The Hidden Power of Scaling Factor in LoRA Optimization

Zicheng Zhang[1] **Haoran Li**[2,3] **Jiaxing Wang**[1,*] **Guoqiang Gong**[1] **Anqi Li**[4,*]
Yudong Hu[1] **Ting Xiong**[1] **Yurong Gao**[5] **Junxing Hu**[1] **Zhida Jiang**[1]
Yifeng Zhang[1] **Pengzhang Liu**[1] **Qixia Jiang**[1]

[1]JD [2]University of Waterloo [3]Vector Institute
[4]School of Mathematical Sciences, Nankai University
[5]School of Advanced Interdisciplinary Sciences, UCAS

Abstract

In Low-Rank Adaptation (LoRA), the scaling factor α is often treated as a mere complement to the learning rate, yet its role in optimization remains poorly understood. In this paper, we reveal that the scaling factor α and the learning rate function differently, with α emerging as the dominant driver of effective optimization, delivering gains that cannot be replicated by learning rate scaling alone. Through the synergy of extensive empirical analysis and a theoretical Signal-Drift framework, we uncover three findings into LoRA's scaling mechanism: First, LoRA's spectral suppression smooths the optimization landscape, rendering standard hyperparameters overly conservative and creating an optimization gap. Second, when leveraging this smoothness to accelerate convergence, α outperforms the learning rate by amplifying the task signal without increasing the drift ratio. Third, the optimal scaling factor follows a sublinear relationship with the rank, well characterized by a square-root law with an unexpectedly large coefficient, revealing the insufficient scaling of existing rank-tied heuristics. Based on these insights, we propose LoRA-α, a minimalist framework that restores α to its principled regime, making LoRA compatible with standard small learning rates. Extensive evaluations across diverse tasks demonstrate that LoRA-α consistently improves performance while streamlining hyperparameter search, unleashing the learning potential of LoRA.

1 Introduction

The rapid growth of large pretrained models [OpenAI Team, 2020, 2023; Meta Team, 2023; Qwen Team, 2023; DeepSeek Team, 2025] has made efficient adaptation a central challenge, driving the development of Parameter-Efficient Fine-Tuning (PEFT) methods [Mangrulkar et al., 2023; Lester et al., 2021; He et al., 2022; Edalati et al., 2023; Zhang et al., 2025a]. Among these, Low-Rank Adaptation (LoRA) [Hu et al., 2022] has emerged as a dominant approach due to its efficiency and stability. LoRA parameterizes weight updates as $\Delta W = \frac{\alpha}{r} BA$, where low-rank factors B and A approximate the update with rank r and scaling factor α. This simple formulation reinforced by framework support [Mangrulkar et al., 2023], has enabled widespread adoption from Natural Language Processing (NLP) [Liu et al., 2022; Ding et al., 2023; Zhao et al., 2024] to multimodal generation [Guo et al., 2024; Blattmann et al., 2023; Ruiz et al., 2023].

Despite its conceptual simplicity, the optimization behavior of LoRA remains poorly understood due to its inherent bilinear architecture and the complex interplay of hyperparameters. The scaling factor α, originating from feature learning principles [Yang & Hu, 2021], aims to decouple the optimal hyperparameters from the rank selection, thereby streamlining hyperparameter search. While

*Corresponding to wangjiaxing41@jd.com and anqili@nankai.edu.cn.

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

research has explored initialization [Wang et al., 2024b; Meng et al., 2024; Zhang et al., 2025b] and learning rates [Biderman et al., 2024; Schulman & Lab, 2025; Chen et al., 2026], the scaling factor α remains systematically underexplored, typically tethered to simplistic rank-based heuristics, such as $\alpha = r$ [Hu et al., 2022] or $2r$ [Biderman et al., 2024]. Consequently, α is often regarded as a secondary alternative to the learning rate for scaling updates, obscuring its pivotal role in reducing hyperparameter search and, more fundamentally, in shaping the underlying optimization regime.

The magnitude principle established in [Zhang et al., 2025b] highlights the importance of weight-update magnitude in LoRA optimization and connects scaling-factor adjustments to joint rescaling of learning rates and initialization. Building on this insight, we further investigate *how the scaling factor and learning rate differently shape LoRA optimization under fixed initialization through a joint empirical and theoretical study.* Across extensive hyperparameter sweeps, we consistently observe that effective optimization hinges more on a sufficiently large scaling factor α than on elevating the learning rate η. To understand this phenomenon, we develop a Signal-Drift framework that delineates LoRA's optimization characteristics. In this view, the task-relevant signal and the bilinear-induced drift respond differently to α and η, providing a principled explanation for our empirical observations. Together, these empirical and theoretical evidence lead to the following three key findings:

- **Spectral Suppression Causes a Scaling Misalignment.** We find that LoRA's low-rank parameterization induces a spectral suppression of the task Hessian, effectively smoothing the optimization landscape. While this improves stability, it also renders standard hyperparameters overly conservative, leading to a significant optimization gap and motivating the aggressive scaling practices observed in prior work [Hayou et al., 2024b; Schulman & Lab, 2025; Zhang et al., 2025b].
- α **and** η **Play Fundamentally Different Roles.** Empirically, we observe that increasing α consistently yields better convergence than increasing η. Our framework explains this by showing that α amplifies the task-aligned signal, whereas η amplifies both the signal and bilinear drift. As a result, α acts as a purity-preserving accelerator, enabling faster and more stable optimization.
- **Optimal Scaling Follows a Sublinear Law.** We identify a sublinear relationship between the optimal α and rank r, concisely characterized by a square-root law with a large scaling coefficient. This reveals that commonly used rank-tied heuristics [Hu et al., 2022; Biderman et al., 2024] operate in a severely under-scaled regime. With proper scaling, LoRA can directly adopt the standard small learning rates used in Full Fine-Tuning (FFT), while achieving superior performance.

Based on both empirical and theoretical findings, we identify a key limitation in prevailing LoRA practices: commonly used heuristics restrict α to insufficient magnitudes, limiting LoRA's optimization capacity. To address this, we propose LoRA-α, which introduces a large base value with a square-root scaling law to restore α to a principled regime. This formulation enables practitioners to bypass costly hyperparameter tuning and directly adopt standard FFT learning rates. Extensive experiments across model scales (184M–12B), task domains (natural language, reasoning, multimodal), and training paradigms (supervised, contrastive, and reinforcement learning) show that larger α with standard small η consistently yields superior performance. In summary, our contributions include:

- Empirically, we establish three key findings regarding LoRA's optimization through extensive hyperparameter sweeps. We uncover that performance hinges critically on a large scaling factor, exposing a pitfall in current rank-tied heuristics that leave LoRA's fitting potential underutilized.
- Theoretically, we develop a Signal-Drift framework to provide a principled explanation of these findings. By characterizing Hessian spectral suppression, we identify α as a better accelerator and derive a square-root law that aligns LoRA's optimization regime with that of FFT.
- We propose LoRA-α, a minimalist framework that elevates the scaling factor to its principled regime while adopting standard FFT learning rates. Across diverse models, tasks, and training paradigms, it significantly improves over LoRA, often reaching performance comparable to FFT.

2 An Empirical Study of LoRA Scaling

To study the scaling behavior of LoRA, we perform systematic hyperparameter sweeps to characterize the interplay between the scaling factor α, learning rate η, and rank r. We focus on *fitting behavior*, isolating optimization dynamics from generalization effects such as implicit low-rank regularization [Jang et al., 2024] and task diversity. This enables a clean analysis of the optimization landscape, while generalization is evaluated separately in Section 5.

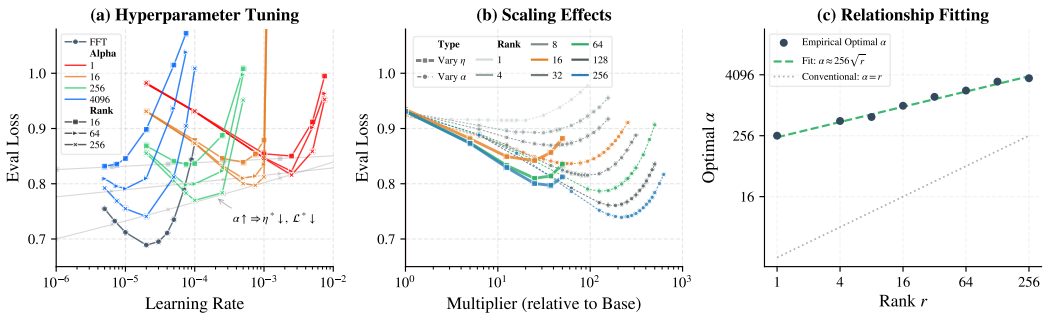

Figure 1: Hyperparameter analysis of Llama 3-1B on the Tulu 3 dataset. (a) Evaluation loss as a function of the learning rate η across different ranks r and scaling factors α. Gray lines denote linear fits of the minimum loss for each (r, α). Increasing α lowers the optimal loss and shifts η^* downward. (b) Scaling paths defined in Eq. (2) relative to the baseline ($\eta_{\text{FFT}} = 2 \times 10^{-5}$, $\alpha_0 = 16$), where η_{FFT} denotes the optimal FFT learning rate. Varying α reaches lower-loss regimes that are inaccessible via η-tuning alone. (c) Optimal scaling factor α^* as a function of rank r, evaluated at η_{FFT}. The observed sublinear trend and large magnitude of α challenge conventional scaling heuristics.

Optimization Setup and Metrics. Following Schulman & Lab [2025], we conduct Supervised Fine-Tuning (SFT) on two model scales (Llama 3-1B and 8B [Meta Team, 2024]) and datasets (Tulu 3 [Lambert et al., 2025] and OpenThoughts [Guha et al., 2026]), training each configuration for one epoch with a batch size of 32 using AdamW. For clarity, we report results on Llama 3-1B with Tulu 3, deferring the rest to Appendix D. To manage the large search space over α, η, and r, we use a 100k-example subset of Tulu 3 with a maximum token length $T = 1024$, and reserve 10k examples as a proxy evaluation set $\mathcal{D}_{prox}$. Let $\theta(r, \eta, \alpha)$ denote the trained parameters under a given configuration. We evaluate fitting via the Expected Negative Log-Likelihood (NLL):

$$\mathcal{L}(r, \eta, \alpha) = \mathbb{E}_{x \sim \mathcal{D}_{prox}} \left[-\Sigma_{t=1}^{T} \log P(x_t \mid x_{<t}; \theta(r, \eta, \alpha)) \right]. \tag{1}$$

Interplay between α and η. Let $\eta^*(r, \alpha) = \arg\min_\eta \mathcal{L}(r, \eta, \alpha)$ denote the optimal learning rate and $\mathcal{L}^*(r, \alpha) = \mathcal{L}(r, \eta^*, \alpha)$ the corresponding minimum loss. As illustrated in Fig. 1(a), we confirm the observation of Schulman & Lab [2025] that LoRA requires roughly $10\times$ larger learning rates than FFT at small α (*e.g.*, $\alpha = 16$). Nonetheless, our extended sweeps reveal a clear coupling: as α increases, η^* consistently decreases, while the optimal loss $\mathcal{L}^*$ improves monotonically. This gain becomes more pronounced at higher ranks (gray lines in Fig. 1(a)). *Finding I: The need for large learning rates in LoRA is a symptom of inadequate α-scaling, rather than an intrinsic property.*

Superiority of α-Scaling over η-Scaling. To disentangle the roles of the learning rate and scaling factor, we trace two independent scaling paths from the same baseline:

$$\mathcal{L}_\eta(m; r, \eta_{\text{FFT}}, \alpha_0) = \mathcal{L}(r, m\eta_{\text{FFT}}, \alpha_0), \quad \mathcal{L}_\alpha(m; r, \eta_{\text{FFT}}, \alpha_0) = \mathcal{L}(r, \eta_{\text{FFT}}, m\alpha_0), \tag{2}$$

where $\eta_{\text{FFT}} = 2 \times 10^{-5}$ is the optimal FFT learning rate, and $\alpha_0 = 16$. As illustrated in Fig. 1(b), although both trajectories exhibit U-shaped loss surfaces, they reveal a clear asymmetry. First, α-scaling accommodates a substantially larger optimal multiplier, indicating a more stable optimization regime. Second, while η-scaling prematurely plateaus, α-scaling consistently converges to deeper loss minima, corroborating the broad trends in Fig. 1(a). *Finding II: α is not merely an alternative to η for scaling step sizes; instead, it reshapes the optimization landscape to facilitate deeper fitting.*

Scaling Law between α and r. We examine the relationship between the rank and the optimal scaling factor, defined as $\alpha^*(r) = \arg\min_\alpha \mathcal{L}(r, \eta_{\text{FFT}}, \alpha)$. Fig. 1(c) shows that α^* follows a sublinear trend:

$$\alpha^*(r) \approx C\sqrt{r}, \quad C \gg 1, \tag{3}$$

with C obtained and rounded via log-scale fitting[1]. Across eight settings, we find that $C = 256$ provides a robust high-magnitude default. The setting-specific fits and sensitivity analysis are reported in Appendix D.4. This indicates that conventional choices are not only misaligned in functional form but also severely under-scaled in magnitude. *Finding III: Conventional scaling is mis-scaled, both in its linear dependence on r and its insufficient magnitude, thereby limiting LoRA's fitting effectiveness.*

[1] Estimated via linear regression in log space, *i.e.*, $\log \alpha = 0.5 \log r + \log C$

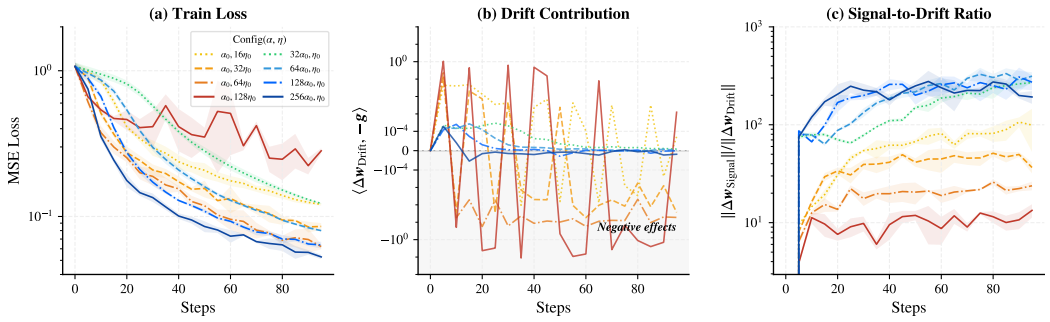

Figure 2: The asymmetric optimization dynamics of α-scaling versus η-scaling from a base configuration of rank $r = 16$, $\alpha_0 = 1$, and $\eta_0 = 10^{-4}$. We compare increasing η (warm colors) versus increasing α (cool colors). (a) η-scaling leads to early saturation and instability, while α-scaling enables smooth acceleration and deeper fitting. (b) Increasing η amplifies the pronounced stochasticity of structural drift, whereas α-scaling does not further amplify these fluctuations, maintaining the smoother optimization profile seen in (a). (c) α-scaling consistently maintains a higher Signal-to-Drift ratio, preserving update purity and accelerating convergence.

Relationship to Prior Work. Our findings unify and extend prior work on LoRA scaling. In terms of scaling magnitude, Hu et al. [2022] uses $\alpha = r$ by default, while Biderman et al. [2024] recommends increasing α but only explores $\alpha \leq 2r$. Our results show that substantially larger scaling can enable effective optimization with small learning rates, suggesting that the large learning rates commonly used for LoRA partly compensate for insufficient scaling [Schulman & Lab, 2025]. Although our rank-dependent scaling resembles RsLoRA [Kalajdzievski, 2023], the two approaches address different questions. RsLoRA is designed to ensure asymptotic stability as $r \to \infty$ and prevent gradient collapse, a concern that does not arise at practical low ranks, but it specifies only the scaling order, not the absolute magnitude needed for optimization. Consequently, its nominal scaling under-scales low-rank adapters (*e.g.*, approximately 1 at $r = 1$). Together with RsLoRA, our results support $\alpha \propto \sqrt{r}$ across rank regimes, while showing that rank stability and practical optimization are distinct considerations.

3 A Theoretical Analysis from the Signal-Drift Perspective

In this section, we formally characterize how LoRA reshapes optimization regime to ground our empirical observations. Our primary analytical approach is to introduce the Signal-Drift decomposition, which isolates the structural drift inherently induced by the adapter's bilinear parameterization.

Notations. Consider a pre-trained weight $W_0 \in \mathbb{R}^{d_{out} \times d_{in}}$ and trainable adapters $\theta = \{A, B\}$, where $B \in \mathbb{R}^{d_{out} \times r}$ and $A \in \mathbb{R}^{r \times d_{in}}$. The LoRA mapping is defined as $W(\boldsymbol{\theta}) = W_0 + \frac{\alpha}{r} BA$. Let $\boldsymbol{w}(\boldsymbol{\theta}) = \text{vec}(W(\boldsymbol{\theta})) \in \mathbb{R}^D$ denote the vectorized weight, and $\boldsymbol{\theta} = [\text{vec}(A); \text{vec}(B)] \in \mathbb{R}^p$ the concatenated parameters, where $D = d_{out} d_{in}$ and $p = r(d_{in} + d_{out})$. For the objective $\ell(\boldsymbol{w})$, we define the task gradient $\boldsymbol{g} = \nabla_{\boldsymbol{w}} \ell \in \mathbb{R}^D$ and the task Hessian $\mathbf{H}_\ell = \nabla^2_{\boldsymbol{w}} \ell \in \mathbb{R}^{D \times D}$, with g_k and w_k representing their k-th scalar entries. The mapping geometry is governed by the Jacobian $J(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \boldsymbol{w} \in \mathbb{R}^{D \times p}$ and the structural Hessian $\nabla^2_{\boldsymbol{\theta}} \boldsymbol{w} \in \mathbb{R}^{D \times p \times p}$, interpreted as a bilinear operator: for any $\boldsymbol{v} \in \mathbb{R}^p$, $\nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}[\boldsymbol{v}, \boldsymbol{v}] \in \mathbb{R}^D$ has entries $\boldsymbol{v}^\top (\nabla^2_{\boldsymbol{\theta}} w_k) \boldsymbol{v}$.

