REPA: RESIDUAL-SPACE PULLBACK FOR SCALE-CONSISTENT LOW-RANK ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Parameter-efficient fine-tuning (PEFT) is widely used to adapt large pretrained models under constrained memory and computational budgets, with Low-Rank Adaptation (LoRA) serving as a common foundation. Existing LoRA variants primarily modify the parameterization, scaling, initialization, or optimization of low-rank updates, but generally do not account for how these updates are transformed after entering a residual stream followed by Root Mean Square Layer Normalization (RMSNorm), where the projected first-order response depends on residual scale. To address this residual-scale mismatch, this paper proposes **REPA (Residual-Space Pullback for Scale-Consistent Low-Rank Adaptation)**. REPA projects the LoRA candidate update onto the subspace orthogonal to the current residual state, where the local RMSNorm Jacobian reduces to a scalar response, and rescales the projected update by the current residual RMS. This construction cancels the resulting inverse-scale modulation while preserving the standard LoRA rank and trainable parameter count. Theoretical analysis establishes exact local realization of the desired projected response, controlled second-order finite-step deviation, and uniqueness and minimum-norm properties of the resulting pullback. Across seven downstream benchmarks, REPA achieves state-of-the-art macro-average accuracy among the evaluated baselines at 81.19% and achieves optimal or near-optimal performance on every task. Compared with LoRA, REPA improves macro-average accuracy by 2.19 percentage points with the same trainable parameter count, while requiring only 1.004 times the full-training time. Controlled residual rescaling from 0.25 times to 4 times produces the predicted inverse-scale response for LoRA while keeping REPA near its reference response, confirming the scale-correction behavior.

1 INTRODUCTION

Parameter-efficient fine-tuning (PEFT) adapts large pretrained models under limited memory and computational budgets by updating only a small subset of parameters while keeping the backbone frozen, substantially reducing adaptation cost (Houlsby et al., 2019). Low-Rank Adaptation (LoRA) (Hu et al., 2022) parameterizes task-specific updates with trainable low-rank factors and is widely used for its simple parameterization, low memory overhead, and broad compatibility.

Existing LoRA variants improve low-rank adaptation along several directions. AdaLoRA (Zhang et al., 2023), QLoRA (Dettmers et al., 2023), and VeRA (Kopiczko et al., 2024) improve resource efficiency or capacity allocation, while E2LoRA (Li et al., 2026) and IGU-LoRA (Cui et al., 2026) further explore adaptive parameter sharing and rank allocation. Optimization, initialization, and scaling are addressed by rsLoRA (Kalajdzievski, 2023), LoRA+ (Hayou et al., 2024), and DoRA (Liu et al., 2024), together with PiSSA (Meng et al., 2024) and LoRA-GA (Wang et al., 2024). More recent methods, including LoRA-Pro (Wang et al., 2025), LoRA-One (Zhang et al., 2025), and LoFT (Tastan et al., 2026), pursue more faithful optimization or closer alignment with full fine-tuning, while LoRA-Dash (Si et al., 2025) and MiSS (Kang & Yin, 2026) further refine the structure or efficiency of low-rank adaptation.

Normalization and update geometry provide another closely related line of work. Root Mean Square Layer Normalization (RMSNorm) (Zhang & Sennrich, 2019) normalizes activations by their root-mean-square magnitude, while normalization placement affects Transformer optimization and gra-

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.

dient behavior (Xiong et al., 2020). Recent methods introduce normalization, projection, or explicit scale control into low-rank adaptation: NoRA (Kang et al., 2026) incorporates normalization, OPLoRA (Xiong & Xie, 2026) applies orthogonal projection, and Scale Regularization (Tan, 2026) explicitly constrains adaptation scale. However, these approaches do not directly address how a learned low-rank update is realized after entering an RMS-normalized residual stream, where its normalized response can still depend on the current residual scale.

This dependence becomes explicit at residual-output interfaces followed by RMSNorm. Let $f(\mathbf{y}) = \mathbf{y}/\rho_\epsilon(\mathbf{y})$ denote the pre-gain RMS normalization map, where $\mathbf{y}$ is the residual state and $\rho_\epsilon(\mathbf{y})$ its stabilized RMS magnitude. For any residual-orthogonal perturbation $\mathbf{v}$ satisfying $\mathbf{y}^\top \mathbf{v} = 0$, the Jacobian $\mathbf{J}_f(\mathbf{y})$ satisfies $\mathbf{J}_f(\mathbf{y})\mathbf{v} = \mathbf{v}/\rho_\epsilon(\mathbf{y})$. Thus, the projected first-order normalized response scales inversely with the current residual RMS. This *residual-scale mismatch* differs from rank-dependent scaling or factor-space optimization: a fixed LoRA scaling coefficient controls the residual-space update but cannot ensure a scale-consistent projected response after RMSNorm.

To address this mismatch, this paper proposes **REPA (Residual-Space Pullback for Scale-Consistent Low-Rank Adaptation)**. REPA preserves the standard LoRA low-rank mapping while changing how its output is realized in the residual stream. The candidate update $\mathbf{q}$ is first restricted to the residual-orthogonal subspace, where the local RMSNorm Jacobian admits a scalar response, and the resulting component $\mathbf{q}_\perp$ is then rescaled by the current residual RMS. This construction yields $\mathbf{J}_f(\mathbf{y})\Delta\mathbf{y}_{\mathrm{REPA}} = \mathbf{q}_\perp$ locally, removing the residual-scale modulation without changing the LoRA parameterization, adapter rank, or trainable parameter count. The main contributions are summarized as follows:

- This paper proposes **REPA (Residual-Space Pullback for Scale-Consistent Low-Rank Adaptation)**, a residual-space adaptation method that preserves the standard LoRA parameterization while correcting how low-rank updates are realized at residual-output interfaces followed by RMSNorm.
- REPA employs residual-orthogonal projection to remove the residual-parallel component and restrict the learned update to the subspace where the local RMSNorm Jacobian has a scalar response, providing a well-conditioned basis for residual-space correction.
- REPA further applies an RMS pullback that rescales the projected update by the current residual RMS, canceling the remaining inverse-scale modulation and producing a scale-consistent projected first-order response without increasing adapter rank or trainable parameter count.
- Theoretical analysis establishes exact local realization of the target projected response, controlled second-order finite-step deviation, and unique/minimum-norm properties of the residual-space pullback, providing a geometric justification for REPA.
- Extensive experiments across seven downstream benchmarks show that REPA achieves the optimal macro-average accuracy of 81.19%, improving LoRA by 2.19 percentage points with the same trainable parameter count. Matched timing requires only 1.004 times the LoRA full-training time, while mechanistic analyses confirm the predicted scale-correction behavior.

2 Related Work

Parameter-Efficient Fine-Tuning and Low-Rank Adaptation. Parameter-efficient fine-tuning (PEFT) adapts pretrained models while updating only a small subset of parameters. Representative approaches include adapter tuning (Houlsby et al., 2019) and Low-Rank Adaptation (LoRA) (Hu et al., 2022). Subsequent methods improve resource efficiency or capacity allocation through dynamic rank budgeting (AdaLoRA (Zhang et al., 2023)), quantized backbones (QLoRA (Dettmers et al., 2023)), shared low-rank bases (VeRA (Kopiczko et al., 2024)), and adaptive sharing or rank allocation (E2LoRA (Li et al., 2026); IGU-LoRA (Cui et al., 2026)). REPA instead corrects how a learned low-rank update is realized in an RMS-normalized residual stream, improving adaptation performance without increasing the trainable parameter count.

Optimization, Initialization, and Scaling of LoRA. LoRA variants also modify optimization, initialization, and scaling. rsLoRA (Kalajdzievski, 2023) changes rank-dependent scaling,

LoRA+ (Hayou et al., 2024) uses asymmetric learning rates, and DoRA (Liu et al., 2024) decouples weight magnitude from direction. PiSSA (Meng et al., 2024) and LoRA-GA (Wang et al., 2024) improve initialization, while LoRA-Pro (Wang et al., 2025), LoRA-One (Zhang et al., 2025), and LoFT (Tastan et al., 2026) pursue closer alignment with full fine-tuning. REPA instead preserves the low-rank parameterization and corrects the state-dependent residual-space realization of its update.

Normalization-, Projection-, and Scale-Aware Adaptation. Root Mean Square Layer Normalization (RMSNorm) (Zhang & Sennrich, 2019) normalizes activations by their root-mean-square magnitude, and normalization placement affects Transformer optimization (Xiong et al., 2020). Recent LoRA variants incorporate normalization, projection, or explicit scale control, including NoRA (Kang et al., 2026), OPLoRA (Xiong & Xie, 2026), and Scale Regularization (Tan, 2026). REPA derives residual-orthogonal projection and RMS pullback jointly from the local RMSNorm response to an additive residual perturbation, yielding a scale-consistent projected response without changing the LoRA parameterization or trainable parameter count.

3 METHOD

3.1 RESIDUAL-SCALE MISMATCH IN PRE-RMSNORM LORA

Modern decoder-only language models commonly use pre-normalization (Xiong et al., 2020). When Low-Rank Adaptation (LoRA) (Hu et al., 2022) is attached to residual-producing output projections, such as the attention output projection or multilayer perceptron (MLP) down-projection, its output is added to the residual stream before a subsequent RMSNorm (Zhang & Sennrich, 2019). At these interfaces, the normalized effect of a low-rank update depends directly on the scale of the current residual state.

Let $\mathbf{z} \in \mathbb{R}^{d_{\text{in}}}$ denote the input to an adapted linear transformation. A rank-r LoRA adapter with trainable factors $\mathbf{A} \in \mathbb{R}^{r \times d_{\text{in}}}$ and $\mathbf{B} \in \mathbb{R}^{d_{\text{out}} \times r}$ produces

$$\mathbf{q} = s\mathbf{B}\mathbf{A}\mathbf{z}, \tag{1}$$

where $\mathbf{q} \in \mathbb{R}^{d_{\text{out}}}$ is the candidate update, $s = \alpha/r$ is the conventional LoRA scaling coefficient, and α is the LoRA scaling hyperparameter.

