Rotation-Preserving Supervised Fine-Tuning

Anonymous Authors[1]

Abstract

Supervised fine-tuning (SFT) improves target-task performance but can rotate pretrained representations in ways that reduce out-of-domain generalization. We propose Rotation-Preserving Supervised Fine-Tuning (RPSFT), a simple regularizer that penalizes drift in the projected top-k singular-vector block of pretrained weight matrices. Across Llama and Qwen models trained on OpenR1-Math, RPSFT improves the in-domain/OOD trade-off over SFT, importance-weighted SFT, and Dynamic Fine-Tuning, and gives strong initializations for downstream RL fine-tuning. The results suggest that controlling dominant-subspace rotation is a practical way to retain pretrained structure while still allowing task adaptation.

1. Introduction

Large language models are commonly post-trained with supervised fine-tuning (SFT) followed by reinforcement learning fine-tuning (RLFT). SFT is useful for rapid task adaptation, but it can degrade capabilities outside the post-training domain (Zhu et al., 2025c; Wu et al., 2025). Recent work links this degradation to geometric drift from the pretrained initialization, especially rotations in dominant singular subspaces of important weight matrices (Jin et al., 2025; Shenfeld et al., 2025; Zhu et al., 2025b).

We ask whether SFT can retain its adaptation benefits while limiting destructive movement in these pretrained directions. Our answer is *Rotation-Preserving Supervised Fine-Tuning* (RPSFT), a soft regularizer that anchors the action of each tuned weight matrix inside the pretrained top-k singular-vector block. The method does not freeze parameters or remove gradients; it keeps full-parameter SFT expressive while discouraging drift in directions that are most likely to carry reusable pretrained structure.

We evaluate RPSFT on Llama and Qwen models trained on OpenR1-Math (Hugging Face, 2025). We define in-domain (ID) and out-of-domain (OOD) relative to the post-training data: math benchmarks are ID, while science QA, instruction-following, safety, factuality, and broad knowledge benchmarks are OOD. The detailed numeric tables are moved to Appendix A; the main paper keeps the central method and summary evidence.

2. Motivation

The central tension in post-training is that the supervised objective rewards movement toward the target distribution even when some of that movement disrupts features reused by other tasks. For a small update $\Delta\mathbf{W}$ to a pretrained matrix, a local expansion gives

$$\mathcal{L}(\mathbf{W}^0 + \Delta\mathbf{W}) \approx \mathcal{L}(\mathbf{W}^0) + \langle \nabla\mathcal{L}(\mathbf{W}^0), \Delta\mathbf{W}\rangle + \frac{1}{2}\Delta\mathbf{W}^\top \mathbf{H}\Delta\mathbf{W}. \quad (1)$$

The linear term captures task adaptation, while the curvature term penalizes movement in loss-sensitive directions. Computing and storing a full curvature model is impractical at LLM scale, so we need a compact coordinate system that identifies a small set of high-value directions to protect.

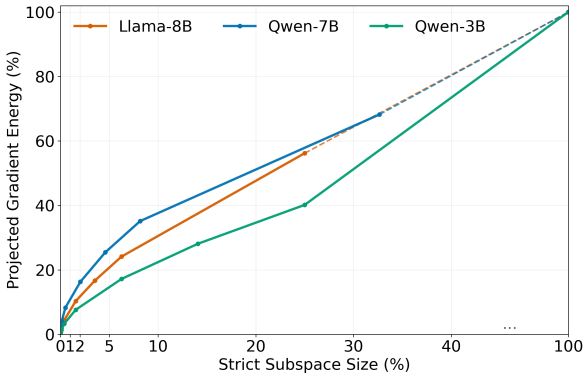


Figure 1. Fisher-projected gradient energy captured by strict top-$r \times r$ pretrained SVD blocks. A small protected block captures a disproportionate share of the training signal, motivating singular-subspace anchoring instead of full weight freezing.

Figure 1 shows that pretrained singular-vector blocks pro-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the ICML 2026 Workshop "Continual Adaptation at Scale: Towards Sustainable AI". Do not distribute.

vide such a coordinate system: for an early attention projection, a small strict top-$r \times r$ block captures much more Fisher-projected gradient energy than its raw dimensional fraction. This motivates preserving the action of tuned matrices inside dominant pretrained singular directions while leaving the remaining parameters free to adapt.

A complementary first-order diagnostic asks whether the SFT update direction remains aligned with ID and OOD gradients. Figure 2 shows that ID alignment alone does not guarantee OOD retention: the OOD signal can be weak or sign-changing across layers, especially for Qwen models. This is the regime where a soft geometric constraint is useful, because it limits movement in sensitive pretrained directions without blocking the whole update.

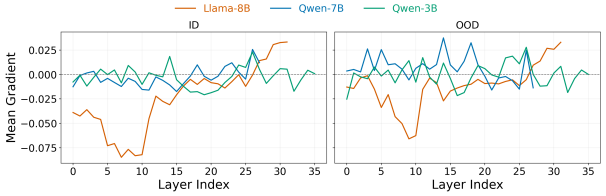


Figure 2. Layerwise first-order signal on the in-domain (left) and out-of-domain (right) distributions across Llama-8B, Qwen-7B, and Qwen-3B. Negative values indicate alignment with the local descent direction; positive values indicate conflict with the dataset gradient.

3. Method

Let θ_0 be the pretrained model and let $\mathbf{W}_\ell^0 \in \mathbb{R}^{m_\ell \times n_\ell}$ denote a pretrained linear-layer matrix selected for regularization. For each selected layer, we compute a truncated SVD once:

$$\mathbf{W}_\ell^0 = \mathbf{U}_\ell^0 \mathbf{\Sigma}_\ell^0 (\mathbf{V}_\ell^0)^\top, \tag{2}$$

and keep the top-k left and right singular vectors $\mathbf{U}_{0,\ell}^{(k)}$ and $\mathbf{V}_{0,\ell}^{(k)}$ fixed during training. For the current tuned matrix $\mathbf{W}_\ell$, RPSFT compares the projected block

$$S_\ell(\mathbf{W}_\ell) = (\mathbf{U}_{0,\ell}^{(k)})^\top \mathbf{W}_\ell \mathbf{V}_{0,\ell}^{(k)} \tag{3}$$

against its pretrained value $S_\ell^{\mathrm{ref}} = S_\ell(\mathbf{W}_\ell^0)$. The full-parameter objective is

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{task}}(\theta) + \lambda \sum_{\ell \in \mathcal{M}'} \left\| S_\ell(\mathbf{W}_\ell) - S_\ell^{\mathrm{ref}} \right\|_F^2, \tag{4}$$

where $\mathcal{M}'$ is the regularized matrix set and λ controls the penalty strength. The penalty is zero at initialization, vanishes to standard SFT at $k = 0$, and becomes full weight-space anchoring when k covers the full matrix rank. We use $\lambda = 1$ and default $k = 768$ unless stated otherwise; Appendix D.5 details the rank rule, cost, and boundary cases.

RPSFT differs from ordinary weight decay or full initialization anchoring because it only compares the action of $\mathbf{W}_\ell$ after projecting into pretrained singular coordinates. Updates can still use directions outside the protected block, and the protected block itself can move when the task loss is strong enough to overcome the soft penalty. In practice, each regularized matrix stores $\mathbf{U}_{0,\ell}^{(k)}$, $\mathbf{V}_{0,\ell}^{(k)}$, and S_ℓ^{ref} once before training; the extra computation is the two projections needed to form $S_\ell(\mathbf{W}_\ell)$. This makes the method compatible with standard full-parameter SFT and with later RL fine-tuning from the resulting checkpoint.

Training loop. Algorithm 1 summarizes the implementation. The SVD bases are fixed before training, and the projected-block penalty is added to the ordinary SFT loss at each regularization step. The full implementation details and PyTorch snippet are in Appendix B.

Algorithm 1 RPSFT training loop

Require: pretrained weights $\{\mathbf{W}_\ell^0\}$, selected matrices $\mathcal{M}'$, rank k, coefficient λ
For each $\ell \in \mathcal{M}'$, cache $\mathbf{U}_{0,\ell}^{(k)}$, $\mathbf{V}_{0,\ell}^{(k)}$, and S_ℓ^{ref}
for each SFT minibatch **do**
  compute task loss $\mathcal{L}_{\mathrm{task}}(\theta)$
  $\mathcal{L}_{\mathrm{reg}} \leftarrow \sum_{\ell \in \mathcal{M}'} \| (\mathbf{U}_{0,\ell}^{(k)})^\top \mathbf{W}_\ell \mathbf{V}_{0,\ell}^{(k)} - S_\ell^{\mathrm{ref}} \|_F^2$
  update θ with $\mathcal{L}_{\mathrm{task}} + \lambda \mathcal{L}_{\mathrm{reg}}$
end for

4. Experiments

Setup. We train the supervised stage on OpenR1-Math and initialize downstream DAPO (Yu et al., 2025) from each supervised checkpoint using DAPO-Math-17k. ID evaluation uses AIME24, AIME25, AMC23, MATH-500, Minerva Math, and OlympiadBench; OOD evaluation uses GPQA, IFEval, MMLU-Pro, SuperGPQA, Safety, and TruthfulQA. Baselines are standard SFT, importance-weighted SFT (IW), and Dynamic Fine-Tuning (DFT).

Summary. Figure 3 reports the central performance pattern. During SFT, RPSFT gives the strongest tuned ID averages across the three model families and reduces the OOD drop relative to competing tuned baselines. After DAPO, the RPSFT initializer remains strongest or near-strongest on ID and OOD averages, showing that the regularizer does not merely preserve the base model but produces useful RL starting points. Detailed supervised and RL benchmark tables are in Appendix Tables 2, 3, and 4.

Training cost. Table 1 reports measured SFT-stage throughput and memory. RPSFT adds modest overhead over vanilla SFT, but remains faster and less memory-intensive than IW and L2 Init.

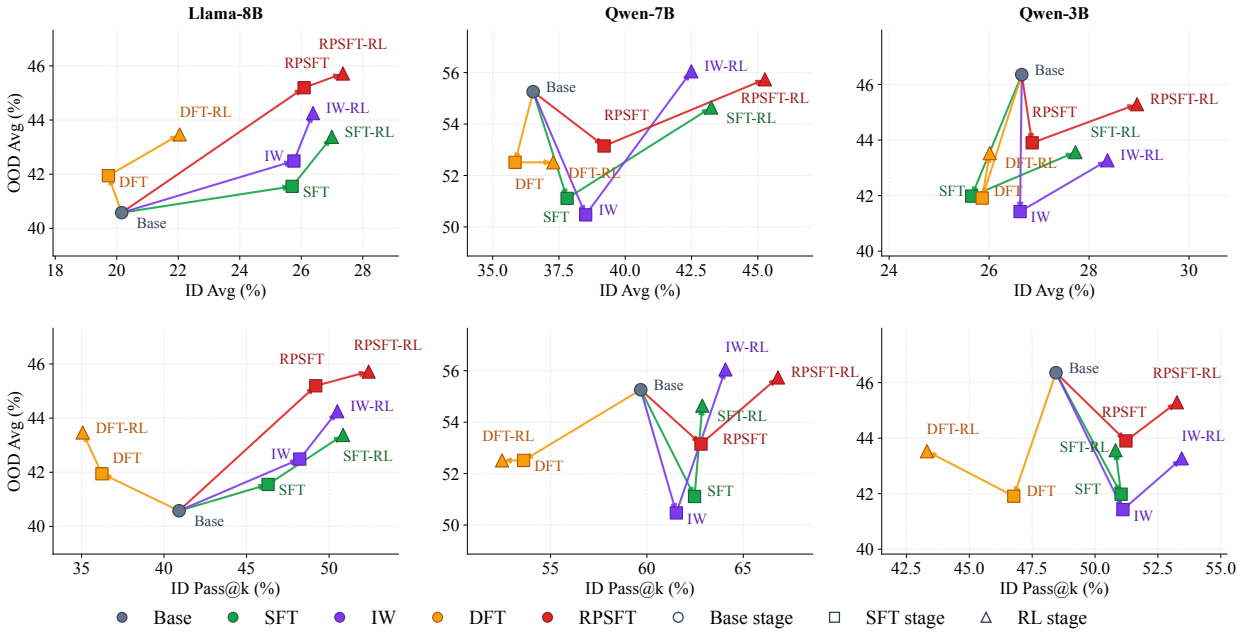


Figure 3. Overview of the two-stage post-training pipeline and evaluation split. We compare SFT, IW, DFT, and RPSFT after supervised fine-tuning and then use each checkpoint to initialize RL fine-tuning. ID and OOD scores average six benchmarks each; RPSFT is strongest or near-strongest in most summary averages.

Table 1. Empirical computation and memory cost for SFT baselines. Samples/s and Steps/s report throughput; peak memory is in GiB.

Method	Epochs	GPU-h	Samples/s	Steps/s	Peak alloc.	Peak reserved
SFT	12	156.3	4.368	0.273	32.4	125.4
DFT	12	159.0	4.294	0.268	40.0	128.1
IW	12	181.8	3.755	0.235	59.3	137.0
L2 Init	12	174.1	3.922	0.245	60.9	136.6
RPSFT	12	165.7	4.119	0.257	52.7	135.2

Geometric diagnostics. The performance trend is consistent with the geometric goal of the method. Figure 4 compares how much each fine-tuning method rotates the dominant left-singular subspaces of pretrained weights. RPSFT stays closer to the pretrained model across architectures, especially in middle layers and across the protected rank spectrum.

Representation drift. We also check activation-space retention; Appendix Figure 5 keeps the figure and its detailed interpretation together with the representation-drift metric.

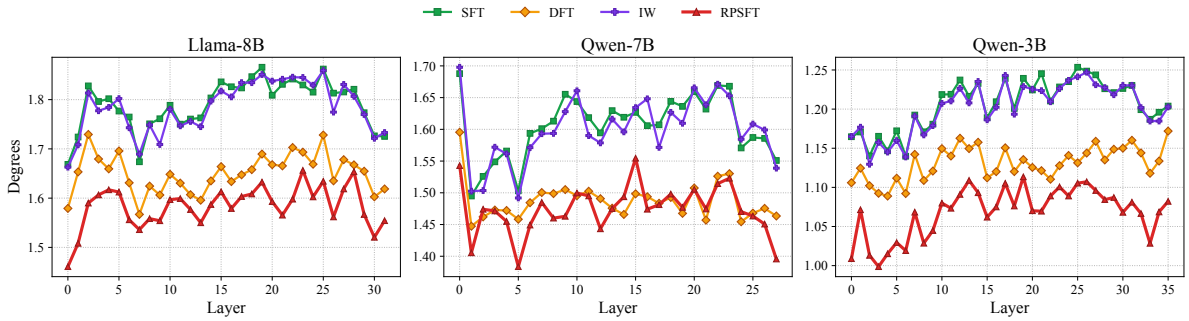
Rank robustness. Appendix Table 5 summarizes the protected-rank trend, and Appendix Tables 7 and 8 give the full sweeps. Very small ranks leave too much sensitive structure unconstrained, while full-rank anchoring approaches ordinary initialization regularization and weakens adaptation. The default $k = 768$ is therefore a practical middle ground: it protects a small high-energy block while keeping enough freedom for adaptation.