3.1 Decomposing the Optimization Dynamics

By parameterizing the objective as $\ell(\boldsymbol{w}(\boldsymbol{\theta}))$, LoRA optimization inevitably inherits artifacts from its bilinear architecture. Specifically, the dynamics couple with the mapping's intrinsic structure $\nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}$, prompting us to decouple the task-aligned projection signal from this task-agnostic structure drift.

Proposition 1 (Signal-Drift Decomposition)**.** *Let $\Delta \boldsymbol{w}_{LoRA} = \boldsymbol{w}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) - \boldsymbol{w}(\boldsymbol{\theta})$ denote the effective step in the weight space, and $\mathcal{H}_{LoRA} = \nabla^2_{\boldsymbol{\theta}} \ell$ denote the parameter-space Hessian. Under the LoRA parameterization, both admit an exact decomposition into a task-aligned signal and a structural drift:*

$$\Delta \boldsymbol{w}_{LoRA} = \underbrace{J(\theta)\Delta\boldsymbol{\theta}}_{\Delta \boldsymbol{w}_{Signal}} + \underbrace{\tfrac{1}{2} \nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}[\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\theta}]}_{\Delta \boldsymbol{w}_{Drift}}, \quad \mathcal{H}_{LoRA} = \underbrace{J(\theta)^\top \mathbf{H}_\ell J(\theta)}_{\mathcal{H}_{Signal}} + \underbrace{\Sigma_{k=1}^D g_k \nabla^2_{\boldsymbol{\theta}} w_k}_{\mathcal{H}_{Drift}}. \tag{4}$$

Algebraically, the components take the form $\Delta \boldsymbol{w}_{\text{Signal}} = \text{vec}\left(\frac{\alpha}{r}(B\Delta A + \Delta B A)\right)$ and the bilinear drift $\Delta \boldsymbol{w}_{\text{Drift}} = \text{vec}\left(\frac{\alpha}{r}\Delta B \Delta A\right)$, where ΔA and ΔB denote the updates within $\Delta\theta$.

Proposition 2 (Geometric Properties of Signal and Drift)**.** *For any gradient-based update step*[2] $\Delta\boldsymbol{\theta} = -\eta\nabla_{\boldsymbol{\theta}}\ell$*, the decomposed components in Proposition 1 satisfy:*

(i) Constructive Signal: *The signal component aligns with the descent direction,* $\langle \Delta \boldsymbol{w}_{\textit{Signal}}, -\boldsymbol{g}\rangle \geq 0$*, and preserves the local convexity of the loss landscape, i.e.,* $\mathcal{H}_{\textit{Signal}} \succeq 0$ *given* $\mathbf{H}_\ell \succeq 0$*.*

(ii) Adversarial Drift: *The structural Hessian* $\mathcal{H}_{\textit{Drift}}$ *is indefinite whenever the task gradient with respect to the adapted weight is nonzero, and its induced update* $\Delta \boldsymbol{w}_{\textit{Drift}}$ *exerts an uncontrolled force with no guaranteed alignment with* $-\boldsymbol{g}$*.*

The signal guarantees descent aligned with the task objective, whereas the drift introduces structural instability from the bilinear parameterization. Detailed proofs are deferred to Appendix G.

3.2 Theoretical Justifications for Empirical Findings

Based on the Signal-Drift decomposition, we theoretically justify the empirical phenomena observed in Section 2. For intuition, we utilize a simple MLP for visualization, circumventing the intractable $\mathcal{O}(D^2)$ complexity of tracking $\mathbf{H}_\ell$ and $\nabla^2_{\boldsymbol{\theta}}\boldsymbol{w}$ in large models. See Appendix E for details.

Justification for Finding I: Spectral Suppression Enables Aggressive Hyperparameters. Empirically, LoRA tolerates much larger learning rates than FFT without diverging. This enhanced stability stems from the inherent spectral suppression of the LoRA optimization landscape.

Proposition 3 (Spectral Suppression)**.** *Given the initial state of LoRA where* $B = 0$ *and* $A_{i,j} \sim \mathcal{N}(0, \sigma_A^2)$*, the expected signal curvature satisfies* $\mathbb{E}[\text{Tr}(\mathcal{H}_{\textit{Signal}})] = \alpha^2\rho\,\text{Tr}(\mathbf{H}_\ell)$*, where* $\rho = \sigma_A^2/r$*.*

Standard initialization like $\sigma_A^2 = 1/d_{in}$ yields $\rho \ll 1$, substantially compressing its maximum eigenvalue $\lambda_{\max}[\mathcal{H}_{\text{Signal}}]$ relative to $\mathbf{H}_\ell$. According to optimization theory [Nesterov, 2013], this reduced maximum eigenvalue lowers the gradient Lipschitz constant, thereby expanding the stable learning rate bound that $\eta \leq 2/\lambda_{\max}[\mathcal{H}_{\text{Signal}}]$. Consequently, spectral suppression smooths the landscape, rendering hyperparameters from FFT overly conservative and suboptimal for LoRA. This behavior is derived in Appendix Lemma 1 and further evidenced by spectral analysis in Fig. 9(a-b).

Justification for Finding II: The Asymmetric Roles of Scaling Factor and Learning Rate. To explain why α-scaling achieves superior fitting capacity compared to η-scaling, our decomposition reveals a fundamental asymmetry between the two hyperparameters.

Proposition 4 (Asymmetric Scaling Effects)**.** *In the local regime, the Signal and Drift terms scale as* $\mathcal{H}_{\text{Signal}} = \Theta(\alpha^2)$ *and* $\mathcal{H}_{\text{Drift}} = \Theta(\alpha)$*, respectively. Accordingly, the corresponding update norms* $(\|\Delta \boldsymbol{w}_{\text{Signal}}\|, \|\Delta \boldsymbol{w}_{\text{Drift}}\|)$ *scale as* $(\Theta(\alpha\eta), \Theta(\alpha\eta^2))$ *for Adam and as* $(\Theta(\alpha^2\eta), \Theta(\alpha^3\eta^2))$ *for GD.*

Corollary 1 (Different Drift under Matched Signal Amplification)**.** *Fix* $k > 1$ *and scale only one of* α *and* η*. At leading order, matching a* k*-fold amplification of the Signal update gives*

$$\textit{Adam:}\quad \alpha\textit{-scaling: } \alpha \mapsto k\alpha,\ \textit{Drift } \Theta(k); \qquad \eta\textit{-scaling: } \eta \mapsto k\eta,\ \textit{Drift } \Theta(k^2),$$

$$\textit{GD:}\quad \alpha\textit{-scaling: } \alpha \mapsto \sqrt{k}\alpha,\ \textit{Drift } \Theta(k^{3/2}); \qquad \eta\textit{-scaling: } \eta \mapsto k\eta,\ \textit{Drift } \Theta(k^2).$$

This asymmetry dictates their distinct optimization roles. Specifically, increasing η strictly degrades the Signal-to-Drift Ratio, defined as $\|\Delta \boldsymbol{w}_{\text{Signal}}\|/\|\Delta \boldsymbol{w}_{\text{Drift}}\|$. This degradation manifests empirically as the drift-induced volatility and premature saturation observed in Fig. 2 for Adam and Fig. 8 for SGD. Conversely, α-scaling enables spectral purification. As validated in Fig. 9(c-d), increasing α reshapes the LoRA landscape $\mathcal{H}_{\text{LoRA}}$ to recover the spectral profile of signal term.

Justification for Finding III: Conventional Scaling is Fundamentally Mis-scaled. Empirically, the optimal α is exceptionally large and scales sublinearly with rank r. This follows from Proposition 3: to counteract spectral suppression and restore curvature, the adapter's expected signal curvature must match the FFT curvature scale.

Corollary 2 (Initial Curvature Alignment)**.** *To match the signal curvature to the full task curvature at initialization, i.e.,* $\mathbb{E}[\text{Tr}(\mathcal{H}_{\textit{Signal}})] = \Theta\big(\text{Tr}(\mathbf{H}_\ell)\big)$*, the scaling factor follows* $\alpha = \Theta\big(\sqrt{r/\sigma_A^2}\big)$*.*

[2]This conceptually extends to adaptive optimizers (*e.g.*, Adam, Muon) operating via preconditioned gradients.

This derivation proves that conventional linear scaling is mis-specified. Instead, α should scale sublinearly with rank and grow substantially with model width under standard initialization.

4 Practical Implications and the LoRA-α Protocol

Our empirical and theoretical analysis suggests two practical rules: First, keep η near the standard FFT learning rate and allocate the primary search budget to α, which amplifies Signal without disproportionately increasing bilinear Drift. Second, when search is limited, initialize from a calibrated scale $\alpha_{\text{base}} = C\sqrt{r}$ rather than from conventional rank-tied heuristics.

These rules motivate *LoRA-α*, a tuning protocol. We factor the conventional scaling parameter into a calibrated base scale α_{base} and a residual multiplier α:

$$\Delta W = \frac{\alpha \cdot \alpha_{\text{base}}}{r} BA, \quad \text{with } \eta = \Theta(\eta_{\text{FFT}}). \tag{5}$$

We set $\alpha = 1$ by default and, when tuning is desired search over a compact neighborhood (*e.g.*, $[0.1, 10]$). We propose two complementary strategies for setting α_{base}:

Variant I: Empirical Scaling for LLMs. Contemporary LLMs commonly share Transformer-based architectures, with hidden dimensions typically on the order of thousands. In practice, LoRA is often applied to the attention and MLP projection matrices. Motivated by the scaling law in Section 2, we use $C = 256$ as a practical default for the evaluated settings, giving $\alpha_{\text{base}} = 256\sqrt{r}$.

Variant II: Analytic Scaling. For general architectures, we derive a principled scaling from Corollary 2 by matching the expected task curvature at initialization: $\alpha_{\text{base}} = \frac{1}{\sigma_A}\sqrt{r}$. This closed-form solution provides a theoretical baseline that aligns LoRA with the hyperparameter regime of FFT.

Importantly, Variant I serves as a *global empirical heuristic*, while Variant II provides a *layer-wise theoretical initialization*. In practice, the empirical scaling in Variant I yields marginally larger magnitudes than the analytic prediction of Variant II for modern LLMs. A detailed magnitude comparison is provided in Appendix F. We identify this gap as an important empirical finding. A plausible explanation is that the analytic scaling matches the average curvature via the trace of the Hessian, whereas optimization stability is governed by the largest eigenvalue. As shown in Fig. 9(a-b), these quantities can differ within a moderate range, making such discrepancies reasonable. Ultimately, this principled choice resolves the $16\times$–$256\times$ under-scaling inherent in prevailing heuristics.

5 Empirical Validation of LoRA-α

To rigorously evaluate the effectiveness of LoRA-α, we conduct extensive experiments along three dimensions: (i) Model diversity, covering encoder, decoder, diffusion, and vision-language architectures; (ii) Task heterogeneity, including natural language and multimodal tasks, encompassing cross entropy, reward-based, flow matching and contrastive learning objectives; (iii) Training horizon, ranging from few-shot adaptation to long-horizon optimization. This comprehensive design validates LoRA-α across both adaptation benchmarks and post-training scenarios. Specifically, we verify the effectiveness of *Analytic Scaling* on diverse architectures in Section 5.1, while adopting *Empirical Scaling* as the default configuration for the remaining LLM-centric experiments. All experiments are implemented using the PEFT library [Mangrulkar et al., 2023] and executed on eight NVIDIA H800 GPUs. Due to space limitations, we briefly present the core settings and main results in the following subsections. Comprehensive implementation details are deferred to Appendix H.

5.1 Performance on Fundamental Adaptation Tasks

We evaluate short-horizon adaptation within a few hours across both small- and large-scale models. We benchmark our approach against vanilla LoRA [Hu et al., 2022] ($\alpha = r$) and representative methods, including scaling-based RsLoRA [Kalajdzievski, 2023], learning-rate-modified LoRA+ [Hayou et al., 2024b], spectral-initialized PiSSA [Meng et al., 2024], and magnitude-initialized LoRAM [Zhang et al., 2025b]. To ensure a fair comparison, we adopt the baseline results from Zhang et al. [2025b], independently training our method under identical settings.

Natural Language Understanding (NLU). At first, we fine-tune and evaluate DeBERTa-v3-base-184M [He et al., 2023] on eight datasets from the GLUE benchmark [Wang et al., 2018]. All methods

Table 1: Comparison on the GLUE benchmark. Experiments are conducted using the DeBERTa-v3-base-184M model with a base learning rate of 1×10^{-4} and a rank of 8. Asterisk ($*$) denotes optimal results from a $0.1\times$–$10\times$ sweep over η or α. Within each block, best and second-best values, determined before rounding, are bold and underlined; no second-best is marked when the best is tied.

Method	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B	Avg.
Full Fine-Tuning	88.3	93.6	89.5	67.3	92.8	91.5	83.8	86.9	86.7
LoRA	90.2	**95.9**	84.1	63.6	93.9	90.6	50.2	87.2	81.9
RsLoRA	90.3	95.6	86.4	64.9	94.0	90.3	60.3	88.4	83.8
LoRA+	90.4	95.3	87.5	64.8	**94.3**	90.9	65.4	89.2	84.7
PiSSA	**90.4**	95.6	89.2	65.1	93.8	91.4	74.4	88.9	86.1
LoRAM	90.3	95.3	**90.0**	65.5	94.1	**91.7**	74.7	89.9	86.4
LoRA-α	**90.4**	95.8	**90.0**	**70.9**	94.0	**91.7**	**80.9**	**90.1**	**88.0**
LoRA-η^*	90.7	96.2	**91.1**	69.3	**94.1**	91.8	84.5	90.7	88.5
LoRA-α^*	**90.8**	**96.4**	90.4	**70.9**	**94.1**	**92.1**	**87.0**	**91.1**	**89.1**

Table 2: Comparison on NLG tasks using Llama 2-7B tuned with a base learning rate of 2×10^{-5}. Bold and underlined values denote the best and second-best results determined before rounding.

Rank (r)	Method	GSM8K	MATH	HumanEval	MBPP	Commonsense	Avg.
N/A	Full Fine-Tuning	60.3	11.7	32.3	39.3	79.2	44.6
16	LoRA	31.5	4.2	16.0	28.7	66.6	29.4
	RsLoRA	39.0	4.9	18.9	28.1	73.2	32.8
	LoRA+	31.7	4.0	18.5	28.0	72.2	30.9
	PiSSA	37.7	5.2	18.4	28.6	73.7	32.7
	LoRAM	40.3	5.3	**18.9**	**28.8**	75.2	33.7
	LoRA-α	**47.6**	**6.8**	**18.9**	28.3	**77.4**	**35.8**
	LoRA-η^*	54.3	9.5	24.4	34.7	**78.2**	40.2
	LoRA-α^*	**55.3**	**9.8**	**28.7**	**35.4**	**78.2**	**41.5**
128	LoRA	40.3	4.7	20.1	28.8	73.6	33.5
	RsLoRA	50.4	7.3	21.3	30.7	77.0	37.4
	LoRA+	40.4	5.3	20.7	29.1	78.2	34.7
	PiSSA	51.5	7.0	21.6	31.1	77.3	37.7
	LoRAM	51.1	7.3	22.0	31.5	77.8	38.0
	LoRA-α	**53.2**	**8.3**	**23.2**	**33.9**	**78.4**	**39.4**
	LoRA-η^*	58.8	10.8	28.7	36.5	78.7	42.7
	LoRA-α^*	**60.0**	**11.7**	**33.5**	**39.9**	**79.3**	**44.9**

are trained with a batch size of 32, learning rate of 1×10^{-4}, and LoRA rank $r = 8$ for 3–5 epochs. As shown in Table 1, LoRA-α achieves the strongest or near-strongest performance across tasks, outperforming both vanilla LoRA and alternative scaling strategies in average performance.

Natural Language Generation (NLG). We then fine-tune Llama 2-7B [Meta Team, 2023] on three datasets covering mathematical reasoning (MetaMathQA), code generation (CodeFeedback), and commonsense reasoning (Commonsense170K), respectively. Training is conducted with a batch size of 128, learning rate of 2×10^{-5}, and LoRA ranks $r \in \{16, 128\}$ for one epoch over 100k samples. As shown in Table 2, LoRA-α matches or surpasses existing LoRA variants across all tasks and rank settings. The advantage becomes more pronounced at higher rank of 128, indicating that proper α scaling is increasingly critical as model capacity grows.

Text-to-Image Synthesis. To validate scalability to multimodal generative settings, we fine-tune Flux.1-12B [Labs, 2024] using the DreamBooth protocol [Ruiz et al., 2023]. All methods are trained with a batch size of 1, learning rate of 1×10^{-4}, and LoRA rank $r = 8$ for 1,000 training steps, supervised by flow matching loss. As illustrated in Fig. 3, LoRA-α produces images with higher fidelity to reference concepts compared to PiSSA and LoRAM, demonstrating that improved scaling generalizes beyond language tasks to complex generative domains.