Let $\mathbf{y} \in \mathbb{R}^{d_{\text{out}}}$ denote the residual state immediately before update injection. For the MLP down-projection used in the experiments, $\mathbf{y}$ contains the skip residual and frozen MLP output. The resulting LoRA residual state is

$$\mathbf{y}^{+}_{\text{LoRA}} = \mathbf{y} + \mathbf{q}. \tag{2}$$

The magnitude of $\mathbf{q}$ does not directly determine its effect after normalization. Define the pre-gain RMS normalization map

$$f(\mathbf{y}) = \frac{\mathbf{y}}{\rho_\epsilon(\mathbf{y})}, \qquad \rho_\epsilon(\mathbf{y}) = \sqrt{\frac{1}{d_{\text{out}}}\|\mathbf{y}\|_2^2 + \epsilon}, \tag{3}$$

where $\epsilon \geq 0$ is the numerical stabilization constant. The learned RMSNorm gain is omitted because it does not alter the residual-scale dependence considered here.

Let $\mathbf{J}_f(\mathbf{y})$ denote the Jacobian of f. Section 3.2 shows that any $\mathbf{v} \perp \mathbf{y}$ satisfies

$$\mathbf{J}_f(\mathbf{y})\mathbf{v} = \frac{1}{\rho_\epsilon(\mathbf{y})}\mathbf{v}. \tag{4}$$

Thus, the first-order normalized response of a residual-orthogonal perturbation scales inversely with residual RMS. A fixed LoRA scaling coefficient therefore does not control its projected post-RMSNorm response. This state dependence is termed *residual-scale mismatch*.

3.2 REPA: SCALE-CONSISTENT RESIDUAL-SPACE PULLBACK

REPA (Residual-Space Pullback for Scale-Consistent Low-Rank Adaptation) modifies how a LoRA update is realized in residual space without changing its low-rank parameterization. The local RMSNorm geometry introduces two coupled effects: residual-parallel and residual-orthogonal perturbations have different sensitivities, while the orthogonal response retains the factor $1/\rho_\epsilon(\mathbf{y})$. REPA

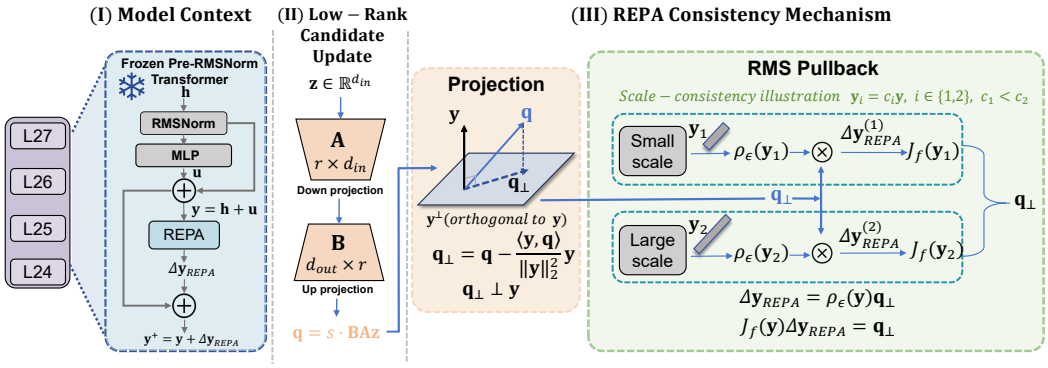

Figure 1: Overview of REPA. Frozen residual computation produces $\mathbf{y}$ and the trainable low-rank factors produce $\mathbf{q} = s\mathbf{BAz}$. Residual-orthogonal projection isolates $\mathbf{q}_\perp$, where the local RMSNorm response is scalar, and RMS-based pullback compensates for residual-scale modulation. The resulting update $\Delta\mathbf{y}_{\mathrm{REPA}} = \rho_\epsilon(\mathbf{y})\mathbf{q}_\perp$ satisfies $\mathbf{J}_f(\mathbf{y})\Delta\mathbf{y}_{\mathrm{REPA}} = \mathbf{q}_\perp$ locally.

addresses them in two steps: residual-orthogonal projection removes the poorly conditioned parallel component, and an RMS pullback cancels the remaining residual-scale dependence. The resulting update yields a projected first-order normalized response that is independent of the current residual RMS. Figure 1 illustrates this construction.

Directional response of RMSNorm. REPA begins by characterizing how RMSNorm transforms a perturbation after it enters the residual stream. For a sufficiently small perturbation $\mathbf{q}$,

$$f(\mathbf{y}+\mathbf{q}) - f(\mathbf{y}) = \mathbf{J}_f(\mathbf{y})\mathbf{q} + \mathcal{O}\big(\|\mathbf{q}\|_2^2\big)\,, \tag{5}$$

where

$$\mathbf{J}_f(\mathbf{y}) = \frac{1}{\rho_\epsilon(\mathbf{y})}\left(\mathbf{I} - \frac{\mathbf{y}\mathbf{y}^\top}{\|\mathbf{y}\|_2^2 + d_{\mathrm{out}}\epsilon}\right), \tag{6}$$

and $\mathbf{I} \in \mathbb{R}^{d_{\mathrm{out}}\times d_{\mathrm{out}}}$ is the identity matrix.

For $\mathbf{y} \neq \mathbf{0}$, decompose the candidate update into its residual-parallel and residual-orthogonal components relative to the current residual direction:

$$\mathbf{q}_\parallel = \frac{\mathbf{y}^\top\mathbf{q}}{\|\mathbf{y}\|_2^2}\mathbf{y}, \qquad \mathbf{q}_\perp = \mathbf{q} - \mathbf{q}_\parallel, \tag{7}$$

so that $\mathbf{y}^\top\mathbf{q}_\perp = 0$. Substituting Eq. (7) into Eq. (6) gives

$$\mathbf{J}_f(\mathbf{y})\mathbf{q}_\perp = \frac{1}{\rho_\epsilon(\mathbf{y})}\mathbf{q}_\perp, \qquad \mathbf{J}_f(\mathbf{y})\mathbf{q}_\parallel = \frac{\epsilon}{\rho_\epsilon(\mathbf{y})^3}\mathbf{q}_\parallel. \tag{8}$$

Denoting the corresponding scalar sensitivities by $\lambda_\parallel$ and $\lambda_\perp$,

$$\frac{\lambda_\parallel}{\lambda_\perp} = \frac{\epsilon}{\rho_\epsilon(\mathbf{y})^2}. \tag{9}$$

Eqs. (8)–(9) show that the parallel component is strongly attenuated when $\epsilon \ll \rho_\epsilon(\mathbf{y})^2$ and becomes a null direction at $\epsilon = 0$, whereas the orthogonal component retains the $1/\rho_\epsilon(\mathbf{y})$ dependence. REPA therefore addresses both directional asymmetry and residual-scale modulation.

Residual-orthogonal projection. Directly inverting the residual-parallel response would require a factor proportional to $\rho_\epsilon(\mathbf{y})^3/\epsilon$, which becomes ill-conditioned as $\epsilon \to 0$. REPA instead restricts the candidate update to

$$\mathcal{T}_\mathbf{y} = \left\{\mathbf{v} \in \mathbb{R}^{d_{\mathrm{out}}} : \mathbf{y}^\top\mathbf{v} = 0\right\}, \tag{10}$$

the tangent space of the constant-ρ_ϵ level set through $\mathbf{y}$. On this subspace, the RMSNorm Jacobian reduces to the scalar response in Eq. (8), making a closed-form pullback possible.

The corresponding projection is

$$\mathbf{P}_{\mathbf{y}} = \mathbf{I} - \frac{\mathbf{y}\mathbf{y}^\top}{\|\mathbf{y}\|_2^2}, \qquad \mathbf{q}_\perp = \mathbf{P}_{\mathbf{y}}\mathbf{q} = \mathbf{q} - \frac{\mathbf{y}^\top\mathbf{q}}{\|\mathbf{y}\|_2^2}\mathbf{y}. \tag{11}$$

Here, $\mathbf{P}_{\mathbf{y}}$ denotes the orthogonal projection onto $\mathcal{T}_{\mathbf{y}}$. The projection isolates the subspace where RMSNorm has a scalar local response, and $\mathbf{q}_\perp$ serves as the target for the subsequent pullback.

RMS pullback. Projection resolves the directional asymmetry, but the projected response still depends on residual RMS through $1/\rho_\epsilon(\mathbf{y})$. REPA cancels this factor by realizing the projected target as

$$\Delta\mathbf{y}_{\mathrm{REPA}} = \rho_\epsilon(\mathbf{y})\mathbf{q}_\perp. \tag{12}$$

The adapted residual state is

$$\mathbf{y}^+_{\mathrm{REPA}} = \mathbf{y} + \Delta\mathbf{y}_{\mathrm{REPA}}. \tag{13}$$

Because $\mathbf{q}_\perp \in \mathcal{T}_{\mathbf{y}}$,

$$\mathbf{J}_f(\mathbf{y})\Delta\mathbf{y}_{\mathrm{REPA}} = \mathbf{q}_\perp. \tag{14}$$

Eq. (14) captures the central property of REPA: projection selects the scalar-response subspace, while the RMS pullback cancels its scale factor. The learned low-rank mapping therefore controls the projected first-order normalized response independently of residual RMS, without increasing adapter rank or trainable parameter count.

