STABLE-LORA$^+$: REGULARIZING LOW-RANK ADAPTATION FOR ROBUST SCALING

Anonymous authors
Paper under double-blind review

ABSTRACT

The efficiency of low-rank adaptation does not ensure stable training as language models increase in size. Settings that are effective at one scale may degrade accuracy or lead to divergence at another. We study robust scaling of adaptation for mathematical reasoning, evaluating training stability and predictive accuracy across model sizes under a fixed training configuration, separately from performance after tuning. We introduce *Stable-LoRA*$^+$, a training method that regularizes the factorized adaptation process. Its premise is that scaling the adapter output alone does not fully constrain factor evolution. The method therefore controls factor growth during training while penalizing update components aligned with the pretrained weight's leading singular subspaces. The same regularization is explicitly implemented for 4-bit backbones. Across more than 130 training-and-evaluation runs on five Qwen backbones (0.5B–14B), the strict fixed-configuration recipe matches vanilla LoRA's macro-average GSM8K accuracy (69.55%). With protocol and seed selection, the broader best-protocol comparison reaches 70.14% and mean rank 1.50. At $\mathrm{lr}{=}10^{-3}$, vanilla LoRA diverges on 3/7B, losing at least 76 percentage points, while our shared 10^{-4} configuration remains stable across the evaluated backbones. A quantization audit also identifies a silently inactive anchor, whose correction recovers 1.36 points at 7B. These results motivate regularization of factorized updates as an approach to robust adaptation across the evaluated model scales.

1 INTRODUCTION

Low-rank adaptation (LoRA) (Hu et al., 2021) makes language-model fine-tuning affordable by training a small update while leaving the backbone frozen. But reducing the number of trainable parameters does not resolve another practical cost: deciding how to train them. When a model grows or its weights are quantized, an adapter configuration that previously worked may become inaccurate or unstable. The resulting search over learning rates and adaptation variants can require multiple runs before the computationally efficient fine-tuning procedure becomes useful. We study robust scaling: the stability and predictive performance of adaptation as backbone size changes. Each backbone is trained separately; keeping the training settings fixed tests this behavior without per-model hyperparameter selection.

The growing family of LoRA variants offers many candidate controls. Rank scaling, initialization, factor-wise optimization, and adaptive rank allocation each modify a different aspect of the update (Kalajdzievski, 2023; Meng et al., 2024; Hayou et al., 2024; Zhang et al., 2023). Yet a gain from one intervention does not establish that it transfers across scales, or that it remains beneficial when combined with another. Our experiments expose this distinction: initialization choices that are useful in isolation can become unstable with amplified scaling, while a regularizer can appear enabled but contribute no constraint under quantization. Understanding stability across scales therefore requires examining how these interventions affect the training process.

Learning-rate tuning makes this problem particularly easy to misinterpret. Lee et al. (2026) report that tuning narrows the gap between vanilla LoRA and its variants to about one percentage point on math and code. This addresses the accuracy attainable after a search. It does not establish that the same configuration is reliable across backbones. In our sweep, vanilla LoRA loses at least 76 percentage points on 3/7B at $\mathrm{lr}{=}10^{-3}$. Our fixed-configuration evaluation instead holds $\mathrm{lr}{=}10^{-4}$ fixed. Together,

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

these comparisons separate the accuracy recovered by a search from the reliability of a specified configuration across backbones.

Our starting point is the factorization $\Delta W = sBA$. Replacing A with cA and B with B/c preserves the update but changes the factor norms. Output scaling therefore leaves aspects of factor evolution unconstrained. We investigate whether regularizing that evolution and the directions represented by BA improves training stability across scales (Figure 1).

Stable-LoRA$^+$ applies this idea during adapter training. Rank-stabilized scaling (Kalajdzievski, 2023) is introduced gradually while gated A-side shrinkage (Wu et al., 2026) limits factor growth according to the prescribed norm criterion. Since factor norms do not determine update orientation, the objective also penalizes the projection of BA onto the pretrained weight's leading singular subspaces. Under QLoRA (Dettmers et al., 2023), computing those subspaces requires recovering the base matrix from quantized storage. Correcting an inactive penalty recovers 1.36 pp at 7B, demonstrating the importance of implementation fidelity when evaluating the intended regularization.

We evaluate this hypothesis in more than 130 runs across five Qwen backbones (Yang et al., 2024; Qwen et al., 2025), eleven PEFT baselines, and a five-rate vanilla-LoRA sweep, using MetaMathQA (Yu et al., 2024) and the full GSM8K test set (Cobbe et al., 2021). We separate the strict fixed configuration (69.55% macro-average, matching vanilla LoRA) from the best-protocol comparison (70.14%, mean rank 1.50), which includes protocol and seed selection and a reduced recipe at 0.5B. The study connects a factorization-aware training method to observed cross-scale dynamics, quantization behavior, and controlled ablations. It assesses cross-scale training behavior separately from the accuracy attainable through protocol selection.

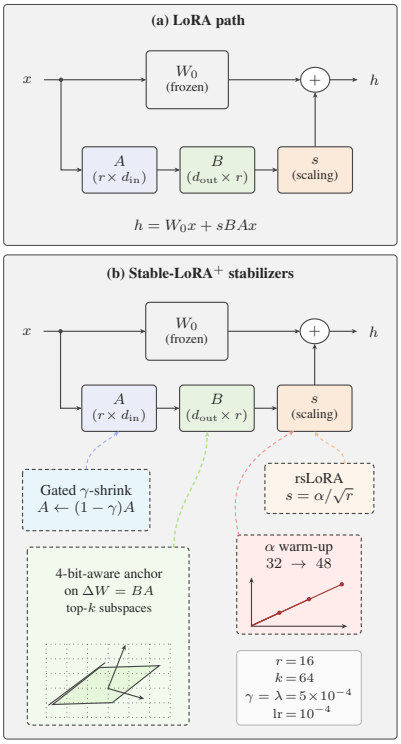


Figure 1: Forward path and training controls in Stable-LoRA$^+$. The 4-bit-aware anchor uses subspaces of the dequantized base weight.

2 Related Work

Standard LoRA and rank effects. LoRA (Hu et al., 2021) freezes the pretrained weight $W^{(0)} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ and parameterizes the update as $\Delta W = (\alpha/r)\, BA$ with $A \in \mathbb{R}^{r \times d_{\text{in}}}$, $B \in \mathbb{R}^{d_{\text{out}} \times r}$ and small rank r. The original α/r scaling causes effective updates to collapse as r grows; rsLoRA (Kalajdzievski, 2023) replaces α/r with $\alpha/\sqrt{r}$.

Initialization-side variants. PiSSA (Meng et al., 2024) initializes A, B from the top-r SVD of $W^{(0)}$ and freezes the residual; OLoRA (Büyükakyüz, 2024) uses orthonormal QR initialization; LoRA-GA (Wang et al., 2024) initializes from the leading singular components of the first full-batch gradient. These methods accelerate convergence in their proposed settings but are unstable when composed with rsLoRA's amplified scaling on multi-billion-parameter backbones (Table 1).

Optimization-side variants. LoRA+ (Hayou et al., 2024) uses an asymmetric learning rate for B versus A. Stable-LoRA (Wu et al., 2026) prescribes A-side multiplicative shrinkage in early training from a feature-stability perspective; we use gated shrinkage to regularize factor evolution under rank-stabilized scaling. A separate line — LoRA-Pro (Wang et al., 2025), AltLoRA (Yu et al., 2026), and LoRA-MGPO (Chang et al., 2025) — reshapes LoRA's gradient or perturbs its weights to approximate full fine-tuning; these primarily modify the gradient approximation, whereas our method regularizes the factorized update during training.

Budget allocation and rank reparameterization. AdaLoRA (Zhang et al., 2023) adaptively allocates rank across layers via a soft SVD; DoRA (Liu et al., 2024) decomposes the update into magnitude and direction; VeRA (Kopiczko et al., 2024) shares random adapter matrices modulated by trainable scaling vectors. A broader family reparameterizes the rank or budget itself — LoRA-XS, MoRA, MELoRA, LoRA-Mini, the mixture-of-experts MiLoRA, the 2-bit LowRA, and the random-matrix-theory α-LoRA (Bałazy et al., 2024; Jiang et al., 2024; Ren et al., 2024; Singh et al., 2024; Zhang et al., 2024; Zhou et al., 2025; Firdoussi et al., 2025). These mechanisms address rank or budget allocation, whereas our recipe fixes r and the α schedule; we report DoRA and VeRA as strong baselines and discuss the rest as concurrent work.

Quantization-aware adaptation. QLoRA (Dettmers et al., 2023) introduces 4-bit NormalFloat (NF4) quantization for the frozen backbone so adapters of large models fit in commodity GPU memory. We rely on QLoRA for 7/14B runs and report a silent failure mode of subspace-anchored regularizers on `Linear4bit` layers.

Math reasoning and LLM benchmarks. GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) are the de facto benchmarks for multi-step mathematical reasoning, with MetaMathQA (Yu et al., 2024) the standard fine-tuning corpus; we follow the chain-of-thought, instruction-tuned protocol (Wei et al., 2022; 2021; Luo et al., 2025).

The learning-rate-matters debate. Lee et al. (2026) re-evaluate LoRA variants under per-method learning-rate sweeps and report that all major variants perform within 1 pp of vanilla LoRA. Our results distinguish performance after tuning from sensitivity to the chosen learning rate: $\mathrm{lr}{=}10^{-3}$ causes losses of at least 76 pp on 3/7B (Figure 2d).

3 Method: Stable-LoRA$^+$

Preliminaries. Let $W^{(0)} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ be a frozen pretrained linear weight. Standard LoRA injects $\Delta W = s \cdot BA$ with rank $r \ll \min(d_{\text{out}}, d_{\text{in}})$, scaling factor s, Kaiming-initialized A and zero-initialized B.

Rank-stabilized scaling. Following Kalajdzievski (2023), $s = \alpha/\sqrt{r}$, which mitigates the rank dependence of update and gradient magnitudes. We fix $r{=}16$, while Figure 4(c) sweeps $r \in \{4, 8, 16, 32, 64\}$ to map the rank dependence.

Linear α-schedule. Our fixed-α ablation shows a trade-off between accuracy and stability: $\alpha{=}32$ yields lower accuracy, whereas $\alpha{=}64$ exhibits late instability. We use a linear warm-up schedule:

$$\alpha(t) = \begin{cases} \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \, \dfrac{t}{T_{\text{warm}}}, & t < T_{\text{warm}}, \\ \alpha_{\max}, & t \geq T_{\text{warm}}, \end{cases} \tag{1}$$

with $\alpha_{\min}{=}32$, $\alpha_{\max}{=}48$, and $T_{\text{warm}}{=}0.30\,T_{\text{total}}$, applied uniformly to all adapter modules.

A-side multiplicative shrinkage. Wu et al. (2026) argue that the asymmetric initialization ($B{=}0$, A Kaiming) destabilizes early feature learning. Inside our AdamW (Loshchilov & Hutter, 2017) optimizer we apply $A \leftarrow (1{-}\gamma)A$, $\gamma{=}5{\times}10^{-4}$, to the down-projection after each step (B untouched). The shrinkage is *gated*: applied only while A still dominates the adapter norm ($\|A\|_F \geq (|A|/|B|)\|B\|_F$) and disabled otherwise, so the adapter is not driven to degeneracy late in training (Appendix E).