Comparison of α-scaling and η-scaling. We perform hyperparameter sweeps over both α and η on NLU and NLG tasks, with full results provided in Appendix H. As reported in Tables 1 and 2, α-scaling consistently yields superior performance compared to η-scaling. Moreover, the performance gap widens significantly at larger ranks, corroborating the initial scaling findings from Fig. 1.



Figure 3: Qualitative comparison of image customization on Flux.1-12B with a base learning rate of 1×10^{-4} and a rank of 8. LoRA-α consistently achieves higher fidelity to reference objects.

Table 3: Comparison on the MMEB benchmark via post-tuning Qwen 2-VL ($r = 16$, base $\eta = 2 \times 10^{-5}$). "VLM2VEC" denotes officially released results. IND and OOD indicate in- and out-of-distribution splits, respectively.

Model	Method	Meta-Task Average Score				Average Score		
		Classification	VQA	Retrieval	Grounding	IND	OOD	Overall
	# of datasets →	10	10	12	4	20	16	36
2B	VLM2Vec	59.0	49.4	65.4	73.4	66.0	52.6	60.1
	Full Fine-Tuning	58.3	48.9	64.8	74.5	**67.2**	50.3	59.6
	LoRA	58.7	47.2	64.5	70.5	64.0	52.1	58.7
	DoRA	57.3	43.2	60.8	67.0	61.3	48.6	55.6
	PiSSA	52.5	44.2	61.4	66.1	59.3	48.9	54.7
	LoRA (10× LR)	59.0	50.9	65.8	73.0	66.9	52.5	60.5
	LoRA-α	**60.2**	**52.3**	**66.0**	**75.2**	**67.2**	**54.7**	**61.6**
7B	VLM2Vec	62.6	57.8	69.9	81.7	72.7	57.8	65.8
	LoRA	60.7	57.3	68.1	81.2	70.7	56.8	64.5
	DoRA	61.6	54.2	66.5	78.2	69.0	55.6	63.0
	PiSSA	58.2	55.0	65.4	77.9	67.9	54.4	61.9
	LoRA (10× LR)	62.0	58.9	69.3	81.3	72.6	55.4	65.3
	LoRA-α	**63.0**	**59.4**	**70.6**	**84.2**	**73.3**	**58.8**	**66.9**

5.2 Performance on Multimodal Representation Learning

Moving beyond short-horizon tasks, we evaluate LoRA-α in a long-horizon representation learning setting by transforming a multimodal LLM into a dense retriever on the MMEB benchmark [Jiang et al., 2025]. Following VLM2Vec [Jiang et al., 2025], we train Qwen 2-VL (2B and 7B) using an InfoNCE objective, with a batch size of 1024, learning rate of 2×10^{-5}, maximum sequence length of 4096 tokens, and LoRA rank of $r = 16$, taking $\sim$3 days per run. We compare LoRA-α against standard LoRA, LoRA with $10\times$ learning rate [Schulman & Lab, 2025], DoRA [Liu et al., 2024], PiSSA, and FFT, where FFT is only feasible for the 2B model due to memory constraints.

Results. As shown in Table 3, LoRA-α consistently outperforms all PEFT baselines and FFT across both model scales. Notably, it surpasses the best checkpoint from official VLM2Vec, indicating that improved scaling alone can unlock additional performance without architectural modifications. The largest gains are observed on OOD evaluation, where LoRA-α significantly improves over FFT and VLM2Vec, demonstrating stronger robustness under distribution shifts.

Table 4: Comparison of Pass@1 accuracy across reasoning benchmarks. We train Qwen 2.5-Math-7B with a base learning rate of 4×10^{-5}. n denotes the number of samples used for Pass@1 estimation.

Rank (r)	**Method**	**AIME 24** ($n = 64$)	**AIME 25** ($n = 64$)	**MATH-500** ($n = 4$)	**GPQA Dia.** ($n = 8$)	**LCB v4** ($n = 16$)	**Avg.**
N/A	Full Fine-Tuning	37.0	28.7	86.5	39.4	28.7	44.1
64	LoRA	23.1	19.7	83.0	34.6	14.9	35.1
	LoRA (10× LR)	28.3	**25.8**	**86.0**	**37.8**	**19.8**	**39.6**
	LoRA-α	**28.7**	**25.8**	85.8	37.0	**19.8**	39.4
256	LoRA	26.5	23.7	84.6	35.5	20.8	38.2
	LoRA (10× LR)	33.7	26.7	83.0	38.3	22.8	40.9
	LoRA-α	**34.2**	**29.0**	**85.8**	**38.5**	**26.7**	**42.8**

Table 5: Comparison of Pass@1 accuracy on reasoning benchmarks after GRPO learning. We train DeepSeek-R1-Distill-Qwen with a base learning rate of 1×10^{-6} and a fixed LoRA rank of 64.

Model	**Method**	**AIME 24** ($n = 64$)	**AIME 25** ($n = 64$)	**MATH-500** ($n = 4$)	**AMC 23** ($n = 32$)	**MINERVA** ($n = 8$)	**HMMT** ($n = 32$)	**Avg.**
1.5B	Base model	21.5	13.3	31.3	67.5	13.3	**32.4**	29.9
	Full Fine-Tuning	29.8	22.5	**82.0**	62.5	**31.3**	6.7	39.1
	LoRA	22.3	6.7	32.4	**72.5**	27.6	13.3	29.1
	LoRA (10× LR)	22.3	10.0	79.4	65.0	29.0	13.3	36.5
	LoRA-α	**32.3**	**22.8**	79.8	70.0	29.4	13.3	**41.3**
7B	Base model	51.8	**37.5**	86.6	90.0	40.8	**16.7**	53.9
	Full Fine-Tuning	39.0	16.7	42.7	**92.5**	40.8	**16.7**	41.4
	LoRA	51.8	37.3	87.2	90.0	**41.5**	13.3	53.5
	LoRA (10× LR)	53.3	36.5	87.6	87.5	40.1	13.3	53.1
	LoRA-α	**55.5**	35.6	**88.6**	90.0	**41.5**	**16.7**	**54.7**

5.3 Performance on Reasoning-based Supervised Fine-Tuning

We evaluate LoRA-α in a long-context reasoning setting. Following [HuggingFace Team, 2025], we post-tune Qwen 2.5-Math-7B [Qwen Team, 2024] on the Mixture-of-Thoughts dataset including 350k reasoning traces across math, programming, and science with a 32k token context. Training uses a batch size of 128, a learning rate of 4×10^{-5}, and LoRA ranks $r \in \{64, 256\}$ for one epoch, taking $\sim$2 days per run. We compare FFT, standard LoRA, LoRA with $10\times$ learning rate, and LoRA-α on AIME 2024/2025, MATH-500, GPQA Diamond, and LiveCodeBench v4 using Pass@1.

Results. As shown in Table 4, LoRA-α performs comparably to the $10\times$ learning rate variant at smaller ranks, but achieves stronger gains as the rank increases. In contrast, learning-rate scaling becomes unstable at higher ranks. Notably, due to the intrinsic difficulty of this task, LoRA-based methods do not surpass FFT, reflecting the substantial learning capacity required. Nevertheless, LoRA-α consistently emerges as the closest approximation to FFT.

5.4 Performance on Reinforcement Learning Paradigm

We evaluate LoRA-α under the reinforcement learning using Group Relative Policy Optimization (GRPO) [DeepSeek Team, 2025]. Following [Yin et al., 2025], experiments are conducted on DeepSeek-R1-Distill-Qwen (1.5B and 7B) models, trained on the DAPO-Math-17k dataset [Yu et al., 2025] with a LoRA rank of 64, learning rate of 1×10^{-6} and batch size of 128. We use a maximum completion length of 16k tokens with $G = 8$ rollouts per prompt, taking $\sim$2 days per run. Evaluation is conducted on a suite of challenging mathematical benchmarks, using Pass@1 as the primary metric.

Results. As shown in Table 5, LoRA-α achieves the best overall performance across both model scales. In contrast, increasing the learning rate fails to yield reliable gains and often degrades performance, highlighting the instability induced by high-variance policy gradients. FFT also exhibits inconsistent behavior, with noticeable degradation on several benchmarks, suggesting that updating all parameters can disrupt pretrained reasoning structures. Combined with the results in Table 4, these findings echo the observations of [Schulman & Lab, 2025] and suggest a broader implication: compared to SFT, reinforcement learning typically requires lower learning capacity while being more susceptible to optimization instability. In this regime, LoRA is better suited than FFT, and properly scaling α preserves training stability and improves learning effectiveness.

6 Conclusion

This paper investigates the scaling mechanisms of LoRA, combining extensive empirical evidence with a novel Signal-Drift framework. We show that LoRA's stability stems from spectral suppression of the task Hessian, which creates a significant optimization gap. Crucially, we demonstrate that scaling α is more effective than increasing the learning rate, as it amplifies task-aligned signal without introducing additional drift. Building on this insight, we establish a sublinear scaling law and propose LoRA-α, which restores α to its principled regime and enables direct reuse of FFT learning rates to substantially improve LoRA performance. Overall, our work provides both theoretical insight and practical guidance for unlocking LoRA's potential through a simple yet effective scaling mechanism.

Looking ahead, developing novel adapter architectures that structurally mitigate this drift without compromising task-signal integrity remains an exciting frontier for future research.

Acknowledgments

Anqi Li acknowledges funding support from the Tianyuan Fund Project for Mathematical Sciences of the National Natural Science Foundation of China (Nos. 12526588) and the Fundamental Research Funds for the Central Universities (Nos. 63261074).

References

[1] J Austin, A Odena, M Nye, M Bosma, H Michalewski, D Dohan, E Jiang, C Cai, M Terry, Q Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[2] M Balunović, J Dekoninck, I Petrov, N Jovanović, and M Vechev. Matharena: Evaluating llms on uncontaminated math competitions. In *NeurIPS*, 2025.

[3] D Biderman, J Portes, J J G Ortiz, M Paul, P Greengard, C Jennings, D King, S Havens, V Chiley, J Frankle, et al. Lora learns less and forgets less. *TMLR*, 2024.

[4] A Blattmann, T Dockhorn, S Kulal, D Mendelevitch, M Kilian, D Lorenz, Y Levi, Z English, V Voleti, A Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[5] N Chen, S Villar, and S Hayou. Learning rate scaling across lora ranks and transfer to full finetuning. *arXiv preprint arXiv:2602.06204*, 2026.

[6] K Cobbe, V Kosaraju, M Bavarian, M Chen, H Jun, L Kaiser, M Plappert, J Tworek, J Hilton, R Nakano, C Hesse, and J Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[7] DeepSeek Team. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[8] N Ding, Y Qin, G Yang, F Wei, Z Yang, Y Su, S Hu, Y Chen, C M Chan, W Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 2023.

[9] A Edalati, M S Tahaei, I Kobyzev, V Nia, J J Clark, and M Rezagholizadeh. Krona: Parameter efficient tuning with kronecker adapter. In *NeurIPS Workshop*, 2023.

[10] E Guha, R Marten, S Keh, N Raoof, G Smyrnis, H Bansal, M Nezhurina, J Mercat, T Vu, Z Sprague, et al. Openthoughts: Data recipes for reasoning models. In *ICLR*, 2026.

[11] Y Guo, C Yang, A Rao, Z Liang, Y Wang, Y Qiao, M Agrawala, D Lin, and B Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. In *ICLR*, 2024.

[12] N Habib, C Fourrier, H Kydlíček, T Wolf, and L Tunstall. Lighteval: A lightweight framework for llm evaluation, 2023. URL `https://github.com/huggingface/lighteval`.

[13] S Hayou, N Ghosh, and B Yu. The impact of initialization on lora finetuning dynamics. In *NeurIPS*, 2024a.

[14] S Hayou, N Ghosh, and B Yu. Lora+: Efficient low rank adaptation of large models. In *ICML*, 2024b.

[15] H He, J Ye, M Li, Z Wang, T Chen, L Bai, and P Ye. A unified study of lora variants: Taxonomy, review, codebase, and empirical evaluation. *arXiv preprint arXiv:2601.22708*, 2026.

[16] J He, C Zhou, X Ma, T Berg-Kirkpatrick, and G Neubig. Towards a unified view of parameter-efficient transfer learning. In *ICLR*, 2022.

[17] P He, J Gao, and W Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. In *ICLR*, 2023.

[18] D Hendrycks, C Burns, S Kadavath, A Arora, S Basart, E Tang, D Song, and J Steinhardt. Measuring mathematical problem solving with the math dataset. In *NeurIPS*, 2021.

[19] E J Hu, Y Shen, P Wallis, Z Allen-Zhu, Y Li, S Wang, and W Chen. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022.

[20] Z Hu, L Wang, Y Lan, W Xu, E P Lim, L Bing, X Xu, S Poria, and R Lee. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *EMNLP*, 2023.

[21] HuggingFace Team. Open r1: A fully open reproduction of deepseek-r1, 2025. URL `https://github.com/huggingface/open-r1`.

[22] N Jain, K Han, A Gu, W D Li, F Yan, T Zhang, S Wang, A Solar-Lezama, K Sen, and I Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *ICLR*, 2025.

[23] U Jang, J D Lee, and E K Ryu. Lora training in the NTK regime has no spurious local minima. In *ICML*, 2024.

[24] T Jiang, S Huang, S Luo, Z Zhang, H Huang, F Wei, W Deng, F Sun, Q Zhang, D Wang, et al. Mora: High-rank updating for parameter-efficient fine-tuning. *arXiv preprint arXiv:2405.12130*, 2024.

[25] Z Jiang, R Meng, X Yang, S Yavuz, Y Zhou, and W Chen. VLM2vec: Training vision-language models for massive multimodal embedding tasks. In *ICLR*, 2025.

[26] D Kalajdzievski. A rank stabilization scaling factor for fine-tuning with lora. *arXiv preprint arXiv:2312.03732*, 2023.

[27] W Kwon, Z Li, S Zhuang, Y Sheng, L Zheng, C H Yu, J E Gonzalez, H Zhang, and I Stoica. Efficient memory management for large language model serving with pagedattention. In *ACM SOSP*, 2023.

[28] B F Labs. Flux, 2024. URL `https://github.com/black-forest-labs/flux`.

[29] N Lambert, J Morrison, V Pyatkin, S Huang, H Ivison, F Brahman, L J V Miranda, A Liu, N Dziri, S Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. In *COLM*, 2025.

[30] S Lee and J Lee. Beware of the batch size: Hyperparameter bias in evaluating lora. *arXiv preprint arXiv:2602.09492*, 2026.

[31] Y A Lee, C Y Ko, P Y Chen, and M Y Yeh. Learning rate matters: Vanilla lora may suffice for llm fine-tuning. *arXiv preprint arXiv:2602.04998*, 2026.

[32] B Lester, R Al-Rfou, and N Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP*, 2021.

[33] A Lewkowycz, A Andreassen, D Dohan, E Dyer, H Michalewski, V Ramasesh, A Slone, C Anil, I Schlag, T Gutman-Solo, et al. Solving quantitative reasoning problems with language models. In *NeurIPS*, 2022.

[34] B Li, L Zhang, A Mokhtari, and N He. On the crucial role of initialization for matrix factorization. In *ICLR*, 2025.

[35] V Lialin, S Muckatira, N Shivagunde, and A Rumshisky. Relora: High-rank training through low-rank updates. In *ICLR*, 2024.

[36] H Liu, D Tam, M Muqeeth, J Mohta, T Huang, M Bansal, and C A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *NeurIPS*, 2022.

[37] J Liu, Z Kong, P Dong, X Shen, P Zhao, H Tang, G Yuan, W Niu, W Zhang, X Lin, D Huang, and Y Wang. Rora: Efficient fine-tuning of llm with reliability optimization for rank adaptation. In *ICASSP*, 2025.

[38] S Y Liu, C Y Wang, H Yin, P Molchanov, Y C F Wang, K T Cheng, and M H Chen. Dora: Weight-decomposed low-rank adaptation. In *ICML*, 2024.

[39] MAA. Amc 2023 problems, 2023. URL https://artofproblemsolving.com/wiki/index.php/2023_AMC_12A_Problems.

[40] MAA. Aime 2024 problems, 2024. URL https://artofproblemsolving.com/wiki/index.php/2024_AIME_I_Problems.

[41] MAA. Aime 2025 problems, 2025. URL https://artofproblemsolving.com/wiki/index.php/2025_AIME_I_Problems.

[42] S Malladi, A Wettig, D Yu, D Chen, and S Arora. A kernel-based view of language model fine-tuning. In *ICML*, 2023.

[43] S Mangrulkar, S Gugger, L Debut, Y Belkada, S Paul, and B Bossan. Peft: state-of-the-art parameter-efficient fine-tuning methods, 2023. URL https://github.com/huggingface/peft.

[44] Y Mao, Y Ge, Y Fan, W Xu, Y Mi, Z Hu, and Y Gao. A survey on lora of large language models. *Frontiers of Computer Science*, 2025.

[45] F Meng, Z Wang, and M Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. In *NeurIPS*, 2024.

[46] Meta Team. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[47] Meta Team. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[48] S Mu and D Klabjan. On the convergence rate of lora gradient descent. *arXiv preprint arXiv:2512.18248*, 2025.

[49] Y Nesterov. *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013.

[50] OpenAI Team. Language models are few-shot learners. In *NeurIPS*, 2020.

