LOCAL CONFLICT, TARGETED REPAIR: A PERTURBATION ANALYSIS OF MULTI-DOMAIN RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) post-training improves large language models on individual domains but can degrade previously acquired capabilities. We study this interference in mathematical reasoning, code generation, question answering (QA), and creative writing (CW). Full-model domain gradients can be nearly orthogonal despite substantial selective degradation. Our structural analyses show sparse, weakly overlapping edits on computation routes that reasoning domains share, with directional alignment associated with transfer and conflict. A local perturbation model separates loss changes into a first-order term and a path-averaged curvature term, decomposes the latter along the actual rollback intervention, and provides a conditional explanation for recovery by short refresh. The experiments support this local sensitivity account. After Code $\rightarrow$ Math $\rightarrow$ QA $\rightarrow$ CW training, a short Re-Math refresh raises Math from 57.66 to 66.04 and achieves the highest average score among the evaluated baselines. A training-free rollback at a 9.8% budget recovers 57.4% of the QA-induced Math accuracy drop with a small QA change. Curvature probes at the Math checkpoint further show higher sensitivity along the selected direction than along matched random directions. Together, these results support sparse intervention and targeted refresh as effective ways to mitigate cross-domain degradation.

1 INTRODUCTION

RL has become a primary technique for improving large language models (LLMs) across reasoning, coding, question answering, and open-ended generation. Real deployments need a single model strong across all of these domains rather than a collection of single-domain experts, and model iteration makes this demand incremental: new domain capabilities are often added on top of an already-trained model. The two standard recipes are joint (mixed) training over all domains at once and sequential training one domain after another.

A sequential curriculum makes the problem clear. Under the Omni-Thinker (Li et al., 2025) curriculum Code $\rightarrow$ Math $\rightarrow$ QA $\rightarrow$ CW (Table 1), Math rises to 66.49 after its own stage but falls to 57.66 after the later QA and CW stages, while Code and QA remain comparatively stable. In our comparisons, joint training and replaying all four domains also leave Math below its single-domain expert (Section 6.1). These observations motivate a closer look at the mechanism of forgetting (Kirkpatrick et al., 2016; Shi et al., 2024): the full-model gradient cosine stays near zero even as an earlier domain is damaged, so global gradient conflict (Sener & Koltun, 2018; Liang et al., 2026) alone does not characterize the observed interference. Where does the relevant cross-domain sensitivity reside?

Table 1: Motivating example under the Omni-Thinker curriculum: RL improves the current domain but later stages selectively damage Math.

	Code_o	Math_o	QA_o	CW_o
Code	**52.67**	50.69	50.99	50.47
Math	59.63	**66.49**	59.90	57.66
QA	60.89	60.52	**62.34**	62.34
CW	82.40	81.44	81.79	**86.52**

We examine the local routes through which updates act. Layer- and module-level analysis reveals localized conflict and synergy beneath global near-orthogonality. Single-domain RL produces sparse, small-magnitude edits with weak overlap among the top-changed neurons. Meanwhile, reasoning domains share many active computation routes, and the direction of their edits on these routes is

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.

associated with transfer and conflict. These observations motivate a local account of interference beyond global gradient alignment or edit overlap alone.

We formalize this account with a local perturbation model. The loss change caused by a later-domain displacement has an exact decomposition into a signed first-order term and a path-averaged curvature term. Splitting the displacement into the part removed by a sparse intervention and the part retained identifies the corresponding curvature contributions and their interaction. Under local positive-curvature and weak-coupling conditions, an idealized short refresh has a geometric upper bound on its harmful displacement. This theory offers a conditional explanation for selective degradation and repair; the empirical case rests on the intervention and recovery results.

We examine the account through three experiments. First, a short Re-Math refresh from CW_o raises Math from 57.66 to 66.04, close to its single-domain expert, with limited changes to the other domains and the highest average score among the evaluated baselines (66.39). Second, training-free rollback recovers 20.4% of the QA-induced Math accuracy drop at a 1.3% budget and 57.4% at 9.8%, with small QA changes. Third, at Math_o, Math curvature along the selected direction is 4–$7\times$ that of matched random directions. Together, these results support sparse intervention and targeted refresh as repair strategies, with directional curvature providing a complementary view of local sensitivity.

Our contributions are threefold:

- We identify local structure in cross-domain interference through sparse edits, shared active routes, and directional alignment, despite weak full-model gradient conflict.
- We derive an exact local loss decomposition using the actual rollback operator, including fractional reversion of shared key and value rows, and establish a conditional bound for short refresh.
- We demonstrate effective recovery through short targeted refresh and sparse training-free rollback, supported by directional curvature diagnostics.

2 Related Work

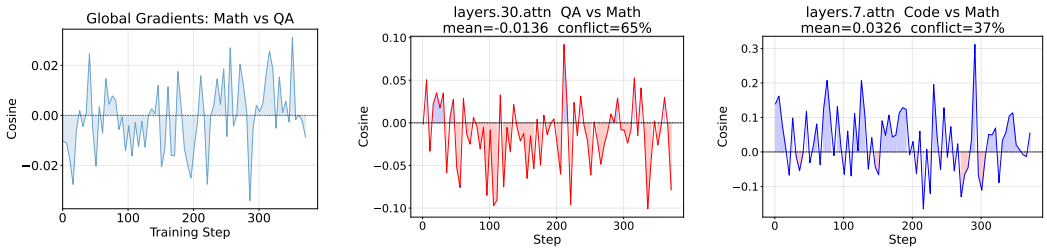
RL post-training and multi-domain RL mainly optimize aggregate performance, leaving cross-domain interactions inside the model poorly understood. The line runs from preference-based RLHF (Christiano et al., 2017; Stiennon et al., 2020; Schulman et al., 2017) through process supervision and large-scale reasoning RL (Lightman et al., 2024; Shao et al., 2024; DeepSeek-AI, 2025; Team, 2025a) to recent multi-domain methods such as Omni-Thinker (Li et al., 2025) and CGPO (Liang et al., 2026). We instead ask where cross-domain interference resides, why it is selective, and why a short domain refresh restores performance.

Cross-domain degradation is often discussed in terms of continual learning (CL), forgetting, or gradient conflict. Recent studies show that apparent forgetting can reflect shifts in alignment or policy behavior (Zheng et al., 2025). SFT–RL comparisons likewise suggest that RL may yield milder drift (Chu et al., 2025; Lai et al., 2025; Shenfeld et al., 2025; Chen et al., 2025), with additional evidence for reasoning transfer (Huan et al., 2025). Multi-task optimization offers a complementary global view through gradient balancing (Chen et al., 2018; Sener & Koltun, 2018) and gradient surgery (Yu et al., 2020; Liu et al., 2021). Our results suggest that full-model gradients can appear nearly orthogonal even when localized conflict harms earlier domains.

Task-vector and mechanistic work suggests that model changes can often be localized. In weight space, model soups (Wortsman et al., 2022) and TIES-style merging (Yadav et al., 2023) attribute transfer and interference to redundant or sign-conflicting updates (Matena & Raffel, 2022). Mechanistic studies localize knowledge and skills to compact subsets of parameters (Bau et al., 2019) or neurons (Dai et al., 2022; Wang et al., 2022; Leng & Xiong, 2025). Recent feature-level analyses suggest that RL mostly preserves the base representation while strengthening compact task-relevant features (Shi et al., 2026). We build on this locality view to study changed neurons, active routes, and direction-dependent interference.

3 Empirical Study Setup

To investigate where cross-domain interference arises and how recovery later occurs, we construct a controlled multi-domain RL setting on Qwen3-4B-Thinking-2507 (Team, 2025c) over four domains:



(a) Full-model gradient cosine. (b) Most conflicting module. (c) Most synergistic module.

Figure 1: Gradient-level view of Math-QA interference in joint training: the full-model cosine stays near zero (a), yet a per-module decomposition reveals consistent conflict (b) and synergy (c).

mathematical reasoning (Math), code generation (Code), question answering (QA), and creative writing (CW).

Data. Math is sampled from OpenR1-math (Hugging Face, 2025), Code from KlearReasoner-CodeSub-15K (Su et al., 2025), QA from SuperGPQA (Team, 2025b) stratified by subfield and difficulty, and CW from the crownelius/Creative-Writing series[1] (details in Appendix A.1).

Training. All runs use GRPO (Shao et al., 2024) in VeRL (Sheng et al., 2025) with per-domain reward functions (Appendix A.3); except for the training domain and the initialization checkpoint, all experiments share the hyperparameters of Appendix A.2. We first train four single-domain experts from the base model (Math_s, Code_s, QA_s, CW_s), then run sequential multi-domain training in the fixed Omni-Thinker order (Li et al., 2025) Code $\rightarrow$ Math $\rightarrow$ QA $\rightarrow$ CW, each stage continuing from the previous checkpoint and yielding Code_o, Math_o, QA_o, and CW_o. Section 6 adds two four-domain mixed baselines: JT, naive joint training in which each batch contains equal amounts of data from all domains and parameters are updated in the standard way, and CGPO (Liang et al., 2026), which follows the official implementation in computing domain-wise updates within each batch and then updating the affected parameters with a specific learning rate.

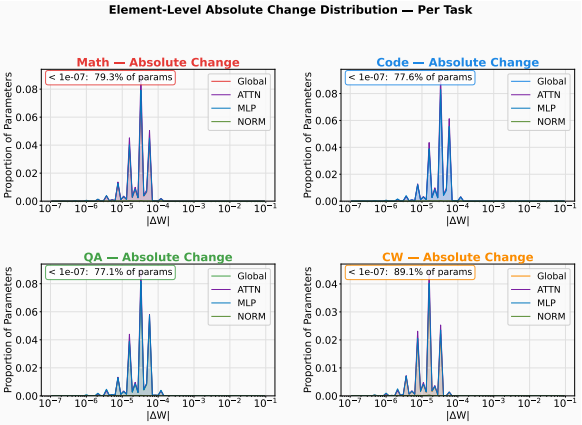


Figure 2: Absolute parameter changes of the four single-domain experts relative to the base model.

Checkpoints and evaluation. Each stage trains to convergence and selects the checkpoint with the best validation performance on its current training domain. Final evaluation uses independent benchmarks: AIME24/25/26 (Art of Problem Solving, 2024a), OlympiadBench (He et al., 2024) and HMMT (Dekoninck et al., 2026) for Math; LiveCodeBench-v6 (Jain et al., 2025) for Code; SuperGPQA-test (Team, 2025b) and MMLU-Pro (Wang et al., 2024) for QA; and WritingBench (Wu et al., 2025) for CW (Appendix A.1).

4 Structural Evidence for Localized Cross-Domain Interference

This section asks where cross-domain interference resides. We proceed by elimination: full-model gradients rule out global antagonism, and sparse, weakly overlapping parameter edits rule out co-editing of the same weights, which shifts the perspective from training to the more direct inference stage, taking shared active routes as the candidate site. We then show that the direction of the edits carried on those routes is what separates transfer from damage.

[1] https://huggingface.co/Crownelius

4.1 Global Gradients Miss Local Conflict

A natural first question is whether cross-domain interference in domain RL can be explained by global gradient conflict. During JT training, domain-specific gradients are periodically computed and the cosine similarity between domain pairs is measured, $\cos(\boldsymbol{g}_{d_i}, \boldsymbol{g}_{d_j}) = \frac{\boldsymbol{g}_{d_i}^\top \boldsymbol{g}_{d_j}}{\|\boldsymbol{g}_{d_i}\|_2 \|\boldsymbol{g}_{d_j}\|_2}$, where $\boldsymbol{g}_d$ is the gradient for domain d. Positive, negative, and near-zero values indicate alignment, conflict, and approximate orthogonality, respectively.

Figure 1a shows that the full-model gradient cosine between Math and QA stays close to zero throughout training, so the later drop in Math cannot be fully attributed to full-model gradient antagonism. Global near-orthogonality does not, however, imply that interference is absent. A coarse first probe is to decompose the same gradients by layer and module: individual attention and MLP modules then show consistent conflict (Figure 1b) or synergy (Figure 1c), most visibly for the Math-QA pair (Appendix B.1). Because this decomposition is at whole-module granularity and describes instantaneous gradients, we treat it only as a hint that interference is localized.

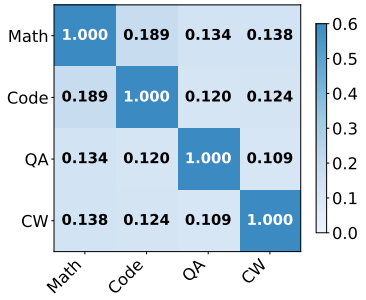
4.2 Edits Are Sparse and Small

Gradients describe what training would do at an instant, not what was written into the weights. Following the distribution-shift view under which catastrophic forgetting is usually characterized, we compare each single-domain expert with the model it was trained from, measuring element-wise absolute and relative changes $\Delta \boldsymbol{W} = \boldsymbol{W}_{\text{expert}} - \boldsymbol{W}_{\text{base}}$ and $r(\boldsymbol{W}) = |\Delta \boldsymbol{W}|/|\boldsymbol{W}_{\text{base}}|$. Across the four experts, 77%–89% of parameters have $|\Delta \boldsymbol{W}| < 10^{-7}$ and $r(\boldsymbol{W}) < 10^{-3}$ (Figure 2), and even the parameters that do move move only slightly. Single-domain RL therefore acts as a mild perturbation of the base model rather than a global parameter rewrite. The relative-change distributions and the parameter analysis of sequential training are in Appendix B.2.

