LoRA Fails under Non-IID Conditions: Rethinking Federated Low-Rank Adaptation

Anonymous authors
Paper under double-blind review

Abstract

Low-Rank Adaptation (LoRA) has become a popular technique for memory-efficient fine-tuning of large models and has recently been adopted in federated learning (FL) due to its reduced parameter footprint. However, we show that LoRA significantly underperforms full-parameter fine-tuning (FFT) in FL, especially under non-IID client distributions. Our neural tangent kernel (NTK) analysis points to a simple cause: non-IID shifts diversify and misalign client gradients, increasing the effective rank (spectral energy) of the NTK / gradient-Gram matrix. Because LoRA commits to a fixed low-rank subspace, it cannot capture this additional structure; the induced kernel deviates and its spectral floor drops, leading to slower convergence and weaker generalization. Based on this finding, we argue that low-rank compression methods—such as GaLore—are inherently better suited for FL than low-rank reparameterization. Motivated by this insight, we propose `FedLore`. On the client side, `FedLore` uses a GaLore-style optimizer while replacing SVD with randomized SVD to reduce computational overhead. On the server side, `FedLore` estimates a shared low-rank gradient from client updates and broadcasts it to configure each client's GaLore projector, aligning update subspaces and mitigating drift under heterogeneity. Across NLU, vision, `FedLore` consistently achieves higher accuracy and robustness under non-IID conditions than LoRA-based strategies, while using comparable or less memory.

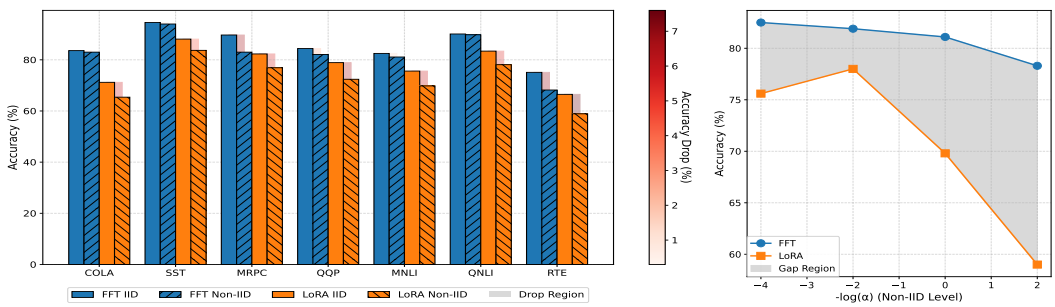
1 Introduction

Foundation Models (FMs) such as GPT, LLaMA, and SAM (Bommasani et al., 2021; Brown et al., 2020; Touvron et al., 2023; Kirillov et al., 2023) have transformed machine learning through the pretrain–finetune paradigm. However, fully finetuning these ever-growing models (Kaplan et al., 2020) is often computationally infeasible. To address this challenge, parameter-efficient finetuning (PEFT) methods, most notably Low-Rank Adaptation (LoRA) (Hu et al., 2022), have emerged. LoRA attains performance comparable to full finetuning while drastically reducing the number of trainable parameters and lowering peak GPU memory usage by decomposing weight updates into low-rank matrices (e.g., $\boldsymbol{A}$ and $\boldsymbol{B}$) while keeping pretrained weights frozen.

Building on these advantages, recent work has applied LoRA to Federated Learning (FL) to enable efficient on-device adaptation. A central obstacle in this setting is non-independent and identically distributed (non-IID) client data. In practice, local datasets often differ substantially across clients due to variations in user behavior and device capabilities, leading to statistical heterogeneity. While empirical results show that LoRA performs on par with full fine-tuning (FFT) in centralized settings (Hu et al., 2022), its performance deteriorates in heterogeneous federated environments (Babakniya et al., 2023). As shown in Figure 1, Federated LoRA not only underperforms FFT but also exhibits a widening gap as non-IID conditions intensify. This motivates our investigation into two central questions: *(1) Why does non-IID*[1] *data severely degrade LoRA's effectiveness in FL? and (2) Can we design parameter-efficient approaches that remain robust under such non-IID conditions?*

To address these questions, we analyze fine-tuning dynamics through the neural tangent kernel (NTK) lens (Jacot et al., 2018). Empirically, pretrained language models operate in a kernel-like

[1]Throughout this paper, we use "data heterogeneity" and "non-IID" interchangeably to refer to differing data distributions across clients



(a) IID vs. Non-IID (b) Varying Non-IID Levels

Figure 1: Comparison of full-parameter fine-tuning (FFT), FedAvg, and Federated LoRA under Dirichlet non-IID data (α). (a) At $\alpha = 0.5$, FFT is more robust than LoRA, with a smaller drop from the IID baseline. (b) On MNLI, the performance gap widens as α decreases.

regime at initialization (Malladi et al., 2023), so full-parameter fine-tuning is well approximated by kernel regression with kernel $\mathcal{K}_{\mathrm{FFT}}$. LoRA restricts updates to a fixed low-rank subspace, which is equivalent to applying a random low-rank projection to the input-side features defining $\mathcal{K}_{\mathrm{FFT}}$, thereby inducing a kernel $\mathcal{K}_{\mathrm{LoRA}}$. By the Johnson–Lindenstrauss (JL) lemma (Johnson et al., 1984), when the projection rank k is sufficiently large, $\mathcal{K}_{\mathrm{LoRA}} \approx \mathcal{K}_{\mathrm{FFT}}$ with high probability, explaining LoRA's strong performance in centralized (IID) settings. Under non-IID client distributions, however, heterogeneous data diversifies and misaligns gradient directions, increasing the effective rank (energy) of the NTK (gradient Gram) matrix. With fixed k, $\mathcal{K}_{\mathrm{LoRA}}$ cannot capture this additional structure, so $\|\mathcal{K}_{\mathrm{FFT}} - \mathcal{K}_{\mathrm{LoRA}}\|$ grows and the spectral floor $\lambda_{\min}(\mathcal{K}_{\mathrm{LoRA}})$ drops, leading to slower convergence and weaker generalization. By contrast, gradient compression methods such as GaLore (Zhao et al., 2024) do not fix a low-rank subspace but adaptively compress gradients in the current span, thereby preserving the kernel $\mathcal{K}_{\mathrm{FFT}}$ and avoiding spectral-floor collapse, making them more robust under non-IID shifts.

Based on the above analysis, we propose a novel federated low-rank adaptation strategy, `FedLore`. On the client side, `FedLore` builds on the GaLore optimizer, where gradients are projected via Singular Value Decomposition (SVD). While effective, exact SVD is computationally expensive. To address this, we adopt randomized SVD, which is more efficient while retaining comparable optimization performance.

On the server side, we observe that client gradients exhibit a shared low-rank structure with additive client-specific perturbations and noise, $\boldsymbol{G}_i = \boldsymbol{G} + \boldsymbol{P} + \text{noise}$. To align client update subspaces, `FedLore` introduces an aggregation strategy that extracts the shared low-rank component $\boldsymbol{G}$, broadcasts it to clients, and initializes their GaLore projectors accordingly. This ensures each local round begins with an aligned subspace. In experiments across NLU, vision `FedLore` consistently achieves higher accuracy and robustness under non-IID conditions than LoRA-based strategies, while using comparable or less memory.

Contribution Our main contributions are threefold. First, we provide an NTK-based analysis of LoRA in federated settings, showing that it fails under non-IID data due to increased gradient rank and spectral-floor collapse. Building on this insight, we introduce `FedLore`, a federated low-rank adaptation strategy that employs efficient GaLore-based optimization on the client side and a server-side aggregation scheme to align update subspaces. Finally, through extensive experiments on NLU, vision benchmarks, we show that `FedLore` achieves higher accuracy and robustness under heterogeneous data while using comparable or less memory than LoRA-based approaches.

2 Related Works

LoRA LoRA's parameter efficiency has inspired a wide range of extensions. AdaLoRA (Zhang et al., 2023) dynamically prunes singular values to optimize rank budgets, VeRA (Kopiczko et al., 2023) shares low-rank matrices across layers, and DoRA (Liu et al., 2024) decomposes pretrained weights into magnitude and direction. Other works focus on initialization, such as ReLoRA (Lialin

et al., 2024) and PeriodicLoRA (Meng et al., 2024b), or adopt data-driven strategies including Pissa (Meng et al., 2024a), LoRA-SB (Ponkshe et al., 2024), LoRA-GA (Wang et al., 2024a), and EVA (Paischer et al., 2024). These approaches improve efficiency in centralized training but do not address challenges unique to federated settings. Meanwhile, several studies examine the theory of LoRA: Zeng & Lee (2023) analyzed its expressive power, Jang et al. (2024) investigated its optimization landscape under convexity assumptions, and Xu et al. (2025) studied its dynamics for matrix factorization. However, none of these works consider non-IID client shifts in federated learning.

GaLore Beyond LoRA, GaLore (Zhao et al., 2024) has emerged as a training strategy that achieves memory efficiency by projecting gradient matrices into low-rank subspaces. WeLore (Jaiswal et al., 2024) adaptively selects the projection rank, while OwLore (Li et al., 2024) introduces layer-wise updates to improve flexibility and efficiency. Hao et al. (2024) established a connection between LoRA and GaLore, and Liu et al. (2025) analyzed their optimization landscapes, showing that GaLore enjoys more favorable optimization properties.

Low-Rank Adaptation in Federated Learning LoRA is also the most widely adopted parameter-efficient fine-tuning approach in federated learning, and several works adapt it to this setting. FedIT (Zhang et al., 2024) averages client LoRA matrices, while subsequent variants improve aggregation (e.g., FLoRA (Wang et al., 2024b), LoRA-Fair (Bian et al., 2024)) or initialization (e.g., FR-LoRA, FedERA). Other approaches modify the sharing of the $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices, such as FedSA-LoRA (Guo et al., 2024) and FFA-LoRA (Sun et al., 2024). The most closely related work, FedFTG (Mahla et al., 2024), applies GaLore directly as an optimizer in federated settings. In contrast, we provide an NTK-based analysis of non-IID client shifts and extend GaLore with an improved projector and a novel aggregation strategy.

3 Preliminaries

To set the stage for our analysis, we first establish notation for neural networks and gradient updates, and then review the formulations of LoRA and GaLore.

We consider a neural network $f(\boldsymbol{x};\boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$. Linear or affine submodules (e.g., attention or MLP projections) are indexed by l. After t training steps, each block has weight matrix $\boldsymbol{W}_l^{(t)} \in \mathbb{R}^{d_{\text{out}}\times d_{\text{in}}}$ and pre-activation $\boldsymbol{h}_l^{(t)} = \boldsymbol{W}_l^{(t)}\boldsymbol{x}_l$. LoRA injected at layer l reparameterizes the weight as

$$\boldsymbol{W}_l^{(t)} = \boldsymbol{W}_l^{(0)} + \boldsymbol{B}_l^{(t)}\boldsymbol{A}_l^{(t)}, \tag{1}$$

where $\boldsymbol{B}_l \in \mathbb{R}^{d_{\text{out}}\times r}$ and $\boldsymbol{A}_l \in \mathbb{R}^{r\times d_{\text{in}}}$, with $r \ll \min\{d_{\text{out}}, d_{\text{in}}\}$ denoting the adapter rank. In standard LoRA, $\boldsymbol{W}_l^{(0)}$ is frozen and $(\boldsymbol{B}_l, \boldsymbol{A}_l)$ are trainable. The matrices are typically initialized as $\boldsymbol{B}_l^{(0)} = \boldsymbol{0}$ and $\boldsymbol{A}_l^{(0)} \sim \mathcal{N}(0, 1/r)$.

