THE PATH MATTERS: RETENTION-BUDGET CONTINUATION FOR LOW-RANK LLM POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-rank adaptation (LoRA) enables efficient specialization of large language models, but low rank alone does not ensure capability retention. How the optimization path shapes specialization under a fixed behavioral allowance and finite computation remains unclear. We introduce Retention-Budget Continuation for LoRA (RBC-LoRA), a homotopy method that constrains forward sequence Kullback-Leibler divergence from the original model on reference prompts. A temporary quadratic factor anchor supplies a unique starting solution; continuation then strengthens the task objective, removes the anchor, and expands the retention allowance. First-order primal-dual correction and empirical acceptance checks guide progression toward the prescribed constrained endpoint while preserving adapter rank. Under local regularity and correction assumptions, we characterize solution-path sensitivity and tracking error; a divergence bound also controls changes in bounded expected scores on the reference distribution. Experiments show that RBC-LoRA improves target-task scores over independently tuned direct constrained training on math and code tasks. These gains accompany less measured degradation of pretrained behaviour, reflected in lower audit divergence and smaller retained-benchmark losses. These findings establish the optimization path as a practical design choice for improving the specialization–retention tradeoff without increasing adapter capacity.

1 INTRODUCTION

Large language models (LLMs) are used for tasks ranging from code generation to math reasoning, yet adapting one to a new task by updating every weight is costly at the scale of modern models (Ouyang et al., 2022). Parameter-efficient fine-tuning (PEFT) reduces this burden by training only a small part of the model. Low-rank adaptation (LoRA) (Hu et al., 2021) freezes the pretrained weights and learns each selected update as the product of two low-rank matrices. This makes task-specific training and storage much cheaper than full fine-tuning.

Efficiency alone does not protect the behavior learned before adaptation. LoRA can forget less than full fine-tuning on math and coding tasks while also learning less (Biderman et al., 2024). Spectral evidence links some LoRA updates to forgetting (Shuttleworth et al., 2026). Low rank restricts update structure without bounding its magnitude or behavioral effect. Existing retention methods penalize overlap with protected subspaces or changes in sensitive parameter directions (Lu et al., 2025; Sliwa et al., 2025). These penalties can reduce forgetting, but they do not prescribe a behavioral allowance relative to the original model. An explicit output-divergence constraint supplies that allowance, yet direct optimization can still produce a weak target-task solution within a finite training budget (Figure 1).

The training trajectory may explain this remaining gap. With the full task objective active from the first step, early updates can favor directions that later become hard to revise near the retention boundary. This proposed mechanism concerns optimization under finite computation. Prior work has shaped trajectories through source-sensitive gradient preconditioning (Michel et al., 2019) and progressive adapter activation (Zhuang et al., 2025). Homotopy methods trace a path of related optimization problems, starting from an easier one (Lin et al., 2023; Li & Li, 2024). We hypothesize

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

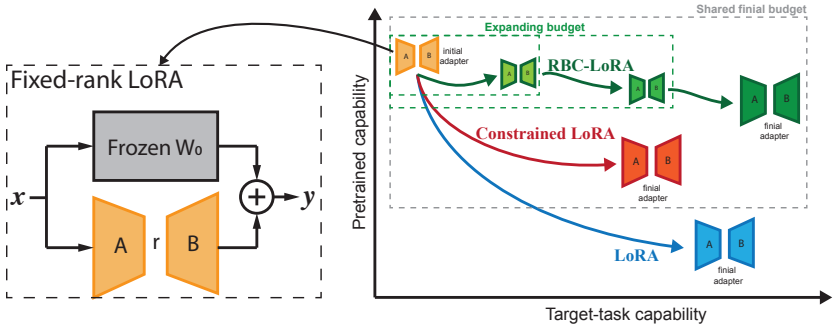

Figure 1: Schematic motivation for RBC-LoRA. Ordinary LoRA can improve target-task capability while degrading pretrained capability (blue). Directly constrained LoRA limits this degradation but may yield smaller target-task gains under finite computation (red). RBC-LoRA gradually increases task pressure and the retention allowance along a homotopy path (green), seeking higher target-task capability at the same final retention budget.

that such a path can reach a better solution at the same adapter rank, reference prompts, and final retention allowance than direct constrained training under comparable computation (Figure 1).

We introduce *Retention-Budget Continuation for LoRA* (RBC-LoRA), which approaches a fixed retention-constrained endpoint through nearby problems. A temporary quadratic factor anchor supplies a unique start, while primal-dual correction and empirical checks guide rising task pressure and a widening forward sequence Kullback-Leibler (KL) allowance relative to the original policy on reference prompts as the anchor vanishes. Our analysis establishes the regular start, conditional local path-sensitivity and tracking results, and a bound on changes in bounded expected scores under the reference distribution. In math and code tasks, RBC-LoRA attains higher target scores than tuned direct constrained training. These gains accompany less measured degradation of pretrained behaviour, reflected in lower audit divergence and smaller losses on retained benchmarks.

Our contributions are:

- *Methodologically*, we develop fixed-rank retention-budget continuation with a uniquely anchored start, adaptive correction, and the same final constrained problem as direct training.
- *Theoretically*, we characterize local solution paths and conditional tracking error, and relate the KL allowance to bounded expected scores on the reference distribution.
- *Empirically*, we compare RBC-LoRA with tuned direct constrained training under a common GPU-time cap and find higher target scores in math and code tasks, accompanied by lower audit divergence and smaller measured losses on retained benchmarks.

2 Related Work

PEFT and LoRA. PEFT methods differ in what they train. Adapter tuning adds small modules to a pretrained network (Houlsby et al., 2019), whereas prefix tuning learns continuous prefixes used during generation (Li & Liang, 2021). LoRA (Hu et al., 2021) keeps the backbone frozen. It represents selected weight updates as low-rank matrix products, reducing training and storage costs while allowing the update to be merged into the original weights at inference. Optimization remains consequential. Learning-rate selection can strongly affect standard LoRA (Lee et al., 2026), and comparisons with full fine-tuning show that LoRA can learn less while also forgetting less (Biderman et al., 2024). Spectral analyses associate some LoRA update directions with forgetting (Shuttleworth et al., 2026).

Retention and training-path baselines. LoRA baselines protect retained behavior or reshape training. CLoRA penalizes overlap between LoRA factors and predefined subspaces (Lu et al., 2025), while LaLoRA uses a Laplace approximation to protect directions estimated to matter for retained behavior (Sliwa et al., 2025). CoTo changes the training path. It gradually increases adapter

activation, providing a progressive-training baseline (Zhuang et al., 2025). The first two methods protect parameter directions, whereas CoTo changes the schedule of adapter participation. None sets an explicit allowance for behavioral change relative to the original policy. RBC-LoRA varies such an allowance while keeping adapter rank and the final constraint fixed.

Homotopy continuation. Homotopy continuation approaches a difficult target problem through a sequence of simpler intermediate problems. Chen & Hao (2019) follows the resulting solution path while progressively adding layers and nodes to a fully connected network. Li & Li (2024) constructs a regular starting problem for fully connected networks and uses predictor-corrector steps to trace its deformation. Continuation path learning learns a model that returns solutions across homotopy levels (Lin et al., 2023). Their guarantees concern those particular homotopies. RBC-LoRA applies continuation to fixed-rank adaptation with a changing retention allowance.

3 METHOD

Directly imposing the final retention allowance leaves the training path to a nonconvex solver, which may reach a weak task solution under finite computation. Motivated by homotopy continuation (Lin et al., 2023; Li & Li, 2024), Retention-Budget Continuation for LoRA (RBC-LoRA) starts from a solved factor-space problem and changes both task pressure and the permitted divergence from the original model. The fixed-rank endpoint is specified first, followed by the continuation family and the correction rule that advances between its members. Algorithm 1 gives the full procedure.

3.1 PRELIMINARIES

Let π_0 be the pretrained policy. Its backbone stays frozen, and each selected weight matrix receives a LoRA update of rank at most r_ℓ,

$$W_\ell = W_{0,\ell} + B_\ell A_\ell, \qquad B_\ell \in \mathbb{R}^{d_{\mathrm{out},\ell} \times r_\ell}, \quad A_\ell \in \mathbb{R}^{r_\ell \times d_{\mathrm{in},\ell}}. \tag{1}$$

The ranks, adapted matrices, and LoRA scaling are fixed during training. Let θ collect all factors and θ_0 their initialization, chosen so that $B_{\ell,0} = 0$ and $\pi_{\theta_0} = \pi_0$. Appendix A.1 specifies the factor convention used by the anchor.

For target prompt-response pairs $(x, y) \sim \mathcal{D}_{\mathrm{new}}$, the supervised objective averages token negative log-likelihood within each response and then across examples:

$$L_{\mathrm{new}}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\mathrm{new}}} \left[-\frac{1}{|y|} \log \pi_\theta(y \mid x) \right]. \tag{2}$$

Rank limits the update's structure, but gives no behavioral bound. For reference prompts $x \sim \mathcal{D}_{\mathrm{anchor}}$, define $\pi_0(\cdot \mid x)$ and $\pi_\theta(\cdot \mid x)$ on a common response horizon T. The retention cost and the final problem are

$$c(\theta) = \mathbb{E}_{x\sim\mathcal{D}_{\mathrm{anchor}}} D_{\mathrm{KL}}(\pi_0(\cdot \mid x) \,\|\, \pi_\theta(\cdot \mid x)), \tag{3}$$

$$\min_\theta \quad L_{\mathrm{new}}(\theta) \quad \text{subject to} \quad c(\theta) \le \epsilon_f, \qquad \epsilon_f > 0. \tag{4}$$

The reference in c remains π_0 at every stage, so ϵ_f measures divergence from the original policy rather than from an earlier checkpoint. We use forward sequence KL with full softmax probabilities at temperature one. Its estimator, response termination convention, and separate training, controller, and final-audit samples appear in Appendix A.2. Controller feasibility is empirical. Section 4.4 states the population interpretation.

3.2 ANCHORED RETENTION-BUDGET CONTINUATION

The endpoint in Eq. (4) couples a nonconvex task loss to a behavioral constraint, and the LoRA factors have redundant parameterizations. These features call for a factor-space anchor at the start and empirical checks during progression. The implementation follows a discrete sequence of empirical subproblems. Smooth solution branches enter only the local analysis. For $\lambda \in [0, 1]$, RBC-LoRA

defines

$$F_\lambda(\theta) = \lambda L_{\text{new}}(\theta) + \frac{\gamma(1-\lambda)}{2}\|\theta - \theta_0\|_2^2, \qquad \gamma > 0, \tag{5}$$

$$\epsilon(\lambda) = \epsilon_0 + \lambda(\epsilon_f - \epsilon_0), \qquad 0 < \epsilon_0 < \epsilon_f, \tag{6}$$

$$\min_\theta \quad F_\lambda(\theta) \quad \text{subject to} \quad c(\theta) \le \epsilon(\lambda). \tag{7}$$

The factor anchor makes θ_0 the unique minimizer at $\lambda = 0$, with Hessian γI. Since $c(\theta_0) = 0 < \epsilon_0$, this start lies strictly inside the initial allowance, away from the KL boundary where $\nabla c(\theta_0) = 0$. The anchor acts on the factors rather than only on BA: a product anchor would leave many zero-update factorizations. These choices supply a regular starting problem in the coordinates actually optimized.

