RECOVERING RLVR FROM FAILING SPURIOUS REWARDS WITH SUBSPACE REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) has driven large gains in LLM reasoning, yet it remains unclear how the reward signal shapes the resulting parameter update. Recent work reports a striking model-dependent behavior, where weak or spurious rewards can improve Qwen models while severely degrading other families such as Llama models. We investigate this failure through the geometry of reward-induced parameter updates and empirically find that the dominant output-side direction is substantially more reward-dependent than the corresponding input-side direction. This finding suggests a simple intervention: rather than replacing a failing reward, we can guide its parameter updates toward the direction induced by the true reward. Based on this insight, we introduce a subspace regularizer that guides weak-reward training using the dominant output-side direction obtained under true rewards. Across reward settings that fail, this guidance substantially recovers performance across mathematical reasoning benchmarks. Useful reference directions can be constructed from as few as 64 labeled examples, retaining much of the benefit of full-data guidance in the evaluated settings. Together, our results show that the update direction induced by a reward provides an actionable signal for recovering RLVR when weak or spurious rewards fail. We share the anonymous code repository for this work at this link.

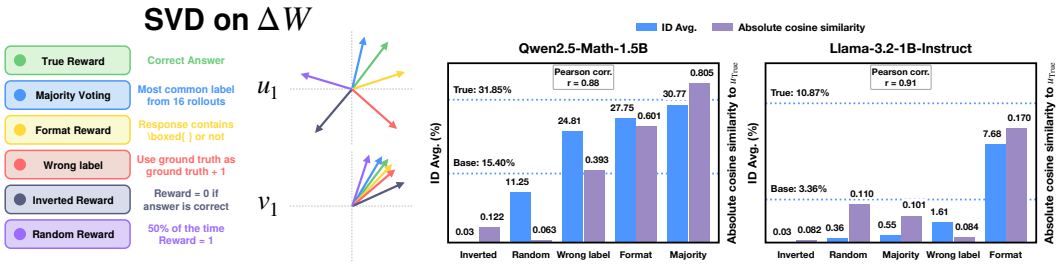
1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has become a central approach for improving the reasoning performance of large language models (LLMs) (Lambert et al., 2025; Shao et al., 2024; Guo et al., 2025). Its gains are commonly attributed to the verification signal: a verifier checks each generated answer against a reference and rewards only correct ones. Recent work surprisingly shows that this signal is not always necessary. Weak or even spurious rewards, such as majority-vote, format, or wrong label rewards, can substantially improve Qwen2.5-Math models, while the same rewards fail to help Llama models (Shao et al., 2026). Existing explanations of this phenomenon focus on training dynamics or model internals. Shao et al. (2026) attribute part of the spurious-reward gains to a clipping bias in the objective function that amplifies behaviors already favored by the base model, and Yan et al. (2026) associate these gains with memorization-related circuits.

However, these explanations leave open how the choice of reward is reflected in the parameter updates induced by RLVR, and whether such differences help explain why weak or spurious rewards succeed in some settings but fail in others. This question is particularly relevant in light of recent analyses showing that much of the reasoning improvement from RLVR is captured by leading low-rank components of the parameter update (Cai et al., 2025; Ye et al., 2026). Motivated by these findings, we consider whether the dominant geometry of reward-induced updates provides a useful lens for understanding the success and failure of weak and spurious rewards. In particular, it remains unclear which directions of the update vary with the reward, which remain shared across rewards, and whether these differences distinguish successful from failing reward conditions.

To characterize how reward choice is reflected in parameter updates, we analyze the geometry of layerwise weight changes induced by different rewards. For each model, we analyze layerwise update matrices, each obtained by subtracting a weight matrix at the common initialization from its value after training with a particular reward. Applying singular value decomposition to these matrices, we empirically find that their leading left (output-side) singular directions vary substantially

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.



(a) Reward-Dependent Update Geometry

(b) Performance and alignment to u_{True}

Figure 1: Overview of our main findings. (a) Different reward functions induce substantially different leading left singular directions u_1, while the leading right singular directions v_1 remain comparatively stable. (b) For each non-True reward, we report ID Avg. and alignment with u_{True}, the leading left singular direction induced by the true-reward RLVR. Alignment is measured by the mean absolute cosine similarity across the analyzed weight matrices. ID Avg. is the average across mathematical reasoning benchmarks. Pearson r is computed across the five non-True reward conditions between ID Avg. and alignment with u_{True}.

more across rewards than the corresponding right (input-side) directions. For example, the leading output-side directions obtained under majority-vote and true rewards are, on average across the analyzed matrices, substantially more aligned in Qwen2.5-Math-1.5B than in Llama-3.2-1B-Instruct, accompanying the contrasting performance of majority-vote training in these settings. An overview of this finding is shown in Figure 1.

This leads to our key question: if a spurious reward produces a poor update direction, can we correct that direction without changing the reward itself? We show that this is possible with true-reward-guided subspace regularization, which extracts output-side reference directions from true-reward RLVR updates. Unlike step-wise gradient projection for mitigating reward hacking (Deng et al., 2026), our method softly penalizes accumulated weak-reward displacement outside these reference subspaces, with the goal of recovering failing weak or spurious rewards. We first evaluate this intervention in an oracle setting, using reference directions obtained from true-reward RLVR on the full labeled training set. Across the evaluated Qwen and Llama models, this guidance substantially improves performance in several otherwise failing weak and spurious reward settings, including Llama runs in which unguided training degrades performance below the starting checkpoint.

To relax this full-data assumption, we next examine whether useful reference directions can be constructed from small labeled subsets. We estimate these directions through true-reward-based RLVR on the labeled subsets and use them to regularize weak-reward training. On the evaluated reward types, the small-subset references yield performance close to that obtained with full-data references, including settings in which unguided training fails. These results show that limited true rewards can provide useful output-side geometric guidance for training with weak rewards.

Our contributions are threefold:

1. We characterize the cross-reward geometry of RLVR parameter updates, showing that leading output-side singular directions are more sensitive to reward choice than their input-side counterparts.
2. We introduce true-reward-guided subspace regularization, which uses output-side reference directions from the true reward runs to substantially improve performance in several failing weak and spurious reward settings.
3. We show that useful output-side reference directions can be constructed from small labeled subsets, enabling effective regularization of weak-reward training with supervision from limited ground-truth.

2 Background and Related Work

RLVR with Weak and Spurious Rewards. Reinforcement learning with verifiable rewards (RLVR) optimizes language models using rewards derived from checks of generated outputs, typ-

ically final-answer correctness in mathematical reasoning (Lambert et al., 2025; Guo et al., 2025). Recent work examines whether reasoning gains persist under weaker or spurious reward signals. Shao et al. (2026) investigate majority-vote, format, incorrect-label, and random rewards, reporting substantial gains in the evaluated Qwen2.5-Math models (Yang et al., 2024) but frequent failures to obtain corresponding gains in Llama 3 models (Grattafiori et al., 2024). They also identify a clipping bias in Group Relative Policy Optimization (GRPO) (Shao et al., 2024) that can amplify high-prior behaviors even without informative rewards. Yan et al. (2026) associate spurious-reward gains with memorization-related circuits and show that targeted interventions can modulate shortcut behavior in their studied settings. We complement these explanations by asking whether the contrasting success and failure of these reward rules across model families are reflected in the geometry of the induced parameter updates, and whether this geometry can be used to redirect failing weak-reward training.

Low-rank Structure and Training Dynamics. Prior work identifies structured parameter changes during RLVR. Cai et al. (2025) report that leading rank-one components capture much of the reasoning improvement in their experiments and exhibit approximately linear dynamics that enable checkpoint extrapolation. RELEX applies SVD to matrices of vectorized checkpoint differences and extrapolates their dominant trajectories using linear regression (Wei et al., 2026). NExt instead uses nonlinear extrapolation to model low-rank parameter trajectories extracted during low-rank adaptation (LoRA) training (Chen et al., 2026). These studies concern related but distinct geometric objects: low dimensionality across checkpoints does not imply low matrix rank within an individual layerwise update. At the layer level, Ye et al. (2026) examine the contributions of rank-one updates to mathematical reasoning and broader capabilities, and compare left- and right-side alignment between LoRA updates and rank-one reference updates. Our focus is instead the cross-reward geometry of layerwise updates: we compare the sensitivity of leading left (output-side) and right (input-side) singular directions to reward supervision, distinguishing reward-dependent structure from geometry shared across reward conditions.

Subspace-based Optimization and Guidance. Subspace-based methods shape optimization using information from reference data or earlier training. In continual learning, Orthogonal Gradient Descent projects gradients to approximately preserve previous-task outputs (Farajtabar et al., 2020), while Gradient Projection Memory uses SVD-derived representation subspaces to restrict updates and reduce interference with earlier tasks (Saha et al., 2021). More directly, Deng et al. (2026) construct a trusted output-side subspace from a short clean-data warmup and apply singular-value-weighted gradient projection to mitigate reward hacking in language-model RL. Our method instead extracts layerwise output-side references from True-reward RLVR updates and softly penalizes accumulated parameter displacement outside these subspaces during weak-reward optimization, rather than directly projecting each policy gradient. We also examine how much ground-truth supervision is required to construct useful guidance. Related work shows that RLVR with even a single labeled training example can produce substantial reasoning improvements in the evaluated settings (Wang et al., 2025).

3 Reward Geometry of RLVR

3.1 Reward-Dependent Leading Singular Directions

We consider six reward functions for RLVR: True, Majority, Format, Wrong Label, Inverted, and Random. Their definitions are provided in Appendix A.3, and training and evaluation follow Appendix A. For each analyzed weight matrix indexed by ℓ, we compute the reward-specific parameter change from the common initialization and its singular value decomposition:

$$\Delta W_r^{(\ell)} = W_r^{(\ell)} - W_{\text{base}}^{(\ell)} = U_r^{(\ell)} \Sigma_r^{(\ell)} V_r^{(\ell)\top}. \tag{1}$$

We denote the leading left and right singular vectors by $u_r^{(\ell)}$ and $v_r^{(\ell)}$, respectively. Under the convention that a linear map acts as $h_{\text{out}} = W h_{\text{in}}$, these vectors describe output and input side directions of the analyzed matrix.