[51] OpenAI Team. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[52] Qwen Team. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[53] Qwen Team. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

[54] D Rein, B L Hou, A C Stickland, J Petty, R Y Pang, J Dirani, J Michael, and S R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *COLM*, 2024.

[55] N Ruiz, Y Li, V Jampani, Y Pritch, M Rubinstein, and K Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *CVPR*, 2023.

[56] J Schulman and T M Lab. Lora without regret. Thinking Machines Lab: Connectionism, 2025. URL https://thinkingmachines.ai/blog/lora.

[57] A Wang, A Singh, J Michael, F Hill, O Levy, and S R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *EMNLP Workshop*, 2018.

[58] P Wang, S Bai, S Tan, S Wang, Z Fan, J Bai, K Chen, X Liu, J Wang, W Ge, Y Fan, K Dang, M Du, X Ren, R Men, D Liu, C Zhou, J Zhou, and J Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024a.

[59] S Wang, L Yu, and J Li. Lora-ga: Low-rank adaptation with gradient approximation. In *NeurIPS*, 2024b.

[60] W Wang, Q Liu, and Mind Lab. 0.03 parameters, 100 potential: Reflexivity of compute optimal rank scaling. Mind Lab: A Lab for Experiential Intelligence, 2026. URL https://macaron.im/mindlab/research/003-parameters-100-potential-reflexivity-of-compute-optimal-rank-scaling.

[61] W Xia, C Qin, and E Hazan. Chain of lora: Efficient fine-tuning of language models via residual learning. In *ICML*, 2024.

[62] Z Xu, H Min, L E MacDonald, J Luo, S Tarmoun, E Mallada, and R Vidal. Understanding the learning dynamics of lora: A gradient flow perspective on low-rank adaptation in matrix factorization. In *AISTATS*, 2025.

[63] G Yang and E J Hu. Feature learning in infinite-width neural networks. In *ICML*, 2021.

[64] M Yang, J Chen, Y Zhang, J Liu, J Zhang, Q Ma, H Verma, Q Zhang, M Zhou, I King, et al. Low-rank adaptation for foundation models: A comprehensive review. *arXiv preprint arXiv:2501.00365*, 2024.

[65] Q Yin, Y Wu, Z Shen, S Li, Z Wang, Y Li, C T Leong, J Kang, and J Gu. Evaluating parameter efficient methods for rlvr. *arXiv preprint arXiv:2512.23165*, 2025.

[66] L Yu, W Jiang, H Shi, J Yu, Z Liu, Y Zhang, J T Kwok, Z Li, A Weller, and W Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *ICLR*, 2024.

[67] Q Yu, Z Zhang, R Zhu, Y Yuan, X Zuo, Y Yue, W Dai, T Fan, G Liu, L Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. In *NeurIPS*, 2025.

[68] D Zhang, T Feng, L Xue, Y Wang, Y Dong, and J Tang. Parameter-efficient fine-tuning for foundation models. *arXiv preprint arXiv:2501.13787*, 2025a.

[69] Z Zhang, H Li, Y Zhang, G Gong, J Wang, J Hu, P Liu, and Q Jiang. The primacy of magnitude in low-rank adaptation. In *NeurIPS*, 2025b.

[70] J Zhao, T Wang, W Abid, G Angus, A Garg, J Kinnison, A Sherstinsky, P Molino, T Addair, and D Rishi. Lora land: 310 fine-tuned llms that rival gpt-4, a technical report. *arXiv preprint arXiv:2405.00732*, 2024.

[71] Q Zheng, X Xia, X Zou, Y Dong, S Wang, Y Xue, Z Wang, L Shen, A Wang, Y Li, et al. Codegeex: A pre-trained model for code generation with multilingual evaluations on humaneval-x. In *KDD*, 2023.

[72] T Zheng, G Zhang, T Shen, X Liu, B Y Lin, J Fu, W Chen, and X Yue. Opencodeinterpreter: Integrating code generation with execution and refinement. In *ACL Findings*, 2024.

[73] Y Zhong, H Jiang, L Li, R Nakada, T Liu, L Zhang, H Yao, and H Wang. Neat: Nonlinear parameter-efficient adaptation of pre-trained models. *arXiv preprint arXiv:2410.01870*, 2024.

[74] J Zhu, K Greenewald, K Nadjahi, H S d O Borde, R B Gabrielsson, L Choshen, M Ghassemi, M Yurochkin, and J Solomon. Asymmetry in low-rank adapters of foundation models. In *ICML*, 2024.

A Limitations

Our empirical recommendation should be interpreted within its intended scope. The choice $\alpha_{\text{base}} = 256\sqrt{r}$ provides a robust starting point and often improves upon representative PEFT baselines across our evaluated settings, but it is not a universal optimum; the preferred coefficient may depend on the model, task, and base learning rate. More broadly, increasing the scaling factor is important when fine-tuning requires substantial adaptation or acquisition of new knowledge. In lightweight alignment or capability-preserving settings, excessive scaling may instead lead to over-adaptation. We therefore recommend treating the proposed value as an good initialization for validation, rather than as a task-independent prescription.

Our theoretical results are subject to complementary qualifications. The Signal–Drift scaling analysis characterizes the local update geometry at a fixed adapter state, while the global convergence certificate relies on smoothness, bounded iterates, and a uniform descent condition. Moreover, the indefinite Drift term is not necessarily detrimental at every iteration: it may either assist or oppose descent. Our results should thus be understood as controlling its worst-case, sign-uncontrolled contribution under matched Signal amplification, rather than asserting that Drift is uniformly harmful throughout training.

B Broader Impact

This work studies the scaling mechanism of LoRA and proposes a principled scaling strategy that improves training effectiveness. By enabling more effective fine-tuning and reduced hyperparameter search, our method can lower the computational cost associated with adapting large models, potentially making such techniques more accessible. At the same time, improved training effectiveness may facilitate the broader deployment of large language models, including in settings where misuse is possible. Nevertheless, our work does not introduce new capabilities beyond improving optimization, and thus does not fundamentally alter the risk profile of existing models. Overall, we view this work as a step toward more efficient and principled model adaptation, while emphasizing that downstream applications should be developed and deployed with appropriate safeguards.

C Related Work

LoRA [Hu et al., 2022] leverages low-rank decomposition to efficiently adapt large models, and has become a key technique for scalable deployment in modern large-scale systems, significantly reducing training [Mangrulkar et al., 2023; Schulman & Lab, 2025; Wang et al., 2026] and serving costs [Kwon et al., 2023]. Comprehensive overviews of LoRA and its extensions can be found in [Yang et al., 2024; Mao et al., 2025; He et al., 2026]. To improve the expressiveness of low-rank updates, prior works explore architectural enhancements such as stacking strategies [Lialin et al., 2024; Xia et al., 2024], higher-order parameterizations [Edalati et al., 2023], and customized adapter designs [Jiang et al., 2024; Liu et al., 2024; Zhong et al., 2024]. More relevant to our work, research focuses on improving optimization and training efficiency. Existing approaches investigate scaling strategies [Kalajdzievski, 2023; Biderman et al., 2024], learning rate design [Hayou et al., 2024b], initialization schemes [Meng et al., 2024; Wang et al., 2024b; Zhang et al., 2025b], optimizer choices [Li et al., 2025], and batch size configurations [Lee & Lee, 2026].

Optimization Dynamics of LoRA. The optimization behavior of LoRA remains poorly understood due to its bilinear parameterization and non-convex landscape [Li et al., 2025; Xu et al., 2025]. Existing theoretical analyses are often restricted to simplified regimes, such as lazy training or infinite-width limits [Malladi et al., 2023; Hayou et al., 2024a]. Kalajdzievski [2023]; Liu et al. [2025] study stability conditions, showing that improper scaling can lead to gradient collapse or inefficient training. [Hayou et al., 2024b; Zhu et al., 2024] further highlight structural asymmetries between the two low-rank factors, motivating asymmetric optimization strategies such as distinct learning rates. [Mu & Klabjan, 2025] studies a non-asymptotic convergence analysis of the original LoRA gradient descent algorithm. While these efforts provide valuable insights, they do not fully explain how scaling choices affect optimization efficiency in practical settings.

The η–α Relationship. In practice, LoRA is typically trained with a fixed scaling factor (often $\alpha \propto r$) while tuning the learning rate η [Hu et al., 2022]. Empirically, LoRA requires significantly larger

learning rates, often exceeding $10\times$ those of full fine-tuning, to achieve competitive performance [Biderman et al., 2024; Schulman & Lab, 2025]. Recent theoretical results reveal a coupling between η and α [Zhang et al., 2025b; Schulman & Lab, 2025], showing that *increasing α can mimic the effect of simultaneously scaling the learning rate and initialization under adaptive optimizers.* We are inspired by these theoretical analyses, which we interpret as a hint that α and η are not interchangeable, with α exerting a stronger influence on optimization than the learning rate. More recently, Chen et al. [2026] propose the Maximal-Update Adaptation framework to study rank-invariant learning rates, while Lee et al. [2026] show that different LoRA variants favor distinct learning rate ranges despite achieving similar peak performance. Nevertheless, these studies are largely confined to the regime $\alpha = r$. Our empirical results show that this constraint limits LoRA's fitting capacity and leaves a significant optimization gap.

Discussion. *To the best of our knowledge, this is the first work to systematically characterize the role of the scaling factor α in LoRA optimization.* While existing analyses emphasize learning rate or initialization, we demonstrate that α plays a fundamentally different role. From a Hessian perspective, we explain why LoRA requires aggressive hyperparameters, and reveal a key asymmetry: α primarily amplifies task-aligned curvature, whereas η amplifies the magnitude of bilinear-induced drift. This decomposition suggests an optimal regime that increases α while keeping η conservative, enabling stronger signal amplification without destabilizing optimization.

D Details for Empirical Study

In this section, we provide comprehensive details of the experimental setup and extended optimization results across different model scales and datasets.

D.1 Experimental Setup and Search Strategy

Hyperparameter Search Strategy. To characterize the optimization landscape of LoRA, we tailored our hyperparameter search strategies based on the dataset and our evolving empirical understanding:

- **Tulu 3:** Given the lack of established heuristics for the optimal scaling regime at the onset of our study, we adopted a two-stage approach for the Tulu 3 dataset. We first defined a broad, coarse-grained grid across the learning rate η and the scaling factor α. Upon identifying the general region of convergence, we conducted a finer-grained localized search around the minimum validation loss to accurately pinpoint the optimal configurations η^* and α^*.
- **OpenThoughts:** For the OpenThoughts dataset, rather than repeating an exhaustive grid search, we leveraged the empirical law derived from the Tulu 3 experiments, where the base scaling coefficient was identified near $C = 256$. We anchored our search space around this empirical baseline, systematically sweeping α within a $10\times$ range (both scaling up and down) to validate the transferability of the sublinear scaling law.

OpenThoughts Dataset Details. To evaluate whether our findings generalize to reasoning-heavy tasks, which typically exhibit different optimization dynamics than general instruction tuning, we utilized the OpenThoughts dataset [Guha et al., 2026]. We sampled a 10k training subset and a 1k validation set. Because reasoning traces inherently require substantially more context, we extended the maximum sequence length to $T = 8192$ (compared to $T = 1024$ for Tulu 3). Note that FFT experiments were omitted for OpenThoughts, as the primary objective here is to validate the relative scaling dynamics between α and η established in the main text.

D.2 Scaling Behavior of Llama 3-8B on Tulu 3

Fig. 4 illustrates the hyperparameter analysis for the Llama 3-8B model on the Tulu 3 dataset. The observations strictly corroborate our findings from the 1B model. Notably, α-scaling continues to demonstrate superior optimization capacity over η-scaling. The optimal scaling factor α^* fits a sublinear relationship $\alpha = 512\sqrt{r}$, confirming that larger models also operate optimally at an α magnitude substantially larger than conventional rank-tied heuristics.

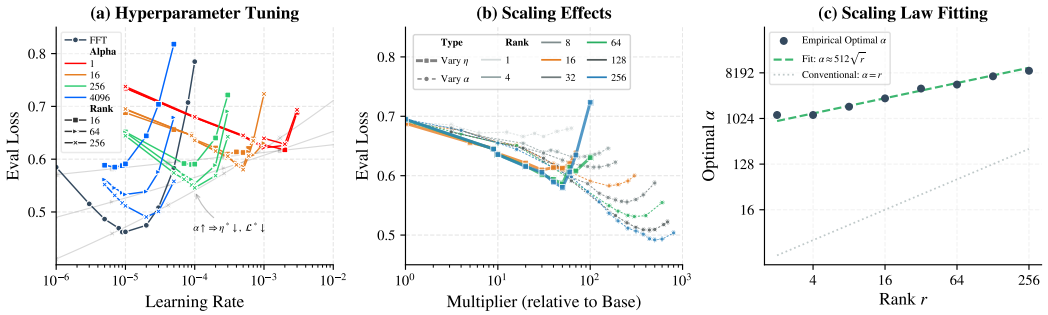

Figure 4: Hyperparameter analysis of Llama 3-8B on the Tulu 3 dataset. (a) Evaluation loss as a function of the learning rate η across different ranks r and scaling factors α. Gray lines denote linear fits of the minimum loss for each (r, α). Increasing α lowers the optimal loss and shifts η^* downward. (b) Scaling paths defined in Eq. (2) relative to the baseline ($\eta_{\text{FFT}} = 1 \times 10^{-5}$, $\alpha_0 = 16$), where η_0 denotes the optimal FFT learning rate. Varying α reaches lower-loss regimes that are inaccessible via η-tuning alone. (c) Optimal scaling factor α^* as a function of rank r, evaluated at η_0. The observed sublinear trend and large magnitude of α challenge conventional scaling heuristics.

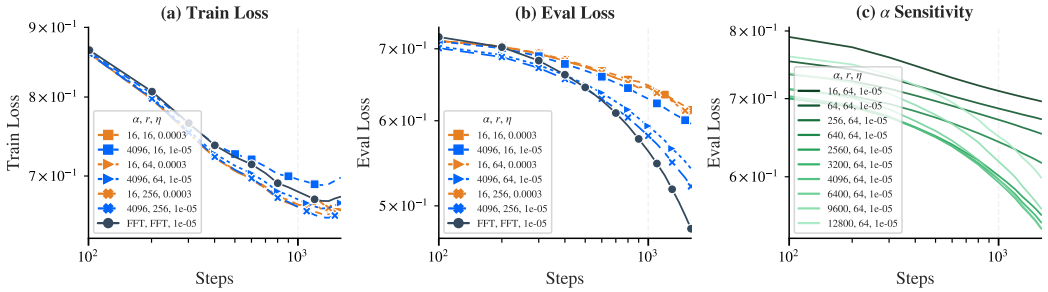

Figure 5: Optimization dynamics and α sensitivity for Llama 3-8B on the Tulu 3 dataset over training steps. (a) Training loss and (b) Evaluation loss curves for various combinations of rank r, scaling factor α, and learning rate η, compared alongside FFT. Properly scaled configurations (*e.g.*, larger α with standard η) tightly approximate the FFT trajectory. (c) Sensitivity analysis of the scaling factor α, illustrating how increasing α smoothly accelerates convergence.

Furthermore, Fig. 5 provides a granular view of the optimization dynamics over training steps. The loss trajectories demonstrate that properly scaled configurations, *i.e.*, operating with a large α and a standard small η, tightly approximate the descent trajectory of FFT.

D.3 Validation on OpenThoughts Reasoning Tasks

Fig. 6 and 7 present the optimization behaviors on the OpenThoughts dataset for Llama 3-1B and 8B, respectively. Despite the increased sequence length and task complexity, the fundamental asymmetry between α and η remains persistent. Both models consistently avoid early saturation and reach deeper loss minima when α is scaled. Interestingly, the fitted optimal scaling laws yield large coefficients $C = 1024$, indicating that complex reasoning tasks may demand an even more aggressive restoration of the task signal curvature.

D.4 Sensitivity of the Empirical Scaling Coefficient

Our empirical variant uses $\alpha = C\sqrt{r}$ with $C = 256$. We emphasize that $C = 256$ is not intended as a rigid universal optimum, but as a conservative and locally robust default. As shown in Table 6, the coefficient fitted from the individual sweeps varies with the model, dataset, and base learning rate.

The fitted coefficients span $C_{\text{fit}} \in [256, 1024]$. For each matched model–dataset pair, increasing the base learning rate from 1×10^{-5} to 2×10^{-5} reduces the fitted coefficient. To quantify robustness beyond these point estimates, we sweep $C \in \{64, 128, 256, 512, 1024\}$ over

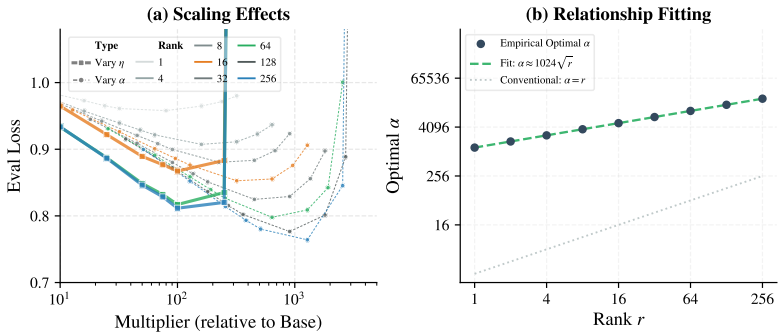

Figure 6: Hyperparameter analysis of Llama 3-1B on the OpenThoughts dataset with $\eta_{\text{FFT}} = 1 \times 10^{-5}$ and $\alpha_0 = 16$. (a) Scaling paths comparing α-scaling and η-scaling. Consistent with the main findings, increasing α enables the model to reach deeper loss minima that are inaccessible via η-tuning alone. (b) Optimal scaling factor α^* as a function of rank r. The fitted sublinear relationship confirms the scaling law established in the main text, demonstrating its robustness across different datasets.