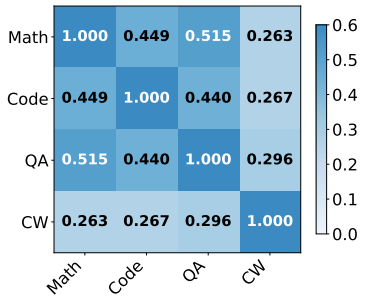
4.3 Domains Edit Weakly Overlapping Neurons

Since domain RL updates are sparse, a natural explanation for strong interference is direct co-editing: different domains may concentrate their large updates on the same set of functional units. We test this explanation by lifting the analysis from parameters to MLP neurons and measuring the overlap among the most strongly changed neurons across domain experts. For each layer, we select the top 10% most changed neurons for each domain expert and compute the pairwise Jaccard overlap between these sets. Full definitions are provided in Appendix B.3.

Figure 3a shows that top-changed neurons overlap only weakly across domains, with average Jaccard coefficients below 0.19 for all domain pairs. Domain RL therefore targets largely distinct local neuron subsets. This makes broad co-editing an incomplete description of the observed interference, although the small intersection may still matter. The next two subsections examine whether these edits act on functionally shared routes and whether their directions help characterize transfer and conflict.



(a) Changed-neuron overlap.



(b) Active-neuron overlap.

Figure 3: Neuron-overlap rates under different settings.

4.4 Sparse Edits Share Computation Routes

Performance damage is itself evaluated at inference time, which makes it natural to shift the view from the parameters that were changed to the units that are used: low edit overlap does not imply functional independence, since two domains can write to almost disjoint weights and still read from the same units at inference. Using the same MLP-neuron definition as above, we rank neurons in each layer by their average activation magnitude on each domain, select the top 10% most

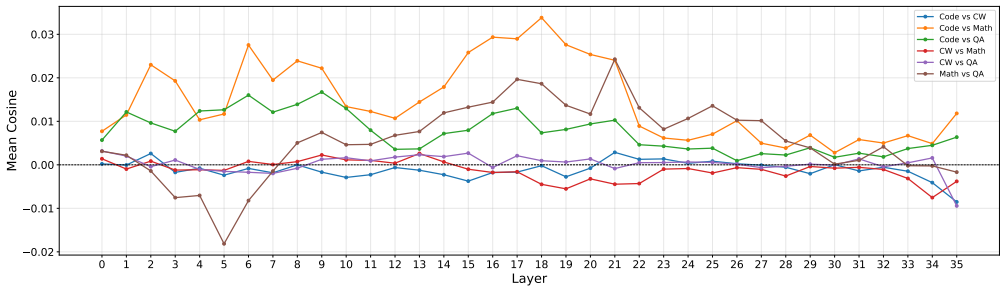


Figure 4: Layer-wise average directional cosine on shared top-changed neurons across domain pairs.

active neurons, and compute the pairwise Jaccard overlap of these active-neuron sets across domains. Full metric definitions are provided in Appendix B.4.

Figure 3b shows that active-neuron overlap among reasoning-oriented domains is far higher than changed-neuron overlap. Math-QA is the most coupled pair at 0.515, whereas CW overlaps with the other three domains only at 0.263–0.296. This ranking mirrors the training-time selective damage of Table 2: Math is damaged most by subsequent QA training, while CW is the most route-independent domain. We therefore localize cross-domain interference to sparse edits reaching another domain through the routes that both domains rely on at inference time.

4.5 Edit Direction Governs Transfer vs. Conflict

Section 4.4 identified shared active routes as the candidate site of interference, but overlap alone cannot distinguish transfer from damage: the missing ingredient is the *direction* in which each domain moves a shared unit. We therefore restrict attention to the units that both domains actually edit. For each domain pair we intersect the two top-10% changed-neuron sets and compute the cosine between the corresponding neuron-level update vectors (full definitions in Appendix B.5).

Figure 4 shows that domain pairs exhibit distinct directional patterns on these shared edited units. Code-Math is predominantly aligned, with the average cosine staying clearly positive across depth, consistent with the positive Code→Math transfer in Table 2. Math-QA, by contrast, is not globally antagonistic but splits by depth: the average cosine is negative in layers L3–6 and positive in layers L14–21. Direction, rather than overlap, is what separates the synergistic pair from the damaging one, and it does so at a granularity that the module-level gradient view of Section 4.1 cannot resolve.

Three localization signals. Together with the preceding subsections, this yields three measurable signals for selecting candidate repair coordinates: shared activation between the two domains (Section 4.4), the magnitude of the later-domain edit (Section 4.2), and directional conflict on the shared unit (this subsection). The same signals mark the coordinates that the direct intervention of Section 6.2 later reverts; what they imply for how damage forms, and whether it can be undone, is the subject of the next section.

5 A Local Perturbation Model of Interference and Repair

The structural observations of Section 4 motivate a local perturbation model of interference. Shared active routes define candidate parameter subspaces, while edit magnitudes and directions motivate examining how displacement interacts with the earlier domain's loss geometry. We analyze fixed smooth objectives to separate first-order drift from curvature-sensitive loss changes and derive conditions under which a short refresh reduces the displacement on a shared subspace. This framework connects the structure of a later-domain update to the local sensitivity of an earlier-domain objective.

5.1 Notation and Structural Assumptions

Let $L_d(\boldsymbol{\theta})$ be a local objective for domain d, with gradient $\boldsymbol{g}_d$ and Hessian $\boldsymbol{H}_d$. After training on domain A, later training on B induces displacement $\boldsymbol{\delta}_B$ from the reference checkpoint $\boldsymbol{\theta}^*_A$. We define

interference by

$$\Delta_{A\leftarrow B} = L_A(\boldsymbol{\theta}_A^* + \boldsymbol{\delta}_B) - L_A(\boldsymbol{\theta}_A^*).$$

The damage identities require local smoothness. Approximate stationarity bounds the first-order term, while the refresh result additionally assumes positive curvature on the relevant subspace, bounded coupling, and a region containing the iterates. These conditions are stated in Appendix C.1.

From shared routes to a parameter subspace. Each unit i in layer ℓ is associated with parameter coordinates $\mathcal{C}_i^{(\ell)}$ (gate and up rows and the down column for an MLP neuron; attention units analogously). For a set $R_{A,B}$ of shared active units edited by the later domain, define

$$S_{A,B} = \operatorname{span}\Big\{\boldsymbol{e}_j : j \in \bigcup_{(i,\ell)\in R_{A,B}} \mathcal{C}_i^{(\ell)}\Big\}, \tag{1}$$

and let $\boldsymbol{P}_S$ denote its orthogonal projector. The empirical selector is a coordinate proxy for such routes. Its actual rollback operator, defined below, also allows partial reversion of shared attention weights. We first express the total loss change in terms of drift and curvature (Proposition 1), decompose its curvature contribution along the rollback operator (Proposition 2), and then analyze refresh on $S_{A,B}$ (Theorem 1).

5.2 First-Order Drift and Curvature-Sensitive Damage

Consider first the total damage $\Delta_{A\leftarrow B}$. The relevant curvature is averaged along the displacement:

$$\bar{\boldsymbol{H}}_A = 2\int_0^1 (1-s)\boldsymbol{H}_A(\boldsymbol{\theta}_A^* + s\boldsymbol{\delta}_B)\,ds, \qquad \bar{\kappa}_A(\boldsymbol{u}) = \frac{\boldsymbol{u}^\top \bar{\boldsymbol{H}}_A \boldsymbol{u}}{\|\boldsymbol{\delta}_B\|_2^2}. \tag{2}$$

Proposition 1. *Under local smoothness,*

$$\Delta_{A\leftarrow B} = \underbrace{\boldsymbol{g}_A(\boldsymbol{\theta}_A^*)^\top \boldsymbol{\delta}_B}_{a} + \tfrac{1}{2}\|\boldsymbol{\delta}_B\|_2^2 \bar{\kappa}_A(\boldsymbol{\delta}_B). \tag{3}$$

If $\|\boldsymbol{g}_A(\boldsymbol{\theta}_A^*)\|_2 \le \varepsilon_A$, then $|a| \le \varepsilon_A\|\boldsymbol{\delta}_B\|_2$.

This identity retains both contributions. Curvature dominates when $|a|$ is small relative to the full curvature term; near-stationarity alone does not determine their relative sizes. Diagonalizing $\bar{\boldsymbol{H}}_A$ gives a curvature contribution $\frac{1}{2}\sum_k \bar{\lambda}_k c_k^2$, where c_k is the displacement along eigenvector k, so a small update in sensitive directions can cause damage even when global gradient alignment is weak. The path average accounts for curvature variation along the displacement. The proof is in Appendix C.2.

5.3 Damage Decomposition Along a Sparse Intervention

Proposition 1 separates first-order drift from the total curvature contribution. To resolve the latter along a sparse intervention, we split the displacement into the component removed by a structured reversion and the component retained. Let $\boldsymbol{W}$ be the diagonal rollback operator: its entries are one for fully reverted coordinates, k/g for shared key and value rows selected by k of their g query heads, and zero otherwise. Write

$$\boldsymbol{v} = \boldsymbol{W}\boldsymbol{\delta}_B, \qquad \boldsymbol{r} = (\boldsymbol{I} - \boldsymbol{W})\boldsymbol{\delta}_B, \qquad \boldsymbol{\delta}_B = \boldsymbol{v} + \boldsymbol{r}, \tag{4}$$

and define

$$Q_v = \boldsymbol{v}^\top \bar{\boldsymbol{H}}_A \boldsymbol{v}, \qquad C = \boldsymbol{v}^\top \bar{\boldsymbol{H}}_A \boldsymbol{r}, \qquad Q_r = \boldsymbol{r}^\top \bar{\boldsymbol{H}}_A \boldsymbol{r}. \tag{5}$$

The reverted and retained update components can share coordinates under partial rollback; no orthogonality is required.

Proposition 2. *Under local smoothness,*

$$\Delta_{A\leftarrow B} = a + \tfrac{1}{2}Q_v + C + \tfrac{1}{2}Q_r. \tag{6}$$

Consequently, if $Q_v > 0$*,*

$$\left| \frac{\Delta_{A \leftarrow B}}{Q_v/2} - 1 \right| \leq \underbrace{\frac{2|a|}{Q_v}}_{\eta_1} + \underbrace{\frac{2|C| + |Q_r|}{Q_v}}_{\eta_{\text{loc}}}. \tag{7}$$

Equation 7 bounds the discrepancy between the total loss change and the selected curvature contribution. The selected contribution approximates the total when the first-order term, the retained curvature, and the interaction term are small relative to Q_v. The decomposition thus identifies the contributions relevant to localization, rather than establishing it from sparsity alone. In particular, C captures interaction with the retained update: the selected component cannot generally be assessed in isolation. This is an accounting of the original displacement, not a rollback-gain formula, since the remaining displacement $\boldsymbol{r}$ follows a different path from the reference checkpoint. We evaluate the rollback's effect on task performance in Section 6.2 and examine the selected direction's local sensitivity in Section 6.3. The proof is in Appendix C.3.

5.4 A Conditional Explanation for Short Refresh

The preceding decomposition characterizes how the later-domain displacement changes the earlier objective. We now consider repair by resuming optimization on A for a short time. Starting from $\boldsymbol{\theta}_0 = \boldsymbol{\theta}_A^* + \boldsymbol{\delta}_B$, we model refresh as local gradient updates $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \boldsymbol{g}_A(\boldsymbol{\theta}_t)$. Positive curvature on $S_{A,B}$ and bounded coupling from its complement yield the following bound.

Theorem 1. *Under Assumptions 1–3 of Appendix C.1, with* $0 < \alpha \leq 1/\bar{\beta}_A$*,*

$$\|\boldsymbol{P}_S(\boldsymbol{\theta}_t - \boldsymbol{\theta}_A^*)\|_2 \leq (1 - \alpha\mu_A)^t \|\boldsymbol{P}_S \boldsymbol{\delta}_B\|_2 + \frac{\varepsilon_A + \xi_A R}{\mu_A}. \tag{8}$$

Here μ_A and $\bar{\beta}_A$ bound the projected average curvature, ξ_A bounds cross-subspace coupling, and R is an assumed radius containing the refresh trajectory. The bound combines a geometrically decaying term with an additive allowance for nonstationarity and coupling. This allowance controls the upper bound on the projected displacement; the actual displacement can be smaller. For a fixed admissible step size, larger μ_A gives faster decay, whereas stronger coupling increases the allowance. Thus, recovery depends on the subspace's loss geometry, not merely its size. The result gives a conditional explanation for rapid local recovery under gradient descent, not a quantitative guarantee for our adaptive-optimizer RL runs. Section 6.1 evaluates short refresh in our RL setting, and Appendix C.4 gives the proof.