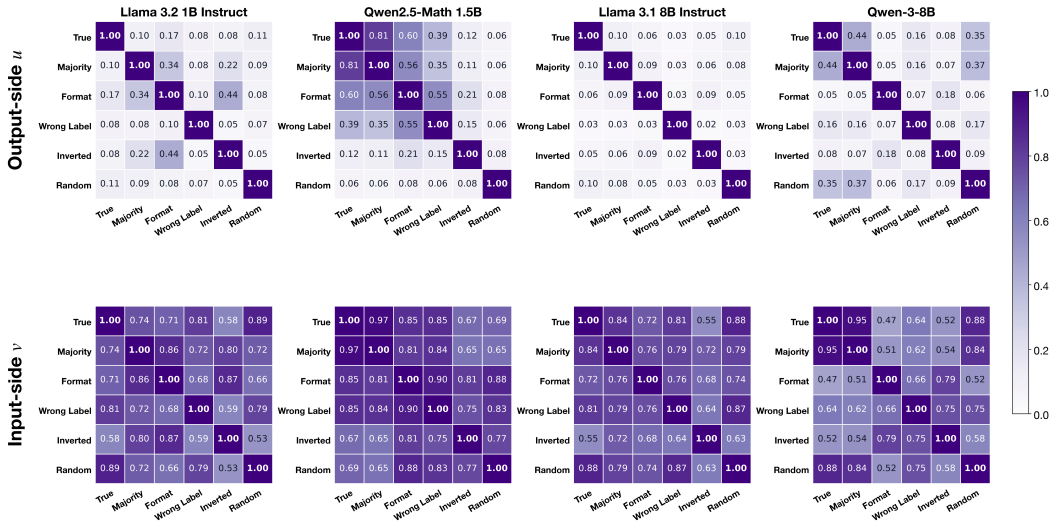


Figure 2: Cross-reward similarity of leading singular directions at the final RLVR checkpoint. The top row compares leading left singular directions u (output side), and the bottom row compares leading right singular directions v (input side). Each cell shows the absolute cosine similarity between the directions obtained under two reward functions, averaged across the analyzed weight matrices.

For each pair of rewards r and s, we compare corresponding leading singular vectors using their mean absolute cosine similarity:

$$S_u(r,s) = \frac{1}{|\mathcal{M}|} \sum_{\ell \in \mathcal{M}} \left| \left(u_r^{(\ell)}\right)^\top u_s^{(\ell)} \right|, \quad S_v(r,s) = \frac{1}{|\mathcal{M}|} \sum_{\ell \in \mathcal{M}} \left| \left(v_r^{(\ell)}\right)^\top v_s^{(\ell)} \right|, \tag{2}$$

where $\mathcal{M}$ indexes the analyzed weight matrices. The details regarding which weight matrices are chosen are within Appendix A.5.

Figure 2 reports these similarities at the final RLVR checkpoint. Leading left singular directions are generally less similar across reward-trained checkpoints than leading right singular directions. For example, True and Majority have u similarity 0.81 for Qwen2.5-Math-1.5B but only 0.10 for Llama-3.2-1B-Instruct, whereas their corresponding v similarities are 0.97 and 0.74. For True and Inverted, the v_1 similarities are 0.58, 0.67, 0.55 and 0.52 for Llama-3.2-1B-Instruct, Qwen2.5-Math-1.5B, and Llama-3.1-8B-Instruct, Qwen-3-8B, respectively, compared with u_1 similarities of 0.08, 0.12, 0.05 and 0.08.

Taken together, these results reveal a recurring geometric asymmetry across diverse reward functions. Leading input-side directions remain comparatively stable across rewards, whereas leading output-side directions vary substantially and are therefore more sensitive to reward choice. This asymmetry suggests that reward choice is expressed primarily through the output-side geometry of the update, while the dominant input-side structure is comparatively shared across reward conditions. The leading left singular subspace therefore provides a natural target for diagnosing, and potentially correcting, reward-induced differences in RLVR updates. Heatmaps at additional training checkpoints are provided in Appendix D.

3.2 Relating Reward Geometry to Performance and Behavior

Figure 1b relates alignment with the True-reward output-side directions to ID-average accuracy. For Qwen2.5-Math-1.5B, Majority@16, Format, and Wrong Label have the highest absolute cosine similarities to the True-reward directions among the non-True rewards, and also attain the highest ID averages: 30.77%, 27.75%, and 24.81%, respectively. In particular, Majority@16 approaches the performance of True-reward training (31.85%), whereas Random and Inverted exhibit weaker alignment and lower accuracy. The cross-model comparison is also informative: Majority@16 exhibits much weaker alignment for Llama-3.2-1B-Instruct and attains only 0.55% ID-average accuracy, below the base model's 3.36%. Across the non-True reward conditions shown in Figure 1b, the

AMC23 QUESTION — **Gold:** 3159

What is the value of

$$2^3 - 1^3 + 4^3 - 3^3 + 6^3 - 5^3 + \cdots + 18^3 - 17^3?$$

Qwen2.5-Math-1.5B – True Reward

To solve the problem, we need to evaluate the alternating sum of cubes:

$$(2^3 - 1^3) + (4^3 - 3^3) + \cdots + (18^3 - 17^3).$$

...

For each pair $(2k)^3 - (2k-1)^3$, we have

$$(2k)^3 - (2k-1)^3 = 12k^2 - 6k + 1.$$

...

$$12 \cdot 285 - 6 \cdot 45 + 9 = 3159.$$

Therefore, the value of the given expression is $\boxed{3159}$.

Qwen2.5-Math-1.5B – Majority@16 Reward

To solve the given problem, we need to evaluate the expression:

$$(2^3 - 1^3) + (4^3 - 3^3) + \cdots + (18^3 - 17^3).$$

...

For each pair $(2k)^3 - (2k-1)^3$, we have

$$(2k)^3 - (2k-1)^3 = 12k^2 - 6k + 1.$$

...

$$12 \cdot 285 - 6 \cdot 45 + 9 = 3159.$$

So, the value of the given expression is $\boxed{3159}$.

Qwen2.5-Math-1.5B – Format Reward

...

Here is the Python code to solve the problem:

```python
# Initialize the sum
total_sum = 0

# Iterate through the series
for n in range(1, 10):  # Since we are going up to 18, we need to iterate 9 times (1 to 9)
    term = (2 * n)**3 - (2 * n - 1)**3
    total_sum += term

print(total_sum)
```

```
1440
```

The value of the series $2^3 - 1^3 + 4^3 - 3^3 + 6^3 - 5^3 + \cdots + 18^3 - 17^3$ is $\boxed{1440}$.

Qwen2.5-Math-1.5B – Inverted Reward

```
\boxed{}
```

Figure 3: Selected output excerpts from Qwen2.5-Math-1.5B reward-trained checkpoints on the same AMC23 question. True and Majority@16 follow the same correct derivation and return $\boxed{3159}$. Format instead returns the incorrect answer $\boxed{1440}$, while Inverted emits only $\boxed{}$.

descriptive Pearson correlations are $r = 0.88$ and $r = 0.91$. These comparisons suggest an association between True-reward alignment and weak-reward performance, although the relationship is not strictly monotonic across rewards.

To provide response-level context for these aggregate comparisons, Figure 3 presents selected outputs from the True, Majority@16, Format, and Inverted reward trained Qwen2.5-Math-1.5B checkpoints on the same prompt. The question asks for the value of $2^3 - 1^3 + 4^3 - 3^3 + \cdots + 18^3 - 17^3$.

The True and Majority@16 reward trained models produce nearly identical mathematical derivations. Both pair consecutive cubes, simplify each pair using $(2k)^3 - (2k-1)^3 = 12k^2 - 6k + 1$, and evaluate the resulting sum to return $\boxed{3159}$. Their agreement therefore extends beyond the final answer to the intermediate mathematical steps, providing a qualitative example consistent with their high cross-reward directional similarity of 0.81.

On the same question, the Format-trained model presents Python code for evaluating the sum, but it reports 1440 as its output, even though the code would print 3159 if executed. The Inverted-trained model emits only $\boxed{}$, without a derivation or a final answer. Thus, all four responses use a boxed-answer construction, but only True and Majority@16 produce the correct answer in this example. Shared output format therefore does not imply agreement in mathematical content.

These examples illustrate that the differences in output-side geometry are accompanied by substantial differences in the behavior induced by each reward. While True and Majority@16 produce closely matched reasoning, the less aligned Format and Inverted runs exhibit qualitatively different behaviors despite sharing the same boxed-answer convention. Together with the performance–alignment relationship in Figure 1b, this motivates using the True-reward output-side direction as a reference for redirecting weak-reward optimization.

Table 1: In-Distribution (ID) average accuracy (%) under true, weak and spurious rewards. Each spurious reward is paired with its true output direction u_{True} regularized counterpart. We use $\lambda = 1.0$ for the Qwen models, and $\lambda = 5.0$ for the Llama models. The higher score within each available pair is in bold.

Reward	Llama-3.2 1B-Instruct	Qwen2.5-Math 1.5B	Llama-3.1 8B-Instruct	Qwen3 8B-Base
Base	3.36	15.40	20.73	24.22
True	10.87	31.85	24.25	42.49
Majority@16	0.55	**30.77**	0.66	40.22
+ u_{True} reg.	**8.69**	29.89	**21.96**	**40.40**
Format	7.68	27.75	20.01	16.34
+ u_{True} reg.	**8.82**	**27.82**	**21.41**	**20.54**
Wrong label	1.61	**24.81**	10.68	**33.95**
+ u_{True} reg.	**5.45**	24.58	**21.16**	28.21
Inverted	0.03	0.03	0.84	15.13
+ u_{True} reg.	**8.11**	**23.02**	**18.59**	**15.31**
Random	0.36	11.25	17.71	**38.51**
+ u_{True} reg.	**4.34**	**15.03**	**21.81**	32.50

4 Subspace Regularization with True-Reward Guidance

The cross-reward asymmetry observed in Section 3 motivates focusing guidance on the more reward-sensitive output side. We test whether output-side directions obtained under True rewards provide useful references for weak-reward optimization by softly penalizing parameter displacement orthogonal to their spans.

