim \mathrm{Beta}\left(\frac{1}{2}, \frac{n-1}{2} \right)$$

with mean $1/n$. The quantile statement follows from continuity of the distribution.

(ii) Since $\bar{Z} = \hat{a}^\top H \hat{a} / \operatorname{tr} H$ and $\mathbb{E}[\hat{a}\hat{a}^\top] = I_n/n$, we have $\mathbb{E}[\bar{Z} \mid g] = 1/n$. For the variance, write $z = \|z\|\hat{a}$ with $z \sim \mathcal{N}(0, I_n)$, where $\|z\|$ and $\hat{a}$ are independent. Then $\mathbb{E}[(z^\top H z)^2] = (\operatorname{tr} H)^2 + 2\operatorname{tr}(H^2)$ and $\mathbb{E}\|z\|^4 = n(n+2)$, so

$$\mathbb{E}\big[(\hat{a}^\top H \hat{a})^2\big] = \frac{(\operatorname{tr} H)^2 + 2\operatorname{tr}(H^2)}{n(n+2)}.$$

Dividing by $(\operatorname{tr} H)^2$ and subtracting $1/n^2$,

$$\mathrm{Var}(\bar{Z} \mid g) = \frac{1 + 2/r_{\mathrm{eff}}(H)}{n(n+2)} - \frac{1}{n^2} = \frac{2}{n(n+2)} \left(\frac{1}{r_{\mathrm{eff}}(H)} - \frac{1}{n} \right).$$

Chebyshev's inequality gives the stated tail bound. When $r_{\mathrm{eff}} = 1$, this reduces to $\frac{2(n-1)}{n^2(n+2)}$, which matches the variance of $\mathrm{Beta}(\frac{1}{2}, \frac{n-1}{2})$ in part (i).

(iii) Since $\tilde{g}^t$ is $\mathcal{F}_t^-$-measurable and A^t is independent of $\mathcal{F}_t^-$, part (ii) gives

$$\mathbb{E}[\bar{Z}_t \mid \mathcal{F}_t^-] = \frac{1}{n}.$$

Moreover, $\bar{Z}_t$ is $\mathcal{F}_{t+1}^{-}$-measurable and lies in $[0,1]$, so $D_t := \bar{Z}_t - \frac{1}{n}$ is a martingale-difference sequence with increments contained in an interval of length 1. The Azuma–Hoeffding inequality therefore gives

$$\Pr\left(\left|\sum_{t<T} D_t\right| \geq Ts\right) \leq 2e^{-2Ts^2}.$$

Since $\epsilon_t - (1 - \frac{1}{n}) = -D_t$, the result follows. □

J Complexity Analysis and MAPO Flexibility

Propositions 3.3 to 3.5 established that the factorization error of MAPO depends only on the size of the projection vector, regardless of how layers are reshaped or vectorized. In this section, we provide a detailed computational and memory analysis that complements the theoretical results in the main text. We show that MAPO not only matches the factorization accuracy of layer-wise methods under the same communication budget, but also achieves substantially lower computation and memory overhead in practice. We further compare MAPO with EvoFed and FedLoRU to highlight the efficiency advantages and flexibility of our design.

Baseline Matrix Multiplication Costs. We first recall standard matrix multiplication costs. Given matrices $A \in I\!R^{n\times m}$ and $B \in I\!R^{p\times n}$, the cost of computing $C = BA \in I\!R^{p\times m}$ and the memory needed to store all three matrices is

$$\text{Time}_{BA} = O(mnp), \quad \text{Memory}_{B+A+C} = O(nm + pn + pm).$$

In what follows, we measure overhead relative to a standard forward and backward pass of the base model. Let d denote the total number of model parameters (viewing each parameter tensor as a $d_1^i \times d_2^i$ matrix),

$$d = \sum_{i=1}^{N} d_1^i d_2^i,$$

and let b denote the batch size. Assuming a standard forward and backward pass costs approximately $6bd$ FLOPs, which is linear in d, scaled by the batch size b.

J.1 Computational Complexity

MAPO. In MAPO, the full parameter vector is reshaped into a matrix $W \in I\!R^{k\times\lceil d/k\rceil}$. We factorize W using a reconstruction vector $A \in I\!R^{1\times\lceil d/k\rceil}$ and a projection vector $B \in I\!R^{k\times 1}$. Both projection and reconstruction reduce to a single matrix product with cost:

$$O\big(\lceil d/k\rceil \times k \times 1\big) \approx O(d).$$

Each matrix-vector multiplication requires approximately one multiplication and one addition per element, hence the total cost for projection and reconstruction is $4d$ FLOPs per optimization step. The relative computational overhead of MAPO is therefore

$$\frac{4d}{6bd} = \frac{2}{3b}.$$

This quantity does not depend on the model dimension d. For example, for $b = 32$ the overhead is about 2.1% and it decreases for larger batch sizes.