Corrected subspace anchor. To control overlap between the unscaled update BA and the leading singular subspaces of $W^{(0)}$, we impose a bilinear penalty on the adapter's projection onto the top-k left and right singular subspaces of each base weight:

$$\mathcal{L}_{\text{anchor}} = \frac{\lambda}{k\,r} \sum_{\ell} \left\| U_{k,\ell}^{\top} B_\ell A_\ell \, V_{k,\ell} \right\|_F^2, \tag{2}$$

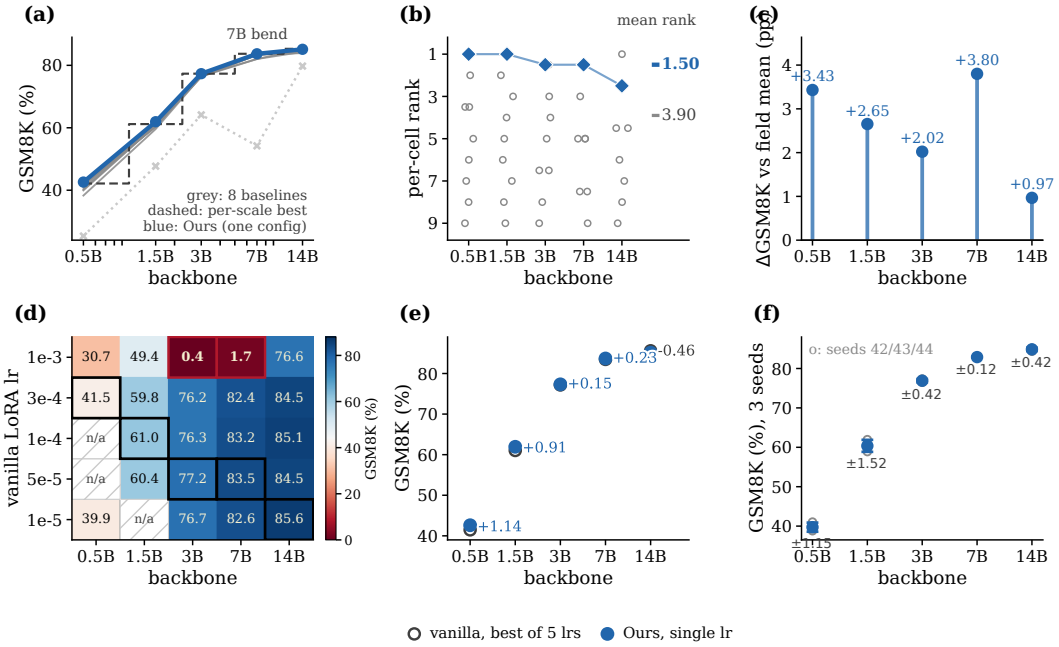


Figure 2: Stable-LoRA$^+$ across model scales. (a) GSM8K accuracy vs backbone size for the best-protocol checkpoints: the single-LR field in grey (dotted: OLoRA, which collapses), the per-scale best baseline (dashed), and Stable-LoRA$^+$ in blue. (b) Per-cell rank across the five backbones among the nine single-LR methods (open circles: the eight baselines; diamonds: Stable-LoRA$^+$; lower is better); Stable-LoRA$^+$ attains the lowest mean rank (1.50, versus 3.90 for the next best) and is the only method never to leave the top band. (c) Per-cell GSM8K gain of Stable-LoRA$^+$ over the cross-method mean: above zero at every scale. (d) Learning-rate sensitivity of vanilla LoRA: GSM8K over five learning rates and five backbones; outside a narrow per-scale band the penalty exceeds 80 pp, with a sharp performance decline at $\mathrm{lr}{=}10^{-3}$ on 3/7B (red box); the per-scale optimum (black box) shifts from $3{\times}10^{-4}$ at 0.5B to 10^{-5} at 14B; hatched cells have no recorded run. (e) Vanilla LoRA's best-of-five-LR accuracy per scale vs Stable-LoRA$^+$ at a single fixed $\mathrm{lr}{=}10^{-4}$ using best-protocol checkpoints. (f) Three-seed spread of the full recipe at every scale (σ from 0.12 to 1.52 pp, smallest at 7B QLoRA).

where $U_{k,\ell}, V_{k,\ell}$ are the top-k left/right singular vectors of $W_\ell^{(0)}$ and $1/(kr)$ keeps λ scale-comparable. We fix $k{=}64$, $\lambda{=}5{\times}10^{-4}$; Figure 4(b) sweeps the pair.

4-bit-aware implementation. Under QLoRA the linear modules become bitsandbytes `Linear4bit`, whose `.weight` is an opaque NF4 tensor rather than the matrix SVD needs, so a layer-selection path that assumes ordinary dense weights can leave the anchor inactive in the evaluated implementation. We instead detect the quantized module, dequantize to recover $W^{(0)}$, and take a randomized top-k SVD once at startup (Appendix E).

Integrated training procedure. Algorithm 1 summarizes one training step. We fix the same hyperparameter set across every backbone ($r{=}16$, $\alpha_{\min}{=}32$, $\alpha_{\max}{=}48$, warm-up 0.30, $\gamma{=}\lambda{=}5{\times}10^{-4}$, $k{=}64$), targeting the $\{q, k, v, o\}$-projections.

4 Experiments

Setup. We fine-tune Qwen-2-0.5B/1.5B and Qwen-2.5-3B/7B/14B (Yang et al., 2024; Qwen et al., 2025). Training uses the first 10,000 MetaMathQA instances unless noted; the data-scaling ablation uses 30,000. We evaluate all 1,319 GSM8K test problems with greedy decoding and exact-match scoring. AdamW uses $\beta_1{=}0.9$, $\beta_2{=}0.999$, weight decay 0.01, default learning rate 10^{-4}, one epoch,

Algorithm 1 One training step of Stable-LoRA$^+$

Input: minibatch (x, y), frozen $W^{(0)}$, adapter (A, B), step t, total steps T
Parameters: $r, \alpha_{\min}, \alpha_{\max}, \gamma, \lambda, k$
Pre-computed: U_k, V_k from SVD of dequantized $W^{(0)}$

$\alpha(t) \leftarrow$ linear schedule between $\alpha_{\min}, \alpha_{\max}$
$s \leftarrow \alpha(t)/\sqrt{r}$; set adapter scaling to s
$\hat{y} \leftarrow \text{forward}(x; W^{(0)} + s\,BA)$; $\mathcal{L}_{\text{task}} \leftarrow \text{NLL}(\hat{y}, y)$
$\mathcal{L}_{\text{anc}} \leftarrow \frac{\lambda}{kr} \sum_\ell \|U_k^\top B_\ell A_\ell V_k\|_F^2$
backward and AdamW step on $\mathcal{L}_{\text{task}} + \mathcal{L}_{\text{anc}}$
if $\|A\|_F \geq \frac{|A|}{|B|}\|B\|_F$: $A \leftarrow (1-\gamma)\,A$ {gated A-shrink}
return (A, B)

Method	Year	Venue	GSM8K accuracy (%) by Qwen backbone					Mean
			0.5B	1.5B	3B	7B	14B	
LoRA (Hu et al., 2021)	2022	ICLR	42.15	60.20	**77.33**	83.47	84.61	69.55
AdaLoRA (Zhang et al., 2023)	2023	ICLR	40.79	61.18	76.72	82.03	85.14	69.17
QLoRA (Dettmers et al., 2023)	2023	NeurIPS	41.62[§]	60.42[§]	76.50[§]	83.32	84.69	69.31
rsLoRA (Kalajdzievski, 2023)	2023	arXiv	39.95	60.73	76.27	**83.70**	**85.29**	69.19
DoRA (Liu et al., 2024)	2024	ICML	40.26	60.80	76.65	83.32	83.93	68.99
LoRA+ (Hayou et al., 2024)	2024	ICML	24.64[‡]	39.20[‡]	2.50[‡]	0.00[‡]	–[◇]	–
PiSSA (Meng et al., 2024)	2024	NeurIPS	2.05[‡]	24.34[‡]	36.47[‡]	–[‡]	–[‡]	–
LoRA-GA (Wang et al., 2024)[♯]	2024	NeurIPS	38.29	59.59	76.42	82.03	84.31	68.13
VeRA (Kopiczko et al., 2024)	2024	ICLR	28.58	28.96	51.71	58.45	–[♭]	–
OLoRA (Büyükakyüz, 2024)	2024	arXiv	25.32	47.69	64.14	54.21	79.76	54.22
Stable-LoRA (Wu et al., 2026)	2026	arXiv	41.62	61.03	76.42	83.32	84.69	69.42
Oracle: per-scale LR sweep (5 rates)								
LoRA (tuned LR) (Lee et al., 2026)	2026	arXiv	42.15	61.03	77.18	83.47	**85.60**	69.89
Stable-LoRA$^+$ (Ours)[♮]	—	—	**42.61**	**61.94**	**77.33**	**83.70**	85.14	**70.14**

Table 1: GSM8K (%) across 11 PEFT methods and five Qwen backbones (r=16, AdamW, default LR 10^{-4} unless noted, 1 epoch on 10K MetaMathQA, single-greedy on the full 1,319-problem test set). Bold marks per-column best among the single-LR rows; the "LoRA (tuned LR)" oracle (best of five rates per scale) is excluded from the bold. Stable-LoRA$^+$ is unique-best at 0.5/1.5B, tied first at 3/7B, and within seed-σ at 14B, with the highest macro-average of all methods (70.14%). Failure-mode markers ([♯]LoRA-GA re-run after a code fix, [‡]PiSSA, [◇]LoRA+, [♭]VeRA, [§]QLoRA) are detailed in Appendix D; [♮]best-protocol cells (the strict single-config 10K full-recipe macro-average is 69.55%, level with vanilla LoRA; the 3-seed-mean macro-average is 68.95%) are detailed in Appendix F and Table 3. [§]QLoRA reuses LoRA's adapter and substitutes NF4 for the backbone.

and effective batch size 16. Backbones up to 3B use `bfloat16`; 7/14B use NF4 QLoRA. Each run uses one RTX-3090 (24 GB), with seeds $\{42, 43, 44\}$ for repeated-seed evaluations.

4.1 Main comparison and Pareto frontier

Table 1 compares Stable-LoRA$^+$ with eleven PEFT baselines across five backbones; Figure 2 visualizes the results. Under best-protocol selection, Stable-LoRA$^+$ is unique-best at 0.5/1.5B, tied with vanilla LoRA at 3B and rsLoRA at 7B, and 0.15 pp below rsLoRA at 14B, within the recipe's seed standard deviation (σ=0.42). Its macro-average is 70.14%, compared with 69.89% for the per-scale LR oracle and 69.55% for default vanilla LoRA. These aggregate gains include protocol and seed selection, as documented in Appendix F; the strict 10K fixed recipe averages 69.55%. We therefore interpret the best-protocol ranking as a characterization of attainable performance, and use the fixed-recipe runs to assess cross-scale hyperparameter reuse. The comparison also reveals pronounced failures: several variants collapse or diverge, and the strongest baseline changes with scale.

We summarize the best-protocol results using average ranks among the nine eligible single-LR methods (Table 2). Stable-LoRA$^+$ has mean rank 1.50, followed by vanilla LoRA at 3.90. The separate per-scale

Method	0.5B	1.5B	3B	7B	14B	Mean
Stable-LoRA$^+$ (Ours)	**1.0**	**1.0**	**1.5**	**1.5**	2.5	**1.50**
LoRA (default)	2.0	7.0	**1.5**	3.0	6.0	3.90
AdaLoRA	5.0	2.0	3.0	7.5	2.5	4.00
rsLoRA	7.0	5.0	8.0	**1.5**	**1.0**	4.50
Stable-LoRA	3.5	3.0	6.5	5.0	4.5	4.50
QLoRA	3.5	6.0	5.0	5.0	4.5	4.80
DoRA	6.0	4.0	4.0	5.0	8.0	5.40
LoRA-GA	8.0	8.0	6.5	7.5	7.0	7.40
OLoRA	9.0	9.0	9.0	9.0	9.0	9.00

Table 2: Per-cell rank across the five backbones among Stable-LoRA$^+$ and eight single-LR baselines (lower is better, ties shared; oracle LoRA-tuned-LR excluded). Stable-LoRA$^+$ has the lowest mean rank (1.50), unique-best at 0.5/1.5B and tied for first at 3/7B; vanilla LoRA is second (3.90). Methods that fail to instantiate or destabilize are excluded.