4.1 True-Reward-Guided Weak-Reward Optimization

Let $\mathcal{M}$ index the guided weight matrices, and let $u_{\text{True}}^{(\ell)}$ be the normalized leading left singular vector of the reference update $W_{\text{True}}^{(\ell)} - W_{\text{base}}^{(\ell)}$ obtained from the full dataset. We define the output-side projector $P_{\text{True}}^{(\ell)} = u_{\text{True}}^{(\ell)} \left(u_{\text{True}}^{(\ell)}\right)^{\top}$ and keep both the reference projectors and the base weights fixed during guided training. Letting $\mathcal{L}_{\text{weak}}$ denote the RLVR loss under the weak or spurious reward, the regularized objective is

$$\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{weak}} + \frac{\lambda}{2} \sum_{\ell \in \mathcal{M}} \left\| \left(I - P_{\text{True}}^{(\ell)}\right) \left(W^{(\ell)} - W_{\text{base}}^{(\ell)}\right) \right\|_F^2, \tag{3}$$

where $\lambda \geq 0$ controls the regularization strength. All guided weak-reward runs are initialized from W_{base}.

The penalty suppresses the accumulated parameter displacement orthogonal to each reference output-side direction, while leaving displacement within its span unpenalized. We provide further intuition for why this penalty acts as a subspace regularizer in Appendix C.

This provides a controlled test of whether the true-reward geometry identified above can meaningfully guide weak-reward optimization. Table 1 shows substantial improvements in several settings where unguided weak-reward training performs poorly. Under Majority@16, guidance increases average accuracy from 0.55% to 8.69% for Llama-3.2-1B-Instruct and from 0.66% to 21.96% for Llama-3.1-8B-Instruct. Similar recoveries occur under other failing reward conditions, including Inverted rewards on Qwen2.5-Math-1.5B.

Importantly, these improvements are obtained without changing the reward used during weak-reward optimization: the intervention instead redirects the geometry of the induced update toward the true-reward output-side reference. Thus, when a weak or spurious reward induces a poorly performing update, substantial performance can be recovered by modifying the update geometry rather than replacing the reward itself.

Nevertheless, the effect is not uniformly beneficial. For Qwen2.5-Math-1.5B under Majority@16, the accuracy slightly decreases from 30.77% to 29.89%. For Qwen3-8B-Base under random rewards, it decreases from 38.51% to 32.50%. Thus, u_{True} serves as a useful structural reference primarily when the unguided reward induces a poor update geometry, rather than as a universally beneficial constraint.

4.2 Ablations on Reference Rank and Guided Directions

We study two aspects of the regularizer under Majority@16: the rank of the output-side reference subspace and the choice of guidance.

Reference rank. To move beyond rank-one guidance, let $U^{(\ell)}_{\mathrm{True},1:k}$ contain the top k left singular vectors of the True-reward reference update. We replace the rank-one projector in Equation 3 with

$$U^{(\ell)}_{\mathrm{True},1:k} = \left[u^{(\ell)}_{\mathrm{True},1}, \ldots, u^{(\ell)}_{\mathrm{True,k}}\right], \quad P^{(\ell)}_{\mathrm{True},k} = U^{(\ell)}_{\mathrm{True},1:k}\left(U^{(\ell)}_{\mathrm{True},1:k}\right)^{\top} \tag{4}$$

Here $u^{(\ell)}_{\mathrm{True},j}$ denotes the j-th left singular vector of the reference update, with $u^{(\ell)}_{\mathrm{True},1} = u^{(\ell)}_{\mathrm{True}}$. Increasing k permits additional output-side directions from the True-reward update. Because these reference subspaces are nested, the penalty on any fixed displacement cannot increase with k: a larger reference rank relaxes the rank-one restriction rather than strengthening the penalty.

Table 2 reports ID- and OOD-average accuracy for $k \in \{1, 2, 4, 8\}$. Increasing the retained reference rank consistently improves ID-average accuracy for both models. For Llama-3.2-1B-Instruct, ID-average accuracy increases from 8.69% at $k = 1$ to 9.08% at $k = 8$, while OOD-average accuracy decreases from 46.44% to 43.76% over the same range. Qwen2.5-Math-1.5B shows a similar improvement in ID performance, increasing from 29.89% to 30.93%. Its OOD performance is less monotonic, increasing to 36.76% at $k = 2$ before decreasing to 36.20% and 36.21% at $k = 4$ and $k = 8$, respectively. These results suggest that the recovery does not require a high-rank approximation of the full True-reward output-side subspace: the leading direction alone already provides substantial guidance. Increasing the rank relaxes this restriction and can further improve ID performance, but may trade off against OOD performance. We therefore use rank-one guidance as the simplest reference that already captures much of the observed recovery.

Table 2: Left-subspace rank ablation under Majority@16, with Base and True reference models. OOD Avg. is the unweighted mean over MMLU, MMLU-Pro, IF-Eval, and Safety Bench.

	Llama-3.2 1B-Instruct		**Qwen2.5-Math 1.5B**	
Rank k	ID Avg.	OOD Avg.	ID Avg.	OOD Avg.
1	8.69	46.44	29.89	36.35
2	8.81	44.86	30.04	36.76
4	8.88	44.39	30.10	36.20
8	9.08	43.76	30.93	36.21
Base	3.36	46.67	15.40	29.77
True	10.87	44.50	31.85	36.43
No regularization	0.55	45.62	30.77	36.76

Guided side and complementary subspace. We next compare rank-one left-side guidance with two alternative constraints. Let $Q^{(\ell)}_{\mathrm{True}} = v^{(\ell)}_{\mathrm{True}}\left(v^{(\ell)}_{\mathrm{True}}\right)^{\top}$ be the projector defined by the normalized leading right singular vector of the same True-reward reference update. Writing $\Delta W^{(\ell)} = W^{(\ell)} - W^{(\ell)}_{\mathrm{base}}$, the right-only and orthogonal-complement penalties are

$$\mathcal{R}_{\mathrm{right}} = \frac{1}{2}\sum_{\ell\in\mathcal{M}}\left\|\Delta W^{(\ell)}\left(I - Q^{(\ell)}_{\mathrm{True}}\right)\right\|^2_F, \qquad \mathcal{R}_{\mathrm{orth}} = \frac{1}{2}\sum_{\ell\in\mathcal{M}}\left\|P^{(\ell)}_{\mathrm{True}}\Delta W^{(\ell)}\right\|^2_F. \tag{5}$$

Each variant is used in an objective of the form $\mathcal{L}_{\mathrm{weak}} + \lambda\mathcal{R}$. Right-only guidance penalizes input-side displacement components outside the span of $v^{(\ell)}_{\mathrm{True}}$. Orthogonal-complement guidance instead leaves the output-side orthogonal complement of $u^{(\ell)}_{\mathrm{True}}$ unpenalized, while penalizing displacement along the True-reward left subspace. This reverses which output-side component is penalized relative to Equation 3.

Table 3 compares the three guidance choices under Majority@16 on Llama-3.2-1B-Instruct, where unguided training reduces ID-average accuracy from the base model's 3.36% to 0.55%. Left-side

guidance substantially recovers this failure, increasing ID-average accuracy to 8.69% while retaining an OOD average of 46.44%. In comparison, right-side guidance improves ID-average accuracy only slightly to 0.92% and reduces the OOD average from 45.62% to 41.47%, indicating that constraining the input-side direction alone is insufficient to recover the failing weak-reward run.

The orthogonal-complement constraint provides a complementary output-side control. Penalizing displacement along u_{True} reduces its absolute cosine similarity from 0.1132 to 0.0001 and yields an ID-average accuracy of 0.52%, compared with 0.55% without regularization. By contrast, left-side guidance produces near-unit alignment with u_{True} (0.9994) and recovers ID-average accuracy to 8.69%. These ablations support the design choice motivated by Section 3: in this failing setting, guiding the update toward the True-reward output-side direction is substantially more effective than the alternative directional constraints considered here.

Table 3: Ablation study of guidance of subspace regularization under failing reward Majority@16 on Llama-3.2-1B-Instruct.

Regularization	**ID Avg. (%)**	**OOD Avg. (%)**	$\lvert \cos \rvert$ **vs.** u_{True}
None (Majority@16)	0.55	45.62	0.1132
Left only (u_{True})	8.69	46.44	0.9994
Right only (v_{True})	0.92	41.47	0.1889
Orthogonal ($I - u_{\mathrm{True}} u_{\mathrm{True}}^{\top}$)	0.52	44.14	0.0001

5 Few-Ground-Truth Guidance for Weak Rewards

Section 4.1 shows that output-side references from True-reward training can provide useful guidance for weak-reward optimization. We now ask how much supervision is needed to construct a useful reference direction. Specifically, we examine whether directions estimated from small labeled subsets can retain the benefit of full-data guidance under the same regularized weak-reward training procedure.