5. Discussion

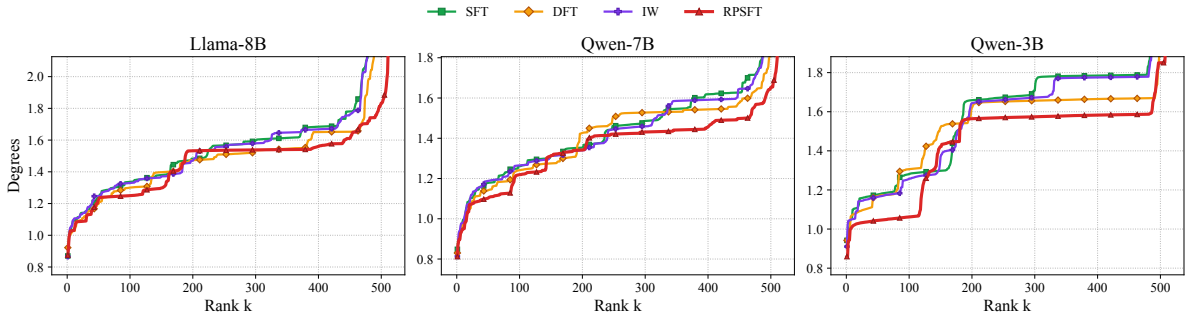
RPSFT is most useful when the supervised task update is only partially aligned with broad generalization. In that setting, full freezing is too restrictive and ordinary SFT is too permissive. The projected-block penalty gives an intermediate option: it protects the pretrained action inside a compact singular subspace, while the orthogonal complement remains available for task-specific adaptation. This is why the method can improve ID math performance without matching the OOD degradation of less structured SFT baselines.

The method also clarifies why the downstream RL results are not just a direct consequence of stronger SFT scores. RL fine-tuning can recover some forgotten behavior, but it still starts from the geometry of the supervised checkpoint. If SFT rotates dominant pretrained directions heavily, RL must first undo that movement before improving the reward objective. RPSFT gives RL a checkpoint that is already closer to the pretrained representation geometry, so reward optimization starts from a less distorted model.

The rank choice controls the adaptation-retention trade-off. Smaller ranks approach ordinary SFT; larger ranks approach full initialization anchoring. Our default protects only a small high-energy block, which is enough to reduce destructive rotation while avoiding the rigidity of full-rank penalties. The main limitation is that the best rank and layer set may depend on model scale and domain shift.



(a) Layerwise U-space rotation across model families.



(b) Rankwise U-space rotation for one case-study matrix per model.

Figure 4. Rotation diagnostics across model families. RPSFT consistently shows smaller geometric drift from the pretrained model, both across layers and through the protected rank spectrum.

6. Limitations

RPSFT assumes that pretrained dominant singular directions are useful coordinates for preserving reusable behavior. The diagnostics support this assumption for the model families and math post-training setting studied here, but it may not hold equally for every architecture, modality, or domain shift. A more aggressive distribution shift may require regularizing different layers, changing the protected rank, or combining the projected-block penalty with data mixing.

The method also adds SVD preprocessing and projection overhead for the selected matrices. This overhead is modest relative to full SFT when the protected rank is small, but it is not free, and full-rank settings become unnecessarily restrictive. In practice, RPSFT should be treated as a structured retention regularizer rather than a replacement for careful validation on both ID and OOD tasks.

7. Conclusion

RPSFT is a lightweight regularizer for SFT that anchors the projected top-k pretrained singular block while leaving the rest of the model free to adapt. Across model families and downstream RL initialization, it improves the adaptation/retention trade-off relative to standard SFT and strong regularized baselines.

The appendix rank analysis also gives a practical selection rule: choose the smallest protected rank whose strict SVD block captures a meaningful share of Fisher-projected training signal while remaining a small fraction of the full block. In our runs, the default $k = 768$ covers roughly a 5% strict block and captures about 20% of the early-layer gradient energy, which explains why it behaves as an intermediate constraint rather than ordinary full-weight anchoring.

More broadly, the results support a geometric view of post-training: forgetting is not only a matter of how far the model moves from initialization, but also which pretrained directions are rotated. Penalizing the projected dominant block is a simple way to bias SFT toward updates that preserve reusable structure. This makes RPSFT a practical candidate for post-training pipelines where target-domain improvement and broad retention must be optimized together.

Impact Statement

This paper presents a fine-tuning regularizer whose goal is to improve the adaptation-retention trade-off of large language models. Better retention and stronger RL initializations may improve downstream reliability when models are adapted to new domains. Deployment still requires model-use policies and safety evaluation appropriate to the base model and application.

References

Chung, W., Cherif, L., Meger, D., and Precup, D. Parseval regularization for continual reinforcement learning, 2024. URL https://arxiv.org/abs/2412.07224.

Elsayed, M., Lan, Q., Lyle, C., and Mahmood, A. R. Weight clipping for deep continual and reinforcement learning, 2024. URL https://arxiv.org/abs/2407.01704.

Foster, T., Sims, A., Forkel, J., Fellows, M., and Foerster, J. Learning to reason at the frontier of learnability, 2025. URL https://arxiv.org/abs/2502.12272.

Franke, J. K., Hefenbrock, M., and Hutter, F. Preserving principal subspaces to reduce catastrophic forgetting in fine-tuning. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024. URL https://openreview.net/forum?id=XoWtroECJU.

He, C., Luo, R., Bai, Y., Hu, S., Thai, Z. L., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., Liu, J., Qi, L., Liu, Z., and Sun, M. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems, 2024. URL https://arxiv.org/abs/2402.14008.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874.

Huan, M., Li, Y., Zheng, T., Xu, X., Kim, S., Du, M., Poovendran, R., Neubig, G., and Yue, X. Does math reasoning improve general llm capabilities? understanding transferability of llm reasoning, 2025. URL https://arxiv.org/abs/2507.00432.

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

Jin, H., Luan, S., Lyu, S., Rabusseau, G., Rabbany, R., Precup, D., and Hamdaqa, M. Rl fine-tuning heals ood forgetting in sft, 2025. URL https://arxiv.org/abs/2509.12235.

Kang, K., Setlur, A., Ghosh, D., Steinhardt, J., Tomlin, C., Levine, S., and Kumar, A. What do learning dynamics reveal about generalization in llm reasoning?, 2024. URL https://arxiv.org/abs/2411.07681.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114 (13):3521–3526, March 2017. ISSN 1091-6490. doi: 10.1073/pnas.1611835114. URL http://dx.doi.org/10.1073/pnas.1611835114.

Kumar, S., Marklund, H., and Roy, B. V. Maintaining plasticity in continual learning via regenerative regularization, 2024. URL https://arxiv.org/abs/2308.11958.

Lai, S., Zhao, H., Feng, R., Ma, C., Liu, W., Zhao, H., Lin, X., Yi, D., Xie, M., Zhang, Q., Liu, H., Meng, G., and Zhu, F. Reinforcement fine-tuning naturally mitigates forgetting in continual post-training, 2025. URL https://arxiv.org/abs/2507.05386.

Lewandowski, A., Bortkiewicz, M., Kumar, S., Gyorgy, A., Schuurmans, D., Ostaszewski, M., and Machado, M. C. Learning continually by spectral regularization, 2024. URL https://arxiv.org/abs/2406.06811.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., Wu, Y., Neyshabur, B., Gur-Ari, G., and Misra, V. Solving quantitative reasoning problems with language models, 2022. URL https://arxiv.org/abs/2206.14858.

Lin, J., Wang, Z., Qian, K., Wang, T., Srinivasan, A., Zeng, H., Jiao, R., Zhou, X., Gesi, J., Wang, D., Guo, Y., Zhong, K., Zhang, W., Sanghavi, S., Chen, C., Yun, H., and Li, L. Sft doesn't always hurt general capabilities: Revisiting domain-specific fine-tuning in llms, 2025. URL https://arxiv.org/abs/2509.20758.

Lin, S., Hilton, J., and Evans, O. Truthfulqa: Measuring how models mimic human falsehoods, 2022. URL https://arxiv.org/abs/2109.07958.

Meng, F., Wang, Z., and Zhang, M. Pissa: Principal singular values and singular vectors adaptation of large language models, 2025. URL https://arxiv.org/abs/2404.02948.

Mukherjee, S., Yuan, L., Hakkani-Tur, D., and Peng, H. Reinforcement learning finetunes small subnetworks in large language models, 2025. URL https://arxiv.org/abs/2505.11711.

Nayak, N. S., Killamsetty, K., Han, L., Bhandwaldar, A., Chanda, P., Xu, K., Wang, H., Pareja, A., Silkin, O., Eyceoz, M., and Srivastava, A. Sculpting subspaces: Constrained full fine-tuning in llms for continual learning, 2025. URL https://arxiv.org/abs/2504.07097.

Ni, T., Nie, A., Chaudhary, S., Liu, Y., Rangwala, H., and Fakoor, R. Offline learning and forgetting for reasoning with large language models, 2025. URL https://arxiv.org/abs/2504.11364.

Qin, C. and Springenberg, J. T. Supervised fine tuning on curated data is reinforcement learning (and can be improved), 2025. URL https://arxiv.org/abs/2507.12856.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Shenfeld, I., Pari, J., and Agrawal, P. Rl's razor: Why online reinforcement learning forgets less, 2025. URL https://arxiv.org/abs/2509.04259.

Team, P., Du, X., Yao, Y., Ma, K., Wang, B., Zheng, T., Zhu, K., Liu, M., Liang, Y., Jin, X., Wei, Z., Zheng, C., Deng, K., Gavin, S., Jia, S., Jiang, S., Liao, Y., Li, R., Li, Q., Li, S., Li, Y., Li, Y., Ma, D., Ni, Y., Que, H., Wang, Q., Wen, Z., Wu, S., Hsing, T., Xu, M., Yang, Z., Wang, Z. M., Zhou, J., Bai, Y., Bu, X., Cai, C., Chen, L., Chen, Y., Cheng, C., Cheng, T., Ding, K., Huang, S., Huang, Y., Li, Y., Li, Y., Li, Z., Liang, T., Lin, C., Lin, H., Ma, Y., Pang, T., Peng, Z., Peng, Z., Qi, Q., Qiu, S., Qu, X., Quan, S., Tan, Y., Wang, Z., Wang, C., Wang, H., Wang, Y., Wang, Y., Xu, J., Yang, K., Yuan, R., Yue, Y., Zhan, T., Zhang, C., Zhang, J., Zhang, X., Zhang, X., Zhang, Y., Zhao, Y., Zheng, X., Zhong, C., Gao, Y., Li, Z., Liu, D., Liu, Q., Liu, T., Ni, S., Peng, J., Qin, Y., Su, W., Wang, G., Wang, S., Yang, J., Yang, M., Cao, M., Yue, X., Zhang, Z., Zhou, W., Liu, J., Lin, Q., Huang, W., and Zhang, G. Supergpqa: Scaling llm evaluation across 285 graduate disciplines, 2025. URL https://arxiv.org/abs/2502.14739.

Wang, C., Lyu, Y., Sun, Z., and Jing, L. Continual gradient low-rank projection fine-tuning for llms, 2025a. URL https://arxiv.org/abs/2507.02503.

Wang, H., Li, Y., Wang, S., Chen, G., and Chen, Y. Milora: Harnessing minor singular components for parameter-efficient llm finetuning, 2025b. URL https://arxiv.org/abs/2406.09044.

Wang, L., Zhang, X., Su, H., and Zhu, J. A comprehensive survey of continual learning: Theory, method and application, 2024a. URL https://arxiv.org/abs/2302.00487.

Wang, X., Chen, T., Ge, Q., Xia, H., Bao, R., Zheng, R., Zhang, Q., Gui, T., and Huang, X. Orthogonal subspace learning for language model continual learning, 2023. URL https://arxiv.org/abs/2310.14152.

Wang, Y., Ma, X., Zhang, G., Ni, Y., Chandra, A., Guo, S., Ren, W., Arulraj, A., He, X., Jiang, Z., Li, T., Ku, M., Wang, K., Zhuang, A., Fan, R., Yue, X., and Chen, W. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024b. URL https://arxiv.org/abs/2406.01574.

Wu, Y., Zhou, Y., Ziheng, Z., Peng, Y., Ye, X., Hu, X., Zhu, W., Qi, L., Yang, M.-H., and Yang, X. On the generalization of sft: A reinforcement learning perspective with reward rectification, 2025. URL https://arxiv.org/abs/2508.05629.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Dai, W., Fan, T., Liu, G., Liu, L., Liu, X., Lin, H., Lin, Z., Ma, B., Sheng, G., Tong, Y., Zhang, C., Zhang, M., Zhang, W., Zhu, H., Zhu, J., Chen, J., Chen, J., Wang, C., Yu, H., Song, Y., Wei, X., Zhou, H., Liu, J., Ma, W.-Y., Zhang, Y.-Q., Yan, L., Qiao, M., Wu, Y., and Wang, M. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Zhou, J., Lu, T., Mishra, S., Brahma, S., Basu, S., Luan, Y., Zhou, D., and Hou, L. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.

Zhu, H., Zhang, Z., Huang, H., Su, D., Liu, Z., Zhao, J., Fedorov, I., Pirsiavash, H., Lee, J., Pan, D. Z., Wang, Z., Tian, Y., and Tai, K. S. Why rl updates look sparse: An implicit compass drives optimization bias, 2025a. URL https://openreview.net/forum?id=Q4mF4tLGbf. Submitted to ICLR 2026.

Zhu, H., Zhang, Z., Huang, H., Su, D., Liu, Z., Zhao, J., Fedorov, I., Pirsiavash, H., Sha, Z., Lee, J., Pan, D. Z., Wang, Z., Tian, Y., and Tai, K. S. The path not taken: Rlvr provably learns off the principals, 2025b. URL https://arxiv.org/abs/2511.08567.

Zhu, W., Xie, R., Wang, R., Sun, X., Wang, D., and Liu, P. Proximal supervised fine-tuning, 2025c. URL https://arxiv.org/abs/2508.17784.

A. Detailed Results

This section contains the full tables and remaining diagnostic figures for the shortened main paper.

A.1. Full-Parameter Fine-tuning

Table 2. FPFT results on in-domain and out-of-domain benchmarks. ID blocks report Avg@k and Pass@k; the OOD block reports Pass@1. Bold marks the strongest tuned method in each block, and Δ rows report change relative to the base average.