EvoFed. In EvoFed, the full model update is partitioned into p parts, and each is factorized separately with population rank q. We can write it as $W \in \mathbb{R}^{p\times\lceil d/p\rceil}$ factorized by a reconstruction vector $A \in \mathbb{R}^{q\times\lceil d/p\rceil}$ and a projection vector $B \in \mathbb{R}^{p\times q}$. Both projection and reconstruction have a cost:

$$O(\lceil d/p\rceil \times q \times p) = O(dq),$$

which is applied once per round rather than once per batch. If each client performs e local epochs per round, the relative overhead becomes

$$\frac{4dq}{6bed} = \frac{2q}{3be}.$$

which is negligible for realistic hyperparameter values. For instance, if $b = 32, e = 200, p = \frac{d}{10}$, and $q = 10$ the overhead is around 0.1%. The overhead decreases with larger b and e but increases with the population size q, and is independent of the number of partitions p.

FedLoRU. In FedLoRU, low-rank updates are performed layer-wise. Consider the i-th layer with parameter matrix $W_i \in \mathbb{R}^{d_1^i \times d_2^i}$. FedLoRU introduces a low-rank factorization with rank q, using matrices

$$A_i \in \mathbb{R}^{q \times d_2^i}, \qquad B_i \in \mathbb{R}^{d_1^i \times q}.$$

The corresponding layer-wise update involves matrix products of the form $B_i A_i$, which have cost

$$O\big(d_1^i \times q \times d_2^i\big) = O\big(q d_1^i d_2^i\big).$$

Summing over all layers, the total extra cost is

$$O\Big(q \sum_{i=1}^{N} d_1^i d_2^i\Big) = O(qd).$$

The total number of additional FLOPs per batch due to FedLoRU is roughly $4qd$. Relative to the baseline cost $6bd$, the overhead is

$$\frac{4qd}{6bd} = \frac{2q}{3b}.$$

This quantity is independent of the model dimension d, but grows linearly with the FedLoRU rank q. For example, with $q = 4$ and $b = 32$, the overhead is about

$$\frac{2 \cdot 4}{3 \cdot 32} \approx 8.3\%.$$

Thus, for the same batch size and error tolerance, MAPO achieves a factor-of-q reduction in computational overhead compared to FedLoRU.

J.2 Empirical Wall-Clock Measurements

All experiments were executed on a single NVIDIA RTX 3090 GPU with 24 GB of memory. Table 14 reports the average client-side time per round (in seconds) over the full training process for FedAvg, MAPO, EvoFed, and FedLoRU. The measured overheads are fully consistent with the theoretical analysis above.

Table 14: Average client-side wall-clock time per round (in seconds) and relative overhead compared to FedAvg.

Dataset	Batch	FedAvg	MAPO	Over	EvoFed	Over	FedLoRU	Over
MNIST	32	5.277	5.347	1.33%	5.280	0.06%	5.439	3.07%
FMNIST	32	5.277	5.356	1.50%	5.283	0.11%	5.468	3.62%
CIFAR-10	32	47.393	48.173	1.64%	47.435	0.09%	50.143	5.80%
CIFAR-100	32	67.705	68.750	1.54%	67.724	0.03%	71.139	5.07%
TinyImageNet	32	117.965	119.409	1.22%	118.044	0.06%	130.517	10.64%
Shakespeare	32	0.634	0.642	1.26%	0.635	0.16%	0.656	3.47%
Sentiment140	32	0.304	0.308	1.40%	0.305	0.33%	0.314	3.29%
SVHN	32	32.478	32.961	1.49%	32.490	0.04%	34.061	4.87%
GLUE	128	127.835	128.924	0.85%	128.056	0.17%	146.989	14.98%

Across all datasets, the overhead of MAPO remains at or below two percent, EvoFed stays within a fraction of a percent, and FedLoRU incurs a significantly higher cost, especially on larger models such as TinyImageNet and GLUE. These measurements confirm that the additional computation required by MAPO is negligible in practice and does not offset its communication benefits.

J.3 FACTOR-STORAGE MEMORY OF MAPO, EVOFED, AND FEDLORU

We count *factor storage*: one unit per stored factor entry, plus one per entry of a trainable factor (its gradient), relative to the $2d$ units of the model parameters and their gradients. Buffers of size $O(d)$ are discussed at the end of this subsection.

MAPO. MAPO introduces two additional vectors, $A \in \mathbb{R}^{1\times\lceil d/k\rceil}$ and $B \in \mathbb{R}^{k\times 1}$. Only B is trained, so the factor storage is $\lceil d/k\rceil + 2k$ and the relative overhead is

$$\frac{\lceil d/k\rceil + 2k}{2d} \approx \frac{1}{2k} + \frac{k}{d}.$$

For a typical choice $k = d/100$, this becomes $\frac{50}{d} + \frac{1}{100}$, which is slightly above 1% for a realistic model size.

EvoFed. In EvoFed, the matrix $W \in \mathbb{R}^{p\times\lceil d/p\rceil}$ is factorized by a reconstruction vector $A \in \mathbb{R}^{q\times\lceil d/p\rceil}$ and a projection vector $B \in \mathbb{R}^{p\times q}$. EvoFed compresses the update after local training, so neither factor is trained and no factor gradients are stored; the factor storage is

$$q(\frac{d}{p} + p)$$

and the relative memory overhead will be

$$\frac{q}{2}(\frac{1}{p} + \frac{p}{d}).$$

Given the practical values of $q = 10, p = \frac{d}{10}$ we get

$$5(\frac{10}{d} + \frac{1}{10}),$$

which is around 50% for a realistic model size.