With $G_A = \sup_{t<T} \|\boldsymbol{g}_A(\boldsymbol{\theta}_t)\|_2$, these idealized updates travel at most $\alpha T G_A$ from their starting point. Smoothness then controls changes to another domain.

Proposition 3. *If* L_C *is* β_C*-smooth and* $\|\boldsymbol{g}_C(\boldsymbol{\theta}_0)\|_2 \leq \varepsilon_C$*, a* T*-step refresh satisfies*

$$L_C(\boldsymbol{\theta}_T) - L_C(\boldsymbol{\theta}_0) \leq \varepsilon_C \alpha T G_A + \tfrac{1}{2}\beta_C \alpha^2 T^2 G_A^2. \tag{9}$$

This bound permits small collateral changes when the displacement and the other domain's local sensitivity are small. Longer refresh tightens the contraction bound but increases this collateral bound, motivating a trade-off rather than prescribing an optimal duration; we choose the refresh length empirically (Appendix D.2). Appendix C.5 gives the proof, and Appendix C.6 extends the first-order view to repeated alternating refresh over several domains. Propositions 1–2 characterize loss changes along a later-domain displacement, while Theorem 1 gives conditions for contracting its projection onto a shared subspace. We next evaluate refresh and rollback through task performance and probe the local sensitivity of the selected direction.

6 Task-Level Validation and Direct Intervention

We evaluate targeted repair through three experiments. Section 6.1 measures recovery of the damaged domain and changes to the other domains after a short refresh. Section 6.2 evaluates training-free rollback on a sparse coordinate proxy for $S_{A,B}$. Section 6.3 compares the Math objective's directional curvature along the selected rollback direction with matched random controls.

Table 2: Performance of single-domain experts, sequential training, mixed-domain baselines

Task	Base	Code_s	Math_s	QA_s	CW_s	Code_o	Math_o	QA_o	CW_o	CGPO	JT	Replay-all	Re-Math
Math	43.19	59.63	**66.84**	55.31	39.78	59.63	66.49	59.90	57.66	61.93	64.80	63.54	66.04
Code	29.57	**52.67**	34.65	32.07	28.15	**52.67**	50.69	50.99	50.47	50.05	48.61	51.09	51.05
QA	60.64	60.89	60.76	**63.31**	60.50	60.89	60.52	62.34	62.34	62.48	62.11	62.56	62.49
CW	82.44	82.40	81.38	81.76	86.24	82.40	81.44	81.79	86.52	86.73	**86.97**	86.08	85.96
AVG	53.96	63.90	60.91	58.11	53.67	63.90	64.79	63.76	64.25	65.30	65.62	65.82	**66.39**

6.1 Recovery by Short Refresh

Setup. Table 2 reports the single-domain experts, the sequential checkpoints, the mixed baselines (JT, CGPO, and a replay-all mixed refresh over all four domains from CW_o), and Re-Math, a short Math-only refresh from the same CW_o checkpoint (benchmark-level results in Appendix D.1).

Table 3: Selective rollback on a coordinate proxy ($\mathrm{QA}_o \to \mathrm{Math}_o$); Joint MLP+Attn adds attention coordinates. Recovery is the fraction of the Math accuracy drop recovered.

Selector	Budget	Math Avg	Recovery	Δ QA Avg
Math_o	–	66.49	100.0%	−1.81
QA_o	–	59.90	0.0%	0.00
Random	1.3%	59.49	-6.3%	+0.01
A×M	1.3%	60.53	9.6%	+0.04
A×C	1.3%	60.55	9.9%	−0.05
M×C	1.3%	61.11	18.3%	+0.04
A×M×C	1.3%	**61.25**	**20.4%**	−0.06
Joint MLP+Attn	9.8%	**63.68**	**57.4%**	−0.13

Selective interference and fast recovery. Math reaches 66.49 after Code → Math, close to its single-domain expert (66.84), then drops to 57.66 over the later QA and CW stages. A short Math refresh from CW_o restores Math to 66.04, with Code at 51.05, QA at 62.49, and CW at 85.96, giving the highest average score of 66.39. The large Math recovery and comparatively limited changes to the other domains support selective repair (Appendices D.2 and D.3).

Comparison with baselines. Re-Math achieves a higher Math score and macro-average than the evaluated joint and mixed-refresh baselines. Replay-all retains slightly higher scores on the other three domains, so the comparison is a task trade-off rather than uniform dominance. These results support targeted refresh for the damaged domain under our evaluated budgets, though the local theory establishes no general ordering between targeted and mixed training. The recovery pattern also holds when excluding the validation benchmark AIME25 (Appendix D.1).

Table 4: Normalized Math curvature density along the proxy $\boldsymbol{\delta}_S$ vs. two equal-budget random directions at four QA training checkpoints (units 10^{-3}).

Ckpt	$\kappa(\hat{S})$	$\kappa(R_1)$	$\kappa(R_g)$	$\hat{S}/R_1$	$\hat{S}/R_g$
Math_o	+3.23	+0.72	+0.43	**4.51**	**7.44**
step 150	+17.2	+13.4	+7.77	1.28	2.21
QA_o	−8.69	+7.47	+4.35	–	–
step 450	−4.28	−1.00	−0.73	4.26	5.86

Generalization across model families. The pattern replicates on a Phi-family model: under a Math→QA→Re-Math schedule, later QA training again degrades Math—by a smaller but directionally consistent margin—and a short refresh restores it with QA essentially unchanged (Appendix D.4).

6.2 Conflict Subspace Proxy

The refresh results demonstrate recoverability. We next test whether a sparse coordinate intervention can recover part of the QA-induced Math accuracy drop without further training. Using the structural signals of Section 4.5, we construct a coordinate proxy and directly evaluate its rollback response.

Concretely, we score each MLP neuron by shared activation under Math and QA (A), the magnitude of the QA update (M), and directional conflict between the Math and QA task vectors (C). Defining $\boldsymbol{\delta}_{Q|M} = \mathrm{QA}_o - \mathrm{Math}_o$, the intervention is $\boldsymbol{\theta}_{\mathrm{rev}} = \mathrm{QA}_o - \boldsymbol{W}\boldsymbol{\delta}_{Q|M}$. For MLP-only selection, $\boldsymbol{W} = \boldsymbol{P}_{\hat{S}}$ is a binary coordinate projector; the joint MLP+Attn version also permits fractional reversion of shared KV weights.

Table 3 shows selective recovery. The 1.3% A×M×C intervention recovers 20.4% of the QA-induced Math accuracy drop with a QA Avg change of -0.06, while the matched-budget random intervention

slightly reduces Math performance. At the same 1.3% budget, removing magnitude or directional conflict roughly halves recovery (A×C 9.9%, A×M 9.6%), while M×C remains close at 18.3%. Layer allocation also matters: activation-only ranking matches the full selector only when inheriting its layer budgets (Appendix D.5). The joint intervention at 9.8% recovers 57.4% with a QA change of -0.13. The two budgets use different selectors, so the comparison does not establish a monotone relation between budget and recovery.

6.3 Directional Curvature of the Conflict Proxy

We examine the Math objective's directional curvature along the selected rollback direction and matched random directions. These checkpoint-level probes characterize the local sensitivity of the repair direction.

Fixing the actual rollback direction $\boldsymbol{\delta}_S = \boldsymbol{W}\boldsymbol{\delta}_{Q|M}$ for the 9.8% joint selection, we measure $\kappa(\hat{S};\boldsymbol{\theta}) = \boldsymbol{\delta}_S^\top \boldsymbol{H}_{\mathrm{Math}}(\boldsymbol{\theta})\boldsymbol{\delta}_S / \|\boldsymbol{\delta}_S\|_2^2$ by centered finite differences of Math GRPO gradients on a fixed diagnostic batch. The control R_{l} matches per-layer, per-module unit counts, while R_{g} matches the total MLP and attention counts (Appendix D.6). Their ratios compare the selected direction with these controls at each checkpoint.

Higher curvature along the selected direction. At Math_o, curvature along $\boldsymbol{\delta}_S$ is $4.5\times$ that of R_{l} and $7.4\times$ that of R_{g} (Table 4). Reducing the probe step from 10^{-3} to 3×10^{-4} changes the absolute estimate, while the selected direction retains approximately $4\times$ and $5\times$ the curvature of the respective controls (Appendix D.6). The comparison with the layer-matched control supports sensitivity within layers as well as across them. Combined with the rollback results, these probes support the selected direction as a useful repair target.

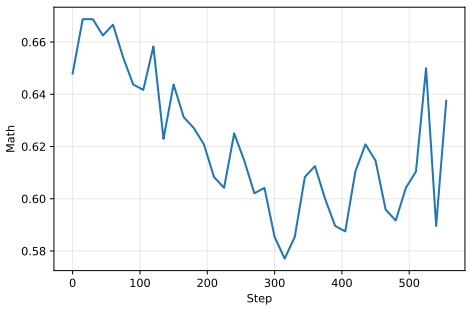


Figure 5: Math validation accuracy over QA training. Directional curvature at four checkpoints is reported in Table 4.

Curvature varies along the QA trajectory. The selected direction has positive Math curvature at Math_o and step 150, and negative curvature at QA_o and step 450. Table 4 and Figure 5 describe this evolving local geometry alongside the Math validation trajectory. These probes characterize curvature at individual training checkpoints. The loss decomposition in Proposition 2 uses the average Hessian along the parameter segment joining Math_o and QA_o, whose value also depends on curvature between the endpoints.

7 Conclusion

Our results support a local sensitivity account of cross-domain interference in multi-domain RL. Sparse edits, shared active routes, and directional alignment reveal structure that full-model gradient comparisons alone do not capture. The local perturbation analysis provides an exact loss decomposition and a conditional explanation for short refresh. Empirically, targeted refresh recovers the damaged Math domain with limited changes elsewhere, and sparse training-free rollback also restores a substantial part of the accuracy drop. Curvature probes provide complementary evidence that the selected directions are locally sensitive. Together, these findings motivate targeted intervention as a practical way to mitigate cross-domain degradation.

Our experiments use single training runs, so small score differences should be interpreted cautiously. The coordinate proxy is one practical intervention and does not identify a minimal conflict subspace. The theory is conditional on local smoothness, curvature, and coupling assumptions; its loss-level bounds are not quantitative predictions of benchmark accuracy. Theorem 1 and Proposition 3 analyze gradient descent on fixed local objectives, whereas our RL runs use an adaptive optimizer and a moving policy-gradient surrogate. Our finite-difference probes characterize checkpoint-level directional sensitivity, and their absolute magnitudes depend on the perturbation step. Measuring the path-averaged loss decomposition, extending the refresh analysis to adaptive policy optimization, and automatically selecting repair directions and budgets are directions for future work.

Ethics Statement

This work is a diagnostic study of cross-domain interference in multi-domain RL post-training. All experiments use a publicly available model (Qwen3-4B-Thinking-2507) and public datasets (OpenR1-Math, KlearReasoner-CodeSub-15K, SuperGPQA, and the Creative-Writing series), and involve no human subjects, private data, or personally identifiable information. Our analysis and the proposed targeted refresh introduce no new generative capabilities beyond standard domain RL; rather, they aim to make multi-domain post-training more stable and compute-efficient, which we regard as a positive broader impact. The standard risks of large language models—hallucination, and biased or unsafe generations—apply to the underlying model and are neither amplified nor specifically addressed here. We foresee no direct avenue for misuse of our findings.

Reproducibility Statement

We provide the setup for the reported experiments: data construction and evaluation benchmarks (Appendix A.1), training hyperparameters (Appendix A.2), and per-domain rewards (Appendix A.3). Definitions of changed neurons, active routes, and directional alignment are given in Appendices B.3, B.4, and B.5. The local theory's assumptions and proofs are stated in Appendix C. The rollback selector, finite-difference curvature estimator, and matched random controls are described in Appendices D.5 and D.6. We will release the training and analysis code together with the domain configurations.

AI Use Statement

In accordance with the ICLR 2027 AI Policy for Authors, we disclose the following uses of generative AI. Qwen3-235B-A22B-Instruct-2507 was used to generate replacement reference responses for 2,560 creative-writing training examples (Appendix A.1) and as an automated judge for the creative-writing reward during training and evaluation (Appendix A.3). Large language model assistants were additionally used for writing and coding assistance. The authors retain responsibility for the research design, experiments, verification of all claims and proofs, and the final content of this paper, including all AI-assisted text, code, data artifacts, and evaluations.

References

Art of Problem Solving. Aime problems and solutions. https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions, 2024a. Accessed: 2025-12-18.

Anthony Bau, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James R. Glass. Identifying and controlling important neurons in neural machine translation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=H1z-PsR5KX.

Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by doing: The role of on-policy data in mitigating forgetting. *CoRR*, abs/2510.18874, 2025. doi: 10.48550/ARXIV.2510.18874. URL https://doi.org/10.48550/arXiv.2510.18874.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 793–802. PMLR, 2018. URL http://proceedings.mlr.press/v80/chen18a.html.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA,*

USA, pp. 4299–4307, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/d5e2c0adad503c91f91df240d0cd4e49-Abstract.html.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. SFT memorizes, RL generalizes: A comparative study of foundation model post-training. *CoRR*, abs/2501.17161, 2025. doi: 10.48550/ARXIV.2501.17161. URL https://doi.org/10.48550/arXiv.2501.17161.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pp. 8493–8502. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.ACL-LONG.581. URL https://doi.org/10.18653/v1/2022.acl-long.581.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvalddson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms. *CoRR*, abs/2605.00674, 2026. URL https://arxiv.org/abs/2605.00674.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 3828–3850. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.211. URL https://doi.org/10.18653/v1/2024.acl-long.211.

Maggie Huan, Yuetai Li, Tuney Zheng, Xiaoyu Xu, Seungone Kim, Minxin Du, Radha Poovendran, Graham Neubig, and Xiang Yue. Does math reasoning improve general LLM capabilities? understanding transferability of LLM reasoning. *CoRR*, abs/2507.00432, 2025. doi: 10.48550/ARXIV.2507.00432. URL https://doi.org/10.48550/arXiv.2507.00432.

Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796, 2016. URL http://arxiv.org/abs/1612.00796.

Song Lai, Haohan Zhao, Rong Feng, Changyi Ma, Wenzhuo Liu, Hongbo Zhao, Xi Lin, Dong Yi, Min Xie, Qingfu Zhang, Hongbin Liu, Gaofeng Meng, and Fei Zhu. Reinforcement fine-tuning naturally mitigates forgetting in continual post-training. *CoRR*, abs/2507.05386, 2025. doi: 10.48550/ARXIV.2507.05386. URL https://doi.org/10.48550/arXiv.2507.05386.

Yongqi Leng and Deyi Xiong. Towards understanding multi-task learning (generalization) of LLMs via detecting and exploring task-specific neurons. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert (eds.), *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pp. 2969–2987. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.coling-main.200/.

Derek Li, Jiaming Zhou, Amirreza Kazemi, Qianyi Sun, Abbas Ghaddar, Mohammad Ali Alomrani, Liheng Ma, Yu Luo, Dong Li, Feng Wen, Jianye Hao, Mark Coates, and Yingxue Zhang. Omni-thinker: Scaling cross-domain generalization in llms via multi-task RL with hybrid rewards. *CoRR*, abs/2507.14783, 2025. doi: 10.48550/ARXIV.2507.14783. URL https://doi.org/10.48550/arXiv.2507.14783.

Xize Liang, Lin Yang, Jie Wang, Rui Liu, Yang Lu, Jinliang Zeng, Hanzhu Chen, Dong Li, and Jianye Hao. Boosting multi-domain reasoning of LLMs via curvature-guided policy optimization. In *International Conference on Learning Representations*, 2026.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 18878–18890, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/9d27fdf2477ffbff837d73ef7ae23db9-Abstract.html.

Michael Matena and Colin Raffel. Merging models with fisher-weighted averaging. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/70c26937fbf3d4600b69a129031b66ec-Abstract-Conference.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 525–536, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/432aca3a1e345e339f35a30c8f65edce-Abstract.html.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. RL's razor: Why online reinforcement learning forgets less. *CoRR*, abs/2509.04259, 2025. doi: 10.48550/ARXIV.2509.04259. URL https://doi.org/10.48550/arXiv.2509.04259.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems, EuroSys 2025, Rotterdam, The Netherlands, 30 March 2025 - 3 April 2025*, pp. 1279–1297. ACM, 2025. doi: 10.1145/3689031.3696075. URL https://doi.org/10.1145/3689031.3696075.

Dan Shi, Zhuowen Han, Simon Ostermann, Renren Jin, Josef van Genabith, and Deyi Xiong. Why does reinforcement learning generalize? a feature-level mechanistic study of post-training in large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, 2026.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, and Hao Wang. Continual learning of large language models: A comprehensive survey. *CoRR*, abs/2404.16789,

2024. doi: 10.48550/ARXIV.2404.16789. URL https://doi.org/10.48550/arXiv.2404.16789.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize with human feedback. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1f89885d556929e98d3ef9b86448f951-Abstract.html.

Zhenpeng Su, Leiyu Pan, Minxuan Lv, Yuntao Li, Wenping Hu, Fuzheng Zhang, Kun Gai, and Guorui Zhou. Ce-gppo: Controlling entropy via gradient-preserving clipping policy optimization in reinforcement learning, 2025. URL https://arxiv.org/abs/2509.20712.

Kimi Team. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599, 2025a. doi: 10.48550/ARXIV.2501.12599. URL https://doi.org/10.48550/arXiv.2501.12599.

M-A-P Team. Supergpqa: Scaling LLM evaluation across 285 graduate disciplines. *CoRR*, abs/2502.14739, 2025b. doi: 10.48550/ARXIV.2502.14739. URL https://doi.org/10.48550/arXiv.2502.14739.

Qwen Team. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025c. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.

Xiaozhi Wang, Kaiyue Wen, Zhengyan Zhang, Lei Hou, Zhiyuan Liu, and Juanzi Li. Finding skill neurons in pre-trained transformer-based language models. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pp. 11132–11152. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.EMNLP-MAIN.765. URL https://doi.org/10.18653/v1/2022.emnlp-main.765.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/ad236edc564f3e3156e1b2feafb99a24-Abstract-Datasets_and_Benchmarks_Track.html.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 23965–23998. PMLR, 2022. URL https://proceedings.mlr.press/v162/wortsman22a.html.

Yuning Wu, Jiahao Mei, Ming Yan, Chenliang Li, Shaopeng Lai, Yuran Ren, Zijia Wang, Ji Zhang, Mengyue Wu, Qin Jin, and Fei Huang. Writingbench: A comprehensive benchmark for generative writing. *CoRR*, abs/2503.05244, 2025. doi: 10.48550/ARXIV.2503.05244. URL https://doi.org/10.48550/arXiv.2503.05244.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/1644c9af28ab7916874f6fd6228a9bcf-Abstract-Conference.html.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3fe78a8acf5fda99de95303940a2420c-Abstract.html.

Junhao Zheng, Xidi Cai, Shengjie Qiu, and Qianli Ma. Spurious forgetting in continual learning of language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=ScI7IlKGdI.

Table 5: Training hyperparameters.

train batch size	ppo mini batch size	max prompt length	max response length	adv estimator	filter overlong prompts
256	256	2048	16384	grpo	True
lr	**use dynamic bsz**	**use kl loss**	**kl loss coef**	**kl loss type**	**clip ratio**
1e-6	True	False	0.0	low_var_kl	0.2
entropy coeff	**rollout n**	**rollout temperature**	**rollout top k**	**rollout top p**	**val top k**
0	8	1.0	-1	1.0	-1
val top p	**val temperature**	**val n**	**val do sample**	**use kl in reward**	
0.9	0.7	8	True	False	

A Implementation Details

A.1 Data Construction Details

We ensure that the prompt length of all training data is within 2,048. Each domain uses 5,120 training examples. Validation sets are constructed from non-training examples in the corresponding data sources. For Math, we use AIME25 (Art of Problem Solving, 2024a) as the validation set, which contains 30 problems. For Code, QA, and CW, each validation set contains 50 examples.

For CW, we sample 5,120 examples from the crownelius/Creative-Writing series.[2] The data come from four subsets: Sonnet4.6-800x,[3] Gemini3Pro-2700x,[4] Reasoning-KimiK2.5-600x,[5] and Qwen3.5Plus-2000x.[6] For 2,560 of these examples, we resample the reference responses using Qwen3-235B-A22B-Instruct-2507 and use the newly sampled responses as references.

For evaluation, LiveCodeBench-v6 contains 175 problems spanning January 2025 to April 2025. For QA evaluation, after excluding the training and validation data from SuperGPQA, we sample 10% of the remaining data, resulting in 2,141 test examples. Similarly, we sample 10% of MMLU-Pro, resulting in 1,203 test examples.

A.2 Training Hyperparameters

All experiments use the same GRPO training configuration. The full hyperparameters are summarized in Table 5.

At each training stage, we train the model until convergence and select the checkpoint with the optimal validation performance in the current training domain. Specifically, single-domain training adopts the best-performing checkpoint within the single domain, while multi-domain mixed training selects the checkpoint with the optimal average performance across all domains.

A.3 Reward Functions

Math and QA use answer-correctness rewards. The model is required to put the final answer in boxed. We use math_verify to parse the answer and match it against the reference answer. The reward is binary: 1 for a correct answer and 0 otherwise.

Code uses execution-based rewards, computed from the fraction of test cases passed.

CW uses an LLM-as-a-judge preference reward. For crownelius/Creative-Writing series, given a prompt, a reference response, and a model response, the judge compares their quality. If the model response is better, the reward is 1; if the two responses are comparable, the reward is 0.5; if the reference response is better, the reward is 0. For each query of WritingBench, the model generates one response, which is scored along multiple WritingBench evaluation dimensions. We use Qwen3-235B-A22B-Instruct-2507 (Team, 2025c) as the judge for training and evaluation.

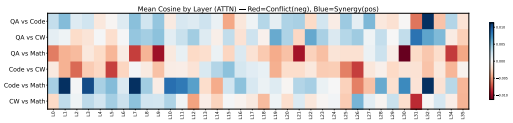
[2] https://huggingface.co/Crownelius

[3] https://huggingface.co/datasets/Crownelius/Creative-Writing-Sonnet4.6-800x

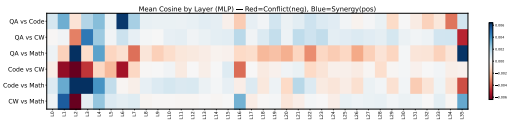
[4] https://huggingface.co/datasets/Crownelius/Creative-Writing-Gemini3Pro-2700x

[5] https://huggingface.co/datasets/Crownelius/Creative-Writing-Reasoning-KimiK2.5-600x

[6] https://huggingface.co/datasets/Crownelius/Creative-Writing-Qwen3.5Plus-2000x

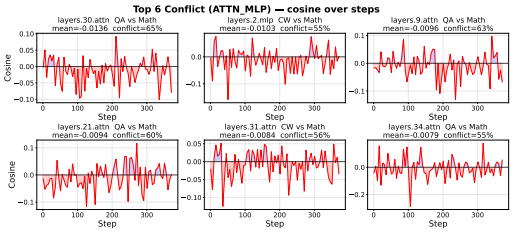


(a) Attention-module heatmap of pairwise gradient cosine.

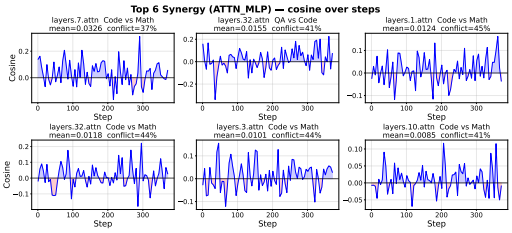


(b) MLP-module heatmap of pairwise gradient cosine.

Figure 6: Layer-wise heatmaps of pairwise gradient cosine in attention and MLP modules.



(a) Top six attention and MLP modules with the strongest conflict.



(b) Top six attention and MLP modules with the strongest synergy.

Figure 7: Module-level conflict and synergy trends across the most prominent attention and MLP modules.

B Additional Structural Analysis

B.1 Extended Module-Level Gradient Conflict and Synergy

Figure 1(b,c) in the main text tracks the single most conflicting and most synergistic module for the Math-QA pair over joint training. Figure 7 extends this to the six attention and MLP modules with the most prominent conflict and synergy, respectively, and Figure 6 presents the layer-wise heatmaps of pairwise gradient cosine in attention and MLP modules. These results show that conflicts are not uniformly distributed across all parameters but are concentrated in specific regions of the network. In particular, local conflicts are most significant for the Math-QA domain pair. In addition, certain local conflicts also exist between Code-CW. Nevertheless, despite relatively clear module interactions, the corresponding cosine values remain small and are largely orthogonal, which is why Section 4 treats this module-level decomposition only as a hint rather than as a localization of the interference.