		Llama-3.1-8B-Instruct					Qwen2.5-7B-Instruct					Qwen2.5-3B-Instruct				
Domain	Benchmark	Base	SFT	IW	DFT	RPSFT	Base	SFT	IW	DFT	RPSFT	Base	SFT	IW	DFT	RPSFT
ID Avg@k	AIME24	3.13	**5.00**	3.13	4.17	4.38	12.08	**15.21**	14.17	12.29	14.38	4.79	3.96	**6.67**	2.71	6.46
	AIME25	0.83	5.00	**6.88**	1.88	6.67	7.70	**19.38**	18.54	9.38	18.75	2.29	5.00	**6.04**	4.79	5.00
	AMC23	21.25	32.19	**33.59**	19.53	31.56	52.03	50.47	52.50	48.91	**55.31**	37.03	35.00	**36.25**	34.06	34.84
	MATH-500	43.65	59.45	58.50	46.40	**60.45**	72.85	72.85	72.60	73.35	**74.95**	63.35	60.95	61.05	62.05	**63.35**
	Minerva	22.33	26.88	26.75	20.17	**27.48**	37.09	31.52	**34.05**	33.73	33.82	26.24	22.75	23.71	**26.56**	23.85
	Olympiad	14.63	25.59	25.71	**26.22**	25.52	37.41	37.41	**39.18**	37.41	38.00	26.22	26.25	26.00	25.00	**27.67**
	$\overline{\text{Avg}}$	17.64	25.70	25.76	19.73	**26.01**	36.53	37.81	38.51	35.84	**39.20**	26.65	25.65	26.62	25.86	**26.86**
	Δ vs. Base	—	↑8.06	↑8.12	↑2.09	↑**8.37**	—	↑1.28	↑1.98	↓0.69	↑**2.67**	—	↓1.00	↓0.03	↓0.79	↑**0.21**
ID Pass@k	AIME24	20.00	**20.00**	16.67	13.33	**20.00**	36.67	**50.00**	46.67	23.33	43.33	20.00	30.00	26.66	23.33	**33.33**
	AIME25	13.33	13.33	**26.67**	16.67	20.00	36.67	36.67	36.67	33.33	**43.33**	20.00	23.33	**30.00**	23.33	23.33
	AMC23	72.50	77.50	80.00	62.50	**85.00**	90.00	87.50	85.00	82.50	**90.00**	80.00	**82.50**	**82.50**	77.50	80.00
	MATH-500	65.00	**78.60**	76.20	61.20	77.20	84.60	87.60	**89.00**	84.20	87.40	79.20	**80.60**	79.60	75.80	79.80
	Minerva	44.85	47.06	48.16	37.50	**50.74**	55.17	**58.09**	55.51	48.53	56.62	50.00	**47.42**	45.22	42.28	47.06
	Olympiad	29.78	41.33	41.63	26.22	**42.22**	54.96	54.96	**56.30**	50.07	56.15	41.48	42.37	42.67	38.37	**43.85**
	$\overline{\text{Avg}}$	40.91	46.30	48.22	36.24	**49.19**	59.68	62.47	61.53	53.66	**62.81**	48.45	51.04	51.11	46.77	**51.23**
	Δ vs. Base	—	↑5.39	↑7.31	↓4.67	↑**8.28**	—	↑2.79	↑1.85	↓6.02	↑**3.13**	—	↑2.59	↑2.66	↓1.68	↑**2.78**
OOD Pass@1	GPQA	25.89	31.70	**37.72**	27.68	34.60	33.93	32.80	32.37	30.36	**34.60**	27.01	30.58	27.01	26.56	**31.70**
	IFEval	32.53	31.42	32.53	31.98	**33.46**	61.37	52.13	54.34	**59.70**	54.34	54.16	47.32	46.03	51.20	**52.13**
	MMLU-Pro	47.14	52.86	51.43	52.86	**58.57**	67.14	65.71	58.57	62.86	**70.00**	52.86	**45.71**	**45.71**	34.29	44.28
	SuperGPQA	18.77	19.75	21.48	17.53	**24.20**	27.65	26.42	**26.91**	25.19	26.42	18.27	19.26	19.51	16.30	**21.48**
	Safety	61.10	57.90	**62.20**	60.80	61.70	74.50	70.10	70.10	**71.90**	71.10	65.90	59.60	60.00	**65.80**	62.20
	TruthfulQA	58.04	55.70	49.56	**60.82**	58.63	66.96	59.50	60.53	**65.06**	62.43	59.94	49.42	50.29	**57.31**	51.61
	$\overline{\text{Avg}}$	40.58	41.55	42.49	41.95	**45.19**	55.26	51.11	50.47	52.52	**53.15**	46.36	41.99	41.43	41.91	**43.90**
	Δ vs. Base	—	↑0.97	↑1.91	↑1.37	↑**4.61**	—	↓4.15	↓4.79	↓2.74	↓**2.11**	—	↓4.37	↓4.93	↓4.45	↓**2.46**

Finding 1. Table 2 shows that in the SFT stage, RPSFT gives the best tuned in-domain Avg@k on all three model families, and it also gives the best tuned in-domain Pass@k on Llama-3.1-8B and Qwen2.5-7B. Qwen2.5-3B is closer between RPSFT and IW, but RPSFT still gives the strongest tuned summary rows there. On the out-of-domain block, the pretrained Qwen checkpoints remain the strongest overall reference, but RPSFT is still the best tuned variant on both Qwen sizes and causes the smallest average OOD drop. For Llama-3.1-8B, RPSFT is also the strongest tuned model on OOD, which suggests that the regularizer can improve in-domain performance without paying the same OOD cost as the other tuned baselines.

Finding 2. Figure 2 supports the rotation-preservation view: first-order ID gains do not guarantee OOD retention when the transfer signal is weak or sign-changing. The metric is defined in Appendix G.2.

A.2. RL Fine-Tuning

Finding 3. Table 3 shows that after DAPO (Yu et al., 2025), RPSFT gives the strongest in-domain Avg@k and Pass@k averages on Llama-3.1-8B and Qwen2.5-7B. Qwen2.5-3B is more mixed: RPSFT gives the strongest final Avg@k average, while IW is only slightly better on the final Pass@k average (53.45 vs. 53.26). Overall, the stronger SFT initializer usually carries over to stronger downstream RL performance.

Finding 4. Table 4 shows that for OOD RL under DAPO (Yu et al., 2025), RPSFT gives the best final average on Llama-3.1-8B and Qwen2.5-3B. For Qwen2.5-7B: IW-DAPO has the highest average there, but the gap over RPSFT-DAPO is small (56.01 vs. 55.74). RPSFT-DAPO still gives the best final values on GPQA and SuperGPQA for Qwen2.5-7B and remains close to the best overall average.

Table 3. Downstream RL results on the six in-domain math benchmarks under DAPO (Yu et al., 2025). Entries are init→DAPO; bold marks the highest value among methods.

Metric	Method	AIME24	AIME25	AMC23	MATH-500	Minerva	Olympiad	$\overline{\text{Avg}}$
Llama-3.1-8B-Instruct								
Avg@k	SFT	5.00→**6.46**	5.00→5.42	32.19→**34.38**	59.45→**61.05**	26.88→**29.09**	25.59→25.56	25.70→26.99
	IW	3.13→5.21	6.88→6.46	33.59→32.81	58.50→59.60	26.75→28.77	25.71→25.44	25.76→26.38
	DFT	4.17→3.13	1.88→0.83	19.53→24.53	46.40→48.50	20.17→22.98	26.22→17.41	19.73→22.05
	RPSFT	4.38→6.25	6.67→**7.92**	31.56→33.59	60.45→60.80	27.48→28.54	25.52→**27.03**	26.01→**27.36**
Pass@k	SFT	20.00→20.00	13.33→26.67	77.50→82.50	78.60→77.80	47.06→51.47	41.33→42.07	46.30→50.09
	IW	16.67→23.33	26.67→**30.00**	80.00→77.50	76.20→78.40	48.16→51.84	41.63→41.93	48.22→50.50
	DFT	13.33→10.00	16.67→10.00	62.50→60.00	61.20→63.40	37.50→38.60	26.22→28.44	36.24→35.07
	RPSFT	20.00→**30.00**	20.00→23.33	85.00→**85.00**	77.20→**79.40**	50.74→**52.94**	42.22→**43.70**	49.19→**52.40**
Qwen2.5-7B-Instruct								
Avg@k	SFT	15.21→18.75	19.38→20.83	50.47→58.12	72.85→79.35	31.52→38.97	37.41→43.44	37.81→43.24
	IW	14.17→17.29	18.54→20.00	52.50→60.00	72.60→77.95	34.05→36.76	39.18→42.96	38.51→42.49
	DFT	12.29→11.04	9.38→12.29	48.91→53.13	73.35→74.50	33.73→35.94	37.41→36.81	35.84→37.28
	RPSFT	14.38→**21.25**	18.75→**22.71**	55.31→**62.19**	74.95→**80.50**	33.82→**39.94**	38.00→**46.78**	39.20→**45.56**
Pass@k	SFT	50.00→46.67	36.67→33.33	87.50→90.00	87.60→89.60	58.09→58.09	54.96→59.55	62.47→62.87
	IW	46.67→46.66	36.67→**43.33**	85.00→92.50	89.00→88.00	55.51→55.14	56.30→58.81	61.53→64.07
	DFT	23.33→23.33	33.33→23.33	82.50→82.50	84.20→84.20	48.53→52.94	50.07→48.59	53.66→52.48
	RPSFT	43.33→**53.33**	43.33→**43.33**	90.00→**95.00**	87.40→**90.20**	56.62→**59.19**	56.15→**61.78**	62.81→**67.14**
Qwen2.5-3B-Instruct								
Avg@k	SFT	3.96→4.79	5.00→5.42	35.00→37.50	60.95→65.90	22.75→25.32	26.25→27.44	25.65→27.73
	IW	6.67→6.25	6.04→**6.25**	36.25→**40.94**	61.05→64.65	23.71→24.54	26.00→27.59	26.62→28.37
	DFT	2.71→2.50	4.79→4.38	34.06→39.53	62.05→60.70	26.56→23.67	25.00→25.30	25.86→26.01
	RPSFT	6.46→**6.46**	5.00→6.04	34.84→40.00	63.35→**65.75**	23.85→**26.47**	27.67→**29.04**	26.86→**28.96**
Pass@k	SFT	30.00→23.33	23.33→**33.33**	82.50→75.00	80.60→**81.00**	47.42→47.79	42.37→44.44	51.04→50.81
	IW	26.66→**33.33**	30.00→**33.33**	82.50→**85.00**	79.60→80.40	45.22→45.96	42.67→42.66	51.11→**53.45**
	DFT	23.33→13.33	23.33→26.67	77.50→72.50	75.80→70.80	42.28→39.71	38.37→36.89	46.77→43.32
	RPSFT	33.33→30.00	23.33→30.00	80.00→82.50	79.80→80.00	47.06→**50.37**	43.85→**46.67**	51.23→53.26

Table 4. Downstream RL results on the six out-of-domain benchmarks under DAPO (Yu et al., 2025) (Pass@1 in %). Entries are init→DAPO; final checkpoints are selected by in-domain performance.

Method	GPQA	IFEval	MMLU-Pro	SuperGPQA	Safety	TruthfulQA	$\overline{\text{Avg}}$
Llama-3.1-8B-Instruct							
SFT	31.70→35.27	31.42→31.79	52.86→50.00	19.75→24.94	57.90→59.20	55.70→59.06	41.55→43.38
IW	37.72→**35.71**	32.53→31.24	51.43→58.57	21.48→24.69	62.20→60.50	49.56→54.82	42.49→44.25
DFT	27.68→27.23	31.98→31.05	52.86→**60.00**	17.53→19.75	60.80→58.60	60.82→**64.18**	41.95→43.47
RPSFT	34.60→35.27	33.46→**32.53**	58.57→54.29	24.20→**29.14**	61.70→**60.80**	58.63→62.28	45.19→**45.72**
Qwen2.5-7B-Instruct							
SFT	32.80→33.92	52.13→52.31	65.71→71.43	26.42→32.10	70.10→72.90	59.50→65.20	51.11→54.64
IW	32.37→37.28	54.34→56.38	58.57→**74.29**	26.91→32.59	70.10→73.50	60.53→61.99	50.47→**56.01**
DFT	30.36→31.25	59.70→**60.63**	62.86→54.29	25.19→26.17	71.90→**73.70**	65.06→**68.27**	52.52→52.38
RPSFT	34.60→**39.29**	54.34→54.53	70.00→71.43	26.42→**32.84**	71.10→72.60	62.43→63.74	53.15→55.74
Qwen2.5-3B-Instruct							
SFT	30.58→27.68	47.32→48.43	45.71→47.14	19.26→22.72	59.60→62.50	49.42→52.92	41.99→43.56
IW	27.01→27.90	46.03→46.58	45.71→48.57	19.51→**22.96**	60.00→62.30	50.29→51.32	41.43→43.27
DFT	26.56→25.67	51.20→**51.76**	34.29→44.29	16.30→17.78	65.80→**64.80**	57.31→**56.87**	41.91→43.53
RPSFT	31.70→**28.79**	52.13→48.61	44.28→**55.71**	21.48→20.99	62.20→63.70	51.61→53.95	43.90→**45.29**

A.3. Rotation Across Model Families

To compare geometric drift across architectures, we visualize how different fine-tuning methods rotate the dominant left-singular subspaces of pretrained weights across layers. Here, smaller values mean the tuned model stays closer to the pretrained model. The rotation metrics are in Appendix G.3.

Finding 5. Figures 4a and 4b show that the effect is not isolated to one architecture: across Llama3.1-8B, Qwen2.5-7B, and Qwen2.5-3B, RPSFT keeps the tuned model closer to the base model through most layers. The separation is strongest through the middle layers and through the protected rank spectrum, suggesting that RPSFT protects leading singular directions that carry more of the pretrained structure.

A.4. Rank Robustness

Finding 6. Table 5 summarizes the protected-rank sweep from Appendix E. Intermediate ranks preserve the strongest joint trade-off: $k = 512$ gives the best Llama ID average, while the default $k = 768$ gives the best Llama OOD average and the strongest Qwen ID average among the swept settings. Stronger anchoring can improve OOD retention but sacrifices more adaptation, supporting a moderate protected rank.

Table 5. Rank robustness summary from the full sweeps in Appendix E.

	Llama-8B		Qwen-7B	
Setting	ID	OOD	ID	OOD
SFT	25.69	41.55	37.81	51.11
LoRA	18.90	41.11	35.20	44.64
RPSFT-256	26.02	44.17	38.63	50.94
RPSFT-512	**26.50**	43.66	38.88	50.80
RPSFT-768	26.01	**45.19**	**39.20**	53.15
L2 Init	18.24	41.10	37.13	**55.26**

A.5. Representation Drift

At the representation level, we examine whether different SFT methods preserve the pretrained geometry. The exact hidden-state drift metric is defined in Appendix G.6.