Properties and practical implications. REPA is closed-form and introduces no additional trainable parameters. Its projection and RMS pullback require only $\mathcal{O}(d_{\mathrm{out}})$ vector operations, without explicit Jacobian construction or matrix inversion. During optimization, $\mathbf{P}_{\mathbf{y}}$ and $\rho_\epsilon(\mathbf{y})$ are computed from a stop-gradient residual state, preserving gradients to the LoRA factors without additional gradient paths.

Here, *scale-consistent* refers specifically to the projected first-order response. Because the correction remains state-dependent through $\rho_\epsilon(\mathbf{y})$ and $\mathbf{P}_{\mathbf{y}}$, it cannot generally be reduced to a single statically merged weight update. The formulation applies to residual-output interfaces followed by RMSNorm, internal projections require analysis of the corresponding composite mapping.

3.3 Theoretical Properties

REPA admits exact local realization, controlled finite-step behavior, and a geometrically characterized residual-space preimage. Complete proofs appear in Appendix A.

Theorem 1 (Exact Local Realization). Let $\mathbf{y} \neq \mathbf{0}$ and $\mathbf{q}_\perp \in \mathcal{T}_{\mathbf{y}}$. For any $\epsilon \geq 0$, the REPA update in Eq. (12) satisfies

$$\mathbf{J}_f(\mathbf{y})\Delta\mathbf{y}_{\mathrm{REPA}} = \mathbf{q}_\perp. \tag{15}$$

Proof. See Appendix A.1. □

Theorem 1 establishes exact local scale consistency: the RMS pullback cancels the $1/\rho_\epsilon(\mathbf{y})$ modulation on the residual-orthogonal subspace, rather than introducing a heuristic scaling rule.

Theorem 2 (Exact Finite-Step Characterization). Let $\mathbf{y} \neq \mathbf{0}$, $\mathbf{q}_\perp \in \mathcal{T}_{\mathbf{y}}$, and $\epsilon \geq 0$. For a scalar step multiplier $\eta \geq 0$,

$$f(\mathbf{y} + \eta\Delta\mathbf{y}_{\mathrm{REPA}}) = \frac{f(\mathbf{y}) + \eta\mathbf{q}_\perp}{\sqrt{1 + \frac{\eta^2}{d_{\mathrm{out}}}\|\mathbf{q}_\perp\|_2^2}}, \tag{16}$$

and consequently

$$\|f(\mathbf{y} + \eta\Delta\mathbf{y}_{\mathrm{REPA}}) - f(\mathbf{y}) - \eta\mathbf{q}_\perp\|_2 = \mathcal{O}(\eta^2), \qquad \eta \to 0. \tag{17}$$

Proof. See Appendix A.2. □

Theorem 2 shows that finite perturbations recover the target displacement to first order with second-order deviation.

Table 1: Trainable parameter overhead under the main configuration. Δ Params. and overhead are relative to LoRA.

Method	Trainable Params.	Δ Params.	Overhead
LoRA	335,872	0	0.000%
rsLoRA	335,872	0	0.000%
DoRA	342,016	+6,144	+1.829%
AdaLoRA	503,856	+167,984	+50.014%
VeRA	6,176	−329,696	−98.161%
PiSSA	335,872	0	0.000%
LoRA-Dash	335,904	+32	+0.010%
MiSS	49,152	−286,720	−85.366%
REPA	**335,872**	**0**	**0.000%**

Theorem 3 (Unique and Minimum-Norm Pullback). Let $\mathbf{y} \neq \mathbf{0}$ and $\mathbf{q}_\perp \in \mathcal{T}_\mathbf{y}$. Consider any $\Delta\mathbf{x} \in \mathbb{R}^{d_{\text{out}}}$ satisfying

$$\mathbf{J}_f(\mathbf{y})\Delta\mathbf{x} = \mathbf{q}_\perp. \tag{18}$$

If $\epsilon > 0$, the unique solution is

$$\Delta\mathbf{x} = \rho_\epsilon(\mathbf{y})\mathbf{q}_\perp. \tag{19}$$

If $\epsilon = 0$, define $\rho_0(\mathbf{y}) = \rho_\epsilon(\mathbf{y})|_{\epsilon=0}$. Every feasible solution then has the form

$$\Delta\mathbf{x} = \rho_0(\mathbf{y})\mathbf{q}_\perp + \beta\mathbf{y}, \qquad \beta \in \mathbb{R}. \tag{20}$$

REPA is then the unique minimum-Euclidean-norm solution:

$$\Delta\mathbf{y}_{\text{REPA}} = \underset{\mathbf{J}_f(\mathbf{y})\Delta\mathbf{x}=\mathbf{q}_\perp}{\operatorname{argmin}} \|\Delta\mathbf{x}\|_2. \tag{21}$$

Proof. See Appendix A.3. □

Theorem 3 establishes a unique feasible pullback for $\epsilon > 0$ and the unique minimum-norm feasible pullback for $\epsilon = 0$, excluding unnecessary residual-parallel components.

3.4 Complexity Analysis

For a rank-r LoRA adapter, let $N_r = r(d_{\text{in}} + d_{\text{out}})$ and $C_r = \mathcal{O}(rd_{\text{in}} + rd_{\text{out}})$ denote its trainable parameter count and adapter-side computation. REPA preserves the same rank-r factors and adds only $\mathcal{O}(d_{\text{out}})$ residual-space vector operations:

$$N_{\text{REPA}} = N_r, \qquad C_{\text{REPA}} = C_r + \mathcal{O}(d_{\text{out}}). \tag{22}$$

Table 1 summarizes trainable-parameter overhead under the main configuration. REPA adds no trainable parameters beyond LoRA and requires only linear-cost residual-space operations.

4 Experiments

The experiments assess REPA in downstream performance, adaptation efficiency, rank sensitivity, and the residual-scale mechanism predicted by the theory. Appendix B.5 further separates the effects of residual-orthogonal projection and RMS-based scaling.

4.1 Experimental Setup

Backbone and adaptation scope. All experiments use Qwen2.5-1.5B-Instruct (Yang et al., 2024) with the pretrained backbone frozen. Unless otherwise specified, adapters are applied to the MLP `down_proj` modules of the final four Transformer blocks (layers 24–27), whose outputs enter the residual stream directly. The main benchmark uses nominal rank $r = 8$ where applicable, with method-specific rank semantics following the corresponding implementations.

Table 2: Final-evaluation accuracy (%) on seven downstream benchmarks, averaged over three optimization seeds. The Avg. column reports the macro-average across tasks. Best results are shown in **bold** and second-best results are underlined, ties receive the same formatting. REPA achieves the highest macro-average and ranks first or second on all seven tasks.

Method	SST-2	RTE	MRPC	AG News	PAWS	MNLI	WiC	Avg.
LoRA	90.33	86.67	72.00	87.33	82.00	79.00	55.67	79.00
rsLoRA	91.67	87.00	73.00	87.33	84.00	82.00	**57.67**	80.38
DoRA	91.00	86.33	72.00	87.33	82.00	78.00	55.33	78.86
AdaLoRA	90.00	87.33	74.00	87.00	77.00	75.00	53.67	77.71
VeRA	90.33	83.00	74.00	84.00	77.00	70.00	53.67	76.00
PiSSA	91.00	87.33	72.33	87.33	**84.67**	80.67	57.33	80.09
LoRA-Dash	91.00	**87.67**	72.00	**88.67**	82.33	79.00	55.33	79.43
MiSS	**92.33**	**87.67**	74.33	87.00	82.67	76.67	**57.67**	79.76
REPA	92.00	**87.67**	**75.67**	88.00	84.00	**83.33**	**57.67**	**81.19**

Datasets and protocol. The main benchmark covers SST-2, RTE, MRPC, and MNLI from the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019), AG News (Zhang et al., 2015), PAWS (Zhang et al., 2019), and WiC (Pilehvar & Camacho-Collados, 2019). Each run uses 500 training examples, batch size 1, 32-bit floating-point precision, and 10 epochs (5,000 optimizer steps), with results averaged over optimization seeds $\{2026, 2027, 2028\}$. Task-specific sequence lengths and full training settings appear in Appendix B.1.

All methods use the same adaptation scope, training budget, checkpoint-selection rule, and evaluation procedure. Dataset construction is deterministic, and data files, split seeds, and hashes associated with each formal run are recorded in the run metadata. Baselines use a common outer configuration, whereas REPA learning rates and scaling coefficients are selected per dataset using only held-out development data. Final-evaluation data are not used for hyperparameter or checkpoint selection. Tasks use fixed-label conditional scoring with matched-length verbalizers under the Qwen2.5 tokenizer.

Baselines. REPA is compared with LoRA (Hu et al., 2022), rsLoRA (Kalajdzievski, 2023), DoRA (Liu et al., 2024), AdaLoRA (Zhang et al., 2023), VeRA (Kopiczko et al., 2024), PiSSA (Meng et al., 2024), LoRA-Dash (Si et al., 2025), and MiSS (Kang & Yin, 2026). Method-specific operations follow their corresponding implementations; full implementation and reproducibility details appear in Appendices B.1 and B.2.

4.2 Main Benchmark Results

Table 2 shows that REPA achieves the optimal macro-average accuracy of 81.19%, improving over LoRA by 2.19 percentage points at the same adapter rank and trainable-parameter count, while achieving optimal or near-optimal performance on all seven tasks. It improves over LoRA on every benchmark, with the largest gains on MNLI ($+4.33$ points) and MRPC ($+3.67$), achieves the best results on MRPC and MNLI, and ties for best on RTE and WiC.

4.3 Adaptation Efficiency

REPA's residual-space correction adds little practical overhead. Under matched rank-8, 5,000-step runs, Table 3 reports relative full-training wall-clock time normalized to LoRA and geometrically averaged across seven dataset-level three-seed means. REPA requires only 1.004 times the LoRA full-training time, corresponding to approximately 0.4% additional wall-clock cost.