B.2 Sequential Parameter Changes Remain Sparse

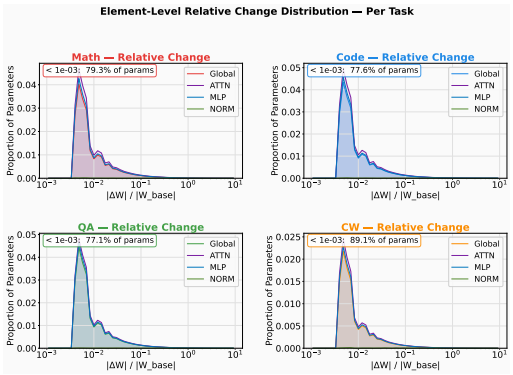
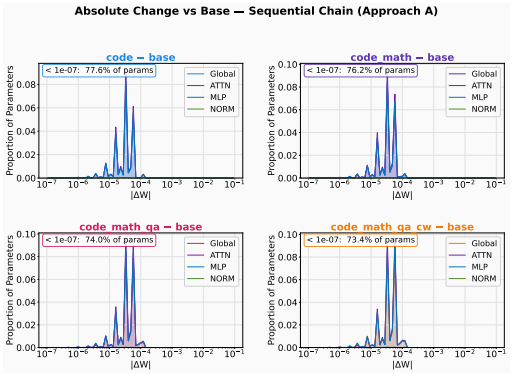
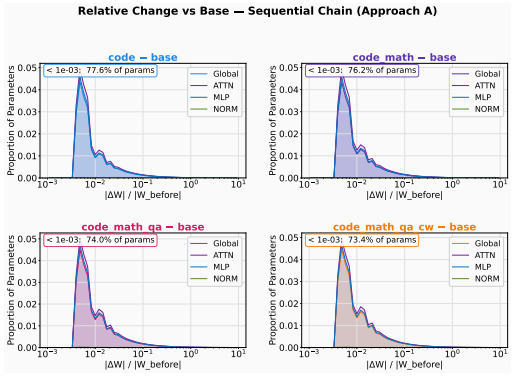


Figure 8: Relative parameter changes $r(\boldsymbol{W})$ of the four single-domain experts relative to the base model, the companion of Figure 2 omitted from Section 4.2.

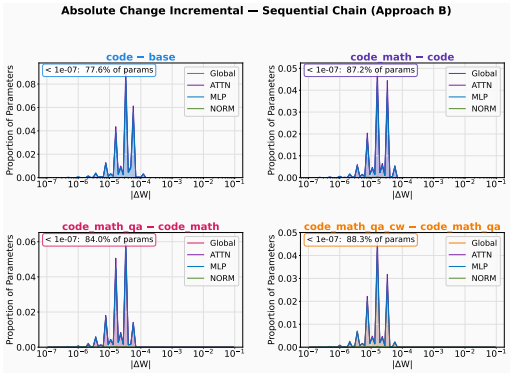


(a) Absolute cumulative parameter changes relative to the base model.

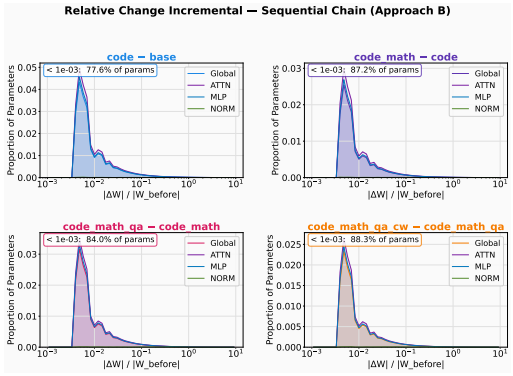


(b) Relative cumulative parameter changes relative to the base model.

Figure 9: Cumulative parameter-change distributions along the sequential domain RL chain relative to the base model.



(a) Absolute incremental parameter changes relative to the previous checkpoint.



(b) Relative incremental parameter changes relative to the previous checkpoint.

Figure 10: Incremental parameter-change distributions at each stage of the sequential domain RL chain.

Figure 8 gives the relative-change distributions that accompany the absolute changes reported in Section 4.2: the two views agree, with the bulk of the mass below $r(\boldsymbol{W}) = 10^{-3}$ for every expert, so neither the absolute nor the normalized scale reveals a global rewrite.

In addition to single-domain experts, we also measure parameter changes along the sequential training chain: Code $\rightarrow$ Math $\rightarrow$ QA $\rightarrow$ CW. We analyze both cumulative changes relative to the base model and incremental changes relative to the previous checkpoint.

As shown in Figure 9, cumulative deviation from the base model gradually increases as more domains are added. The fraction of parameters with $|\Delta W| < 10^{-7}$ and $r(W) < 10^{-3}$ decreases from 77.6% after Code training to 73.4% after Code-Math-QA-CW training.

However, as shown in Figure 10, each individual stage remains sparse. When comparing each checkpoint with the immediately preceding one, later domain stages modify only a small fraction of parameters: 87.2%, 84.0%, and 88.3% of parameters remain below the 10^{-7} threshold for the Math, QA, and CW stages, respectively.

These results show that sequential domain RL accumulates parameter shifts over domains, but each stage still acts as a sparse incremental update rather than a global parameter rewrite.

B.3 Neuron-Level Edit Definition

For each MLP layer ℓ, we define the i-th intermediate channel as one neuron. Its associated parameters consist of the corresponding row of the gate projection, the corresponding row of the up projection, and the corresponding column of the down projection:

$$\boldsymbol{g}_i^{(\ell)} = \boldsymbol{W}_{\text{gate}}^{(\ell)}[i,:], \quad \boldsymbol{u}_i^{(\ell)} = \boldsymbol{W}_{\text{up}}^{(\ell)}[i,:], \quad \boldsymbol{d}_i^{(\ell)} = \boldsymbol{W}_{\text{down}}^{(\ell)}[:,i]. \tag{10}$$

For domain d, let $\Delta\boldsymbol{g}_{d,i}^{(\ell)}$, $\Delta\boldsymbol{u}_{d,i}^{(\ell)}$, and $\Delta\boldsymbol{d}_{d,i}^{(\ell)}$ denote the corresponding parameter changes between the domain expert and the base model. We measure the edit magnitude of neuron i in layer ℓ by aggregating the squared parameter changes over these three components:

$$s_{d,i}^{(\ell)} = \left\|\Delta\boldsymbol{g}_{d,i}^{(\ell)}\right\|_2^2 + \left\|\Delta\boldsymbol{u}_{d,i}^{(\ell)}\right\|_2^2 + \left\|\Delta\boldsymbol{d}_{d,i}^{(\ell)}\right\|_2^2. \tag{11}$$

For each layer and each domain, we select the top 10% neurons with the largest edit magnitude:

$$\mathcal{N}_d^{(\ell)} = \text{Top}_{10\%}\left\{s_{d,i}^{(\ell)}\right\}_i. \tag{12}$$

The layer-wise overlap between two domains A and B is measured by the Jaccard coefficient:

$$J^{(\ell)}(A,B) = \frac{|\mathcal{N}_A^{(\ell)} \cap \mathcal{N}_B^{(\ell)}|}{|\mathcal{N}_A^{(\ell)} \cup \mathcal{N}_B^{(\ell)}|}. \tag{13}$$

Finally, we report the average overlap across the L MLP layers:

$$J(A,B) = \frac{1}{L}\sum_{\ell=1}^{L} J^{(\ell)}(A,B). \tag{14}$$

B.4 Active-Route Overlap Metric

Using the same MLP-neuron definition as in Appendix B.3, we measure whether two domains activate overlapping active routes during inference. For domain d and layer ℓ, let $h_{d,i,t}^{(\ell)}(x)$ denote the activation of neuron i at token position t for sample $x \in \mathcal{D}_d$, and let T_x be the number of tokens in x. We define the sample-level average activation magnitude as

$$\bar{a}_{d,i}^{(\ell)}(x) = \frac{1}{T_x}\sum_{t=1}^{T_x}\left|h_{d,i,t}^{(\ell)}(x)\right|. \tag{15}$$

The dataset-level activation score is

$$a_{d,i}^{(\ell)} = \frac{1}{|\mathcal{D}_d|}\sum_{x\in\mathcal{D}_d}\bar{a}_{d,i}^{(\ell)}(x). \tag{16}$$

For each layer and each domain, we select the top 10% neurons with the largest dataset-level activation score:

$$\mathcal{A}_d^{(\ell)} = \text{Top}_{10\%}\left\{a_{d,i}^{(\ell)}\right\}_i. \tag{17}$$

The layer-wise active-route overlap between two domains A and B is measured by the Jaccard coefficient:

$$J_{\text{act}}^{(\ell)}(A,B) = \frac{|\mathcal{A}_A^{(\ell)} \cap \mathcal{A}_B^{(\ell)}|}{|\mathcal{A}_A^{(\ell)} \cup \mathcal{A}_B^{(\ell)}|}. \tag{18}$$

Finally, we report the average active-route overlap across the L MLP layers:

$$J_{\text{act}}(A,B) = \frac{1}{L}\sum_{\ell=1}^{L} J_{\text{act}}^{(\ell)}(A,B). \tag{19}$$

B.5 Directional Alignment on Shared Edited Neurons

For each domain d, layer ℓ, and MLP neuron i, we define the neuron-level update vector by concatenating the parameter changes associated with its gate, up, and down projections:

$$\Delta \boldsymbol{v}_{d,i}^{(\ell)} = \left[\Delta \boldsymbol{g}_{d,i}^{(\ell)}; \Delta \boldsymbol{u}_{d,i}^{(\ell)}; \Delta \boldsymbol{d}_{d,i}^{(\ell)}\right]. \tag{20}$$

For two domains A and B, we define the shared edited neuron set in layer ℓ as

$$\mathcal{Q}_{A,B}^{(\ell)} = \mathcal{N}_A^{(\ell)} \cap \mathcal{N}_B^{(\ell)}, \tag{21}$$

where $\mathcal{N}_d^{(\ell)}$ is the Top-10% changed-neuron set defined in Appendix B.3. For each shared edited neuron, we compute the cosine similarity between the two domain update vectors:

$$c_{A,B,i}^{(\ell)} = \frac{\langle \Delta \boldsymbol{v}_{A,i}^{(\ell)}, \Delta \boldsymbol{v}_{B,i}^{(\ell)} \rangle}{\|\Delta \boldsymbol{v}_{A,i}^{(\ell)}\|_2 \|\Delta \boldsymbol{v}_{B,i}^{(\ell)}\|_2}. \tag{22}$$

The layer-wise average directional alignment on shared edited neurons is

$$C^{(\ell)}(A,B) = \frac{1}{|\mathcal{Q}_{A,B}^{(\ell)}|} \sum_{i \in \mathcal{Q}_{A,B}^{(\ell)}} c_{A,B,i}^{(\ell)}. \tag{23}$$

Positive values indicate that two domains edit their shared neurons in aligned directions on average, while negative values indicate conflicting directions.

Coverage of shared edited neurons by shared active routes. To assess how closely the shared edited units in Section 4.5 correspond to shared active routes, we compute the coverage rate

$$\mathrm{Cov}(A,B) = \frac{\sum_{\ell=1}^{L} \left|\mathcal{Q}_{A,B}^{(\ell)} \cap \mathcal{A}_A^{(\ell)} \cap \mathcal{A}_B^{(\ell)}\right|}{\sum_{\ell=1}^{L} \left|\mathcal{Q}_{A,B}^{(\ell)}\right|}, \tag{24}$$

where $\mathcal{A}_d^{(\ell)}$ is the Top-10% active-neuron set of Appendix B.4. Averaged over all domain pairs and all layers, $\mathrm{Cov} = 97.39\%$: almost every unit that two domains both edit heavily is also a unit that both domains activate heavily. This supports analyzing directional alignment on these units as a diagnostic of shared routes. The remaining 2.61% lies outside the shared top-active intersection.

C Detailed Proof of Our Theory

C.1 Structural Assumptions

The analysis uses a fixed local objective L_d for each domain. The diagonal operator $\boldsymbol{W}$ defines the actual rollback, with $\boldsymbol{v} = \boldsymbol{W}\boldsymbol{\delta}_B$ and $\boldsymbol{r} = (\boldsymbol{I} - \boldsymbol{W})\boldsymbol{\delta}_B$. The orthogonal projector $\boldsymbol{P}_S$ is used separately in the idealized refresh analysis. The following assumptions state the conditions of the bounds; we do not report empirical estimates of all their constants.

Assumption 1: local smoothness. Each L_d is twice continuously differentiable on a neighborhood containing the relevant segments and iterates, with $\|\boldsymbol{H}_d(\boldsymbol{\theta})\|_2 \le \beta_d$. The path average

$$\bar{\boldsymbol{H}}_A = 2\int_0^1 (1-s)\boldsymbol{H}_A(\boldsymbol{\theta}_A^* + s\boldsymbol{\delta}_B)\, ds \tag{25}$$

defines the exact curvature term in Propositions 1 and 2.

Assumption 2: approximate stationarity. At the reference point,

$$\|\boldsymbol{g}_A(\boldsymbol{\theta}_A^*)\|_2 \le \varepsilon_A. \tag{26}$$

This condition controls the first-order upper bound and the additive term in the refresh bound. The damage identities themselves retain the signed drift a and do not require it to vanish.