Unlike LoRA's reparameterization, GaLore applies low-rank projection directly to the gradients during optimization. At training step t, the full gradient of $\boldsymbol{W}_l^{(t)}$ is $\boldsymbol{G}_l^{(t)} = \nabla_{\boldsymbol{W}_l}\mathcal{L}(f(\boldsymbol{x};\boldsymbol{\theta}))$, where $\mathcal{L}$ is the loss function. GaLore maintains a projection matrix $\boldsymbol{P}_l^{(t)} \in \mathbb{R}^{d_{\text{in}}\times r}$ with $r \ll \min\{d_{\text{out}}, d_{\text{in}}\}$. The gradient is then compressed and reconstructed as

$$\tilde{\boldsymbol{G}}_l^{(t)} = \boldsymbol{G}_l^{(t)}\boldsymbol{P}_l^{(t)}\boldsymbol{P}_l^{(t)\top}, \tag{2}$$

and used by the optimizer to perform parameter updates, including accumulation of momentum in adaptive methods such as Adam (Kingma & Ba, 2014) or AdamW (Loshchilov & Hutter, 2017). In practice, $\boldsymbol{P}_l^{(t)}$ is obtained from a low-rank factorization (e.g., truncated SVD) of recent gradients and is periodically refreshed.

4 Analysis

In this section, we analyze the training dynamics of LoRA and GaLore in comparison to full-parameter fine-tuning (FFT). Since our analysis relies on the neural tangent kernel (NTK), it requires

the assumption that fine-tuning of a pretrained language model $\boldsymbol{\theta}$ exhibits *kernel behavior*. This assumption is not only theoretically convenient but also empirically validated: pretrained language models fine-tuned from $\boldsymbol{\theta}^{(0)}$ have been observed to follow kernel dynamics, with training trajectories well predicted by the corresponding kernels (Malladi et al., 2023).

Definition 4.1 (Kernel behavior (Malladi et al., 2023; Woodworth et al., 2020))**.** Training is said to exhibit kernel behavior at $\boldsymbol{\theta}^{(0)}$ if, along the fine-tuning trajectory,

$$\begin{aligned} f(\boldsymbol{x};\boldsymbol{\theta}) \;&\approx\; f(\boldsymbol{x};\boldsymbol{\theta}^{(0)}) \;+\; \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta}^{(0)})^{\top}(\boldsymbol{\theta}-\boldsymbol{\theta}^{(0)}) \qquad \textbf{linearization} \\ \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta}^{(t)}) &\approx \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta}^{(0)}) \qquad \textbf{fixed features} \end{aligned} \tag{3}$$

Under this assumption, one training step with optimizer $\mathcal{A}$ updates predictions as

$$f(\,\cdot\,;\boldsymbol{\theta}^{(t+1)}) - f(\,\cdot\,;\boldsymbol{\theta}^{(t)}) \;\approx\; -\eta_t\,\chi_t\,\mathcal{K}_{\mathcal{A}}(\,\cdot\,,\boldsymbol{x}^{(t)}) \tag{4}$$

where $\chi_t = \partial\mathcal{L}(f(\boldsymbol{x}^{(t)};\boldsymbol{\theta}^{(t)}),y^{(t)})/\partial f$ and $\mathcal{K}_{\mathcal{A}}$ is an optimizer-specific kernel. In other words, fine-tuning reduces to kernel gradient descent and the training dynamics are fully determined by $\mathcal{K}_{\mathcal{A}}$.

For SGD, $\mathcal{K}_{\mathrm{SGD}}$ is the NTK (Jacot et al., 2018) $\mathcal{K}_{\mathrm{SGD}}(\boldsymbol{x}_i,\boldsymbol{x}_j) = \langle \nabla f(\boldsymbol{x}_i;\boldsymbol{\theta}^{(0)}), \nabla f(\boldsymbol{x}_j;\boldsymbol{\theta}^{(0)})\rangle$, where $\langle\cdot\rangle$ denotes the inner product. For adaptive optimizers such as Adam and AdamW, early-stage dynamics admit a sign-based kernel of the form $\langle \mathrm{sign}(\nabla f(\boldsymbol{x}_i;\boldsymbol{\theta}^{(0)})), \mathrm{sign}(\nabla f(\boldsymbol{x}_j;\boldsymbol{\theta}^{(0)}))\rangle$ as shown by Littwin & Yang (2023), with AdamW additionally introducing a decoupled shrinkage in function space.

4.1 LoRA approximate the dynamics of Full-parameter finetuning

Building on this foundation, we now analyze the kernel induced by LoRA. We begin by defining the per-layer contribution of block l to the full-parameter kernel

$$\mathcal{K}^{(l)}_{\mathrm{FFT}}(i,j) \;=\; \underbrace{\big\langle \boldsymbol{d}_{\boldsymbol{h}_l}(i),\, \boldsymbol{d}_{\boldsymbol{h}_l}(j)\big\rangle}_{\boldsymbol{S}_l(i,j)} \cdot \underbrace{\big\langle \boldsymbol{x}_{l,i},\, \boldsymbol{x}_{l,j}\big\rangle}_{\boldsymbol{G}_l(i,j)}, \tag{5}$$

where $\boldsymbol{x}_{l,i}$ is the input to block l for sample i and $\boldsymbol{d}_{\boldsymbol{h}_l}(i)$ is the backprop signal into the pre-activation of block

$$\boldsymbol{d}_{\boldsymbol{h}_l}(i) \coloneqq \frac{\partial f(\boldsymbol{x}_i;\boldsymbol{\theta})}{\partial \boldsymbol{h}_l}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(0)}} \tag{6}$$

Theorem 4.2 (LoRA's kernel)**.** *When LoRA is injected at block l with right projection $\boldsymbol{A}_l$, the induced kernel is*

$$\mathcal{K}^{(l)}_{\mathrm{LoRA}}(i,j) \;=\; \boldsymbol{S}_l(i,j)\cdot\big\langle \boldsymbol{A}_l\boldsymbol{x}_{l,i},\, \boldsymbol{A}_l\boldsymbol{x}_{l,j}\big\rangle. \tag{7}$$

Proof. We defer the complete proof to Appendix B and provide a sketch here. At block l, with pre-activation $\boldsymbol{h}_l = \boldsymbol{W}_l\boldsymbol{x}_l$, the layerwise Jacobian at $\boldsymbol{\theta}^{(0)}$ factorizes as $\boldsymbol{\theta}^{(0)}$: $\nabla_{\boldsymbol{W}_l} f(\boldsymbol{x}_i;\boldsymbol{\theta}^{(0)}) = \boldsymbol{d}_{\boldsymbol{h}_l}(i)\,\boldsymbol{x}_{l,i}^{\top}$. Under LoRA $\boldsymbol{W}_l = \boldsymbol{W}_l^{(0)} + \boldsymbol{B}_l\boldsymbol{A}_l$ and standard initialization $\boldsymbol{B}_l^{(0)} = 0$, the first-order parameter is $\boldsymbol{B}_l$. Thus $\nabla_{\boldsymbol{B}_l} f(\boldsymbol{x}_i;\boldsymbol{\theta}^{(0)}) = \boldsymbol{d}_{\boldsymbol{h}_l}(i)\,(\boldsymbol{A}_l\boldsymbol{x}_{l,i})^{\top}$. Taking Frobenius inner products of these Jacobians across two samples gives Eq. equation 7 □

Theorem 4.2 states that LoRA does not alter the backprop block $\boldsymbol{S}_l(i,j) = \langle \boldsymbol{d}_{\boldsymbol{h}_l}(i), \boldsymbol{d}_{\boldsymbol{h}_l}(j)\rangle$; it only replaces the input inner product $\langle \boldsymbol{x}_{l,i},\boldsymbol{x}_{l,j}\rangle$ by $\langle \boldsymbol{A}_l\boldsymbol{x}_{l,i}, \boldsymbol{A}_l\boldsymbol{x}_{l,j}\rangle$. Consequently, if $\boldsymbol{A}_l$ approximately preserves pairwise inner products on the training inputs, the LoRA kernel closely matches the full-parameter kernel. Denote the network-level kernel as $\mathcal{K}_{\mathrm{FFT}} = \sum_l \mathcal{K}^{(l)}_{\mathrm{FFT}}$, $\mathcal{K}_{\mathrm{LoRA}} = \sum_l \mathcal{K}^{(l)}_{\mathrm{LoRA}}$, we have

Theorem 4.3 (LoRA mimic the dynamics of FFT with sufficient rank.)**.** *Consider a finite training set $\boldsymbol{x}_{l,1},\ldots,\boldsymbol{x}_{l,N}$ and for each block where LoRA is applied. Assume $|\boldsymbol{d}\boldsymbol{h}_l(i)| \le B_l$ and $|\boldsymbol{x}_{l,i}| \le R_l$ for all i. Assume $\|\boldsymbol{d}_{\boldsymbol{h}_l}(i)\| \le B_l$ and $\|\boldsymbol{x}_{l,i}\| \le R_l$ for all i. Let each $\boldsymbol{A}_l$ have IID subgaussian entries with variance 1 and set $\widetilde{\boldsymbol{A}}_l \coloneqq \boldsymbol{A}_l/\sqrt{r}$. If*

$$r \;\ge\; \frac{4}{\varepsilon^2-\varepsilon^3}\,\log\Big(\frac{4N^2\,|\mathcal{L}_{\mathrm{LoRA}}|}{\delta}\Big),$$

then with probability at least $1-\delta$*,*

$$\left|\mathcal{K}_{\mathrm{LoRA}}(i,j)-\mathcal{K}_{\mathrm{FFT}}(i,j)\right| \;\le\; \varepsilon \sum_{l\in\mathcal{L}_{\mathrm{LoRA}}} B_l^2\,\frac{\|\boldsymbol{x}_{l,i}\|^2+\|\boldsymbol{x}_{l,j}\|^2}{2} \qquad \forall\, i,j\in[N]. \tag{8}$$

In particular, if $\|\boldsymbol{x}_{l,i}\|\le 1$ *(e.g., post-LayerNorm) then* $\left|\mathcal{K}_{\mathrm{LoRA}}(i,j)-\mathcal{K}_{\mathrm{FFT}}(i,j)\right| \le \varepsilon \sum_l B_l^2$.