Near-peak target attainment. To evaluate how quickly each method approaches LoRA's attainable validation improvement, the target for each dataset–seed pair is

$$\tau_\gamma = a_0 + \gamma \left(a_{\text{best}}^{\text{LoRA}} - a_0\right), \qquad \gamma \in [0, 1], \tag{23}$$

where a_0 is the pre-adaptation validation accuracy and $a_{\text{best}}^{\text{LoRA}}$ is LoRA's best validation accuracy. Time-to-target (TTA) is the cumulative training time at the first checkpoint reaching τ_γ.

Table 3: Relative full-training wall-clock time under matched rank-8, 5,000-step runs, normalized to LoRA and geometrically averaged across seven dataset-level three-seed means.

Method	Relative Runtime ↓
LoRA	1.000×
AdaLoRA	1.029×
DoRA	1.158×
REPA	**1.004**×

Table 4: Near-peak target attainment at $\gamma = 0.95$ across seven tasks with three seeds each. Reach denotes target-reaching runs; Steps and time-to-target (TTA) are medians over reached runs. The final columns report paired comparisons with REPA and percentile-bootstrap 95% confidence intervals (CIs).

	Target Attainment			Paired Comparison with REPA	
Method	Reach ↑	Steps ↓	TTA (s) ↓	Earlier / paired	Reduction [95% CI] ↑
LoRA	21/21	3500	411.82	13/20	+18.81 [–5.50, 37.09]%
rsLoRA	20/21	2500	348.64	10/20	–0.03 [–0.45, 15.56]%
DoRA	20/21	3250	530.85	16/20	+29.26 [7.63, 45.73]%
AdaLoRA	4/21	2250†	430.63†	0/3	–
VeRA	1/21	2000†	314.04†	0/1	–
PiSSA	20/21	2000	293.65	9/19	–0.04 [–28.49, 30.90]%
LoRA-Dash	19/21	2500	378.85	13/19	+25.58 [–9.65, 49.85]%
MiSS	17/21	2500	390.71	8/16	+11.17 [–11.01, 49.76]%
REPA	**20/21**	2500	338.21	–	–

† Conditional median over the small subset of target-reaching runs, paired reductions are omitted for AdaLoRA and VeRA because only three and one comparable pairs are available, respectively.

For comparison method m, REPA's paired TTA reduction is

$$\Delta_{\mathrm{TTA}}(m) = 100 \times \frac{T_m - T_{\mathrm{REPA}}}{T_m}, \tag{24}$$

with positive values favoring REPA; only jointly reached targets are included.

At $\gamma = 0.95$, REPA reaches the target in 20/21 runs at a median of 2,500 steps and 338.21 seconds. Among 20 comparable LoRA–REPA pairs, REPA is earlier in 13, with a median paired TTA reduction of 18.81%; the corresponding confidence interval includes zero. Against DoRA, REPA is earlier in 16/20 pairs, with a 29.26% median reduction (95% CI $[7.63\%, 45.73\%]$). At $\gamma = 1.00$, the median reduction relative to LoRA remains 16.79%. Together with its near-LoRA full-training cost, these results indicate competitive target-attainment efficiency at stringent performance thresholds. Full results appear in Appendix B.6.

4.4 Performance Across Rank Budgets

Rank robustness is evaluated on AG News for $r \in \{1, 2, 4, 8, 16, 32, 64\}$ with the backbone, adaptation locations, training data, optimization budget, and evaluation protocol fixed. Each rank uses five seeds, a learning rate of 7.5×10^{-5}, and $\alpha = r$, such that $\alpha/r = 1$.

Figure 2(a) shows that REPA remains robust across a wide range of rank budgets, including strongly constrained settings. At $r = 1$, REPA uses 87.5% fewer trainable parameters than at $r = 8$ with only a 0.4-point reduction in mean accuracy, indicating limited sensitivity to severe rank reduction. Across the 64-fold range in trainable adapter parameters, mean accuracy remains within 87.0%–88.2%, demonstrating stable performance despite large changes in adapter capacity.

4.5 Mechanistic Analysis

The mechanistic analysis examines whether residual scale varies substantially across consecutive residual-output interfaces in practice and whether REPA exhibits the scale-correction behavior predicted by the local RMSNorm geometry under controlled residual rescaling.

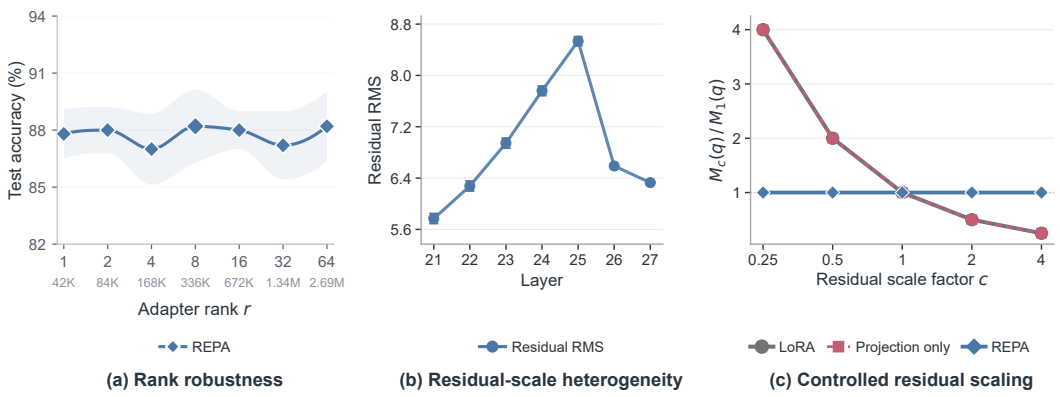

Figure 2: Rank robustness and mechanistic analysis of REPA. **(a)** Mean test accuracy on AG News across seven adapter ranks and five seeds, with one standard deviation shaded; $r = 8$ is highlighted and the second row reports trainable adapter parameters. **(b)** Mean residual RMS across seven consecutive interfaces with 95% bootstrap confidence intervals; layers 21–23 are probe-only and layers 24–27 are adapted interfaces. **(c)** Under controlled residual rescaling $\mathbf{y} \to c\mathbf{y}$ with $\mathbf{q}$ fixed, LoRA and projection only exhibit the predicted inverse-scale response, whereas REPA remains approximately constant.

Figure 2(b) shows substantial variation in residual RMS across consecutive interfaces, indicating that the scale term in Section 3.1 is not constant in practice and can differ markedly across nearby layers. This layer-dependent variation provides the residual-scale context in which REPA operates and shows that the relevant scale is not constant across nearby residual-output interfaces.

To isolate the effect of residual scale, the residual state is rescaled while $\mathbf{q}$ is held fixed:

$$\mathbf{y} \leftarrow c\mathbf{y}, \qquad c \in \{0.25, 0.5, 1, 2, 4\}, \tag{25}$$

$$M_c(\mathbf{q}) = \|f(c\mathbf{y} + \Delta\mathbf{y}_c) - f(c\mathbf{y})\|_2\,, \tag{26}$$

where $\Delta\mathbf{y}_c$ is the corresponding perturbation and the responses are normalized by $M_1(\mathbf{q})$.

Figure 2(c) shows that LoRA and projection only follow the predicted inverse-scale response, whereas REPA maintains $M_c(\mathbf{q})/M_1(\mathbf{q}) \approx 1$ throughout the controlled intervention. The comparison separates the roles of the two components: residual-orthogonal projection identifies the scalar-response subspace, but the residual-scale dependence remains until RMS-based pullback cancels the corresponding modulation.

The component analysis provides complementary evidence at the task level. Projection alone leaves the three-task macro-average validation accuracy unchanged, whereas variants using RMS-based scaling attain the higher macro-average, and static RMS projection matches full REPA on this subset. These results associate the observed validation improvement with RMS-based scaling, while the controlled intervention separately confirms the scale-correction behavior predicted for full REPA. Full results appear in Appendices B.4 and B.5.

5 Conclusion

REPA addresses residual-scale mismatch through residual-orthogonal projection and RMS pullback, yielding a scale-consistent projected response without additional trainable parameters. Theory establishes its local and finite-step properties. Across seven benchmarks, REPA achieves the optimal macro-average accuracy of 81.19% among the evaluated methods. Compared with LoRA, it uses the same trainable parameter count and only 1.004 times the full-training time while improving macro-average accuracy by 2.19 percentage points. Mechanistic analyses further support the predicted scale-correction behavior. Together, these theoretical and empirical results show that residual-space correction provides an efficient and geometrically grounded approach to low-rank adaptation at RMS-normalized residual interfaces.

AI USE STATEMENT

Generative AI tools were used for language polishing, literature retrieval and discovery, research ideation and methodological consultation, drafting and revising portions of the manuscript, LaTeX formatting, code drafting and debugging, and assistance with mathematical derivations and proofs. All AI-assisted content was reviewed and verified by the authors.

REFERENCES

Xuan Cui, HuiYue Li, Run Zeng, Yunfei Zhao, Jinrui Qian, Wei Duan, Bo Liu, and Zhanpeng Zhou. IGU-LoRA: Adaptive rank allocation via integrated gradients and uncertainty-aware scoring. In *International Conference on Learning Representations*, 2026.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115, 2023. doi: 10.52202/075280-0441.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. LoRA+: Efficient low rank adaptation of large models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17783–17806. PMLR, 2024.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 2019.

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

Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with LoRA. *arXiv preprint arXiv:2312.03732*, 2023.

Jiale Kang and Qingyu Yin. MiSS: Revisiting the trade-off in LoRA with an efficient shard-sharing structure. In *International Conference on Learning Representations*, 2026.

Jiale Kang, Ziyin Yue, Zheng Zhan, Yangyi Huang, and Weiyang Liu. Normalized low-rank adaptation. *arXiv preprint arXiv:2608.31036*, 2026.