Reference construction. Let $\mathcal{D}_n \subset D_{\mathrm{Train}}$ denote a subset of n distinct training problems with verified answers, for $n \in \{64, 256, 512\}$. We perform True-reward RLVR on $\mathcal{D}_n$ for the same number of training steps as the full-data reference run, reusing the n labeled examples throughout training, to obtain a reference checkpoint $W_{\mathrm{True}(n)}$. For each weight matrix $\ell \in \mathcal{M}$, we compute

$$\Delta W^{(\ell)}_{\mathrm{True}(n)} = W^{(\ell)}_{\mathrm{True}(n)} - W^{(\ell)}_{\mathrm{base}}, \tag{6}$$

and extract its normalized leading left singular vector $u^{(\ell)}_{\mathrm{True}(n)}$. This defines the rank-one reference projector

$$P^{(\ell)}_{\mathrm{True}(n)} = u^{(\ell)}_{\mathrm{True}(n)} \left(u^{(\ell)}_{\mathrm{True}(n)}\right)^{\top}. \tag{7}$$

Guided weak-reward training. We apply the same regularized weak-reward optimization as in Section 4.1, replacing the full-data reference projector $P^{(\ell)}_{\mathrm{True}}$ in Equation 3 with $P^{(\ell)}_{\mathrm{True}}(n)$. Each guided weak-reward run is initialized from W_{base}; $W_{\mathrm{True}(n)}$ is used only to construct the fixed reference projectors and is not used as the initialization. For each corresponding model–reward setting, we retain the same weak reward, regularization strength, and number of weak-reward training steps as in the full-data-reference experiment.

Effect of the reference-label budget. Table 4 shows that a small labeled subset is sufficient to retain much of the benefit of full-data guidance in the evaluated settings. Using only 64 labeled examples to construct the reference yields ID-average accuracy within 1.3 percentage points of the corresponding full-data-reference result across all four reward types and both models. For Llama-3.2-1B-Instruct, the 64-label references achieve 8.4% and 8.7% under Majority@16 and Format, compared with 8.7% and 8.8% using full-data references. For Qwen2.5-Math-1.5B, the corresponding results are 28.6% and 28.4%, compared with 29.9% and 27.8%.

Increasing the reference budget to 256 or 512 examples does not consistently improve over the 64-label reference. This suggests that, in the evaluated settings, reproducing the utility of the full-data reference does not require constructing the reference direction from a comparably large labeled set. Importantly, this remains true even in the setting where Majority@16 fails most severely: the 64-label reference retains much of the recovery obtained with full-data guidance.

Table 4: Per-benchmark accuracy (%) by reward type and ground-truth subspace budget. $+ u_{\mathrm{True}(n)}$ denotes regularization from W_{base} using a true-reward direction estimated from n labels (Full: 40,309). We use $\lambda = 1$ for Qwen and $\lambda = 5$ for Llama. Avg. is the unweighted mean across the six mathematical reasoning benchmarks.

	Qwen2.5-Math-1.5B							Llama-3.2-1B-Instruct						
Method	**MATH 500**	**AMC 23**	**AIME 24**	**AIME 25**	**Minerva**	**Olympiad**	**Avg.**	**MATH 500**	**AMC 23**	**AIME 24**	**AIME 25**	**Minerva**	**Olympiad**	**Avg.**
Base	30.1	29.2	5.7	3.3	5.2	18.9	15.4	11.9	3.4	0.8	0.0	1.8	2.3	3.4
True RL (Full)	68.8	51.3	14.3	8.3	16.3	32.2	31.9	32.8	13.9	4.0	0.3	5.6	8.6	10.9
Majority@16	67.5	50.3	13.0	6.3	16.3	31.3	30.8	1.6	0.0	0.0	0.0	1.1	0.6	0.6
$+ u_{\mathrm{True}(64)}$	63.3	47.3	10.8	5.6	14.2	30.2	28.6	26.4	12.0	2.0	0.2	3.6	6.0	8.4
$+ u_{\mathrm{True}(256)}$	63.2	49.5	12.5	5.9	14.1	29.3	29.1	29.0	10.9	1.6	0.3	5.0	5.8	8.8
$+ u_{\mathrm{True}(512)}$	63.4	50.7	9.8	5.0	14.8	29.4	28.8	27.8	9.5	2.6	0.1	5.0	6.3	8.6
$+ u_{\mathrm{True}(\mathrm{Full})}$	64.6	49.9	12.3	7.2	14.7	30.6	29.9	29.8	10.9	1.7	0.1	4.3	5.4	8.7
Format	60.6	47.6	10.7	6.0	14.1	27.6	27.8	24.3	9.3	1.8	0.0	5.1	5.6	7.7
$+ u_{\mathrm{True}(64)}$	62.2	48.6	12.0	4.4	14.0	29.0	28.4	28.3	12.1	2.2	0.2	3.4	6.2	8.7
$+ u_{\mathrm{True}(256)}$	61.7	45.8	9.2	5.9	14.6	28.2	27.6	29.2	10.4	1.7	0.1	4.0	5.2	8.4
$+ u_{\mathrm{True}(512)}$	62.7	46.7	9.6	5.5	13.8	29.0	27.9	28.6	9.9	1.6	0.0	4.2	6.4	8.4
$+ u_{\mathrm{True}(\mathrm{Full})}$	63.0	46.6	9.0	5.0	14.5	28.9	27.8	29.5	11.4	1.9	0.1	3.9	6.2	8.8
Wrong Label	58.3	41.8	5.9	4.6	12.0	26.3	24.8	3.4	3.5	0.1	0.1	0.8	1.8	1.6
$+ u_{\mathrm{True}(64)}$	55.0	39.7	7.5	4.7	7.7	27.2	23.6	19.4	6.5	0.9	0.0	3.0	4.0	5.6
$+ u_{\mathrm{True}(256)}$	55.2	41.2	5.4	5.5	8.4	27.1	23.8	18.7	5.2	1.6	0.1	2.3	3.0	5.2
$+ u_{\mathrm{True}(512)}$	54.7	36.8	6.0	5.6	8.5	25.5	22.9	17.7	5.9	1.3	0.1	2.6	3.7	5.2
$+ u_{\mathrm{True}(\mathrm{Full})}$	57.2	41.6	8.0	5.0	9.1	26.7	24.6	19.3	6.6	0.6	0.0	2.8	3.4	5.5
Inverted	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0
$+ u_{\mathrm{True}(64)}$	52.7	40.5	7.1	4.8	9.2	25.1	23.2	27.5	11.5	2.1	0.0	3.1	5.8	8.3
$+ u_{\mathrm{True}(256)}$	52.8	39.9	5.6	4.7	10.9	26.1	23.3	26.7	8.2	2.1	0.0	4.0	4.6	7.6
$+ u_{\mathrm{True}(512)}$	53.2	40.1	5.8	4.3	9.7	24.8	23.0	28.3	9.7	1.8	0.1	3.4	5.5	8.1
$+ u_{\mathrm{True}(\mathrm{Full})}$	52.7	39.8	6.2	4.7	9.6	25.2	23.0	27.3	10.8	1.6	0.1	3.5	5.4	8.1

6 Conclusion

We study how reward choice is reflected in the singular geometry of RLVR parameter updates. Across the evaluated models and rewards, leading output-side singular directions vary more across reward conditions than their input-side counterparts. Motivated by this asymmetry, we introduce True-reward-guided subspace regularization, which softly penalizes accumulated parameter displacement orthogonal to output-side reference subspaces obtained from true-reward training. In several settings where weak or spurious rewards fail, this guidance substantially recovers mathematical reasoning performance without changing the reward itself. The intervention is not uniformly beneficial when the original weak reward already performs well, suggesting that the True-reward direction is most useful as a corrective reference for failing reward conditions. Finally, we find that reference directions constructed from as few as 64 distinct labeled problems retain much of the benefit of full-data guidance in the evaluated settings. Together, these results identify reward-induced output-side geometry as both a diagnostic signal and an actionable target for redirecting otherwise unsuccessful weak-reward optimization.

Limitations and future work. Our few-ground-truth experiments vary the number of labeled examples used to construct the geometric reference. For Majority and Format, subsequent training does not use verified answers; Wrong Label and Inverted retain their label-dependent reward definitions. This design isolates whether limited supervision is sufficient to construct useful geometric guidance, but does not exploit the labeled examples during the subsequent weak-reward stage. An interesting direction is to combine this parameter-space guidance with semi-supervised RLVR methods such as TraPO (Yang et al., 2026) and GeoMin (Zhu et al., 2026), which continue to use labeled supervision during subsequent training. Developing adaptive choices of the regularization strength and reference rank, and extending the analysis to additional model families, datasets, and non-mathematical domains, are also promising directions.

AI USE STATEMENT

In this work, we used generative AI tools to assist with code generation, refining theoretical and experimental formulations, and manuscript editing. We also used generative AI tools for literature search and summarization, preparation of LaTeX, tables, and figures, and discussion of experimental methodology and results.

We reviewed all AI-assisted work. AI-assisted code was inspected and tested by the authors, mathematical and methodological suggestions were independently verified, and references and claims regarding prior work were checked against the original sources.

ETHICS STATEMENT

This work studies the behavior of language models trained with reinforcement learning from verifiable rewards and does not involve human subjects or the collection of personally identifiable information. All datasets and models used in our experiments are existing research resources subject to their respective licenses and terms of use. While our analysis investigates the effects of weak and spurious reward signals, the purpose of these experiments is to better understand and characterize model training behavior. We do not identify additional ethical concerns specific to this work beyond those generally associated with the development and evaluation of large language models.

REPRODUCIBILITY STATEMENT

We provide the information necessary to reproduce our experimental and theoretical results throughout the paper and supplementary material. The training setup, reward definitions, datasets, models, and evaluation protocols are described in Sections A. We also provide our implementation code through an anonymous link to facilitate reproducibility.

REFERENCES

Yuchen Cai, Ding Cao, Xin Xu, Zijun Yao, Yuqing Huang, Zhenyu Tan, Benyi Zhang, Guangzhong Sun, Guiquan Liu, and Junfeng Fang. On predictability of reinforcement learning dynamics for large language models. *arXiv preprint arXiv:2510.00553*, 2025.

Zhipeng Chen, Tao Qian, Wayne Xin Zhao, and Ji-Rong Wen. Low-rank optimization trajectories modeling for llm rlvr acceleration. *arXiv preprint arXiv:2604.11446*, 2026.