Increasing λ strengthens the task term and relaxes the temporary anchor while the allowance grows from ϵ_0 to ϵ_f. For $\lambda > 0$, dividing F_λ by λ leaves its minimizers unchanged and gives the anchor coefficient $\gamma(1-\lambda)/(2\lambda)$. The two schedules serve separate roles: fixing the allowance at ϵ_f would anneal the objective alone, whereas fixing $F_\lambda = L_{\text{new}}$ would expose the optimizer to full task pressure from the first step. At $\lambda = 1$, the anchor is gone and Eq. (7) is exactly Eq. (4), with the same rank and feasible set as direct constrained training. The construction targets endpoint quality under finite computation. The exact endpoint optimum is unchanged. Section 4 analyzes the start and local paths under explicit regularity assumptions.

3.3 Correction, acceptance, and progression

The family specifies which problems to solve, but finite stochastic updates do not automatically follow their solutions or satisfy the changing constraint. Starting from an accepted state $(\lambda_k, \theta_k, \nu_k)$, propose $\lambda^+ = \min(1, \lambda_k + h_k)$ and warm-start the factors and nonnegative multiplier. A capped secant predictor can replace the factor warm start after compatible accepted stages (Appendix A.4). At the proposed level, correction uses the empirical Lagrangian

$$\widehat{\mathcal{L}}_\lambda(\theta, \nu) = \lambda \widehat{L}_{\text{new}}(\theta) + \frac{\gamma(1-\lambda)}{2}\|\theta - \theta_0\|_2^2 + \nu\big(\widehat{c}_{\text{tr}}(\theta) - \epsilon(\lambda)\big). \tag{8}$$

The factors take first-order steps using $\nabla_\theta \widehat{\mathcal{L}}_\lambda$, while the multiplier follows

$$\nu_{j+1} = \big[\nu_j + \eta_\nu\big(\widehat{c}_{\text{tr}}(\theta_j) - \epsilon(\lambda)\big)\big]_+ . \tag{9}$$

Here $[a]_+ = \max(a, 0)$, j indexes correction steps, and $\widehat{c}_{\text{tr}}$ uses the reference training cache. Exceeding the current allowance increases ν and thus the retention gradient in the next factor step. The objective is implemented at the scaling in Eq. (5). The factor optimizer and state handling are specified in Appendix A.5.

Progression depends on measured feasibility and correction quality. Two successive diagnostic checks must find both the training and separate controller costs at most $(1 + \tau_{\text{feas}})\epsilon(\lambda^+)$, with normalized Karush–Kuhn–Tucker (KKT) stationarity and complementarity residuals at most τ_{stat} and τ_{comp}. All checked quantities must be finite. The controller samples are separate from the training cache because acceptance should be tested on reference prompts that did not supply the factor gradients. Normalized residuals avoid treating a small raw gradient at low λ as sufficient correction merely because the task term is scaled down. Appendix A.3 defines the checks and their fixed data subsets.

Each proposal is bounded by an update cap and the remaining charged-work budget. On failure, training restores the last accepted factors, multiplier, and optimizer state, halves the increment, and retries at a closer level; rejected work remains charged. Two accepted stages requiring at most half the update cap each permit a larger increment. Acceptance applies to checkpoints, so intermediate minibatch iterates may exceed the allowance. After an accepted proposal reaches $\lambda = 1$, a separate correction uses the reserved work with zero anchor coefficient. Training records progression and endpoint-correction failures separately, then locks the returned checkpoint for an independent audit against ϵ_f without the controller tolerance. Appendix A.5 specifies the caps, rollback, and returned status.

4 THEORETICAL ANALYSIS

We analyze the solution path and retention guarantees of Eq. (7) using exact losses in fixed LoRA factor coordinates. The results characterize the anchored start, sensitivity and tracking along regular path segments, stationarity at the prescribed endpoint, and behavioral change under the KL budget. Assumptions and proofs appear in Appendix B.

4.1 REGULAR START AND INITIAL DRIFT

Set $g_0 = \nabla L_{\text{new}}(\theta_0)$ and $\mathcal{I}_0 = \nabla^2 c(\theta_0)$, the reference-policy Fisher information in factor coordinates.

Proposition 1 (Regular start and local departure) *At $\lambda = 0$, θ_0 uniquely solves Eq. (7), its constraint is inactive, and $\nabla^2 F_0 = \gamma I$. If L_{new} and c are C^2 near θ_0, a locally unique C^1 branch of strict local minimizers satisfies*

$$\theta^{*\prime}(0) = -g_0/\gamma, \tag{10}$$

$$L_{\text{new}}(\theta^*(\lambda)) = L_{\text{new}}(\theta_0) - \frac{\lambda}{\gamma}\|g_0\|_2^2 + o(\lambda),$$

$$c(\theta^*(\lambda)) = \frac{\lambda^2}{2\gamma^2} g_0^\top \mathcal{I}_0 g_0 + o(\lambda^2). \tag{11}$$

The factor anchor supplies curvature at θ_0, including directions in which the reference KL is flat. When $g_0 \neq 0$, the initial task improvement is first order in λ, while reference divergence is $O(\lambda^2)$. The two rates characterize the controlled departure from the pretrained policy.

4.2 SENSITIVITY ON AN ACTIVE PATH

At an active stage, the exact Lagrangian is $\mathcal{L}_\lambda = F_\lambda + \nu[c - \epsilon(\lambda)]$. Its Karush–Kuhn–Tucker (KKT) conditions are

$$\nabla F_\lambda + \nu \nabla c = 0, \qquad c \le \epsilon(\lambda), \quad \nu \ge 0, \quad \nu[c - \epsilon(\lambda)] = 0. \tag{12}$$

Theorem 2 (Local active-path sensitivity) *At an active KKT point with $\lambda \in (0, 1)$, suppose $\nu > 0$, $a = \nabla c(\theta) \neq 0$, and $H = \nabla^2 F_\lambda(\theta) + \nu \nabla^2 c(\theta)$ is positive definite on $\{d : a^\top d = 0\}$. A locally unique C^1 branch of strict local minimizers and multipliers then satisfies*

$$\underbrace{\begin{bmatrix} H & a \\ a^\top & 0 \end{bmatrix}}_{K} \begin{bmatrix} \theta^{*\prime}(\lambda) \\ \nu^{*\prime}(\lambda) \end{bmatrix} = \begin{bmatrix} -\nabla L_{\text{new}}(\theta^*) + \gamma(\theta^* - \theta_0) \\ \epsilon_f - \epsilon_0 \end{bmatrix}. \tag{13}$$

The two terms on the right separate objective deformation from budget expansion. The inverse KKT matrix converts them into factor and multiplier motion; its norm bounds local sensitivity (Appendix B.3). This relation gives a basis for adjusting the continuation increment when correction becomes difficult.

4.3 PREDICTION, CORRECTION, AND ENDPOINT

For an exact active solution and tangent $v = \theta^{*\prime}(\lambda)$, let $h > 0$ be small enough to remain in a neighborhood where ∇c is L_c-Lipschitz. Then

$$[c(\theta^*(\lambda) + hv) - \epsilon(\lambda + h)]_+ \le \frac{L_c}{2} h^2 \|v\|_2^2. \tag{14}$$

The tangent matches the affine budget change to first order, leaving at most a quadratic feasibility residual for correction.

Theorem 3 (Local tracking under correction) *On a C^2 regular solution branch $z^*(\lambda) = (\theta^*(\lambda), \nu^*(\lambda))$ with $\|z^{*\prime\prime}\|_2 \le M$, suppose a tangent field agrees with $z^{*\prime}$ on the branch and is L_v-Lipschitz. If the predictor direction has error at most δ_k relative to that field and the next-stage*

corrector reduces distance to the branch by $q < 1$ with additive error ξ_k, then, for steps in these neighborhoods, $e_k = \|z_k - z^*(\lambda_k)\|_2$ obeys

$$e_{k+1} \le q\left[(1 + h_k L_v)e_k + h_k\delta_k + \frac{M}{2}h_k^2\right] + \xi_k. \tag{15}$$

When $q(1+h_kL_v)$ is uniformly below one and the per-step errors are bounded, the recurrence yields a geometric transient and a bounded tracking-error floor. It quantifies how increment size, prediction error, and correction accuracy jointly control adherence to a regular path (Appendix B.5).

Proposition 4 (Endpoint stationarity) *For a fixed smooth problem, suppose $\lambda_k \to 1$, (θ_k, ν_k) remains bounded with $\nu_k \ge 0$, and raw stationarity, feasibility-violation, and complementarity residuals tend to zero. Every accumulation point satisfies the KKT conditions of Eq. (4).*

Under these conditions, the continuation sequence has stationary feasible accumulation points for the prescribed constrained endpoint.

4.4 Behavioral meaning of the budget

Proposition 5 (Bounded-score retention) *For any measurable $u(x, y) \in [0, 1]$ under the reference prompt distribution and the response policies in Eq. (3), $c(\theta) \le \epsilon_f$ implies*

$$|\mathbb{E}_{\mathcal{D}_{\text{anchor}},\pi_\theta} u - \mathbb{E}_{\mathcal{D}_{\text{anchor}},\pi_0} u| \le \sqrt{c(\theta)/2} \le \sqrt{\epsilon_f/2}. \tag{16}$$

The forward-KL allowance thus bounds changes in bounded expected scores on the reference distribution. Appendix B.8 gives a confidence bound that connects an independent audit to population KL.

Under their stated conditions, these results establish a uniquely solved start, local path-sensitivity and tracking bounds, endpoint KKT stationarity, and a bound on reference-score change. The path results inform adaptive increments and correction checks, while the KL bound translates the fixed reference allowance into a bounded change in expected score.

5 Experiments

We evaluate whether RBC-LoRA improves target-task specialization while limiting changes to the pretrained model under a fixed adapter rank and retention allowance. We compare target scores and retained capabilities with LoRA and retention-aware baselines, isolate the continuation components, and examine a second backbone, code generation, training cost, and optimization trajectories.

5.1 Experimental setup

Overview. The primary experiment adapts Qwen2.5-3B-Instruct to GSM8K at three final KL allowances to test the specialization–retention trade-off. Component controls examine the allowance schedule, task ramp, factor anchor, and stage pacing. We then repeat the intermediate-allowance comparison on Llama-3.2-3B-Instruct and adapt the primary backbone to MBPP. Detailed training configurations and evaluation protocols appear in Appendix C.

