SPARC: CALIBRATING SELF-PREFERENCE ACROSS REASONING PROGRESS IN PROCESS REWARD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Process reward models (PRMs) are increasingly used to evaluate intermediate reasoning steps and guide inference-time computation in large language models. Yet it remains unclear whether PRMs score equally valid reasoning differently depending on its source, particularly whether they favor trajectories generated by their own base models. Our analysis reveals two key properties of self-preference in PRMs. First, even among verified-correct solutions to the same problem, PRMs often assign higher scores to trajectories generated by their own base models. Second, self-preference is already present at intermediate reasoning steps and often becomes stronger over the intermediate stages of reasoning. Motivated by these findings, we introduce Self Preference Aware Reward Calibration (SPARC), a process-aware method that calibrates source-dependent score differences across reasoning progress while preserving the PRM's ability to distinguish correct from incorrect solutions. Across multiple PRMs and mathematical reasoning benchmarks, SPARC reduces source-dependent score deviations from parity by 72.8% on average across reasoning progress while preserving correct–incorrect discrimination. Code is available at `https://anonymous.4open.science/r/SPARC-8205`.

1 INTRODUCTION

Process reward models (PRMs) (Lightman et al., 2024) have become increasingly important for evaluating intermediate reasoning (Wang et al., 2024; Li & Li, 2025) and guiding inference-time computation, including reranking, candidate selection, and test-time scaling (Chen et al., 2024; Snell et al., 2025; Setlur et al., 2025). In these settings, PRM scores serve as signals for comparing the quality of different reasoning trajectories (Yuan et al., 2025). Therefore, PRM scores should ideally provide a faithful measure of reasoning quality, independent of which model generated the trajectory. Yet it remains unclear to what extent the identity of the generating model affects PRM scores.

Prior work on LLM-based evaluation has shown that LLMs can exhibit self-preference, favoring their own generations when evaluating model outputs (Panickssery et al., 2024; Chen et al., 2025b; Zheng et al., 2023). PRMs are typically derived from pretrained LLMs through additional reward-model training (Lightman et al., 2024; Wang et al., 2024). However, whether such self-preference persists after reward-model training remains unclear, particularly whether PRMs favor reasoning trajectories generated by their own base models over those generated by other models.

To investigate this issue, we compare how PRMs score trajectories from their own base models and from other models across multiple mathematical reasoning benchmarks. To isolate source preference from differences in answer correctness, we focus on verified-correct trajectories for the same problems. Our analysis reveals two findings. First, PRMs exhibit systematic own-base preference: across PRMs and benchmarks, trajectories generated by a PRM's own base model are often assigned higher scores than those generated by other models. Second, this preference is not confined to the final trajectory score. It is already present at intermediate reasoning steps and often becomes stronger over the intermediate stages of reasoning, although its evolution varies across PRMs and benchmarks. Figure 1 provides an illustration of this phenomenon, including cases where an incorrect own-base trajectory receives higher scores than a correct trajectory from another model.

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

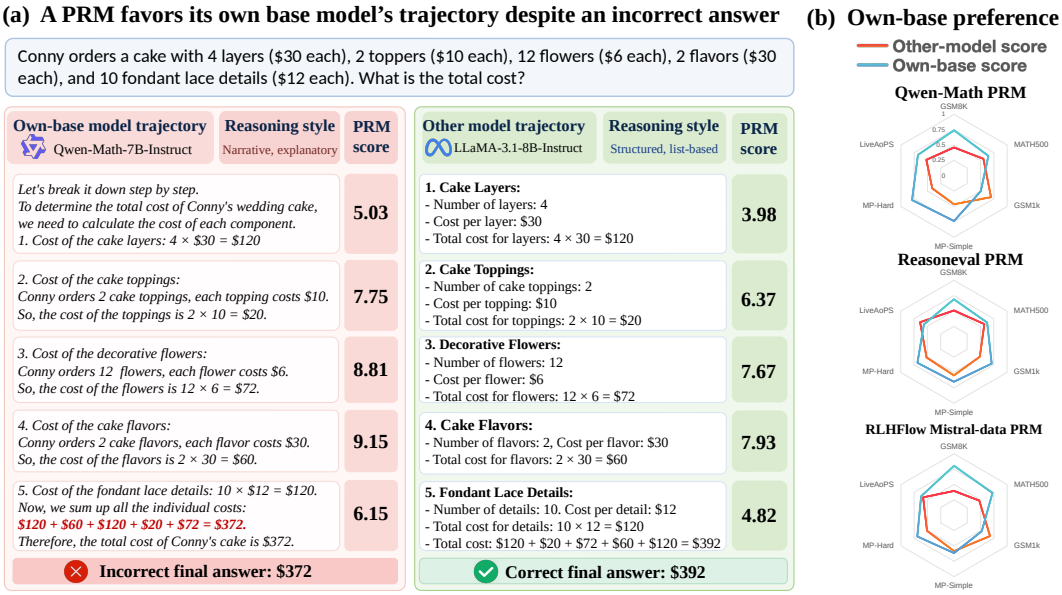


Figure 1: **PRMs exhibit own-base preference. (a)** A GSM1k example where Qwen-Math assigns higher scores to its own incorrect trajectory than to a correct LLaMA-3.1 trajectory. **(b)** Radar plots compare incorrect own-base and correct other-model scores across six benchmarks.