Proof. For each l and pair (i,j), By the classical Johnson–Lindenstrauss lemma (Johnson et al., 1984) $\left|\langle \widetilde{A}_l x_{l,i}, \widetilde{A}_l x_{l,j}\rangle - \langle x_{l,i}, x_{l,j}\rangle\right| \le \varepsilon(\|x_{l,i}\|^2+\|x_{l,j}\|^2)/2$ with probability $1-4\exp\left(-(\varepsilon^2-\varepsilon^3)r/4\right)$. Multiplying by $|\boldsymbol{S}_l(i,j)| \le |\boldsymbol{d}_{\boldsymbol{h}_l}(i)|, |\boldsymbol{d}_{\boldsymbol{h}_l}(j)| \le B_l^2$ and applying a union bound over all N^2 pairs and blocks gives equation 8. A complete proof is deferred to Appendix C. □

Theorem 4.3 shows that, with sufficiently large adapter rank r, the LoRA kernel $\mathcal{K}_{\mathrm{LoRA}}$ uniformly approximates the full-parameter kernel $\mathcal{K}_{\mathrm{FFT}}$ on the training set. This explains why, in centralized (IID) settings, LoRA often matches the performance of full-parameter fine-tuning. However, its behavior under non-IID client data differs substantially. We analyze this regime next.

4.2 Why LoRA break under client drift

In federated learning, each client c has access to a local dataset $\mathcal{D}_c$ drawn from its own distribution P_c. When data are heterogeneous ($P_{c_1} \neq P_{c_2}$), the local objectives

$$\mathcal{L}_c(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim P_c}\left[\ell(f(\boldsymbol{x};\boldsymbol{\theta}),y)\right]$$

differ across clients. Consequently, the corresponding local gradients $\nabla_{\boldsymbol{\theta}}\mathcal{L}_c(\boldsymbol{\theta})$ point in different directions. This misalignment between local and global gradients is referred to as *client drift*(Karimireddy et al., 2020). Formally we define:

Definition 4.4 (Non-degenerate client drift)**.** Client drift at layer l is *non-degenerate* if the per-example backprop signals acquire additional variance and rotate into directions that were not previously dominant. Writing $\boldsymbol{D}_l = [\boldsymbol{d}_{\boldsymbol{h}_l}(1),\ldots,\boldsymbol{d}_{\boldsymbol{h}_l}(N)]$ and $\boldsymbol{S}_l = \boldsymbol{D}_l^\top \boldsymbol{D}_l$, drift produces a new Gram $\boldsymbol{S}_l'$ such that

$$\mathrm{tr}(\boldsymbol{S}_l') > \mathrm{tr}(\boldsymbol{S}_l) \quad \text{and} \quad \mathrm{srank}(\boldsymbol{S}_l') > \mathrm{srank}(\boldsymbol{S}_l),$$

where $\mathrm{srank}(A) := \|A\|_F^2/\|A\|_2^2$ is the stable (effective) rank.

Definition 4.4 excludes *trivial drift* like pure rescaling ($\boldsymbol{S}_l' = c\,\boldsymbol{S}_l$) or adding energy only along the current top direction. It captures the structural effect of data heterogeneity: local data injects genuinely new variance directions so that both energy and effective rank grow. In Appendix D we give a formal "covariance dominance + incoherence" condition implying Definition 4.4 and show that it holds in standard factor–plus–noise models of non-IID shift.

Under non-degenerate client drift, the backprop block gets *larger* and *less concentrated*, while LoRA keeps the same fixed input-side projection. This amplifies the LoRA–vs–FFT kernel deviation at fixed rank and weakens the provable lower bound of $\lambda_{min}(\mathcal{K}_{\mathrm{LoRA}})$. Formally we have,

Theorem 4.5 (Drift amplifies LoRA's kernel deviation and lowers its spectral floor)**.** *Consider LoRA at blocks* $l\in\mathcal{L}_{\mathrm{LoRA}}$ *with right matrices* $\boldsymbol{A}_l\in\mathbb{R}^{r\times d_{\mathrm{in}}}$ *and* $\widetilde{\boldsymbol{A}}_l := \boldsymbol{A}_l/\sqrt{r}$*. Let* $\boldsymbol{X}_l = [\boldsymbol{x}_{l,1},\ldots,\boldsymbol{x}_{l,N}]$ *and* $r_{x,l} = \mathrm{rank}(\boldsymbol{X}_l)$*. Denote optimizer-specific kernels (full vs. LoRA) after non-degenerate as* K'_{FFT} *and* K'_{LoRA}*. Then, with probability at least* $1-\delta$ *over the LoRA projections,*

$$\left\|K'_{\mathrm{LoRA}} - K'_{\mathrm{FFT}}\right\|_2 \;\le\; \sum_{l\in\mathcal{L}_{\mathrm{LoRA}}} C\,\|\boldsymbol{S}_l'\|_2\,\|\boldsymbol{X}_l\|_2^2\,\sqrt{\tfrac{r_{x,l}}{r}}, \tag{9}$$

$$\lambda_{\min}\!\left(K'_{\mathrm{LoRA}}\right) \;\ge\; \lambda_{\min}\!\left(K'_{\mathrm{FFT}}\right) \;-\; \sum_{l\in\mathcal{L}_{\mathrm{LoRA}}} C\,\|\boldsymbol{S}_l'\|_2\,\|\boldsymbol{X}_l\|_2^2\,\sqrt{\tfrac{r_{x,l}}{r}} \tag{10}$$

An $\|\cdot\|_F$ *variant also holds:*

$$\left\|K'_{\mathrm{LoRA}} - K'_{\mathrm{FFT}}\right\|_F \;\le\; \sum_{l\in\mathcal{L}_{\mathrm{LoRA}}} C\,\|\boldsymbol{S}_l'\|_F\,\|\boldsymbol{X}_l\|_2^2\,\frac{r_{x,l}}{\sqrt{r}}\,, \tag{11}$$

which increases whenever $\|\boldsymbol{S}_l'\|_F$ *increases (a consequence of Definition 4.4 under the mild bounded-spikiness condition in Appendix D).*

Theorem 4.5 implies that client drift increases the energy of the backprop signals, which in turn loosens the otherwise tight bound on the deviation between LoRA and full-parameter fine-tuning. As this energy grows, the gap between $\mathcal{K}_{\mathrm{LoRA}}$ and $\mathcal{K}_{\mathrm{FFT}}$ widens monotonically. In effect, non-IID heterogeneity makes LoRA progressively less robust: what was a close approximation under i.i.d. data becomes increasingly fragile under drift, and the deviation can only get worse as energy continues to rise.

At the same time, the lower bound on $\lambda_{\min}(\mathcal{K}_{\mathrm{LoRA}})$ deteriorates, implying slower convergence and weaker generalization. Chen et al. (2023) show that for a loss function $\mathcal{L}(\cdot)$ that is L_f-Lipschitz continuous, the convergence rate and generalization gap are controlled by the condition number L_f/μ, where

$$\mu \coloneqq \inf_{\boldsymbol{W}} \lambda_{\min}(\mathcal{K}).$$

In our setting, client drift reduces the lower bound of $\lambda_{\min}(\mathcal{K}_{\mathrm{LoRA}})$, thereby decreasing μ and enlarging L_f/μ. This directly translates into slower convergence and a larger generalization gap.

4.3 GaLore is more robust against client drift

Unlike LoRA, GaLore does not reparameterize $\boldsymbol{W}_l$; it operates directly on the *step* by projecting the gradient $\boldsymbol{G}_l^{(t)}$ onto a data-adapted rank-r right subspace $\boldsymbol{P}_l^{(t)}$ and updating with

$$\tilde{\boldsymbol{G}}_l^{(t)} \;=\; \boldsymbol{G}_l^{(t)}\,\boldsymbol{P}_l^{(t)}\boldsymbol{P}_l^{(t)\top}.$$

Because the parameters remain in the full space, the underlying Jacobian J and kernel $\mathcal{K}$ are unchanged. There is no fixed input-side distortion $x \mapsto \boldsymbol{A}_l x$ as in LoRA. As a result, the *spectral floor* (minimum eigenvalue) that enters PL-based guarantees is preserved, and GaLore does not suffer the drift-dependent penalty in equation 10.

We define the *captured gradient energy* at (l,t) as

$$\alpha_{l,t} := \frac{\|\boldsymbol{G}_l^{(t)}\boldsymbol{P}_l^{(t)}\boldsymbol{P}_l^{(t)\top}\|_F^2}{\|\boldsymbol{G}_l^{(t)}\|_F^2} \in [0,1], \qquad \alpha_t := \min_l \alpha_{l,t}.$$

If $\boldsymbol{P}_l^{(t)}$ is chosen as the top-r right singular directions of recent $\boldsymbol{G}_l^{(t)}$ (truncated SVD), then by the Eckart–Young theorem $\alpha_{l,t}$ is maximal among all rank-r right projections. Linearizing one step gives

$$\Delta f_t(\cdot) \;\approx\; -\eta_t\, J\Big(\bigoplus_l \Pi_{l,t}\Big) J^\top \chi_t, \qquad \Pi_{l,t} : G \mapsto G\,\boldsymbol{P}_l^{(t)}\boldsymbol{P}_l^{(t)\top},$$

, where $\bigoplus_l \Pi_{l,t}$ denotes the block-diagonal operator that applies the projection $\Pi_{l,t}$ independently at each block l. So the GaLore update is exactly the full-FT update *scaled* in the current descent direction, with a per-step contraction factor α_t:

$$\|\Delta f_t^{\text{GaLore}} - \Delta f_t^{\text{FFT}}\| \;\le\; (1-\alpha_t)\,\|\Delta f_t^{\text{FFT}}\|.$$

In words: GaLore preserves the kernel geometry (hence $\lambda_{\min}(\mathcal{K})$) while only reducing per-step progress by at most $(1-\alpha_t)$. By contrast, LoRA fixes $\boldsymbol{A}_l$ *ex ante*, so when drift rotates and spreads gradient directions (Def. 4.4), its kernel deviation scales with $\|\boldsymbol{S}_l'\|_2$ (Theorem 4.5) and its spectral floor degrades. GaLore instead *refreshes* $\boldsymbol{P}_l^{(t)}$ from the *observed* gradients, so as directions rotate, the chosen subspace follows them. As long as the top-r singular directions capture most of the mass, $\alpha_{l,t}$ remains close to 1 even when the effective rank grows — precisely the regime where LoRA's fixed projection accumulates error. Crucially, GaLore's discrepancy bound depends only on $(1-\alpha_t)$ and not on $\|\boldsymbol{S}_l'\|$, so increasing drift "energy" does *not* directly erode any certified kernel floor.

5 FedLoRe: Federated Low-Rank Adaptation with Aligned Projections

Client Training Recall that GaLore compresses the layerwise gradient $\boldsymbol{G}_l^{(t)}$ by projecting onto a rank-r right subspace,

$$\tilde{\boldsymbol{G}}_l^{(t)} \;=\; \boldsymbol{G}_l^{(t)}\,\boldsymbol{P}_l^{(t)}\boldsymbol{P}_l^{(t)\top}.$$

In vanilla GaLore, the projector $\boldsymbol{P}_l^{(t)} \in \mathbb{R}^{d_{\text{in}} \times r}$ is obtained from the top-r right singular vectors of $\boldsymbol{G}_l^{(t)}$, i.e., via truncated SVD:

$$\boldsymbol{G}_l^{(t)} \;=\; U_l \Sigma_l V_l^\top, \qquad \boldsymbol{P}_l^{(t)} = V_{l,1:r}.$$

By the Eckart–Young theorem, this choice maximizes the captured gradient energy among all rank-r projections. However, computing an exact SVD at every training step is computationally prohibitive and requires storing large matrices.