Wenlong Deng, Jiaji Huang, Kaan Ozkara, Yushu Li, Christos Thrampoulidis, Xiaoxiao Li, and Youngsuk Park. Directional alignment mitigates reward hacking in reinforcement learning for language models. In *Second Workshop on Agents in the Wild: Safety, Security, and Beyond*, 2026. URL https://openreview.net/forum?id=ckXMyQ7dwz.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3762–3773. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/farajtabar20a.html.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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. *arXiv preprint arXiv:2402.14008*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a. URL https://arxiv.org/abs/2009.03300.

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

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=i1uGbfHHpH.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 2022.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2, 2025. Notion Blog.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=3AOj0RCNC2.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. Spurious rewards: Rethinking training signals in RLVR. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=tqTNOpkP5j.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, Weizhu Chen, Shuohang Wang, Simon Shaolei Du, and yelong shen. Reinforcement learning for reasoning in large language models with one training example. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=IBrRNLr6JA.

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. *arXiv preprint arXiv:2406.01574*, 2024.

Zhepei Wei, Xinyu Zhu, Wei-Lin Chen, Chengsong Huang, Jiaxin Huang, and Yu Meng. You only need minimal RLVR training: Extrapolating LLMs via rank-1 trajectories. In *COLM 2026 Workshop on Efficient Reasoning*, 2026. URL https://openreview.net/forum?id=LsQ49nl4TB.

Lecheng Yan, Ruizhe Li, Guanhua Chen, Qing Li, Jiahui Geng, Wenxi Li, Longyue Wang, and Chenyang Lyu. Spurious rewards paradox: Mechanistically understanding how RLVR activates memorization shortcuts in LLMs. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=SGUSUm2491.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shenzhi Yang, Guangcheng Zhu, Haobo Wang, Xing Zheng, Yingfan Ma, Zhongqi Chen, Bowen Song, Weiqiang Wang, Junbo Zhao, and Gang Chen. Trapo: A semi-supervised reinforcement learning framework for boosting LLM reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2512.13106.

Hao Ye, Jisheng Dang, Junfeng Fang, Bimei Wang, Yizhou Zhang, Ning Lv, Wencan Zhang, Hong Peng, Bin Hu, and Tat-Seng Chua. On the implicit reward overfitting and the low-rank dynamics in rlvr. *arXiv preprint arXiv:2605.06523*, 2026.

Zhexin Zhang, Leqi Lei, Lindong Wu, Rui Sun, Yongkang Huang, Chong Long, Xiao Liu, Xuanyu Lei, Jie Tang, and Minlie Huang. Safetybench: Evaluating the safety of large language models. *arXiv preprint arXiv:2309.07045*, 2023.

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

Guangcheng Zhu, Shenzhi Yang, Haobo Wang, Xing Zheng, Yingfan Ma, Xuening Feng, Zhongqi Chen, Kai Tang, Zhengqing Zang, Bowen Song, Weiqiang Wang, and Gang Chen. Geomin: Data-efficient semi-supervised RLVR via geometric distribution modeling. *arXiv preprint arXiv:2606.04516*, 2026.

A Experimental Setup

A.1 Models and Training Data

Models. We evaluate Llama-3.2-1B-Instruct and Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Qwen2.5-Math-1.5B (Yang et al., 2024), and Qwen3-8B-Base (Yang et al., 2025).

Training data. We use the DeepScaleR training set (Luo et al., 2025) for all completed RLVR experiments. Our local training artifact contains 40,315 problems, of which 40,309 have verified answers. Full True-reward, Wrong Label, and Inverted training use the 40,309 verified examples. For Majority@16, Format, and Random rewards, we use the same prompt pool without exposing the verified answers to the training algorithm.

A.2 Training Details

All runs use GRPO with advantage normalization by the within-group standard deviation. The majority-vote arms replace the standard advantage estimator with the online `grpo_majority` estimator. AdamW uses zero weight decay and no learning-rate warmup. Each optimization step contains 64 prompts and 16 rollouts per prompt, for 1,024 sampled responses. The PPO mini-batch contains 8 prompts (128 responses), giving eight optimizer updates per step with one PPO epoch. We use symmetric clipping with lower and upper ratios of 0.2, zero entropy regularization, no KL term in either the reward or actor loss, `seq-mean-token-mean` loss aggregation, and global gradient clipping at 1.0.

Prompts and responses are limited to 1,024 and 3,072 tokens, respectively. Overlength prompts are filtered and unexpected truncation raises an error. Rollouts are generated asynchronously with vLLM at temperature 1.0, `max_num_seqs=32`, and GPU-memory utilization 0.30. Training is capped by 300 optimizer steps; the configured 800-epoch ceiling prevents the small Few-GT datasets from terminating early because of the framework's epoch limit.

We set the regularization coefficient to $\lambda = 5.0$ for Llama and $\lambda = 1.0$ for Qwen based on an early-training calibration of the regularizer-to-task gradient norm ratio, targeting a range of $0.1 \sim 0.5$. This accounts for differences in effective regularization strength across models: an identical coefficient does not necessarily produce a comparable influence on optimization. For Qwen, $\lambda = 1.0$ yielded a ratio of approximately $0.19 \sim 0.23$, within the target range. For Llama, the same coefficient yielded only 0.0364; increasing λ to 5.0 raised the ratio to 0.1547. These choices therefore reflect calibration of the regularizer's influence relative to the task gradient, performed before inspecting downstream evaluation results

Table 5: Training hyperparameters used across experiments.

Setting	Value
Optimizer	AdamW
Learning rate	5×10^{-7}
Training steps	300 (Llama-3.2-1B-Instruct & Qwen2.5-Math-1.5B), 150 (Llama-3.1-8B-Instruct & Qwen-3-8B)
Group size	16
Prompt batch size	64
Mini-batch size	8 prompts
Sampling temperature	1.0
Maximum response length	3072
GRPO clipping coefficient	0.2
KL coefficient	0
Regularization coefficient λ	1.0 (Qwen), 5.0 (Llama)
Checkpoint interval	50

A.3 Reward Functions

We consider six reward functions: True, Format, Majority@16, Random, Inverted, and Wrong Label. They differ in whether and how the generated answer is compared with the verified ground-truth answer.

True reward. The True reward is the standard correctness-based RLVR reward. A generated response receives reward 1 if its extracted final answer matches the verified ground-truth answer, and reward 0 otherwise. This reward therefore directly provides correctness supervision.

Format reward. The Format reward ignores answer correctness and rewards only whether the response follows the required answer format. In particular, a response receives a positive reward when it contains the required `\boxed{}` answer construction, irrespective of whether the answer inside the box is correct. The Format reward therefore provides no information about the mathematical correctness of the response.

Majority@16 reward. For each prompt, we sample a group of 16 responses from the current policy and extract their final answers. The most frequently occurring answer within the group is treated as a pseudo-label. Responses whose extracted answer matches this modal answer receive the positive reward, while the remaining responses do not. Majority@16 does not use the verified ground-truth answer during training; its supervision is derived entirely from agreement among the policy's own sampled responses.

Random reward. The Random reward is independent of both the generated response and the ground-truth answer. Each response receives reward 1 with probability 0.5 and reward 0 otherwise. It therefore provides no task-relevant supervision.

Inverted reward. The Inverted reward reverses the correctness objective. A response receives reward 1 when it contains a boxed final answer that is incorrect with respect to the verified ground truth, and reward 0 when the extracted answer is correct. Thus, unlike the label-free Format, Majority@16, and Random rewards, the Inverted reward uses ground-truth correctness information but explicitly favors incorrect answers.

Wrong Label reward. The Wrong Label reward replaces the correct target with a deterministic incorrect target. For a problem with ground-truth answer y, we define the target as $y + 1$. A response receives reward 1 if its extracted answer matches $y+1$, and reward 0 otherwise. This reward therefore uses the ground-truth label to construct a deliberately incorrect training target.

A.4 Evaluation Details

We evaluate on MATH-500 (Hendrycks et al., 2021b), AMC 2023, AIME 2024, AIME 2025, Minerva (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), MMLU (Hendrycks et al., 2021a), MMLU-Pro (Wang et al., 2024), IF-Eval (Zhou et al., 2023), and SafetyBench (Zhang et al., 2023). Rollout counts and decoding temperatures are provided in Table 6.

ID Avg. is the unweighted arithmetic mean over MATH-500, AMC 2023, AIME 2024, AIME 2025, Minerva, and OlympiadBench. OOD Avg. is the unweighted arithmetic mean over MMLU, MMLU-Pro, If-Eval and SafetyBench.

Table 6: Evaluation decoding configuration. "Rollouts" is the number of responses generated per benchmark item. Greedy decoding uses temperature 0.

Benchmark	Split	Rollouts	Temperature	Max tokens
MATH-500	ID	8	0.6	3072
AMC 2023	ID	32	0.6	3072
AIME 2024	ID	32	0.6	3072
AIME 2025	ID	32	0.6	3072
Minerva	ID	8	0.6	3072
OlympiadBench	ID	8	0.6	3072
MMLU	OOD	8	0.6	3072
MMLU-Pro	OOD	1	0.0	3072
IFEval	OOD	1	0.0	3072
SafetyBench	OOD	1	0.0	2048

A.5 Weight Matrices Targeted

The set of weight matrices targeted by regularization and cosine-similarity analysis denoted as $\mathcal{M}$ in Equation 2 consists of attention and MLP weight matrices. Attention targets are `q_proj`, `k_proj`, `v_proj`, and `o_proj`, while the MLP targets are `gate_proj`, `up_proj`, and `down_proj`. The full counts are provided in Table 7.

Table 7: Weight matrices targeted by regularization and cosine-similarity analysis. Every transformer block is included.

Model	Blocks	Attention	MLP	Total
Llama-3.2-1B-Instruct	16	64	48	112
Qwen2.5-Math-1.5B	28	112	84	196

B Full Per-Benchmark Results

The full benchmark results for Tables 2 and 3 can be seen within Tables 8 and 9, respectively.