Finding 7. Figure 5 shows that across the benchmark panels, RPSFT usually keeps the tuned centroid among the closest to the base model, while SFT, IW, and especially DFT often shift farther away. This indicates that RPSFT protects representation-space structure during fine-tuning. The same qualitative pattern also appears on the larger models. Appendix Figures 9 and 10 show that for Llama-8B and Qwen2.5-7B, RPSFT again usually produces smaller centroid shifts away from the base model across benchmark panels.

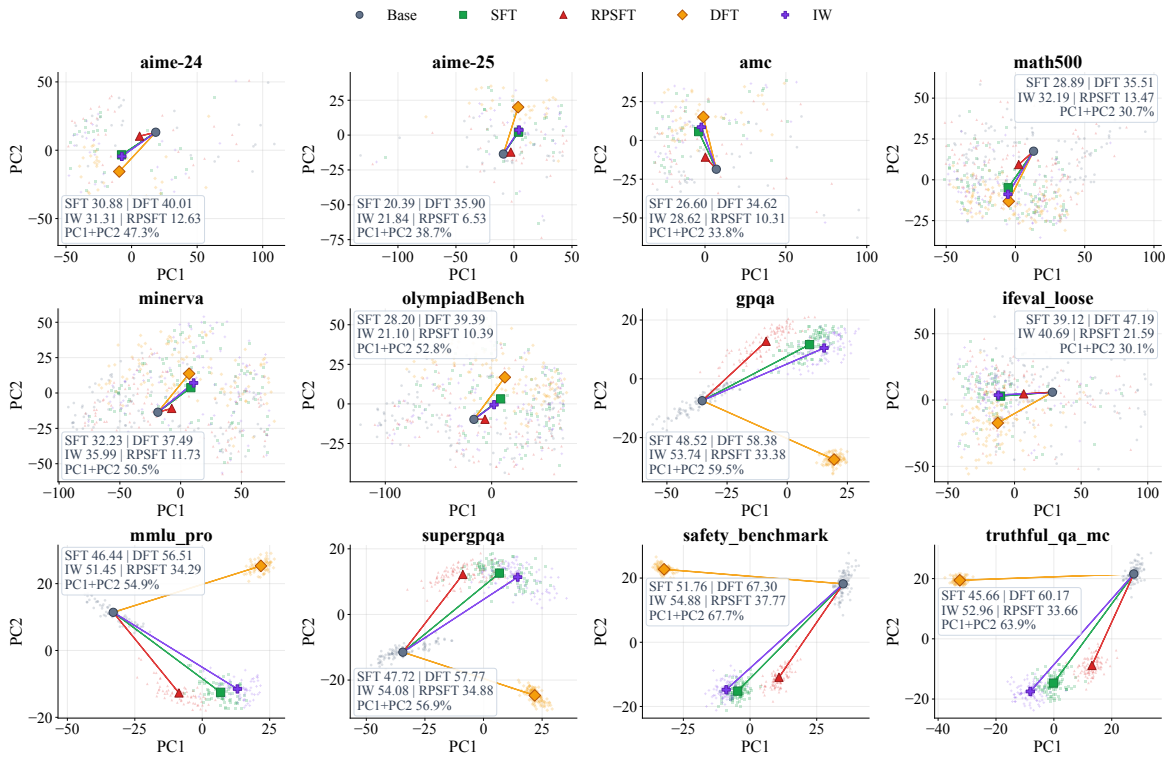


Figure 5. Hidden-state drift on Qwen2.5-3B-Instruct. RPSFT usually stays among the closest tuned centroids to the base model, while SFT, IW, and especially DFT often shift farther.

B. Algorithms

Algorithm 2 Rotation-Preserving SFT (FPFT)

Require: Pretrained weights $\{\mathbf{W}_\ell^0\}$, dataset $\mathcal{D}$, regularized layer set $\mathcal{M}'$, rank k, coefficient λ, update period s

Precompute: for each $\ell \in \mathcal{M}'$, compute $(\mathbf{U}_{0,\ell}^{(k)}, \mathbf{V}_{0,\ell}^{(k)})$ from SVD of $\mathbf{W}_\ell^0$; set $S_\ell^{\text{ref}} \leftarrow (\mathbf{U}_{0,\ell}^{(k)})^\top \mathbf{W}_\ell^0 \mathbf{V}_{0,\ell}^{(k)}$
Initialize $\theta \leftarrow \theta_0$
for training step $t = 1, 2, \ldots$ **do**
  Sample minibatch $(x, y) \sim \mathcal{D}$
  Compute task loss $\mathcal{L}_{\text{task}}(\theta)$
  $\mathcal{L}_{\text{reg}} \leftarrow 0$
  for each $\ell \in \mathcal{M}'$ **do**
    $S \leftarrow (\mathbf{U}_{0,\ell}^{(k)})^\top \mathbf{W}_\ell \mathbf{V}_{0,\ell}^{(k)}$
    $\mathcal{L}_{\text{reg}} \leftarrow \mathcal{L}_{\text{reg}} + \|S - S_\ell^{\text{ref}}\|_F^2$
  end for
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{reg}}$
  Update $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$
end for

Minimal PyTorch implementation. The core implementation only caches the pretrained singular bases once and adds a projected-block penalty to the usual SFT loss:

```
# Precompute once for each selected pretrained weight W0
with torch.no_grad():
    U, _, Vh = torch.linalg.svd(W0.float(), full_matrices=False)
    Uk = U[:, :k].to(device=W0.device, dtype=W0.dtype)
    Vk = Vh[:k].T.to(device=W0.device, dtype=W0.dtype)
```

```
    S0 = Uk.T @ W0 @ Vk

# During SFT, sum this penalty over selected layers when step % s == 0
rpsft = 0.0
for W, Uk, Vk, S0 in rpsft_layers:
    S = Uk.T @ W @ Vk
    rpsft = rpsft + (S - S0).pow(2).sum()

loss = task_loss + lam * rpsft
loss.backward()
```

B.1. SVD Notation

For a weight matrix $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, singular value decomposition writes $\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where d_{out} and d_{in} are the output and input dimensions, $\mathbf{U} \in \mathbb{R}^{d_{\text{out}} \times r}$ and $\mathbf{V} \in \mathbb{R}^{d_{\text{in}} \times r}$ have orthonormal columns, $\mathbf{\Sigma} \in \mathbb{R}^{r \times r}$ is the diagonal matrix of singular values in descending order, and $r = \operatorname{rank}(\mathbf{W})$. The leading singular directions capture the dominant transformations implemented by the model and define the pretrained subspaces that RPSFT protects.

B.2. DAPO Objective

DAPO samples a group of G responses from the rollout policy $\pi_{\theta_{\text{old}}}(\cdot \mid x)$ and computes the normalized group-relative advantage

$$\hat{A}_i = \frac{r_i - \frac{1}{G}\sum_{j=1}^{G} r_j}{\operatorname{std}(\{r_j\}_{j=1}^{G}) + \varepsilon}, \qquad r_i = r(x, y_i). \tag{5}$$

With token-level PPO ratio

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t} \mid x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, y_{i,<t})}, \tag{6}$$

the clipped DAPO surrogate is

$$\mathcal{J}_{\text{DAPO}}(\theta) = \mathbb{E}_{x,\{y_i\}\sim\pi_{\theta_{\text{old}}}(\cdot|x)} \left[\frac{1}{\sum_{i=1}^{G}|y_i|} \sum_{i,t} \min\Big(\rho_{i,t}\hat{A}_i, \; \operatorname{clip}\big(\rho_{i,t}, 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}}\big)\hat{A}_i \Big) \right]. \tag{7}$$

Here x is the prompt, i indexes sampled responses, t ranges over response tokens, $|y_i|$ is the response length, and $\epsilon_{\text{low}}, \epsilon_{\text{high}}$ are the clipping thresholds.

B.3. LoRA Objective

For completeness, RPSFT can be applied to LoRA by adding the same projected-block drift penalty to the effective weight. This optional variant is not used in the reported experiments; the LoRA rows in Appendix E are vanilla LoRA baselines with adapter rank $r = 32$.

$$\mathcal{L}(A, B) = \mathcal{L}_{\text{task}}(A, B) + \lambda \sum_{\ell \in \mathcal{M}'} \left\| S_\ell(\mathbf{W}_\ell^0 + \Delta\mathbf{W}_\ell) - S_\ell^{\text{ref}} \right\|_F^2, \tag{8}$$

where $\Delta\mathbf{W}_\ell = B_\ell A_\ell$. This keeps the regularizer aligned with FPFT: it penalizes drift inside the pretrained dominant subspace even when only low-rank parameters are trained.

C. Related Work

C.1. Traditional Continual-Learning Regularization

Continual-learning surveys commonly separate replay, architecture expansion, and regularization-based approaches (Wang et al., 2024a). Our work is closest to the regularization family: rather than storing old data or changing the model architecture, these methods modify the optimization objective or update rule to control parameter drift. A classical example is Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017), which uses a Fisher-weighted quadratic penalty to discourage

movement of important parameters. A simpler related baseline is anchoring or regenerative regularization, which penalizes uniform Euclidean movement from a reference model, typically through a term proportional to $\|\theta - \theta_0\|_2^2$ (Kumar et al., 2024). Other recent continual-learning regularizers control the weight geometry more directly: spectral regularization keeps the maximum singular value of each layer near one (Lewandowski et al., 2024), weight clipping bounds weight magnitudes after each update (Elsayed et al., 2024), and Parseval regularization encourages near-orthogonal weight matrices (Chung et al., 2024). These methods motivate regularizing the geometry of the update, but they do not target the pretrained singular directions that are most tied to OOD forgetting in our analysis.

C.2. LLM Post-Training and Reasoning Generalization

Recent LLM studies show that reasoning post-training can either generalize or over-specialize depending on the learning signal and update geometry. Learning-dynamics analyses find that pre-memorization training accuracy can predict reasoning generalization (Kang et al., 2024), while learnability-based RL curricula prioritize prompts with high reward variance, where the group-relative advantage has nonzero signal (Foster et al., 2025). Other work shows that math-only post-training often transfers imperfectly: SFT can induce representation and output drift, whereas RL tends to preserve broader capabilities (Huan et al., 2025). This is consistent with findings that RL fine-tuning mitigates forgetting through reward-variance scaling (Lai et al., 2025), that online RL is biased toward KL-minimal solutions (Shenfeld et al., 2025), and that RL updates may be sparse, spectrum-preserving, or off-principal in parameter space (Mukherjee et al., 2025; Zhu et al., 2025a;b). Offline reasoning studies also show that naive SFT can harm search capability and that smaller learning rates can reduce degradation (Ni et al., 2025; Lin et al., 2025). These results motivate our focus on the SFT stage: RPSFT explicitly controls SFT-induced movement in dominant pretrained singular subspaces, aiming to obtain rapid task adaptation without causing the representation drift that later RL analyses suggest is avoidable.

Several LLM fine-tuning methods are methodologically close to this goal. Proximal SFT constrains policy drift with PPO-style updates (Zhu et al., 2025c); Dynamic SFT and importance-weighted SFT rescale token or example losses (Wu et al., 2025; Qin & Springenberg, 2025); and subspace-constrained methods project gradients away from protected directions, including principal-subspace preservation and gradient low-rank projection (Franke et al., 2024; Nayak et al., 2025; Wang et al., 2025a). Orthogonality-based PEFT methods such as O-LoRA similarly reduce interference by constraining the adapter subspace (Wang et al., 2023). Spectral PEFT methods such as PiSSA and MiLoRA use singular directions for efficient adaptation (Meng et al., 2025; Wang et al., 2025b), but their goal is parameter efficiency rather than explicitly mitigating OOD forgetting. In contrast, RPSFT keeps the full fine-tuning parameterization and adds a soft projected-block penalty in the pretrained singular-vector basis.

C.3. Comparison with Regularization Methods

Let θ_0 denote the pretrained parameters, and let $\mathbf{W}_\ell^0 = \mathbf{U}_\ell^0 \mathbf{\Sigma}_\ell^0 (\mathbf{V}_\ell^0)^\top$ be the SVD of layer ℓ. Table 6 compares methods whose update rules or objectives are explicit enough to make a direct theoretical comparison. We omit papers that are primarily diagnostic or empirical motivation when the paper does not define a comparable regularizer.

Table 6. Comparison between RPSFT and related regularization or constraint methods. P denotes the number of trainable parameters, and k denotes the protected rank used by RPSFT.

Method	Objective or update	Code	Compute	Memory	Key difference
EWC (Kirkpatrick et al., 2017)	$\mathcal{L}_{\text{task}} + \frac{\lambda}{2}\sum_i F_i(\theta_i - \theta_{0,i})^2$	Med.	Fisher pass $+O(P)$	θ_0, F $(2P)$	parameter-wise importance
L2 anchor (Kumar et al., 2024)	$\mathcal{L}_{\text{task}} + \lambda\|\theta - \theta_0\|_2^2$.	Low	$O(P)$	θ_0 (P)	global, all directions
Weight clipping (Elsayed et al., 2024)	$\theta_i \leftarrow \text{clip}(\theta_i, -\epsilon, \epsilon)$	Low	$O(P)$	0	absolute magnitude only
Spectral / Parseval regularization (Lewandowski et al., 2024; Chung et al., 2024)	$\sigma_{\max}(\mathbf{W}_\ell) \approx c$ or $\mathbf{W}_\ell^\top \mathbf{W}_\ell \approx \mathbf{I}$	Med.	SVD/power iter.	low	shapes values, not basis
Hard subspace projection (Franke et al., 2024; Nayak et al., 2025)	$g \leftarrow (\mathbf{I} - \mathbf{P})g$ or projected update	High	projection step	protected bases	hard removal of gradients
RPSFT (ours)	$\mathcal{L}_{\text{task}} + \lambda \sum_\ell \|S_\ell - S_\ell^0\|_F^2$, $S_\ell = (\mathbf{U}_0^k)^\top \mathbf{W}_\ell \mathbf{V}_0^k$	Low	SVD; low-rank proj.	bases $+S^0$	soft top-k SVD block

D. Theory: Properties of RPSFT Regularization

D.1. Setup and Notation

Consider a weight matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$ with pretrained initialization $\mathbf{W}_0$. Let the (thin) SVD of $\mathbf{W}_0$ be

$$\mathbf{W}_0 = \mathbf{U}_0\mathbf{\Sigma}_0\mathbf{V}_0^\top, \tag{9}$$

where $\mathbf{U}_0 \in \mathbb{R}^{m\times r}$, $\mathbf{V}_0 \in \mathbb{R}^{n\times r}$ have orthonormal columns, $\mathbf{\Sigma}_0 \in \mathbb{R}^{r\times r}$ is diagonal, and $r = \mathrm{rank}(\mathbf{W}_0)$. Let $\mathbf{U}_k \in \mathbb{R}^{m\times k}$ and $\mathbf{V}_k \in \mathbb{R}^{n\times k}$ denote the top-k left and right singular vectors of $\mathbf{W}_0$. Define the linear operator

$$\mathcal{P}_k(\mathbf{W}) \triangleq \mathbf{U}_k^\top \mathbf{W}\mathbf{V}_k \in \mathbb{R}^{k\times k}. \tag{10}$$

RPSFT adds the penalty

$$\mathcal{R}_k(\mathbf{W}) \triangleq \|\mathcal{P}_k(\mathbf{W}) - \mathcal{P}_k(\mathbf{W}_0)\|_F^2 = \left\|\mathbf{U}_k^\top(\mathbf{W}-\mathbf{W}_0)\mathbf{V}_k\right\|_F^2. \tag{11}$$

Given a task loss $f(\mathbf{W})$ (e.g., the SFT objective), the RPSFT objective is

$$F_\lambda(\mathbf{W}) \triangleq f(\mathbf{W}) + \lambda\,\mathcal{R}_k(\mathbf{W}), \qquad \lambda \geq 0. \tag{12}$$

D.2. Boundary Cases and Limiting Behavior

We first record basic limiting cases that clarify how λ and k interpolate between standard SFT and constrained training.