Models and baselines. The Qwen2.5-3B-Instruct model (Qwen et al., 2025) uses rank-8 LoRA on the attention query, key, value, and output projections; the Llama-3.2-3B-Instruct replication uses rank 16. Table 1 compares RBC-LoRA with ordinary LoRA (Hu et al., 2021), fixed-KL regularization, direct constrained training, CLoRA (Lu et al., 2025), diagonal LaLoRA (Sliwa et al., 2025), and CoTo (Zhuang et al., 2025). Direct training solves the same final constrained problem as RBC-LoRA with the same reference information. We test final allowances of 0.8, 2.0, and 6.0 nats per response on Qwen2.5; the Llama and MBPP comparisons use 2.0. Methods are selected separately on target validation performance and trained under a 24 charged GPU-hour cap on four A100-80GB devices.

Evaluation. We measure GSM8K greedy exact-answer accuracy and MBPP greedy pass@1. Retention has two measures: forward sequence KL from the original model on held-out Dolly prompts, reported in nats per response, and score changes on MMLU (Hendrycks et al., 2020), HellaSwag (Zellers et al., 2019), and strict prompt-level IFEval (Zhou et al., 2023). The maximum capability loss is the largest of these three score drops for each run, clipped at zero before averaging. Tables report means and sample SDs over paired seeds 1–3, including unsuccessful endpoint returns. The MBPP comparison also has paired item-bootstrap intervals conditional on those trained seeds.

Datasets. We train the Qwen2.5 and Llama math adapters on GSM8K (Cobbe et al., 2021) and evaluate them on its held-out test set. For code specialization, we train Qwen2.5 on MBPP (Austin et al., 2021) and evaluate on the MBPP test tasks. Dolly-15k (Conover et al., 2023) prompts provide separate reference-gradient, controller, and final-audit collections for retention control. MMLU, HellaSwag, and IFEval assess retained capabilities after adaptation.

5.2 Result

Table 1: GSM8K specialization and retention on Qwen2.5-3B-Instruct at $\epsilon_f = 2.0$. Values are mean $\pm$ seed SD over three runs; capability changes are adapted minus original scores in percentage points (pp).

Method	GSM8K (%) ↑	Audit KL ↓	ΔMMLU (pp)	ΔHellaSwag (pp)	ΔIFEval (pp)
Original model	71.2	0.00	+0.0	+0.0	+0.0
LoRA	76.9±0.4	6.27±0.58	−2.5±0.6	−2.1±0.5	−4.4±1.1
Fixed KL	75.9±0.3	2.34±0.15	−1.1±0.4	−0.9±0.2	−2.1±0.5
Direct constrained	76.6±0.3	1.86±0.19	−0.9±0.3	−0.8±0.3	−1.5±0.6
CLoRA	76.2±0.4	2.81±0.09	−1.3±0.5	−1.1±0.3	−2.1±0.6
LaLoRA (diag.)	76.3±0.3	2.39±0.14	−1.0±0.3	−0.9±0.2	−2.0±0.6
CoTo	77.1±0.4	5.11±0.39	−2.1±0.7	−1.9±0.3	−3.6±1.2
RBC-LoRA	**77.6±0.2**	**1.78±0.07**	**−0.4±0.2**	**−0.4±0.1**	**−0.9±0.2**

Target-task improvement. At the intermediate allowance, RBC-LoRA reaches a reported $77.6 \pm 0.2\%$ GSM8K accuracy, compared with $76.6 \pm 0.3\%$ for direct constrained training and $76.9 \pm 0.4\%$ for ordinary LoRA (Table 1). The paired advantage over direct training is 0.94 ± 0.18 percentage points (pp), with a positive difference in each seed. RBC-LoRA also leads direct training in target gain at the tight and loose allowances (Figure 2a,b), consistent with the proposed benefit of approaching the shared endpoint through continuation.

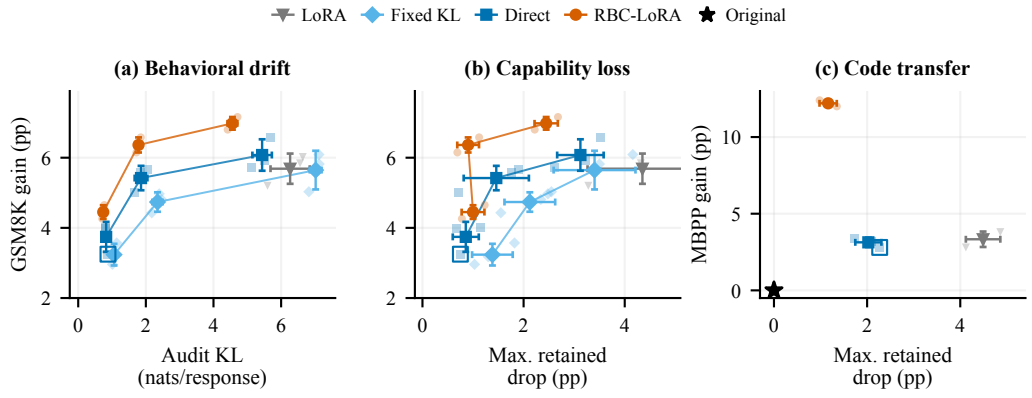

Figure 2: Specialization and retention at measured operating points: (a) GSM8K gain versus audit KL, (b) GSM8K gain versus maximum capability loss, and (c) MBPP gain versus maximum capability loss. Large markers are three-seed means with SD bars; faint markers are individual runs; hollow markers indicate failed endpoints.

Pretrained-capability retention. The target gain accompanies less measured degradation of pretrained behaviour. At the intermediate allowance, RBC-LoRA's mean audit KL is 1.78 nats per response, compared with 6.27 for ordinary LoRA and 1.86 for direct training; its mean maximum loss across MMLU, HellaSwag, and IFEval is 0.90 pp, compared with 4.35 and 1.46 pp, respectively. RBC-LoRA has the smallest mean loss on each retained benchmark in Table 1. Across the three allowances, it has lower mean audit KL than direct training; at the tight allowance, however, its maximum benchmark loss is 1.00 pp versus 0.86 pp for direct training. Figure 4 in the appendix shows the complete baseline sweep.

Transfer to code. RBC-LoRA attains $69.4 \pm 0.2\%$ MBPP pass@1, up from 57.2% for the original model and above the $60.3 \pm 0.3\%$ of direct training. Its paired advantage over direct training is 9.07 pp, with a 95% item-bootstrap interval of $[6.47, 11.74]$ pp conditional on the observed seeds. Mean audit KL is 1.71 for RBC-LoRA and 1.95 for direct training, while maximum retained-benchmark losses are 1.16 and 2.03 pp (Figure 2c; Appendix C.3). The tested math and code tasks both show higher target scores alongside smaller measured retention losses under the intermediate allowance.

5.3 Ablation Analysis

We ablate the allowance schedule, factor anchor, task ramp, and adaptive stage pacing at the intermediate allowance to identify their contributions under the same backbone, adapter rank, and work cap. The controls also test whether a learning-rate warmup can account for the observed gain.

Table 2: Component controls on Qwen2.5-3B-Instruct at $\epsilon_f = 2.0$. Values are mean $\pm$ seed SD over three runs; endpoint counts are reported successful returns. Max. drop is the largest retained-benchmark loss per run.

Variant	ΔGSM8K (pp) ↑	Audit KL ↓	Max. drop (pp) ↓	Endpoint	GPU h ↓
RBC-LoRA	**+6.4±0.2**	1.78±0.07	**0.9±0.2**	3/3	**20.6±0.4**
Direct constrained	+5.4±0.3	1.86±0.19	1.5±0.6	3/3	20.9±0.4
Constant-budget anchor	+5.5±0.5	1.90±0.16	1.5±0.3	3/3	20.9±0.3
Budget-only (full task)	+4.8±0.6	2.07±0.27	1.9±0.6	2/3	22.1±1.6
No-anchor (task ramp)	+5.7±0.2	1.90±0.04	1.3±0.2	3/3	21.3±0.3
Direct + warmup	+5.5±0.5	1.90±0.10	1.6±0.3	3/3	20.8±0.4
Fixed increments	+5.7±0.5	**1.73±0.13**	1.2±0.2	2/3	22.1±1.6

Retention-budget schedule. Constant-budget anchor annealing keeps the task ramp and decaying anchor but applies the final allowance from the start. RBC-LoRA yields a 6.4 pp GSM8K gain over the original model, versus 5.5 pp for this control, and reduces mean audit KL from 1.90 to 1.78 and maximum capability loss from 1.5 to 0.9 pp (Table 2). Changing the allowance along the path contributes to the measured target and retention gains.

Factor anchor. The no-anchor task-ramp control retains RBC-LoRA's task and allowance schedules while setting $\gamma = 0$. Its GSM8K gain is 5.7 pp versus 6.4 pp for RBC-LoRA, and both variants have 3/3 reported successful endpoints. With the schedules held fixed, the target-score difference is associated with the regular anchored start.

Task ramp. Budget-only training removes the task ramp from the no-anchor control while keeping the increasing allowance. The target gain falls from 5.7 to 4.8 pp, and reported endpoint completion falls from 3/3 to 2/3. Direct training with a 500-step learning-rate warmup reaches a 5.5 pp gain, below RBC-LoRA's 6.4 pp, showing that the recorded warmup does not reproduce the full continuation result.

Stage pacing. Replacing adaptive pacing with six fixed continuation levels gives a 5.7 pp target gain and 2/3 reported successful endpoints, compared with 6.4 pp and 3/3 for RBC-LoRA. The fixed-level control retains the objective and allowance schedules, so its lower target gain and endpoint-completion rate support adaptive stage selection under the recorded work cap.

5.4 Discussion

Across model architectures. On Llama-3.2-3B-Instruct with rank-16 adapters, RBC-LoRA gains 5.8 ± 0.2 pp on GSM8K versus 4.8 ± 0.2 pp for direct training. It also has lower mean audit KL (1.76 versus 1.96) and a smaller maximum retained-benchmark loss (1.3 versus 2.0 pp), with reported endpoint completion in 3/3 versus 2/3 runs (Appendix C.3). Together with the rank-8 Qwen2.5 result (Figure 3a), this gives evidence of the same specialization and retention pattern across the two tested backbones.

Training time. In the primary setting, RBC-LoRA uses 20.6 charged GPU hours on average, compared with 20.9 for direct constrained training and 21.6 for ordinary LoRA (Figure 3b; Appendix C.3). The cap is 24 hours per run, and charged time includes reference preparation, diagnostic checks, and rejected proposals. RBC-LoRA reaches its reported endpoint with a similar training-time cost to direct training in these runs. Evaluating the reference-policy KL adds memory cost relative to ordinary LoRA.