Motivated by the presence of own-base preference throughout the reasoning process and its tendency to strengthen as reasoning progresses, we propose **Self Preference Aware Reward Calibration (SPARC)**, a process-aware method for mitigating own-base preference in PRMs. SPARC reduces source-dependent score gaps at matched reasoning progress and constrains their changes across adjacent reasoning steps. Crucially, reducing source-dependent score differences without explicitly protecting correctness discrimination can compromise the PRM's ability to distinguish correct from incorrect reasoning. We therefore introduce a terminal correctness protection loss to preserve this capability during calibration. SPARC optimizes these losses through a shared LoRA adapter while keeping the original PRM backbone and reward head frozen.

Experiments across multiple PRMs and mathematical reasoning benchmarks demonstrate that SPARC brings scores for matched own-base and other-model trajectories substantially closer to parity while generally preserving the PRM's ability to distinguish correct from incorrect reasoning. We further find that this calibration can improve PRM performance on established process-evaluation and candidate-selection benchmarks. Our main contributions are as follows:

- We identify systematic own-base preference in PRMs and show, through progress-resolved analysis, that this preference is already present at intermediate reasoning steps and often strengthens as reasoning progresses.
- We propose **Self Preference Aware Reward Calibration (SPARC)**, a process-aware method that calibrates own-base preference at matched progress points and across adjacent reasoning steps while preserving correct–incorrect discrimination.
- We show that self-preference calibration can improve practical PRM performance on established process-evaluation and candidate-selection benchmarks, while characterizing the trade-offs across different PRMs.

2 Self-Preference in Process Reward Models

2.1 Self-Preference Across PRMs

We first test whether PRMs assign systematically different scores to trajectories depending on their source model. To control for answer correctness, we compare verified-correct solutions to the same problem: one generated by the PRM's own base model and one by another model, following the paired score-difference approach of Bharadwaj et al. (2026). We summarize the comparison by

GAP, the exponentiated mean score gap with equal problem weight. For log-odds scorers, GAP is a weighted geometric mean odds ratio. A value of $1\times$ denotes no average source preference; values above $1\times$ favor own-base trajectories and values below $1\times$ favor other-model trajectories. Because PRMs use different score parameterizations, we interpret GAP relative to $1\times$ within each PRM. Full definitions and aggregation details are in Appendix B.

Across the evaluated PRMs and benchmarks, own-base preference is systematic. As shown in Figure 2, nearly all evaluated PRM–benchmark pairs have $\mathrm{GAP} > 1\times$, and each PRM favors trajectories from its own base model on most benchmarks. Thus, source-dependent score differences persist even when the compared trajectories solve the same problem correctly. The example in Figure 1 illustrates why such source dependence can matter in practice: an incorrect own-base trajectory can receive a higher score than a correct trajectory from another model. The diagnostics also include PURE (Cheng et al., 2025), ReasonFlux (Zou et al., 2025), and ReasonEval (Xia et al., 2025); Appendix C.1 lists all PRMs and their base models.

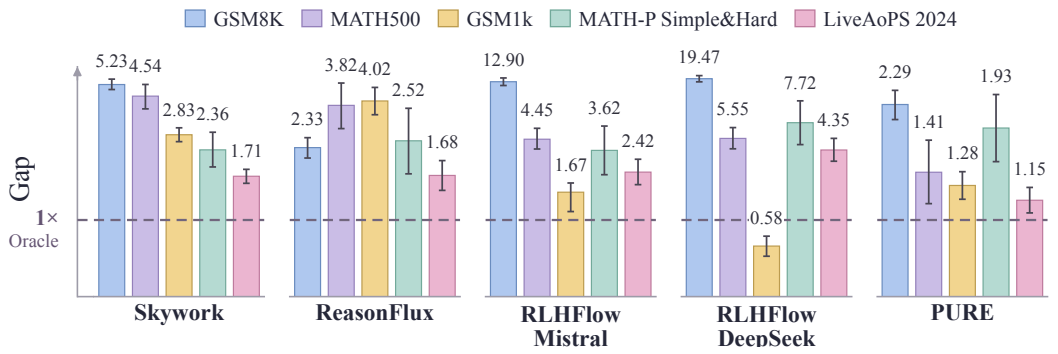


Figure 2: Self-preference on complete trajectories across benchmarks. Bars show GAP and whiskers show pointwise 95% problem-bootstrap intervals. The dashed line marks $1\times$, corresponding to no average source preference. Nearly all evaluated PRM–benchmark pairs lie above $1\times$.