Case $\lambda \to 0$. As $\lambda \to 0$, $F_\lambda(\mathbf{W}) \to f(\mathbf{W})$ and RPSFT reduces to the original SFT objective.

Case $\lambda \to \infty$. For $\lambda \to \infty$, minimizers of F_λ converge to solutions of the constrained problem:

$$\min_{\mathbf{W}} f(\mathbf{W}) \quad \text{s.t.} \quad \mathcal{P}_k(\mathbf{W}) = \mathcal{P}_k(\mathbf{W}_0) \iff \mathbf{U}_k^\top(\mathbf{W}-\mathbf{W}_0)\mathbf{V}_k = 0. \tag{13}$$

which fixes the component of $\mathbf{W}$ acting between the pretrained top-k left and right singular subspaces while leaving all other components unconstrained.

Case $k = 0$. $\mathcal{R}_0(\mathbf{W}) \equiv 0$, hence RPSFT is exactly standard SFT for any λ.

Case $k = \min(m, n)$ (full rank). Let $k = \min(m, n)$ and let $\mathbf{U} \in \mathbb{R}^{m\times m}$, $\mathbf{V} \in \mathbb{R}^{n\times n}$ be any orthogonal completions of the singular vector bases of $\mathbf{W}_0$. Then, by orthogonal invariance of the Frobenius norm,

$$\mathcal{R}_k(\mathbf{W}) = \|\mathbf{U}^\top(\mathbf{W}-\mathbf{W}_0)\mathbf{V}\|_F^2 = \|\mathbf{W}-\mathbf{W}_0\|_F^2, \tag{14}$$

so RPSFT becomes standard ℓ_2 anchoring around the pretrained weights.

D.3. Block Decomposition Interpretation

Let $\mathbf{U} = [\mathbf{U}_k \;\; \mathbf{U}_\perp] \in \mathbb{R}^{m\times m}$ and $\mathbf{V} = [\mathbf{V}_k \;\; \mathbf{V}_\perp] \in \mathbb{R}^{n\times n}$ be orthogonal matrices extending $\mathbf{U}_k, \mathbf{V}_k$. Define the rotated coordinates

$$\widetilde{\mathbf{W}} \triangleq \mathbf{U}^\top\mathbf{W}\mathbf{V} = \begin{pmatrix} A & B \\ C & D \end{pmatrix}, \tag{15}$$

where $A \in \mathbb{R}^{k\times k}$. Similarly $\widetilde{\mathbf{W}}_0 = \mathbf{U}^\top\mathbf{W}_0\mathbf{V}$ has top-left block A_0. Then

$$\mathcal{R}_k(\mathbf{W}) = \|A - A_0\|_F^2. \tag{16}$$

Thus, RPSFT penalizes drift of the top-left $k \times k$ block in the pretrained singular-vector basis, while allowing updates through the other blocks B, C, D.

D.4. Optimality Condition and $O(1/\lambda)$ Drift Control

We give a simple bound showing how λ controls the protected-block deviation.

Proposition D.1 (Stationary Condition)**.** *Assume f is differentiable. Any stationary point $\mathbf{W}_\lambda$ of F_λ satisfies*

$$\nabla f(\mathbf{W}_\lambda) + 2\lambda\, \mathbf{U}_k\big(\mathbf{U}_k^\top(\mathbf{W}_\lambda - \mathbf{W}_0)\mathbf{V}_k\big)\mathbf{V}_k^\top = 0. \tag{17}$$

Proof. Using $\mathcal{R}_k(\mathbf{W}) = \|\mathbf{U}_k^\top(\mathbf{W} - \mathbf{W}_0)\mathbf{V}_k\|_F^2$ and standard matrix calculus, $\nabla_{\mathbf{W}}\mathcal{R}_k(\mathbf{W}) = 2\mathbf{U}_k(\mathbf{U}_k^\top(\mathbf{W} - \mathbf{W}_0)\mathbf{V}_k)\mathbf{V}_k^\top$. Setting $\nabla F_\lambda(\mathbf{W}) = 0$ yields Eq. (17). □

Proposition D.2 ($1/\lambda$ Control of Protected Drift)**.** *Let $\mathbf{W}_\lambda$ be any stationary point of F_λ. Then*

$$\big\|\mathbf{U}_k^\top(\mathbf{W}_\lambda - \mathbf{W}_0)\mathbf{V}_k\big\|_F \le \frac{\|\nabla f(\mathbf{W}_\lambda)\|_F}{2\lambda}. \tag{18}$$

Proof. Left-multiply Eq. (17) by $\mathbf{U}_k^\top$ and right-multiply by $\mathbf{V}_k$:

$$\mathbf{U}_k^\top \nabla f(\mathbf{W}_\lambda)\mathbf{V}_k + 2\lambda\, \mathbf{U}_k^\top(\mathbf{W}_\lambda - \mathbf{W}_0)\mathbf{V}_k = 0. \tag{19}$$

Taking Frobenius norms and using $\|\mathbf{U}_k^\top X \mathbf{V}_k\|_F \le \|X\|_F$ gives Eq. (18). □

Implication. Eq. (18) shows that 1) at any stationary point, the protected-block deviation is bounded by the task-gradient magnitude scaled by $O(1/\lambda)$; 2) as λ increases, the deviation of the protected block $\mathbf{U}_k^\top(\mathbf{W}_\lambda - \mathbf{W}_0)\mathbf{V}_k$ shrinks at rate $O(1/\lambda)$, formalizing a smooth interpolation between unconstrained SFT ($\lambda = 0$) and the constrained regime obtained as $\lambda \to \infty$.

D.5. Rank Selection: Boundary and Threshold Rule

We now formalize why an excessively large protected rank can hurt rapid adaptation. We work in the forgetting-dominated regime discussed above, where the pretrained model is locally close to an OOD stationary point, so the first-order OOD term is negligible.

Let $\Delta \coloneqq \mathbf{W} - \mathbf{W}_0$ and

$$\widetilde{\Delta} \coloneqq \mathbf{U}^\top \Delta \mathbf{V}, \tag{20}$$

and index the entries of $\widetilde{\Delta}$ by $s = (i, j)$, with coordinate value δ_s. For each rank k, define the protected coordinate set

$$S_k \coloneqq \{(i, j) : 1 \le i \le k,\ 1 \le j \le k\}. \tag{21}$$

Thus $s \in S_k$ means that δ_s lies in the protected top-left $k \times k$ block of the pretrained singular-vector basis.

We use the local quadratic model

$$f_{\mathrm{id}}(\mathbf{W}_0 + \Delta) - f_{\mathrm{id}}(\mathbf{W}_0) \approx -\sum_s g_s\delta_s + \frac{1}{2}\sum_s h_s\delta_s^2, \qquad h_s > 0, \tag{22}$$

where g_s is the in-domain learning drive along coordinate s, and h_s is the corresponding local curvature. Under RPSFT, the local objective becomes

$$-\sum_s g_s\delta_s + \frac{1}{2}\sum_s h_s\delta_s^2 + \lambda \sum_{s\in S_k} \delta_s^2. \tag{23}$$

For OOD forgetting, we use the quadratic proxy

$$f_{\mathrm{ood}}(\mathbf{W}_0 + \Delta) - f_{\mathrm{ood}}(\mathbf{W}_0) \approx \frac{1}{2}\sum_s c_s\delta_s^2, \qquad c_s \ge 0, \tag{24}$$

where c_s measures how sensitive OOD performance is to movement along coordinate s: larger c_s means larger OOD degradation.

Proposition D.3 (Optimal local step)**.** *For each coordinate s, the minimizer of the local regularized objective is*

$$\delta_s^\star(k) = \begin{cases} \dfrac{g_s}{h_s + 2\lambda}, & s \in S_k, \\ \dfrac{g_s}{h_s}, & s \notin S_k. \end{cases} \tag{25}$$

The resulting OOD increase is

$$F_{\mathrm{ood}}(k) = \frac{1}{2} \sum_{s \in S_k} \frac{c_s g_s^2}{(h_s + 2\lambda)^2} + \frac{1}{2} \sum_{s \notin S_k} \frac{c_s g_s^2}{h_s^2}. \tag{26}$$

The resulting ID gain, measured on the unregularized local ID proxy after taking the regularized step, is

$$G_{\mathrm{id}}(k) = \sum_{s \in S_k} \left(\frac{g_s^2}{h_s + 2\lambda} - \frac{1}{2} h_s \frac{g_s^2}{(h_s + 2\lambda)^2} \right) + \frac{1}{2} \sum_{s \notin S_k} \frac{g_s^2}{h_s}. \tag{27}$$

Proof. The objective is separable across coordinates, so the optimizer is obtained coordinatewise. Substituting the resulting $\delta_s^\star(k)$ into the ID and OOD quadratic proxies gives the stated formulas. □

Proposition D.4 (Existence of an upper rank boundary)**.** *Assume there exists q such that*

$$c_s = 0 \qquad \text{for all } s \notin S_q. \tag{28}$$

That is, all OOD-sensitive coordinates are already contained in the protected top-q block. Define the scalarized utility

$$\Phi(k) \coloneqq G_{\mathrm{id}}(k) - \beta F_{\mathrm{ood}}(k), \qquad \beta > 0, \tag{29}$$

where β controls how strongly OOD forgetting is penalized relative to ID gain. Then every maximizer $k^\star$ of Φ satisfies

$$k^\star \le q. \tag{30}$$

Proof. For $k \ge q$, enlarging the protected set no longer changes $F_{\mathrm{ood}}(k)$, because every coordinate with $c_s > 0$ is already protected at rank q. Hence

$$F_{\mathrm{ood}}(k) = F_{\mathrm{ood}}(q), \qquad k \ge q. \tag{31}$$

On the other hand, protecting any additional coordinate weakly decreases its contribution to $G_{\mathrm{id}}(k)$, with strict decrease whenever $\lambda > 0$ and $g_s \neq 0$. Therefore

$$G_{\mathrm{id}}(k) \le G_{\mathrm{id}}(q), \qquad k \ge q, \tag{32}$$

and thus

$$\Phi(k) \le \Phi(q), \qquad k \ge q. \tag{33}$$

So no maximizer can lie above q. □

Corollary D.5 (Threshold rule)**.** *Consider a coordinate s that is currently unprotected. Define the ID cost of protecting this coordinate by*

$$\Delta_{\mathrm{ID},s} \coloneqq \frac{1}{2} \frac{g_s^2}{h_s} - \left(\frac{g_s^2}{h_s + 2\lambda} - \frac{1}{2} h_s \frac{g_s^2}{(h_s + 2\lambda)^2} \right) = \frac{2\lambda^2 g_s^2}{h_s (h_s + 2\lambda)^2}, \tag{34}$$

and define the OOD gain of protecting this coordinate by

$$\Delta_{\mathrm{OOD},s} \coloneqq \frac{1}{2} c_s \frac{g_s^2}{h_s^2} - \frac{1}{2} c_s \frac{g_s^2}{(h_s + 2\lambda)^2} = \frac{2 c_s \lambda (h_s + \lambda) g_s^2}{h_s^2 (h_s + 2\lambda)^2}. \tag{35}$$

Then protecting coordinate s improves Φ iff

$$\beta \, \Delta_{\mathrm{OOD},s} > \Delta_{\mathrm{ID},s}, \tag{36}$$

which is equivalent to

$$c_s > \frac{\lambda h_s}{\beta (h_s + \lambda)}. \tag{37}$$

Proof. This follows by comparing the protected and unprotected one-coordinate contributions in Proposition D.3 and simplifying. □

Interpretation. The upper-boundary proposition shows that once the protected rank exceeds the support of OOD-sensitive coordinates, increasing k can no longer improve robustness, but can still worsen the trade-off between mitigating forgetting and rapid adaptation by over-suppressing directions that are useful for in-domain adaptation. The threshold rule gives the per-coordinate version of the same idea: a direction should be protected only if its OOD sensitivity c_s is large enough to outweigh the in-domain gain lost by shrinking that direction. In this view, c_s can be interpreted as the OOD sensitivity of coordinate s in the pretrained singular basis, where larger c_s means that updating this direction causes greater OOD degradation. If OOD sensitivity decays with singular index, this naturally induces a finite rank boundary, so the optimal rank k should be large enough to cover the high-c_s directions, but not so large that it also protects many low-c_s directions. This provides a simple explanation for the rank sweep in Appendix E: moving from $k = 0$ to a moderate rank improves robustness because it protects the most OOD-sensitive directions, whereas overly large ranks behave more like global anchoring and degrade the trade-off between mitigating forgetting and rapid adaptation by over-constraining directions that are not strongly OOD-sensitive.

Practical guidance. Before SFT, we choose k using the base model and the same Fisher-projected gradient-energy diagnostic shown in Figure 1. Concretely, we compute per-sample gradients on a small batch from the SFT data, project them into the pretrained SVD basis of an early attention matrix such as layer-1 `q_proj`, and sweep candidate ranks r. We then inspect the curve $x(r) = 100r^2/R^2$ versus $y(r) = 100\,\mathrm{tr}(\mathbf{P}_{\mathrm{svd},r}\mathbf{F})/\mathrm{tr}(\mathbf{F})$. A practical default is the smallest r whose strict top-$r \times r$ block captures about 20% of the gradient energy in the early attention layer: this protects a meaningful fraction of loss-sensitive directions while keeping the protected block small enough for adaptation. In our experiments, $r = 768$ corresponds to a roughly 5% strict block and already captures about 20% of the gradient energy, so we use $k = 768$ as the default protected rank.