Trajectory stability. Across the three recorded primary runs, RBC-LoRA reaches the final continuation level and a reported successful endpoint (Figure 5 in the appendix). The controller rejects a proposal whose KL exceeds its current allowance; rollback restores the preceding accepted state, after which training progresses to the endpoint. Final validation negative log-likelihood averages 0.4383 nats per token, compared with 0.4660 for direct training (Figure 3c). The accepted paths and recovery after rejection illustrate how budget checks and adaptive correction guide progression in the tested runs, consistent with the path-following rationale in Section 3.2.

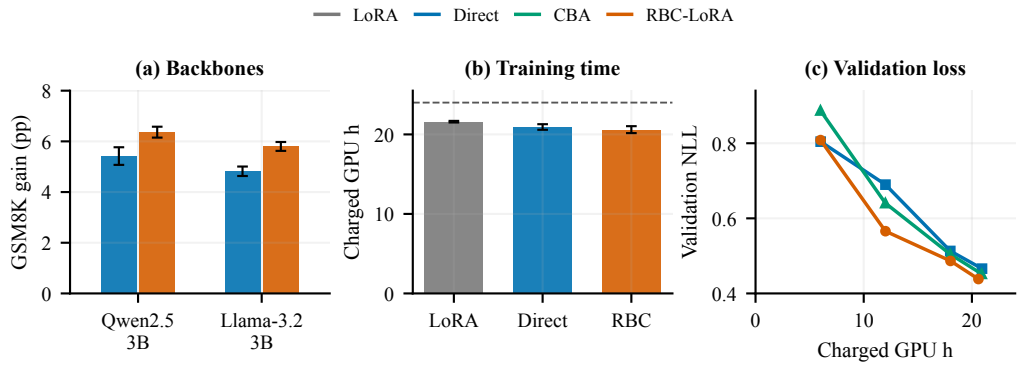

Figure 3: Results across backbones, charged training time, and validation loss. (a) GSM8K gains for two backbones; (b) primary charged GPU hours (dashed: 24-hour cap); (c) three-seed mean validation NLL near 6, 12, and 18 charged hours and at the endpoint. Bars show three-seed means $\pm$ SD. CBA denotes constant-budget anchor annealing.

6 Conclusion

We introduced RBC-LoRA, a fixed-rank homotopy that begins at a uniquely anchored solution, gradually increases task pressure and the retention allowance, and ends at the prescribed constrained problem. Our analysis establishes the regular start, characterizes local path sensitivity and conditional tracking, and bounds changes in bounded expected scores on the reference distribution. In the tested math and code settings, RBC-LoRA improves target scores over tuned direct constrained training while showing lower audit divergence and smaller measured losses on retained benchmarks at the intermediate allowance. Together, this work provides a continuation procedure and conditional path analysis, with empirical evidence that its trajectory improves the measured specialization-retention trade-off under finite computation. Future work should verify the path assumptions for the implemented stochastic corrector and test whether the gains persist in larger language models.

AI USE STATEMENT

In this paper, we used generative AI in two aspects of this work. First, AI tools helped polish manuscript drafts and identify and correct grammatical errors. Second, we used Codex to write and debug code for the experiments reported in this paper. The authors take responsibility for the final manuscript, code, results, and claims.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, et al. Lora learns less and forgets less. *arXiv preprint arXiv:2405.09673*, 2024.

Qipin Chen and Wenrui Hao. A homotopy training algorithm for fully connected neural networks. *Proceedings. Mathematical, Physical, and Engineering Sciences*, 475(2231):20190662, 2019.

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.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023. URL `https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm`.

Thomas M Cover, Joy A Thomas, and John Kieffer. *Elements of information theory*, volume 2. wiley New York, 1991.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

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 *International conference on machine learning*, 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. *arXiv preprint arXiv:2106.09685*, 2021.

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

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.

Peixuan Li and Yuanbo Li. Differentiable homotopy methods for gradually reinforcing the training of fully connected neural networks. *Neurocomputing*, 605:128374, 2024.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)*, pp. 4582–4597, 2021.

Xi Lin, Zhiyuan Yang, Xiaoyuan Zhang, and Qingfu Zhang. Continuation path learning for homotopy optimization. In *International Conference on Machine Learning*, pp. 21288–21311. PMLR, 2023.

Yuheng Lu, Bingshuo Qian, Caixia Yuan, Huixing Jiang, and Xiaojie Wang. Controlled low-rank adaptation with subspace regularization for continued training on large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19165–19181, 2025.

Paul Michel, Elisabeth Salesky, and Graham Neubig. Regularizing trajectories to mitigate catastrophic forgetting. *Preprint*, 2019.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

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.

Reece Shuttleworth, Jacob Andreas, Antonio Torralba, and Pratyusha Sharma. Lora vs full fine-tuning: An illusion of equivalence. *Advances in Neural Information Processing Systems*, 38: 174627–174662, 2026.

Joanna Sliwa, Frank Schneider, Philipp Hennig, and José Miguel Hernández-Lobato. Mitigating forgetting in low rank adaptation. *arXiv preprint arXiv:2512.17720*, 2025.

Georg Still. Lectures on parametric optimization: An introduction. *Optimization Online*, 2, 2018.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

Zhan Zhuang, Xiequn Wang, Wei Li, Yulong Zhang, Qiushi Huang, Shuhao Chen, Xuehao Wang, Yanbin Wei, Yuhe Nie, Kede Ma, et al. Come together, but not right now: a progressive strategy to boost low-rank adaptation. *arXiv preprint arXiv:2506.05713*, 2025.

A Algorithm and implementation details

This appendix specifies the estimator, diagnostic checks, and training procedure used in Section 3. Algorithm 1 separates continuation proposals from the reserved endpoint correction.

A.1 LoRA coordinates and task loss

Each adapted matrix uses $\alpha_\ell/r_\ell = 1$, with $B_{\ell,0} = 0$ and nonzero $A_{\ell,0}$. The quadratic anchor acts on these factor coordinates, which remain unrebalanced during training. Setting both factors to zero would remove the initial gradient through $B_\ell A_\ell$. In Eq. (2), $|y|$ counts response tokens and excludes the prompt. Each response loss is token-averaged before examples are averaged; the same normalization defines $\widehat{L}_{\mathrm{new}}$.

A.2 Reference process and empirical retention cost

Both policies use full-vocabulary softmax at temperature one and the same tokenizer and response horizon T. Before the first end-of-sequence (EOS) token, every next-token distribution enters the KL calculation, including the distribution from which EOS is sampled. After EOS, both processes emit a deterministic padding symbol and contribute zero KL. Responses without EOS are truncated at T. Reference rollouts use no top-k or top-p truncation; dropout is disabled when evaluating logits. The resulting cost is in nats per response.

The chain rule for relative entropy (Cover et al., 1991) gives

$$c(\theta) = \mathbb{E}_{\substack{x\sim\mathcal{D}_{\mathrm{anchor}}\\ y\sim\pi_0(\cdot|x)}} \sum_{t=1}^{T} D_{\mathrm{KL}}(\pi_0(\cdot \mid x, y_{<t}) \,\|\, \pi_\theta(\cdot \mid x, y_{<t}))\,. \tag{17}$$

For a cache $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^N$ of reference prompts and original-policy rollouts, let m_{it} mark a prefix before the first EOS, and let $p_{0,it}$ and $p_{\theta,it}$ be the full next-token distributions. The cost used for training, controller checks, or audit is

$$\widehat{c}_{\mathcal{S}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T} m_{it} \sum_{v\in\mathcal{V}} p_{0,it}(v)\,[\log p_{0,it}(v) - \log p_{\theta,it}(v)]\,. \tag{18}$$

The EOS-generating position has $m_{it} = 1$. Sequence sums precede the example average; no token-count normalization is applied. Teacher probabilities are held fixed during differentiation. Log-softmax and optional vocabulary chunking implement the full-vocabulary sum.

The task and reference training caches supply minibatches; fixed subsets of those caches supply $\widehat{L}_{\mathrm{d}}$ and $\widehat{c}_{\mathrm{d}}$. A separate controller collection supplies $\widehat{c}_{\mathrm{ctrl}}$ for proposal checks and may be queried repeatedly. The independent audit is evaluated after the returned adapter is locked and is excluded from training and controller selection. Deduplicate reference prompts, keep controller and audit prompts separate from training, and exclude benchmark examples from training and controller data.

A.3 Diagnostic checks

On fixed diagnostic subsets, let $g_{\mathrm{task}} = \nabla\widehat{L}_{\mathrm{d}}(\theta)$ and $g_{\mathrm{ret}} = \nabla\widehat{c}_{\mathrm{d}}(\theta)$. Gradients are taken with respect to the concatenated factors before clipping. Define

$$\begin{aligned} r_{\mathrm{stat}} &= \|\lambda g_{\mathrm{task}} + \gamma(1-\lambda)(\theta-\theta_0) + \nu g_{\mathrm{ret}}\|_2\,,\\ s_g &= \max\{\delta_g, \lambda\|g_{\mathrm{task}}\|_2 + \gamma(1-\lambda)\|\theta-\theta_0\|_2 + \nu\|g_{\mathrm{ret}}\|_2\}\,,\\ \bar{r}_{\mathrm{stat}} &= r_{\mathrm{stat}}/s_g, \qquad \bar{r}_{\mathrm{comp}} = \frac{\left|\nu\big(\widehat{c}_{\mathrm{d}}(\theta) - \epsilon(\lambda)\big)\right|}{\epsilon(\lambda)\max(1,\nu)}. \end{aligned} \tag{19}$$

Use a fixed $\delta_g > 0$; 10^{-12} with float32 norm accumulation is the declared starting value. A check passes when

$$\max\{\widehat{c}_{\mathrm{d}}, \widehat{c}_{\mathrm{ctrl}}\} \le (1+\tau_{\mathrm{feas}})\epsilon(\lambda), \qquad \bar{r}_{\mathrm{stat}} \le \tau_{\mathrm{stat}}, \qquad \bar{r}_{\mathrm{comp}} \le \tau_{\mathrm{comp}}, \tag{20}$$

and the costs, residuals, and optimizer state are finite. The pass counter resets at each proposal and after a failed check. Correction succeeds after two consecutive passes within its update and work limits. Numerical failure rejects the proposal immediately. Record raw costs and residuals alongside the normalized values.

A.4 Optional secant predictor

The base procedure warm-starts from the last accepted factors. After two accepted stages with the same estimated constraint-active status, the optional factor predictor for a proposed increment $h > 0$ is

$$v_k = \frac{\theta_k - \theta_{k-1}}{\lambda_k - \lambda_{k-1}}, \qquad d_k(h) = h v_k \min\left\{1, \frac{D_{\text{pred}}}{\|h v_k\|_2}\right\}, \qquad \theta^{\text{pred}} = \theta_k + d_k(h). \tag{21}$$

Set $d_k(h) = 0$ when $v_k = 0$, and fix the displacement cap $D_{\text{pred}} > 0$ and relative slack threshold for active-status estimation before training. Revert to a warm start after a status change, a nonfinite secant, or rejection. The multiplier and optimizer moments start from the accepted state.