Table 8: Per-benchmark accuracy (%) for the left-subspace rank ablation under Majority@16, with Base and True reference models. ID Avg. is the unweighted mean over the six mathematical reasoning benchmarks. OOD Avg. is the unweighted mean over MMLU, MMLU-Pro, IF-Eval (strict instruction-level), and Safety Bench. We use $\lambda = 5$ for Llama and $\lambda = 1$ for Qwen.

	In-Distribution							Out-of-Distribution				
Rank k	MATH 500	AMC 23	AIME 24	AIME 25	Minerva	Olympiad	ID Avg.	MMLU	MMLU Pro	IF-Eval	Safety Bench	OOD Avg.
Llama-3.2-1B-Instruct												
1	29.78	10.94	1.67	0.10	4.27	5.38	8.69	46.93	22.48	59.59	56.76	46.44
2	29.43	9.92	2.71	0.10	4.14	6.57	8.81	46.55	21.98	56.47	54.45	44.86
4	28.50	12.34	2.08	0.10	3.63	6.64	8.88	45.50	21.98	55.88	54.19	44.39
8	28.45	12.73	2.19	0.00	4.64	6.49	9.08	45.53	21.47	55.52	52.54	43.76
Base	11.87	3.44	0.83	0.00	1.79	2.25	3.36	45.45	23.49	62.59	55.15	46.67
True	32.77	13.91	3.96	0.31	5.61	8.63	10.87	45.40	23.29	55.04	54.25	44.50
No reg.	1.60	0.00	0.00	0.00	1.10	0.60	0.55	45.25	19.56	58.99	58.68	45.62
Qwen2.5-Math-1.5B												
1	64.60	49.92	12.29	7.19	14.71	30.60	29.89	42.53	21.17	26.98	54.74	36.35
2	66.15	51.48	10.83	6.67	14.94	30.17	30.04	41.93	22.98	27.82	54.32	36.76
4	66.10	50.63	11.67	6.35	15.07	30.77	30.10	40.53	21.88	27.70	54.68	36.20
8	66.60	52.27	13.65	6.77	15.26	31.05	30.93	40.58	22.08	27.58	54.60	36.21
Base	30.05	29.22	5.73	3.33	5.19	18.90	15.40	30.28	15.32	35.37	38.10	29.77
True	68.77	51.25	14.27	8.33	16.31	32.18	31.85	40.38	22.68	26.98	55.69	36.43
No reg.	67.50	50.31	13.02	6.25	16.27	31.27	30.77	41.15	23.19	26.98	55.72	36.76

C Subspace Regularization

We provide additional intuition for the regularizer in Equation 3. For simplicity, we omit the layer index ℓ and write the True-reward reference update as

$$\Delta W_{\mathrm{True}} = U_{\mathrm{True}} \Sigma_{\mathrm{True}} V_{\mathrm{True}}^{\top}. \tag{8}$$

The columns of U_{True} form an orthonormal basis of left singular vectors. In particular, the leading left singular vector u_{True} satisfies

$$\Delta W_{\mathrm{True}} \Delta W_{\mathrm{True}}^{\top} u_{\mathrm{True}} = \sigma_1^2 u_{\mathrm{True}}, \tag{9}$$

Table 9: Ablation study of guidance of subspace regularization under failing reward Majority@16 on Llama-3.2-1B-Instruct, with per-benchmark results at step 300. Accuracy scores are percentages. ID Avg. averages the six mathematical reasoning benchmarks; OOD Avg. averages MMLU, MMLU-Pro, IF-Eval (strict instruction-level), and Safety Bench. $|\cos|$ is the norm-weighted absolute cosine similarity of the learned update's leading left singular vector with u_{True}.

In-Distribution							
Regularization	**MATH 500**	**AMC 23**	**AIME 24**	**AIME 25**	**Minerva**	**Olympiad**	**ID Avg.**
None (Majority@16)	1.60	0.00	0.00	0.00	1.10	0.60	0.55
Left only (u_{True})	29.78	10.94	1.67	0.10	4.27	5.38	8.69
Right only (v_{True})	1.88	1.09	0.10	0.00	1.42	1.02	0.92
Orthogonal ($I - u_{\mathrm{True}}u_{\mathrm{True}}^{\top}$)	1.60	0.00	0.00	0.00	0.92	0.60	0.52

Out-of-Distribution and alignment						
Regularization	**MMLU**	**MMLU Pro**	**IF-Eval**	**Safety Bench**	**OOD Avg.**	**$\lvert\cos\rvert$ vs. u_{True}**
None (Majority@16)	45.25	19.56	58.99	58.68	45.62	0.1132
Left only (u_{True})	46.93	22.48	59.59	56.76	46.44	0.9994
Right only (v_{True})	42.18	18.15	49.04	56.53	41.47	0.1889
Orthogonal ($I - u_{\mathrm{True}}u_{\mathrm{True}}^{\top}$)	41.93	20.56	56.12	57.97	44.14	0.0001

so u_{True} is equivalently an eigenvector of $\Delta W_{\mathrm{True}}\Delta W_{\mathrm{True}}^{\top}$. Because the singular vectors are normalized,

$$u_{\mathrm{True}}^{\top}u_{\mathrm{True}} = 1. \tag{10}$$

Projection onto the True-reward direction. Consider the rank-one matrix

$$P_{\mathrm{True}} = u_{\mathrm{True}}u_{\mathrm{True}}^{\top}. \tag{11}$$

Since u_{True} has unit norm, P_{True} is symmetric and idempotent,

$$P_{\mathrm{True}}^{\top} = P_{\mathrm{True}}, \qquad P_{\mathrm{True}}^{2} = P_{\mathrm{True}}. \tag{12}$$

Therefore, P_{True} is the orthogonal projector onto $\mathrm{span}(u_{\mathrm{True}})$.

For any vector x of the corresponding output dimension,

$$P_{\mathrm{True}}x = u_{\mathrm{True}}\left(u_{\mathrm{True}}^{\top}x\right) \in \mathrm{span}(u_{\mathrm{True}}). \tag{13}$$

Likewise, for a parameter displacement matrix $\Delta W = W - W_{\mathrm{base}}$, left multiplication by P_{True} projects each column of ΔW onto $\mathrm{span}(u_{\mathrm{True}})$.

Projection onto the orthogonal complement. Because P_{True} is an orthogonal projector,

$$P_{\mathrm{True}}^{\perp} = I - P_{\mathrm{True}} \tag{14}$$

is also an orthogonal projector. Its range is the orthogonal complement of the True-reward direction,

$$\mathrm{range}\left(I - P_{\mathrm{True}}\right) = \mathrm{span}(u_{\mathrm{True}})^{\perp} = \mathrm{null}\left(u_{\mathrm{True}}^{\top}\right). \tag{15}$$

Thus, every displacement can be decomposed as

$$\Delta W = P_{\mathrm{True}}\Delta W + \left(I - P_{\mathrm{True}}\right)\Delta W, \tag{16}$$

where the first term lies along the preferred output-side direction and the second term contains the component orthogonal to it.

These two components are orthogonal under the Frobenius inner product. Consequently,

$$\|\Delta W\|_F^2 = \|P_{\mathrm{True}}\Delta W\|_F^2 + \|(I - P_{\mathrm{True}})\Delta W\|_F^2. \tag{17}$$

Effect of the regularizer. The regularization term in Equation 3 is

$$\mathcal{R}(\Delta W) = \frac{1}{2} \|(I - P_{\text{True}})\Delta W\|_F^2 . \tag{18}$$

It therefore measures exactly the squared magnitude of the parameter displacement lying outside $\text{span}(u_{\text{True}})$. Minimizing this term suppresses the component of the update in the orthogonal complement while leaving the component along u_{True} unpenalized.

Equivalently, using $P_{\text{True}} = u_{\text{True}} u_{\text{True}}^\top$,

$$\|(I - P_{\text{True}})\Delta W\|_F^2 = \|\Delta W\|_F^2 - \left\|u_{\text{True}}^\top \Delta W\right\|_2^2 . \tag{19}$$

Hence, for a displacement of fixed Frobenius norm, reducing the regularization penalty is equivalent to placing more of its magnitude in the output-side subspace spanned by u_{True}.

If the regularization term were minimized in isolation, its minimum would be achieved whenever

$$(I - P_{\text{True}})\Delta W = 0, \tag{20}$$

or equivalently,

$$\Delta W = P_{\text{True}} \Delta W = u_{\text{True}} a^\top \tag{21}$$

for some vector a. In the full objective, however, the regularizer is combined with the weak-reward loss. It therefore acts as a soft constraint rather than forcing the update to lie exactly in the rank-one subspace. The resulting optimization penalizes movement away from the True-reward output-side direction while allowing the weak-reward objective to determine the magnitude and input-side structure of the update.

D FULL u_1 AND v_1 ABSOLUTE COSINE SIMILARITY HEATMAPS ACROSS ALL CHECKPOINTS

The full cosine similarity heatmaps checked every 50 steps can be seen within Figures 4, 5, 6, 7. The similarity values degrade monotonically for the output-side in most cases, but the input-side similarity oddly shows an increase in some cases.