D.6. Gradient-Flow View: Exponentially Damped Task-Induced Drift

To gain insight into the optimization dynamics induced by RPSFT, we consider the continuous-time gradient-flow limit of the regularized objective

$$F_\lambda(\mathbf{W}) = f(\mathbf{W}) + \lambda\|\mathbf{U}_k^\top(\mathbf{W} - \mathbf{W}_0)\mathbf{V}_k\|_F^2.$$

Assume that f is differentiable with locally Lipschitz gradient, so that the gradient flow is well-defined. The resulting dynamics are

$$\frac{d\mathbf{W}(t)}{dt} = -\nabla f(\mathbf{W}(t)) - 2\lambda\,\mathbf{U}_k\big(\mathbf{U}_k^\top(\mathbf{W}(t) - \mathbf{W}_0)\mathbf{V}_k\big)\mathbf{V}_k^\top. \tag{38}$$

Protected coordinates. Define the protected coordinates

$$A(t) \triangleq \mathbf{U}_k^\top(\mathbf{W}(t) - \mathbf{W}_0)\mathbf{V}_k,$$

which measure the deviation of the weights from the pretrained model within the top-k singular subspace of $\mathbf{W}_0$. Multiplying Eq. (38) on the left by $\mathbf{U}_k^\top$ and on the right by $\mathbf{V}_k$ yields

$$\frac{dA(t)}{dt} = -\,\mathbf{U}_k^\top\nabla f(\mathbf{W}(t))\mathbf{V}_k - 2\lambda A(t). \tag{39}$$

Closed-form solution. Eq. (39) is a linear ODE with a time-dependent forcing term. Define

$$G(t) \triangleq \mathbf{U}_k^\top\nabla f(\mathbf{W}(t))\mathbf{V}_k,$$

so that $\dot{A}(t) + 2\lambda A(t) = -G(t)$. Solving this equation yields

$$A(t) = e^{-2\lambda t}A(0) - \int_0^t e^{-2\lambda(t-s)}G(s)\,ds, \tag{40}$$

or equivalently,

$$A(t) = e^{-2\lambda t}A(0) - \int_0^t e^{-2\lambda(t-s)}\big(\mathbf{U}_k^\top\nabla f(\mathbf{W}(s))\mathbf{V}_k\big)\,ds.$$

The first term represents an exponentially decaying initial deviation from the pretrained weights, while the second term is an exponentially weighted accumulation of task-induced gradients in the protected subspace.

Interpretation. Eq. (40) shows that RPSFT acts as an exponential damping mechanism on task-induced drift within the pretrained top-k singular subspace. Even when fine-tuning is initialized from the pretrained model (so that $A(0) = 0$), task gradients generally introduce nonzero drift, but their influence is continuously decayed with rate 2λ. Consequently, RPSFT behaves as a temporal low-pass filter on gradients in the protected subspace, stabilizing dominant pretrained representations while still allowing task-relevant adaptation.

E. Additional Robustness Results

We study the sensitivity of RPSFT to the protected rank k on Llama-3.1-8B-Instruct as the empirical counterpart to the rank-boundary analysis in Appendix D.5. Here, SFT corresponds to $k = 0$, and our chosen configuration is $k = 768$. The LoRA row is a vanilla LoRA baseline with adapter rank $r = 32$; no RPSFT penalty is applied to it. The full-rank setting $k = 4096$ reduces RPSFT to weight-space anchoring around the pretrained model, so we label it as L2 Init in Table 7. This matches the regenerative regularization view of Kumar et al. (2024). The main insight is that Llama benefits from an intermediate protected rank: $k = 256$–768 preserves adaptation, but larger k progressively weakens both in-domain adaptation and OOD generalization. This matches the positive Llama OOD first-order signal in Figure 2: when the update direction can still help OOD performance, over-protecting too many directions removes useful transfer rather than only preventing forgetting. The same pattern agrees with the rank-boundary analysis in Appendix D.5: once the protected block already covers the most OOD-sensitive directions, expanding it mainly constrains useful task updates.

Table 7. Rank robustness on Llama-3.1-8B-Instruct across ID Avg@k, ID Pass@k, and OOD Avg@1 (%). The LoRA row is vanilla LoRA with adapter rank $r = 32$; RPSFT rows vary protected rank k. Base cells are left unmarked, and bold marks the best non-base value in each domain block and column.

Domain	Setting	AIME24	AIME25	AMC23	MATH-500	Minerva	Olympiad	$\overline{\text{Avg}}$
ID Avg@k	Base	3.13	0.83	21.25	43.65	22.33	14.63	17.64
	SFT	5.00	5.00	32.19	59.45	26.88	25.59	25.69
	Vanilla LoRA (r=32)	3.13	2.50	23.75	47.80	21.60	14.63	18.90
	RPSFT-256	**5.42**	6.46	29.84	59.05	**28.63**	26.70	26.02
	RPSFT-512	4.58	**7.50**	**34.06**	58.20	27.34	**27.34**	**26.50**
	RPSFT-768	4.38	6.67	31.56	**60.45**	27.48	25.52	26.01
	RPSFT-2048	4.17	5.21	28.44	56.70	25.64	23.63	23.97
	L2 Init (k=4096)	4.38	1.04	21.88	44.60	21.51	16.00	18.24
ID Pass@k	Base	20.00	13.33	72.50	65.00	44.85	29.78	40.91
	SFT	20.00	13.33	77.50	**78.60**	47.06	41.33	46.30
	Vanilla LoRA (r=32)	23.33	20.00	80.00	67.40	48.53	29.78	44.84
	RPSFT-256	20.00	**26.66**	75.00	75.40	**50.74**	42.67	48.41
	RPSFT-512	20.00	23.33	**87.50**	76.60	49.26	**49.26**	**50.99**
	RPSFT-768	20.00	20.00	85.00	77.20	**50.74**	42.22	49.19
	RPSFT-2048	16.67	20.00	72.50	76.60	47.79	39.85	45.57
	L2 Init (k=4096)	**26.66**	13.33	72.50	67.60	47.79	31.85	43.29
		GPQA	IFEval	MMLU-Pro	SuperGPQA	Safety	TruthfulQA	$\overline{\text{Avg}}$
OOD Avg@1	Base	25.89	32.53	47.14	18.77	61.10	58.04	40.58
	SFT	31.70	31.42	52.86	19.75	57.90	55.70	41.55
	Vanilla LoRA (r=32)	28.57	31.79	48.57	17.78	63.80	56.14	41.11
	RPSFT-256	33.93	30.87	**58.57**	21.98	63.40	56.29	44.17
	RPSFT-512	32.81	30.87	52.86	20.74	**68.70**	55.99	43.66
	RPSFT-768	**34.60**	**33.46**	**58.57**	**24.20**	61.70	58.63	**45.19**
	RPSFT-2048	33.26	31.05	54.29	23.95	58.80	56.58	42.99
	L2 Init (k=4096)	26.34	32.53	47.14	16.79	62.70	**61.11**	41.10

Qwen2.5-7B robustness sweep. Table 8 reports the same protected-rank sweep on Qwen2.5-7B-Instruct. Here, SFT corresponds to $k = 0$, the LoRA row is vanilla LoRA with adapter rank $r = 32$, the chosen RPSFT configuration is again $k = 768$, and $k = 3584$ is equivalent to L2 Init. Qwen shows the other side of the same trade-off: increasing k reduces in-domain adaptation but steadily mitigates OOD forgetting, with the full-rank L2 Init row closest to the base

model on OOD average. This matches Figure 2, where the Qwen OOD first-order signal is weaker and changes sign more often, so unconstrained task adaptation is less reliably aligned with OOD improvement. The trend is also consistent with Appendix D.5: protecting more OOD-sensitive directions improves retention, but beyond a moderate rank the extra protection starts to suppress task learning. Vanilla LoRA remains weaker, especially on Qwen OOD retention, indicating that parameter-efficient adaptation alone does not provide the same forgetting control as the RPSFT projected-block penalty.

Table 8. Rank robustness on Qwen2.5-7B-Instruct across ID Avg@k, ID Pass@k, and OOD Avg@1 (%). The LoRA row is vanilla LoRA with adapter rank $r = 32$; RPSFT rows vary protected rank k. Base cells are left unmarked, and bold marks the best non-base value in each domain block and column.

Domain	Setting	AIME24	AIME25	AMC23	MATH-500	Minerva	Olympiad	$\overline{\text{Avg}}$
ID Avg@k	Base	12.08	7.70	52.03	72.85	37.09	37.41	36.53
	SFT	15.21	**19.38**	50.47	72.85	31.52	37.41	37.81
	Vanilla LoRA (r=32)	11.25	13.54	46.41	71.80	34.24	33.96	35.20
	RPSFT-256	**16.25**	17.50	54.38	73.25	32.81	37.56	38.63
	RPSFT-512	15.00	**19.38**	52.97	73.30	33.23	**39.37**	38.88
	RPSFT-768	14.38	18.75	**55.31**	**74.95**	33.82	38.00	**39.20**
	RPSFT-2048	14.17	17.08	51.72	74.70	33.50	39.15	38.39
	L2 Init (k=3584)	11.88	9.38	51.88	74.70	**37.55**	37.37	37.13
ID Pass@k	Base	36.67	36.67	90.00	84.60	55.17	54.96	59.68
	SFT	**50.00**	36.67	87.50	**87.60**	**58.09**	54.96	62.47
	Vanilla LoRA (r=32)	40.00	30.00	77.50	84.60	55.88	52.15	56.69
	RPSFT-256	46.67	36.67	**90.00**	86.60	54.04	54.37	61.39
	RPSFT-512	43.33	40.00	85.00	86.60	58.46	**57.04**	61.74
	RPSFT-768	43.33	**43.33**	**90.00**	87.40	56.62	56.15	**62.81**
	RPSFT-2048	43.33	36.67	85.00	86.60	57.35	56.29	60.87
	L2 Init (k=3584)	30.00	40.00	**90.00**	84.80	**58.09**	52.15	59.17
		GPQA	IFEval	MMLU-Pro	SuperGPQA	Safety	TruthfulQA	$\overline{\text{Avg}}$
OOD Avg@1	Base	33.93	61.37	67.14	27.65	74.50	66.96	55.26
	SFT	32.80	52.13	65.71	26.42	70.10	59.50	51.11
	Vanilla LoRA (r=32)	32.59	45.84	58.57	23.21	67.60	64.04	48.64
	RPSFT-256	34.82	56.38	67.14	29.38	70.00	59.80	52.92
	RPSFT-512	**36.38**	56.01	65.71	**29.63**	68.80	61.26	52.97
	RPSFT-768	34.60	54.34	70.00	26.42	**71.10**	62.43	53.15
	RPSFT-2048	33.71	57.67	**74.29**	27.41	70.90	63.45	54.57
	L2 Init (k=3584)	35.49	**63.03**	68.57	27.41	70.10	**66.67**	**55.21**

F. Experiment Details

Precomputation. We compute the baseline SVD blocks immediately after loading the pretrained model. For each $\ell \in \mathcal{M}'$, we compute top-k singular vectors of $\mathbf{W}_\ell^0$ (e.g., via truncated SVD) and store $(\mathbf{U}_{0,\ell}^{(k)}, \mathbf{V}_{0,\ell}^{(k)}, S_\ell^{\text{ref}})$.

Training datasets For supervised fine-tuning, we use OpenR1-Math (Hugging Face, 2025). For downstream reinforcement learning, we use the DAPO-Math-17k set together with the DAPO objective (Yu et al., 2025). Every DAPO (Yu et al., 2025) run is initialized from the corresponding supervised checkpoint, so the RL comparisons isolate the effect of the SFT initializer.

Training setup For supervised fine-tuning, we train Qwen2.5-7B and Qwen2.5-3B for 12 epochs, and we train Llama-3.1-8B for 20 epochs because its base checkpoint is weaker. We evaluate the same saved checkpoints across methods. All shared training hyperparameters are kept the same across SFT baselines, and only the method-specific parameters are changed, using the default settings for each method.

Table 9 lists the shared SFT hyperparameters used for all SFT-stage methods. Unless stated otherwise, RPSFT uses protected rank $k = 768$ and regularization coefficient $\lambda = 1$.

Table 9. Shared supervised fine-tuning hyperparameters.

Parameter	Value
Learning rate	1×10^{-6}
Epochs	12 for Qwen; 20 for Llama
Precision	`bf16`
LR scheduler	cosine
Warmup ratio	0.03
Weight decay	0.0

For downstream reinforcement learning, we use DAPO and initialize every run from the corresponding supervised checkpoint. We train each run for 100 total training steps and then select the best checkpoint according to in-domain performance for the reported evaluation. Table 10 lists the shared RL-stage hyperparameters.

Table 10. Shared reinforcement-learning fine-tuning hyperparameters.

Parameter	Value
Actor learning rate	1×10^{-6}
Train batch size	256 prompts
PPO mini-batch size	32 prompts
Generation batch size	128 prompts
Responses per prompt	8
Max prompt length	2048 tokens
Max response length	10240 tokens
Temperature	1.0
Validation top-p	0.7
PPO clip low / high	0.2 / 0.28
Total training steps	100
Overlong buffer length	2048 tokens
Overlong penalty factor	1.0

Compute and artifacts. All SFT and RL training runs use 8 H200 GPUs. We provide the code, dataset preparation and evaluation artifacts, training configurations, and documentation as supplemental material for reproduction, and will make the artifact public on GitHub after the anonymous review period.

Benchmarks In-domain tasks: AIME24, AIME25, AMC23, MATH-500 (Hendrycks et al., 2021), Minerva Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024).

Out-of-domain tasks: GPQA (Rein et al., 2023), IFEval-loose (Zhou et al., 2023), MMLU-Pro (Wang et al., 2024b), SuperGPQA (Team et al., 2025), Safety Benchmark, and TruthfulQA (Lin et al., 2022).

G. Metrics for Analysis Figures

G.1. Strict SVD Subspace Energy

Figure 1 measures the ratio of how much of the empirical Fisher curvature is captured by the strict top singular block. Let $\mathbf{G}_t \in \mathbb{R}^{m \times n}$ denote the gradient matrix of the t-th sample, where $t = 1, \ldots, N$ indexes the N sampled gradients used in the diagnostic, and let

$$g_t = \mathrm{vec}(\mathbf{G}_t)$$

be its vectorized form. We define the empirical Fisher matrix as

$$\mathbf{F} = \sum_{t=1}^{N} g_t g_t^\top.$$

Let $u_i \in \mathbb{R}^m$ and $v_j \in \mathbb{R}^n$ be the i-th left and j-th right singular vectors of the pretrained weight matrix, let r be the protected singular rank, and let $\mathbf{P}_{\mathrm{svd},r}$ denote the orthogonal projector onto the strict top-$r \times r$ singular-vector product

subspace

$$\operatorname{span}\{\operatorname{vec}(u_i v_j^\top) : 1 \le i, j \le r\}.$$

Then the fraction of gradient energy captured by the strict top-$r \times r$ singular block is

$$\frac{\operatorname{tr}(\mathbf{P}_{\mathrm{svd},r}\mathbf{F})}{\operatorname{tr}(\mathbf{F})} = \frac{\sum_{t=1}^{N} \|\mathbf{P}_{\mathrm{svd},r} g_t\|_2^2}{\sum_{t=1}^{N} \|g_t\|_2^2}.$$

Equivalently, in matrix form, this can be written as

$$\frac{\operatorname{tr}(\mathbf{P}_{\mathrm{svd},r}\mathbf{F})}{\operatorname{tr}(\mathbf{F})} = \frac{\sum_{t=1}^{N} \left\|\mathbf{U}_r^\top \mathbf{G}_t \mathbf{V}_r\right\|_F^2}{\sum_{t=1}^{N} \|\mathbf{G}_t\|_F^2},$$

where $\mathbf{U}_r = [u_1, \ldots, u_r]$ and $\mathbf{V}_r = [v_1, \ldots, v_r]$.