Backbone	S42	S43	S44	Mean $\pm$ std
0.5B	41.02	38.82	39.35	39.73 ± 1.15
1.5B	61.94	58.91	60.27	60.37 ± 1.52
3B	76.88	76.50	77.33	76.90 ± 0.42
7B	82.87	83.02	82.79	82.89 ± 0.12
14B	85.06	84.38	85.14	84.86 ± 0.42

Table 3: Stable-LoRA$^+$: 3-seed GSM8K accuracy at every scale. The observed standard deviation is smallest at 7B QLoRA; this comparison does not isolate the effect of quantization.

LR-oracle comparison reports mean rank 2.10. These rankings summarize the reported checkpoints; protocol and seed selection prevent interpreting them as a comparison under equal search budgets.

4.2 Robustness profile

Appendix D distinguishes optimization instability from implementation incompatibility. PiSSA and LoRA+ exhibit failures under the evaluated settings, including incompatibilities with the NF4 path. AdaLoRA requires a corrected update order, and LoRA-GA requires an explicit initialization implementation; we report their corrected results. OLoRA and VeRA converge to lower-accuracy solutions. Vanilla LoRA diverges at $\mathrm{lr}{=}10^{-3}$ on 3/7B, whereas DoRA incurs additional training time (Appendix B). These outcomes characterize the tested configurations, not intrinsic limitations of each method. The fixed Stable-LoRA$^+$ runs complete without observed divergence; the catalogue is a descriptive audit, not an independent robustness benchmark.

4.3 Multi-seed measurements and LR sensitivity

We evaluate Stable-LoRA$^+$ with three seeds at every scale under the canonical 10K protocol (Table 3). The standard deviation ranges from 0.12 to 1.52 pp, with the largest variability at the smaller backbones. At 1.5B, the selected-checkpoint advantage over Stable-LoRA (0.91 pp) is smaller than this variability. Per-question comparisons (Appendix G) characterize disagreement between selected checkpoints but do not establish significance across seeds. The mean rank of 1.50 therefore describes the best-protocol comparison rather than a seed-robust ordering.

The vanilla-LoRA sweep (Figure 2d) shows that the best observed rate shifts from 3×10^{-4} at 0.5B to 10^{-5} at 14B. Among non-divergent runs, differences at 3/7/14B are relatively small, consistent with Lee et al. (2026); at 10^{-3}, losses reach 76 pp on 3B and 82 pp on 7B. Our shared 10^{-4} configuration remains stable, but this does not establish robustness to all learning rates.

We also compare the fixed rate with the per-scale rate selected for vanilla LoRA, using three seeds for our recipe. The fixed 10^{-4} rate yields higher mean accuracy by 0.13–1.16 pp on 3/7/14B (Table 5). Thus, transferring vanilla LoRA's selected rate does not improve the recipe in these comparisons. A complete per-method sweep would be needed to characterize its learning-rate response more fully.

4.4 Scaling behavior

The best-protocol accuracy curve exhibits diminishing returns with model scale (Figure 2a). Its approximate slope decreases from 45 pp per decade through 3B to 16 between 3B and 7B and 4.7 between 7B and 14B. This trend is specific to the evaluated checkpoints: protocol selection and differences between Qwen versions preclude treating it as a controlled scaling law or an intrinsic capacity limit.

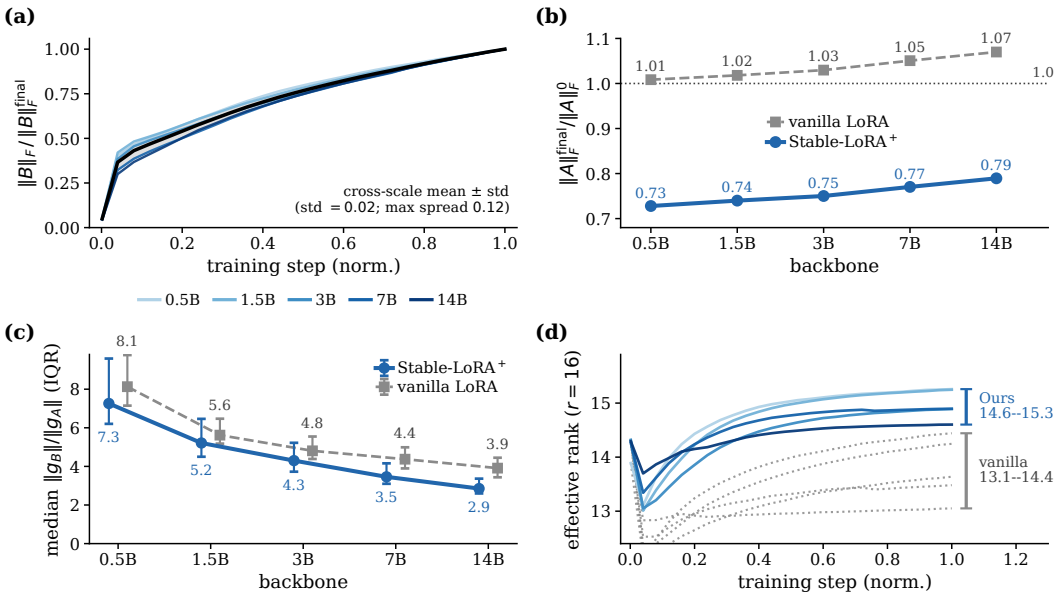


Figure 3: Normalized adapter dynamics across model scales. (a) Up-projection norm $\|B\|_F$ (per-step trace, normalized by its final value) for Stable-LoRA$^+$ at all five scales; the curves collapse across a $28\times$ parameter span (cross-scale mean $\pm$ std overlaid). (b) Realized A-side contraction $\|A\|_F^{\text{final}} / \|A\|_F^0$: a fixed γ yields contraction ratios from 0.73 (0.5B) to 0.79 (14B), while vanilla LoRA leaves $\|A\|$ unchanged. (c) Median steady-state gradient ratio $\|g_B\| / \|g_A\|$ (IQR bars) falls from 7.3 to 2.9 as the backbone grows (vanilla LoRA: 8.1 to 3.9), and the realized A-contraction tracks it. (d) Effective rank of the adapter trajectory (r=16): the recipe holds the update 0.8–1.6 ranks above vanilla LoRA at every scale.

The fixed-recipe runs assess shared hyperparameters through training dynamics (Figure 3). After normalization by the final value, $\|B\|_F$ follows similar trajectories across the five scales, with mean cross-scale standard deviation 0.02 and maximum spread 0.12. Because normalization removes absolute magnitude, this agreement describes relative growth rather than equality of the effective updates.

The same shrinkage coefficient yields final-to-initial A-norm ratios between 0.73 and 0.79, whereas vanilla LoRA and rsLoRA remain near 1.02 and 1.00, respectively. Together with the higher effective ranks, these observations support regularization of factorized adaptation as an interpretation of the fixed-recipe behavior. They do not isolate the causal contribution of each component; the following ablations address that question within the tested configurations.

5 Ablations and Analysis

5.1 Component, anchor, and rank ablations

Figure 4(a) shows the cumulative component ablation: at Qwen-1.5B, each stabilizer adds 0.27–0.54 pp on top of vanilla LoRA, ending at 61.94%. The largest increment is from adding the α-schedule warm-up ($+0.54$ pp); at Qwen-3B the evaluated configurations differ within a 0.68-pp window. Figure 4(b) sweeps the anchor's (k, λ) pair: the chosen $(64, 5\times10^{-4})$ lies in a 3×3 flat basin within 0.6 pp of optimal. Figure 4(c) sweeps the LoRA rank: vanilla LoRA peaks at r=16 and degrades at r=64 under α/r scaling; Stable-LoRA$^+$ adds an accuracy improvement of 0.6–1.5 pp across $r \in \{4, 8, 16, 32, 64\}$ — robustness to a sixteen-fold range of ranks.

5.2 Consolidated ablations: anchor, shrinkage variants, data scale

Table 4 consolidates three ablation families:

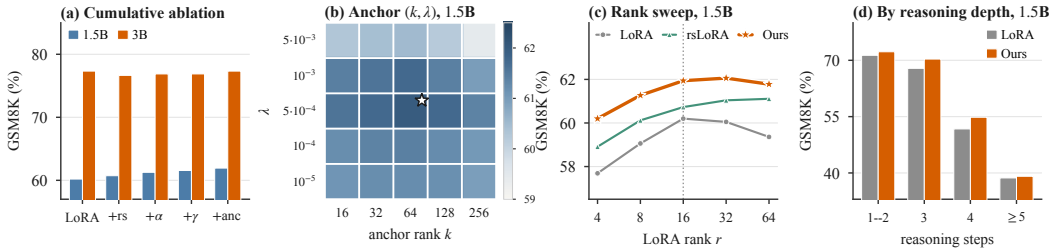


Figure 4: (a) Cumulative component ablation. Each of the four stabilizers contributes a measurable fraction of the $+1.74$ pp gain at 1.5B; the configurations differ by at most 0.68 pp at 3B. (b) Anchor (k, λ) sweep on Qwen-1.5B; the star marks our chosen $(64, 5{\cdot}10^{-4})$, lying in a $3{\times}3$ flat basin within 0.6 pp of the optimum. (c) Adapter rank sweep: vanilla LoRA peaks at $r{=}16$ and degrades at $r{=}64$ under α/r scaling; Stable-LoRA^{+} adds a uniform $+0.6$ to $+1.5$ pp lift across all five ranks. (d) Accuracy at 1.5B by reasoning depth (gold-solution operations): the gain peaks on medium-hard 3–4-step problems ($+2.5$/$+3.1$ pp) and is smaller in the lowest- and highest-operation bins.

(A) The 4-bit-aware anchor delivers $+1.36$ pp at 7B QLoRA (82.89 vs. 81.53, three seeds; anchor-on std 0.12 pp); the naive `nn.Linear` probe is indistinguishable from no anchor on `Linear4bit`, and at 14B all three settings converge ($\sim 84.9\%$, similar observed accuracy). **(B)** A spectral variant scaling shrinkage by per-layer conditioning gives no gain over constant γ (60.80 vs. 61.94) despite $\kappa(BA)$ ranging 1.9–9.1 (Figure 7b), and symmetric $\{A, B\}$ shrinkage destabilizes training (56.25); replacing the linear $32 \to 48$ schedule with any fixed $\alpha \in \{32, 48, 64\}$ drops 0.91–1.74 pp at 1.5B. **(C)** Tripling data and doubling epochs produce differences within the observed seed variation at 3/7B, while 2-epoch training degrades the smaller 1.5B adapter — a scale-dependent response to additional training.

Ablation axis	**GSM8K (%)**	
(A) Anchor on QLoRA (3-seed mean)	7B	14B
None	81.53	84.89
Naive (no-op on `Linear4bit`)	81.53	84.89
4-bit-aware (Ours)	**82.89**	**84.86**
Δ from no-anchor	$+\mathbf{1.36}$	-0.03
(B) A-side γ variants on 1.5B	(%)	
rsLoRA + AdamW (no γ-shrink)	61.03	
+ constant $\gamma{=}5{\times}10^{-4}$ (Ours)	**61.94**	
+ per-layer $\gamma_{\text{base}}\kappa^{0.5}$	60.80	
+ symmetric γ on $\{A, B\}$	56.25	
(C) Data and epoch axes	Baseline	Variant
Qwen-1.5B / 1 ep $\to$ 2 ep	61.03	59.89
Qwen-2.5-3B / 10K $\to$ 30K	76.88	77.03
Qwen-2.5-3B / 1 ep $\to$ 2 ep + anchor	76.88	77.10
Qwen-2.5-7B / 10K $\to$ 30K (QLoRA)	83.32	83.70

Table 4: Consolidated ablations. **(A)** 4-bit-aware anchor on QLoRA; **(B)** A-side γ variants; **(C)** data scaling together with multi-epoch training.