To address this, `FedLore` replaces exact SVD with randomized SVD (RSVD) (**?**). Given $\boldsymbol{G}_l^{(t)} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, we draw a random Gaussian test matrix $\Omega \in \mathbb{R}^{d_{\text{in}} \times (r+p)}$ with a small oversampling factor p, form the sketch $Y = \boldsymbol{G}_l^{(t)}\Omega$, and compute an orthonormal basis $Q = \text{orth}(Y)$. The projector is then obtained from the SVD of the reduced matrix $\boldsymbol{G}_l^{(t)}Q$:

$$\boldsymbol{G}_l^{(t)}Q \;=\; \tilde{U}\tilde{\Sigma}\tilde{V}^\top, \qquad \boldsymbol{P}_l^{(t)} = Q\tilde{V}_{:,1:r}.$$

RSVD approximates the top-r right singular subspace at much lower cost by reducing the dimension of the SVD calculation. In particular, it reduces the computational complexity from $\mathcal{O}(d_{\text{out}} d_{\text{in}}^2)$ for exact SVD to $\mathcal{O}(d_{\text{out}} d_{\text{in}} r)$ for rank-r approximation, thereby significantly improving efficiency while still retaining near-optimal accuracy with high probability.

Server Aggregation Our analysis and experiments show that the *initial* projection matrix plays a decisive role in both convergence speed and final accuracy. If the first projector $\boldsymbol{P}_l^{(0)}$ is misaligned, the model converges more slowly and often plateaus at a higher loss, even when subsequent projectors are updated adaptively. This is because the very first projection defines the descent subspace for early training, and errors introduced at this stage propagate across rounds.

Figure 2 compares FFT, LoRA, GaLore, and GaLore-Random (where the initial projector is random). GaLore closely tracks FFT, while GaLore-Random converges more slowly and reaches higher loss, showing that aligning the first projection with dominant gradient directions is essential for robust performance.

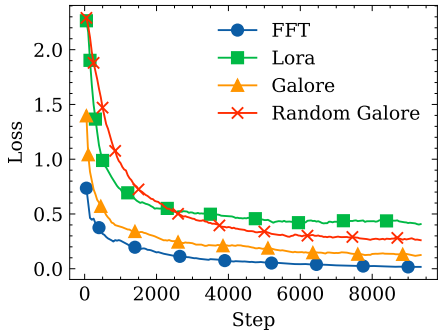


Figure 2: Effect of the initial projection matrix on CIFAR-10 with an MLP.

This highlights the importance of the *first initialization* of local training: the initial projection must be aligned with the dominant gradient subspace. Without this alignment, early updates take place in a suboptimal subspace, slowing convergence and leading to higher final loss. Motivated by this observation, our goal is to identify the *dominant subspace* that captures the gradient structure shared across clients. We model the local gradient of client c at layer l as

$$\boldsymbol{G}_{c,l} \;=\; \boldsymbol{G}_l^* \;+\; \boldsymbol{L}_{c,l} \;+\; \boldsymbol{E}_{c,l},$$

where $\boldsymbol{G}_l^*$ denotes the shared low-rank component common across clients, $\boldsymbol{L}_{c,l}$ is a client-specific low-rank perturbation, and $\boldsymbol{E}_{c,l}$ is random noise.

This decomposition is supported by empirical evidence: gradients in deep networks are often approximately low-rank Zhao et al. (2024), and additional structure introduced by client drift can also be captured in low-rank form. The remaining stochastic variability from optimization manifests as noise. Thus, extracting the shared component $\boldsymbol{G}_l^*$ provides a principled way to estimate the dominant subspace. The server can then broadcast this shared subspace to clients, ensuring that each local round begins from an aligned projector and avoiding the instability caused by misaligned initialization.

To separate shared from client-specific structure, we draw on the classical *Joint and Individual Variation Explained* (JIVE) framework. For each layer l, we model local gradients $\boldsymbol{G}_{c,l}$ as the sum of a shared component $\boldsymbol{G}_l^*$, client-specific perturbations $\boldsymbol{L}_{c,l}$, and noise. Extracting $\boldsymbol{G}_l^*$ admits a convex surrogate via nuclear-norm regularization:

$$\min_{\boldsymbol{G}_l^*, \{\boldsymbol{L}_{c,l}\}} \; \tfrac{1}{2} \sum_{c=1}^{N} w_c \, \|\boldsymbol{G}_{c,l} - \boldsymbol{G}_l^* - \boldsymbol{L}_{c,l}\|_F^2 + \lambda_* \|\boldsymbol{G}_l^*\|_* + \lambda_{\text{ind}} \sum_{c=1}^{N} \|\boldsymbol{L}_{c,l}\|_*. \tag{12}$$

This convex approximation guarantees global optimality while directly encoding our drift model.

We adopt an ADMM solver: the quadratic loss is split from the nuclear norms, and each subproblem reduces to singular-value thresholding (SVT). All SVD calls are truncated and computed via randomized SVD (RSVD), yielding complexity $\tilde{\mathcal{O}}(d_{\text{out}} d_{\text{in}} r)$. This ensures scalability even with many clients. From the optimal $\boldsymbol{G}_l^*$, we extract the right singular vectors $\boldsymbol{V}_{l,1:r}^*$, which serve as the server's shared projector. Broadcasting these projectors to all clients ensures that each round of local training begins from an aligned descent subspace anchored in the dominant shared gradient directions.

The overall FedLoRe pipeline therefore alternates between (i) local client updates with GaLore (using RSVD for projector refresh) and (ii) server-side extraction of the shared subspace via JIVE+ADMM. Clients send both their parameter state and last-step gradients; the server solves equation 12 to isolate $\boldsymbol{G}_l^*$, computes the shared projector, and broadcasts it back. This tight loop allows FedLoRe to continually adapt client subspaces to the evolving dominant gradient directions, mitigating drift and stabilizing convergence. The full pseudo-code of FedLoRe (including client GaLore, server JIVE+ADMM, and communication protocol) is provided in Appendix F.

6 Experiments

We evaluate `FedLore` across three diverse tasks: (1) **NLU**: RoBERTa-base (Liu, 2019) on 7 GLUE text classification tasks (Wang, 2018). (2) **Vision**: ViT-base (Dosovitskiy, 2020) on 6 DomainNet visual domains (Peng et al., 2019).

All GaLore-based methods use the GaLore variant of AdamW (Loshchilov & Hutter, 2017), while other baselines use standard AdamW with the following settings: learning rate 3×10^{-5}, weight decay 0.01, $\beta_1 = 0.9$, $\beta_2 = 0.98$, linear LR warmup, and gradient clipping at 1.0. Training configurations are: GLUE (15 global rounds, 2 local epochs), DomainNet (30 rounds, 3 local epochs). We simulate 5 clients per round.

We compare `FedLore` against the following baselines: FedAvg (full fine-tuning), FedIT (Zhang et al., 2024), FFA-LoRA (Sun et al., 2024) (freezes $\boldsymbol{A}$), FLoRA (Wang et al., 2024b), and FedFTG (Mahla et al., 2024). In particular, FedFTG corresponds to the naive GaLore baseline in Federated Leraning and serves as an ablation. For fairness, all LoRA-based methods use the same rank: $r = 8$ (GLUE), $r = 16$ (DomainNet).

We report performance under both IID (✓) and Non-IID (✗) splits. Non-IID data are generated with LDA partitioning (see Appendix A). The *performance gap* Δ between IID and Non-IID settings serves as a measure of robustness to heterogeneity: smaller Δ indicates better handling of Non-IID client distributions.

Results and Discussion We first evaluate on the GLUE benchmark (Table 1). The full-parameter FedAvg baseline shows relatively small performance gaps (Δ) across most tasks (CoLA, SST, MRPC, QNLI, and RTE), indicating that updating all parameters naturally provides robustness to data heterogeneity. In contrast, LoRA-based methods such as FedIT, FFA-LoRA, and FLoRA suffer from noticeably larger gaps under Non-IID splits. `FedLore` consistently maintains high accuracy under both IID and Non-IID conditions, narrowing the robustness gap while requiring far fewer trainable parameters. FedFTG achieves strong IID performance but exhibits worse robustness to Non-IID data, as it lacks the aggregation correction mechanism introduced in `FedLore`. A minor exception arises on STS-B, a *regression* task, where `FedLore` does not always achieve the best Non-IID score. Nevertheless, across classification tasks, our method demonstrates consistently strong performance and robustness under heterogeneity, with a substantially reduced parameter footprint.

We further evaluate on DomainNet. Since the ViT backbone is pretrained on ImageNet, it naturally performs well on the *Real* domain. We therefore focus on the Non-IID performance gaps across the *Clipart*, *Painting*, *Infograph*, *Quickdraw*, and *Sketch* domains to assess robustness. As shown in Table 2, the full-parameter FedAvg (100% trainable) again performs strongly. FedFTG achieves accuracy comparable to FedAvg and `FedLore` under IID splits, but its robustness degrades significantly under Non-IID conditions, mirroring our observations on GLUE. In contrast, `FedLore` attains equal or better accuracy in both IID and Non-IID settings, consistently showing higher tolerance to data heterogeneity. These DomainNet findings reinforce our GLUE results, demonstrating that `FedLore` enhances both accuracy and robustness under heterogeneity.

Table 1: GLUE Benchmark Results. ✕ denotes Non-IID, Δ indicates the difference.

Method	**CoLA** Acc%			**SST** Acc%			**MRPC** Acc%			**QQP** Acc%		
	✓	✕	Δ	✓	✕	Δ	✓	✕	Δ	✓	✕	Δ
FedAvg-Full	83.6	83.0	↓0.6	94.6	94.0	↓0.6	89.7	83.0	↓0.8	84.4	82.1	↓2.3
FedIT	71.2	65.4	↓5.8	88.1	83.7	↓4.4	82.3	76.9	↓5.4	78.9	72.4	↓6.5
FFA-LoRA	84.0	78.3	↓5.7	94.1	88.4	↓5.7	89.3	82.1	↓7.2	84.7	78.2	↓6.5
FLoRA	**84.7**	75.1	↓9.6	92.1	85.2	↓6.9	87.8	82.5	↓5.3	**86.3**	77.1	↓9.2
FedFTG	83.2	81.4	↓1.8	**94.5**	90.2	↓4.3	89.8	84.5	↓5.3	85.4	81.5	↓3.9
`FedLore`	83.9	**83.1**	**↓0.8**	94.8	**93.9**	**↓0.9**	**89.8**	**88.2**	**↓1.6**	84.3	**82.5**	**↓1.8**

Method	**MNLI** Acc%			**QNLI** Acc%			**RTE** Acc%			**STS-B** MSE		
	✓	✕	Δ	✓	✕	Δ	✓	✕	Δ	✓	✕	Δ
FedAvg-Full	82.5	81.1	↓1.4	90.1	89.8	↓0.3	75.1	68.2	↓6.9	0.45	0.67	↓.22
FedIT	75.6	69.8	↓5.8	83.4	78.1	↓5.3	66.5	58.9	↓7.6	0.38	0.55	↑.17
FFA-LoRA	82.2	75.8	↓6.4	89.9	83.4	↓6.5	74.8	65.3	↓9.5	0.44	0.56	**↑.16**
FLoRA	**82.7**	73.2	↓9.5	92.0	86.7	↓5.3	75.1	**68.2**	↓6.9	**0.39**	0.53	↑.14
FTG	81.2	77.4	↓3.8	**93.3**	88.1	↓5.2	**76.2**	68.0	↓8.2	**0.33**	**0.51**	↑.18
`FedLore`	**82.7**	**81.4**	**↓1.3**	90.3	**89.8**	**↓0.5**	75.2	**68.2**	**↓7.0**	0.45	0.67	↑.22

Table 2: DomainNet Results (excluding Real domain). Accuracies in %. ✓ denotes IID, ✕ denotes Non-IID, Δ indicates the difference.