Here, the denominator $\operatorname{tr}(\mathbf{F}) = \sum_{t=1}^{N} \|g_t\|_2^2$ is the total gradient energy across all samples, meaning the total sum of squared gradient norms, while the numerator $\operatorname{tr}(\mathbf{P}_{\mathrm{svd},r}\mathbf{F})$ is the portion of that energy lying in the strict top-$r \times r$ singular block. In the figure, the x-axis reports $x(r) = 100\, r^2/R^2$, where $R = \operatorname{rank}(\mathbf{W}_0)$ is the full singular rank, and the y-axis reports $100 \operatorname{tr}(\mathbf{P}_{\mathrm{svd},r}\mathbf{F})/\operatorname{tr}(\mathbf{F})$. We apply this diagnostic to the layer-1 attention `q_proj` weight for Llama-8B, Qwen-7B, and Qwen-3B.

G.2. Layerwise First-Order Signal

To diagnose whether a checkpoint update helps or hurts a dataset at first order, we compare the update direction against the dataset gradient. For a checkpoint weight $\mathbf{W}_{\mathrm{ckpt}}$ and base-model weight $\mathbf{W}_{\mathrm{base}}$, the update is

$$\Delta\mathbf{W} = \mathbf{W}_{\mathrm{ckpt}} - \mathbf{W}_{\mathrm{base}}. \tag{41}$$

For a batch B from an ID or OOD dataset, the average gradient on matrix $\mathbf{W}$ is

$$g_{\mathbf{W}} = \frac{1}{|B|} \sum_{(x,y)\in B} \nabla_{\mathbf{W}} \ell(f_{\mathbf{W}}(x), y). \tag{42}$$

Let $\mathcal{P}_\ell$ be the set of matrices assigned to layer ℓ in the plot. The layerwise first-order signal is the average inner product

$$m_\ell = \frac{1}{|\mathcal{P}_\ell|} \sum_{\mathbf{W}\in\mathcal{P}_\ell} \langle g_{\mathbf{W}}, \Delta\mathbf{W} \rangle. \tag{43}$$

Negative values mean the checkpoint update points along the local descent direction for that dataset, so it reduces the loss at first order. Positive values mean the update conflicts with the dataset gradient and tends to increase the loss at first order.

G.3. Rotation Metrics

For each layer ℓ and weight type

$$t \in \{\texttt{q\_proj}, \texttt{k\_proj}, \texttt{v\_proj}, \texttt{o\_proj}, \texttt{up\_proj}, \texttt{down\_proj}, \texttt{gate\_proj}\},$$

let the base-model weight and tuned-model weight be

$$\mathbf{W}_{\mathrm{base}}^{(\ell,t)}, \qquad \mathbf{W}_{m}^{(\ell,t)}.$$

Here m indexes the tuned method or checkpoint being compared. Let $d_{\mathrm{out}}^{(\ell,t)}$ and $d_{\mathrm{in}}^{(\ell,t)}$ denote the output and input dimensions of this matrix. We compute truncated SVD with

$$K_{\ell,t} = \min\big(512, d_{\mathrm{out}}^{(\ell,t)}, d_{\mathrm{in}}^{(\ell,t)}\big),$$

so that

$$\mathbf{W}_{\text{base}}^{(\ell,t)} = \mathbf{U}_{\text{base}}^{(\ell,t)}\mathbf{\Sigma}_{\text{base}}^{(\ell,t)}{\mathbf{V}_{\text{base}}^{(\ell,t)}}^{\top}, \qquad \mathbf{W}_{m}^{(\ell,t)} = \mathbf{U}_{m}^{(\ell,t)}\mathbf{\Sigma}_{m}^{(\ell,t)}{\mathbf{V}_{m}^{(\ell,t)}}^{\top},$$

where

$$\mathbf{U}_{\text{base}}^{(\ell,t)},\ \mathbf{U}_{m}^{(\ell,t)} \in \mathbb{R}^{d_{\text{out}}^{(\ell,t)} \times K_{\ell,t}}, \qquad \mathbf{V}_{\text{base}}^{(\ell,t)},\ \mathbf{V}_{m}^{(\ell,t)} \in \mathbb{R}^{d_{\text{in}}^{(\ell,t)} \times K_{\ell,t}}.$$

To measure U-space rotation, we compare the left-singular subspaces through

$$\mathbf{M}_{U}^{(\ell,t,m)} = \left(\mathbf{U}_{\text{base}}^{(\ell,t)}\right)^{\top} \mathbf{U}_{m}^{(\ell,t)}.$$

If the singular values of $\mathbf{M}_{U}^{(\ell,t,m)}$ are $\sigma_1^{(\ell,t,m)}, \ldots, \sigma_{K_{\ell,t}}^{(\ell,t,m)}$, the principal angles are

$$\theta_i^{(\ell,t,m)} = \arccos\left(\sigma_i^{(\ell,t,m)}\right).$$

The mean U-space rotation for layer/type pair (ℓ, t) is

$$r_{U}^{(\ell,t,m)} = \frac{180}{\pi} \cdot \frac{1}{K_{\ell,t}} \sum_{i=1}^{K_{\ell,t}} \theta_i^{(\ell,t,m)}.$$

The layerwise plot in Figure 4a uses `types-all`, so the value at layer ℓ averages these per-type rotations over the available matrix types:

$$y_m(\ell) = \frac{\sum_{t \in \mathcal{T}_\ell} c^{(\ell,t,m)}\, r_{U}^{(\ell,t,m)}}{\sum_{t \in \mathcal{T}_\ell} c^{(\ell,t,m)}},$$

where $\mathcal{T}_\ell$ is the set of available types in layer ℓ. In the current plotting script, $c^{(\ell,t,m)} = 1$ for each available entry, so this reduces to the simple mean across types:

$$y_m(\ell) = \frac{1}{|\mathcal{T}_\ell|} \sum_{t \in \mathcal{T}_\ell} \left(\frac{180}{\pi} \cdot \frac{1}{K_{\ell,t}} \sum_{i=1}^{K_{\ell,t}} \arccos\left(\sigma_i^{(\ell,t,m)}\right) \right).$$

Thus, each curve reports how much the top left-singular subspaces rotate away from the base model across layers after averaging over all selected matrix types.

For the rankwise case-study plot in Figure 4b, we fix one matrix case for each model and vary the prefix rank $r \le K_{\ell,t}$. At each r, we replace the full truncated bases by their first r columns, compute the singular values of

$$\mathbf{M}_{U}^{(r,m)} = \left(\mathbf{U}_{\text{base}}^{(r)}\right)^{\top} \mathbf{U}_{m}^{(r)},$$

where $\mathbf{U}_{\text{base}}^{(r)}$ and $\mathbf{U}_{m}^{(r)}$ are the first r columns of the truncated left-singular bases for the base and tuned matrices, and report the mean principal angle

$$y_m(r) = \frac{180}{\pi} \cdot \frac{1}{r} \sum_{i=1}^{r} \arccos\left(\sigma_i^{(r,m)}\right).$$

This keeps the same rotation definition and only changes the swept dimension: the layerwise figure varies ℓ after averaging over types, while the rankwise figure varies the prefix rank within one fixed case-study matrix.

G.4. Entropy Metric

For prompt j, method m generates tokens

$$y^{(m,j)} = \left(y_1^{(m,j)}, \ldots, y_{T_{m,j}}^{(m,j)}\right),$$

with greedy decoding and $T_{m,j} \leq 128$. At generation step t, let

$$p_t^{(m,j)}(v) = \mathrm{softmax}(z_t^{(m,j)})_v.$$

The token entropy is

$$h_t^{(m,j)} = -\sum_{v \in \mathcal{V}} p_t^{(m,j)}(v) \log p_t^{(m,j)}(v),$$

and the sample-level average entropy is

$$e_{m,j} = \frac{1}{T_{m,j}} \sum_{t=1}^{T_{m,j}} h_t^{(m,j)}.$$

Each figure plots a Gaussian KDE over $\{e_{m,j}\}_{j=1}^{N_m}$:

$$\hat{f}_m(x) = \frac{1}{N_m h_m} \sum_{j=1}^{N_m} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2} \left(\frac{x - e_{m,j}}{h_m} \right)^2 \right),$$

using the default bandwidth

$$h_m = 1.06\, s_m\, N_m^{-1/5},$$

where s_m is the sample standard deviation. We use the same definition for both datasets: AIME25 for the in-domain plot and GPQA for the out-of-domain plot.

G.5. Auxiliary Entropy Plots Across Model Families

We include these entropy plots as auxiliary analysis rather than as a primary claim. Compared with the hidden-state drift results in the main paper, the entropy differences are more model-dependent and less uniformly separated across baselines.

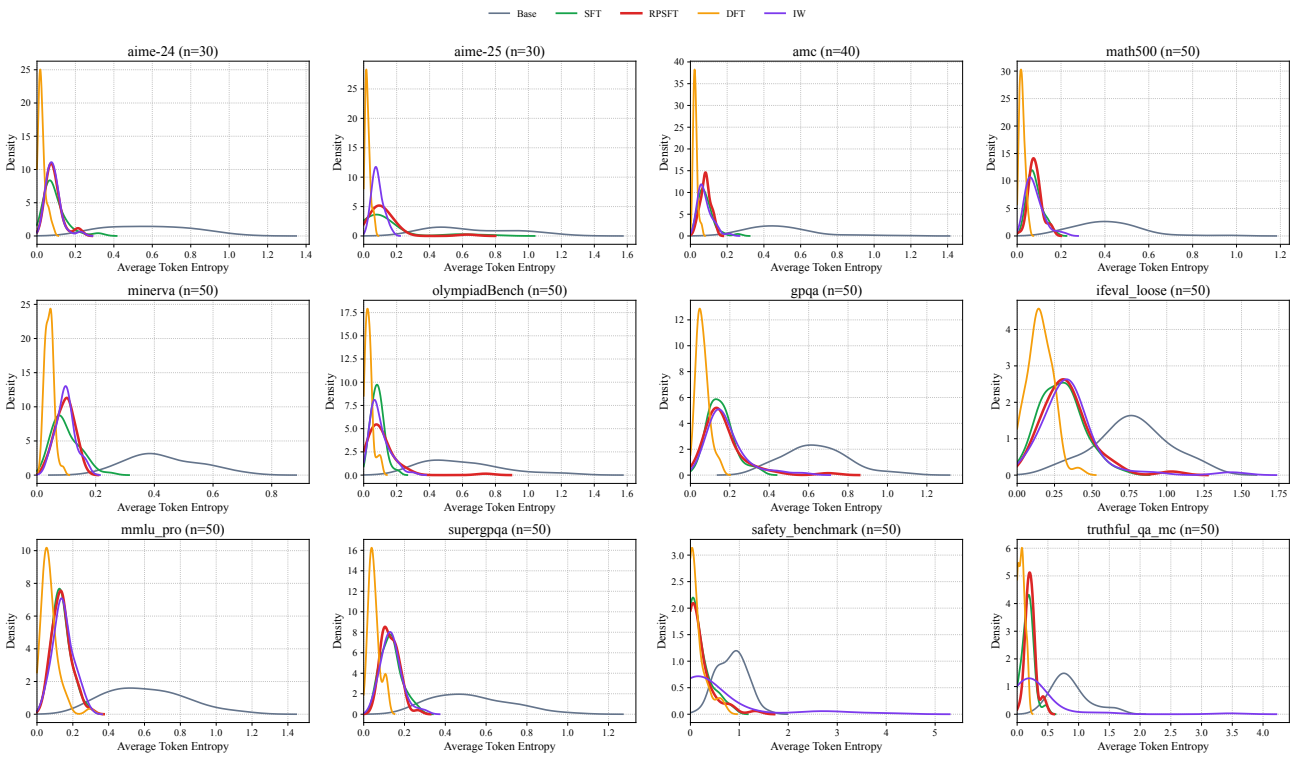


Figure 6. Average-token-entropy distributions across the twelve in-domain and out-of-domain benchmarks for Llama-3.1-8B-Instruct. The differences are modest and benchmark-dependent: DFT more often shifts toward lower entropy, while RPSFT generally stays in a similar range to the stronger tuned baselines.

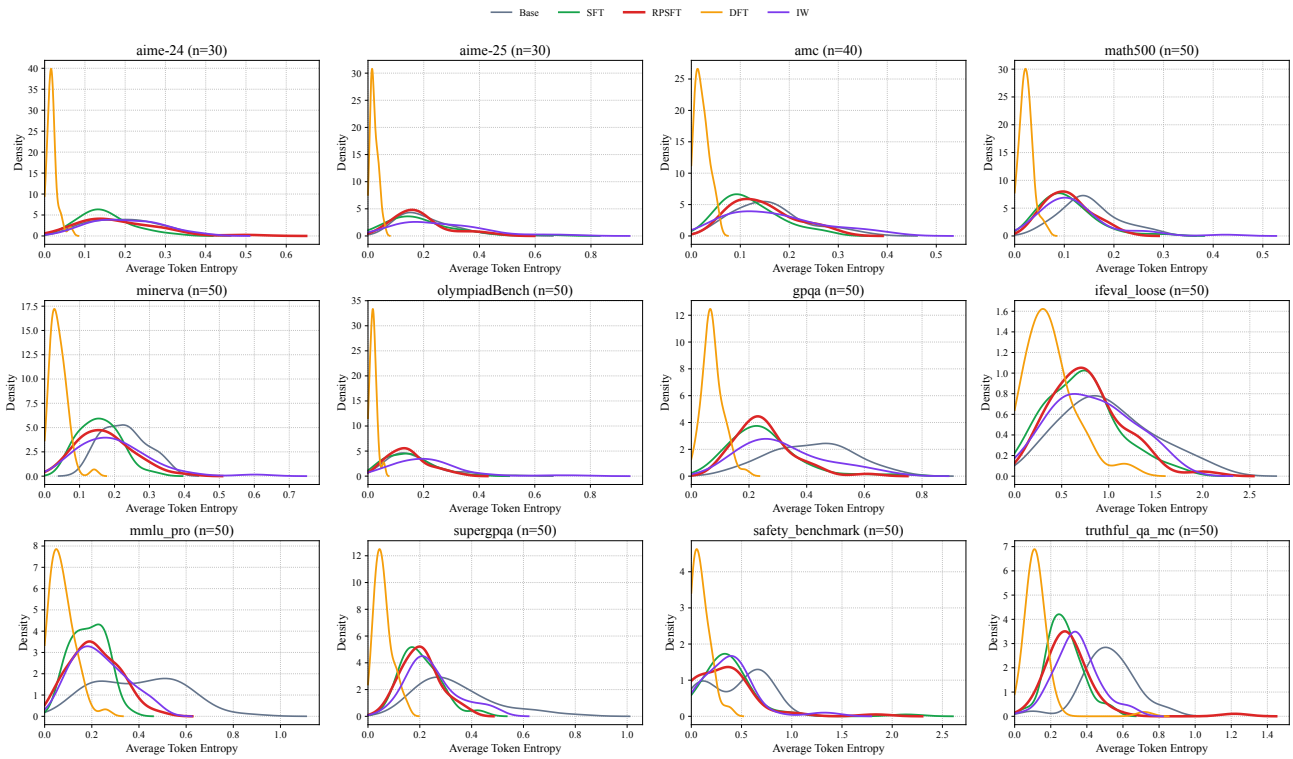


Figure 7. Average-token-entropy distributions across the same twelve benchmarks for Qwen2.5-7B-Instruct. RPSFT often remains above DFT and within the broader range of the tuned baselines, but the gaps are not uniformly large across all panels.