5.3 Where the gain concentrates

Binning the 1,319 GSM8K test problems by the number of calculator operations in their gold solution — a proxy for reasoning depth — localizes where the advantage arises (Figure 4d). At Qwen-1.5B the gain is not uniform: on the easiest problems (1–2 ops) and the hardest (≥ 5 ops) the gains are smaller ($+0.9$/$+0.4$ pp), whereas the advantage peaks in between, on medium-hard 3- and 4-operation problems ($+2.5$/$+3.1$ pp), indicating that the observed improvement is concentrated in problems of intermediate operation count.

5.4 Out-of-distribution transfer

We evaluate the unchanged canonical 10K checkpoints on MATH-500 (Hendrycks et al., 2021) and AQuA-RAT (Table 6) to assess transfer beyond GSM8K. From 0.5B to 3B, accuracy increases from 9.6% to 28.6% on MATH-500 and from 17.7% to 41.3% on AQuA-RAT. The lower absolute scores indicate substantial remaining difficulty on these tasks. These measurements establish performance of the transferred checkpoints, but without matched baseline evaluations they do not isolate the contribution of the stabilizers to out-of-distribution generalization.

Stable-LoRA$^+$ (3-seed)	3B	7B	14B
lr$=10^{-4}$ (Ours)	**76.90**	**82.89**	**84.86**
LoRA-optimal LR*	76.77	81.73	83.85
Δ favoring 10^{-4}	$+0.13$	$+1.16$	$+1.01$

*LR chosen per scale: 5×10^{-5} at 3/7B; 10^{-5} at 14B.

Table 5: Fixed lr$=10^{-4}$ vs. the per-scale LoRA-optimal LR for Stable-LoRA$^+$ (3-seed mean GSM8K): the fixed rate outperforms the tested alternative at all three scales.

Backbone	**MATH-500**	**AQuA-RAT**
Qwen-2-0.5B	9.60	17.72
Qwen-2-1.5B	18.00	27.17
Qwen-2.5-3B	**28.60**	**41.34**

Table 6: Out-of-distribution transfer of the canonical 10K Stable-LoRA$^+$ checkpoints (%, single-greedy); accuracy scales with backbone size, mirroring GSM8K (Table 1).

6 Discussion

Interpreting performance across scales. The best-protocol aggregate (70.14%, mean rank 1.50) summarizes selected checkpoints, while the fixed-recipe runs assess training stability and accuracy across scales. These views should not be conflated: selection can improve aggregate accuracy without establishing a uniform configuration advantage. At 1.5B, the recipe uses the same 4.36M trainable parameters as vanilla LoRA (Appendix G), so additional adapter capacity does not explain the observed gain. Its concentration at intermediate operation counts motivates further study of how update geometry affects reasoning performance.

Why factorization matters. The anchor depends on BA and is invariant to the reciprocal rescaling $A \mapsto cA$, $B \mapsto B/c$ for $c \neq 0$ (Equation 2). The norm-based shrinkage gate need not be invariant to that transformation. Thus, equivalent weight updates can receive different factor-space regularization during training. This distinction explains why an output penalty alone does not specify the full training dynamics. The ablations assess the empirical value of the proposed regularization, while its interaction with the optimizer warrants further analysis. With orthonormal U_k and V_k, each unweighted anchor term is the squared Frobenius norm of the projected block $U_k U_k^\top BAV_k V_k^\top$. It therefore penalizes a specific component of the update rather than its entire norm, leaving directions outside that block unpenalized by the anchor.

Implications for adaptation. The recipe preserves LoRA's trainable-parameter budget and projection targets. Its fixed settings provide a reproducible starting point for the evaluated Qwen backbones, with the 0.5B subset result demonstrating that stronger regularization is not uniformly beneficial. Robust scaling should therefore be assessed jointly with accuracy, implementation compatibility, and the cost of selecting a configuration.

7 Limitations and Conclusion

Limitations. Most baselines lack repeated-seed evaluations, so the observed variability of our recipe does not establish uncertainty in pairwise differences. Best-protocol selection further limits interpretation of the aggregate ranking. Only vanilla LoRA receives a full learning-rate sweep, and the out-of-distribution evaluation lacks matched baselines. Model size, Qwen version, and numerical precision also vary jointly; their effects are not fully separable. The results support an empirical design hypothesis over 0.5B–14B, rather than a guarantee of transfer to other model families, training budgets, or tasks.

Conclusion. Stable-LoRA$^+$ investigates regularization of factorized updates as an approach to robust scaling of low-rank adaptation. Across the evaluated Qwen backbones, the strict fixed recipe matches vanilla LoRA's macro-average accuracy; the broader best-protocol comparison achieves mean rank 1.50. The observed cross-scale dynamics and component ablations support the design, while the quantization audit shows why a mathematical constraint must also be verified in its implementation. Together, these results motivate evaluating adaptation by both predictive accuracy and the search required to obtain reliable behavior on a new backbone.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript drafting and revision, mathematical exposition, checks of consistency between claims and reported results, and code for figure preparation and LaTeX layout. These activities included suggestions for interpreting the reported training dynamics and for identifying limitations of the evidence. AI-assisted edits were checked against the manuscript's tables and recorded data-provenance documentation, and the resulting document was compiled and visually inspected. The authors are responsible for the final text, claims, and artifacts, including all AI-assisted content.

ETHICS STATEMENT

This study evaluates parameter-efficient adaptation using existing language models and mathematical-reasoning datasets. Its empirical scope is benchmark accuracy and training behavior; it does not establish deployment safety, fairness, or reliability in consequential settings. Adapted models may retain biases from their training data and produce plausible but incorrect reasoning. Improved benchmark performance should therefore not substitute for application-specific validation and appropriate human oversight. Reuse or redistribution of the models, data, and derived artifacts should respect the applicable licenses and access conditions. To support research integrity, we distinguish fixed-configuration results from selected-protocol results and report implementation failures and uncertainty limitations.

REPRODUCIBILITY STATEMENT

The method is specified in Algorithms 1 and 2, with the anchor objective in Equation 2. Appendix A describes the software environment, data selection, training configuration, and evaluation procedure; Appendix E provides implementation details, and Appendix D records baseline compatibility issues and fixes. Appendices C and F describe the recorded configurations, checkpoints, and predictions, including deviations from the canonical protocol. Repeated-seed results appear in Table 3, and hardware and training times in Appendix H. These references document the reported experiments and distinguish fixed-configuration measurements from best-protocol selection.

REFERENCES

Klaudia Bałazy, Mohammadreza Banaei, Karl Aberer, and Jacek Tabor. Lora-xs: Low-rank adaptation with extremely small number of parameters. In *ECAI 2025: 28th European Conference on Artificial Intelligence 25-30 October 2025, Bologna, Italy-Including 14th Conference on Prestigious Applications of Intelligent Systems (PAIS 2025) Proceedings*, pp. 3194–3201. SAGE Publications 1 Oliver's Yard, 55 City Road, London, EC1Y 1SP, 2024.

Kerim Büyükakyüz. Olora: Orthonormal low-rank adaptation of large language models. *arXiv preprint arXiv:2406.01775*, 2024.

Yupeng Chang, Chenlu Guo, Yi Chang, and Yuan Wu. Lora-mgpo: Mitigating double descent in low-rank adaptation via momentum-guided perturbation optimization. In *EMNLP (Findings)*, pp. 648–659, 2025.

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

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. volume 36, pp. 10088–10115, 2023.

Aymane El Firdoussi, El Mahdi Chayti, Mohamed El Amine Seddik, and Martin Jaggi. α-lora: Effective fine-tuning via base model rescaling. 2025. URL https://arxiv.org/abs/2510.21345.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, et al. Mora: High-rank updating for parameter-efficient fine-tuning. *arXiv preprint arXiv:2405.12130*, 2024.

Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with lora, 2023.

Dawid Kopiczko, Tijmen Blankevoort, and Yuki Asano. Vera: Vector-based random matrix adaptation. In *International Conference on Learning Representations*, volume 2024, pp. 6815–6835, 2024.

Yu-Ang Lee, Ching-Yun Ko, Pin-Yu Chen, and Mi-Yen Yeh. Learning rate matters: Vanilla lora may suffice for llm fine-tuning. *arXiv preprint arXiv:2602.04998*, 2026.

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

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

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jian-Guang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, et al. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. In *International Conference on Learning Representations*, volume 2025, pp. 49573–49609, 2025.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. volume 37, pp. 121038–121072, 2024.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Pengjie Ren, Chengshun Shi, Shiguang Wu, Mengqi Zhang, Zhaochun Ren, Maarten de Rijke, Zhumin Chen, and Jiahuan Pei. Melora: Mini-ensemble low-rank adapters for parameter-efficient fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3052–3064, 2024.

Ayush Singh, Rajdeep Aher, and Shivank Garg. Lora-mini: Adaptation matrices decomposition and selective training. *arXiv preprint arXiv:2411.15804*, 2024.

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

Zhengbo Wang, Jian Liang, Ran He, Zilei Wang, and Tieniu Tan. Lora-pro: Are low-rank adapters properly optimized? In *International Conference on Learning Representations*, volume 2025, pp. 93787–93808, 2025.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Yize Wu, Ke Gao, Ling Li, and Yanjun Wu. Stable-lora: Stabilizing feature learning of low-rank adaptation. *arXiv preprint arXiv:2603.05204*, 2026.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024. URL https://arxiv.org/abs/2407.10671.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 45040–45061, 2024.

Xin Yu, Yujia Wang, Jinghui Chen, and Lingzhou Xue. Altlora: Towards better gradient approximation in low-rank adaptation with alternating projections. *Advances in Neural Information Processing Systems*, 38:37484–37522, 2026.

Jingfan Zhang, Yi Zhao, Dan Chen, Xing Tian, Huanran Zheng, and Wei Zhu. Milora: Efficient mixture of low-rank adaptation for large language models fine-tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 17071–17084, 2024.

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

Zikai Zhou, Qizheng Zhang, Hermann Kumbong, and Kunle Olukotun. Lowra: Accurate and efficient lora fine-tuning of llms under 2 bits. *arXiv preprint arXiv:2502.08141*, 2025.

SUPPLEMENTARY MATERIAL — APPENDIX

A ADDITIONAL EXPERIMENTAL DETAILS

Software stack. PyTorch 2.5.1+cu124, PEFT 0.14.0, transformers 4.46.3, bitsandbytes 0.49.2. All runs share a fixed Python environment and a common set of training, model, data, and metric utilities. Each run's entry point is frozen on disk at run time, and each run writes a self-contained artifact directory holding the saved adapter, the full configuration, a per-step metric log, the per-question predictions, and the final aggregate accuracy.

Hyperparameter recipe. Across all five backbones we use a single hyperparameter set: rank r=16, $\alpha_{\min}$=32, $\alpha_{\max}$=48, warm-up fraction 0.30, γ=5×10^{-4}, λ=5×10^{-4}, k=64, default learning rate 10^{-4}, AdamW with β_1=0.9, β_2=0.999, weight decay 0.01, 1 epoch on 10,000 MetaMathQA samples, max sequence length 512. We target the $\{q, k, v, o\}$-projections of every transformer block. Single RTX-3090 (24 GB) per run, effective batch size 16 (microbatch 1–2 with gradient accumulation), gradient checkpointing for $\geq$ 3B, NF4 QLoRA for $\geq$ 7B.

Prompt and evaluation protocol. Both training and evaluation wrap each question in the Alpaca-style template `"Below is an instruction...\n### Instruction:\n{q}\n\n### Response:"`. At evaluation we greedily decode up to 256 new tokens (left-padded batches, `bf16`; NF4 for $\geq$ 7B), extract the gold answer with the GSM8K regex `####\s*(-?\d+(?:\.\d+)?)`, and compare by numeric exact-match. Accuracy is reported as the count correct/1319 over the full GSM8K test set.