Method	**Clipart**			**Painting**			**Infograph**			**Quickdraw**			**Sketch**		
	✓	✕	Δ	✓	✕	Δ	✓	✕	Δ	✓	✕	Δ	✓	✕	Δ
FedAvg-Full	82.3	78.9	↓3.4	79.2	75.8	↓3.4	54.3	48.5	↓5.8	71.0	67.2	↓3.8	78.3	74.1	↓4.2
FedIT	80.1	74.2	↓5.9	77.5	70.8	↓6.7	52.8	45.9	↓6.9	69.4	62.8	↓6.6	76.2	69.1	↓7.1
FFA-LoRA	79.8	72.5	↓6.3	76.9	71.2	↓5.7	51.9	46.2	**↓5.7**	68.7	63.5	↓5.2	75.8	68.4	↓7.4
FLoRA	80.7	74.1	↓6.6	77.8	71.3	↓6.5	52.1	45.0	↓7.1	69.5	63.2	↓6.3	76.0	68.5	↓7.5
FTG	82.1	71.3	↓10.8	**79.5**	72.4	↓7.1	**53.8**	41.9	↓11.9	69.2	64.3	↓4.9	76.8	**70.5**	↓ 6.3
`FedLore`	**81.9**	**78.2**	**↓3.7**	78.8	**75.1**	**↓3.7**	**53.8**	**47.6**	↓6.2	**70.5**	**66.4**	**↓4.1**	**77.9**	**73.5**	**↓4.4**

7 Conclusion

In this work, we analyzed why LoRA, despite its popularity in centralized settings, fails under non-IID client data in federated learning. Our NTK-based analysis showed that client drift increases gradient energy and rank, loosening LoRA's approximation to full-parameter fine-tuning and lowering its spectral floor. In contrast, GaLore adaptively compresses gradients without altering kernel geometry, making it more robust to drift. Building on these insights, we proposed `FedLore`, which combines client-side GaLore optimization with server-side projector alignment via a JIVE-style convex decomposition. Experiments across NLU, vision benchmarks demonstrated that `FedLore` improves robustness to heterogeneity while training only a small fraction of parameters. Overall, our results suggest adaptive gradient compression is a promising direction for scalable and robust federated fine-tuning.

Limitations and Future Work While `FedLore` shows strong robustness and efficiency, there remain several directions for improvement. First, the server-side projector extraction introduces extra computation compared to simpler aggregators, and scaling this to ultra-large models or thousands of clients will require further system optimization. Second, our theoretical analysis relies on the NTK regime; extending the guarantees to later training stages or non-kernel settings is an open challenge. Finally, in terms of privacy, `FedLore` follows the same assumptions as FedAvg, but could benefit from stronger analysis or integration with privacy-preserving techniques.

8 Ethics Statement

This work adheres to the ICLR Code of Ethics.[2] Our study does not involve human subjects, personally identifiable information, or sensitive data. We use only publicly available datasets (GLUE, DomainNet, under their respective licenses, and our methods are intended for advancing the robustness and efficiency of federated learning. We do not foresee any direct ethical concerns or potential harms arising from this research.

9 Reproducibility Statement

We have taken several steps to ensure the reproducibility of our results. All experimental configurations, including model architectures, hyperparameters, and training schedules, are described in Section 6. Additional details for theoretical analysis, including proofs and assumptions, are provided in Appendix, and implementation details for optimization and aggregation are described in Appendix F. Upon acceptance, we will release the full source code and scripts for dataset preprocessing, training, and evaluation to facilitate independent verification of our findings.

References

Sara Babakniya, Ahmed Roushdy Elkordy, Yahya H Ezzeldin, Qingfeng Liu, Kee-Bong Song, Mostafa El-Khamy, and Salman Avestimehr. Slora: Federated parameter efficient fine-tuning of language models. *arXiv preprint arXiv:2308.06522*, 2023.

Jieming Bian, Lei Wang, Letian Zhang, and Jie Xu. Lora-fair: Federated lora fine-tuning with aggregation and initialization refinement. *arXiv preprint arXiv:2411.14961*, 2024.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Yixuan Chen, Yubin Shi, Mingzhi Dong, Xiaochen Yang, Dongsheng Li, Yujiang Wang, Robert P. Dick, Qin Lv, Yingying Zhao, Fan Yang, Ning Gu, and Li Shang. Over-parameterized model optimization with polyak-{\L}ojasiewicz condition. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=aBIpZvMdS56.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Pengxin Guo, Shuang Zeng, Yanran Wang, Huijie Fan, Feifei Wang, and Liangqiong Qu. Selective aggregation for low-rank adaptation in federated learning. *arXiv preprint arXiv:2410.01463*, 2024.

Yongchang Hao, Yanshuai Cao, and Lili Mou. Flora: Low-rank adapters are secretly gradient compressors. *arXiv preprint arXiv:2402.03293*, 2024.

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 *Proceedings of the 10th International Conference on Learning Representations (ICLR'22)*, 2022.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Ajay Jaiswal, Lu Yin, Zhenyu Zhang, Shiwei Liu, Jiawei Zhao, Yuandong Tian, and Zhangyang Wang. From galore to welore: How low-rank weights non-uniformly emerge from low-rank gradients. *arXiv preprint arXiv:2407.11239*, 2024.

[2] https://iclr.cc/public/CodeOfEthics

Uijeong Jang, Jason D Lee, and Ernest K Ryu. Lora training in the ntk regime has no spurious local minima. *arXiv preprint arXiv:2402.11867*, 2024.

William B Johnson, Joram Lindenstrauss, et al. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023.

Dawid J Kopiczko, Tijmen Blankevoort, and Yuki M Asano. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

Pengxiang Li, Lu Yin, Xiaowei Gao, and Shiwei Liu. Owlore: Outlier-weighed layerwise sampled low-rank projection for memory-efficient llm fine-tuning. *arXiv preprint arXiv:2405.18380*, 2024.

Qinbin Li, Bingsheng He, and Dawn Song. Practical one-shot federated learning for cross-silo setting. *arXiv preprint arXiv:2010.01017*, 2020.

Vladislav Lialin, Sherin Muckatira, Namrata Shivagunde, and Anna Rumshisky. Relora: High-rank training through low-rank updates. In *Proceedings of the 12th International Conference on Learning Representations (ICLR'24)*, 2024.

Etai Littwin and Greg Yang. Adaptive optimization in the ∞-width limit. In *The Eleventh International Conference on Learning Representations*, 2023.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

Xu-Hui Liu, Yali Du, Jun Wang, and Yang Yu. On the optimization landscape of low rank adaptation methods for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Navyansh Mahla, Kshitij Sharad Jadhav, and Ganesh Ramakrishnan. Exploring gradient subspaces: Addressing and overcoming lora's limitations in federated fine-tuning of large language models. *arXiv preprint arXiv:2410.23111*, 2024.

Sadhika Malladi, Alexander Wettig, Dingli Yu, Danqi Chen, and Sanjeev Arora. A kernel-based view of language model fine-tuning. In *International Conference on Machine Learning*, pp. 23610–23641. PMLR, 2023.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. *arXiv preprint arXiv:2404.02948*, 2024a.

Xiangdi Meng, Damai Dai, Weiyao Luo, Zhe Yang, Shaoxiang Wu, Xiaochen Wang, Peiyi Wang, Qingxiu Dong, Liang Chen, and Zhifang Sui. Periodiclora: Breaking the low-rank bottleneck in lora optimization. *arXiv preprint arXiv:2402.16141*, 2024b.

Fabian Paischer, Lukas Hauzenberger, Thomas Schmied, Benedikt Alkin, Marc Peter Deisenroth, and Sepp Hochreiter. One initialization to rule them all: Fine-tuning via explained variance adaptation. *arXiv preprint arXiv:2410.07170*, 2024.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1406–1415, 2019.

Kaustubh Ponkshe, Raghav Singhal, Eduard Gorbunov, Alexey Tumanov, Samuel Horvath, and Praneeth Vepakomma. Initialization using update approximation is a silver bullet for extremely efficient low-rank fine-tuning. *arXiv preprint arXiv:2411.19557*, 2024.

Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving lora in privacy-preserving federated learning. In *Proceedings of the 12th International Conference on Learning Representations (ICLR'24)*, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Alex Wang. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020.

Shaowen Wang, Linxi Yu, and Jian Li. Lora-ga: Low-rank adaptation with gradient approximation. *Advances in Neural Information Processing Systems*, 37:54905–54931, 2024a.

Ziyao Wang, Zheyu Shen, Yexiao He, Guoheng Sun, Hongyi Wang, Lingjuan Lyu, and Ang Li. Flora: Federated fine-tuning large language models with heterogeneous low-rank adaptations. *arXiv preprint arXiv:2409.05976*, 2024b.

Blake Woodworth, Suriya Gunasekar, Jason D Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in overparametrized models. In *Conference on Learning Theory*, pp. 3635–3673. PMLR, 2020.

Ziqing Xu, Hancheng Min, Lachlan Ewen MacDonald, Jinqi Luo, Salma Tarmoun, Enrique Mallada, and René Vidal. Understanding the learning dynamics of lora: A gradient flow perspective on low-rank adaptation in matrix factorization. *arXiv preprint arXiv:2503.06982*, 2025.

Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International conference on machine learning*, pp. 7252–7261. PMLR, 2019.

Yuchen Zeng and Kangwook Lee. The expressive power of low-rank adaptation. *arXiv preprint arXiv:2310.17513*, 2023.