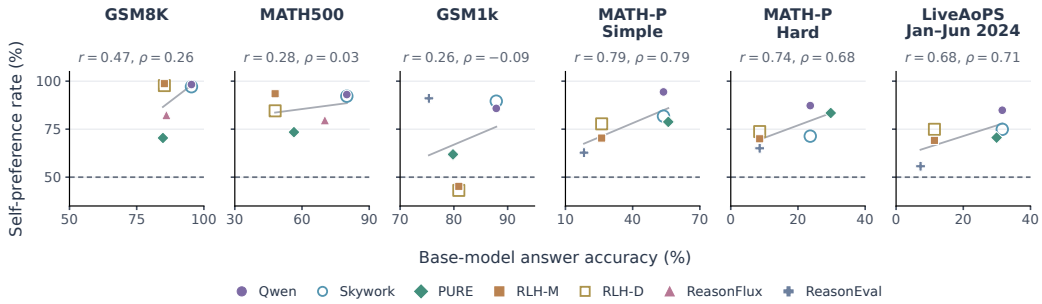


Figure 3: Relationship between base-model answer accuracy and self-preference when trajectories are scored regardless of final correctness. Each panel corresponds to one dataset; colour and fill identify the PRM. The dashed line marks 50% self-preference, and grey lines show least-squares fits, with r and ρ denoting Pearson and Spearman correlations. Positive associations appear on several datasets but vary substantially in strength. Scoring protocols are in Appendix D.2.

Conditioning on correctness is important because otherwise source preference can be entangled with differences in base-model capability. Prior work on LLM evaluators similarly finds that apparent self-preference can partly reflect differences in response quality (Chen et al., 2025a;b). We therefore repeat the analysis without conditioning on final correctness. As shown in Figure 3, the resulting self-preference rate is positively associated with base-model answer accuracy on several datasets, while the association is weak on others. Unfiltered comparisons can therefore mix source-dependent scoring with differences in the quality of the generated solutions.

For the remainder of our analysis, we therefore use matched correct–correct comparisons to separate source preference from answer correctness. Having established own-base preference on complete trajectories, we next ask whether it is already present before the reasoning process is complete and how it evolves with reasoning progress.

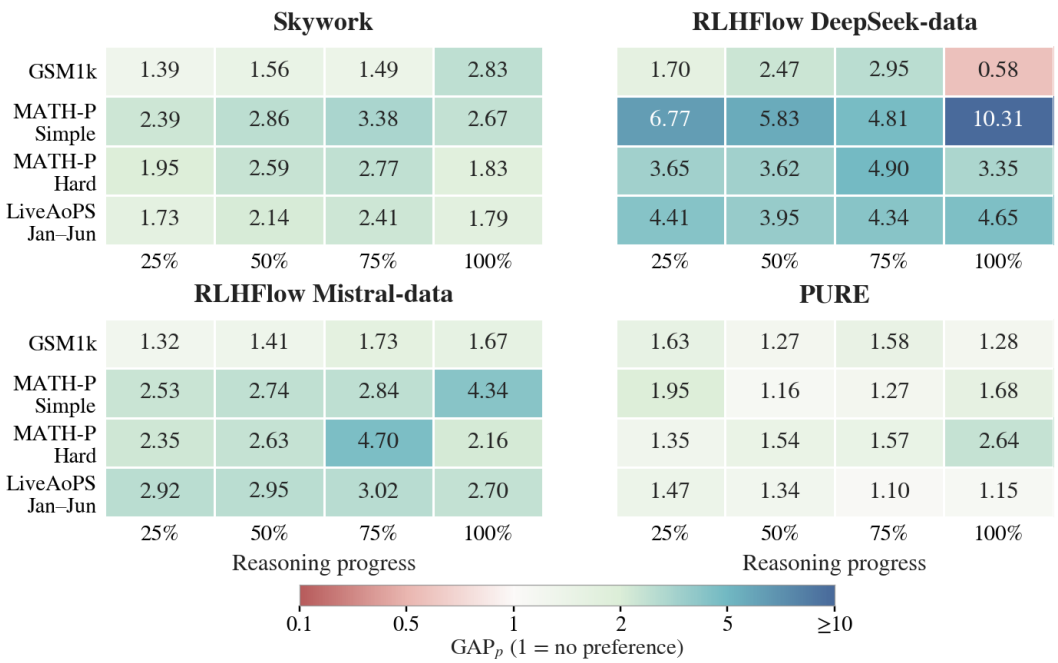


Figure 4: Self-preference across reasoning progress. Cells report GAP_p at 25/50/75/100% progress; $1\times$ denotes no source preference, with larger values favoring own-base trajectories and smaller values favoring other-model trajectories. Self-preference is often visible early and strengthens over intermediate reasoning progress, with variation across PRMs and benchmarks.

2.2 Self-Preference Across Reasoning Progress

A terminal-level gap does not reveal when own-base preference emerges. We therefore score verified-correct trajectories at $p \in \{0.25, 0.50, 0.75, 1\}$, using the prefix ending at step $\lceil pN \rceil$ for a trajectory of N natural steps. At each progress point, we apply the same paired correct–correct comparison as in Section 2.1 and report GAP_p. Final-answer correctness does not necessarily imply that every intermediate prefix is correct; further details are in Appendices C.2 and C.3.

Figure 4 reveals two main patterns. First, own-base preference is already visible before completion: for most PRM–benchmark pairs, PRMs already favor their own base models at 25% reasoning progress. Second, the preference often strengthens over intermediate reasoning progress, although its trajectory varies across PRMs and benchmarks and is not always monotonic. These results motivate process-aware calibration that addresses source-dependent score differences throughout reasoning rather than only at the terminal step.