Cross-method learning-rate behavior. We swept all five learning rates only for vanilla LoRA (Figure 2d), which is the method whose LR fragility is under debate; a full cross-method 5×5 sweep at every scale was beyond our budget. For Stable-LoRA$^+$ we instead report, in the main text, the 3-seed comparison of its fixed lr=10^{-4} against each scale's *LoRA-optimal* LR (§Multi-seed): 10^{-4} remains best for our recipe at the three tested scales; this comparison does not replace a full sweep.

Per-method training-loss trajectories. Figure 5 plots the recorded per-step training loss on Qwen-1.5B for seven methods. The four convergent methods converge to a near-identical ≈ 0.165, yet this low-loss cluster still spans 60–62% accuracy while PiSSA reaches the same loss regime at 24%: training loss is a poor predictor of test accuracy, the cross-method counterpart of the optimization-vs-generalization split in Appendix I.

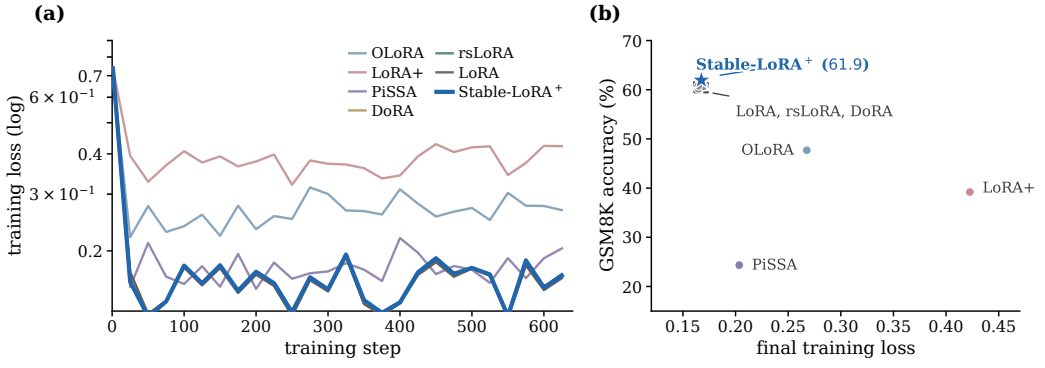


Figure 5: Recorded training loss on Qwen-1.5B (the per-step training logs), seven methods. (a) The convergent methods (LoRA, rsLoRA, DoRA, Stable-LoRA$^+$) converge to a near-identical ≈ 0.165; OLoRA and LoRA+ plateau higher. (b) Final training loss vs. GSM8K accuracy: the ≈ 0.165 cluster spans 60–62%, so terminal training loss alone does not order these checkpoints by test accuracy.

B Compute–Accuracy Pareto

Figure 6 compares recorded training time and accuracy across backbones. The stabilizers preserve the number of trainable parameters, but introduce scaling, shrinkage, and anchor computations. The anchor additionally requires an initial SVD (under 90 s at 14B in the recorded implementation). The measured runtimes remain comparable to LoRA; these observations should not be interpreted as zero additional FLOPs. DoRA requires 10–20% more wall time in the reported comparison.

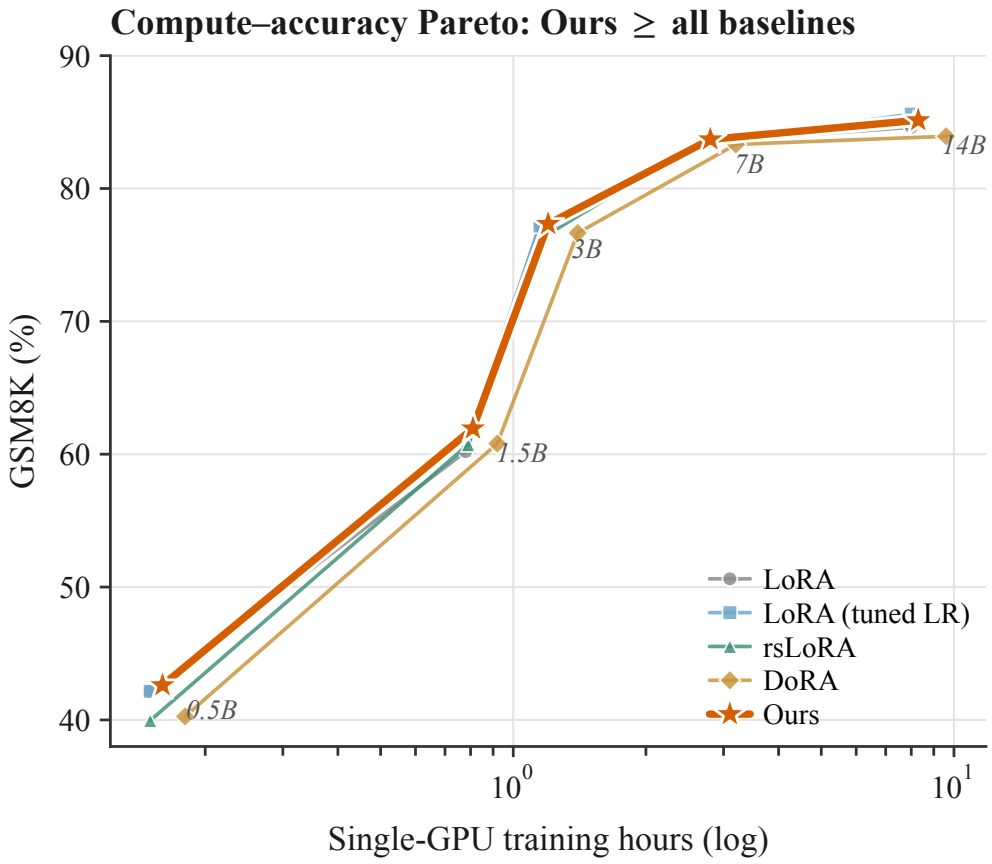


Figure 6: Single-GPU RTX-3090 training hours versus final GSM8K accuracy across 5 Qwen backbones. The plotted checkpoints summarize the measured compute–accuracy trade-off under the reported protocols.

C Reproducibility

We record random seeds and configurations to support reproducibility; seed variation is reported separately in Table 3. Each run records a complete configuration, per-step JSON metrics, per-question predictions, and a saved PEFT adapter checkpoint; total disk footprint $\sim$ 38 GB across all runs. Canonical (10K-data, 1-epoch) Stable-LoRA$^+$ *training* time on a single RTX-3090, read from the run metadata, is $\sim$ 10 min (0.5B, 587 s), $\sim$ 49 min (1.5B, 2915 s), $\sim$ 1.2 h (3B, 3.8–4.8 ks), $\sim$ 2.8 h (7B QLoRA-NF4, 10.1 ks), and $\sim$ 8.3 h (14B QLoRA-NF4, 29.8 ks); single-greedy GSM8K evaluation of the full 1,319-problem test set adds roughly 0.7–1.0$\times$ the training time at the small scales.

Multi-seed coverage summary. 3-seed measurements are available for Stable-LoRA$^+$ at every scale (Table 3) and for rsLoRA + StableLora at 7B QLoRA (82.94 ± 0.28). Single-seed measurements are used for the remaining baselines, limiting uncertainty estimates for pairwise comparisons.

Initialization Frobenius norms and per-layer conditioning. Figure 7 corroborates Tables 11 and 4(B): PiSSA / OLoRA initial $\|sBA\|_F$ rivals the row norms of $W^{(0)}$, and the per-layer adapter conditioning $\kappa(BA)=\sigma_1/\sigma_{16}$ of the converged Stable-LoRA$^+$ adapter spans 1.9–9.1 across the 28 blocks and four projections (median 2.7), concentrated in the v-projections.

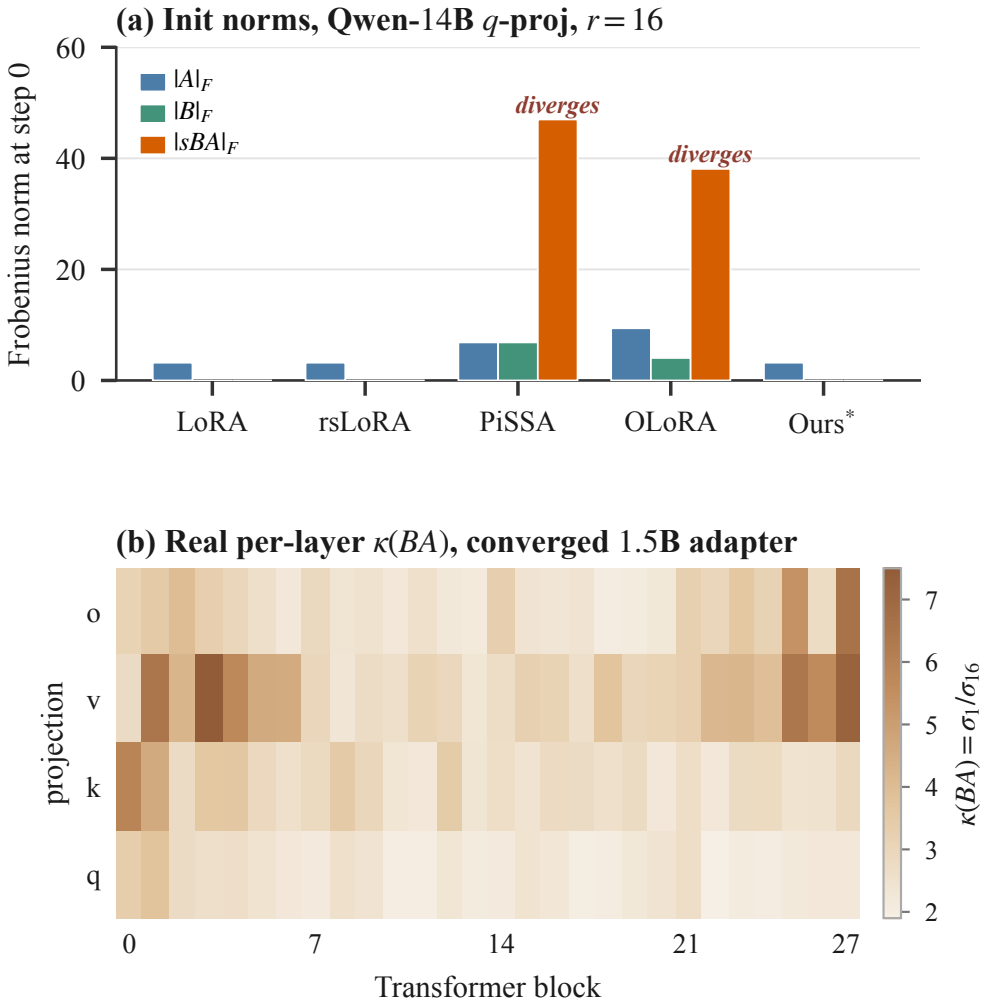


Figure 7: (a) Initial Frobenius norms of A, B, and the effective adapter update sBA at Qwen-14B, r=16. PiSSA's and OLoRA's $\|sBA\|_F$ at step 0 rivals the row norms of $W^{(0)}$. (b) Measured per-layer adapter conditioning $\kappa(BA)=\sigma_1/\sigma_{16}$ of the converged Qwen-1.5B adapter across the 28 blocks and four projections (from the saved adapter): κ spans 1.9–9.1 (median 2.7, highest in the v-projections), yet scaling shrinkage by it does not improve on constant γ (Table 4B).