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

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection. *arXiv preprint arXiv:2403.03507*, 2024.

r

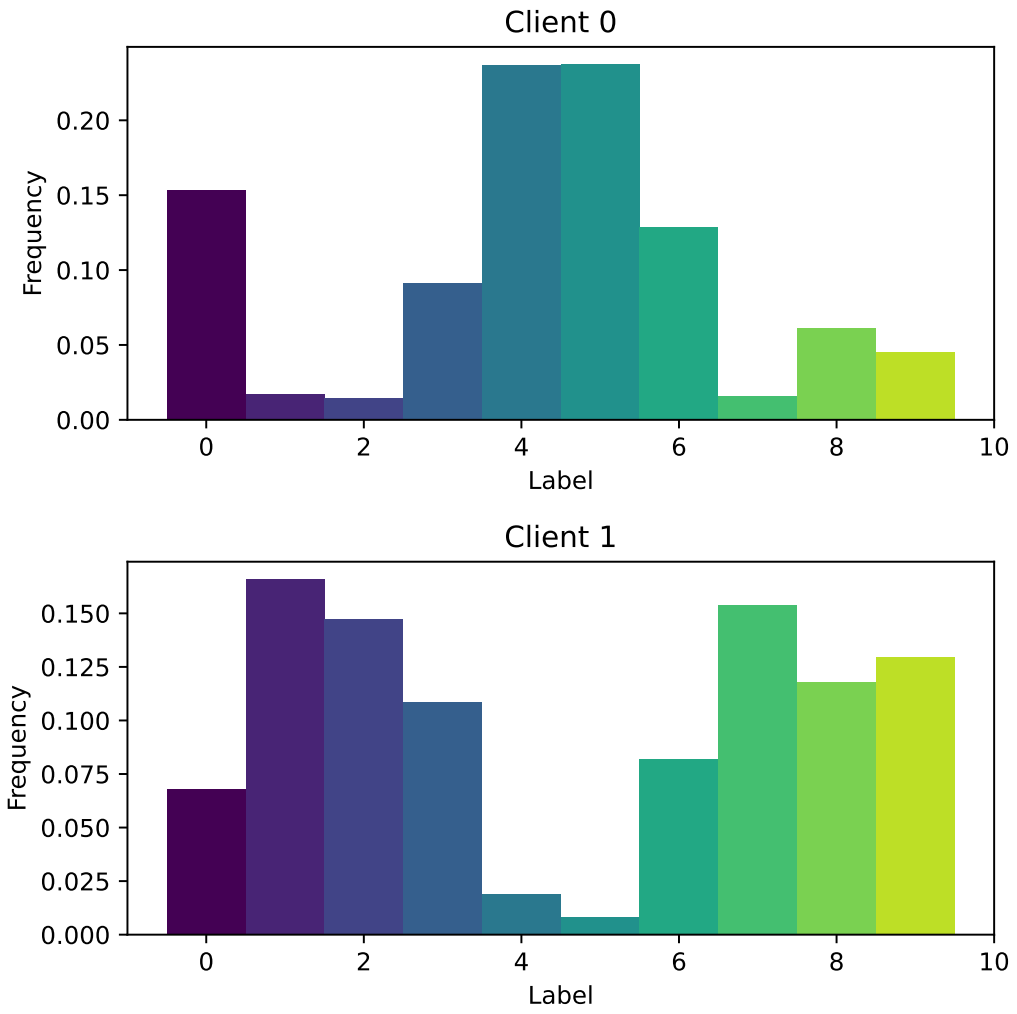


Figure 3: Illustration of Dirichlet-based data partitioning on MNIST ($\alpha = 0.5$) with two clients. Each bar represents the label distribution of one client.

A Dirichlet Distribution for Modeling Non-IID Data

Following common practice in federated learning (Yurochkin et al., 2019; Wang et al., 2020; Li et al., 2020), we model client data heterogeneity using a Dirichlet distribution. Specifically, for a classification task with K classes, the label distribution for each client i is sampled from a Dirichlet distribution:

$$p_i \sim \text{Dir}(\alpha \cdot \mathbf{1}_K),$$

where $\alpha > 0$ is the concentration parameter and $\mathbf{1}_K$ is a K-dimensional vector of ones. A smaller α produces more skewed client label distributions (i.e., stronger non-IID conditions), while a larger α yields more balanced distributions approaching the IID case.

After sampling p_i, we partition the dataset by allocating examples to client i according to p_i. This procedure ensures controlled heterogeneity across clients while keeping the total number of samples per client fixed. Figure 3 illustrates how the Dirichlet partition leads to distinct label distributions across clients. In our experiments, we vary α to study the impact of data heterogeneity on Federated LoRA and full-parameter fine-tuning (FFT).

B Proof of Theorem 4.2

Proof. **Step 1 (Jacobian factorization at $\boldsymbol{\theta}^{(0)}$).** By the chain rule,

$$\nabla_{\boldsymbol{W}_l} f(\boldsymbol{x}_i; \boldsymbol{\theta}^{(0)}) \;=\; \boldsymbol{d}_{\boldsymbol{h}_l}(i)\, \boldsymbol{x}_{l,i}^\top \;\in\; \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}. \tag{13}$$

Because $\boldsymbol{W}_l = \boldsymbol{W}_l^{(0)} + \boldsymbol{B}_l \boldsymbol{A}_l$, an infinitesimal variation gives $d\boldsymbol{W}_l = d\boldsymbol{B}_l\, \boldsymbol{A}_l + \boldsymbol{B}_l\, d\boldsymbol{A}_l$. Using Frobenius calculus,

$$df \;=\; \big\langle \nabla_{\boldsymbol{W}_l} f,\; d\boldsymbol{W}_l \big\rangle_F \;=\; \big\langle \nabla_{\boldsymbol{W}_l} f\, \boldsymbol{A}_l^\top,\; d\boldsymbol{B}_l \big\rangle_F \;+\; \big\langle \boldsymbol{B}_l^\top \nabla_{\boldsymbol{W}_l} f,\; d\boldsymbol{A}_l \big\rangle_F.$$

Hence the block-gradients are

$$\nabla_{\boldsymbol{B}_l} f \;=\; \nabla_{\boldsymbol{W}_l} f\; \boldsymbol{A}_l^\top, \qquad \nabla_{\boldsymbol{A}_l} f \;=\; \boldsymbol{B}_l^\top\, \nabla_{\boldsymbol{W}_l} f. \tag{14}$$

Evaluating equation 14 at $\boldsymbol{\theta}^{(0)}$ and using equation 13 yields

$$\nabla_{\boldsymbol{B}_l} f(\boldsymbol{x}_i; \boldsymbol{\theta}^{(0)}) \;=\; \boldsymbol{d}_{\boldsymbol{h}_l}(i)\, \big(\boldsymbol{A}_l^{(0)} \boldsymbol{x}_{l,i}\big)^\top \;\in\; \mathbb{R}^{d_{\text{out}} \times r}, \tag{15}$$

$$\nabla_{\boldsymbol{A}_l} f(\boldsymbol{x}_i; \boldsymbol{\theta}^{(0)}) \;=\; \big(\boldsymbol{B}_l^{(0)}\big)^\top\, \boldsymbol{d}_{\boldsymbol{h}_l}(i)\, \boldsymbol{x}_{l,i}^\top \;=\; 0 \;\in\; \mathbb{R}^{r \times d_{\text{in}}}, \tag{16}$$

since $\boldsymbol{B}_l^{(0)} = \mathbf{0}$. Thus, at $\boldsymbol{\theta}^{(0)}$, the A-block tangent feature vanishes identically, while the B-block carries feature $\boldsymbol{A}_l^{(0)} \boldsymbol{x}_{l,i}$ on the input side and $\boldsymbol{d}_{\boldsymbol{h}_l}(i)$ on the backprop side.

Step 2 (Kernel analog under kernel behavior). Kernel behavior (Def. 4.1) asserts that tangent features are fixed (along training) and equal to their values at $\boldsymbol{\theta}^{(0)}$ up to higher-order error. Therefore, the per-layer kernel analog is the inner product of the *nonzero* Jacobian blocks at $\boldsymbol{\theta}^{(0)}$:

$$\mathcal{K}^{(l)}_{\text{LoRA}}(i,j) \;=\; \Big\langle \nabla_{\boldsymbol{B}_l} f(\boldsymbol{x}_i; \boldsymbol{\theta}^{(0)}),\; \nabla_{\boldsymbol{B}_l} f(\boldsymbol{x}_j; \boldsymbol{\theta}^{(0)}) \Big\rangle_F,$$

because the A-block contributes $\langle 0, 0 \rangle_F = 0$, and cross-terms between $(\boldsymbol{B}_l, \boldsymbol{A}_l)$ vanish by block orthogonality. Using $\langle uv^\top,\, u'v'^\top \rangle_F = \langle u, u' \rangle \langle v, v' \rangle$ with equation 15 gives equation 7.

Notice that First-order dominance of the B-block (training A does not appear. This is because, from equation 14 at $\boldsymbol{\theta}^{(0)}$ we have $\nabla_{\boldsymbol{A}_l} f = 0$, so the first update of $\boldsymbol{A}_l$ is $\Delta \boldsymbol{A}_l^{(0)} = -\eta_0\, \nabla_{\boldsymbol{A}_l} L_S(\boldsymbol{\theta}^{(0)}) = 0$. After one step, $\boldsymbol{B}_l^{(1)} = O(\eta_0)$, so $\nabla_{\boldsymbol{A}_l} f(\cdot; \boldsymbol{\theta}^{(1)}) = \boldsymbol{B}_l^{(1)\top} \nabla_{\boldsymbol{W}_l} f = O(\eta_0)$ and $\Delta \boldsymbol{A}_l^{(1)} = O(\eta_0^2)$. Thus any contribution of the A-block to a kernel computed along the lazy trajectory is *second order* in step size, while the B-block appears at first order. The kernel analog, defined at $\boldsymbol{\theta}^{(0)}$ under fixed features, therefore depends only on equation 15, establishing equation 7. □

C Proof of Theorem 4.3

Proof. We first recall the definitions of the per–layer kernel contributions at $\boldsymbol{\theta}^{(0)}$. For block l,

$$\mathcal{K}^{(l)}_{\mathrm{FFT}}(i,j) \;=\; \boldsymbol{S}_l(i,j)\,\boldsymbol{G}_l(i,j) \quad \text{with} \quad \boldsymbol{S}_l(i,j) := \big\langle \boldsymbol{d}_{\boldsymbol{h}_l}(i), \boldsymbol{d}_{\boldsymbol{h}_l}(j) \big\rangle,\;\; \boldsymbol{G}_l(i,j) := \big\langle \boldsymbol{x}_{l,i}, \boldsymbol{x}_{l,j} \big\rangle,$$

and, when LoRA is inserted at l with right matrix $\boldsymbol{A}_l$,

$$\mathcal{K}^{(l)}_{\mathrm{LoRA}}(i,j) \;=\; \boldsymbol{S}_l(i,j)\,\widetilde{\boldsymbol{G}}_l(i,j), \qquad \widetilde{\boldsymbol{G}}_l(i,j) := \big\langle \widetilde{\boldsymbol{A}}_l \boldsymbol{x}_{l,i},\, \widetilde{\boldsymbol{A}}_l \boldsymbol{x}_{l,j} \big\rangle, \quad \widetilde{\boldsymbol{A}}_l := \boldsymbol{A}_l/\sqrt{r}.$$

(If block $l \notin \mathcal{L}_{\mathrm{LoRA}}$, then $\widetilde{\boldsymbol{G}}_l = \boldsymbol{G}_l$.) Hence, for any pair (i,j),

$$\big|\mathcal{K}^{(l)}_{\mathrm{LoRA}}(i,j) - \mathcal{K}^{(l)}_{\mathrm{FFT}}(i,j)\big| \;=\; |\boldsymbol{S}_l(i,j)| \cdot \big|\, \widetilde{\boldsymbol{G}}_l(i,j) - \boldsymbol{G}_l(i,j)\,\big|. \tag{17}$$