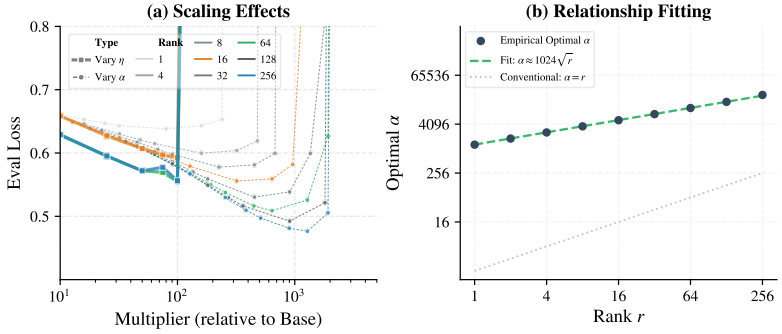

Figure 7: Hyperparameter analysis of Llama 3-8B on the OpenThoughts dataset with $\eta_{\text{FFT}} = 1 \times 10^{-5}$ and $\alpha_0 = 16$. (a) Scaling paths comparing α-scaling and η-scaling. Consistent with the main findings, increasing α enables the model to reach deeper loss minima that are inaccessible via η-tuning alone. (b) Optimal scaling factor α^* as a function of rank r. The fitted sublinear relationship confirms the scaling law established in the main text, demonstrating its robustness across different datasets.

$r \in \{1, 4, 16, 32, 64, 128, 256\}$ on both Tulu 3 and OpenThoughts. We report two complementary quantities: the relative regret $D(C)$, which measures the percentage loss gap from the rank-wise optimal coefficient C_r^*, and the local flatness $F(C)$, which measures the local percentage loss slope per doubling of C. The results are provided in Table 7, where lower values are better for both metrics.

Although $C = 512$ attains marginally lower mean regret than $C = 256$ (1.53% versus 1.65%), $C = 256$ has a substantially flatter local neighborhood (2.40% versus 4.25%). Increasing the coefficient to $C = 1024$ enters a sharply sensitive regime, with mean local flatness rising to 42.15%. These results motivate $C = 256$ as a low-risk default that is near-optimal across model scales, datasets, and ranks without requiring a new coefficient search for every setting.

E Empirical Validation of Signal-Drift Dynamics

To empirically validate the Signal-Drift dynamics and visualize the intractable Hessian matrices without the memory bottleneck, we construct a controlled environment using a multi-layer perceptron (MLP). This setup allows us to precisely track the optimization trajectories and perform exact spectral analysis on the optimization landscape.

Setups. We employ a 5-layer MLP utilizing the SiLU activation function, trained to fit synthetic Gaussian data. We generate a target projection matrix W_{true} constructed via Singular Value Decomposition (SVD), where the singular values follow a strict decay rate of $\sigma_i = 1/i$. The training set

Table 6: Fitted coefficients in $\alpha^*(r) \approx C_{\text{fit}}\sqrt{r}$ across models, datasets, and learning rates. The coefficients are obtained by log-scale fitting and rounded to the reported coefficient grid.

Model	Dataset	Learning rate η	C_{fit}
Llama 3-1B	Tulu 3	2×10^{-5}	256
Llama 3-1B	OpenThoughts	2×10^{-5}	512
Llama 3-8B	Tulu 3	2×10^{-5}	256
Llama 3-8B	OpenThoughts	2×10^{-5}	512
Llama 3-1B	Tulu 3	1×10^{-5}	512
Llama 3-1B	OpenThoughts	1×10^{-5}	1024
Llama 3-8B	Tulu 3	1×10^{-5}	512
Llama 3-8B	OpenThoughts	1×10^{-5}	1024

Table 7: Sensitivity of the empirical scaling coefficient in $\alpha = C\sqrt{r}$. Each cell reports relative regret $D(C)$ / local flatness $F(C)$ in percent, aggregated over the evaluated ranks. Lower is better. All settings use the same base learning rate $\eta = 2 \times 10^{-5}$.

Model	Dataset	Learning rate η	$C = 64$	$C = 128$	$C = 256$	$C = 512$	$C = 1024$
Llama 3-1B	Tulu 3	2×10^{-5}	3.98 / 2.41	0.74 / 1.72	0.10 / 0.70	2.97 / 5.98	8.87 / 10.54
Llama 3-1B	OpenThoughts	2×10^{-5}	9.04 / 3.33	5.71 / 3.27	2.50 / 2.75	0.33 / 1.37	1.22 / 21.68
Llama 3-8B	Tulu 3	2×10^{-5}	9.52 / 5.28	3.45 / 4.48	0.16 / 1.95	2.31 / 7.18	13.04 / 18.27
Llama 3-8B	OpenThoughts	2×10^{-5}	13.31 / 4.50	8.81 / 4.73	3.84 / 4.21	0.49 / 2.46	1.30 / 118.11
Mean across settings		–	**8.96 / 3.88**	**4.68 / 3.55**	**1.65 / 2.40**	**1.53 / 4.25**	**6.10 / 42.15**

consists of $n = 100$ samples drawn from a standard normal distribution, with targets generated as $Y = XW_{\text{true}}^\top$. LoRA adapters are applied to all linear layers with a fixed rank of $r = 16$. Pre-trained weights W_0 and adapter matrices A are initialized from $\mathcal{N}(0, 1/d)$, while B is initialized to zero. For general optimization tracking, we use a hidden dimension of $d = 256$. For exact Hessian spectral analysis, which scales quadratically with parameter count, we reduce the dimension to $d = 32$.

Optimization Dynamics (η-scaling vs. α-scaling). The model is optimized using Adam for 100 steps with a MSE loss. The base learning rate and base scaling factor are set to $\eta_0 = 10^{-4}$ and $\alpha_0 = 1$, respectively. To compare their behavioral differences, we conduct experiments across two distinct regimes:

- **η-scaling:** We fix $\alpha = \alpha_0$ and strictly scale the learning rate by factors of $\{16, 32, 64, 128\} \times \eta_0$.
- **α-scaling:** We anchor the learning rate at $\eta = \eta_0$ and scale α by factors of $\{32, 64, 128, 256\} \times \alpha_0$.

These two regimes yield fundamentally different optimization dynamics. While configurations $\eta = 64\eta_0$ and $\alpha = 128\alpha_0$ initially exhibit comparable convergence trends, the η-scaled trajectory already manifests noticeable oscillations. Pushing the scaling further exposes the structural bottleneck: $\eta = 128\eta_0$ causes the training to catastrophically diverge, whereas $\alpha = 256\alpha_0$ remains strictly stable and continues to unlock deeper fitting (as evidenced in Fig. 2).

Validation under SGD. To verify that the asymmetric scaling behavior is not specific to adaptive optimization, we repeat the controlled experiment using vanilla SGD for 500 steps. Starting from (α_0, η_0), we compare learning-rate scaling $\eta/\eta_0 \in \{1024, 4096, 8192\}$ at fixed α_0 with scaling-factor scaling $\alpha/\alpha_0 \in \{128, 256, 512, 768\}$ at fixed η_0. As shown in Fig. 8, increasing α achieves substantially lower training loss while maintaining smaller Drift excursions and markedly higher Signal-to-Drift ratios. In contrast, large learning rates amplify the sign-varying Drift and stall at higher loss. This confirms that the Signal–Drift asymmetry persists under SGD and is not an artifact of gradient normalization.

Spectral Compression under Low-Rank Parameterization. To understand why LoRA scales differently from FFT, we explicitly compute the full-parameter Hessian $\mathbf{H}_\ell$, the LoRA Hessian $\mathcal{H}_{\text{LoRA}}$, and its components $\mathcal{H}_{\text{Signal}}$ and $\mathcal{H}_{\text{Drift}}$ via exact automatic differentiation. As shown in Fig. 9(a), the spectrum of $\mathcal{H}_{\text{Signal}}$ is a compressed version of $\mathbf{H}_\ell$. The dominant eigenvalues are

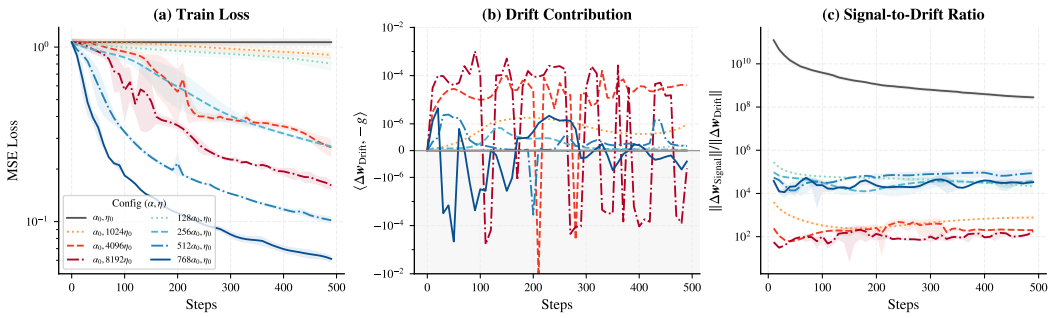

Figure 8: Validation of Signal–Drift asymmetry under vanilla SGD. (a) Training loss across learning-rate and scaling-factor sweeps. (b) Signed alignment of the Drift update with the negative task gradient. (c) Ratio between the Signal and Drift update magnitudes. Scaling α produces deeper fitting with more controlled Drift than directly increasing η.

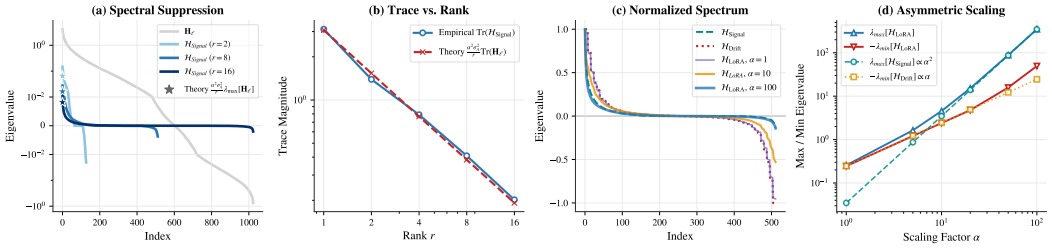

Figure 9: Spectral analysis of Hessian matrices. (a) Comparison between the full-parameter Hessian $\mathbf{H}_\ell$ and LoRA's signal Hessian $\mathcal{H}_{\text{Signal}}$, illustrating the compression of total curvature energy. (b) The predicted sum of eigenvalues from our Signal-Drift framework closely matches the empirical values across various ranks, validating our theoretical derivation. (c) The exact LoRA Hessian $\mathcal{H}_{\text{LoRA}}$ is inherently indefinite due to the structural drift $\mathcal{H}_{\text{Drift}}$. Increasing α triggers spectral purification, where the spectrum converges toward the positive semi-definite $\mathcal{H}_{\text{Signal}}$. (d) Empirical verification of asymmetric scaling: the signal curvature magnitude grows as $\Theta(\alpha^2)$ while drift noise grows only as $\Theta(\alpha)$. Consequently, as α increases, $\lambda_{\max}[\mathcal{H}_{\text{LoRA}}]$ perfectly aligns with $\lambda_{\max}[\mathcal{H}_{\text{Signal}}]$.

significantly reduced, and the trace explicitly follows $\mathrm{Tr}(\mathcal{H}_{\text{Signal}}) \approx \frac{\alpha^2 \sigma_A^2}{r} \mathrm{Tr}(\mathbf{H}_\ell)$. Notably, Fig. 9(a) illustrates that the maximum eigenvalue $\lambda_{\max}$ empirically aligns with the predicted order of magnitude, although it is less strictly constrained than the trace. Since the trace equals the sum of all eigenvalues ($\mathrm{Tr}(\mathbf{H}) = \sum_i \lambda_i$), this trace-level compression inherently necessitates a suppression of the dominant spectral components. This confirms that low-rank parameterization severely restricts curvature energy, providing a quantitative explanation for why a larger α is fundamentally required to compensate for rank-induced suppression.

Indefiniteness and Spectral Purification. Unlike the full Hessian, $\mathcal{H}_{\text{LoRA}}$ is inherently indefinite due to the presence of the structural drift $\mathcal{H}_{\text{Drift}}$. As shown in Fig. 9(c), this drift component introduces negative eigenvalues, which destabilize optimization. However, increasing α progressively amplifies the signal term relative to the drift, leading to a *spectral purification* effect: the overall indefinite spectrum becomes increasingly aligned with the positive semi-definite $\mathcal{H}_{\text{Signal}}$.

Asymmetric Scaling of Signal and Drift. Fig. 9(d) empirically verifies the asymmetric scaling behavior. We observe a strict divergence in growth rates:

$$\lambda_{\max}[\mathcal{H}_{\text{Signal}}] = \Theta(\alpha^2), \tag{6}$$

$$|\lambda_{\min}[\mathcal{H}_{\text{Drift}}]| = \Theta(\alpha). \tag{7}$$

This mismatch guarantees that increasing α disproportionately strengthens the task-aligned curvature while only linearly amplifying the destabilizing drift. As a result, the dominant eigenvalue of $\mathcal{H}_{\text{LoRA}}$ increasingly aligns with that of $\mathcal{H}_{\text{Signal}}$.

Together, these controlled experiments provide a validated spectral explanation for the empirical superiority of α-scaling: (i) low-rank parameterization suppresses curvature magnitude; (ii) α restores signal strength in a controlled manner; and (iii) the asymmetric scaling between signal and drift naturally dictates that increasing α, rather than the learning rate, is the principled path to stable and accelerated optimization.

F Comparison of Scaling Factor Magnitudes

Table 8: Comparison of the standard heuristic ($\alpha = r$) with our proposed empirical value ($256\sqrt{r}$) and analytic value ($\frac{1}{\sigma_A}\sqrt{r} = \sqrt{3d_{in}r}$ with $d_{in} = 4096$).

Rank (r)	**Heuristic** ($\alpha = r$)	**Empirical** ($256\sqrt{r}$)	**Theoretical** ($\frac{1}{\sigma_A}\sqrt{r}$)	**Magnitude Increase**
1	1	256	111	256.0 ×
4	4	512	222	128.0 ×
8	8	724	314	90.5 ×
16	16	1024	443	64.0 ×
32	32	1448	627	45.3 ×
64	64	2048	887	32.0 ×
128	128	2896	1254	22.6 ×
256	256	4096	1774	16.0 ×

Quantitative Comparison of Scaling Magnitudes. To further illustrate the optimization gap, we provide a quantitative comparison between our proposed scaling law and prevailing heuristics in Table 8. Conventional practices typically set $\alpha = r$, which implicitly confines the scaling factor to a severely under-scaled regime as the rank increases. According to our Signal-Drift framework, restoring the curvature energy (Hessian trace) to a level comparable to FFT requires α to scale with $\frac{1}{\sigma_A}\sqrt{r}$. For a typical model dimension $d_{in} = 4096$ and $\sigma_A^2 = 1/(3d_{in})$, the theoretical coefficient $\frac{1}{\sigma_A}$ is approximately 111. Our empirical default $C = 256$ is an even more optimized choice that ensures deep fitting. As shown in the table, for a standard rank $r = 8$, LoRA-α provides a scaling magnitude that is 90.5× larger than the common heuristic. Even at $r = 256$, the gap remains as high as 16×. This comparison clarifies why standard FFT learning rates have historically been inadequate for driving LoRA to its full potential under legacy scaling rules.

Note on Initialization Variance. The parameter $\sigma_A^2 = 1/(3d_{in})$ is chosen to align with the default initialization protocol in standard deep learning libraries (*e.g.*, PyTorch and PEFT). Specifically, LoRA adapters are typically initialized using a Kaiming uniform distribution $\mathcal{U}(-\sqrt{1/d_{in}}, \sqrt{1/d_{in}})$. For a uniform distribution $\mathcal{U}(-k, k)$, the variance is given by $k^2/3$. By substituting $k^2 = 1/d_{in}$, we obtain $\sigma_A^2 = 1/(3d_{in})$. This ensures that the magnitudes predicted by our theoretical framework are consistent in order of magnitude with those used in practical implementations.