D Failure-Mode Catalogue

Several baselines do not run as released in the PEFT version used. **(a)** *AdaLoRA* (Zhang et al., 2023) fails at runtime with `Parameter * NoneType` inside `update_ipt`, because the stock training loop calls `update_and_allocate()` *after* `optimizer.zero_grad()`, so the importance update reads already-cleared (`None`) gradients. Re-ordering to `backward` → `step` → `update_and_allocate` → `zero_grad` (a one-line fix in a standalone script, not a PEFT modification) makes AdaLoRA train normally; the Table 1 AdaLoRA row (40.79/61.18/76.72/82.03 at 0.5–7B) uses this corrected loop, and the selected Stable-LoRA$^+$ checkpoints match or exceed it. **(b)** *LoRA-GA* (Wang et al., 2024) is rejected by `LoraConfig(init_lora_weights="lora_ga")` in the PEFT version we used; we re-implement its gradient-SVD initialization in a standalone script (SVD

of the first-batch full gradient initializes A, B so the first update aligns with the full-FT direction). It then trains at all five scales but has lower reported accuracy than the selected Stable-LoRA$^+$ checkpoints under the evaluated settings (Table 1). **(c)** ***PiSSA*** (Meng et al., 2024) destabilizes on $\leq$ 3B once composed with rsLoRA's amplified scaling (Table 11) and is incompatible with the QLoRA-NF4 path used on $\geq$ 7B due to a TypeError in the PiSSA initialization helper. **(d)** ***LoRA+*** (Hayou et al., 2024) at the recommended η_B/η_A=16 ratio diverges at $\geq$ 3B (Table 1) and is incompatible with our QLoRA-NF4 path on $\geq$ 7B. These implementation-specific failures are reported separately from accuracy comparisons; they do not establish intrinsic method limitations.

E Algorithmic Details

The training procedure uses standard PEFT and AdamW operations without custom CUDA kernels or modified attention. At initialization, we compute the base weight's leading singular subspaces. During training, the anchor penalty enters the objective, the optimizer applies gated shrinkage, and the adapter scaling follows its prescribed schedule. Algorithm 2 specifies the procedure; the following paragraphs provide implementation details for reproduction.

Algorithm 2 Stable-LoRA$^+$ step

Pre-compute (startup, per LoRA layer ℓ):
  $W_\ell \leftarrow \textsc{Dequant}_{\text{4bit}}(W_\ell^{(0)}) + \texttt{get_delta_weight}()$
  $U_{k,\ell}, V_{k,\ell} \leftarrow \textsc{svd_lowrank}(W_\ell, q{=}k, \text{niter}{=}4)$
for step $t = 1 \dots T$ **do**
  $\hat{y} \leftarrow \textsc{Forward}(x;\, W^{(0)} {+} s\, BA)$; $\mathcal{L}_{\text{task}} \leftarrow \text{NLL}(\hat{y}, y)$
  $\mathcal{L}_{\text{anc}} \leftarrow \frac{\lambda}{kr} \sum_\ell \|U_{k,\ell}^\top B_\ell A_\ell V_{k,\ell}\|_F^2$
  $(\mathcal{L}_{\text{task}} {+} \mathcal{L}_{\text{anc}})\texttt{.backward()}$
  $\textsc{StableLoRA.step}()$ {AdamW + gated A-shrink}
  $\alpha(t) \leftarrow \alpha_{\min} {+} (\alpha_{\max} {-} \alpha_{\min}) \min(1, \frac{t}{T_{\text{warm}}})$; $s \leftarrow \alpha(t)/\sqrt{r}$
return adapter (A, B)

(1) rsLoRA scaling. Standard LoRA forward is $h = W^{(0)}x + (\alpha/r)\, BAx$, whose effective update collapses as r grows. We build the adapter with rank-stabilized scaling enabled, replacing α/r with $\alpha/\sqrt{r}$. At r=16, α=48 the operating scale is s=48/$\sqrt{16}$=12, a 4$\times$ amplification over the α/r=3 default. This higher operating scale motivates gradual warm-up and regularization of the factorized update.

(2) α-schedule. Rather than fix α, we ramp it linearly from $\alpha_{\min}$=32 to $\alpha_{\max}$=48 over the first T_{warm}=0.30 T steps, re-deriving the operating scale s=$\alpha(t)/\sqrt{r}$ after every optimizer step so the effective amplification grows smoothly rather than jumping to its final value. This reduces early amplification while retaining the final operating scale, and yields the largest incremental ablation gain at 1.5B (+0.54 pp; Figure 4a).

(3) A-side γ-shrinkage (gated). Wrapping AdamW, the optimizer applies a multiplicative shrinkage $A \leftarrow (1{-}\gamma)A$ to the down-projection only (the up-projection B is untouched) before each AdamW update, following the feature-stability argument of Wu et al. (2026). The shrinkage is *gated* per layer: once A's norm falls below $(|A|/|B|)\, \|B\|$ — i.e. A no longer dominates the adapter — shrinkage is disabled for that layer, preventing late-training collapse. With γ=5$\times$10^{-4}, the recorded 600-step run retains a moderately conditioned adapter (the converged per-layer $\kappa(BA)$ spans 1.9–9.1; Figure 7b).

Mechanistic signature. The per-step logs make the stabilizers' effect directly visible (Figure 8, from the per-step training logs on Qwen-1.5B, same data and seed as vanilla LoRA). The A-side γ-shrinkage drives $\|A\|_F$ *down* by 26% while vanilla LoRA lets it drift *up* by 2%, and the effective rank of BA climbs to 15.25 (of the rank-16 ceiling) versus a full point lower (14.25) for vanilla LoRA. These

concurrent changes are consistent with a geometric effect of shrinkage, but do not establish that noise reduction causes the rank increase.

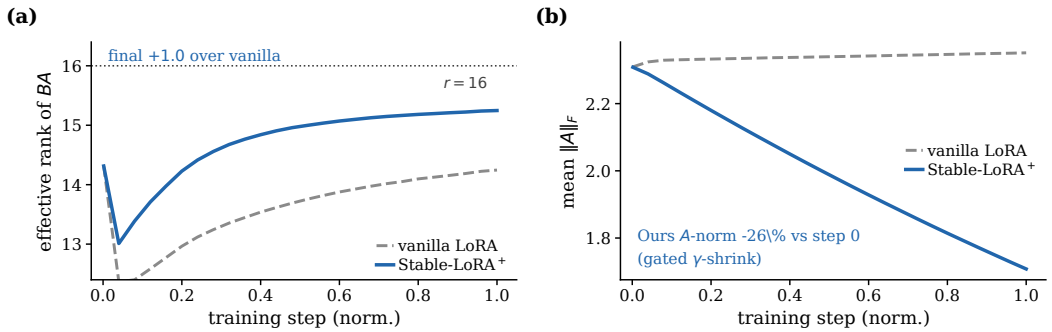


Figure 8: Training dynamics on Qwen-1.5B (recorded training logs). (a) Effective rank of BA: Stable-LoRA$^+$ climbs toward the rank-16 ceiling ($+1.0$ over vanilla LoRA), using more of the adapter budget. (b) Mean $\|A\|_F$: our gated γ-shrinkage cuts the down-projection norm by 26% while vanilla LoRA's increases by 2%.

This rank-budget advantage is not specific to 1.5B: across all five scales the gated shrinkage holds Stable-LoRA$^+$'s converged effective rank near 15 of 16 while vanilla LoRA settles at 13–14.4, and the margin *widens* from 0.8 at 0.5B to 1.6 at 14B (Figure 9) — indicating a larger effective-rank difference at the evaluated larger scales.

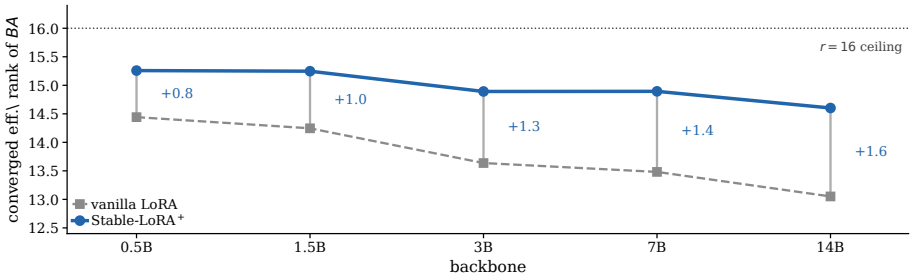


Figure 9: Converged effective rank of BA (of the rank-16 ceiling) for Stable-LoRA$^+$ and vanilla LoRA at every scale. Gated A-shrinkage keeps more of the budget usable throughout, and the margin grows with scale, alongside the changing gradient-norm ratio (Figure 11).

The gradient norms are asymmetric (Figure 10). At initialization, $B = 0$ implies a zero task-loss gradient for A, whereas the gradient for B is nonzero (4.36 in the recorded first step). Subsequently, B has a gradient norm 4–11 times that of A. This motivates examining factor-specific regularization, but gradient magnitude alone does not determine the signal-to-noise ratio. The asymmetric shrinkage ablation provides the relevant empirical comparison.

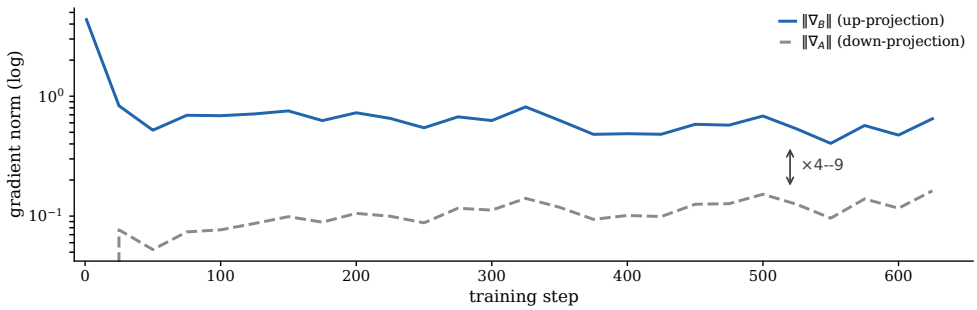


Figure 10: Per-step gradient norms of the two LoRA factors on Qwen-1.5B (log scale, from the per-step training logs). The up-projection B carries 4–11$\times$ the gradient norm of the down-projection A after initialization. These norms characterize factor asymmetry, not gradient noise directly.

The steady-state median $\|\nabla_B\|/\|\nabla_A\|$ decreases from 7.3 at 0.5B to 2.9 at 14B (Figure 11). The observed A-norm contraction is also weaker at larger scales under the same γ. Their covariation is consistent with scale-dependent factor dynamics, although it does not identify the gradient ratio as the cause of the change in realized shrinkage.

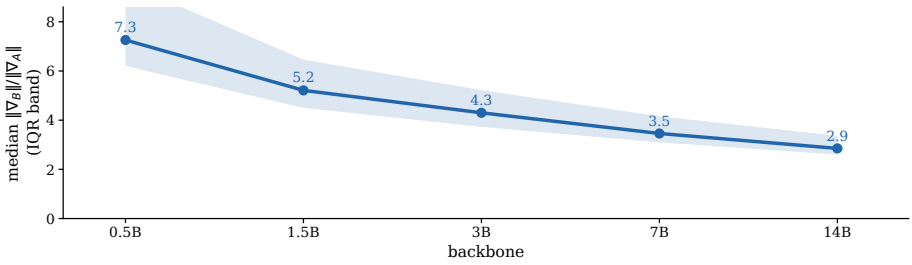


Figure 11: Steady-state ratio of up- to down-projection gradient norms $\|\nabla_B\|/\|\nabla_A\|$ (median and interquartile band) for the fixed Stable-LoRA$^+$ recipe at every scale. The B-side dominance that motivates one-sided A-shrinkage weakens from 7.3 (0.5B) to 2.9 (14B), accompanying the variation in realized shrinkage across scales (Figure 3b).

(4) Corrected 4-bit-aware subspace anchor. At startup, for each LoRA layer we recover the full base weight $W^{(0)}$ and take a randomized top-k SVD, storing both left (U_k) and right (V_k) singular vectors. The per-step penalty is bilinear (Algorithm 2, line 6) — it suppresses the component of the raw delta BA that lies in the top-k left *and* right singular subspaces of $W^{(0)}$, normalized by kr. The fix for quantized models lies in recovering $W^{(0)}$ correctly: a `Linear4bit` base layer stores its weight as an opaque NF4 tensor rather than the $d_{\text{out}} \times d_{\text{in}}$ matrix the SVD requires, so the anchor must explicitly detect the quantized module and dequantize before factorizing. The evaluated layer-selection path fails to apply the anchor to these modules; detecting and dequantizing recovers $+1.36$ pp at 7B QLoRA (Table 4A).