3 SPARC: Self Preference Aware Reward Calibration

The preceding analysis shows that self-preference is already present at intermediate reasoning steps and often becomes stronger as reasoning progresses. Accordingly, source-dependent score differences should be calibrated throughout reasoning process rather than only at terminal step. At the same time, the calibration should not degrade PRM's correct–incorrect discrimination performance. SPARC addresses these two objectives jointly. It calibrates source-dependent score differences across reasoning progress (Section 3.1), protects terminal correct–incorrect discrimination (Section 3.2), and regularizes the adaptation to limit excessive changes to the original PRM (Section 3.3).

3.1 Preference Calibration

We keep the original PRM and reward head frozen and train a shared LoRA adapter in the query and value projections (Hu et al., 2022). Let s_{θ_0} denote the original scorer and $s_\phi(x, y) = s_{\theta_0 + \Delta\theta(\phi)}(x, y)$ the adapted scorer.

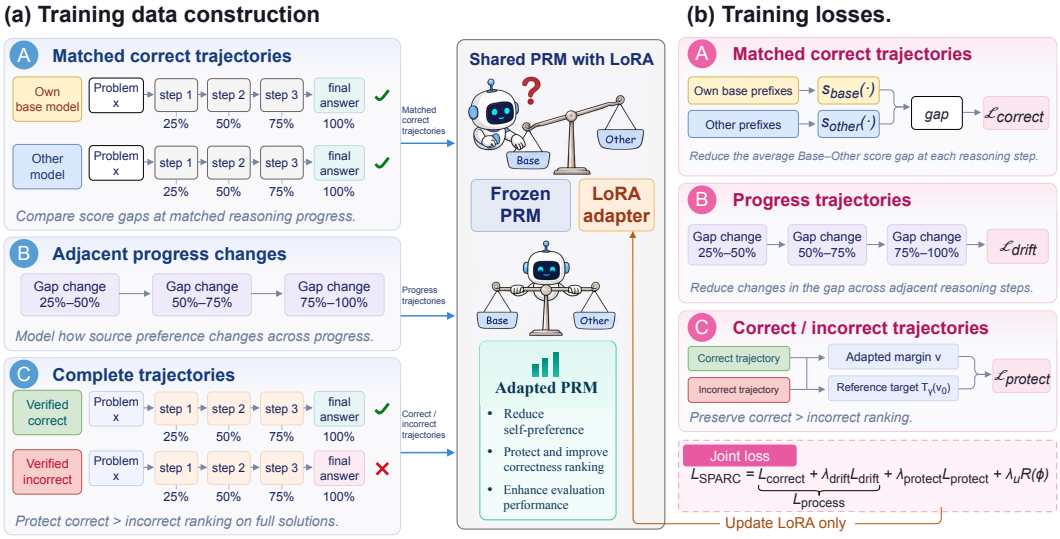


Figure 5: Overview of SPARC. **(a)** We construct comparison pairs from both correct–correct and correct–incorrect trajectories, and cut the trajectories into natural prefixes at $25/50/75/100\%$ progress, which makes training more efficient. **(b)** The loss is designed to calibrate the self-preference at each reasoning step while preserving the PRM's ability to tell correct solutions from incorrect ones. SPARC updates a shared LoRA adapter, while the PRM backbone and reward head remain frozen.

Correct-pair loss. We start with the score difference between two correct solutions to the same problem at the same reasoning step, one from the PRM's own base model and one from another model. Groups $c \in \mathcal{C}$ collect the pairs that share a benchmark, an other model, and a reasoning step. The correct-pair loss penalizes the mean gap of each group rather than the individual gaps:

$$L_{\text{correct}} = \sum_{c \in \mathcal{C}} \omega_c \big(\mathbb{E}_c[\text{gap}_p]\big)^2, \qquad \sum_{c \in \mathcal{C}} \omega_c = 1, \tag{1}$$

where $\text{gap}_p = s_\phi(x, y^p_{\text{base}}) - s_\phi(x, y^p_{\text{other}})$ is the paired difference under the adapted scorer, $\mathbb{E}_c$ averages over the pairs in c with equal problem weight, and ω_c weights the groups. Penalizing the group mean targets systematic source dependence without requiring two distinct correct solutions to receive identical scores.

Adjacent-step loss. Because source dependence can vary over reasoning progress, we additionally regularize its change between adjacent progress points. For each adjacent pair of reasoning steps $(0.25, 0.50)$, $(0.50, 0.75)$ and $(0.75, 1)$, the adjacent-step loss penalizes the mean shift in the gap:

$$L_{\text{drift}} = \sum_{h \in \mathcal{H}} \omega_h \Big(\mathbb{E}_h\big[\text{gap}_{p_{k+1}} - \text{gap}_{p_k}\big]\Big)^2, \qquad \sum_{h \in \mathcal{H}} \omega_h = 1, \tag{2}$$

where a drift group $h \in \mathcal{H}$ collects the pairs that share a benchmark, an other model, and an interval, ω_h follows the same hierarchy as ω_c with intervals in place of reasoning steps, and $\mathbb{E}_h$ averages over matched pairs whose two trajectories have distinct observed steps at both endpoints. The two losses constrain different aspects of one preference. The correct-pair loss penalizes the gap at each reasoning step on its own, while the adjacent-step loss constrains its change between consecutive steps. Combining both losses gives

$$L_{\text{process}} = L_{\text{correct}} + \lambda_{\text{drift}} L_{\text{drift}}. \tag{3}$$

Both losses square a group mean, so a single minibatch mean would also penalize the variance among the pairs inside it; we therefore estimate each square from two independent minibatches, see Appendix E.3 for the derivation.