G Theoretical Proofs

Notations. Consider a pre-trained weight $W_0 \in \mathbb{R}^{d_{out}\times d_{in}}$ and trainable adapters $\theta = \{A, B\}$, where $B \in \mathbb{R}^{d_{out}\times r}$ and $A \in \mathbb{R}^{r\times d_{in}}$. The LoRA mapping is defined as $W(\boldsymbol{\theta}) = W_0 + \frac{\alpha}{r}BA$. Let $\boldsymbol{w}(\boldsymbol{\theta}) = \mathrm{vec}(W(\boldsymbol{\theta})) \in \mathbb{R}^D$ denote the vectorized weight, and $\boldsymbol{\theta} = [\mathrm{vec}(A); \mathrm{vec}(B)] \in \mathbb{R}^p$ the concatenated parameters, where $D = d_{out}d_{in}$ and $p = r(d_{in} + d_{out})$. For the objective $\ell(\boldsymbol{w})$, we define the task gradient $\boldsymbol{g} = \nabla_{\boldsymbol{w}}\ell \in \mathbb{R}^D$ and the task Hessian $\mathbf{H}_\ell = \nabla^2_{\boldsymbol{w}}\ell \in \mathbb{R}^{D\times D}$, with g_k and w_k representing their k-th scalar entries. The mapping geometry is governed by the Jacobian $J(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}}\boldsymbol{w} \in \mathbb{R}^{D\times p}$ and the structural Hessian $\nabla^2_{\boldsymbol{\theta}}\boldsymbol{w} \in \mathbb{R}^{D\times p\times p}$, interpreted as a bilinear operator: for any $\boldsymbol{v} \in \mathbb{R}^p$, $\nabla^2_{\boldsymbol{\theta}}\boldsymbol{w}[\boldsymbol{v}, \boldsymbol{v}] \in \mathbb{R}^D$ has entries $\boldsymbol{v}^\top(\nabla^2_{\boldsymbol{\theta}}w_k)\boldsymbol{v}$.

G.1 Proof for Proposition 1

Proposition (Signal-Drift Decomposition)**.** *Let $\Delta\boldsymbol{w}_{LoRA} = \boldsymbol{w}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) - \boldsymbol{w}(\boldsymbol{\theta})$ denote the effective step in the weight space, and $\mathcal{H}_{LoRA} = \nabla^2_{\boldsymbol{\theta}}\ell$ denote the parameter-space Hessian. Under the LoRA*

parameterization, both admit an exact decomposition into a task-aligned signal and a structural drift:

$$\Delta \boldsymbol{w}_{LoRA} = \underbrace{J(\boldsymbol{\theta})\Delta\boldsymbol{\theta}}_{\Delta \boldsymbol{w}_{Signal}} + \underbrace{\tfrac{1}{2}\nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}[\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\theta}]}_{\Delta \boldsymbol{w}_{Drift}}, \; \mathcal{H}_{LoRA} = \underbrace{J(\boldsymbol{\theta})^\top \mathbf{H}_\ell J(\boldsymbol{\theta})}_{\mathcal{H}_{Signal}} + \underbrace{\Sigma_{k=1}^D g_k \nabla^2_{\boldsymbol{\theta}} w_k}_{\mathcal{H}_{Drift}}. \tag{8}$$

Proof. **Part I: Decomposition of the Weight Update $\Delta \boldsymbol{w}$.**
Recall that the general multivariate Taylor series expansion of the mapping $\boldsymbol{w}$ around $\boldsymbol{\theta}$ for a parameter update $\Delta\boldsymbol{\theta}$ is given by:

$$\boldsymbol{w}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) = \boldsymbol{w}(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}} \boldsymbol{w}(\boldsymbol{\theta})\Delta\boldsymbol{\theta} + \frac{1}{2}\nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}[\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\theta}] + \mathcal{O}(\|\Delta\boldsymbol{\theta}\|^3). \tag{9}$$

Under the LoRA framework, the mapping from the parameters $\boldsymbol{\theta} \coloneqq [\mathrm{vec}(A); \mathrm{vec}(B)]$ to the vectorized weights $\boldsymbol{w}$ is strictly bilinear. Because each element of $\boldsymbol{w}(\boldsymbol{\theta})$ contains products of at most two parameter elements, all third-order and higher-order derivatives of $\boldsymbol{w}$ with respect to $\boldsymbol{\theta}$ vanish identically. Consequently, the higher-order residual term $\mathcal{O}(\|\Delta\boldsymbol{\theta}\|^3)$ is exactly zero, and the expansion terminates exactly at the second-order term. Thus, we have the exact equality:

$$\boldsymbol{w}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) = \boldsymbol{w}(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}} \boldsymbol{w}(\boldsymbol{\theta})\Delta\boldsymbol{\theta} + \frac{1}{2}\nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}[\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\theta}]. \tag{10}$$

By definition, the weight displacement is $\Delta \boldsymbol{w}_{\text{LoRA}} \coloneqq \boldsymbol{w}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) - \boldsymbol{w}(\boldsymbol{\theta})$ and the Jacobian is $J(\boldsymbol{\theta}) \coloneqq \nabla_{\boldsymbol{\theta}} \boldsymbol{w}(\boldsymbol{\theta})$. Substituting these definitions directly yields the first decomposition:

$$\Delta \boldsymbol{w}_{\text{LoRA}} = \underbrace{J(\boldsymbol{\theta})\Delta\boldsymbol{\theta}}_{\Delta \boldsymbol{w}_{\text{Signal}}} + \underbrace{\frac{1}{2}\nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}[\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\theta}]}_{\Delta \boldsymbol{w}_{\text{Drift}}}. \tag{11}$$

Part II: Decomposition of the Hessian $\mathcal{H}_{\text{LoRA}}$.
First, we apply the multivariate chain rule to find the gradient of ℓ with respect to the i-th parameter θ_i:

$$\frac{\partial \ell}{\partial \theta_i} = \sum_{k=1}^{D} \frac{\partial \ell}{\partial w_k} \frac{\partial w_k}{\partial \theta_i} = \sum_{k=1}^{D} g_k \frac{\partial w_k}{\partial \theta_i}, \tag{12}$$

where $g_k = \frac{\partial \ell}{\partial w_k}$ is the k-th element of the task gradient $\boldsymbol{g}$.

Next, we differentiate this expression with respect to another parameter θ_j to obtain the (i, j)-th entry of the parameter Hessian:

$$[\mathcal{H}_{\text{LoRA}}]_{i,j} = \frac{\partial^2 \ell}{\partial \theta_i \partial \theta_j} = \frac{\partial}{\partial \theta_j} \left(\sum_{k=1}^{D} g_k \frac{\partial w_k}{\partial \theta_i} \right). \tag{13}$$

Applying the product rule, we obtain two separate terms:

$$[\mathcal{H}_{\text{LoRA}}]_{i,j} = \sum_{k=1}^{D} \left(\frac{\partial g_k}{\partial \theta_j} \frac{\partial w_k}{\partial \theta_i} + g_k \frac{\partial^2 w_k}{\partial \theta_i \partial \theta_j} \right). \tag{14}$$

We now expand the term $\frac{\partial g_k}{\partial \theta_j}$ using the chain rule once more:

$$\frac{\partial g_k}{\partial \theta_j} = \sum_{m=1}^{D} \frac{\partial g_k}{\partial w_m} \frac{\partial w_m}{\partial \theta_j} = \sum_{m=1}^{D} [\mathbf{H}_\ell]_{k,m} J_{m,j}, \tag{15}$$

where $[\mathbf{H}_\ell]_{k,m} = \frac{\partial^2 \ell}{\partial w_k \partial w_m}$ is the (k, m)-th entry of the task Hessian $\mathbf{H}_\ell$, and $J_{m,j} = \frac{\partial w_m}{\partial \theta_j}$ is the (m, j)-th entry of the Jacobian $J(\boldsymbol{\theta})$. Substituting this expansion back into the expression for $[\mathcal{H}_{\text{LoRA}}]_{i,j}$:

$$[\mathcal{H}_{\text{LoRA}}]_{i,j} = \sum_{k=1}^{D} \sum_{m=1}^{D} \underbrace{\frac{\partial w_k}{\partial \theta_i}}_{J_{k,i}} [\mathbf{H}_\ell]_{k,m} \underbrace{\frac{\partial w_m}{\partial \theta_j}}_{J_{m,j}} + \sum_{k=1}^{D} g_k \frac{\partial^2 w_k}{\partial \theta_i \partial \theta_j}. \tag{16}$$

Observe that the double summation on the left corresponds exactly to the matrix multiplication $(J(\boldsymbol{\theta})^\top \mathbf{H}_\ell J(\boldsymbol{\theta}))_{i,j}$. The summation on the right corresponds exactly to the (i,j)-th entry of the matrix $\sum_{k=1}^{D} g_k \nabla^2_{\boldsymbol{\theta}} w_k$.

Converting this element-wise equality back into matrix notation, we arrive at the final decomposition:

$$\mathcal{H}_{\text{LoRA}} = \underbrace{J(\boldsymbol{\theta})^\top \mathbf{H}_\ell J(\boldsymbol{\theta})}_{\mathcal{H}_{\text{Signal}}} + \underbrace{\sum_{k=1}^{D} g_k \nabla^2_{\boldsymbol{\theta}} w_k}_{\mathcal{H}_{\text{Drift}}}. \tag{17}$$

This establishes both partitions and completes the proof. □

By the definition of the LoRA forward pass, the weight matrix is $W(\theta) = W + \frac{\alpha}{r} BA$. For a parameter perturbation $\Delta\theta = \{\Delta A, \Delta B\}$, the perturbed weight matrix is:

$$W(\theta + \Delta\theta) = W + \frac{\alpha}{r}(B + \Delta B)(A + \Delta A) \tag{18}$$

$$= W + \frac{\alpha}{r} BA + \frac{\alpha}{r}(B\Delta A + \Delta B A) + \frac{\alpha}{r}\Delta B \Delta A. \tag{19}$$

The first-order variation with respect to the parameters corresponds strictly to the linear terms in ΔA and ΔB. Vectorizing this linear component directly yields the exact algebraic form of the task signal:

$$\Delta \boldsymbol{w}_{\text{Signal}} = J(\boldsymbol{\theta})\Delta\boldsymbol{\theta} = \text{vec}\left(\frac{\alpha}{r}(B\Delta A + \Delta B A)\right). \tag{20}$$

Vectorizing this bilinear cross-term gives the structural drift:

$$\Delta \boldsymbol{w}_{\text{Drift}} = \frac{1}{2}\nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}[\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\theta}] = \text{vec}\left(\frac{\alpha}{r}\Delta B \Delta A\right). \tag{21}$$

G.2 Proof for Proposition 2

Proposition (Geometric Properties of Signal and Drift)**.** *For any gradient-based update step $\Delta\boldsymbol{\theta} = -\eta\nabla_{\boldsymbol{\theta}}\ell$, the decomposed components in Proposition 1 satisfy:*

(i) Constructive Signal: *The signal component aligns with the descent direction, $\langle \Delta\boldsymbol{w}_{Signal}, -\boldsymbol{g}\rangle \geq 0$, and preserves the local convexity of the loss landscape, i.e., $\mathcal{H}_{Signal} \succeq 0$ given $\mathbf{H}_\ell \succeq 0$.*

(ii) Adversarial Drift: *The structural Hessian $\mathcal{H}_{Drift}$ is indefinite whenever the task gradient with respect to the adapted weight is nonzero, and its induced update $\Delta\boldsymbol{w}_{Drift}$ exerts an uncontrolled force with no guaranteed alignment with $-\boldsymbol{g}$.*

Proof. **Part I: Properties of the Signal Terms.**

1. Gradient alignment ($\langle \Delta\boldsymbol{w}_{Signal}, -\boldsymbol{g}\rangle \geq 0$):
We evaluate the inner product between the signal component and the negative task gradient $-\boldsymbol{g}$ (the ideal steepest descent direction in the full-weight space). By the definition of the signal term and the chain rule ($\nabla_{\boldsymbol{\theta}}\ell = J(\boldsymbol{\theta})^\top \boldsymbol{g}$), this inner product elegantly reduces to the parameter space:

$$\langle \Delta\boldsymbol{w}_{\text{Signal}}, -\boldsymbol{g}\rangle = \langle J(\boldsymbol{\theta})\Delta\boldsymbol{\theta}, -\boldsymbol{g}\rangle = \langle \Delta\boldsymbol{\theta}, -J(\boldsymbol{\theta})^\top \boldsymbol{g}\rangle = \langle \Delta\boldsymbol{\theta}, -\nabla_{\boldsymbol{\theta}}\ell\rangle. \tag{22}$$

This establishes that the alignment of the projected signal in the full-weight space is mathematically equivalent to the alignment of the parameter update with the parameter gradient. We analyze this property under three distinct optimization regimes with learning rate $\eta > 0$:

- **Case 1: Standard Gradient Descent (GD).**
  Under GD, $\Delta\boldsymbol{\theta} = -\eta\nabla_{\boldsymbol{\theta}}\ell$. The alignment yields the squared ℓ_2-norm:
  $$\langle \Delta\boldsymbol{\theta}, -\nabla_{\boldsymbol{\theta}}\ell\rangle = \langle -\eta\nabla_{\boldsymbol{\theta}}\ell, -\nabla_{\boldsymbol{\theta}}\ell\rangle = \eta\|\nabla_{\boldsymbol{\theta}}\ell\|_2^2 \geq 0. \tag{23}$$
- **Case 2: Adam (Sign Gradient).**
  To isolate the geometric effect of Adam's adaptive denominator, we abstract its update rule as scaled sign gradient descent, $\Delta\boldsymbol{\theta} = -\eta\,\text{sign}(\nabla_{\boldsymbol{\theta}}\ell)$, applied element-wise. Since $x \cdot \text{sign}(x) = |x|$, the alignment resolves to the ℓ_1-norm:
  $$\langle \Delta\boldsymbol{\theta}, -\nabla_{\boldsymbol{\theta}}\ell\rangle = \eta\sum_i \text{sign}([\nabla_{\boldsymbol{\theta}}\ell]_i)[\nabla_{\boldsymbol{\theta}}\ell]_i = \eta\|\nabla_{\boldsymbol{\theta}}\ell\|_1 \geq 0. \tag{24}$$

- **Case 3: Muon (Orthogonalized Gradient).**
  Muon operates on the matrix parameters directly. Let $X \in \{A, B\}$ denote a parameter block with gradient $G_X = \nabla_X \ell$. Using the compact SVD, $G_X = U_X \Sigma_X V_X^\top$. Muon drops the singular values to yield the orthogonalized update $\Delta X = -\eta U_X V_X^\top$. The total parameter alignment is the sum of alignments over the blocks. Using the trace inner product $\langle A, B \rangle = \mathrm{Tr}(A^\top B)$ and its cyclic permutation property:

$$\langle \Delta X, -G_X \rangle = \mathrm{Tr}\left((-\eta U_X V_X^\top)^\top (-U_X \Sigma_X V_X^\top)\right) \tag{25}$$

$$= \eta \,\mathrm{Tr}\left(V_X U_X^\top U_X \Sigma_X V_X^\top\right) \tag{26}$$

$$= \eta \,\mathrm{Tr}\left(\Sigma_X V_X^\top V_X\right) = \eta \,\mathrm{Tr}(\Sigma_X). \tag{27}$$

  Since $\mathrm{Tr}(\Sigma_X)$ is the sum of singular values, it exactly equals the nuclear norm $\|G_X\|_*$. Summing over the adapter blocks yields:

$$\langle \Delta\boldsymbol{\theta}, -\nabla_{\boldsymbol{\theta}} \ell \rangle = \eta(\|G_A\|_* + \|G_B\|_*) \geq 0. \tag{28}$$

In all cases, the update strictly aligns with the negative gradient, guaranteeing that the signal term mimics a valid FFT descent direction measured in different geometric norms.

2. Positive semi-definiteness of $\mathcal{H}_{Signal}$:
Recall from the decomposition that $\mathcal{H}_{\text{Signal}} = J(\boldsymbol{\theta})^\top \mathbf{H}_\ell J(\boldsymbol{\theta})$. Assume the task landscape is locally convex, meaning the task Hessian is positive semi-definite ($\mathbf{H}_\ell \succeq 0$). For any arbitrary non-zero vector $\boldsymbol{v} \in \mathbb{R}^p$, we have:

$$\boldsymbol{v}^\top \mathcal{H}_{\text{Signal}} \boldsymbol{v} = \boldsymbol{v}^\top (J(\boldsymbol{\theta})^\top \mathbf{H}_\ell J(\boldsymbol{\theta})) \boldsymbol{v} = (J(\boldsymbol{\theta})\boldsymbol{v})^\top \mathbf{H}_\ell (J(\boldsymbol{\theta})\boldsymbol{v}). \tag{29}$$

Let $\boldsymbol{u} \coloneqq J(\boldsymbol{\theta})\boldsymbol{v} \in \mathbb{R}^D$. Since $\mathbf{H}_\ell \succeq 0$, it follows that $\boldsymbol{u}^\top \mathbf{H}_\ell \boldsymbol{u} \geq 0$ for all $\boldsymbol{u}$. Therefore, $\boldsymbol{v}^\top \mathcal{H}_{\text{Signal}} \boldsymbol{v} \geq 0$, which proves $\mathcal{H}_{\text{Signal}} \succeq 0$.

Part II: Properties of the Drift Terms.

1. Indefinite saddle structure of $\mathcal{H}_{Drift}$:
To prove that the structural drift introduces a saddle point, we analyze the block structure of $\mathcal{H}_{\text{Drift}}$. Since the mapping $W(\theta) = W_0 + \frac{\alpha}{r} BA$ is strictly bilinear, all intra-variable second derivatives (*e.g.*, with respect to A or B alone) vanish identically. The parameter Hessian of any scalar entry w_k thus consists solely of cross-derivatives, yielding a strictly block off-diagonal form. The aggregated drift Hessian naturally inherits this structure:

$$\nabla^2_{\boldsymbol{\theta}} w_k = \frac{\alpha}{r} \begin{bmatrix} \mathbf{0} & \mathbf{C}_k \\ \mathbf{C}_k^\top & \mathbf{0} \end{bmatrix} \quad \Longrightarrow \quad \mathcal{H}_{\text{Drift}} = \sum_{k=1}^{D} g_k \nabla^2_{\boldsymbol{\theta}} w_k = \frac{\alpha}{r} \begin{bmatrix} \mathbf{0} & \mathbf{C} \\ \mathbf{C}^\top & \mathbf{0} \end{bmatrix}, \tag{30}$$

where $\mathbf{C} = \sum_{k=1}^{D} g_k \mathbf{C}_k$ aggregates the gradient-weighted cross-derivatives.