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

Minglei Li, Peng Ye, Jingqi Ye, Haonan He, and Tao Chen. E^2LoRA: Efficient and effective low-rank adaptation with entropy-guided adaptive sharing. In *International Conference on Learning Representations*, 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. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32100–32121. PMLR, 2024.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. PiSSA: Principal singular values and singular vectors adaptation of large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 121038–121072, 2024. doi: 10.52202/079017-3846.

Mohammad Taher Pilehvar and Jose Camacho-Collados. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1267–1273. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1128.

Chongjie Si, Zhiyi Shi, Shifan Zhang, Xiaokang Yang, Hanspeter Pfister, and Wei Shen. Unleashing the power of task-specific directions in parameter efficient fine-tuning. In *International Conference on Learning Representations*, 2025.

Xeng Ian Tan. Scale regularization for stable low-rank adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(48):41528–41530, 2026. doi: 10.1609/aaai.v40i48.42329.

Nurbek Tastan, Stefanos Laskaridis, Martin Takáč, Karthik Nandakumar, and Samuel Horváth. LoFT: Low-rank adaptation that behaves like full fine-tuning. In *International Conference on Learning Representations*, 2026.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019.

Shaowen Wang, Linxi Yu, and Jian Li. LoRA-GA: Low-rank adaptation with gradient approximation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 54905–54931, 2024. doi: 10.52202/079017-1741.

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*, 2025.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10524–10533. PMLR, 2020.

Yifeng Xiong and Xiaohui Xie. OPLoRA: Orthogonal projection LoRA prevents catastrophic forgetting during parameter-efficient fine-tuning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(40):34088–34096, 2026. doi: 10.1609/aaai.v40i40.40703.

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. *arXiv preprint arXiv:2412.15115*, 2024.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, volume 32, pp. 12381–12392, 2019.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2023.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, pp. 649–657, 2015.

Yuan Zhang, Jason Baldridge, and Luheng He. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1298–1308. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1131.

Yuanhe Zhang, Fanghui Liu, and Yudong Chen. LoRA-One: One-step full gradient could suffice for fine-tuning large language models, provably and efficiently. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 75513–75574. PMLR, 2025.

A Detailed Proofs of Theoretical Results

This appendix provides complete proofs of the theoretical results in Section 3.3. The notation follows Section 3.2. Recall the pre-gain RMS normalization map

$$f(\mathbf{y}) = \frac{\mathbf{y}}{\rho_\epsilon(\mathbf{y})}, \qquad \rho_\epsilon(\mathbf{y}) = \sqrt{\frac{1}{d_{\text{out}}}\|\mathbf{y}\|_2^2 + \epsilon}, \tag{27}$$

where $\mathbf{y} \in \mathbb{R}^{d_{\text{out}}}$ is the residual state, d_{out} is the residual dimension, and $\epsilon \geq 0$ is the numerical stabilization constant.

The residual-orthogonal subspace is

$$\mathcal{T}_\mathbf{y} = \left\{\mathbf{v} \in \mathbb{R}^{d_{\text{out}}} : \mathbf{y}^\top \mathbf{v} = 0\right\}. \tag{28}$$

For a LoRA candidate update $\mathbf{q}$, REPA retains the residual-orthogonal component

$$\mathbf{q}_\perp = \mathbf{P}_\mathbf{y}\mathbf{q}, \qquad \mathbf{P}_\mathbf{y} = \mathbf{I} - \frac{\mathbf{y}\mathbf{y}^\top}{\|\mathbf{y}\|_2^2}, \tag{29}$$

where $\mathbf{P}_\mathbf{y}$ is the orthogonal projection matrix. The REPA residual-space perturbation is

$$\Delta\mathbf{y}_{\text{REPA}} = \rho_\epsilon(\mathbf{y})\mathbf{q}_\perp. \tag{30}$$

A.1 Proof of Theorem 1: Exact Local Realization

Proof. For $\mathbf{y} \neq \mathbf{0}$, the Jacobian of $f(\mathbf{y}) = \mathbf{y}/\rho_\epsilon(\mathbf{y})$ is

$$\mathbf{J}_f(\mathbf{y}) = \frac{1}{\rho_\epsilon(\mathbf{y})}\left(\mathbf{I} - \frac{\mathbf{y}\mathbf{y}^\top}{\|\mathbf{y}\|_2^2 + d_{\text{out}}\epsilon}\right). \tag{31}$$

By construction, $\mathbf{q}_\perp \in \mathcal{T}_\mathbf{y}$, and therefore

$$\mathbf{y}^\top \mathbf{q}_\perp = 0. \tag{32}$$

Applying the Jacobian in Eq. (31) to $\mathbf{q}_\perp$ gives

$$\begin{aligned}\mathbf{J}_f(\mathbf{y})\mathbf{q}_\perp &= \frac{1}{\rho_\epsilon(\mathbf{y})}\left(\mathbf{q}_\perp - \frac{\mathbf{y}\mathbf{y}^\top\mathbf{q}_\perp}{\|\mathbf{y}\|_2^2 + d_{\text{out}}\epsilon}\right) \\ &= \frac{1}{\rho_\epsilon(\mathbf{y})}\mathbf{q}_\perp,\end{aligned} \tag{33}$$

where the second equality follows from $\mathbf{y}^\top\mathbf{q}_\perp = 0$.

Substituting the REPA update from Eq. (30) yields

$$\begin{aligned}\mathbf{J}_f(\mathbf{y})\Delta\mathbf{y}_{\text{REPA}} &= \mathbf{J}_f(\mathbf{y})\left[\rho_\epsilon(\mathbf{y})\mathbf{q}_\perp\right] \\ &= \rho_\epsilon(\mathbf{y})\mathbf{J}_f(\mathbf{y})\mathbf{q}_\perp \\ &= \rho_\epsilon(\mathbf{y})\frac{1}{\rho_\epsilon(\mathbf{y})}\mathbf{q}_\perp \\ &= \mathbf{q}_\perp.\end{aligned} \tag{34}$$

Thus, on the residual-orthogonal subspace $\mathcal{T}_\mathbf{y}$, the REPA pullback exactly cancels the $1/\rho_\epsilon(\mathbf{y})$ factor in the local RMSNorm response:

$$\mathbf{J}_f(\mathbf{y})\Delta\mathbf{y}_{\text{REPA}} = \mathbf{q}_\perp. \tag{35}$$

This proves the exact local realization property. □

A.2 Proof of Theorem 2: Exact Finite-Step Characterization

Proof. Let $\mathbf{q}_\perp \in \mathcal{T}_\mathbf{y}$ and define

$$\Delta\mathbf{y}_{\mathrm{REPA}} = \rho_\epsilon(\mathbf{y})\mathbf{q}_\perp. \tag{36}$$

For a scalar step multiplier $\eta \geq 0$, define

$$\mathbf{y}_\eta = \mathbf{y} + \eta\Delta\mathbf{y}_{\mathrm{REPA}} = \mathbf{y} + \eta\rho_\epsilon(\mathbf{y})\mathbf{q}_\perp. \tag{37}$$

Because $\mathbf{q}_\perp \in \mathcal{T}_\mathbf{y}$, the orthogonality condition $\mathbf{y}^\top\mathbf{q}_\perp = 0$ gives

$$\begin{aligned}\|\mathbf{y}_\eta\|_2^2 &= \|\mathbf{y} + \eta\rho_\epsilon(\mathbf{y})\mathbf{q}_\perp\|_2^2 \\ &= \|\mathbf{y}\|_2^2 + 2\eta\rho_\epsilon(\mathbf{y})\mathbf{y}^\top\mathbf{q}_\perp + \eta^2\rho_\epsilon(\mathbf{y})^2\|\mathbf{q}_\perp\|_2^2 \\ &= \|\mathbf{y}\|_2^2 + \eta^2\rho_\epsilon(\mathbf{y})^2\|\mathbf{q}_\perp\|_2^2. \end{aligned} \tag{38}$$

Using

$$\rho_\epsilon(\mathbf{y})^2 = \frac{1}{d_{\mathrm{out}}}\|\mathbf{y}\|_2^2 + \epsilon, \tag{39}$$

the squared RMS magnitude of $\mathbf{y}_\eta$ becomes

$$\begin{aligned}\rho_\epsilon(\mathbf{y}_\eta)^2 &= \frac{1}{d_{\mathrm{out}}}\|\mathbf{y}_\eta\|_2^2 + \epsilon \\ &= \rho_\epsilon(\mathbf{y})^2 + \frac{\eta^2}{d_{\mathrm{out}}}\rho_\epsilon(\mathbf{y})^2\|\mathbf{q}_\perp\|_2^2 \\ &= \rho_\epsilon(\mathbf{y})^2\left(1 + \frac{\eta^2}{d_{\mathrm{out}}}\|\mathbf{q}_\perp\|_2^2\right). \end{aligned} \tag{40}$$

Hence,

$$\rho_\epsilon(\mathbf{y}_\eta) = \rho_\epsilon(\mathbf{y})\sqrt{1 + \frac{\eta^2}{d_{\mathrm{out}}}\|\mathbf{q}_\perp\|_2^2}. \tag{41}$$

Substituting Eqs. (37) and (41) into the normalization map gives

$$\begin{aligned}f(\mathbf{y}_\eta) &= \frac{\mathbf{y} + \eta\rho_\epsilon(\mathbf{y})\mathbf{q}_\perp}{\rho_\epsilon(\mathbf{y})\sqrt{1 + \frac{\eta^2}{d_{\mathrm{out}}}\|\mathbf{q}_\perp\|_2^2}} \\ &= \frac{f(\mathbf{y}) + \eta\mathbf{q}_\perp}{\sqrt{1 + \frac{\eta^2}{d_{\mathrm{out}}}\|\mathbf{q}_\perp\|_2^2}}. \end{aligned} \tag{42}$$