3.2 Terminal Correctness Protection

Despite reducing the gap at every step, the two losses above compare correct solutions only and do not constrain the PRM's ability to distinguish between correct and incorrect solutions. To preserve

that ability, we follow the ranking view of PRMs (Li & Li, 2025) and constrain three terminal comparison conditions, where y^+ and y^- denote trajectories with a correct and an incorrect final answer, respectively:

$$A : y^+_{\text{base}} \succ y^-_{\text{other}}, \qquad B : y^+_{\text{other}} \succ y^-_{\text{base}}, \qquad G : y^+ \succ y^-. \tag{4}$$

Terminal performance is the accuracy of the PRM on these three comparisons. The global condition G places no restriction on the source of either trajectory. Conditions A and B differ exactly in which model contributes the correct solution, and requiring both prevents the shortcut of lowering base-model scores uniformly, which would satisfy B while breaking A. Together, the three conditions ensure that the calibration lowers self-preference while preserving the PRM's accuracy on these comparisons.

An intuitive way to enforce the three conditions is to encourage each terminal pair to attain a fixed positive margin ρ. However, this fixed target does not preserve the larger margins already learned by the original PRM: calibration can shrink them while they remain above ρ. We therefore make the target depend on the original margin $\nu_0 = s_{\theta_0}(x, y^+) - s_{\theta_0}(x, y^-)$, raising weak or incorrectly ordered pairs to ρ while leaving margins above a cap γ free to shrink:

$$T_\gamma(\nu_0) = \rho + \text{sp}_t(\nu_0 - \rho) - \text{sp}_t(\nu_0 - \gamma), \tag{5}$$

where $\text{sp}_t(z) = t \log(1 + \exp(z/t))$ is a smooth softplus and $0 < \rho < \gamma$. As $t \to 0$ the target approaches $\min\{\gamma, \max\{\rho, \nu_0\}\}$, the capped margin itself. We keep t finite so that the target stays smooth and the loss can be optimized directly. With $\nu = s_\phi(x, y^+) - s_\phi(x, y^-)$ the adapted margin, the terminal protection loss penalizes shortfalls from this target:

$$L_{\text{protect}} = \frac{1}{3} \sum_{K \in \{A,B,G\}} \mathbb{E}_K \left[\text{sp}_t(T_\gamma(\nu_0) - \nu)^2 \right], \tag{6}$$

where $\mathbb{E}_K$ averages within condition K over benchmark families, benchmarks, and problems in that order, with one terminal pair per problem. The penalty grows roughly quadratically with a large violation and becomes small once a pair is above its target, so the calibration pressure concentrates on the pairs whose correct ordering is weak. We measure the ranking that L_{protect} protects by the accuracy of the three conditions,

$$\text{Acc} = (\text{Acc}_A + \text{Acc}_B + \text{Acc}_G)/3, \tag{7}$$

which isolates terminal discrimination from the source preference measured by GAP.

3.3 The Overall Objective

Our loss combines preference calibration, the terminal protection, and the adapter regularizer:

$$L(\phi) = L_{\text{process}} + \lambda_{\text{protect}} L_{\text{protect}} + \lambda_u \mathcal{R}(\phi), \tag{8}$$

where $\mathcal{R}(\phi) = \sum_\ell \|\Delta W_\ell(\phi)\|_F^2$ is the squared Frobenius norm of the effective adapter update in each adapted projection ℓ. $\mathcal{R}(\phi)$ is computed on the effective update rather than on its two low-rank factors, so the penalty measures the distance between the adapted scorer and the frozen original. The details of SPARC are reported in Appendix E. We ablate the calibration and protection terms in Section 4.4. L_{process} and L_{protect} update the same adapter, so one trained model reduces the self-preference at each reasoning step while preserving the PRM's terminal performance.

4 Experiments

4.1 Experimental Setup

Datasets and evaluation. We measure self-preference on GSM8K (Cobbe et al., 2021), MATH500 (Hendrycks et al., 2021; Lightman et al., 2024), GSM1k (Zhang et al., 2024), MATH-P-Simple and MATH-P-Hard (Huang et al., 2025a), and LiveAoPS (Mahdavi et al., 2025). For calibration experiments, we use model-generated trajectories from GSM1k, both MATH-P variants, and LiveAoPS, with final-answer correctness determined by benchmark-specific checkers. GSM1k and MATH-P use an approximately 70/10/20 problem-level train/development/test split. LiveAoPS

Table 1: Main results on GSM1k, pooled MATH-P-Simple&Hard, and LiveAoPS across the Qwen, LLaMA, and Skywork PRMs. We report terminal GAP, group RMS, and Acc; GAP closer to $1\times$, lower RMS, and higher Acc are better. Progress-resolved GAP is shown in Figure 6. Bold and underline denote the best and second-best values within each PRM block.