Assumption 3: local refresh curvature, coupling, and radius. Let $\boldsymbol{e}_t = \boldsymbol{\theta}_t - \boldsymbol{\theta}_A^*$ and $\tilde{\boldsymbol{H}}_t = \int_0^1 \boldsymbol{H}_A(\boldsymbol{\theta}_A^* + s\boldsymbol{e}_t)\, ds$. Along the refresh trajectory, suppose $\|\boldsymbol{e}_t\|_2 \le R$ and

$$\mu_A \boldsymbol{I}_S \preceq \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t \boldsymbol{P}_S\big|_{S_{A,B}} \preceq \bar{\beta}_A \boldsymbol{I}_S, \qquad \|\boldsymbol{P}_S \tilde{\boldsymbol{H}}_t (\boldsymbol{I} - \boldsymbol{P}_S)\|_2 \le \xi_A, \tag{27}$$

where $0 < \mu_A \le \bar{\beta}_A$. The step size satisfies $0 < \alpha \le 1/\bar{\beta}_A$. These are conditions on the relevant average curvature and cross-subspace coupling, not a claim of global convexity.

C.2 Proof of Proposition 1

We provide the full derivation for Proposition 1 and make explicit its sensitivity-based interpretation. Recall that training first reaches a domain-A checkpoint $\boldsymbol{\theta}_A^*$ and later domain-B training induces a local displacement $\boldsymbol{\delta}_B$, so the degraded checkpoint is $\boldsymbol{\theta}_A^* + \boldsymbol{\delta}_B$. The interference from B to A is

$$\Delta_{A \leftarrow B} = L_A(\boldsymbol{\theta}_A^* + \boldsymbol{\delta}_B) - L_A(\boldsymbol{\theta}_A^*). \tag{28}$$

Let $\phi(s) = L_A(\boldsymbol{\theta}_A^* + s\boldsymbol{\delta}_B)$ for $s \in [0, 1]$, which is twice continuously differentiable by Assumption 1 with $\phi'(s) = \boldsymbol{g}_A(\boldsymbol{\theta}_A^* + s\boldsymbol{\delta}_B)^\top \boldsymbol{\delta}_B$ and $\phi''(s) = \boldsymbol{\delta}_B^\top \boldsymbol{H}_A(\boldsymbol{\theta}_A^* + s\boldsymbol{\delta}_B)\boldsymbol{\delta}_B$. Integrating by parts twice, or equivalently applying Taylor's theorem with integral remainder to ϕ at $s = 0$,

$$\Delta_{A \leftarrow B} = \phi(1) - \phi(0) = \phi'(0) + \int_0^1 (1 - s)\, \phi''(s)\, ds. \tag{29}$$

Substituting $\phi'(0) = \boldsymbol{g}_A(\boldsymbol{\theta}_A^*)^\top \boldsymbol{\delta}_B$ and $\phi''(s) = \boldsymbol{\delta}_B^\top \boldsymbol{H}_A(\boldsymbol{\theta}_A^* + s\boldsymbol{\delta}_B)\boldsymbol{\delta}_B$ gives

$$\Delta_{A \leftarrow B} = \boldsymbol{g}_A(\boldsymbol{\theta}_A^*)^\top \boldsymbol{\delta}_B + \int_0^1 (1 - s)\, \boldsymbol{\delta}_B^\top \boldsymbol{H}_A(\boldsymbol{\theta}_A^* + s\boldsymbol{\delta}_B)\boldsymbol{\delta}_B\, ds. \tag{30}$$

By the definition of the path-averaged Hessian $\bar{\boldsymbol{H}}_A = 2\int_0^1 (1 - s)\boldsymbol{H}_A(\boldsymbol{\theta}_A^* + s\boldsymbol{\delta}_B)\, ds$ and of the normalized path curvature $\bar{\kappa}_A(\boldsymbol{u}) = \boldsymbol{u}^\top \bar{\boldsymbol{H}}_A \boldsymbol{u}/\|\boldsymbol{\delta}_B\|_2^2$, the second term equals $\frac{1}{2}\boldsymbol{\delta}_B^\top \bar{\boldsymbol{H}}_A \boldsymbol{\delta}_B = \frac{1}{2}\|\boldsymbol{\delta}_B\|_2^2\, \bar{\kappa}_A(\boldsymbol{\delta}_B)$, so

$$\Delta_{A \leftarrow B} = \boldsymbol{g}_A(\boldsymbol{\theta}_A^*)^\top \boldsymbol{\delta}_B + \tfrac{1}{2}\|\boldsymbol{\delta}_B\|_2^2\, \bar{\kappa}_A(\boldsymbol{\delta}_B). \tag{31}$$

This is the integral form of the loss change, with the first-order drift retained. Because the average Hessian depends on the displacement, the identity does not assert that loss scales quadratically over arbitrary displacement magnitudes.

Under Assumption 2 the checkpoint $\boldsymbol{\theta}_A^*$ is approximately stationary for domain A, which bounds the first term by

$$|\boldsymbol{g}_A(\boldsymbol{\theta}_A^*)^\top \boldsymbol{\delta}_B| \le \|\boldsymbol{g}_A(\boldsymbol{\theta}_A^*)\|_2 \|\boldsymbol{\delta}_B\|_2 \le \varepsilon_A \|\boldsymbol{\delta}_B\|_2, \tag{32}$$

and substituting into Equation 31 yields

$$\left|\Delta_{A \leftarrow B} - \tfrac{1}{2}\|\boldsymbol{\delta}_B\|_2^2\, \bar{\kappa}_A(\boldsymbol{\delta}_B)\right| \le \varepsilon_A \|\boldsymbol{\delta}_B\|_2, \tag{33}$$

which proves Proposition 1. The identity separates first-order drift from the curvature contribution without dropping a higher-order remainder, and it separates displacement magnitude $\|\boldsymbol{\delta}_B\|_2^2$ from directional sensitivity $\bar{\kappa}_A(\boldsymbol{\delta}_B)$. The norm-based bound on a may be loose: neither a small directional term nor a small gradient cosine alone fixes the relative size of the curvature contribution, so curvature dominates only when its full contribution is large relative to $|a|$.

C.3 Proof of Proposition 2

Since $\boldsymbol{\delta}_B = \boldsymbol{v} + \boldsymbol{r}$ and $\bar{\boldsymbol{H}}_A$ is symmetric,

$$\boldsymbol{\delta}_B^\top \bar{\boldsymbol{H}}_A \boldsymbol{\delta}_B = \boldsymbol{v}^\top \bar{\boldsymbol{H}}_A \boldsymbol{v} + 2\boldsymbol{v}^\top \bar{\boldsymbol{H}}_A \boldsymbol{r} + \boldsymbol{r}^\top \bar{\boldsymbol{H}}_A \boldsymbol{r} = Q_v + 2C + Q_r. \tag{34}$$

Substitution into Proposition 1 gives Equation 6. For $Q_v > 0$, division by $Q_v/2$ and the triangle inequality yield

$$\left|\frac{\Delta_{A \leftarrow B}}{Q_v/2} - 1\right| = \left|\frac{2a + 2C + Q_r}{Q_v}\right| \le \frac{2|a|}{Q_v} + \frac{2|C| + |Q_r|}{Q_v}, \tag{35}$$

which proves Equation 7. No orthogonality or idempotence of $\boldsymbol{W}$ is required: all three quadratic forms use the same path-averaged Hessian. The decomposition retains the interaction between the reverted and retained components.

C.4 Proof of Theorem 1

Let

$$\boldsymbol{e}_t = \boldsymbol{\theta}_t - \boldsymbol{\theta}_A^*, \quad \boldsymbol{e}_t^S = \boldsymbol{P}_S \boldsymbol{e}_t. \tag{36}$$

The refresh iteration is

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \boldsymbol{g}_A(\boldsymbol{\theta}_t), \tag{37}$$

which implies

$$\boldsymbol{e}_{t+1} = \boldsymbol{e}_t - \alpha \boldsymbol{g}_A(\boldsymbol{\theta}_t). \tag{38}$$

Projecting onto the shared conflict subspace gives

$$\boldsymbol{e}_{t+1}^S = \boldsymbol{P}_S \boldsymbol{e}_t - \alpha \boldsymbol{P}_S \boldsymbol{g}_A(\boldsymbol{\theta}_t) = \boldsymbol{e}_t^S - \alpha \boldsymbol{P}_S \boldsymbol{g}_A(\boldsymbol{\theta}_t). \tag{39}$$

By the mean value theorem for vector-valued maps, which Assumption 1 licenses on the segment from $\boldsymbol{\theta}_A^*$ to $\boldsymbol{\theta}_t$, the gradient decomposes *exactly* as

$$\boldsymbol{g}_A(\boldsymbol{\theta}_t) = \boldsymbol{g}_A(\boldsymbol{\theta}_A^*) + \tilde{\boldsymbol{H}}_t \boldsymbol{e}_t, \qquad \tilde{\boldsymbol{H}}_t = \int_0^1 \boldsymbol{H}_A(\boldsymbol{\theta}_A^* + s\boldsymbol{e}_t)\, ds, \tag{40}$$

with no remainder, so no Hessian-Lipschitz constant and no quadratic error term enter the recursion. Substituting into the projected recursion gives

$$\boldsymbol{e}_{t+1}^S = \boldsymbol{e}_t^S - \alpha \boldsymbol{P}_S \boldsymbol{g}_A(\boldsymbol{\theta}_A^*) - \alpha \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t \boldsymbol{e}_t. \tag{41}$$

Decompose $\boldsymbol{e}_t = \boldsymbol{e}_t^S + \boldsymbol{e}_t^\perp$ with $\boldsymbol{e}_t^\perp = (\boldsymbol{I} - \boldsymbol{P}_S)\boldsymbol{e}_t$. Then

$$\boldsymbol{P}_S \tilde{\boldsymbol{H}}_t \boldsymbol{e}_t = \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t \boldsymbol{P}_S \boldsymbol{e}_t + \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t (\boldsymbol{I} - \boldsymbol{P}_S)\boldsymbol{e}_t, \tag{42}$$

and the projected recursion becomes

$$\boldsymbol{e}_{t+1}^S = \left(\boldsymbol{I} - \alpha \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t \boldsymbol{P}_S\right) \boldsymbol{e}_t^S - \alpha \boldsymbol{P}_S \boldsymbol{g}_A(\boldsymbol{\theta}_A^*) - \alpha \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t (\boldsymbol{I} - \boldsymbol{P}_S)\boldsymbol{e}_t. \tag{43}$$

Assumptions 2 and 3 bound the three terms on the right in turn. First, since $\mu_A \boldsymbol{I}_S \preceq \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t \boldsymbol{P}_S|_{S_{A,B}} \preceq \bar{\beta}_A \boldsymbol{I}_S$ and $0 < \alpha \leq 1/\bar{\beta}_A$, every eigenvalue of $\boldsymbol{I} - \alpha \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t \boldsymbol{P}_S$ restricted to $S_{A,B}$ lies in $[0, 1 - \alpha\mu_A]$, so as an operator on $S_{A,B}$

$$\left\| \boldsymbol{I} - \alpha \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t \boldsymbol{P}_S \right\|_{2,S} \leq 1 - \alpha\mu_A. \tag{44}$$

Second, by Assumption 2, $\|\boldsymbol{P}_S \boldsymbol{g}_A(\boldsymbol{\theta}_A^*)\|_2 \leq \|\boldsymbol{g}_A(\boldsymbol{\theta}_A^*)\|_2 \leq \varepsilon_A$. Third, by the coupling bound of Assumption 3 together with the trust region $\|\boldsymbol{e}_t\|_2 \leq R$, which controls $\|\boldsymbol{e}_t^\perp\|_2 \leq R$,

$$\left\| \boldsymbol{P}_S \tilde{\boldsymbol{H}}_t (\boldsymbol{I} - \boldsymbol{P}_S)\boldsymbol{e}_t \right\|_2 \leq \xi_A \|\boldsymbol{e}_t^\perp\|_2 \leq \xi_A R. \tag{45}$$

Taking norms in Equation 43 therefore yields a recursion in which the perturbation is a constant rather than a function of the iterate being bounded:

$$\|\boldsymbol{e}_{t+1}^S\|_2 \leq (1 - \alpha\mu_A)\|\boldsymbol{e}_t^S\|_2 + \alpha\big(\varepsilon_A + \xi_A R\big). \tag{46}$$

Unrolling and using $\boldsymbol{e}_0 = \boldsymbol{\delta}_B$, so that $\boldsymbol{e}_0^S = \boldsymbol{P}_S \boldsymbol{\delta}_B$, together with $\sum_{k=0}^{t-1}(1 - \alpha\mu_A)^k \leq 1/(\alpha\mu_A)$, gives

$$\|\boldsymbol{P}_S(\boldsymbol{\theta}_t - \boldsymbol{\theta}_A^*)\|_2 \leq (1 - \alpha\mu_A)^t \|\boldsymbol{P}_S \boldsymbol{\delta}_B\|_2 + \frac{\varepsilon_A + \xi_A R}{\mu_A}, \tag{47}$$

which proves the theorem.

The additive term $(\varepsilon_A + \xi_A R)/\mu_A$ is an allowance in the upper bound, not a residual the iterate must incur: actual projected errors can be smaller, including convergence to zero when the allowance is positive.