By the spectral properties of symmetric block off-diagonal matrices, their non-zero eigenvalues strictly appear in symmetric pairs ($\pm\lambda$). Whenever the task gradient with respect to the adapted weight is nonzero, $\mathbf{C} \neq \mathbf{0}$, so $\mathcal{H}_{\text{Drift}}$ is indefinite. Thus, away from stationary points with respect to the adapted weight, the bilinear parameterization introduces a saddle-point component into the optimization landscape.

2. Uncontrolled force of $\Delta \boldsymbol{w}_{Drift}$:
To demonstrate that the structural drift can actively degrade optimization, we examine its alignment with the negative task gradient ($-\boldsymbol{g}$) under standard gradient descent. Recall from Proposition 1 that the weight-space drift is defined via the structural Hessian tensor as $\Delta \boldsymbol{w}_{\text{Drift}} = \frac{1}{2} \nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}[\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\theta}]$. Under GD with learning rate $\eta > 0$, the parameter update is $\Delta\boldsymbol{\theta} = -\eta \nabla_{\boldsymbol{\theta}} \ell = -\eta J(\theta)^\top \boldsymbol{g}$. Evaluating the inner product between the drift step and the ideal descent direction yields:

$$\langle \Delta \boldsymbol{w}_{\text{Drift}}, -\boldsymbol{g} \rangle = -\frac{1}{2} \boldsymbol{g}^\top \left(\nabla^2_{\boldsymbol{\theta}} \boldsymbol{w}[\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\theta}] \right) = -\frac{1}{2} \Delta\boldsymbol{\theta}^\top \left(\sum_{k=1}^{D} g_k \nabla^2_{\boldsymbol{\theta}} w_k \right) \Delta\boldsymbol{\theta}. \tag{31}$$

Remarkably, the term inside the parenthesis is exactly the parameter-space drift Hessian defined in Proposition 1. Thus, the inner product simplifies to a quadratic form:

$$\langle \Delta \boldsymbol{w}_{\text{Drift}}, -\boldsymbol{g} \rangle = -\frac{1}{2} \Delta\boldsymbol{\theta}^\top \mathcal{H}_{\text{Drift}} \Delta\boldsymbol{\theta}. \tag{32}$$

Because $\mathcal{H}_{\text{Drift}}$ is a block off-diagonal indefinite matrix, this quadratic form is not guaranteed to be non-negative, meaning the drift can easily oppose the descent direction.

Counterexample: Consider a scalar network ($D = 1$, rank $r = 1$) with $\alpha = 1$. The effective weight is $w = ab$, parameterized by $\boldsymbol{\theta} = [a, b]^\top$. Assume the current parameters are $\boldsymbol{\theta} = [1, 1]^\top$ and the task gradient is $g = 1$. The Jacobian is $J(\theta) = \nabla_{\boldsymbol{\theta}} w = [b, a] = [1, 1]$, and the parameter update is $\Delta\boldsymbol{\theta} = -\eta J(\theta)^\top g = [-\eta, -\eta]^\top$. The drift Hessian evaluates to $\mathcal{H}_{\text{Drift}} = g\nabla^2_{\boldsymbol{\theta}} w = 1 \cdot \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. Substituting these into Eq. (31), the inner product is:

$$\langle \Delta \boldsymbol{w}_{\text{Drift}}, -g \rangle = -\frac{1}{2} \begin{bmatrix} -\eta & -\eta \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} -\eta \\ -\eta \end{bmatrix} = -\frac{1}{2}(2\eta^2) = -\eta^2 < 0. \tag{33}$$

□

G.3 Proof for Proposition 3

Proposition (Spectral Suppression)**.** *Under $B = 0$ and $A_{i,j} \sim \mathcal{N}(0, \sigma_A^2)$, the expected signal curvature captured by LoRA satisfies $\mathbb{E}[\text{Tr}(\mathcal{H}_{\textit{Signal}})] = \alpha^2 \rho \, \text{Tr}(\mathbf{H}_\ell)$, where $\rho = \sigma_A^2 / r$.*

Proof. Using the property of vectorization $\text{vec}(XYZ) = (Z^\top \otimes X)\,\text{vec}(Y)$, we can express the LoRA mapping in the vectorized space as:

$$\boldsymbol{w} = \text{vec}(W) + \frac{\alpha}{r}(A^\top \otimes I_{d_{out}})\,\text{vec}(B), \tag{34}$$

where $I_{d_{out}}$ is the identity matrix of dimension d_{out}.

Let $J = \nabla_{\boldsymbol{\theta}} \boldsymbol{w} \in \mathbb{R}^{D \times p}$ denote the Jacobian matrix of this mapping, which can be partitioned into blocks corresponding to A and B, such that $J = [J_A, J_B]$. At initialization, since $B = 0$, the partial derivative with respect to A vanishes completely, yielding $J_A = 0$. The partial derivative with respect to B is given by:

$$J_B = \frac{\alpha}{r}(A^\top \otimes I_{d_{out}}). \tag{35}$$

The signal curvature projected into the parameter space is defined by $\mathcal{H}_{\text{Signal}} = J^\top \mathbf{H}_\ell J$. Taking the trace and applying its cyclic property $\text{Tr}(XYZ) = \text{Tr}(ZXY)$, we have:

$$\text{Tr}(\mathcal{H}_{\text{Signal}}) = \text{Tr}(J^\top \mathbf{H}_\ell J) = \text{Tr}(\mathbf{H}_\ell J J^\top) = \text{Tr}(\mathbf{H}_\ell J_B J_B^\top). \tag{36}$$

We now evaluate the expectation of the outer product $J_B J_B^\top$ over the random Gaussian initialization of A. Utilizing the mixed-product property of the Kronecker product, $(X \otimes Y)(U \otimes V) = (XU \otimes YV)$, we obtain:

$$J_B J_B^\top = \left(\frac{\alpha}{r}\right)^2 (A^\top A \otimes I_{d_{out}}). \tag{37}$$

By assumption, the entries of $A \in \mathbb{R}^{r \times d_{in}}$ are drawn i.i.d. from $\mathcal{N}(0, \sigma_A^2)$. Thus, the expectation of its Gram matrix is $\mathbb{E}[A^\top A] = r\sigma_A^2 I_{d_{in}}$. Substituting this into the expectation yields:

$$\mathbb{E}[J_B J_B^\top] = \frac{\alpha^2}{r^2}(\mathbb{E}[A^\top A] \otimes I_{d_{out}}) = \frac{\alpha^2}{r^2}(r\sigma_A^2 I_{d_{in}} \otimes I_{d_{out}}) = \frac{\alpha^2 \sigma_A^2}{r} I_D, \tag{38}$$

where I_D is the identity matrix of dimension $D = d_{in} d_{out}$.Finally, by the linearity of the trace and expectation operators, we conclude:

$$\mathbb{E}[\text{Tr}(\mathcal{H}_{\text{Signal}})] = \text{Tr}\left(\mathbf{H}_\ell \mathbb{E}[J_B J_B^\top]\right) = \text{Tr}\left(\mathbf{H}_\ell \frac{\alpha^2 \sigma_A^2}{r} I_D\right) = \frac{\alpha^2 \sigma_A^2}{r} \text{Tr}(\mathbf{H}_\ell). \tag{39}$$

Defining the variance-to-rank ratio as $\rho = \sigma_A^2 / r$, we arrive at the exact analytical expression:

$$\mathbb{E}[\text{Tr}(\mathcal{H}_{\text{Signal}})] = \alpha^2 \rho \, \text{Tr}(\mathbf{H}_\ell). \tag{40}$$

□

Lemma 1 (Hessian Spectral Bound and Stability)**.** *Consider a second-order differentiable loss function $\ell(\theta)$ with a local Hessian $\mathcal{H} = \nabla^2\ell(\theta)$. If the gradient $\nabla\ell(\theta)$ is L-Lipschitz continuous, the stability of gradient descent with a constant learning rate η is governed by the maximum eigenvalue of the Hessian, denoted as $\lambda_{\max}(\mathcal{H})$. Specifically, the optimization remains stable if:*

$$\eta \leq \frac{2}{L} = \frac{2}{\lambda_{\max}(\mathcal{H})}. \tag{41}$$

Furthermore, a reduction in $\lambda_{\max}(\mathcal{H})$ via spectral suppression increases the upper bound of the stable learning rate, effectively expanding the permissible hyperparameter space.

Proof. By the descent lemma for L-smooth functions [Nesterov, 2013], for any θ_t and $\theta_{t+1} = \theta_t - \eta\nabla\ell(\theta_t)$, we have:

$$\ell(\theta_{t+1}) - \ell(\theta_t) \leq -\eta\|\nabla\ell(\theta_t)\|^2 + \frac{L\eta^2}{2}\|\nabla\ell(\theta_t)\|^2 = -\eta\left(1 - \frac{L\eta}{2}\right)\|\nabla\ell(\theta_t)\|^2. \tag{42}$$

To ensure the loss is non-increasing ($\ell(\theta_{t+1}) \leq \ell(\theta_t)$), we require $1 - \frac{L\eta}{2} \geq 0$, which yields $\eta \leq 2/L$. In the quadratic approximation of the local optimization landscape, the Lipschitz constant L is equivalent to the spectral norm of the Hessian $\|\mathcal{H}\|_2$, which corresponds to its maximum eigenvalue $\lambda_{\max}(\mathcal{H})$. Thus, $\eta \leq 2/\lambda_{\max}(\mathcal{H})$. □

G.4 Proof for Proposition 4

Proposition (Asymmetric Scaling Effects)**.** *In the local regime, the Signal and Drift terms scale as $\mathcal{H}_{\text{Signal}} = \Theta(\alpha^2)$ and $\mathcal{H}_{\text{Drift}} = \Theta(\alpha)$, respectively. Accordingly, the corresponding update norms $(\|\Delta\boldsymbol{w}_{\text{Signal}}\|, \|\Delta\boldsymbol{w}_{\text{Drift}}\|)$ scale as $(\Theta(\alpha\eta), \Theta(\alpha\eta^2))$ for Adam and as $(\Theta(\alpha^2\eta), \Theta(\alpha^3\eta^2))$ for GD.*

Proof. Write $c = \alpha/r$ and keep the rank and adapter state (A, B) fixed. This is a local scaling comparison: the task gradient and Hessian are evaluated at the reference weight and treated as fixed, and the Θ statements for norms assume nonzero, non-cancelling contractions in the directions considered. The LoRA map is $W = W_0 + cBA$. Its Jacobian with respect to $\boldsymbol{\theta} = (A, B)$ is

$$J(\boldsymbol{\theta}) = c\left[I_{d_{in}} \otimes B,\ A^\top \otimes I_{d_{out}}\right], \tag{43}$$

so $J = \Theta(\alpha)$. Bilinearity gives $\nabla^2_{\boldsymbol{\theta}}\boldsymbol{w} = \Theta(\alpha)$, with only the mixed A–B blocks nonzero. By the Hessian decomposition in Proposition 1,

$$\mathcal{H}_{\text{Signal}} = J^\top\mathbf{H}_\ell J = \Theta(\alpha^2), \qquad \mathcal{H}_{\text{Drift}} = \sum_i g_i\nabla^2_{\boldsymbol{\theta}}w_i = \Theta(\alpha). \tag{44}$$

Considering the decomposition $\Delta\boldsymbol{w}_{\text{Signal}} = J(\boldsymbol{\theta})\Delta\boldsymbol{\theta}$ and $\Delta\boldsymbol{w}_{\text{Drift}} = \frac{1}{2}\nabla^2_{\boldsymbol{\theta}}\boldsymbol{w}[\Delta\boldsymbol{\theta}, \Delta\boldsymbol{\theta}]$. For GD, $\Delta\boldsymbol{\theta} = -\eta J^\top\boldsymbol{g} = \Theta(\alpha\eta)$, and $(\|\Delta\boldsymbol{w}_{\text{Signal}}\|, \|\Delta\boldsymbol{w}_{\text{Drift}}\|) = (\Theta(\alpha^2\eta), \Theta(\alpha^3\eta^2))$. For Adam, we have $\Delta\boldsymbol{\theta} = \Theta(\eta)$, which gives $(\|\Delta\boldsymbol{w}_{\text{Signal}}\|, \|\Delta\boldsymbol{w}_{\text{Drift}}\|) = (\Theta(\alpha\eta), \Theta(\alpha\eta^2))$. □

H Experimental Details

We summarize the core configurations and hyperparameters across all six evaluation domains in Table 9. Detailed learning rate schedules and environmental configurations strictly follow the referenced official codebases. Subsequent subsections provide granular details, including datasets and evaluation metrics, for each specific task.

[1] https://github.com/huggingface/transformers/tree/main/examples/pytorch/text-classification
[2] https://github.com/MuLabPKU/PiSSA
[3] https://github.com/huggingface/peft/tree/main/examples/lora_dreambooth
[4] https://github.com/TIGER-AI-Lab/VLM2Vec/tree/v1.0
[5] https://github.com/huggingface/open-r1
[6] https://github.com/MikaStars39/PeRL

Table 9: Comprehensive summary of experimental settings across all evaluation domains. The estimated wall-clock time (Est. Time) highlights our evaluation across diverse training horizons, ranging from rapid adaptation to multi-day post-training scenarios.

Task & Source	Backbone Model	Rank (r)	α Strategy	LR (η)	Batch Size	Duration	Est. Time
NLU[1]	DeBERTa-v3-base-184M	8	Variant II	1×10^{-4}	32	3–5 Epochs	Mins $\sim$ Hours
NLG[2]	Llama-2-7B	16 / 128	Variant II	2×10^{-5}	128	1 Epoch	Mins $\sim$ Hours
DreamBooth[3]	FLUX.1-12B	8	Variant II	1×10^{-4}	1	1,000 Steps	Mins $\sim$ Hours
Multimodal[4]	Qwen 2-VL (2B/7B)	16	Variant I	2×10^{-5}	1,024	2,000 Steps	3-4 Days
Reasoning SFT[5]	Qwen 2.5-Math-7B	64 / 256	Variant I	4×10^{-5}	128	1 Epoch	2–3 Days
Reasoning RL[6]	DeepSeek-R1-Distill (2B/7B)	64	Variant I	1×10^{-6}	128	1 Epoch	2–3 Days

Table 10: Comparison on NLG tasks using Llama 2-7B. Performance is reported as mean $\pm$ std over three independent runs.

Rank (r)	Method	GSM8K	MATH	HumanEval	MBPP	Commonsense
N/A	Full Fine-Tuning	60.3 $\pm$ 1.3	11.7 $\pm$ 0.6	32.3 $\pm$ 1.3	39.3 $\pm$ 1.0	79.2 $\pm$ 1.2
16	LoRA	31.5 $\pm$ 0.3	4.2 $\pm$ 0.3	16.0 $\pm$ 0.2	28.7 $\pm$ 0.5	66.6 $\pm$ 1.2
	RsLoRA	39.0 $\pm$ 0.5	4.9 $\pm$ 0.4	18.9 $\pm$ 0.7	28.1 $\pm$ 0.6	73.2 $\pm$ 0.8
	LoRA+	31.7 $\pm$ 0.6	4.0 $\pm$ 0.4	18.5 $\pm$ 0.5	28.0 $\pm$ 0.8	72.2 $\pm$ 1.4
	PiSSA	37.7 $\pm$ 0.5	5.2 $\pm$ 0.4	18.4 $\pm$ 0.5	28.6 $\pm$ 0.7	73.7 $\pm$ 1.1
	LoRAM	40.3 $\pm$ 0.4	5.3 $\pm$ 0.4	**18.9** $\pm$ 0.6	**28.8** $\pm$ 0.6	75.2 $\pm$ 1.1
	LoRA-α	**47.6** $\pm$ 0.6	**6.8** $\pm$ 0.4	**18.9** $\pm$ 0.2	28.3 $\pm$ 0.7	**77.4** $\pm$ 0.9
128	LoRA	40.3 $\pm$ 0.7	4.7 $\pm$ 0.4	20.1 $\pm$ 0.3	28.8 $\pm$ 0.4	73.6 $\pm$ 1.1
	RsLoRA	50.4 $\pm$ 0.4	7.3 $\pm$ 0.3	21.3 $\pm$ 0.7	30.7 $\pm$ 0.4	77.0 $\pm$ 1.2
	LoRA+	40.4 $\pm$ 0.7	5.3 $\pm$ 0.5	20.7 $\pm$ 0.9	29.1 $\pm$ 0.8	78.2 $\pm$ 1.3
	PiSSA	51.5 $\pm$ 0.8	7.0 $\pm$ 0.5	21.6 $\pm$ 0.5	31.1 $\pm$ 0.7	77.3 $\pm$ 1.0
	LoRAM	51.1 $\pm$ 0.7	7.3 $\pm$ 0.7	22.0 $\pm$ 0.6	31.5 $\pm$ 0.7	77.8 $\pm$ 1.0
	LoRA-α	**53.2** $\pm$ 0.8	**8.3** $\pm$ 0.4	**23.2** $\pm$ 0.5	**33.9** $\pm$ 0.8	**78.4** $\pm$ 0.9

H.1 Natural Language Understanding

Models and Datasets. We evaluate NLU performance using the DeBERTa-v3-base model (184M) [He et al., 2023] on eight tasks from the GLUE benchmark [Wang et al., 2018]. Following standard protocols, we include MNLI, SST-2, MRPC, CoLA, QNLI, QQP, RTE, and STS-B.