A.5 Budgeted correction and progression

At a fixed proposed λ, each correction step evaluates the empirical Lagrangian in Eq. (8) at (θ_j, ν_j). The factors use Adam (Kingma & Ba, 2014) with zero weight decay,

$$\theta_{j+1} = \text{AdamStep}\Big(\theta_j, \nabla_\theta \widehat{\mathcal{L}}_\lambda(\theta_j, \nu_j)\Big), \tag{22}$$

and the multiplier uses Eq. (9) evaluated at the same θ_j. Adam moments and step count carry across accepted levels. Only factor-update gradients may be clipped. Check Eq. (20) at a fixed update interval that allows at least two checks within the cap C. The routine Correct returns a pass flag, its update count n, and charged work w; it stops at the update or work limit. The accepted checkpoint S holds factors, multiplier, optimizer moments and step count, and any dual-averaging state; rejection restores all of S. Charged work counters never roll back.

The first accepted proposal at $\lambda = 1$ uses the continuation allowance; the separate correction at that same level uses the remaining work, including the reserved fraction. Its anchor coefficient is zero. A rejected proposal halves the increment, and two consecutive accepted proposals using at most $C/2$ updates each double it up to $h_{\max}$. The final increment may be smaller than $h_{\min}$ if it reaches one, but a failed retry below $h_{\min}$ stops progression. The returned checkpoint is audited even when its status is a failure; failure is never counted as a solved endpoint. Audit KL is compared with ϵ_f without the controller tolerance. Table 3 records the experimental values of the controller inputs.

The work budget includes reference preparation, factor and dual updates, diagnostics, rejected proposals, and endpoint correction. Final audit and benchmark evaluation occur after training. Reference logits can be obtained by disabling the adapter, without a second backbone copy.

B Details of Theoretical Analysis

This appendix states the assumptions and proves the results in Section 4. It also gives the early-curvature calculation and an independent-audit bound. The fixed reference policy, response horizon, factor scaling, and loss normalization are defined in Section 3.

B.1 Notation and regularity

Write $L = L_{\text{new}}$ within this appendix and $\Delta_\epsilon = \epsilon_f - \epsilon_0 > 0$. The factor vector is $\theta \in \mathbb{R}^p$, the multiplier is a scalar $\nu \geq 0$, and $z = (\theta, \nu) \in \mathbb{R}^{p+1}$. Vector norms are Euclidean and matrix norms are induced operator norms. All derivatives are with respect to the concatenated factors unless specified otherwise. The exact objective, allowance, and Lagrangian are

$$F_\lambda = \lambda L + \tfrac{\gamma(1-\lambda)}{2}\|\theta - \theta_0\|_2^2, \qquad \epsilon(\lambda) = \epsilon_0 + \lambda\Delta_\epsilon, \qquad \mathcal{L}_\lambda = F_\lambda + \nu(c - \epsilon(\lambda)). \tag{23}$$

Assume $L, c \in C^2$ on an open neighborhood of all points and accumulation points under consideration. On the common response support, assume $\log \pi_\theta(y \mid x)$ is C^2 in θ. For population derivatives, assume locally integrable dominating functions for the first and second derivatives so that differentiation and expectation can be interchanged. The response support is parameter independent; EOS padding creates a common absorbing process, rather than a model-dependent truncation of the vocabulary. All expectations and moments used below are finite. The results for L and c concern exact population objectives; replacing them by fixed smooth empirical objectives gives corresponding empirical statements, without equating training diagnostics to population quantities.

Algorithm 1 Retention-budget continuation for LoRA

Require: θ_0, task data, reference cache, controller, audit; $\epsilon_0, \epsilon_f, \gamma$; work B, reserve ρ
Require: $h_0, h_{\min}, h_{\max}$; update cap C, retry limit R; check interval and tolerances
$\lambda \leftarrow 0$, $\theta \leftarrow \theta_0$, $\nu \leftarrow 0$; initialize optimizer and save checkpoint S
$h \leftarrow h_0$, rejections $r \leftarrow 0$, cheap accepts $a \leftarrow 0$
$W \leftarrow$ charged preparation work; $B_{\mathrm{cont}} \leftarrow (1-\rho)B$
while $\lambda < 1$ and $W < B_{\mathrm{cont}}$ **do**
    $h' \leftarrow \min(h, 1-\lambda)$; $\lambda^+ \leftarrow \lambda + h'$
    Restore S; optionally predict factors by Eq. (21) with h'
    $(\mathrm{pass}, n, w) \leftarrow \mathrm{Correct}(\lambda^+)$ within C updates and $B_{\mathrm{cont}} - W$ work
    $W \leftarrow W + w$ ▷ n is correction updates; w is charged work
    if pass **then**
        $\lambda \leftarrow \lambda^+$; save corrected state as S; $r \leftarrow 0$
        $a \leftarrow a + 1$ if $n \le C/2$, otherwise $a \leftarrow 0$
        if $a = 2$ **then**
            $h \leftarrow \min(2h', h_{\max})$; $a \leftarrow 0$
        else
            $h \leftarrow h'$
        end if
    else
        Restore S; $r \leftarrow r + 1$; $a \leftarrow 0$; $h \leftarrow h'/2$
        if $r \ge R$ or $h < h_{\min}$ **then**
            break
        end if
    end if
end while
if $\lambda = 1$ **then**
    Restore S; run a separate endpoint $\mathrm{Correct}(1)$ within remaining work $B - W$
    Charge endpoint work to W
    if endpoint correction passes **then**
        Save corrected state as S; status $\leftarrow$ success
    else
        Restore S; status $\leftarrow$ endpoint failure
    end if
else
    Restore S; status $\leftarrow$ progression failure
end if
Lock factors in S; evaluate independent audit and capability benchmarks
return adapter, status, audit cost, and charged-work record

The local existence results yield C^1 branches. Theorem 3 assumes a C^2 branch with bounded second derivative. Sufficient conditions are C^3 losses and a nonsingular KKT Jacobian on a neighborhood of a compact regular branch segment. Its correction inequality is a separate assumption.

B.2 Reference Fisher information and proof of Proposition 1

KL derivatives at the reference. Let $s_\theta(x, y) = \nabla_\theta \log \pi_\theta(y \mid x)$ and let $\mathbb{E}_0$ denote expectation over $x \sim \mathcal{D}_{\text{anchor}}$ and $y \sim \pi_0(\cdot \mid x)$. Differentiating the forward KL gives

$$\nabla c(\theta) = -\mathbb{E}_0 s_\theta, \qquad \nabla^2 c(\theta) = -\mathbb{E}_0 \nabla^2 \log \pi_\theta(y \mid x). \tag{24}$$

For each prompt, differentiate $\sum_y \pi_\theta(y \mid x) = 1$ once and twice. The resulting identities are

$$\mathbb{E}_{\pi_\theta} s_\theta = 0, \qquad \mathbb{E}_{\pi_\theta}\left[\nabla^2 \log \pi_\theta + s_\theta s_\theta^\top\right] = 0. \tag{25}$$

Since $\pi_{\theta_0} = \pi_0$, $c(\theta_0) = 0$, $\nabla c(\theta_0) = 0$, and

$$\mathcal{I}_0 = \nabla^2 c(\theta_0) = \mathbb{E}_0[s_{\theta_0} s_{\theta_0}^\top] \succeq 0, \qquad c(\theta_0 + d) = \tfrac{1}{2} d^\top \mathcal{I}_0 d + o(\|d\|_2^2). \tag{26}$$

For a direction d changing only A while $B = 0$, every effective weight remains unchanged. Hence $s_{\theta_0}^\top d = 0$ and $\mathcal{I}_0 d = 0$. The initial factor anchor supplies curvature in these directions.

Existence of the initial branch. At $\lambda = 0$, $F_0(\theta) = \gamma\|\theta - \theta_0\|_2^2/2$ has the unique global minimizer θ_0. The constraint has strict slack $c(\theta_0) - \epsilon_0 = -\epsilon_0 < 0$, so it remains inactive near the start. Define the stationarity map

$$G(\theta, \lambda) = \lambda \nabla L(\theta) + \gamma(1 - \lambda)(\theta - \theta_0). \tag{27}$$

It is C^1, $G(\theta_0, 0) = 0$, and $D_\theta G(\theta_0, 0) = \gamma I$. Extend this formula to a small open interval around zero and apply the implicit-function theorem, then restrict to $\lambda \geq 0$. This gives a locally unique C^1 stationary branch. Continuity of the Hessian preserves positive definiteness nearby, making its points strict local minimizers. Differentiating $G(\theta^*(\lambda), \lambda) = 0$ at zero gives $\gamma\theta^{*\prime}(0) + g_0 = 0$.

Initial loss and drift. The branch expansion is $\theta^*(\lambda) - \theta_0 = -\lambda g_0/\gamma + o(\lambda)$. Substitution into the first-order expansion of L and the second-order expansion in Eq. (26) proves Eq. (11). □

Early curvature. Direct differentiation gives

$$\nabla^2 F_\lambda = \lambda \nabla^2 L + \gamma(1 - \lambda) I \succeq [\gamma(1 - \lambda) - \lambda C_L] I \tag{28}$$

wherever $\nabla^2 L \succeq -C_L I$ with $C_L \geq 0$. The lower bound is positive when $\lambda < \gamma/(\gamma + C_L)$ and holds uniformly on a convex neighborhood with the stated Hessian bound.

B.3 Proof of Theorem 2

At an active KKT point $(\bar{\theta}, \bar{\nu}, \bar{\lambda})$, define

$$\Psi(z, \lambda) = \begin{bmatrix} \nabla F_\lambda(\theta) + \nu \nabla c(\theta) \\ c(\theta) - \epsilon(\lambda) \end{bmatrix}, \qquad D_z \Psi = K = \begin{bmatrix} H & a \\ a^\top & 0 \end{bmatrix}. \tag{29}$$

Here $a \in \mathbb{R}^p$, $H \in \mathbb{R}^{p \times p}$, and $K \in \mathbb{R}^{(p+1)\times(p+1)}$. The assumptions $a \neq 0$ and $\bar{\nu} > 0$ are the single-constraint linear independence and strict complementarity conditions, respectively.