FedLoRU. For FedLoRU, each layer i allocates additional low-rank matrices A_i and B_i with total parameter count

$$qd_1^i + qd_2^i.$$

Under assumptions of $d_1^i = d_2^i = d_i$, and a fixed communication budget, MAPO and FedLoRU can be related through

$$k = q\sum_{i=1}^{N} d_i, \qquad d = \sum_{i=1}^{N}(d_i)^2.$$

Both factors are trained, so, using these relations, the factor storage (factors and their gradients) is

$$\text{Memory}_{\text{FedLoRU}} = 2(q\sum_{i=1}^{N} d_i + q\sum_{i=1}^{N} d_i) = 4k.$$

Therefore, the relative memory overhead of FedLoRU is

$$\frac{4k}{2d} = \frac{2k}{d}.$$

For the same choice of $k = d/100$, it becomes 2% overhead.

Reconstructed-update buffer. Factor storage excludes buffers of size $O(d)$. In a direct implementation, MAPO forms the effective weights $W^t + \text{vec}(B^{t,j}A^t)_{[0:d]}$ while keeping W^t, which adds one $O(d)$ buffer (about one extra copy of the model) during local training; the relative overheads above do not include it.

J.4 MAPO FLEXIBILITY

Suppose our neural network has N layers. Let:

$$W_i \in \mathbb{R}^{d_1^i \times d_2^i} \quad \text{for each layer } i = 1, \ldots, N.$$

Let $d = \sum_{i=1}^{N} d_1^i d_2^i$ be the total number of parameters (i.e., the sum of the entries across all layers). Let

$$d_1 \;=\; \sum_{i=1}^{N} d_1^i.$$

In many treatments of LoRA, the main communication or factor-size bottleneck arises from a factor that scales linearly with $q \cdot d_1^i$.

LoRA Factorization Per Layer. LoRA factorizes each layer W_i of dimension $d_1^i \times d_2^i$ with a fixed rank q. Concretely,

$$W_i' \;=\; W_i + B_i A_i, \qquad A_i \in \mathbb{R}^{q \times d_2^i}, \quad B_i \in \mathbb{R}^{d_1^i \times q}.$$

The number of additional parameters introduced by each low-rank pair (A_i, B_i) is

$$\underbrace{d_1^i \cdot q}_{\text{size of } B_i} \;+\; \underbrace{q \cdot d_2^i}_{\text{size of } A_i} \;=\; q\,(\,d_1^i + d_2^i).$$

Summing over all N layers,

$$\sum_{i=1}^{N} \big(d_1^i \cdot q + q \cdot d_2^i\big) \;=\; q \sum_{i=1}^{N} \big(d_1^i + d_2^i\big).$$

When both factors are trained and communicated, as in LoRA, the per-round communication is this full count, $q \sum_{i=1}^{N} (d_1^i + d_2^i)$. Variants that freeze A_i (e.g., FFA-LoRA) communicate only B_i, at cost $q \sum_{i=1}^{N} d_1^i = q\, d_1$. Since q must be an integer, the communication comes in integer multiples of a fixed, architecture-dependent unit:

$$\text{LoRA: } \Big\{\, q \sum_{i=1}^{N} \big(d_1^i + d_2^i\big) \;\Big|\; q = 1, 2, \ldots \Big\}, \qquad \text{fixed-}A\text{: } \big\{\, q\, d_1 \;\big|\; q = 1, 2, \ldots \big\}.$$

There is no way to select a non-integer q. Hence, even with A_i fixed, budgets strictly between $q\, d_1$ and $(q+1)d_1$ (e.g., $1.5\, d_1$) are not possible in layer-wise factorization, and the gaps are larger for LoRA. This **rigidity** is precisely what we seek to overcome in MAPO.

MAPO Factorization. MAPO flattens or reshapes all parameters into one large matrix and then performs a single low-rank factorization with rank 1. A simplified abstraction is:

1. Reshape $W_1, \ldots, W_N$ into a single matrix $W \in \mathbb{R}^{k \times \lceil d/k \rceil}$, where $d = \sum_{i=1}^{N} d_1^i\, d_2^i$ is the total parameter count. 2. Factor $W \approx B\, A$, with

$$A \in \mathbb{R}^{1 \times \lceil d/k \rceil}, \quad B \in \mathbb{R}^{k \times 1},$$

Once all parameters are merged, MAPO can proportionally allocate any communication budget since k can be chosen freely.

$$\underbrace{\lceil d/k \rceil}_{\text{size of } A} \;+\; \underbrace{k}_{\text{size of } B}.$$

Therefore, we can write the total communication as:

$$\text{MAPO total communication} \;\in\; \{\, k \;\mid\; k = 1, 2, \ldots, d\}.$$

This is particularly important in communication-efficient FL since viable solutions can be found with communication cost $k < d_1$ or $d_1 < k < 2d_1$, which architecture-dependent layer-wise factorization cannot offer.