Step 1: Inner–product JL for each LoRA block. We use the following standard inner–product JL lemma (e.g., Johnson et al. (1984)): for any finite set $\mathcal{X} \subset \mathbb{R}^d$ of size N and any $\varepsilon \in (0,1)$, if $\Phi \in \mathbb{R}^{r\times d}$ has IID subgaussian entries with variance $1/r$, then

$$\Pr\Big[\big|\langle \Phi u, \Phi v\rangle - \langle u, v\rangle\big| \;\le\; \frac{\varepsilon}{2}\big(\|u\|^2 + \|v\|^2\big)\;\; \forall u, v \in \mathcal{X}\Big] \;\ge\; 1 - 4N^2 \exp\Big(-\tfrac{(\varepsilon^2-\varepsilon^3)r}{4}\Big). \tag{18}$$

Apply equation 18 with $\mathcal{X} = \{\boldsymbol{x}_{l,1}, \ldots, \boldsymbol{x}_{l,N}\}$ and $\Phi = \widetilde{\boldsymbol{A}}_l$ for each $l \in \mathcal{L}_{\mathrm{LoRA}}$. Let $\mathcal{E}_l$ be the event that equation 18 holds for block l; then

$$\Pr(\mathcal{E}_l) \;\ge\; 1 - 4N^2 \exp\Big(-\tfrac{(\varepsilon^2-\varepsilon^3)r}{4}\Big).$$

By the union bound,

$$\Pr\Big(\bigcap_{l\in\mathcal{L}_{\mathrm{LoRA}}} \mathcal{E}_l\Big) \;\ge\; 1 - 4\,|\mathcal{L}_{\mathrm{LoRA}}|\,N^2 \exp\Big(-\tfrac{(\varepsilon^2-\varepsilon^3)r}{4}\Big).$$

Choosing

$$r \;\ge\; \frac{4}{\varepsilon^2 - \varepsilon^3}\,\log\Big(\frac{4N^2\,|\mathcal{L}_{\mathrm{LoRA}}|}{\delta}\Big)$$

ensures $\Pr(\cap_l \mathcal{E}_l) \ge 1 - \delta$.

On the event $\cap_l \mathcal{E}_l$, for every $l \in \mathcal{L}_{\mathrm{LoRA}}$ and all i, j,

$$\big|\,\widetilde{\boldsymbol{G}}_l(i,j) - \boldsymbol{G}_l(i,j)\,\big| \;\le\; \frac{\varepsilon}{2}\big(\|\boldsymbol{x}_{l,i}\|^2 + \|\boldsymbol{x}_{l,j}\|^2\big). \tag{19}$$

Step 2: Multiply by the bounded backprop factor and sum over blocks. By assumption $\|\boldsymbol{d}_{\boldsymbol{h}_l}(i)\| \le B_l$ for all i, hence $|\boldsymbol{S}_l(i,j)| \le B_l^2$. Combining equation 17 and equation 19, for each LoRA'd block l and all i, j,

$$\big|\mathcal{K}^{(l)}_{\mathrm{LoRA}}(i,j) - \mathcal{K}^{(l)}_{\mathrm{FFT}}(i,j)\big| \;\le\; B_l^2 \cdot \frac{\varepsilon}{2}\big(\|\boldsymbol{x}_{l,i}\|^2 + \|\boldsymbol{x}_{l,j}\|^2\big).$$

Summing over $l \in \mathcal{L}_{\mathrm{LoRA}}$ (blocks without LoRA contribute 0) yields

$$\big|\mathcal{K}_{\mathrm{LoRA}}(i,j) - \mathcal{K}_{\mathrm{FFT}}(i,j)\big| \;\le\; \varepsilon \sum_{l\in\mathcal{L}_{\mathrm{LoRA}}} B_l^2\,\frac{\|\boldsymbol{x}_{l,i}\|^2 + \|\boldsymbol{x}_{l,j}\|^2}{2},$$

uniformly for all $i, j \in [N]$, with probability at least $1 - \delta$. If, in addition, $\|\boldsymbol{x}_{l,i}\| \le 1$ (e.g., post–LayerNorm), the right-hand side simplifies to $\varepsilon \sum_{l\in\mathcal{L}_{\mathrm{LoRA}}} B_l^2$. □

D Formalizing client drift: covariance dominance and incoherence

This appendix provides a precise condition under which client drift increases both the *energy* and the *effective rank* of the backprop Gram, justifying the informal Definition 4.4 in the main text.

Setup. Fix a LoRA'd block l. For training samples $\{x_i\}_{i=1}^N$, stack the per-example backprop signals (evaluated at $\boldsymbol{\theta}^{(0)}$) as columns

$$D_l \;=\; [\, d_{h_l}(1), \ldots, d_{h_l}(N) \,] \in \mathbb{R}^{d_{\text{out}} \times N}, \qquad S_l \;=\; D_l^\top D_l \;\succeq\; 0 \quad (N \times N).$$

Under drift, observed signals are $D_l' = D_l + P_l$, and

$$S_l' \;=\; D_l'^\top D_l' \;=\; S_l \;+\; \underbrace{(D_l^\top P_l + P_l^\top D_l)}_{\text{cross term}} \;+\; P_l^\top P_l. \tag{20}$$

We use the stable/effective rank $\operatorname{srank}(A) := \|A\|_F^2 / \|A\|_2^2$.

D.1 Population condition: covariance dominance and incoherence

Definition D.1 (CDI: covariance dominance + incoherence)**.** Let $T_l := \mathbb{E}[\, P_l^\top P_l \,] \succeq 0$ and let u_1 be a unit top eigenvector of S_l. We say drift at block l satisfies ***CDI***(α, β, r) if there exist $\alpha \geq 0$, $\beta \geq 0$ and a subspace $\mathcal{U} \subset u_1^\perp$ with $\dim(\mathcal{U}) = r$ such that

$$T_l \;\succeq\; \alpha\, I_N \;+\; \beta\, \mathrm{Proj}_{\mathcal{U}}. \tag{21}$$

We say the cross term is *unbiased* if $\mathbb{E}[\, D_l^\top P_l \,] = 0$.

The term αI_N is a population *variance floor* across samples; the $\beta\, \mathrm{Proj}_{\mathcal{U}}$ term imposes *incoherence*: a fixed fraction of energy lands away from the pre-drift top mode.

Lemma D.2 (Population lift)**.** *If* ***CDI***(α, β, r) *holds and the cross term is unbiased, then*

$$\mathbb{E}[\, S_l' \,] \;=\; S_l + T_l \;\succeq\; S_l \;+\; \alpha I_N \;+\; \beta\, \mathrm{Proj}_{\mathcal{U}}.$$

Consequently,

$$\operatorname{tr} \mathbb{E}[S_l'] \;\geq\; \operatorname{tr} S_l + \alpha N + \beta r, \qquad \|\mathbb{E}[S_l']\|_F^2 \;\geq\; \|S_l\|_F^2 + 2\alpha \operatorname{tr} S_l + 2\beta \operatorname{tr}(S_l \mathrm{Proj}_{\mathcal{U}}) + \alpha^2 N + 2\alpha\beta r + \beta^2 r.$$

Moreover, with eigenvalues $\lambda_1 \geq \cdots \geq \lambda_N \geq 0$ of S_l,

$$\operatorname{srank}(S_l + \alpha I_N) \;\geq\; \operatorname{srank}(S_l), \tag{22}$$

with strict inequality unless S_l is a scalar multiple of I_N.

Proof. The PSD inequality and the trace/Frobenius lower bounds follow from $\|A + B\|_F^2 = \|A\|_F^2 + 2\langle A, B \rangle + \|B\|_F^2$ and $\langle S_l, \mathrm{Proj}_{\mathcal{U}} \rangle = \operatorname{tr}(S_l \mathrm{Proj}_{\mathcal{U}})$. For equation 22,

$$\operatorname{srank}(S_l + \alpha I) = \frac{\sum_i (\lambda_i + \alpha)^2}{(\lambda_1 + \alpha)^2} = \frac{\sum_i \lambda_i^2 + 2\alpha \sum_i \lambda_i + N\alpha^2}{\lambda_1^2 + 2\alpha\lambda_1 + \alpha^2} \;\geq\; \frac{\sum_i \lambda_i^2}{\lambda_1^2} = \operatorname{srank}(S_l),$$

with strictness unless all λ_i are equal. □

Lemma D.3 (Stable-rank increase from incoherent lift)**.** *Let $S \succeq 0$ have eigenvalues $\lambda_1 \geq \cdots \geq \lambda_N$ and top eigenvector u_1. Let $\mathrm{Proj}_{\mathcal{U}}$ be a rank-r projector with $\mathcal{U} \subset u_1^\perp$. Then, for any $\beta > 0$,*

$$\|\, S + \beta\, \mathrm{Proj}_{\mathcal{U}} \,\|_F^2 \;\geq\; \|S\|_F^2 + 2\beta \sum_{i=N-r+1}^{N} \lambda_i + \beta^2 r, \qquad \|\, S + \beta\, \mathrm{Proj}_{\mathcal{U}} \,\|_2 \;\leq\; \lambda_1 + \beta.$$

Consequently,

$$\operatorname{srank}(S + \beta\, \mathrm{Proj}_{\mathcal{U}}) \;\geq\; \frac{\|S\|_F^2 + 2\beta \sum_{i=N-r+1}^{N} \lambda_i + \beta^2 r}{(\lambda_1 + \beta)^2} \;>\; \frac{\|S\|_F^2}{\lambda_1^2} \;=\; \operatorname{srank}(S)$$

whenever $\sum_{i=N-r+1}^{N} \lambda_i > 0$.

Proof. Write $\mathrm{Proj}_{\mathcal{U}} = VV^\top$ with $V \in \mathbb{R}^{N \times r}$ orthonormal and $V^\top u_1 = 0$. Then $\|S + \beta VV^\top\|_F^2 = \|S\|_F^2 + 2\beta \operatorname{tr}(SVV^\top) + \beta^2 \|VV^\top\|_F^2$ with $\operatorname{tr}(SVV^\top) = \sum_{k=1}^r v_k^\top S v_k$. Restricted Ky Fan yields $\min_{V^\top u_1 = 0} \operatorname{tr}(SVV^\top) = \sum_{i=N-r+1}^{N} \lambda_i$. Also $\|VV^\top\|_F^2 = \operatorname{tr}(VV^\top) = r$, and $\|S + \beta VV^\top\|_2 \leq \|S\|_2 + \beta$. □

D.2 Sample version and concentration

Assumption D.4 (Subgaussian columns; bounded cross term)**.** The columns p_i of P_l are independent, mean-zero, subgaussian with parameter κ and covariances Σ_i; hence $T_l = \mathbb{E}[\,P_l^\top P_l\,] = \mathrm{diag}(\mathrm{tr}\,\Sigma_1, \ldots, \mathrm{tr}\,\Sigma_N)$. Assume either: (i) $\mathbb{E}[\,D_l^\top P_l\,] = 0$; or (ii) $\|D_l^\top P_l\|_2 \le \gamma$ and $\|P_l^\top D_l\|_2 \le \gamma$ a.s., for some $\gamma \ge 0$.