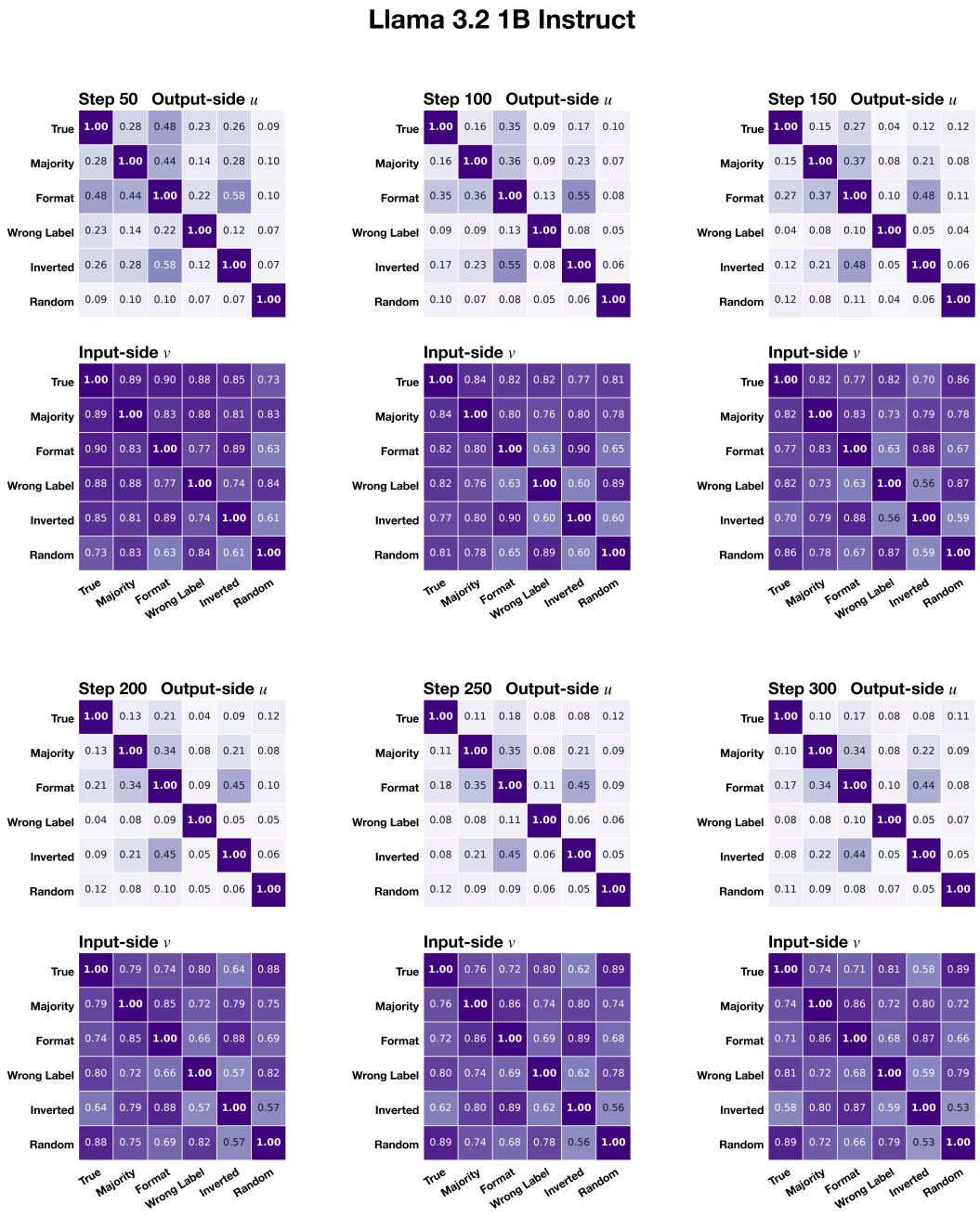


Figure 4: Cosine similarity heatmap across all checkpoints for Llama-3.2-1B-Instruct model.

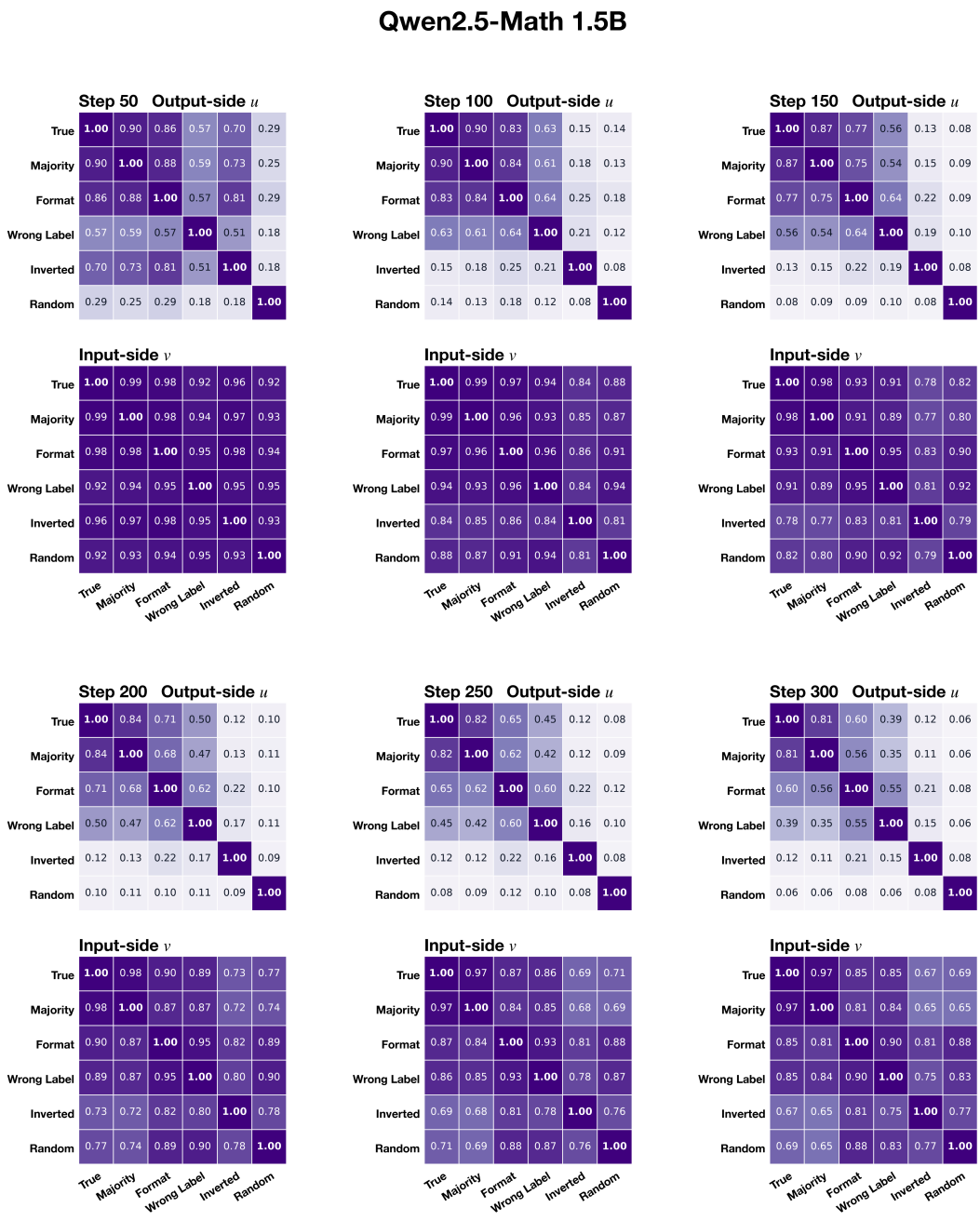


Figure 5: Cosine similarity heatmap across all checkpoints for Qwen-2.5-1.5B-Math model.

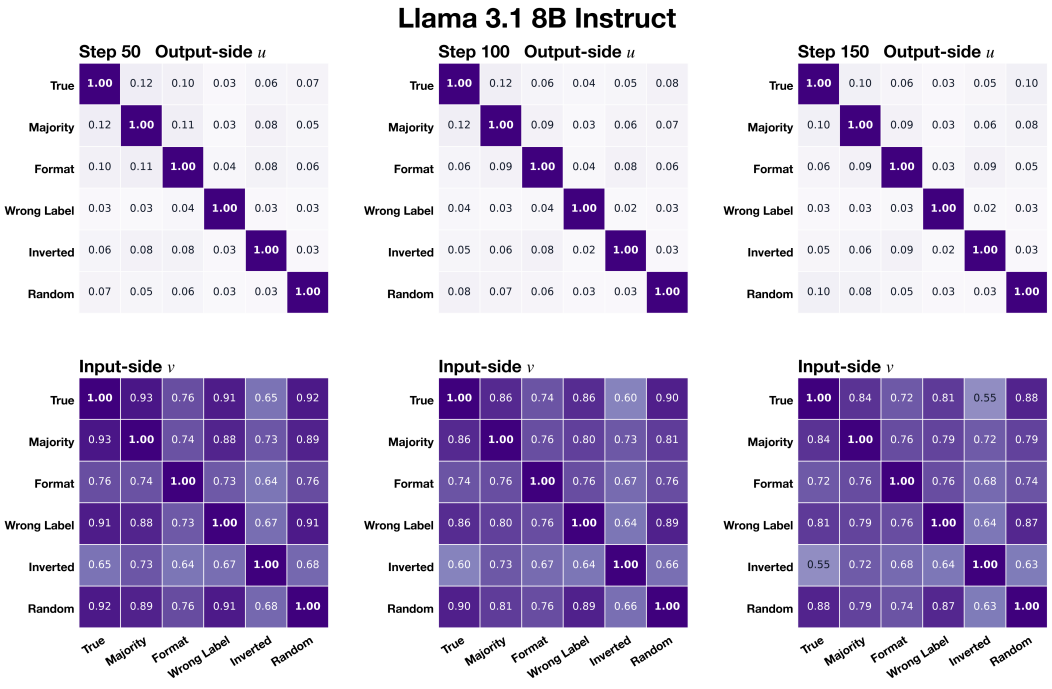


Figure 6: Cosine similarity heatmap across all checkpoints for Llama-3.1-8B-Instruct model.