Nonsingularity and local optimality. If $K(d, s)^\top = 0$, then $Hd + as = 0$ and $a^\top d = 0$. Multiplication by $d^\top$ gives $d^\top H d = 0$, so tangent-space positivity forces $d = 0$. Then $as = 0$ implies $s = 0$. Thus K is nonsingular, and the implicit-function theorem gives a locally unique C^1 zero of Ψ. The constraint stays active, and $\nu > 0$ and $a \neq 0$ persist by continuity. Positive definiteness on the tangent space also persists locally. With strict complementarity, the critical cone is $\{d : a^\top d = 0\}$, so the second-order sufficient condition makes these KKT points strict local minimizers (Still, 2018, Theorem 4.4).

Path derivative. Differentiation along the branch gives $Kz^{*\prime} + \partial_\lambda \Psi = 0$, where

$$\partial_\lambda \Psi = \begin{bmatrix} \nabla L(\theta^*) - \gamma(\theta^* - \theta_0) \\ -\Delta_\epsilon \end{bmatrix}. \tag{30}$$

Moving this term to the right proves Eq. (13). In particular,

$$\|z^{*\prime}(\lambda)\|_2 \le \|K^{-1}\|_2 \sqrt{\|\nabla L(\theta^*) - \gamma(\theta^* - \theta_0)\|_2^2 + \Delta_\epsilon^2}. \tag{31}$$

This gives the sensitivity bound used in Section 4.2. □

B.4 Prediction feasibility

Let $\theta^* = \theta^*(\lambda)$ be active and $v = \theta^{*\prime}(\lambda)$. The active constraint gives $c(\theta^*) = \epsilon(\lambda)$ and $\nabla c(\theta^*)^\top v = \Delta_\epsilon$. Local L_c-Lipschitz continuity of ∇c on the prediction segment gives

$$\begin{aligned} c(\theta^* + hv) &\le c(\theta^*) + h\nabla c(\theta^*)^\top v + \tfrac{L_c}{2} h^2 \|v\|_2^2 \\ &= \epsilon(\lambda + h) + \tfrac{L_c}{2} h^2 \|v\|_2^2. \end{aligned} \tag{32}$$

Taking the positive part proves Eq. (14). □

B.5 Proof of Theorem 3

Fix a compact regular branch interval and neighborhoods on which the following bounds hold. Let $v(z^*(\lambda), \lambda) = z^{*\prime}(\lambda)$ and $\|v(z, \lambda) - v(\widetilde{z}, \lambda)\|_2 \le L_v \|z - \widetilde{z}\|_2$. The implemented direction w_k obeys $\|w_k - v(z_k, \lambda_k)\|_2 \le \delta_k$, with predictor $z_{k+1}^{\mathrm{pred}} = z_k + h_k w_k$. Write $\mathcal{C}_k$ for the stage corrector, including its optimizer history, and assume on its admitted neighborhood that

$$\|\mathcal{C}_k(z) - z^*(\lambda_{k+1})\|_2 \le q\|z - z^*(\lambda_{k+1})\|_2 + \xi_k, \qquad 0 \le q < 1. \tag{33}$$

This is distance reduction relative to the solution, not an assumption that arbitrary pairs of optimizer states contract. If history is carried between stages, the bound must hold for the histories encountered. Assume $\|z^{*\prime\prime}\|_2 \le M$ and that each interval $[\lambda_k, \lambda_{k+1}]$ remains on the regular segment.

Prediction error. Taylor's formula with integral remainder gives

$$z^*(\lambda_{k+1}) = z^*(\lambda_k) + h_k v(z^*(\lambda_k), \lambda_k) + R_k, \qquad \|R_k\|_2 \le \tfrac{M}{2} h_k^2. \tag{34}$$

Subtract this expression from $z_k + h_k w_k$, add and subtract $h_k v(z_k, \lambda_k)$, and use the triangle inequality:

$$\|z_{k+1}^{\mathrm{pred}} - z^*(\lambda_{k+1})\|_2 \le (1 + h_k L_v) e_k + h_k \delta_k + \tfrac{M}{2} h_k^2. \tag{35}$$

Applying Eq. (33) to the predictor proves Eq. (15). □

Uniform control. Define $b_k = q(h_k \delta_k + M h_k^2/2) + \xi_k$. If $q(1 + h_k L_v) \le \bar{q} < 1$ and $b_k \le \bar{b}$ for all considered steps, induction yields

$$e_k \le \bar{q}^k e_0 + \sum_{j=0}^{k-1} \bar{q}^{k-1-j} b_j \le \bar{q}^k e_0 + \bar{b} \frac{1 - \bar{q}^k}{1 - \bar{q}}. \tag{36}$$

The bound holds while the iterates and predictors remain in the stipulated neighborhoods.

B.6 Proof of Proposition 4

For the same exact or fixed empirical problem at every step, define raw residuals

$$\begin{aligned} r_{s,k} &= \|\lambda_k \nabla L(\theta_k) + \gamma(1 - \lambda_k)(\theta_k - \theta_0) + \nu_k \nabla c(\theta_k)\|_2, \\ r_{f,k} &= [c(\theta_k) - \epsilon(\lambda_k)]_+, \qquad r_{c,k} = |\nu_k [c(\theta_k) - \epsilon(\lambda_k)]|. \end{aligned} \tag{37}$$

Boundedness gives an accumulation point $(\bar{\theta}, \bar{\nu})$ with $\bar{\nu} \ge 0$. Along any convergent subsequence, continuity and $r_{f,k} \to 0$ give $c(\bar{\theta}) \le \epsilon_f$. The vanishing complementarity residual gives $\bar{\nu}[c(\bar{\theta}) - \epsilon_f] = 0$. Finally, bounded factors imply $\gamma(1 - \lambda_k)(\theta_k - \theta_0) \to 0$. Taking limits in the stationarity residual yields

$$\nabla L(\bar{\theta}) + \bar{\nu} \nabla c(\bar{\theta}) = 0. \tag{38}$$

These are precisely the endpoint KKT conditions. No nonsingularity of the endpoint KKT matrix is required. □

B.7 Proof of Proposition 5

For each prompt, set $P_x = \pi_0(\cdot \mid x)$ and $Q_x = \pi_\theta(\cdot \mid x)$. Since $u(x, \cdot) \in [0, 1]$, the variational characterization of total variation and Pinsker's inequality in nats (Cover et al., 1991) give

$$|\mathbb{E}_{Q_x} u - \mathbb{E}_{P_x} u| \le \mathrm{TV}(P_x, Q_x) \le \sqrt{D_{\mathrm{KL}}(P_x \| Q_x)/2}. \tag{39}$$

Here $\mathrm{TV}(P, Q) = \frac{1}{2}\|P - Q\|_1$. Applying the triangle inequality before averaging and then Jensen's inequality to the concave square root gives

$$\begin{aligned} |\mathbb{E}_x(\mathbb{E}_{Q_x} u - \mathbb{E}_{P_x} u)| &\le \mathbb{E}_x \sqrt{D_{\mathrm{KL}}(P_x \| Q_x)/2} \\ &\le \sqrt{\mathbb{E}_x D_{\mathrm{KL}}(P_x \| Q_x)/2} = \sqrt{c(\theta)/2}. \end{aligned} \tag{40}$$

Use $c(\theta) \le \epsilon_f$ to conclude. At an exactly feasible intermediate stage, the same argument substitutes $\epsilon(\lambda)$ for ϵ_f. □

B.8 An independent-audit bound

Proposition 6 (Conditional audit certificate) *Let $\widehat{\theta}$ be selected independently of N audit pairs sampled independently from $\mathcal{D}_{\mathrm{anchor}}(x)\pi_0(y \mid x)$. Let Z_i be the full-vocabulary conditional-KL sum in Eq. (18) for pair i. Conditional on the locked model, suppose $0 \le Z_i \le M_c$ almost surely for a justified constant $M_c > 0$. For $\delta_{\mathrm{aud}} \in (0, 1)$, with probability at least $1 - \delta_{\mathrm{aud}}$,*

$$c(\widehat{\theta}) \le U_N := \widehat{c}_{\mathrm{audit}}(\widehat{\theta}) + M_c \sqrt{\frac{\log(1/\delta_{\mathrm{aud}})}{2N}}. \tag{41}$$

Proof. Condition on $\widehat{\theta}$. Independence and the chain rule give $\mathbb{E}[Z_i \mid \widehat{\theta}] = c(\widehat{\theta})$. Applying Hoeffding's bounded-sum inequality (Hoeffding, 1963) to the lower tail of the sample mean gives

$$\Pr\Big(c(\widehat{\theta}) - \widehat{c}_{\mathrm{audit}} \ge t \mid \widehat{\theta}\Big) \le \exp(-2Nt^2/M_c^2). \tag{42}$$

Choose $t = M_c\sqrt{\log(1/\delta_{\mathrm{aud}})/(2N)}$ and average the conditional probability bound over model selection. □

On this event, $U_N \le \epsilon_f$ certifies the allowance, and Proposition 5 bounds the score difference by $\sqrt{U_N/2}$. A sufficient condition for the bounded-cost assumption is $\pi_{\widehat{\theta}}(v \mid x, y_{<t}) \ge p_{\min} > 0$ at every nonabsorbing reference prefix, which gives $M_c = T \log(1/p_{\min})$. With multiple rollouts per prompt, independent prompt-level averages can serve as the Z_i; rollouts sharing a prompt are not independent audit units.

B.9 Scope of the guarantees

The start is unique at $\lambda = 0$, while the initial and active-path branches are local. At the endpoint, the symmetry $BA = (BR)(R^{-1}A)$ can make the factor-space KKT matrix singular, so Theorem 2 need not extend through $\lambda = 1$. Theorem 3 assumes a corrector contraction, which is unverified for the implemented updates; the factor-only predictor can also leave first-order multiplier error. Proposition 4 requires vanishing raw residuals for one fixed objective; finite diagnostic tolerances and sampling error give only approximate empirical checks. Proposition 5 concerns bounded expected scores under reference prompts and stochastic policies, and its unit-range bound is vacuous when $\epsilon_f \ge 2$. The audit bound requires independent samples and a justified almost-sure bound M_c; reusing audit data for model selection invalidates its confidence statement.

C Experimental Details and Supplementary Results

C.1 Training configurations

Models and data splits. Qwen2.5-3B-Instruct uses rank-8 adapters, and the Llama-3.2-3B-Instruct replication uses rank 16. In both math settings, split seed 0 reserves 500 of the 7,473 GSM8K training examples for validation and leaves 6,973 for optimization; the 1,319 test examples are held out. MBPP uses training IDs 601–974, validation IDs 511–600, and test IDs 11–510. Its 374 training tasks and 90 validation tasks support the code experiment at the intermediate allowance. The three disjoint Dolly-15k reference collections contain 1,024 gradient, 256 controller, and 512 audit prompts. Appendix A.2 specifies the fixed reference policy, four temperature-one rollouts per prompt, 256-token horizon, and end-of-sequence convention used to compute sequence KL.

Table 3: Shared training configuration and RBC-LoRA controller. MBPP changes target data. Both use $\epsilon_f = 2.0$. Candidate grids and baseline-specific choices appear below.