		GSM1k			MATH-P Simple&Hard			LiveAoPS Sep.–Dec.		
PRM	Method	GAP	RMS	Acc (%)	GAP	RMS	Acc (%)	GAP	RMS	Acc (%)
Qwen	Original PRM	8.69	2.668	77.4	327.12	6.474	71.3	201.52	5.540	68.8
	BT-LoRA	6.94	2.371	78.6	359.60	6.602	<u>72.2</u>	225.98	5.648	70.5
	PPL-Cal	<u>1.99</u>	1.692	**80.1**	93.75	5.223	62.1	38.70	3.905	72.2
	SAFESCORE-DIR	2.64	<u>1.175</u>	75.8	<u>18.78</u>	<u>3.361</u>	<u>72.2</u>	<u>14.20</u>	<u>2.764</u>	<u>72.5</u>
	Full SPARC	**1.02**	**0.128**	<u>79.7</u>	**2.60**	**1.289**	**81.5**	**2.18**	**0.858**	**77.1**
LLaMA	Original PRM	<u>1.44</u>	<u>2.215</u>	67.6	6.06	2.309	65.1	2.19	1.342	63.9
	BT-LoRA	1.82	2.996	**84.1**	<u>1.60</u>	1.811	**81.4**	1.28	1.380	<u>73.1</u>
	PPL-Cal	1.73	2.337	71.3	5.79	2.277	63.9	3.04	1.612	65.0
	SAFESCORE-DIR	1.82	2.590	80.9	1.71	<u>1.570</u>	79.0	<u>1.19</u>	<u>1.203</u>	71.5
	Full SPARC	**1.25**	**1.999**	<u>83.7</u>	**1.13**	**1.028**	<u>79.4</u>	**1.00**	**0.812**	**74.2**
Skywork	Original PRM	4.68	1.577	85.3	7.19	2.286	63.0	4.79	1.686	<u>83.3</u>
	BT-LoRA	5.67	1.791	81.3	23.18	3.678	68.5	16.05	2.986	82.8
	PPL-Cal	<u>1.33</u>	<u>0.376</u>	84.5	<u>2.12</u>	<u>1.048</u>	50.0	<u>0.91</u>	**0.448**	74.7
	SAFESCORE-DIR	0.59	0.570	<u>85.7</u>	**1.14**	1.201	68.5	**0.94**	0.716	82.4
	Full SPARC	**1.21**	**0.230**	**88.1**	2.48	**0.919**	**68.5**	1.86	<u>0.685</u>	**83.8**

uses January–June 2024 for training, July–August for development, and September–December for evaluation. We report GAP at 25/50/75/100% reasoning progress, terminal group RMS, and terminal ranking Acc: GAP closer to $1\times$ and lower RMS indicate smaller source gaps, while Acc measures correct–incorrect discrimination, macro-averaged over the three conditions in Equation 7. We additionally evaluate process scoring on ProcessBench (Zheng et al., 2025) and PRMBench (Song et al., 2025), and candidate selection on AceMath-RewardBench (Liu et al., 2025).

Models and baselines. We independently calibrate Qwen Math PRM (Qwen Team, 2025), the RLHFlow LLaMA-family PRM (RLHFlow Team, 2024) trained on Mistral data, and Skywork (He et al., 2024). We compare Full SPARC with Original PRM, BT-LoRA, PPL-Cal, and SAFESCORE-DIR. BT-LoRA trains on correct–incorrect terminal pairs using the pairwise Bradley–Terry objective (Zhang et al., 2025a), as used in AceMath (Liu et al., 2025). Inspired by the familiarity-based account of self-preference (Wataoka et al., 2025), PPL-Cal applies a perplexity-based score correction fitted on training data. SAFESCORE-DIR adapts DIR (Li et al., 2026b) to source invariance and adds terminal correctness protection to limit ranking degradation. Trainable methods use rank-8 LoRA adapters in the query and value projections, with the original PRM parameters and reward head frozen. Comparisons within each PRM use the same data partitions and evaluation pairs. We train on two NVIDIA A100 40GB GPUs with AdamW (Loshchilov & Hutter, 2019) at a learning rate of 5×10^{-5} for three epochs and average adapted-model results over seeds 17, 29, and 43. Optimization and baseline construction are in Appendices E and F.

4.2 SELF-PREFERENCE MITIGATION ACROSS PRMS

SPARC improves source calibration and correctness discrimination. Across nearly all of the nine PRM–benchmark blocks in Table 1, Full SPARC achieves the lowest terminal group RMS, indicating smaller source-dependent score differences across source groups. At the same time, it improves terminal Acc over the Original PRM in all nine blocks and attains the best or second-best Acc in every block. Thus, in the main evaluation, mitigating self-preference does not require sacrificing the PRM's ability to distinguish correct from incorrect reasoning; the two improve together.