C.5 Proof of Proposition 3

Let $\boldsymbol{r}_T = \boldsymbol{\theta}_T - \boldsymbol{\theta}_0$. Smoothness and $\|\boldsymbol{g}_C(\boldsymbol{\theta}_0)\|_2 \leq \varepsilon_C$ give

$$L_C(\boldsymbol{\theta}_T) - L_C(\boldsymbol{\theta}_0) \leq \boldsymbol{g}_C(\boldsymbol{\theta}_0)^\top \boldsymbol{r}_T + \tfrac{1}{2}\beta_C \|\boldsymbol{r}_T\|_2^2 \leq \varepsilon_C \|\boldsymbol{r}_T\|_2 + \tfrac{1}{2}\beta_C \|\boldsymbol{r}_T\|_2^2. \tag{48}$$

For the gradient updates considered in Theorem 1,

$$\|\boldsymbol{r}_T\|_2 = \left\| -\alpha \sum_{t<T} \boldsymbol{g}_A(\boldsymbol{\theta}_t) \right\|_2 \leq \alpha \sum_{t<T} \|\boldsymbol{g}_A(\boldsymbol{\theta}_t)\|_2 \leq \alpha T G_A, \tag{49}$$

where $G_A = \sup_{t<T} \|\boldsymbol{g}_A(\boldsymbol{\theta}_t)\|_2$. Substitution proves Equation 9. The bound depends on displacement and local sensitivity; its size is not determined by the refresh length alone.

Table 6: Full benchmark-level results; Step is the selected checkpoint step and AVG averages the four evaluation domains (Math, Code, QA, CW).

Model	Step	Math						Code	QA			CW	AVG
		AIME24	AIME25	AIME26	OlympiadBench	HMMT	Math	LiveCodeBench-v6	SuperGPQA-test	MMLU-Pro	QA	WritingBench	
Base	–	48.75	42.50	40.63	61.68	22.40	43.19	29.57	47.32	73.96	60.64	82.44	53.96
Math_s	525	71.77	67.71	69.79	77.95	46.98	66.84	34.65	47.84	73.68	60.76	81.38	60.91
Code_s	600	61.46	61.46	62.71	71.38	41.15	59.63	52.67	47.84	73.93	60.89	82.40	63.90
QA_s	705	58.65	55.83	56.77	70.19	35.10	55.31	32.07	51.85	74.77	63.31	81.76	58.11
CW_s	120	44.27	38.85	35.00	59.74	21.04	39.78	28.15	47.16	73.84	60.50	86.24	53.67
Code_o	600	61.46	61.46	62.71	71.38	41.15	59.63	52.67	47.84	73.93	60.89	82.40	63.90
Math_o	180	73.33	66.56	70.83	77.57	44.17	66.49	50.69	47.53	73.51	60.52	81.44	64.79
QA_o	300	64.38	59.17	61.98	72.73	41.25	59.90	50.99	49.98	74.69	62.34	81.79	63.76
CW_o	135	62.29	56.04	58.65	70.89	40.42	57.66	50.47	49.91	74.77	62.34	86.52	64.25
CGPO	630	65.10	58.13	66.88	75.35	44.17	61.93	50.05	50.90	74.05	62.48	86.73	65.30
JT	1035	69.58	65.63	65.52	77.12	46.15	64.80	48.61	50.04	74.18	62.11	86.97	65.62
Replay-all	150	68.85	62.19	66.88	74.99	44.79	63.54	51.09	50.43	74.69	62.56	86.08	65.82
Re-Math	135	74.27	64.48	68.65	77.06	45.73	66.04	51.05	50.28	74.69	62.49	85.96	66.39
QA → Math	675	71.15	66.77	68.54	77.04	43.13	65.33	32.53	51.55	74.44	63.00	82.01	60.72
Math → CW	120	72.71	67.71	70.42	77.75	47.19	67.16	32.82	48.31	74.04	61.18	85.77	61.73

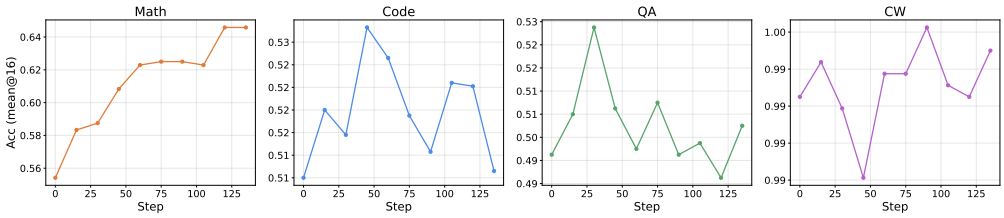


Figure 11: Validation dynamics during Re-Math refresh from CW_o across the four domains.

C.6 Extension: Alternating Refresh as Local Multi-Objective Optimization

A short alternating-refresh cycle has a useful first-order interpretation as a weighted local update. Consider two domains A and B. One cycle first updates on A:

$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \alpha_A \boldsymbol{g}_A(\boldsymbol{\theta}), \tag{50}$$

and then updates on B:

$$\boldsymbol{\theta}'' = \boldsymbol{\theta}' - \alpha_B \boldsymbol{g}_B(\boldsymbol{\theta}'). \tag{51}$$

For small step sizes, local smoothness gives

$$\boldsymbol{g}_B(\boldsymbol{\theta}') = \boldsymbol{g}_B(\boldsymbol{\theta}) + O(\alpha_A \|\boldsymbol{g}_A(\boldsymbol{\theta})\|_2). \tag{52}$$

Thus, one refresh cycle satisfies

$$\boldsymbol{\theta}'' = \boldsymbol{\theta} - \alpha_A \boldsymbol{g}_A(\boldsymbol{\theta}) - \alpha_B \boldsymbol{g}_B(\boldsymbol{\theta}) + O(\alpha_A \alpha_B). \tag{53}$$

To first order, alternating refresh therefore follows gradient descent on the weighted local objective

$$\Phi(\boldsymbol{\theta}) = \alpha_A L_A(\boldsymbol{\theta}) + \alpha_B L_B(\boldsymbol{\theta}). \tag{54}$$

More generally, alternating refresh over multiple domains approximates gradient descent on

$$\Phi(\boldsymbol{\theta}) = \sum_{d \in \mathcal{D}} \alpha_d L_d(\boldsymbol{\theta}). \tag{55}$$

A stationary point of this local objective satisfies

$$\sum_{d \in \mathcal{D}} \alpha_d \boldsymbol{g}_d(\boldsymbol{\theta}^\dagger) \approx 0, \tag{56}$$

which for positive weights $\{\alpha_d\}$ is the stationarity condition of the weighted objective. The cycle approximation motivates viewing alternating refresh as balancing local domain objectives; it does not establish convergence to that point or a Pareto improvement.

D Additional Validation Results

D.1 Full Task-Level Results

Table 6 reports the full benchmark-level results for all model checkpoints used in the main task-level validation. For sequential and refresh runs, the step number in parentheses denotes the selected checkpoint step within the last training stage.

Because AIME25 doubles as the Math validation set (Section 3), we also recompute the Math average over the four held-out Math benchmarks. The pattern is unchanged: Math is 66.48 at Math_o, falls to 60.09 at QA_o and 58.06 at CW_o, and returns to 66.43 after Re-Math, versus 63.88 for Replay-all and 64.59 for JT. The QA-induced drop measured this way is 6.39 points instead of 6.59, and the ordering of all settings is preserved.

D.2 Refresh Dynamics

Figure 11 shows rapid Math improvement early in Re-Math, followed by saturation, while the other domains change comparatively little. This behavior is consistent with local repair and the conditional interpretation of Theorem 1; the accuracy curves are not measurements of its parameter-space contraction bound.

Figure 12 provides a complementary Code → Math → Re-Code example. Early refresh improves Code with modest Math changes, while longer specialization produces a more visible Math decline. This supports using a limited refresh budget and is consistent with displacement-dependent sensitivity in Proposition 1.

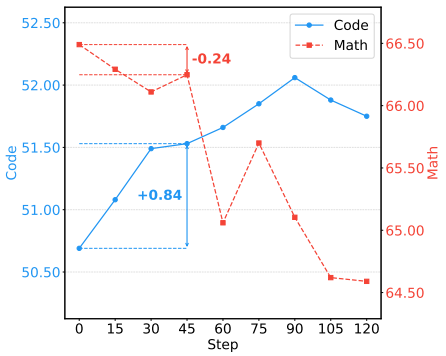


Figure 12: Math and Code performance changes during Re-Code on Math_o.

D.3 Directional Asymmetry of Interference

We additionally compare QA → Math with the reverse Math → QA ordering. As shown in Figure 13, in QA → Math, Math improves steadily during the Math stage while QA remains largely stable, unlike the stronger Math degradation observed when QA is trained after Math.

This provides further evidence that cross-domain interference is directional rather than symmetric. This asymmetry is also consistent with our local perturbation analysis and Proposition 1: the damage to an earlier domain depends on whether the later update enters its curvature-sensitive shared directions. Here, the Math update appears to perturb QA-sensitive directions only weakly, whereas the QA update more strongly perturbs Math-sensitive directions in the reverse ordering.

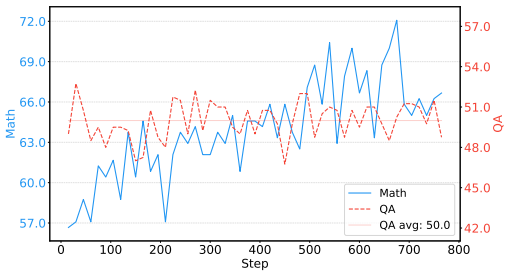


Figure 13: Validation dynamics for the reverse QA → Math ordering.

D.4 Generalization to Another Model Family

To test that our findings are not specific to the Qwen3-4B-Thinking model used in the main text, we replicate the core sequential-interference-and-refresh phenomenon on a Phi-4-mini-reasoning model under a Math → QA → Re-Math schedule (Table 7). RL on Math first lifts both Math (40.15 → 48.14) and QA (38.94 → 47.27). Subsequent QA training then improves QA to 53.40 while degrading Math to 47.56—a smaller but directionally consistent drop—reproducing the asymmetric cross-domain interference of Section 4. A short 45-step Re-Math refresh recovers Math to its best value (48.76, above the earlier Math peak) while leaving QA essentially unchanged (53.40 → 53.29). The recovery with little change in QA extends the task-level pattern across model families.

Table 7: Cross-family results on a Phi-4-mini-reasoning model; parentheses indicate selected checkpoint steps.

	Math						QA		
Checkpoint	AIME24	AIME25	AIME26	OlyBench	HMMT	Avg.	SuperGPQA	MMLU-Pro	Avg.
Base	46.46	31.88	37.40	63.32	21.67	40.15	26.12	51.76	38.94
+ Math (1065)	55.94	44.58	45.73	69.43	25.00	48.14	32.74	61.79	47.27
+ QA (390)	55.42	43.02	45.00	69.69	24.69	47.56	38.25	68.54	**53.40**
+ Re-Math (45)	**57.60**	**44.79**	**46.67**	**69.72**	25.00	**48.76**	38.09	68.48	53.29

D.5 Direct Rollback on a Coordinate Proxy for the Conflict Subspace

To more directly probe the localization claim behind Proposition 2, we intervene on the checkpoint pair $\mathrm{Code}_o \to \mathrm{Math}_o \to \mathrm{QA}_o$. Treating checkpoints as their parameter vectors, define the QA-induced displacement as

$$\boldsymbol{\delta}_{Q|M} = \mathrm{QA}_o - \mathrm{Math}_o. \tag{57}$$

We restrict the intervention to MLP neurons, using the same neuron definition as in Appendix B.3. For neuron i in layer ℓ, let $\Delta \boldsymbol{v}_{M,i}^{(\ell)}$ and $\Delta \boldsymbol{v}_{Q,i}^{(\ell)}$ denote the Math and QA task vectors in the concatenated gate/up/down parameterization from Appendix B.5.

We build a neuron-level proxy score

$$S_i^{(\ell)} = A_i^{(\ell)} M_i^{(\ell)} C_i^{(\ell)}, \tag{58}$$

where

$$A_i^{(\ell)} = \min\Big(\mathrm{Pct}(a_{M,i}^{(\ell)}), \mathrm{Pct}(a_{Q,i}^{(\ell)}) \Big), \tag{59}$$

$$M_i^{(\ell)} = \mathrm{Pct}\Big(\|\Delta \boldsymbol{v}_{Q,i}^{(\ell)}\|_2 \Big), \tag{60}$$

and

$$C_i^{(\ell)} = \max\Big(0, -\cos\Big(\Delta \boldsymbol{v}_{M,i}^{(\ell)}, \Delta \boldsymbol{v}_{Q,i}^{(\ell)}\Big)\Big). \tag{61}$$

Here $a_{d,i}^{(\ell)}$ is the dataset-level activation score defined in Appendix B.4, and $\mathrm{Pct}(\cdot)$ denotes percentile rank within a layer.

For activation collection, we use 512 samples and first generate up to 4096 tokens, followed by masked teacher-forcing forward passes to record MLP intermediate activations. Unless otherwise stated, the MLP-only selector uses a budget of 1.3%, and we write B for the corresponding number of selected units.