Therefore,

$$f(\mathbf{y} + \eta\Delta\mathbf{y}_{\mathrm{REPA}}) = \frac{f(\mathbf{y}) + \eta\mathbf{q}_\perp}{\sqrt{1 + \frac{\eta^2}{d_{\mathrm{out}}}\|\mathbf{q}_\perp\|_2^2}}. \tag{43}$$

To characterize the deviation from the first-order target, define

$$a = \frac{\|\mathbf{q}_\perp\|_2^2}{d_{\mathrm{out}}}. \tag{44}$$

Eq. (43) can then be written as

$$f(\mathbf{y}_\eta) = \left(1 + a\eta^2\right)^{-1/2}\left(f(\mathbf{y}) + \eta\mathbf{q}_\perp\right). \tag{45}$$

Using the Taylor expansion

$$\left(1 + a\eta^2\right)^{-1/2} = 1 - \frac{1}{2}a\eta^2 + \mathcal{O}(\eta^4), \qquad \eta \to 0, \tag{46}$$

gives

$$f(\mathbf{y}_\eta) = \left(1 - \frac{1}{2}a\eta^2 + \mathcal{O}(\eta^4)\right)(f(\mathbf{y}) + \eta\mathbf{q}_\perp)$$
$$= f(\mathbf{y}) + \eta\mathbf{q}_\perp - \frac{1}{2}a\eta^2 f(\mathbf{y}) + \mathcal{O}(\eta^3). \tag{47}$$

It follows that

$$f(\mathbf{y} + \eta\Delta\mathbf{y}_{\mathrm{REPA}}) - f(\mathbf{y}) - \eta\mathbf{q}_\perp = \mathcal{O}(\eta^2), \qquad \eta \to 0. \tag{48}$$

Taking Euclidean norms yields

$$\|f(\mathbf{y} + \eta\Delta\mathbf{y}_{\mathrm{REPA}}) - f(\mathbf{y}) - \eta\mathbf{q}_\perp\|_2 = \mathcal{O}(\eta^2). \tag{49}$$

Thus, Eq. (43) gives the exact finite-step normalized residual state. Relative to the projected first-order target, the leading nonlinear deviation is second order in the step magnitude. This characterization does not imply exact finite-step scale invariance. □

A.3 Proof of Theorem 3: Unique and Minimum-Norm Pullback

Proof. Consider the inverse problem of finding $\mathbf{v} \in \mathbb{R}^{d_{\mathrm{out}}}$ such that

$$\mathbf{J}_f(\mathbf{y})\mathbf{v} = \mathbf{q}_\perp, \tag{50}$$

where $\mathbf{q}_\perp \in \mathcal{T}_\mathbf{y}$ and $\mathbf{y} \neq \mathbf{0}$.

Every perturbation $\mathbf{v}$ admits the unique decomposition

$$\mathbf{v} = \mathbf{v}_\perp + \beta\mathbf{y}, \qquad \mathbf{y}^\top\mathbf{v}_\perp = 0, \tag{51}$$

where $\beta \in \mathbb{R}$.

From Eq. (31), the Jacobian acts on the orthogonal component as

$$\mathbf{J}_f(\mathbf{y})\mathbf{v}_\perp = \frac{1}{\rho_\epsilon(\mathbf{y})}\mathbf{v}_\perp. \tag{52}$$

For the residual-parallel direction,

$$\mathbf{J}_f(\mathbf{y})\mathbf{y} = \frac{1}{\rho_\epsilon(\mathbf{y})}\left(\mathbf{y} - \frac{\mathbf{y}\mathbf{y}^\top\mathbf{y}}{\|\mathbf{y}\|_2^2 + d_{\mathrm{out}}\epsilon}\right)$$
$$= \frac{1}{\rho_\epsilon(\mathbf{y})}\left(1 - \frac{\|\mathbf{y}\|_2^2}{\|\mathbf{y}\|_2^2 + d_{\mathrm{out}}\epsilon}\right)\mathbf{y}. \tag{53}$$

Since

$$\|\mathbf{y}\|_2^2 + d_{\mathrm{out}}\epsilon = d_{\mathrm{out}}\rho_\epsilon(\mathbf{y})^2, \tag{54}$$

the parallel response simplifies to

$$\mathbf{J}_f(\mathbf{y})\mathbf{y} = \frac{\epsilon}{\rho_\epsilon(\mathbf{y})^3}\mathbf{y}. \tag{55}$$

Combining Eqs. (51), (52), and (55) gives

$$\mathbf{J}_f(\mathbf{y})\mathbf{v} = \frac{1}{\rho_\epsilon(\mathbf{y})}\mathbf{v}_\perp + \beta\frac{\epsilon}{\rho_\epsilon(\mathbf{y})^3}\mathbf{y}. \tag{56}$$

Because the target $\mathbf{q}_\perp$ is orthogonal to $\mathbf{y}$, the orthogonal and parallel components in Eq. (56) must match independently.

Case 1: $\epsilon > 0$**.** The orthogonal component requires

$$\frac{1}{\rho_\epsilon(\mathbf{y})}\mathbf{v}_\perp = \mathbf{q}_\perp, \tag{57}$$

and therefore

$$\mathbf{v}_\perp = \rho_\epsilon(\mathbf{y})\mathbf{q}_\perp. \tag{58}$$

The parallel component must satisfy

$$\beta\frac{\epsilon}{\rho_\epsilon(\mathbf{y})^3}\mathbf{y} = \mathbf{0}. \tag{59}$$

Since $\epsilon > 0$, $\rho_\epsilon(\mathbf{y}) > 0$, and $\mathbf{y} \neq \mathbf{0}$, it follows that

$$\beta = 0. \tag{60}$$

Hence, for $\epsilon > 0$, the inverse problem has the unique solution

$$\mathbf{v} = \rho_\epsilon(\mathbf{y})\mathbf{q}_\perp = \Delta\mathbf{y}_{\mathrm{REPA}}. \tag{61}$$

Case 2: $\epsilon = 0$**.** Define

$$\rho_0(\mathbf{y}) = \sqrt{\frac{1}{d_{\mathrm{out}}}\|\mathbf{y}\|_2^2}. \tag{62}$$

When $\epsilon = 0$, Eq. (55) gives

$$\mathbf{J}_f(\mathbf{y})\mathbf{y} = \mathbf{0}, \tag{63}$$

so the residual-parallel direction lies in the null space of the Jacobian.

The orthogonal component remains uniquely determined:

$$\mathbf{v}_\perp = \rho_0(\mathbf{y})\mathbf{q}_\perp, \tag{64}$$

while β is unconstrained. Therefore, every feasible solution has the form

$$\mathbf{v} = \rho_0(\mathbf{y})\mathbf{q}_\perp + \beta\mathbf{y}, \qquad \beta \in \mathbb{R}. \tag{65}$$

Using $\mathbf{q}_\perp \perp \mathbf{y}$, the squared norm is

$$\begin{aligned}\|\mathbf{v}\|_2^2 &= \|\rho_0(\mathbf{y})\mathbf{q}_\perp + \beta\mathbf{y}\|_2^2 \\ &= \rho_0(\mathbf{y})^2\|\mathbf{q}_\perp\|_2^2 + \beta^2\|\mathbf{y}\|_2^2.\end{aligned} \tag{66}$$

Because $\mathbf{y} \neq \mathbf{0}$, the second term is nonnegative and vanishes if and only if $\beta = 0$. The unique minimum-norm solution is therefore

$$\mathbf{v}_{\min} = \rho_0(\mathbf{y})\mathbf{q}_\perp = \Delta\mathbf{y}_{\mathrm{REPA}}. \tag{67}$$

Thus, REPA is the unique residual-space preimage of $\mathbf{q}_\perp$ when $\epsilon > 0$. When $\epsilon = 0$, the Jacobian admits an affine family of feasible preimages differing only by residual-parallel null-space components, and REPA is the unique minimum-Euclidean-norm member. The pullback is therefore determined by the local RMSNorm geometry rather than by an arbitrary rescaling rule. □

B Extended Experimental Results

B.1 Hyperparameters and Evaluation Details

This section provides implementation and evaluation details for the experiments in Section 4.1.

Backbone and adaptation configuration. All experiments use Qwen2.5-1.5B-Instruct as the frozen pretrained backbone. Parameter-efficient adaptation is applied to the MLP `down_proj` modules of the final four Transformer blocks (layers 24–27). Unless otherwise specified, the main benchmark uses nominal rank $r = 8$ where applicable, with method-specific rank semantics following the corresponding implementations. All methods use the same adaptation scope.

Each run uses a fixed subset of 500 training examples, batch size 1, 32-bit floating-point (FP32) precision, and 10 epochs (5,000 optimizer steps). Optimization uses AdamW with gradient clipping at a maximum norm of 1.0. Task-specific sequence-length limits follow the corresponding benchmark protocols. Main-benchmark results are averaged over optimization seeds $\{2026, 2027, 2028\}$.

Table 5: Dataset-specific REPA configurations used in the main benchmark. All experiments use adapter rank $r = 8$; learning rate and α are selected using held-out development data.

Dataset	Learning Rate	α
SST-2	5×10^{-5}	16
RTE	5×10^{-5}	32
MRPC	7.5×10^{-5}	32
AG News	5×10^{-5}	16
PAWS	1×10^{-4}	16
MNLI	1×10^{-4}	32
WiC	7.5×10^{-5}	16

Hyperparameter configuration. Baseline methods use a common outer configuration with learning rate 5×10^{-5} and, where applicable, LoRA scaling coefficient $\alpha = 16$ and nominal rank $r = 8$, together with their method-specific operations. REPA uses dataset-specific learning rates and scaling coefficients selected exclusively on held-out development data, while the backbone, adaptation scope, rank, training-set size, and optimization budget remain fixed. Table 5 reports the resulting configurations.

For MNLI, candidate configurations are first screened on a validation split, after which the two highest-ranked configurations are compared on an independent development-selection split. The selected configuration is fixed before evaluation on the reported final split. Other benchmarks follow their corresponding development-selection protocols. No reported final-evaluation result is used to select the configuration evaluated on that split.