The calibration extends throughout the reasoning process. Figure 6 shows that Full SPARC moves the score gap closer to the $1\times$ parity point than the Original PRM at all 36 evaluated PRM–benchmark–progress points, spanning 25%, 50%, 75%, and 100% reasoning progress. We measure

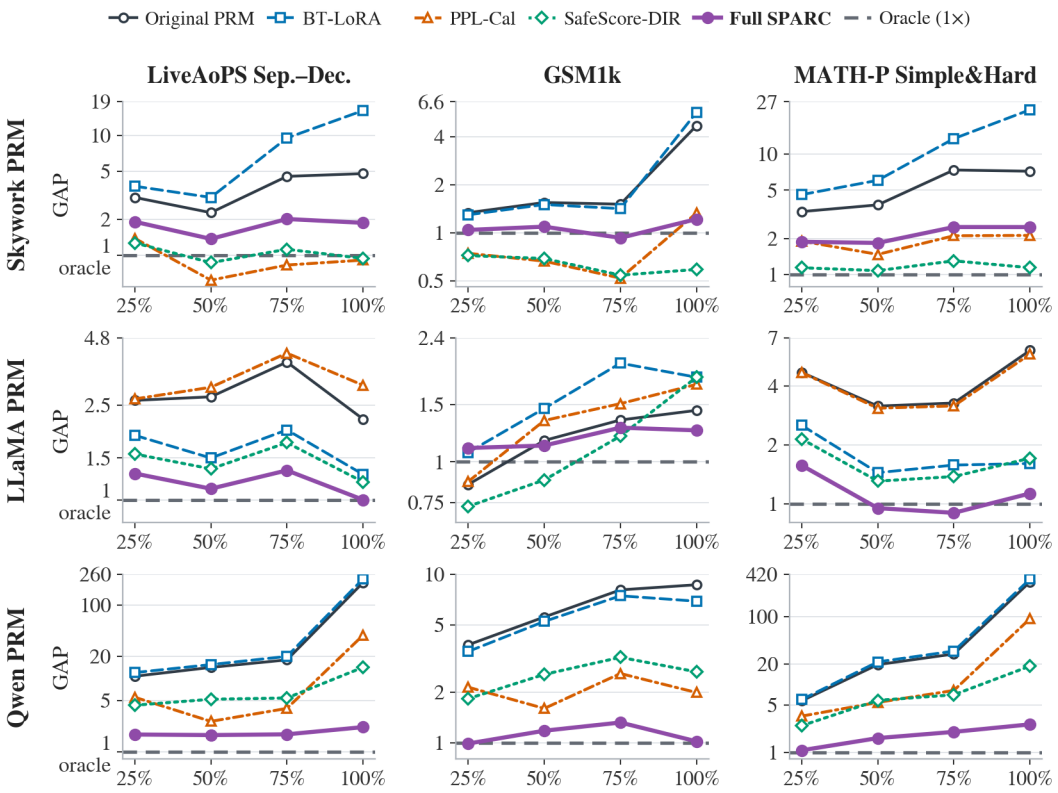


Figure 6: Progress-resolved self-preference for the comparisons. Each panel shows GAP at $25/50/75/100\%$ reasoning progress for one PRM–benchmark pair. Axes are logarithmic, and the grey dashed Oracle line at $1\times$ denotes no self-preference. Full SPARC moves GAP closer to $1\times$ than the Original PRM at all progress points. Terminal group RMS and Acc are reported in Table 1.

deviation from parity by $|\ln \mathrm{GAP}_p|$ and evaluate the reduction $100(1-\bar{d}_{\mathrm{Full}}/\bar{d}_{\mathrm{Original}})$ within each PRM, over four benchmarks (MATH-P-Simple and Hard counted separately) and four progress levels. Averaging the three PRM-level reductions gives 72.8%, against 84.1%, 80.7%, and 53.7% for Qwen, LLaMA, and Skywork (Appendix G). At the terminal step, Full SPARC is closest to parity in seven of the nine main-table blocks. The two exceptions are Skywork on MATH-P and LiveAoPS, where SAFESCORE-DIR attains a terminal GAP closer to $1\times$; however, Full SPARC achieves lower group RMS in both settings while matching or improving Acc. This distinction highlights why calibration should be assessed jointly through progress-resolved gaps, group-level source differences, and correctness discrimination rather than by a single terminal GAP.

4.3 SELF-PREFERENCE CALIBRATION CAN IMPROVE PRM PERFORMANCE

Reducing source-dependent score differences is useful only if the calibrated PRM remains effective at selecting reasoning trajectories. We therefore compare the Original PRM and SPARC on candidate-selection benchmarks, using identical evaluation examples and paired bootstrap intervals.

Table 2: Original PRM versus Full SPARC on ProcessBench and PRMBench. Each cell reports Original → Full SPARC. ProcessBench columns use F1, while PRMBench uses PRMScore; higher is better. Full SPARC averages three training seeds.