F Complete Per-Run Results

Table 7 lists every Stable-LoRA$^+$ run referenced in the paper with its configuration delta and GSM8K accuracy, so each headline number is traceable to a single saved evaluation record. All accuracies are the count $\text{correct}/1319$ under single-greedy decoding on the full test set.

Reproduced baselines. The two baselines that do not run as released (§D) were restored with standalone drivers that reuse the shared data, metric, and loop utilities without modifying any existing file. AdaLoRA is restored by re-ordering its importance update before the gradient reset, reaching 538/807/1012/1082/1123 correct out of 1,319 across the five scales. LoRA-GA is restored by re-implementing its gradient-SVD initialization (one full-batch backward, then a top-r SVD of each target-weight gradient seeds A, B), reaching 505/786/1008/1082/1112 correct. Both converge under the corrected implementation. Their reported accuracies match or fall below the selected Stable-LoRA$^+$ checkpoints; these comparisons do not establish that the methods are generally inferior.

Configuration	Seed	GSM8K
Qwen-2-0.5B (bf16, anchor active)		
full recipe	42	41.02
full recipe	43	38.82
full recipe	44	39.35
rsLoRA+shrink subset	42	**42.61**
Qwen-2-1.5B (bf16, anchor active)		
full recipe	42	61.94
full recipe	43	58.91
full recipe	44	60.27
Qwen-2.5-3B (bf16, anchor active)		
full recipe	42	76.88
full recipe	43	76.50
full recipe	44	77.33
Qwen-2.5-7B (QLoRA-NF4)		
full recipe, anchor on	42	82.87
full recipe, anchor on	43	83.02
full recipe, anchor on	44	82.79
anchor off	42	81.65
anchor off	43	81.73
anchor off	44	81.20
30K MetaMathQA	42	**83.70**
Qwen-2.5-14B (QLoRA-NF4)		
full recipe, anchor on	42	85.06
full recipe, anchor on	43	84.38
full recipe, anchor on	44	85.14
anchor off	42	84.99
anchor off	43	84.69
anchor off	44	84.99

Table 7: Complete Stable-LoRA$^+$ run inventory (bold = the per-scale cell reported in Table 1). The 7B/14B anchor-on/off triplets give the $+1.36$ pp (7B) and -0.03 pp (14B) anchor deltas of Table 4(A). The two headline cells that use a recipe variant are itemized here: the 0.5B cell is the rsLoRA+shrinkage subset (since the α-schedule and anchor are detrimental on a 0.5B adapter), and the 7B cell uses 30K MetaMathQA; the canonical seed-42 10K full-recipe values are 41.02 (0.5B) and 82.87 (7B), giving a strict single-config macro-average of 69.55% (and 68.95% averaged over three seeds), versus the 70.14% best-protocol headline.

Determinism and artifacts. Every run is seeded and writes a self-contained artifact directory under `results/` (adapter, configuration plus wall-time, per-step metrics, per-question predictions, and the aggregate accuracy). Because the GSM8K parser and single-greedy decoder are deterministic, re-evaluating any saved adapter reproduces its table entry exactly, and the saved per-question prediction records let a reader audit predictions or recompute subset accuracies (e.g. by operation count) without re-training. The shared pipeline controls data processing and evaluation, while configuration choices and seed selection remain potential sources of variation.

Additional configuration comparisons. Table 8 summarizes evaluated variants that did not improve on the selected configuration. These include factor-wise learning-rate changes, alternative initializations, projection targets, ranks, and training schedules. They document the explored design space without establishing that the selected four-component recipe is minimal or globally optimal.

Evaluated variant	Scale	GSM8K	Design axis
LoRA+ (η_B/η_A=16)	1.5B	39.20	Factor learning rates
LoRA+ (η_B/η_A=4)	1.5B	58.76	Factor learning rates
PiSSA + rsLoRA	1.5B	56.25	Initialization
VeRA	0.5B	28.96	Shared factors
all-linear targets	0.5B	39.27	Projection targets
fixed α=64	1.5B	60.20	Update scaling
cosine peak 3×10^{-4}	1.5B	57.39	LR schedule
rank r=64	1.5B	57.92	Adapter rank
2 epochs	1.5B	59.89	Training duration

Table 8: Representative ablations that did not improve on the chosen Stable-LoRA$^+$ recipe (recorded GSM8K %, single-greedy). These are selected comparisons from the explored design space, not an exhaustive test of component necessity.

G Per-Question Complementarity and Scaling

Aggregate accuracy hides *which* problems each method solves. Using the per-question correctness flags, we cross-tabulate Stable-LoRA$^+$ against vanilla LoRA at every backbone. Figure 12(a) shows the 2×2 agreement matrix at Qwen-1.5B: the two methods agree on 1,208 of 1,319 problems (750 both-correct, 458 both-wrong), but among the disagreements Stable-LoRA$^+$ solves 67 problems that vanilla LoRA misses versus only 44 the other way — a difference of $23/1319$, or approximately 1.74 pp. This characterizes the selected checkpoint pair and does not exclude seed effects. Figure 12(b) extends this to all five scales: Stable-LoRA$^+$ uniquely solves 34–74 problems the baseline cannot at every scale, with the unique-solve balance tipping in our favour at 1.5B and 14B and remaining substantial even where the macro-accuracies tie. The persistent off-diagonal mass also defines potential headroom for a selector combining the two models — a direction we leave to future work.

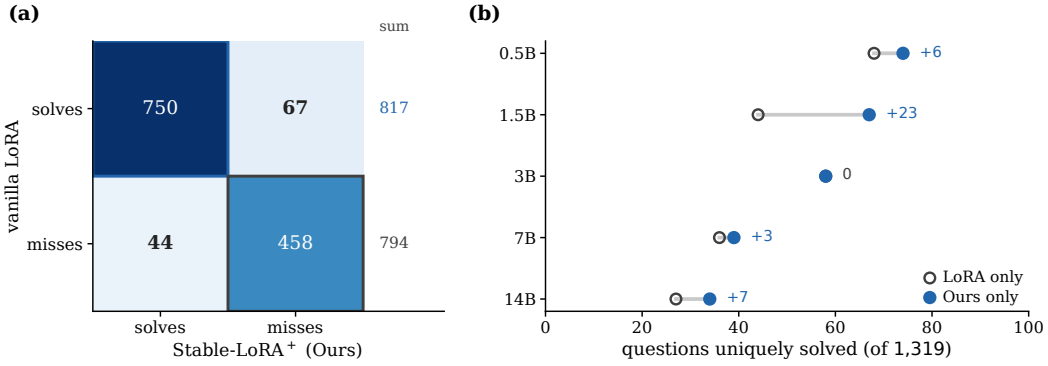


Figure 12: Per-question agreement between Stable-LoRA$^+$ and vanilla LoRA, from the saved per-question predictions. (a) The 1.5B 2×2 contingency: Ours uniquely solves 67 problems vs LoRA's 44. (b) Unique-solve counts at all five scales; both methods retain a large distinct set, so their macro-accuracies understate the per-problem differences.

A per-difficulty view of the same records (Figure 4d, main text) shows the edge is largest on medium-hard 3–4-step problems (+2.5/+3.1 pp) and smaller in the lowest- and highest-operation bins, for these evaluated checkpoints.

The stabilizers introduce no additional trainable parameters: the α-schedule and γ-shrinkage only rescale existing tensors and the anchor is a loss term, so the adapter has exactly the trainable footprint of standard LoRA. Table 9 confirms this at Qwen-1.5B — Stable-LoRA$^+$, vanilla LoRA, rsLoRA, and LoRA-GA all train 4.36M parameters (0.28%), whereas DoRA and AdaLoRA use 2–50% more trainable parameters with lower reported accuracy. The gains thus come from *how* the fixed budget is optimized, not from spending more capacity.

Method	Trainable	% of base	1.5B acc.
vanilla LoRA	4.36M	0.282	60.20
rsLoRA	4.36M	0.282	60.73
LoRA-GA	4.36M	0.282	59.59
DoRA	4.46M	0.288	60.80
AdaLoRA	6.54M	0.422	61.18
Stable-LoRA$^+$	**4.36M**	**0.282**	**61.94**

Table 9: Trainable-parameter budget at Qwen-1.5B (from each run's startup log). Stable-LoRA$^+$ matches the smallest budget (identical to vanilla LoRA) yet reaches the highest accuracy; DoRA and AdaLoRA spend more trainable parameters in this comparison.

Figure 13 plots raw GSM8K accuracy of all nine single-LR methods against backbone size (log axis), extending the best-protocol comparison in Figure 2(a) to the full method set: Stable-LoRA$^+$ sits at or above the dense baseline cluster at every scale, OLoRA trails by 10–30 pp, and the cluster flattens after 7B — diminishing returns observed in the main text.

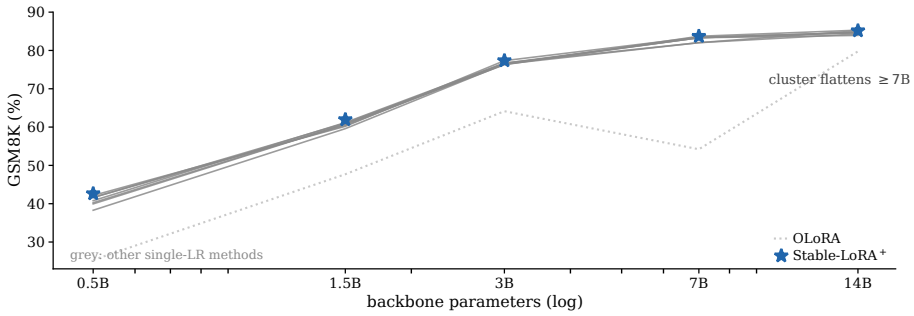


Figure 13: GSM8K accuracy versus backbone parameters (log scale) for all nine single-LR methods. Stable-LoRA$^+$ (blue stars) sits at the top of the cluster at every scale; OLoRA trails throughout and the cluster flattens after 7B, consistent with the observed diminishing returns.

The complementarity in Figure 12 also bounds what a oracle selector between Stable-LoRA$^+$ and LoRA could achieve. The per-question union of the two methods' correct sets — an oracle picking whichever is right — yields 47.8/65.3/81.7/86.4/87.2% across the five scales (Figure 14), each 2–4 pp above the better individual method; the reported average headroom is 3.4 pp. These disagreements alone do not establish robustness across seeds. We report these unions purely as a diagnostic upper bound — all headline numbers are single-model, single-greedy, without per-question selection or ensembling.

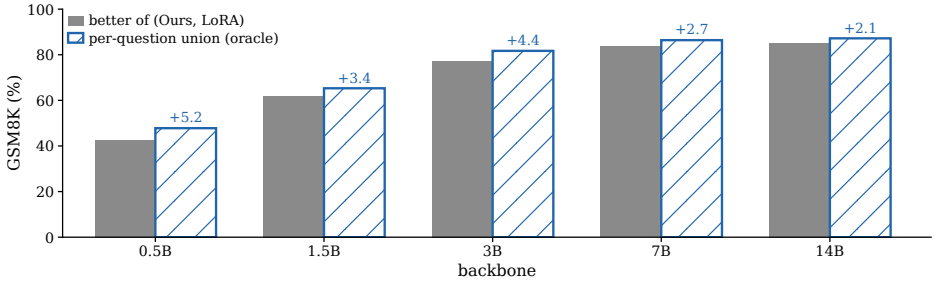


Figure 14: The per-question *union* of Stable-LoRA$^+$ and vanilla LoRA — an oracle picking whichever model is right — sits 2–5 pp above the better individual model at every scale (mean +3.4 pp), quantifying complementary predictions for the selected checkpoint pairs. Reported only as a diagnostic upper bound; the headline numbers use no per-question selection or ensembling.