LoRA Configuration. All parameter-efficient methods are implemented with a fixed rank $r = 8$. For the proposed LoRA-α, we apply the analytic scaling (Variant II) to determine α_{base}. For other baselines, the scaling factor α follows their respective original designs. Adapters are uniformly applied to all linear layers.

Optimization. All models are trained using the AdamW optimizer with a constant learning rate of 1×10^{-4}, aligned with standard FFT settings. Training is conducted for 3 epochs across most tasks, with the exception of MRPC, which is trained for 5 epochs due to its limited sample size. We use a batch size of 32 and follow the default weight decay settings from the Transformers library.

Evaluation. We report task-specific metrics as per GLUE conventions: matched/mismatched accuracy for MNLI, Matthews correlation for CoLA, Pearson correlation for STS-B, and accuracy for the remaining tasks. Following [Meng et al., 2024; Zhang et al., 2025b], results are averaged over three independent runs.

H.2 Natural Language Generation

Models and Datasets. We evaluate generative performance via supervised fine-tuning of Llama 2-7B [Meta Team, 2023] across three distinct domains. Specifically, the model is trained on MetaMathQA [Yu et al., 2024] for mathematical reasoning (evaluated on GSM8K [Cobbe et al., 2021] and MATH [Yu et al., 2024]), CodeFeedback [Zheng et al., 2024] for code generation (evaluated on HumanEval [Zheng et al., 2023] and MBPP [Austin et al., 2021]), and Commonsense170K [Hu et al., 2023] for commonsense reasoning (reporting averaged accuracy across eight sub-datasets).

Table 11: Detailed hyperparameter search results on the GLUE benchmark. The table includes a grid search over learning rate (η) and scaling factor (α). The average is computed across all 8 tasks.

Configuration	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B	Avg.
Learning Rate (η) Search, fixing $\alpha = r$									
$\eta = 1.0 \times 10^{-5}$	87.0	93.0	68.4	0.0	90.0	87.7	47.3	24.1	62.2
$\eta = 2.5 \times 10^{-5}$	88.8	94.5	68.4	58.3	91.7	89.0	47.3	76.0	76.8
$\eta = 5.0 \times 10^{-5}$	89.6	95.0	80.6	61.1	92.9	89.9	47.3	81.4	79.7
$\eta = 7.5 \times 10^{-5}$	90.1	95.5	83.3	60.8	93.4	90.4	47.3	84.8	80.7
$\eta = 1.0 \times 10^{-4}$	90.2	95.9	84.1	63.6	93.9	90.6	50.2	87.2	81.9
$\eta = 2.5 \times 10^{-4}$	**90.7**	**96.2**	89.0	67.3	93.9	91.5	76.2	88.5	86.7
$\eta = 5.0 \times 10^{-4}$	90.5	95.1	89.5	68.5	**94.1**	**91.8**	82.3	89.7	87.7
$\eta = 7.5 \times 10^{-4}$	89.9	96.0	**91.1**	68.1	93.9	91.1	**84.5**	88.5	**87.9**
$\eta = 1.0 \times 10^{-3}$	88.0	94.1	89.2	**69.3**	92.0	90.0	82.1	**90.7**	86.9
Scaling Factor (α) Search, leveraging LoRA-α and fixing $\eta = 1.0 \times 10^{-4}$									
$\alpha = 0.1$	90.7	95.9	85.8	65.3	94.0	91.0	67.2	87.9	84.7
$\alpha = 0.25$	90.7	**96.4**	86.8	65.8	94.0	91.4	76.2	88.3	86.2
$\alpha = 0.5$	**90.8**	96.3	87.0	67.5	**94.1**	91.8	79.8	89.7	87.1
$\alpha = 0.75$	90.5	96.0	88.7	68.2	93.9	**92.1**	83.4	89.8	87.8
$\alpha = 1.0$	90.4	95.8	90.0	**70.9**	94.0	91.7	80.9	90.1	88.0
$\alpha = 2.5$	89.7	95.1	**90.4**	67.8	93.7	91.4	85.9	**91.1**	**88.1**
$\alpha = 5.0$	89.1	93.8	90.0	67.3	92.0	63.2	**87.0**	90.3	84.1
$\alpha = 7.5$	31.8	50.9	89.5	67.0	49.5	63.2	47.3	91.0	61.3
$\alpha = 10.0$	32.7	50.9	88.2	64.6	49.5	63.2	47.3	1.4	49.7
Aggregating above results									
LoRA-η^*	<u>90.7</u>	<u>96.2</u>	**91.1**	<u>69.3</u>	**94.1**	<u>91.8</u>	<u>84.5</u>	<u>90.7</u>	<u>88.5</u>
LoRA-α^*	**90.8**	**96.4**	<u>90.4</u>	**70.9**	**94.1**	**92.1**	**87.0**	**91.1**	**89.1**

Following the PiSSA training framework [Meng et al., 2024], each task strictly utilizes a sampled subset of 100k examples to ensure a balanced and computationally equivalent comparison.

LoRA Configuration. We evaluate performance across two distinct rank regimes: $r \in \{16, 128\}$. To explicitly validate the effectiveness of our theoretically derived scaling rule, LoRA-α strictly utilizes the analytic scaling (Variant II) to determine the scaling factor α. This setup verifies whether the analytic formulation can scale to LLMs. Adapters are uniformly applied to all linear layers.

Optimization. Training is performed using the AdamW optimizer with a learning rate of 2×10^{-5} and a total batch size of 128. All models are optimized for a single epoch, providing a stringent testbed to assess the rapid adaptation efficiency of different scaling strategies. Following [Meng et al., 2024; Zhang et al., 2025b], results are averaged over three independent runs. Table 10 augments the results in Table 2 by including standard deviations (std) for reference.

H.3 Detailed Hyperparameter Search Results

To validate the superiority of α-scaling over conventional η-scaling in generalization, we conduct an exhaustive hyperparameter grid search across both NLU and NLG tasks. The detailed, per-task performance breakdowns are provided in Table 11 and Table 12. *Given the massive scale of this hyperparameter search space, all results reported in these tables are based on a single experimental run per configuration.*

- **NLU Grid Search (Table 11):** We observe that strictly scaling the learning rate (η) often leads to suboptimal plateaus or local instability (*e.g.*, initial failure to converge on CoLA). In contrast, scaling the structural multiplier α unlocks higher performance ceilings across almost all datasets, raising the aggregated peak performance (LoRA-$\alpha^\star$) to 89.1, compared to 88.5 for η-scaling.
- **NLG Grid Search (Table 12):** The structural bottleneck of η-scaling becomes critically evident, especially at larger ranks ($r = 128$). Pushing η to extremes triggers catastrophic divergence (*e.g.*, performance collapsing to 0.0 on MATH and GSM8K). Conversely, α-scaling preserves

Table 12: Detailed hyperparameter search results on NLG tasks (Llama 2-7B). The table includes a grid search over learning rate (η) and scaling factor (α). The average is computed across the 5 tasks.

Configuration	GSM8K	MATH	HumanEval	MBPP	Commonsense	Avg.
Learning Rate (η) Search at Rank $r = 16$, fixing $\alpha = 16$						
$\eta = 2.0 \times 10^{-5}$	31.5	4.2	16.0	28.7	66.6	29.4
$\eta = 1.0 \times 10^{-4}$	42.6	5.9	17.7	29.4	74.7	34.1
$\eta = 2.5 \times 10^{-4}$	50.3	6.5	20.1	30.7	76.3	36.8
$\eta = 5.0 \times 10^{-4}$	**54.3**	8.4	**24.4**	33.6	**78.2**	**39.8**
$\eta = 7.5 \times 10^{-4}$	52.9	9.1	23.8	34.4	**78.2**	39.7
$\eta = 1.0 \times 10^{-3}$	51.4	**9.5**	22.2	**34.7**	78.1	39.2
Scaling Factor (α) Search at Rank $r = 16$, leveraging LoRA-α and fixing $\eta = 2.0 \times 10^{-5}$						
$\alpha = 1.0$	47.6	6.8	18.9	28.3	77.4	35.8
$\alpha = 2.5$	51.7	7.6	21.3	32.5	**78.2**	38.3
$\alpha = 5.0$	54.5	9.2	26.2	34.7	78.2	**40.6**
$\alpha = 7.5$	**55.3**	9.4	23.8	**35.4**	78.0	40.4
$\alpha = 10.0$	52.6	**9.8**	**28.7**	32.5	76.3	40.0
Learning Rate (η) Search at Rank $r = 128$, fixing $\alpha = 128$						
$\eta = 2.0 \times 10^{-5}$	40.3	4.7	20.1	28.8	73.6	33.5
$\eta = 1.0 \times 10^{-4}$	52.4	7.7	21.3	33.9	77.8	38.6
$\eta = 2.5 \times 10^{-4}$	56.7	9.8	25.6	35.2	**78.7**	**41.2**
$\eta = 5.0 \times 10^{-4}$	**58.8**	**10.8**	**28.7**	**36.5**	28.2	32.6
$\eta = 7.5 \times 10^{-4}$	1.3	1.4	28.4	34.1	27.5	18.5
$\eta = 1.0 \times 10^{-3}$	0.0	0.0	28.0	33.9	27.4	17.9
Scaling Factor (α) Search at Rank $r = 128$, leveraging LoRA-α and fixing $\eta = 2.0 \times 10^{-5}$						
$\alpha = 1.0$	53.2	8.3	23.2	33.9	78.4	39.4
$\alpha = 2.5$	57.5	9.8	27.4	36.8	**79.3**	42.2
$\alpha = 5.0$	59.3	**11.7**	28.7	36.0	76.9	**42.5**
$\alpha = 7.5$	**60.0**	11.1	29.3	**39.9**	72.1	42.5
$\alpha = 10.0$	58.5	11.3	**33.5**	35.2	73.3	42.4
Aggregating above results						
LoRA-η^* ($r = 16$)	54.3	9.5	24.4	34.7	**78.2**	40.2
LoRA-α^* ($r = 16$)	**55.3**	**9.8**	**28.7**	**35.4**	**78.2**	**41.5**
LoRA-η^* ($r = 128$)	58.8	10.8	28.7	36.5	78.7	42.7
LoRA-α^* ($r = 128$)	**60.0**	**11.7**	**33.5**	**39.9**	**79.3**	**44.9**

optimization stability under the same aggressive scaling regimes, successfully driving the model to deeper local minima and extending the frontier significantly (averaging 44.9 vs. 42.7).

Ultimately, aggregating the optimal configurations explicitly demonstrates that amplifying the signal via α maintains optimization advantage.

H.4 Text-to-Image Synthesis

Models and Datasets. We evaluate multimodal generation using the Flux.1-12B model [Labs, 2024] for image customization via the DreamBooth protocol [Ruiz et al., 2023]. The implementation is based on the official PEFT DreamBooth examples. The model learns novel concepts from small sets of instance images (typically 4–6) to generate high-fidelity, prompt-conditioned images.

LoRA Configuration. For LoRA-α, we apply analytic scaling (Variant II) to calibrate α_{base} based on the Flux architecture, with a fixed rank of $r = 8$. Adapters are applied to all attention layers.

Optimization. Models are optimized using AdamW with a constant learning rate of 1×10^{-4} and a batch size of 1. Training is conducted for 1,000 steps. This setting provides a challenging environment to test whether α-scaling can accelerate concept acquisition without over-fitting.

Evaluation. Following DreamBooth, we use the unique identifier `"sks"` for concept binding. To test the model's ability to preserve the learned concept under novel contexts, evaluation prompts include category and scene descriptions: `"A sks <object_name> in a sunlit forest, vibrant foliage, ultra-realistic photography"` and `"A sks <object_name>, centered on a pure white background, ultra-realistic photography"`.

H.5 Multimodal Representation Learning

Models and Datasets. We evaluate LoRA-α on the MMEB benchmark [Jiang et al., 2025], which converts generative models into dense retrievers. We use Qwen 2-VL (2B and 7B) [Wang et al., 2024a] as backbones, implementing the contrastive training pipeline based on the VLM2Vec codebase. The dataset includes 36 subsets; 20 in-distribution (ID) subsets are used for training, while 16 out-of-distribution (OOD) subsets are reserved for zero-shot evaluation.

LoRA Configuration. For LoRA-α, we adopt Variant I with $\alpha_{\text{base}} = 256\sqrt{r}$. Adapters are applied to all linear layers with a fixed rank of $r = 16$.

Optimization. Training employs the InfoNCE contrastive loss (temperature 0.02) and the AdamW optimizer. We use a global batch size of 1,024 and a learning rate of 2×10^{-5}. Training lasts for 2,000 steps with gradient caching enabled.

Evaluation. Representations are extracted from the final-token hidden state. Performance is measured via Precision@1 averaged across ID and OOD splits. This setup rigorously tests the model's ability to maintain pretrained multimodal alignment while adapting to discriminative objectives.

H.6 Experimental Details for Reasoning-based SFT

Models and Datasets. We employ Qwen 2.5-Math-7B [Qwen Team, 2024] as the backbone for long-context reasoning SFT. Training is conducted on the Mixture-of-Thoughts dataset (350k samples) [HuggingFace Team, 2025] utilizing the training recipes provided by Open-R1.

LoRA Configuration. We apply adapters to all linear layers with ranks $r \in \{64, 256\}$. Furthermore, to support the generation of special tokens, we additionally unfreeze and train the embedding and output layers. LoRA-α uses Variant I scaling.

Optimization. Models are trained for 1 epoch using the Adam optimizer. We use a peak learning rate of 4.0×10^{-5}. The effective batch size is 128. We enable gradient checkpointing and set the maximum sequence length to 32,768 tokens.

Evaluation. Following the established protocols of Open-R1, we evaluate reasoning performance using the Lighteval framework [Habib et al., 2023]. Our evaluation spans a comprehensive suite of challenging benchmarks, including AIME 24/25 [MAA, 2024, 2025], MATH-500 [Hendrycks et al., 2021], GPQA Diamond [Rein et al., 2024], and LiveCodeBench v4 [Jain et al., 2025]. Performance is reported as Pass@1 accuracy, where the number of generation samples per prompt strictly adheres to the Open-R1 configuration to ensure a standardized and fair comparison across different methods.

H.7 Experimental Details for Reasoning-based RL

Models and Datasets. We transition to policy optimization using Group Relative Policy Optimization (GRPO) [DeepSeek Team, 2025]. We use DeepSeek-R1-Distill-Qwen (1.5B and 7B) as base models, trained on the DAPO-Math-17k dataset [Yu et al., 2025]. We implement the RLVR pipeline based on the PeRL framework [Yin et al., 2025]. This paradigm optimizes reasoning paths using outcome-based rewards.

LoRA Configuration. Adapters are applied to all linear layers with a rank of $r = 64$. We utilize Variant I for LoRA-α. To accelerate rollouts, we use a co-located vLLM generation engine [Kwon et al., 2023] with 0.4 memory utilization.

Optimization. Training is performed with a constant learning rate of 1×10^{-6}, a KL penalty $\beta = 0$, and a total batch size of 128. For each prompt, we generate $G = 8$ rollouts. The maximum completion length is set to 16,384 tokens to test long-form stability.

Evaluation. We assess reasoning performance on MATH-500 [Hendrycks et al., 2021], AMC 23 [MAA, 2023], AIME 24/25 [MAA, 2024, 2025], Minerva [Lewkowycz et al., 2022], and HMMT [Balunović et al., 2025], using Lighteval framework [Habib et al., 2023]. Performance is reported as Pass@1 accuracy estimated over multiple samples.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly outline the three main claims regarding LoRA's spectral suppression, the asymmetric roles of the scaling factor and learning rate, and the proposed sublinear scaling law, all of which are supported by the experiments and theoretical frameworks.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Limitations are explicitly discussed in Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: Complete and formal proofs for all theoretical propositions are provided in Appendix G.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: The paper fully discloses training details, hyperparameter search spaces, baselines, and implementation configurations in Section 5 and provides comprehensive task-by-task specifics in Appendix H.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: The paper utilizes standard, publicly available datasets and relies on open-source libraries like PEFT, providing detailed configuration parameters in the text and appendices to faithfully reproduce the results.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Comprehensive details are elaborated in Appendix H.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: While formal statistical significance tests and error bars are not universally reported due to the high computational cost of fine-tuning large language models, we ensure the reliability and robustness of our conclusions by providing an extensive and diverse suite of experiments. Our evaluation spans multiple model scales (184M to 12B), diverse task domains (NLU, NLG, reasoning, and multimodal synthesis), and various training paradigms. We also provide std value in Table 10. This extensive diversity of experiments consistently validates our findings across different settings.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The paper specifies the use of eight NVIDIA H800 GPUs in Section 5, and details specific setups in Appendix H.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research strictly focuses on optimization methodology for machine learning and conforms to the Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

Justification: A Broader Impact statement is included in Appendix B.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper proposes an optimization framework for existing foundational models and does not release new high-risk datasets or original pre-trained models.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All datasets, models, and software libraries used are appropriately credited and cited in the main text and appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper introduces a methodological framework for LoRA optimization and does not release new data assets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The research does not involve crowdsourcing or human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The research does not involve human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: While LLMs are utilized as the subjects for fine-tuning to validate the optimization framework, they are not used as an important, original, or non-standard tool in developing the core methodology itself.

    Guidelines:

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