To distribute the global budget across layers, we first compute a layer score by averaging the top-ρ neuron scores within each layer, with $\rho = 10\%$. The final layer budget is then allocated proportionally to

$$\exp\Big((\bar{S}^{(\ell)})^{\gamma} \Big), \tag{62}$$

with a sharpening exponent $\gamma = 2.0$, followed by rounding so that $\sum_\ell B_\ell = B$.

Given the selected neuron set $\hat{S}$, we revert only the QA increment on its gate/up/down parameters:

$$\boldsymbol{\theta}_{\mathrm{rev}} = \mathrm{QA}_o - \boldsymbol{P}_{\hat{S}} \boldsymbol{\delta}_{Q|M}, \tag{63}$$

where $\boldsymbol{P}_{\hat{S}}$ is the coordinate projection induced by $\hat{S}$. This intervention requires no additional optimization and provides a direct proxy-level test of whether a sparse subset of the QA displacement carries measurable Math damage.

For single-factor selectors (A, M, or C), we report two variants. In the fixed-budget variant, we keep the layer allocation from the full $A \times M \times C$ selector and only change the within-layer ranking. In the recomputed-budget variant, we recompute the layer allocation from the corresponding selector itself.

Table 8: Additional ablations for the proxy conflict-subspace intervention at a 1.3% budget: fixed-budget variants keep the full selector's layer allocation and change only within-layer ranking, while recomputed-budget variants recompute both ranking and layer allocation.

Selector	AIME24	AIME25	AIME26	OlyBench	HMMT	Math Avg	Δ vs QA_o	Recovery	SuperGPQA	MMLU-Pro	QA Avg	Δ QA Avg
Math_o	73.33	66.56	70.83	77.57	44.17	66.49	+6.59	100.0%	47.53	73.51	60.52	−1.81
QA_o	64.38	59.17	61.98	72.73	41.25	59.90	0.00	0.0%	49.98	74.69	62.34	0.00
Random	63.54	59.90	60.00	73.05	40.94	59.49	−0.41	−6.3%	49.97	74.71	62.34	+0.01
A×M×C (main)	66.25	60.21	63.65	73.42	42.71	61.25	+1.35	**20.4%**	50.11	74.43	62.27	−0.06
M×C	66.67	60.00	61.77	73.36	43.75	61.11	+1.21	18.3%	50.16	74.58	62.37	+0.04
A×C	65.00	59.48	62.81	73.50	41.98	60.55	+0.65	9.9%	49.92	74.65	62.29	−0.05
A×M	65.73	61.46	61.77	72.97	40.73	60.53	+0.63	9.6%	50.25	74.49	62.37	+0.04
M (fixed budget)	64.69	60.10	63.12	73.50	42.40	60.76	+0.86	13.1%	49.93	74.56	62.25	−0.09
A (fixed budget)	65.52	59.90	63.75	73.23	43.85	61.25	+1.35	**20.4%**	49.93	74.55	62.24	−0.09
C (fixed budget)	65.83	60.21	61.77	73.30	42.60	60.74	+0.84	12.7%	49.94	74.59	62.27	−0.07
M (recomputed)	65.21	59.38	63.33	73.47	42.71	60.82	+0.92	13.9%	50.14	74.69	62.41	+0.08
A (recomputed)	66.15	60.42	61.88	73.41	43.12	61.00	+1.09	16.6%	50.18	74.56	62.37	+0.04
C (recomputed)	65.52	61.35	62.71	73.23	42.08	60.98	+1.08	16.3%	50.15	74.57	62.36	+0.03

Table 8 shows that the full selector recovers 20.4% of the QA-induced Math loss, while the two-factor selector $M \times C$ remains close at 18.3%. By contrast, removing either update magnitude or directional conflict reduces recovery to about half of the full result. Interestingly, A-only under the full selector's layer budget matches the full selector, whereas A-only with its own recomputed budget drops to 16.6%. This indicates that shared activation alone is not sufficient to explain the best-performing intervention; part of its apparent strength comes from inheriting the conflict-aware layer allocation.

Figure 14 shows that the depth profile of the intervention is not uniform. Once the conflict term is included, the layer budget becomes strongly U-shaped, concentrating more neurons in shallow and late layers; without the conflict term, the budget becomes much flatter. The selector $M \times C$ is the closest two-factor approximation to the full selector, whereas A-only can exhibit much smaller support overlap despite competitive recovery in the fixed-budget setting. This suggests that the proxy support is partly redundant: different sparse coordinate subsets can play similar functional roles as long as they target the harmful QA displacement.

The MLP-only results above show that a sparse rollback on proxy conflict coordinates can already recover a substantial part of the QA-induced Math loss, but the recovery saturates early and remains incomplete. To test whether the recoverable harmful displacement also extends beyond MLP coordinates, we extend the intervention to attention layers.

For layer ℓ, query head h, and row index r, we define a fine-grained attention unit

$$\mathrm{AttnUnit}(\ell, h, r) = \{\Delta W_Q^{(\ell)}[h, r, :],\ \Delta W_O^{(\ell)}[:, h, r],\ \tfrac{1}{\sqrt{g}}\Delta W_K^{(\ell)}[\kappa(h), r, :],\ \tfrac{1}{\sqrt{g}}\Delta W_V^{(\ell)}[\kappa(h), r, :]\}. \quad (64)$$

where $\kappa(h) = \lfloor h/g \rfloor$ maps a query head to its shared KV head under grouped-query attention, and $g = 4$ is the group size. Since each layer has 32 query heads and head dimension 128, this gives $32 \times 128 = 4096$ attention units per layer.

Unlike the MLP case, collecting per-unit activation statistics for attention would require decomposing attention outputs at the head-row level. We therefore use a conservative two-factor attention score

$$S_{\mathrm{attn}}^{(\ell,h,r)} = M_{\mathrm{attn}}^{(\ell,h,r)}\, C_{\mathrm{attn}}^{(\ell,h,r)}, \quad (65)$$

where M_{attn} measures the QA update magnitude of the unit and C_{attn} measures directional conflict between the Math and QA task vectors. MLP neurons keep the original three-factor score

$$S_{\mathrm{mlp}}^{(\ell,i)} = A_i^{(\ell)} M_i^{(\ell)} C_i^{(\ell)}. \quad (66)$$

For the joint MLP+Attn selector we keep the same budget convention, letting B denote the total number of selected units at a given budget. We first compute a joint layer score by combining the top-ρ scores from MLP and attention units:

$$s_\ell = \frac{\bar{S}_{\mathrm{mlp}}^{\rho}(\ell)\, d_{\mathrm{int}} + \bar{S}_{\mathrm{attn}}^{\rho}(\ell)\, n_{\mathrm{attn}}}{d_{\mathrm{int}} + n_{\mathrm{attn}}}, \quad (67)$$

with $\rho = 0.1$, $d_{\mathrm{int}} = 9728$, and $n_{\mathrm{attn}} = 4096$. The layer budget is then allocated by

$$b_\ell = B \cdot \frac{(s_\ell)^\gamma}{\sum_{\ell'} (s_{\ell'})^\gamma}, \quad \gamma = 2.0. \quad (68)$$

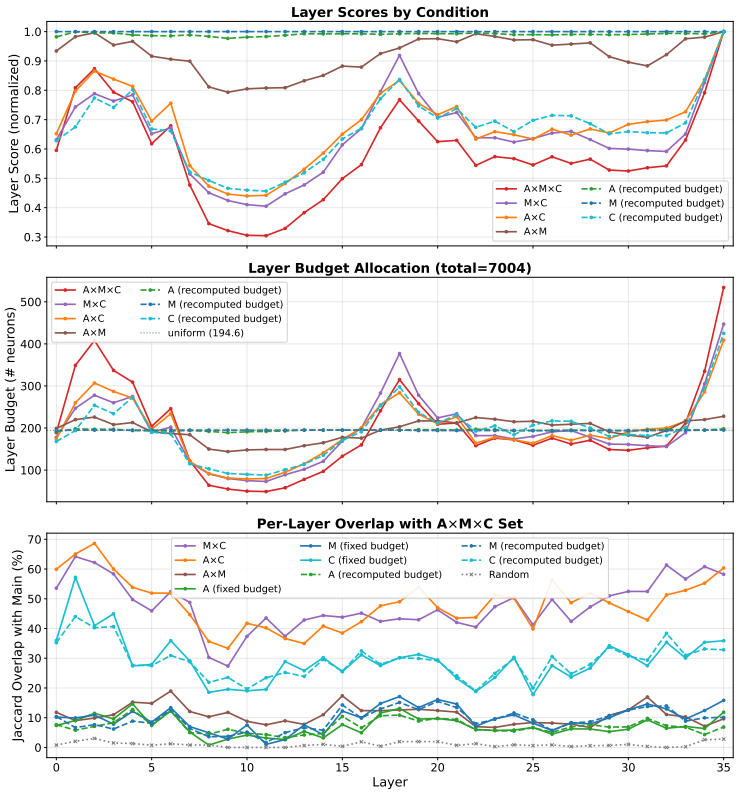


Figure 14: Layer-wise neuron selection across 36 layers—normalized layer scores (top), budget allocation (middle), and Jaccard overlap with the $A \times M \times C$ set (bottom)—where the conflict factor C dominates the non-uniform budget and two-factor combinations containing C recover the highest overlap.

Within each layer, we merge all MLP-neuron and attention-unit scores into a single ranked list and select the top-b_ℓ units regardless of type.

For selected MLP neurons, the revert operation is identical to the MLP-only setting. For a selected attention unit $\mathrm{AttnUnit}(\ell, h, r)$, we fully revert the corresponding W_Q and W_O slices. For shared KV weights, if k out of the g query heads attached to the same KV head select row r, we revert the corresponding W_K and W_V rows by fraction k/g.

At a 9.8% budget the joint selector recovers 57.4% of the QA-induced Math accuracy drop (Table 3), above the MLP-only result at a comparable budget. This supports including attention coordinates as additional repair targets. Recovery remains incomplete, and the accuracy measurements do not identify the location or curvature contribution of the remaining damage. Improving the selection of repair coordinates is a direction for future work.

D.6 Curvature Measurement Details

This section describes the directional curvature measurements in Section 6.3.

Probe directions. The proxy uses the 9.8% joint MLP+Attn selection and its actual revert operator $\boldsymbol{W}$. The direction $\boldsymbol{\delta}_S = \boldsymbol{W}\boldsymbol{\delta}_{Q|M}$, with $\boldsymbol{\delta}_{Q|M} = \mathrm{QA}_o - \mathrm{Math}_o$, is held fixed across checkpoints. Random controls use the same revert scheme: R_l matches per-layer, per-module unit counts, and R_g matches total MLP and attention counts. The squared direction norms are 0.0577 for $\hat{S}$, 0.0482 for R_l, and 0.0495 for R_g. The reported curvature divides by each direction's own squared norm.

Table 9: Sensitivity of the curvature estimate at Math_o to the finite-difference step size η.

η	$\kappa(\hat{S})$ at Math_o	$\hat{S}/R_{\mathrm{l}}$	$\hat{S}/R_{\mathrm{g}}$
3×10^{-4}	1.58×10^{-3}	3.96	5.09
1×10^{-3}	3.23×10^{-3}	4.51	7.44

Finite-difference estimator. The checkpoint-level probes of Section 6.3 compare directions at four points of the QA trajectory, so we need an estimator that is cheap enough to repeat across checkpoints and directions. We estimate the normalized curvature $\kappa(\hat{S};\boldsymbol{\theta})$ by a centered finite difference of Math GRPO gradients:

$$\kappa(\hat{S};\boldsymbol{\theta}) \approx \frac{\boldsymbol{\delta}_S^\top[\boldsymbol{g}_{\mathrm{Math}}(\boldsymbol{\theta}+h\boldsymbol{\delta}_S) - \boldsymbol{g}_{\mathrm{Math}}(\boldsymbol{\theta}-h\boldsymbol{\delta}_S)]}{2h\,\|\boldsymbol{\delta}_S\|_2^2}, \tag{69}$$

where $h = \eta/\|\boldsymbol{\delta}_S\|_2$ normalizes the perturbation to a fixed step size η in parameter space. Gradients are computed in fp32 on a fixed batch of Math GRPO data with identical rollouts and rewards across the paired forward-backward passes, so that the difference isolates the curvature term. Unless otherwise stated, we use $\eta = 10^{-3}$.

Checkpoints. We probe four saved checkpoints: Math_o, QA step 150, QA_o at step 300, and QA step 450. At each checkpoint, we use the same fixed rollback direction and matched controls to measure local directional sensitivity along the QA training trajectory.

Step-size sensitivity. Table 9 repeats the probe at Math_o with $\eta = 3 \times 10^{-4}$. The selected direction retains higher curvature than both random controls, while its absolute estimate changes from 3.23×10^{-3} to 1.58×10^{-3}. The corresponding curvature ratios are 3.96 and 5.09, compared with 4.51 and 7.44 at $\eta = 10^{-3}$. Both step sizes identify higher local sensitivity along the selected direction.