Lemma D.5 (Matrix Bernstein concentration)**.** *Under Assumption D.4, for any $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$\left\| P_l^\top P_l - T_l \right\|_2 \;\le\; C_1(\kappa)\left(\sigma_{\max}\sqrt{N \log \tfrac{2N}{\delta}} \;+\; \sigma_{\max} \log \tfrac{2N}{\delta}\right),$$

where $\sigma_{\max} := \max_i \mathrm{tr}\,\Sigma_i$. Moreover, in case (ii), $\| D_l^\top P_l + P_l^\top D_l \|_2 \le 2\gamma$ deterministically.

Proposition D.6 (High-probability non-degenerate drift)**.** *Suppose CDI(α, β, r) holds for T_l and Assumption D.4 (case (i) or (ii)) holds. Then, for any $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$S_l' \;\succeq\; S_l \;+\; (\alpha - \varepsilon)\, I_N \;+\; (\beta - \varepsilon)\, \mathrm{Proj}_{\mathcal{U}} \;-\; 2\gamma\, I_N,$$

with $\varepsilon := C_1(\kappa)\Big(\sigma_{\max}\sqrt{N \log \tfrac{2N}{\delta}} + \sigma_{\max} \log \tfrac{2N}{\delta}\Big)$. Consequently,

$$\mathrm{tr}(S_l') \;\ge\; \mathrm{tr}(S_l) + N(\alpha - \varepsilon) + r(\beta - \varepsilon) - 2\gamma,$$

and both $\|S_l'\|_F$ and $\mathrm{srank}(S_l')$ exceed their pre-drift values as soon as α or β dominate the concentration/cross-term radii.

Proof. Combine equation 20, Lemma D.5, and CDI equation 21. For the trace and srank claims, apply Lemmas D.2–D.3, noting that $-2\gamma I_N \preceq D_l^\top P_l + P_l^\top D_l \preceq 2\gamma I_N$ in case (ii). □

D.3 A generative model that implies CDI

Proposition D.7 (Factor–plus–noise drift implies CDI)**.** *Let $P_l = U_r C + E$ where $U_r \in \mathbb{R}^{d_{\mathrm{out}} \times r}$ has orthonormal columns, $C \in \mathbb{R}^{r \times N}$, and $E = [e_1, \ldots, e_N]$ has independent mean-zero subgaussian columns with $\mathbb{E}[\,e_i e_i^\top\,] = \Sigma_e \succeq \sigma_s^2 I_{d_{\mathrm{out}}}$. Assume (i) $\frac{1}{N} C C^\top \succeq \beta I_r$ for some $\beta > 0$ and (ii) $U_r^\top u_1 = 0$ (i.e., the new factors are orthogonal to the pre-drift top mode). Then CDI holds with*

$$T_l = \mathbb{E}[\,P_l^\top P_l\,] \;\succeq\; N\sigma_s^2\, I_N \;+\; N\beta\, \mathrm{Proj}_{\mathcal{U}},$$

where $\mathcal{U} = \mathrm{row}(C)$ has dimension r. If additionally $\mathbb{E}[\,D_l^\top E\,] = 0$, the cross term is unbiased.

Proof. $\mathbb{E}[\,P_l^\top P_l\,] = C^\top C + \mathbb{E}[\,E^\top E\,] \succeq N\beta\, \mathrm{Proj}_{\mathrm{row}(C)} + N\sigma_s^2 I_N$. Orthogonality (ii) ensures incoherence with the pre-drift top direction u_1. □

D.4 Summary

Definitions D.1 and D.4, together with Lemmas D.2–D.3 and Proposition D.6, yield the main-text notion of *non-degenerate client drift*: with high probability,

$$\mathrm{tr}(S_l') > \mathrm{tr}(S_l) \quad \text{and} \quad \mathrm{srank}(S_l') > \mathrm{srank}(S_l),$$

ruling out trivial rescaling or purely top-aligned perturbations and capturing the structural effect of heterogeneity (energy increases and spreads across new directions).

E Proof of Thorem 4.5

Proof of Theorem 4.5. Fix a LoRA'd block l. At $\boldsymbol{\theta}^{(0)}$, the per-layer kernels (post-drift backprop, same lazy features) are

$$K_{\mathrm{FFT}}^{\prime(l)} \;=\; S_l' \circ G_l, \qquad K_{\mathrm{LoRA}}^{\prime(l)} \;=\; S_l' \circ \widetilde{G}_l,$$

where $S'_l(i,j) = \langle d'_{h_l}(i), d'_{h_l}(j)\rangle$, $G_l = X_l^\top X_l$ with $X_l = [x_{l,1}, \ldots, x_{l,N}]$, and $\widetilde{G}_l(i,j) = \langle \widetilde{A}_l x_{l,i}, \widetilde{A}_l x_{l,j}\rangle$ with $\widetilde{A}_l := A_l/\sqrt{r}$. Therefore

$$K'^{(l)}_{\mathrm{LoRA}} - K'^{(l)}_{\mathrm{FFT}} \;=\; S'_l \circ \Delta G_l, \qquad \Delta G_l \;:=\; \widetilde{G}_l - G_l \;=\; X_l^\top \big(\widetilde{A}_l^\top \widetilde{A}_l - I\big) X_l. \tag{23}$$

Step 1: OSE/JL bound on ΔG_l. Let $r_{x,l} = \mathrm{rank}(X_l)$. By a subspace Johnson–Lindenstrauss lemma, for subgaussian $\widetilde{A}_l$ and any $\delta_l \in (0,1)$, with probability $\geq 1 - \delta_l$,

$$\|\Delta G_l\|_2 \;=\; \big\|X_l^\top (\widetilde{A}_l^\top \widetilde{A}_l - I) X_l\big\|_2 \;\leq\; C\,\|X_l\|_2^2 \sqrt{\frac{r_{x,l}}{r}}. \tag{24}$$

Step 2: Hadamard–spectral bound per layer. For any symmetric positive semidefinite (PSD) S and any symmetric H, $\|S \circ H\|_2 \leq \max_i S_{ii}\, \|H\|_2 \leq \|S\|_2\, \|H\|_2$. Applying this with $S = S'_l$ and $H = \Delta G_l$, and using equation 24,

$$\big\|K'^{(l)}_{\mathrm{LoRA}} - K'^{(l)}_{\mathrm{FFT}}\big\|_2 \;\leq\; \|S'_l\|_2\, \|\Delta G_l\|_2 \;\leq\; C\,\|S'_l\|_2\, \|X_l\|_2^2 \sqrt{\frac{r_{x,l}}{r}}.$$

Step 3: Sum over LoRA'd blocks; Weyl. Summing the per-layer bounds and using the triangle inequality gives

$$\big\|K'_{\mathrm{LoRA}} - K'_{\mathrm{FFT}}\big\|_2 \;\leq\; \sum_{l \in \mathcal{L}_{\mathrm{LoRA}}} C\,\|S'_l\|_2\, \|X_l\|_2^2 \sqrt{\frac{r_{x,l}}{r}}.$$

Choosing r (or splitting δ) so that all events equation 24 hold simultaneously via a union bound yields the stated probability $1 - \delta$. Finally, Weyl's inequality gives

$$\lambda_{\min}(K'_{\mathrm{LoRA}}) \;\geq\; \lambda_{\min}(K'_{\mathrm{FFT}}) \;-\; \big\|K'_{\mathrm{LoRA}} - K'_{\mathrm{FFT}}\big\|_2,$$

which is equation 10.

Frobenius variant. Using $\|S \circ H\|_F \leq \|S\|_F\, \|H\|_\infty \leq \|S\|_F\, \|H\|_2$ and equation 24,

$$\big\|K'_{\mathrm{LoRA}} - K'_{\mathrm{FFT}}\big\|_F \;\leq\; \sum_{l \in \mathcal{L}_{\mathrm{LoRA}}} \|S'_l\|_F\, \|\Delta G_l\|_2 \;\leq\; \sum_{l \in \mathcal{L}_{\mathrm{LoRA}}} C\,\|S'_l\|_F\, \|X_l\|_2^2 \sqrt{\frac{r_{x,l}}{r}},$$

which is equation 11. □

F PSEUDO CODE FOR FEDLORE

G

Algorithm 1: FedLoRe: Client GaLore + Server JIVE (ADMM)

Init: Server sets per-layer shared projector $\{\boldsymbol{P}_{l,\text{srv}}^{(0)}\}$ (e.g., from a warmup batch or identity).

Each client c sets $\boldsymbol{P}_{c,l}^{(0)} \leftarrow \boldsymbol{P}_{l,\text{srv}}^{(0)}$ for all projected layers l.

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

  On each client c **(in parallel):**;

    for *local step* $\tau = 1, \ldots, E$ **do**

      Compute layerwise gradients $\boldsymbol{G}_{c,l}^{(t,\tau)}$.;

      `// GaLore step with projector refresh (RSVD)`

      Form $\tilde{\boldsymbol{G}}_{c,l}^{(t,\tau)} = \boldsymbol{G}_{c,l}^{(t,\tau)}\, \boldsymbol{P}_{c,l}^{(t)} \boldsymbol{P}_{c,l}^{(t)\top}$ and update params.;

      Periodically (every k steps): build $\boldsymbol{P}_{c,l}^{(t)}$ via RSVD on recent $\boldsymbol{G}_{c,l}^{(t,\tau)}$ sketches.;

    Package state dict $\theta_c^{(t)}$ and the last gradients $\{\boldsymbol{G}_{c,l}^{(t,E)}\}$; **return** to server.;

  On the server:;

    `// Stack per-layer client gradients and solve JIVE`

    For each layer l, collect $\{\boldsymbol{G}_{c,l}^{(t,E)}\}_{c=1}^{N}$.;

    Solve the convex JIVE program (nuclear-norm surrogate) with ADMM:

$$\min_{\boldsymbol{G}_l^*,\, \{\boldsymbol{L}_{c,l}\}} \frac{1}{2} \sum_c w_c \|\boldsymbol{G}_{c,l} - \boldsymbol{G}_l^* - \boldsymbol{L}_{c,l}\|_F^2 + \lambda_* \|\boldsymbol{G}_l^*\|_* + \lambda_{\text{ind}} \sum_c \|\boldsymbol{L}_{c,l}\|_*$$

    Each ADMM SVT subproblem uses truncated SVD via RSVD.;

    Extract shared projector $\boldsymbol{P}_{l,\text{srv}}^{(t+1)} \leftarrow \text{RSVD\_R}(\boldsymbol{G}_l^*, r)$ (right singular vectors).;

    broadcast $\{\boldsymbol{P}_{l,\text{srv}}^{(t+1)}\}_l$ to all clients; set $\boldsymbol{P}_{c,l}^{(t+1)} \leftarrow \boldsymbol{P}_{l,\text{srv}}^{(t+1)}$.;

    Aggregate model states (e.g., FedAvg) to form $\theta^{(t+1)}$ and redistribute.;