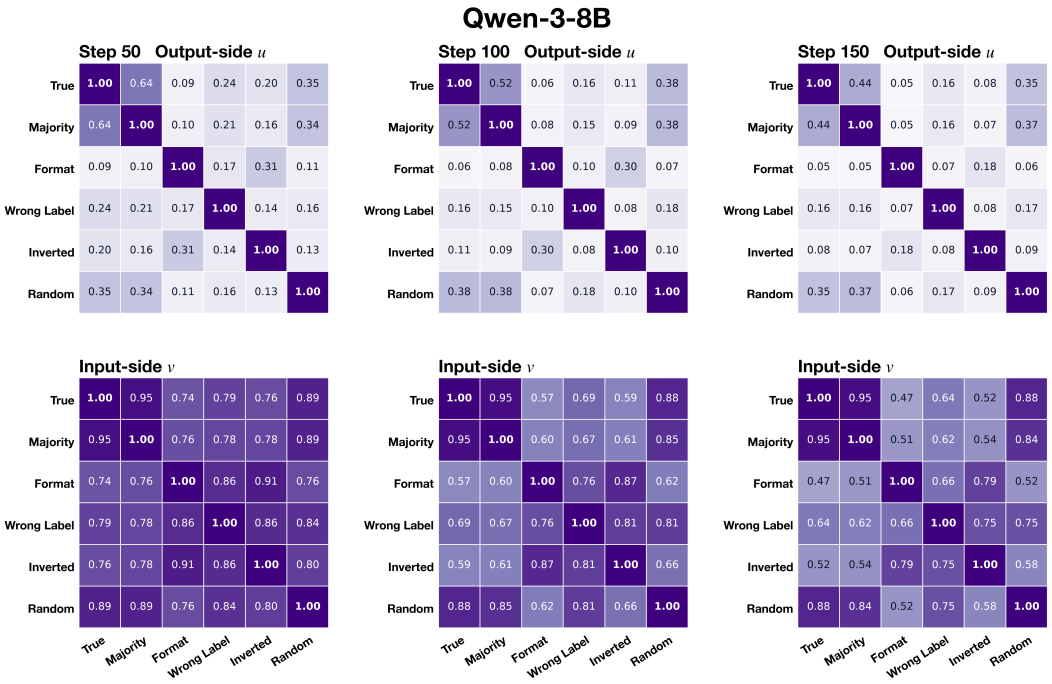


Figure 7: Cosine similarity heatmap across all checkpoints for Qwen-3-8B model.

E EXAMPLE QUALITATIVE RESULTS

As written within Section 3.2, we show additional examples of outputs where True and Majority rewards nearly produce the same outputs, while Format and Inverted produce wrong outputs, but maintain their `\boxed{}` final answers.

We show an example from AIME24 at Figure 8, an example from AMC23 at Figure 9 and an example from OlympiadBench at Figure 10.

AIME24 QUESTION — **Gold:** 699

Let N be the greatest four-digit positive integer with the property that whenever one of its digits is changed to 1, the resulting number is divisible by 7. Let Q and R be the quotient and remainder, respectively, when N is divided by 1000. Find $Q + R$.

Qwen2.5-Math-1.5B – True Reward

...

Let's write a Python function to check this condition.

```python
def is_valid_number(N):
    str_N = str(N)
    for i in range(4):
        new_number = int(str_N[:i] + '1' +
          str_N[i+1:])
        if new_number % 7 != 0:
            return False
    return True

# Find the greatest four-digit number that
  satisfies the condition
for N in range(9999, 999, -1):
    if is_valid_number(N):
        break
...
print((N, Q, R, result))
```

```
(5694, 5, 694, 699)
```

...

The final answer is $\boxed{699}$.

Qwen2.5-Math-1.5B – Majority@16 Reward

...

Let's write a Python function to check this condition.

```python
def is_valid_number(N):
    str_N = str(N)
    for i in range(4):
        # Create a new number by changing
          the i-th digit to 1
        new_number = int(str_N[:i] + '1' +
          str_N[i+1:])
        if new_number % 7 != 0:
            return False
    return True

# Find the greatest four-digit number that
  satisfies the condition
for N in range(9999, 999, -1):
    if is_valid_number(N):
        break
...
print((N, Q, R, result))
```

```
(5694, 5, 694, 699)
```

...

So, the final answer is $\boxed{699}$.

Qwen2.5-Math-1.5B – Format Reward

...

Let's write a Python code to find this number.

```python
def is_divisible_by_7(n):
    return n % 7 == 0

def find_greatest_N():
    for N in range(9999, 999, -1):
        digits = [int(d) for d in str(N)]
        if (is_divisible_by_7(1000 + digits[1]) and
            is_divisible_by_7(1000 * digits[0] + 100 + digits[2]) and
            is_divisible_by_7(1000 * digits[0] + 100 * digits[1] + digits[3]) and
            is_divisible_by_7(1000 * digits[0] + 100 * digits[1] + 10 * digits[2] + 1)):
            return N
    return None

N = find_greatest_N()
print((N, N // 1000, N % 1000, N // 1000 + N % 1000))
```

```
(8151, 8, 151, 159)
```

So, the final answer is $\boxed{159}$.

Qwen2.5-Math-1.5B – Inverted Reward

```
\boxed{}
```

Figure 8: A qualitative output example from the Qwen2.5-1.5B-Math model on the AIME24 dataset.

AMC23 QUESTION **Gold:** 36

Positive real numbers x and y satisfy $y^3 = x^2$ and $(y - x)^2 = 4y^2$. What is $x + y$?

Qwen2.5-Math-1.5B – True Reward

Let's start by solving the given equations step by step.

...

This is a quadratic equation in terms of x. We can factorize it:

$$x^2 - 2xy - 3y^2 = (x - 3y)(x + y) = 0$$

...

Now we substitute $x = 3y$ into the first equation $y^3 = x^2$:

$$y^3 = (3y)^2$$

$$y^3 = 9y^2$$

Since $y \neq 0$, we can divide both sides by y^2:

$$y = 9$$

...

So, $x = 27$ and $y = 9$. Therefore, $x + y = 27 + 9 = 36$.
The final answer is: $\boxed{36}$

Qwen2.5-Math-1.5B – Majority@16 Reward

Let's solve the given problem step by step.

...

This is a quadratic equation in terms of x. We can factorize it:

$$x^2 - 2xy - 3y^2 = (x - 3y)(x + y) = 0$$

...

Now, we substitute $x = 3y$ into the first equation $y^3 = x^2$:

$$y^3 = (3y)^2$$

$$y^3 = 9y^2$$

Since $y \neq 0$, we can divide both sides by y^2:

$$y = 9$$

...

Therefore, $x + y = 27 + 9 = 36$.
The final answer is: $\boxed{36}$

Qwen2.5-Math-1.5B – Format Reward

...

Let's implement this in Python using sympy.

```python
import sympy as sp

# Define the variables
x, y = sp.symbols('x y', real=True, positive=True)

# Define the equations
eq1 = sp.Eq(y**3, x**2)
eq2 = sp.Eq((y - x)**2, 4*y**2)

# Solve the first equation for x
x_expr = sp.solve(eq1, x)[0]

# Substitute x in the second equation
eq2_substituted = eq2.subs(x, x_expr)

# Solve the resulting equation for y
y_value = sp.solve(eq2_substituted, y)[0]

# Find the corresponding x value
x_value = x_expr.subs(y, y_value)

# Calculate x + y
result = x_value + y_value
print(result)
```

```
16
```

The value of $x + y$ that satisfies the equations $y^3 = x^2$ and $(y - x)^2 = 4y^2$ is $\boxed{16}$.

Qwen2.5-Math-1.5B – Inverted Reward

```
\boxed{}
```

Figure 9: A qualitative output example from the Qwen2.5-1.5B-Math model on the AMC23 dataset.

OLYMPIADBENCH QUESTION **Gold:** 29

Hexagon $ABCDEF$ has vertices $A(0,0), B(4,0), C(7,2), D(7,5), E(3,5), F(0,3)$. What is the area of hexagon $ABCDEF$?

Qwen2.5-Math-1.5B – True Reward

To find the area of the hexagon with vertices $A(0,0)$, $B(4,0)$, $C(7,2)$, $D(7,5)$, $E(3,5)$, and $F(0,3)$, we can use the Shoelace formula (also known as Gauss's area formula).

...

$$\text{Area} = \frac{1}{2}|87 - 29|$$

$$\text{Area} = \frac{1}{2}|58|$$

...

So the area of the hexagon $ABCDEF$ is $\boxed{29}$.

Qwen2.5-Math-1.5B – Majority@16 Reward

To find the area of the hexagon with vertices $A(0,0)$, $B(4,0)$, $C(7,2)$, $D(7,5)$, $E(3,5)$, and $F(0,3)$, we can use the Shoelace formula (also known as Gauss's area formula).

...

$$\text{Area} = \frac{1}{2}|87 - 29|$$

$$\text{Area} = \frac{1}{2}|58|$$

...

So the area of the hexagon is $\boxed{29}$.

Qwen2.5-Math-1.5B – Format Reward

...

We can now calculate the area using the Shoelace formula in Python.

```python
# Coordinates of the vertices of the hexagon
vertices = [(0, 0), (4, 0), (7, 2), (7, 5), (3, 5), (0, 3), (0, 0)]

# Using the Shoelace formula to calculate the area
def shoelace_formula(vertices):
    n = len(vertices)
    area = 0
    for i in range(n-1):
        x1, y1 = vertices[i]
        x2, y2 = vertices[i+1]
        area += x1 * y2 - y1 * x2
    return abs(area) / 2

# Calculate the area of the hexagon
area_hexagon = shoelace_formula(vertices)
print(area_hexagon)
```

```
19.0
```

The area of the hexagon $ABCDEF$ with vertices $A(0,0)$, $B(4,0)$, $C(7,2)$, $D(7,5)$, $E(3,5)$, and $F(0,3)$ is $\boxed{19}$.

Qwen2.5-Math-1.5B – Inverted Reward

```
\boxed{}
```

Figure 10: A qualitative output example from the Qwen2.5-1.5B-Math model on the Olympiad Bench dataset.