Reproduction. The canonical full-recipe runs share a training-and-evaluation pipeline and hyperparameter set — rank r=16, α ramped 32→48 over the first 30% of steps, anchor strength λ=5×10^{-4} with subspace size k=64, and learning rate 10^{-4} — with 4-bit quantization enabled at $\geq$ 7B and only the backbone identifier changed between runs. Evaluation re-runs each saved adapter under

single-greedy decoding, so the saved predictions support reconstruction of the reported accuracies; best-protocol exceptions are listed in Table 7.

H Compute Footprint

Table 10 reports the adapter size and single-GPU training time at every scale (from each run's startup log and the run metadata). The trainable footprint stays between 0.13% and 0.44% of the backbone across the evaluated scales. The stabilizers preserve the trainable-parameter budget but require additional operations and anchor storage. Each run fits on a 24 GB RTX-3090; the reported times describe this hardware and implementation rather than a hardware-independent cost guarantee.

Backbone	Precision	Trainable	% base	Train time
Qwen-2-0.5B	bf16	2.16M	0.436	9.8 min
Qwen-2-1.5B	bf16	4.36M	0.282	48.6 min
Qwen-2.5-3B	bf16 + GC	7.37M	0.238	1.06 h
Qwen-2.5-7B	NF4 QLoRA	10.09M	0.132	2.80 h
Qwen-2.5-14B	NF4 QLoRA	25.17M	0.170	8.27 h

Table 10: Stable-LoRA$^+$ compute footprint on a single RTX-3090 (24 GB); parameters from each run's startup log, time from the run metadata (GC = gradient checkpointing).

I Optimization vs. Generalization

Training loss provides a complementary view of the accuracy differences. On Qwen-1.5B, using the same data and seed, Stable-LoRA$^+$ and vanilla LoRA reach similar loss trajectories (Figure 15): both approach the 0.12–0.19 range within 50 steps, ending at 0.1675 and 0.1646, respectively. Their GSM8K accuracies nevertheless differ by 1.74 pp (61.94% versus 60.20%). Thus, lower terminal training loss does not explain the advantage for this checkpoint pair.

Similar scalar losses do not imply identical optimization trajectories or exclude optimization-mediated changes to generalization. The higher effective rank and changed factor norms identify geometric differences that accompany the accuracy gap; determining their causal contribution would require controlled interventions and repeated-seed analysis.

The agreement plot summarizes relative loss differences along the recorded trajectories. Its empirical limits are descriptive: successive training steps are correlated, and the plot is not an equivalence test. Likewise, one paired run cannot establish statistical equality of the terminal losses. For configuration selection, these observations motivate evaluating held-out accuracy alongside training loss, with validation used for selection and the test set reserved for final evaluation.

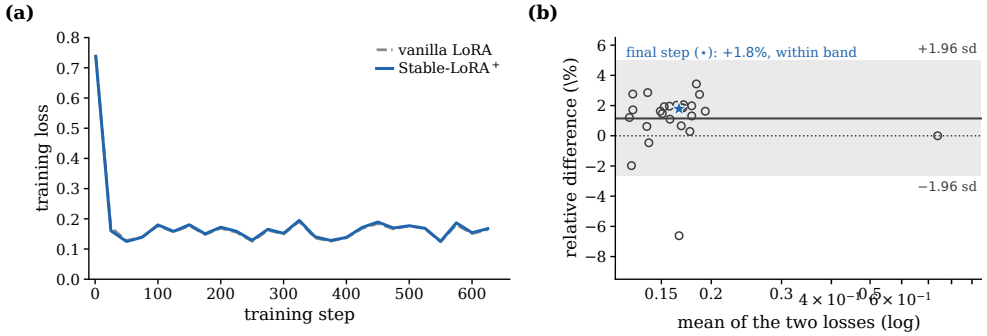


Figure 15: Training-loss trajectories on Qwen-1.5B under the same data and seed. The final losses are 0.1675 and 0.1646 for Stable-LoRA$^+$ and vanilla LoRA, respectively, while test accuracies are 61.9% and 60.2%. Panel (b) summarizes relative differences with empirical ± 1.96 standard-deviation limits; these describe agreement along the trajectories rather than establish statistical equivalence.

J Cross-Scale Structure: Solved Sets and Frontier Ranks

Aggregating the per-question records of the five headline checkpoints also reveals how the *set* of solved problems evolves with scale. Figure 16 reports the containment matrix $C_{ij} = |S_i \cap S_j|/|S_i|$, where S_i is the set of GSM8K problems the scale-i checkpoint solves. Capability grows almost monotonically:

97.2% of the problems solved at 0.5B are still solved at 14B (only 16 of 562 regress), and the 14B column is at least 93.7% for every smaller scale. A scale-invariant core of 468 problems (35.5%) is solved at all five scales, 94 (7.1%) resist every scale, and the union over scales reaches 1,225 (92.9%); the mild 0.5B→1.5B dip (86.5%) shows that the solved sets are not strictly nested. These comparisons use best-protocol checkpoints rather than a controlled change in model size alone.

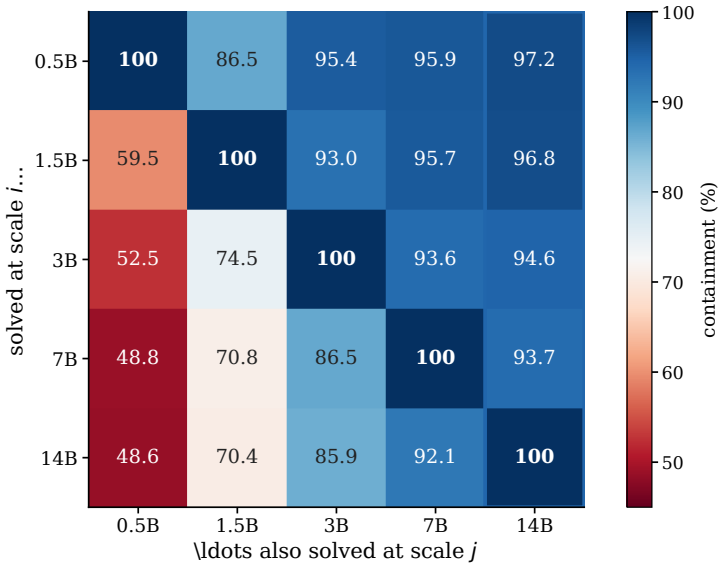


Figure 16: Cross-scale solved-set containment over the five headline checkpoints: cell (i, j) = fraction of problems solved at scale i also solved at scale j. The outlined 14B column ($\geq$ 93.7%) shows the largest backbone nearly subsumes every smaller-scale solution set.

Cross-scale rank comparison. Figure 17 summarizes the nine eligible single-LR methods using the best-protocol accuracies in Table 1, with average ranks for ties. Stable-LoRA$^+$ ranks first at 0.5/1.5B, shares first place at 3/7B, and has rank 2.5 at 14B. Its mean rank is 1.50, compared with 3.90 for vanilla LoRA. The map characterizes variation in relative performance across scales; excluded failures and selected protocols should be considered when interpreting this aggregate.

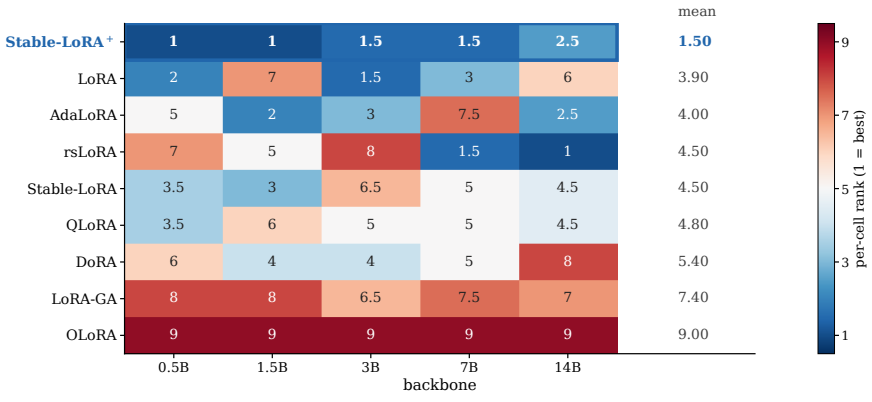


Figure 17: Per-cell rank of every single-learning-rate method at each scale (rank 1 = best, average-tie; "–" marks a diverged or unrun cell), computed from the Table 1 accuracies. Stable-LoRA$^+$ (outlined) is the only method that stays in the top rank band at every scale, giving the lowest mean per-cell rank (1.50); baseline ranks vary across scales.

Synthesis. The analyses provide two complementary descriptions. Fixed-recipe training logs show similar normalized B-norm trajectories, higher effective ranks, and scale-dependent realized A contraction. Best-protocol prediction records show overlapping but nonidentical solved sets and mean rank 1.50. The former characterize training dynamics; the latter characterize selected checkpoint

Init. (rsLoRA)	0.5B	1.5B	3B	7B	14B
PiSSA	2.05	24.34	36.47	–	–
OLoRA (QR)	25.32	47.69	64.14	54.21	79.76
Kaiming (B=0, Ours)	**42.61**	**61.94**	**77.33**	**83.70**	**85.14**

Table 11: Initialization $\times$ amplified scaling on GSM8K. PiSSA and OLoRA destabilize once composed with rsLoRA's strong $\alpha/\sqrt{r}$=12 scaling factor at r=16.

performance. Their agreement motivates further investigation of update geometry, but does not establish that the observed dynamics cause the accuracy differences or that those differences persist under reseeding.

A stronger test would combine matched-seed baselines, equal search budgets, and interventions on individual controls while holding architecture and precision fixed. This would separate the effects of the recipe from model scale, implementation, and selection. The current results supply empirical motivation and reproducible measurements for that investigation.

K Detailed Ablations and Analyses

K.1 Initialization and per-layer conditioning

Vanilla LoRA, rsLoRA, and Stable-LoRA$^+$ begin with sBA=0 because B is zero-initialized; PiSSA and OLoRA, by contrast, place $\|sBA\|_F \approx 47$ and 38 at step 0 on Qwen-14B (Appendix C), rivaling the row norms of $W^{(0)}$. Composed with rsLoRA's amplified $\alpha/\sqrt{r}$=12 scaling, the resulting initialization can substantially perturb base activations, and the evaluated runs exhibit degraded accuracy (Table 11). The per-layer adapter conditioning $\kappa(BA)$ spans 1.9–9.1 across the 112 $\{q, k, v, o\}$ projections of the converged adapter (median 2.7, concentrated in the v-projections; Appendix C), yet scaling the shrinkage by it does not improve on the constant γ of Table 4(B).

K.2 Training dynamics and compute

Per-method training loss on Qwen-1.5B (Appendix E, Figure 5, from the per-step training logs) shows the convergent methods — LoRA, rsLoRA, DoRA, and Ours — converging to a near-identical loss (≈ 0.165), with OLoRA and LoRA+ on elevated plateaus. Training loss does not order the methods by accuracy: the four low-loss methods span 60–62% GSM8K, so the reported test-accuracy advantage is not explained by lower terminal training loss (Appendix I). Compute overhead is negligible (Appendix B), whereas DoRA costs 10–20% more wall-time without a corresponding accuracy gain. The stabilizers' contributions are unevenly distributed across scale: the quantization-aware correction has its clearest measured effect (+1.36 pp) at 7B QLoRA (Table 4A), whereas at 0.5B the full recipe *regresses* — the rsLoRA-subset cell (42.61) exceeds the full recipe (41.02), showing that the full recipe is not uniformly beneficial at the smallest scale.