Data splits. Dataset construction is deterministic. Each formal run records its training and evaluation files, split seed, and corresponding data hashes. Development data are used for checkpoint selection for all methods and, for REPA, hyperparameter selection. The reported final-evaluation data are accessed only after the corresponding configuration and checkpoint have been selected. The data-preparation procedures enforce disjoint source examples across training, development, and final-evaluation partitions.

Checkpoint selection. Within each run, checkpoints are selected by development accuracy, with development loss and then the earlier checkpoint used as deterministic tie-breakers when necessary. Final accuracy is evaluated only at the development-selected checkpoint.

Evaluation protocol. Each benchmark is formulated as fixed-label conditional scoring. For each candidate label, conditional token log-probabilities of a fixed verbalizer are summed, and the label with the highest score is selected. Within each task, candidate verbalizers have matched token lengths under the Qwen2.5 tokenizer. For each dataset and method, the reported result is the mean final-evaluation accuracy over three optimization seeds, and the macro-average assigns equal weight to all seven benchmarks.

Reproducibility. Each formal run is defined by an explicit configuration specifying the dataset, adaptation method, optimization seed, learning rate, adapter rank, scaling coefficient, target layers, target modules, training budget, sequence-length limit, numerical precision, and data files. The formal run format records the corresponding training history, checkpoint-selection outcome, evaluation result, data provenance, and code-version metadata. Additional reproducibility details appear in Appendix B.2.

B.2 Reproducibility Details

Formal experiments are defined by explicit run configurations recording the dataset, adaptation method, optimization seed, learning rate, adapter rank, scaling coefficient, target layers and modules, training budget, sequence-length limit, numerical precision, and data files. The formal run format records training histories, checkpoint-selection outcomes, evaluation records, data-file identities

Table 6: Full rank-sensitivity results for REPA on AG News. Test accuracy (%) is reported for five optimization seeds together with the mean and sample standard deviation (SD). The scaling coefficient satisfies $\alpha = r$ at every rank, keeping $\alpha/r = 1$ fixed throughout the sweep. The reference rank $r = 8$ is shown in bold.

Rank r	Params.	2026	2027	2028	2029	2030	Mean $\pm$ SD
1	41,984	88.0	90.0	87.0	87.0	87.0	87.8 ± 1.30
2	83,968	88.0	86.0	89.0	89.0	88.0	88.0 ± 1.22
4	167,936	87.0	87.0	89.0	88.0	84.0	87.0 ± 1.87
8	**335,872**	**89.0**	**90.0**	**85.0**	**88.0**	**89.0**	$\mathbf{88.2 \pm 1.92}$
16	671,744	89.0	89.0	87.0	88.0	87.0	88.0 ± 1.00
32	1,343,488	89.0	89.0	87.0	85.0	86.0	87.2 ± 1.79
64	2,686,976	90.0	86.0	88.0	87.0	90.0	88.2 ± 1.79

and hashes, and code-version metadata, linking reported results to their corresponding experimental configurations.

For LoRA-Dash, the published two-stage formulation is reproduced through a compatibility implementation adapted to the Qwen2.5 `down_proj` interface. The implementation uses a 100-step prelaunch phase and eight selected task-specific directions, with the reference code commit recorded in the run metadata.

The main benchmark uses optimization seeds $\{2026, 2027, 2028\}$ with Python 3.10.20, PyTorch 2.5.1 with CUDA 12.1, and FP32 precision unless otherwise specified. The seven-interface residual-RMS probe in Appendix B.4 is a separate mechanistic experiment using independently retrained checkpoints for optimization seed 2026 under the same nominal adapter configuration. The probe uses PyTorch 2.14.0+cu130 and Transformers 4.57.6 on an NVIDIA RTX PRO 6000 Blackwell Server Edition. These measurements are analyzed independently and are not combined with other layer-wise measurements or the controlled-intervention artifacts.

B.3 Performance Across Rank Budgets

This section reports the complete rank-budget results corresponding to Section 4.4. REPA is evaluated on AG News with $r \in \{1, 2, 4, 8, 16, 32, 64\}$ across five optimization seeds $\{2026, 2027, 2028, 2029, 2030\}$. All configurations use Qwen2.5-1.5B-Instruct as the frozen backbone and adapt the `down_proj` modules of the final four Transformer blocks. The learning rate is fixed at 7.5×10^{-5}, and the LoRA scaling coefficient is set to $\alpha = r$ at each rank, keeping $\alpha/r = 1$ throughout the sweep.

Each configuration uses 500 training examples, batch size 1, maximum sequence length 512, FP32 precision, and 5,000 optimizer steps. Checkpoints are selected by validation accuracy, with validation loss and then the earlier checkpoint used as deterministic tie-breakers. The held-out test split is evaluated only at the validation-selected checkpoint.

Table 6 shows that mean test accuracy remains between 87.0% and 88.2% across a 64-fold range in trainable adapter parameters. At $r = 1$, REPA uses 87.5% fewer trainable parameters than at $r = 8$, with only a 0.4-percentage-point reduction in mean test accuracy. At $r = 2$, the parameter count is reduced by 75%, while mean accuracy remains within 0.2 percentage points of the $r = 8$ result.

Increasing rank beyond $r = 8$ does not yield a consistent improvement. The $r = 64$ configuration uses eight times as many trainable parameters as $r = 8$, yet both achieve the same mean test accuracy of 88.2%. Thus, REPA remains effective across a broad range of rank budgets, including strongly constrained low-rank settings. This experiment characterizes within-REPA rank sensitivity rather than cross-method rank efficiency.

B.4 Mechanistic Analysis Details

This section provides the experimental details underlying the mechanistic analysis in Section 4.5. The analysis examines residual-scale variation across consecutive residual-output interfaces and the response to controlled residual rescaling predicted by the local RMSNorm geometry.

Residual-state collection. Residual states are collected from Qwen2.5-1.5B-Instruct on SST-2, RTE, and QNLI using seed 2026. The layer-wise analysis covers seven consecutive residual-output interfaces (layers 21–27). Layers 24–27 are adapted `down_proj` interfaces, whereas layers 21–23 are probe-only interfaces with no adapter attached at those layers. The adapted configuration uses rank $r = 8$, scaling coefficient $\alpha = 16$, maximum sequence length 512, and FP32 precision. For each dataset, 200 validation examples are processed, yielding 600 example-level observations per layer.

For residual-RMS statistics, RMS values are first computed over valid tokens within each example–layer pair and then averaged within that pair. Layer-level means and standard deviations are computed over these example-level values, with 95% confidence intervals estimated using 10,000 bootstrap replicates over example–layer units.

Residual-scale variation. The mean residual RMS values for layers 21–27 are 5.768, 6.276, 6.945, 7.760, 8.535, 6.588, and 6.329, respectively. The corresponding coefficients of variation are 17.17%, 15.64%, 13.82%, 11.83%, 10.38%, 2.23%, and 4.66%. The 95% bootstrap confidence intervals are $[5.691, 5.847]$, $[6.199, 6.355]$, $[6.867, 7.023]$, $[7.686, 7.834]$, $[8.464, 8.605]$, $[6.577, 6.600]$, and $[6.306, 6.352]$, respectively.

Residual scale therefore varies substantially across nearby residual-output interfaces, increasing from layers 21 through 25 before dropping at layers 26–27. This provides the empirical setting for the residual-scale mismatch in Section 3.1.

Controlled residual-scale intervention. The controlled intervention is analyzed separately from the seven-interface RMS probe and uses recorded residual–update pairs $(\mathbf{y}, \mathbf{q})$ from the adapted interfaces; measurements from the two analyses are not combined. A reservoir of 1,000 token-state pairs is used, corresponding to 802 unique validation example–layer groups.

For each recorded pair, the residual state is rescaled as

$$\mathbf{y}_c = c\mathbf{y}, \qquad c \in \{0.25, 0.5, 1, 2, 4\}, \tag{68}$$

while the candidate LoRA update $\mathbf{q}$ is held fixed. For each update rule, let $\Delta\mathbf{y}_c$ denote the resulting residual-space perturbation. The normalized-space response is

$$M_c(\mathbf{q}) = \|f(\mathbf{y}_c + \Delta\mathbf{y}_c) - f(\mathbf{y}_c)\|_2\,, \tag{69}$$

and responses are reported relative to $M_1(\mathbf{q})$.

Three residual-update rules are evaluated: direct LoRA injection, projection only, and the full REPA transformation. Statistics are aggregated over validation example–layer units, with 10,000 bootstrap replicates used to estimate 95% confidence intervals.

LoRA closely follows the predicted inverse-scale relation, with normalized responses of approximately 3.999, 2.000, 1.000, 0.500, and 0.250 for $c = \{0.25, 0.5, 1, 2, 4\}$. Projection only exhibits essentially the same trend, showing that orthogonalization alone does not remove the RMS-dependent scalar modulation. In contrast, REPA maintains $M_c(\mathbf{q})/M_1(\mathbf{q}) \approx 1$ across the intervention range, with a maximum deviation of approximately 1.21×10^{-8} from the scale-invariant prediction.

The intervention isolates the role of the RMS pullback: projection restricts the update to the subspace on which the RMSNorm Jacobian has a scalar response, while residual-aware RMS scaling removes the resulting inverse dependence on residual scale.

Residual-parallel update energy. The role of projection is further characterized by the fraction of LoRA update energy aligned with the residual direction,

$$R_{\parallel} = \frac{\|\mathbf{q}_{\parallel}\|_2^2}{\|\mathbf{q}\|_2^2}. \tag{70}$$

Token-level ratios are first averaged within each validation example–layer pair and then aggregated across examples. For adapted layers 24–27, the mean residual-parallel energy ratios are 1.49×10^{-3}, 1.19×10^{-3}, 1.18×10^{-3}, and 1.14×10^{-3}, respectively. Thus, only approximately 0.11–0.15% of the learned LoRA update energy lies in the residual-parallel direction.