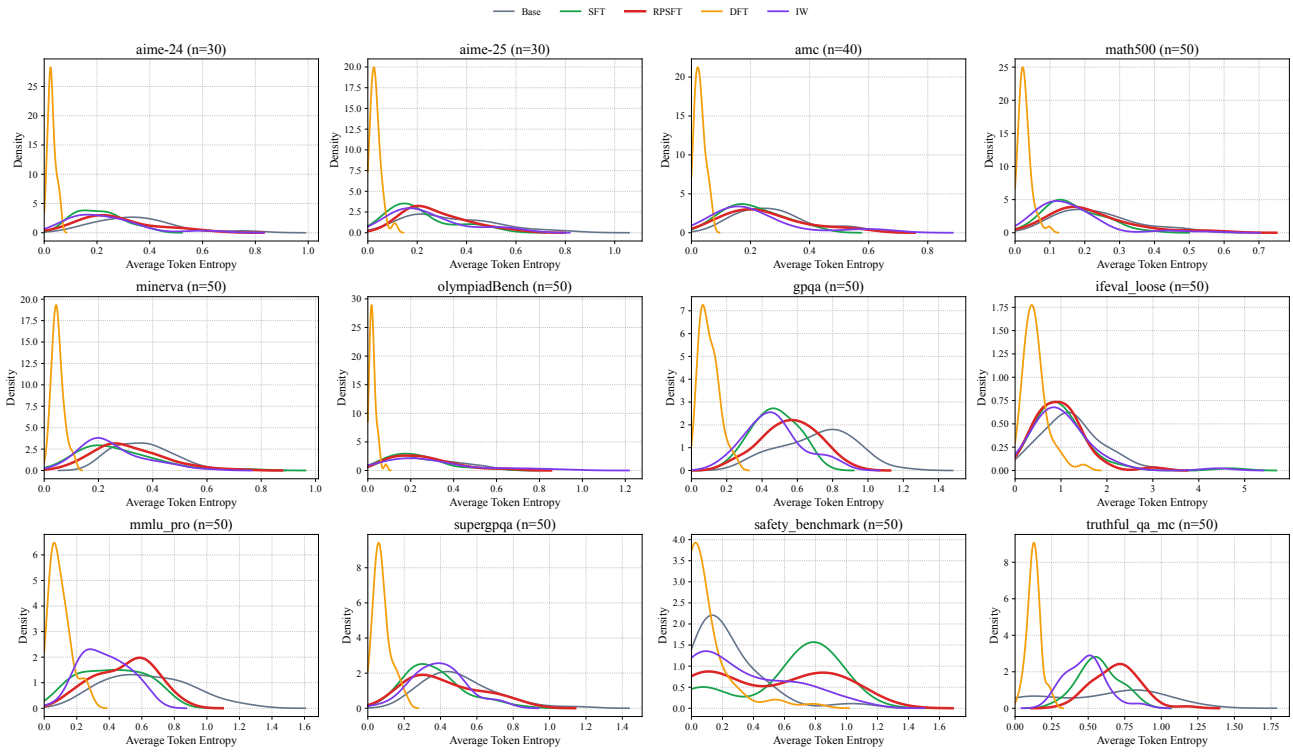


Figure 8. Average-token-entropy distributions across the same twelve benchmarks for Qwen2.5-3B-Instruct. RPSFT often shifts toward a broader entropy profile than standard SFT or DFT, although the base model remains the highest-entropy reference and the gaps are still panel-dependent.

G.6. Base-Model Drift in Hidden Representations

For dataset d with N_d prompts, let $h_i^{(m,d)} \in \mathbb{R}^p$ be the pooled hidden representation for prompt i under model $m \in \{\text{base}, \text{sft}, \text{RPSFT}, \text{dft}, \text{iw}\}$. The hidden-space centroid of model m on dataset d is

$$\mu^{(m,d)} = \frac{1}{N_d} \sum_{i=1}^{N_d} h_i^{(m,d)}.$$

We measure drift from the base model by the centroid distance

$$D_{\text{hidden}}^{(m,d)} = \left\| \mu^{(m,d)} - \mu^{(\text{base},d)} \right\|_2.$$

For visualization, we stack all prompt representations from the five models,

$$X^{(d)} = \begin{bmatrix} H^{(\text{base},d)} \\ H^{(\text{sft},d)} \\ H^{(\text{RPSFT},d)} \\ H^{(\text{dft},d)} \\ H^{(\text{iw},d)} \end{bmatrix} \in \mathbb{R}^{M_d \times p}, \qquad M_d = 5N_d,$$

center the matrix as $\tilde{X}^{(d)} = X^{(d)} - \mathbf{1}\,\bar{x}^{(d)\top}$, and project onto the first two PCA directions:

$$Z^{(d)} = \tilde{X}^{(d)} W_{1:2} \in \mathbb{R}^{M_d \times 2}.$$

If $z_i^{(m,d)} \in \mathbb{R}^2$ is the PCA coordinate of prompt i, the PCA-plane centroid is

$$c^{(m,d)} = \frac{1}{N_d} \sum_{i=1}^{N_d} z_i^{(m,d)},$$

and the plotted centroid shift is

$$D_{\text{PCA}}^{(m,d)} = \left\| c^{(m,d)} - c^{(\text{base},d)} \right\|_2.$$

In the plots, each point is one prompt and each arrow starts at the base centroid $c^{(\text{base},d)}$ and ends at the centroid of a tuned model. We use $D_{\text{hidden}}^{(m,d)}$ for the quantitative drift measurement and $D_{\text{PCA}}^{(m,d)}$ only for the 2D visualization.

H. Limitations and Responsible Use

Our experiments focus on math-domain SFT followed by math RLFT, so the results do not by themselves establish the same trade-off for non-math supervised data, preference data, multimodal models, or much larger model scales. The theory is local and uses quadratic approximations around the pretrained checkpoint; it should be read as an explanatory model for the observed rank trade-off rather than as a global guarantee for nonconvex training. RPSFT also stores pretrained singular bases and projection buffers, so it uses more peak memory than vanilla SFT and DFT, although less than IW and L2 Init in our measured setting.

RPSFT is an optimization method, not a safety mechanism. Better retention and stronger RL initializations can improve benign downstream reliability, but deployment still requires model-use policies, safety evaluation, and release controls appropriate for the base model and application.

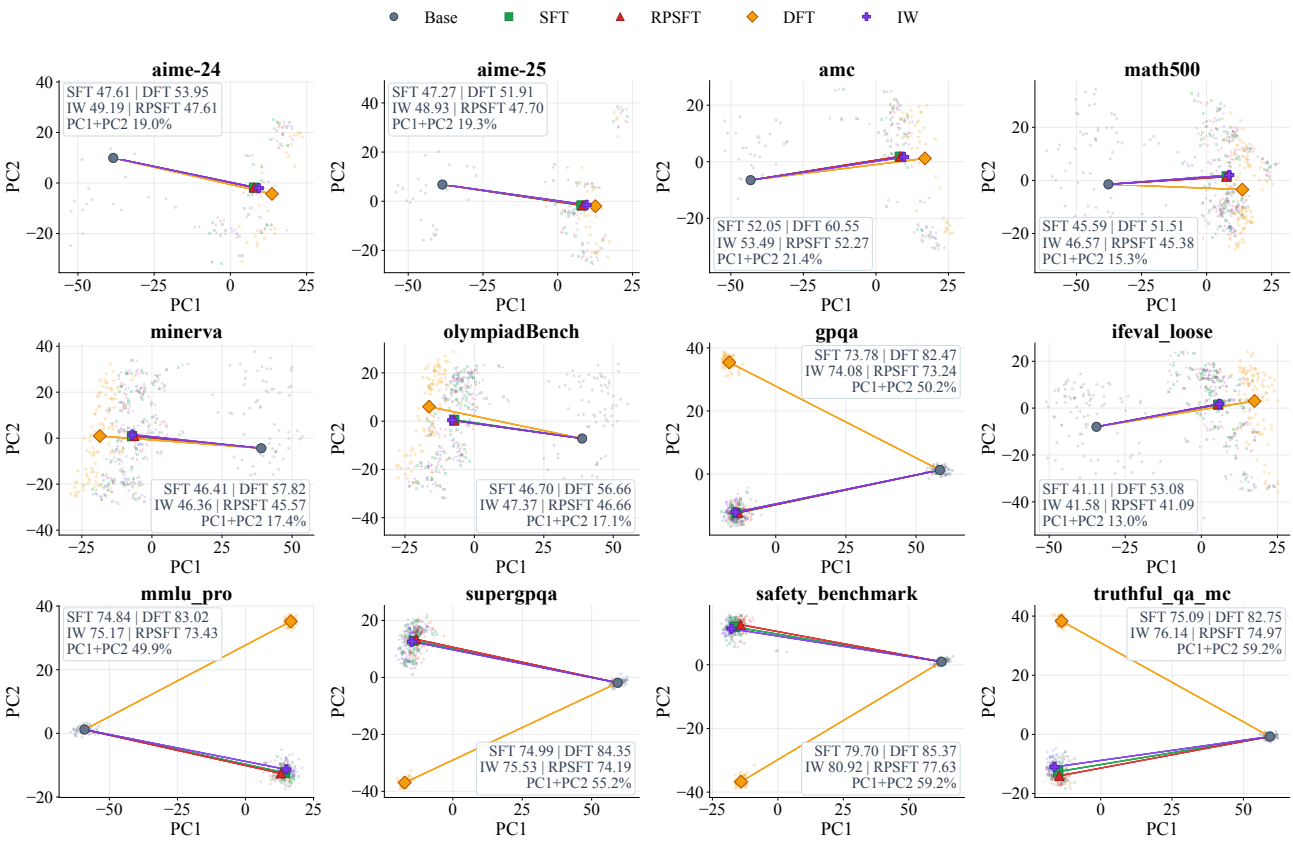


Figure 9. Hidden-state drift on Llama-3.1-8B-Instruct. Across the benchmark panels, RPSFT remains close to the base-model centroid and is competitive with the strongest tuned alternatives, which is consistent with reduced representation drift.

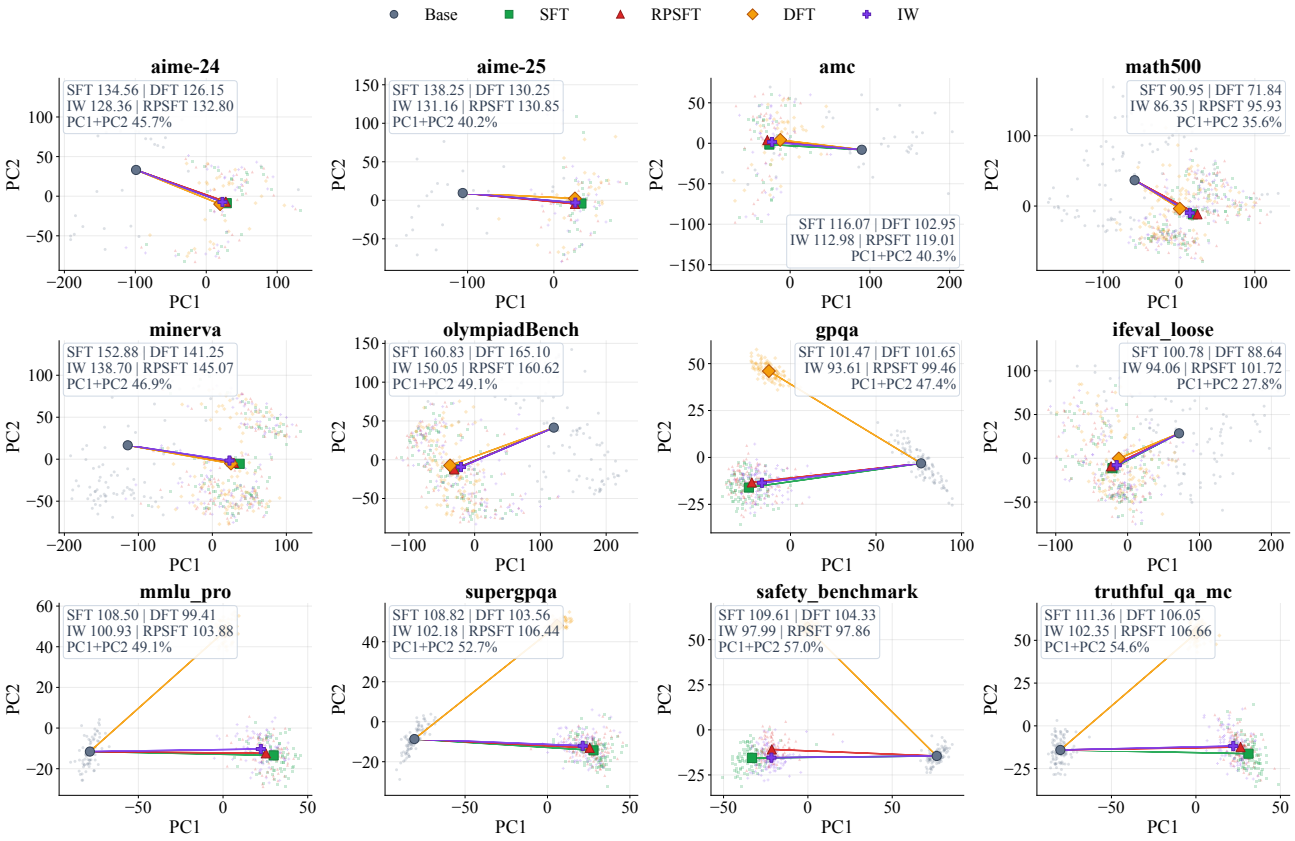


Figure 10. Hidden-state drift on Qwen2.5-7B-Instruct. The PCA view compares centroid shifts away from the base model across benchmark panels. RPSFT is usually among the closest tuned centroids and remains closer than standard SFT in most panels, which is consistent with reduced representation drift after supervised fine-tuning.