Component	Configuration
Adapter	Attention Q/K/V/O; $\alpha/r = 1$; Gaussian A, $B = 0$; dropout 0; no factor rebalancing. The recorded Gaussian parameter is 10^{-2}; its variance/standard-deviation convention is unspecified.
Precision / hardware	BF16 backbone and adapters; FP32 logits and KL accumulation; four A100-80GB (SXM) devices.
Batch	Task: 64 sequences of length 1,024; reference: 16 prompts with four rollouts each per KL-gradient evaluation.
Optimizer	Adam (Kingma & Ba, 2014), $(\beta_1, \beta_2) = (0.9, 0.999)$, $\varepsilon = 10^{-8}$, weight decay 0, gradient clipping at 1.
Selected primal rates	2×10^{-4} for LoRA, CLoRA, and CoTo; 10^{-4} for the other recorded variants.
Dual / anchor	Dual rate 0.05; $\gamma = 3 \times 10^{-4}$; RBC-LoRA $\epsilon_0/\epsilon_f = 0.1$.
Allowance / increments	$\epsilon_f \in \{0.8, 2.0, 6.0\}$; $(h_0, h_{\min}, h_{\max}) = (1/6, 1/96, 1/3)$; halve on rejection and double after two cheap accepts, subject to bounds.
Correction / acceptance	At most 200 updates per proposal and three rejections per stage; checks every 25 updates; two consecutive passes; $(\tau_{\text{feas}}, \tau_{\text{stat}}, \tau_{\text{comp}}) = (0.05, 0.01, 0.02)$.
Work accounting	Cap of 24 charged GPU hours; 30% endpoint reserve; preparation, diagnostics, and rejected work included; search and final evaluation excluded.

Optimization and hyperparameters. The factor anchor is evaluated in the unrebalanced adapter coordinates, so initialization and scaling are part of the method specification. Table 3 records the shared optimizer and controller parameters. The core primal-rate grid is $\{10^{-5}, 3\times10^{-5}, 10^{-4}, 2\times10^{-4}, 3\times10^{-4}\}$. The main baselines and RBC-LoRA additionally consider 5×10^{-4}, and ordinary LoRA also considers 10^{-3}. The component controls use the core grid. Dual rates are selected from $\{0.01, 0.05, 0.2\}$ and anchor strengths from $\{10^{-4}, 3\times10^{-4}, 10^{-3}\}$. The Llama replication uses a primal-rate grid of $\{3\times10^{-5}, 10^{-4}, 2\times10^{-4}\}$ and selects 10^{-4}, with the shared dual and anchor parameters. Code methods inherit their primary grids and selected rates.

Baseline implementations. Fixed-KL LoRA minimizes $\widehat{L}_{\text{new}} + \beta\widehat{c}$, choosing $\beta = 3, 1, 0.3$ at the tight, intermediate, and loose operating points from $\{0.1, 0.3, 1, 3, 10\}$. CLoRA uses regularization width k = 2,048 and 512 at its two operating points, selected from $\{512, 1{,}024, 2{,}048\}$ with coefficient 1. Diagonal LaLoRA selects penalties 10^3 and 10 from $\{1, 10, 10^2, 10^3, 10^4\}$. Its diagonal empirical Fisher uses one batch over the 1,024 Dolly gradient prompts at the zero-effect adapter, replacing the source-proxy batches of the published method (Sliwa et al., 2025). CoTo uses activation probability $p(s) = \min\{4s/(3S), 1\}$ over S training steps, with all adapters active at evaluation (Zhuang et al., 2025).

Component controls. Table 4 defines each variant using $L = \widehat{L}_{\text{new}}(\theta)$, $Q = \|\theta - \theta_0\|_2^2/2$, and $\epsilon_\lambda = \epsilon_0 + \lambda(\epsilon_f - \epsilon_0)$. The no-anchor task-ramp variant begins correction at the first positive λ; budget-only training uses full task pressure at every stage. Fixed increments retain correction, rollback, and acceptance checks at six prescribed levels, with at most three retries per level. Direct training fixes $\lambda = 1$ and ϵ_f throughout, using optimizer-step backtracking after failed correction.

The warmup control adds 500 linear learning-rate steps to direct training. Rejected work is charged for every variant.

Table 4: Objectives and progression for the seven primary component variants. Every row targets L at allowance ϵ_f if the endpoint is reached.

Variant	Stage objective	Allowance	Progression
RBC-LoRA	$\lambda L + \gamma(1-\lambda)Q$	ϵ_λ	Adaptive
Direct constrained	L	ϵ_f	Endpoint throughout
Constant-budget anchor	$\lambda L + \gamma(1-\lambda)Q$	ϵ_f	Adaptive
Budget-only (full task)	L	ϵ_λ	Adaptive
No-anchor (task ramp)	λL	ϵ_λ	Adaptive
Direct + warmup	L	ϵ_f	500-step rate warmup
Fixed increments	$\lambda L + \gamma(1-\lambda)Q$	ϵ_λ	$\lambda = 1/6, 2/6, \ldots, 1$

Selection and stopping. Target validation performance selects configurations, with lower controller KL breaking ties. Test benchmarks and audit prompts do not enter selection. The declared primary search allows at most 12 trials and approximately 70 GPU hours per method, separate from the 24-hour cap on each final run; MBPP allows at most eight trials per method. Candidate grids are sampled rather than searched exhaustively. All final returns contribute to the reported summaries, including constrained runs that do not reach or pass the endpoint. Charged time includes preparation, diagnostics, and rejected work; final evaluation and search are excluded. Peak memory is measured per device.

C.2 Evaluation protocol

Reference-policy divergence. For each reference prompt, we sample four original-policy responses at temperature one and evaluate the full-vocabulary forward KL at every prefix until the first end-of-sequence token or the 256-token horizon. We sum token KL before averaging over responses and prompts, so audit divergence is measured in nats per response. The 1,024-prompt gradient cache supplies retention updates, the 256-prompt controller set governs stage acceptance, and the separate 512-prompt audit set evaluates the locked final adapter without the controller tolerance. Appendix A.2 gives the estimator and termination convention.

Target and capability benchmarks. GSM8K uses greedy exact-answer accuracy. MBPP uses one greedy completion per test task and the mean binary pass indicator. MMLU is evaluated with five shots and is the unweighted mean of subject accuracies over 44 selected subjects, rather than the usual full-suite score or a pooled item mean. HellaSwag uses zero-shot, length-normalized scoring on a fixed 2,048-item validation subset. IFEval uses strict prompt-level accuracy over 541 prompts. For retained benchmark d, let $a_{d,s}$ be a trained run's accuracy and a_d^0 the corresponding original-model score. We compute

$$\Delta_{d,s} = 100(a_{d,s} - a_d^0), \qquad D_s = 100 \max\left\{0, \max_{d \in \{\mathrm{MMLU,HellaSwag,IFEval}\}} (a_d^0 - a_{d,s})\right\}. \quad (43)$$

Both measures are in percentage points. We compute D_s before averaging the three seeds and use the same adapted-minus-original convention for target gains. The original Qwen2.5 scores are 71.2% GSM8K, 64.8% MMLU, 74.3% HellaSwag, and 56.8% IFEval; the Llama scores are 73.4%, 62.1%, 75.1%, and 61.2%, respectively. The original Qwen2.5 model passes 286/500 MBPP tasks.

Seed and item uncertainty. We report means and sample SDs over three training returns. Paired method differences are computed within seed before averaging. For MBPP, every method shares test IDs 11–510; the item bootstrap resamples these 500 IDs jointly across methods and seeds 10,000 times, then takes the 2.5th and 97.5th percentiles. We use NumPy 2.5.3 for each contrast. These intervals describe test-item variation conditional on the trained models, whereas the seed SD describes variation among the three runs.

C.3 SUPPLEMENTARY RESULTS

Training resources. Table 5 reports consumed charged time and peak memory. RBC-LoRA averages 20.6 GPU hours and 43.5 GiB per device, compared with 20.9 hours and 44.2 GiB for direct training and 21.6 hours and 28.4 GiB for ordinary LoRA. The methods share a per-run cap, while their consumed time and task-token exposure differ. The retention-controlled runs evaluate reference-policy KL and use more peak memory than ordinary LoRA.

Table 5: Primary training resources, Llama-3.2 GSM8K replication, and Qwen2.5 MBPP results. Values are mean ± seed SD over three returns at $\epsilon_f = 2.0$ for the secondary settings. Target and capability changes are in pp; audit KL is in nats per response. CBA denotes constant-budget anchor annealing.

Primary method	Charged GPU h ↓	Peak memory (GiB) ↓
LoRA	21.6 ± 0.1	28.4 ± 0.1
Fixed KL	20.9 ± 0.4	43.6 ± 0.2
Direct constrained	20.9 ± 0.4	44.2 ± 0.1
CLoRA	22.2 ± 0.5	30.1 ± 0.1
LaLoRA (diag.)	22.1 ± 0.4	30.9 ± 0.2
CoTo	20.8 ± 0.4	29.0 ± 0.1
RBC-LoRA	20.6 ± 0.4	43.5 ± 0.1

Replication method	ΔGSM8K (pp) ↑	Audit KL ↓	Max. drop (pp) ↓	Endpoint
RBC-LoRA	$\mathbf{+5.8 \pm 0.2}$	$\mathbf{1.76 \pm 0.09}$	$\mathbf{1.3 \pm 0.2}$	3/3
Direct constrained	$+4.8 \pm 0.2$	1.96 ± 0.13	2.0 ± 0.2	2/3
Constant-budget anchor	$+5.1 \pm 0.4$	1.82 ± 0.10	1.9 ± 0.6	3/3

Code method	ΔMBPP ↑	Audit KL ↓	Δ retained domains (pp)			Endpoint	GPU h ↓
			MMLU	HellaSwag	IFEval		
Original model	$+0.0$	0.00	$+0.0$	$+0.0$	$+0.0$	–	–
LoRA	$+3.3 \pm 0.5$	5.68 ± 0.28	-2.1 ± 0.1	-1.9 ± 0.2	-4.5 ± 0.4	–	21.1 ± 0.1
Direct	$+3.1 \pm 0.3$	1.95 ± 0.14	-1.0 ± 0.1	-0.8 ± 0.1	-2.0 ± 0.3	2/3	22.0 ± 1.7
CBA	$+3.3 \pm 0.3$	1.87 ± 0.07	-0.9 ± 0.1	-0.7 ± 0.1	-1.7 ± 0.3	3/3	21.4 ± 0.4
RBC-LoRA	$\mathbf{+12.2 \pm 0.2}$	$\mathbf{1.71 \pm 0.05}$	$\mathbf{-0.5 \pm 0.1}$	$\mathbf{-0.5 \pm 0.1}$	$\mathbf{-1.2 \pm 0.2}$	3/3	$\mathbf{20.6 \pm 0.2}$

Paired MBPP differences (RBC-LoRA minus comparator; mean over the three paired seeds) with 95% conditional item-bootstrap intervals (10,000 joint resamples over the 500 test IDs, resampling seed 20260920): direct $+9.07$ pp $[+6.47, +11.74]$; constant-budget anchor $+8.93$ pp $[+6.33, +11.60]$. Intervals reflect test-item sampling conditional on the observed seeds, not training-population uncertainty; per-seed differences are +9.0/+9.0/+9.2 pp and +9.0/+8.8/+9.0 pp. Peak memory per device spans 43.4–43.6 (RBC-LoRA), 43.7–43.9 (Constant-budget anchor), 44.0–44.3 (Direct constrained), 28.4–28.6 (LoRA) GiB.