The learned updates are therefore already predominantly orthogonal to the residual direction, consistent with the limited downstream change from projection alone in the component analysis. Projection nevertheless remains structurally important because it places the update in the subspace where the RMSNorm Jacobian reduces to a scalar operator and the closed-form pullback applies.

Overall, the mechanism analyses provide complementary evidence: the layer-wise probe establishes substantial residual-scale heterogeneity, while the controlled intervention recovers the predicted inverse-scale response for LoRA and projection only. REPA removes this dependence under controlled residual rescaling.

B.5 Component Analysis

This section separates the roles of residual-orthogonal projection and RMS-based scaling in REPA. The analysis uses a dedicated three-task subset consisting of SST-2, RTE, and QNLI. All variants share the frozen backbone, adaptation scope, adapter rank, LoRA scaling configuration, optimization protocol, data partitions, and validation-based checkpoint-selection rule, differing only in the transformation applied to the candidate low-rank update before residual addition.

Let

$$\mathbf{q} = s\mathbf{B}\mathbf{A}\mathbf{z}, \tag{71}$$

where $\mathbf{z}$ is the adapter input, $\mathbf{A}$ and $\mathbf{B}$ are the LoRA factors, and s is the effective LoRA scaling factor. The residual-orthogonal component is

$$\mathbf{q}_\perp = \mathbf{q} - \frac{\mathbf{y}^\top \mathbf{q}}{\|\mathbf{y}\|_2^2}\mathbf{y}. \tag{72}$$

Raw update. The candidate LoRA update is injected directly into the residual stream:

$$\Delta\mathbf{y} = \mathbf{q}. \tag{73}$$

This variant contains neither residual-orthogonal projection nor RMS-based scaling and serves as the uncorrected LoRA baseline.

Projection only. The candidate update is projected onto the residual-orthogonal subspace:

$$\Delta\mathbf{y} = \mathbf{q}_\perp. \tag{74}$$

This isolates the geometric projection while retaining the residual-scale dependence induced by RMSNorm.

RMS only. The unprojected candidate update is scaled by the current residual RMS:

$$\Delta\mathbf{y} = \rho_\epsilon(\mathbf{y})\mathbf{q}. \tag{75}$$

This isolates state-dependent RMS scaling without restricting the update to the residual-orthogonal subspace on which the RMSNorm Jacobian has the scalar response used in the REPA derivation.

Static RMS projection. Projection is retained, but the current residual RMS is replaced by a layer-specific fixed calibration value $\bar{\rho}_\ell$:

$$\Delta\mathbf{y} = \bar{\rho}_\ell\mathbf{q}_\perp. \tag{76}$$

For each adapted layer ℓ, $\bar{\rho}_\ell$ is calibrated before training using the first 32 examples of the fixed training subset, without validation or held-out test data. Calibration uses the pristine frozen backbone, excludes padding positions, and aggregates token-wise RMS values within each adapted layer. The resulting values remain fixed throughout training, after which the model state is reset before adapter injection. Under a fixed random seed, all variants use identical adapter initialization.

Full REPA. The complete method combines residual-orthogonal projection with the dynamic RMS pullback:

$$\Delta\mathbf{y}_{\mathrm{REPA}} = \rho_\epsilon(\mathbf{y})\mathbf{q}_\perp. \tag{77}$$

Unlike the static variant, the scaling factor is recomputed from the current residual state and can vary across examples, tokens, layers, and training states.

Table 7: Component analysis of residual-orthogonal projection and RMS-based scaling on SST-2, RTE, and QNLI. The final column reports the macro-average best validation accuracy across the three datasets.

Variant	Projection	RMS scaling	Avg. Val. Acc. (%)
Raw update	–	–	87.11
Projection only	✓	–	87.11
RMS only	–	Dynamic	87.44
Static RMS projection	✓	Static	87.44
Full REPA	✓	Dynamic	87.44

Results and interpretation. Table 7 shows that the raw-update and projection-only variants both obtain 87.11% macro-average validation accuracy, whereas RMS only, static RMS projection, and full REPA each reach 87.44%. Within this subset, the observed accuracy difference is therefore associated with RMS-based scaling rather than projection alone.

Projection nevertheless remains structurally important: it removes the residual-parallel component and restricts the update to the subspace on which the RMSNorm Jacobian has the scalar response required by the closed-form pullback. Static RMS projection matches full REPA in macro-average accuracy on this subset. Separately, the controlled intervention in Appendix B.4 shows that full REPA maintains its scale-consistent response as residual scale varies.

The component and mechanism analyses provide complementary evidence: RMS-based scaling is associated with the observed downstream accuracy difference, while projection supplies the geometric subspace required by the analytically derived REPA pullback.

B.6 Dataset-Level Time-to-Target Results

This section provides the target-sensitivity and dataset-level breakdown of the time-to-target (TTA) analysis in Section 4.3. The analysis uses the same dedicated seven-task optimization set and LoRA-defined validation target,

$$\tau_\gamma = a_0 + \gamma \left(a_{\text{best}}^{\text{LoRA}} - a_0\right), \qquad \gamma \in [0, 1], \tag{78}$$

where a_0 is the pre-adaptation validation accuracy and $a_{\text{best}}^{\text{LoRA}}$ is the best validation accuracy attained by LoRA.

For each dataset–seed pair, TTA is defined as the cumulative training time at the first observed validation checkpoint reaching the corresponding target. The paired TTA reduction of REPA relative to LoRA is

$$\Delta_{\text{TTA}} = 100 \times \frac{T_{\text{LoRA}} - T_{\text{REPA}}}{T_{\text{LoRA}}}, \tag{79}$$

where positive values indicate earlier target attainment by REPA. A dataset–seed pair is included at a given target fraction only when both methods reach the same target.

Figure 3 shows that target-attainment behavior depends on both dataset and target stringency. Across $\gamma \in \{0.75, 0.80, 0.85, 0.90\}$, the aggregate median paired reduction remains between approximately -0.03% and 0.48%, indicating near parity at loose and intermediate targets. Larger aggregate median differences appear at the most stringent targets.

At $\gamma = 0.95$, REPA reaches the target in 20 of the 21 dataset–seed runs. Among the 20 comparable pairs, REPA is earlier in 13, with a median paired TTA reduction of 18.81%; the corresponding 95% confidence interval reported in Table 4 includes zero. At $\gamma = 1.00$, REPA reaches the target in 19 of the 21 runs and is earlier than LoRA in 12 of the 19 comparable pairs, with a median paired reduction of 16.79%.

The dataset-level results exhibit substantial heterogeneity. At $\gamma = 0.95$, the median paired reductions are approximately $+23.4\%$ on SST-2, -204.0% on RTE, $+0.1\%$ on QNLI, -10.8% on MRPC, $+71.5\%$ on QQP, $+0.8\%$ on AG News, and $+14.3\%$ on PAWS. For RTE, only two of the three optimization seeds are comparable at $\gamma = 0.95$, so the reported median is conditional on those target-reaching pairs.

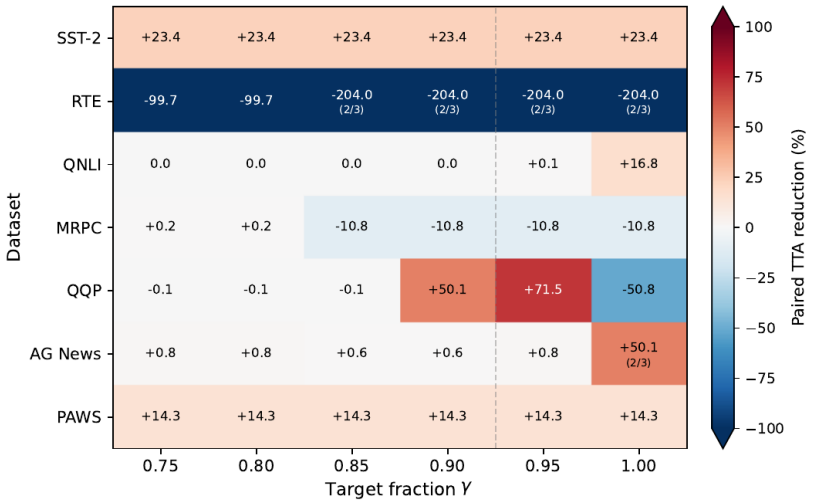

Figure 3: Dataset-level paired time-to-target (TTA) reduction of REPA relative to LoRA across target fractions $\gamma \in \{0.75, 0.80, 0.85, 0.90, 0.95, 1.00\}$. Rows denote datasets and columns denote target fractions. Each cell reports the median paired TTA reduction over comparable optimization seeds. Positive values favor REPA, negative values favor LoRA, and values near zero indicate similar target-attainment time. Colors are clipped to $[-100\%, 100\%]$ for visualization, whereas annotations retain the exact median reductions. Parenthesized values report the number of comparable seeds when fewer than three are available. The dashed vertical line marks the near-peak target $\gamma = 0.95$ used in the main comparison.

At $\gamma = 1.00$, SST-2, QNLI, AG News, and PAWS have positive paired reductions, whereas RTE, MRPC, and QQP have negative reductions. The QQP comparison changes sign between $\gamma = 0.95$ and $\gamma = 1.00$, illustrating the dependence of relative target-attainment efficiency on the chosen performance threshold. AG News has only two comparable seeds at $\gamma = 1.00$.

These results do not support uniformly faster convergence throughout training. Instead, larger aggregate median differences occur near stringent targets, while substantial task-level heterogeneity remains. Together with the multi-method analysis in Section 4.3, the results support competitive near-peak target attainment rather than general acceleration of the full optimization trajectory.