Llama-3.2 replication. RBC-LoRA reaches a reported $79.20 \pm 0.18\%$ GSM8K accuracy, compared with $78.22 \pm 0.19\%$ for direct training and $78.51 \pm 0.39\%$ for constant-budget anchor annealing. The paired RBC-LoRA advantage over direct training is 0.98 ± 0.21 pp. Mean maximum capability losses are 1.28 and 1.99 pp, respectively, and reported successful endpoints are 3/3 and 2/3.

MBPP specialization. The original model passes 286/500 test tasks. RBC-LoRA passes 347, 348, and 346 across seeds, versus 302, 303, and 300 for direct training and 302, 304, and 301 for constant-budget anchor annealing. RBC-LoRA's mean pass@1 is $69.4 \pm 0.2\%$. Its paired advantages are 9.07 pp over direct training and 8.93 pp over constant-budget anchor annealing; the 95% conditional item-bootstrap intervals are $[6.47, 11.74]$ and $[6.33, 11.60]$ pp. All RBC-LoRA and constant-budget returns have reported successful endpoints, compared with two of three direct returns. The failed direct return remains in the mean.

Code-task retention. RBC-LoRA has a mean audit KL of 1.71, compared with 1.95 for direct training, 1.87 for constant-budget anchor annealing, and 5.68 for ordinary LoRA. The corresponding maximum capability losses are 1.16, 2.03, 1.66, and 4.49 pp. Trained-model retention scores reconstruct from item records, while reference-relative losses still depend on the original-model summaries described in Appendix C.2. The bound in Proposition 5 applies to bounded expected scores on the reference distribution; MMLU, HellaSwag, and IFEval are separate evaluations of retained capability.

Complete operating-point display. Figure 4 shows every measured baseline operating point, the accepted-state validation NLL traces, and the constant-budget anchor comparison on MBPP. The main-text Figure 2 displays four primary methods and omits the constant-budget anchor from its code panel for legibility.

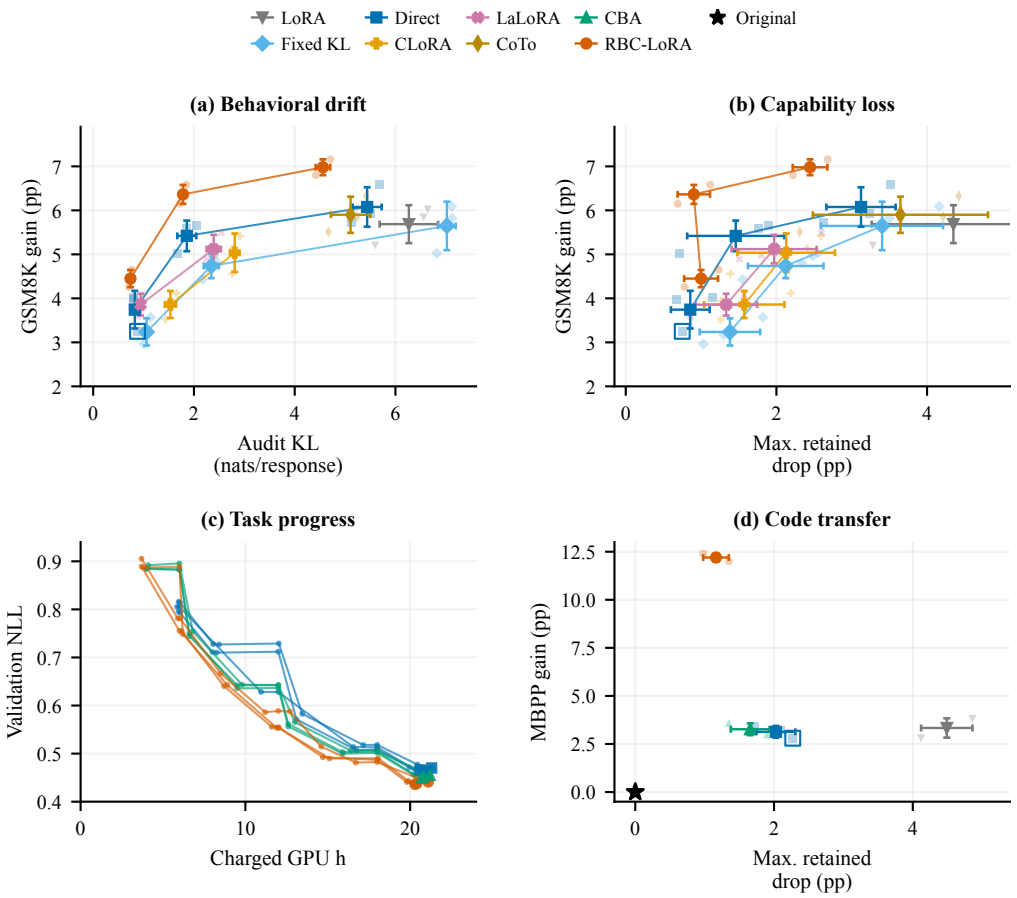

Figure 4: Complete specialization and retention comparisons. (a,b) GSM8K gain versus audit KL and maximum capability loss; (c) validation NLL versus charged time; (d) MBPP gain versus maximum capability loss. Large markers show three-seed means with SD bars, faint markers show individual runs, and hollow markers indicate failed endpoints. CBA denotes constant-budget anchor annealing.

C.4 Optimization trajectories

Progress and task exposure. The traces cover RBC-LoRA, direct training, and constant-budget anchor annealing at three seeds each. They record proposed, accepted, restored, and final states, with validation negative log-likelihood (NLL), KL, normalized residuals, and cumulative task work. Candidate corrections are checked every 25 updates. Measurements at 6, 12, and 18 charged hours and at the final return supply Figure 4c and Figure 3c. Final mean NLL is 0.4383, 0.4660, and

0.4520 nats per token for RBC-LoRA, direct training, and constant-budget anchor annealing, respectively. Mean task-token counts are 13.486, 14.074, and 13.688 million, including rejected work but excluding reference and validation tokens.

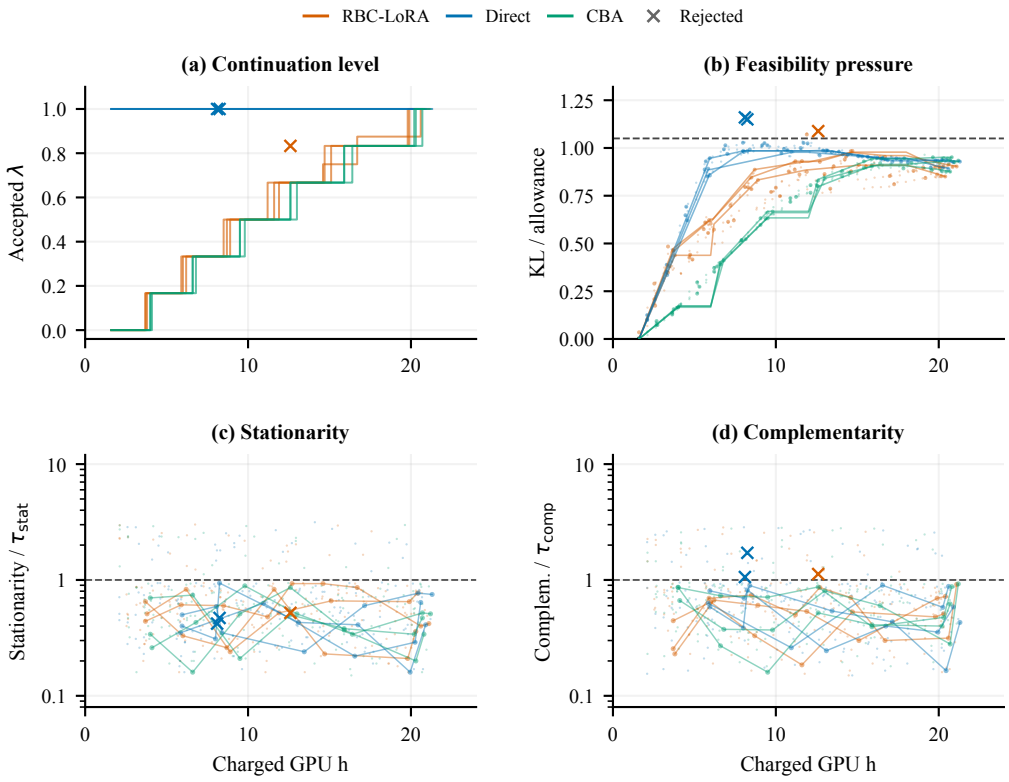

Figure 5: Controller trajectories for three seeds per method at the intermediate allowance. (a) Accepted continuation level; (b) controller KL divided by its allowance; (c,d) normalized stationarity and complementarity residuals divided by their tolerances. Crosses mark rejected proposals, and dashed lines mark acceptance thresholds.

Acceptance and rollback. Figure 5 shows an RBC-LoRA proposal at $\lambda = 0.833$ rejected when controller KL reaches 1.85 against an allowance of 1.70. Rollback restores the preceding accepted state, and the recorded path subsequently reaches the endpoint. First endpoint proposals occur at 15.15, 16.75, and 14.75 charged hours across seeds; accepted final states occur at 19.9, 20.6, and 19.8 hours. The logged accepted levels and rollback illustrate progression under the controller, but some candidate checks omit training-diagnostic KL and one recorded increment differs from the stated halving/doubling schedule.

D LIMITATIONS

While RBC-LoRA provides a structured route to a fixed retention-constrained endpoint, several avenues for further refinement remain: The path-sensitivity and tracking results are local and assume regularity and corrector contraction, neither of which has been verified for the stochastic implementation. They therefore do not establish that finite-time training follows the ideal path or reaches an exact constrained endpoint. We will examine these assumptions for the implemented corrector and study how stochastic updates affect path tracking. Our experiments cover two 3B-parameter backbones, leaving RBC-LoRA's performance on larger language models unknown. We will test larger models to assess whether the target-task gains and measured retention persist as optimization and memory demands grow.