PRM	GSM8K	MATH	OlympiadBench	OmniMath	PRMBench
Qwen	81.77 → **82.26**	77.16 → **77.34**	65.95 → **66.91**	61.35 → **65.92**	**65.45** → 64.60
LLaMA	52.72 → **52.89**	34.41 → **35.26**	14.72 → **14.83**	16.61 → **16.90**	54.57 → **54.97**
Skywork	**67.93** → 66.96	48.56 → **51.22**	15.99 → **19.62**	**18.79** → 18.35	61.30 → **61.44**

Self-preference calibration can improve candidate selection. We evaluate candidate selection using the official eight-candidate sets provided by AceMath-RewardBench across five subsets. Under the benchmark's `rm@8` protocol, the PRM scores the eight candidate solutions for each problem and selects the highest-scoring one; selection accuracy measures whether the selected solution is correct. We use the same candidate sets for the Original PRM and Full SPARC, so differences in selection performance arise solely from changes in the scorer. Full SPARC improves selection accuracy in 12 of the 15 PRM–subset comparisons: Qwen improves on four of five subsets, LLaMA on all five, and Skywork on three of five. The remaining decreases show that the effect is not uniform across PRMs and subsets. Overall, these results show that reducing source-dependent score differences can directly improve how PRMs rank and select candidate solutions, while retaining some model-dependent trade-offs.

4.4 Ablation Studies

Ablation variants. Each variant removes or replaces one component of the SPARC objective, with all other settings identical to those of Full SPARC and the same test comparisons. *w/o step-aware calibration* restricts the correct-pair loss to complete trajectories and drops the drift loss. *w/o drift loss* sets $\lambda_{\text{drift}} = 0$ in Equation 3. *w/ mean squared gap* replaces $(\mathbb{E}_c[\text{gap}_p])^2$ with $\mathbb{E}_c[\text{gap}_p^2]$ in Equation 1. *w/o protect loss* removes L_{protect}. *w/o process loss* drops both calibration losses and retains only the terminal protection. *w/o cap* removes the upper bound on the reference-margin target of Equation 5 ($\gamma \to \infty$). Removing the process loss or the cap moves terminal GAP further from $1\times$ on average for every PRM, so both the calibration and the protection terms are necessary, and their design sets the trade-off between preference reduction and terminal accuracy. Figure 7 summarizes the controls; the progress-aware GAP for each variant appears in Appendix G.

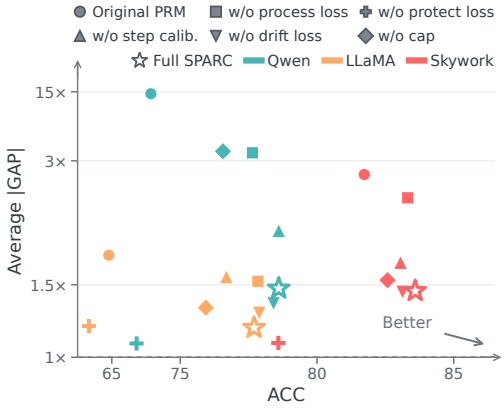


Figure 7: Component ablations across the three PRMs. The x-axis shows Acc and the y-axis mean GAP deviation from parity across reasoning progress; values below $1\times$ are mirrored, so lower right is best.

5 Related Work

PRM research studies step supervision (Lightman et al., 2024; Wang et al., 2024), process evaluation (Zheng et al., 2025; Song et al., 2025), and ranking (Li & Li, 2025). LLM judges (Zheng et al., 2023) can favor their own outputs (Panickssery et al., 2024), familiar text (Wataoka et al., 2025), or related student models (Li et al., 2026a). Post-hoc calibration removes estimated length effects (Huang et al., 2025b); DIR constrains attribute dependence (Li et al., 2026b), while DIAL aligns domain representations (Wu & Choudhury, 2025). SPARC instead calibrates source-group mean score gaps and adjacent-progress changes among same-problem, verified-correct trajectories, with terminal ranking protection. Process evaluation and AceMath-RewardBench candidate selection (Liu et al., 2025) test the resulting reasoning utility. Appendix A details these connections.

6 Conclusion

Self-preference introduces a source dependence into PRM scoring that is visible before a reasoning trajectory is complete. Our analysis shows that this effect persists even when answer correctness is controlled and strengthens over intermediate reasoning progress, motivating calibration at the process level rather than only at the final score. We introduce SPARC to address this issue by calibrating own-base and other-model scores across reasoning progress while retaining correct–incorrect discrimination. Across PRMs and mathematical reasoning benchmarks, SPARC reduces source dependence and generalizes to sources and benchmark families excluded from calibration. These results suggest that mitigating self-preference can make PRM scores less sensitive to trajectory source and, in some settings, translate into better process evaluation and candidate selection.

AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic datasets. We have not used generative AI tools to implement methods, to analyse or interpret experimental results, to clean or reformat data, or to formulate or refine hypotheses, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools to aid and polish the writing, to debug experimental code, and to write auxiliary code. We have reviewed all AI-assisted work: we checked the complete manuscript sentence by sentence, and we produced all figures and references without AI assistance. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All datasets and PRM checkpoints used in this work are publicly available; the source registry in Appendix C.1 lists each PRM together with its documented base model. The trajectories analysed here are generated by the models in that registry, and Appendix C describes their answer verification, the natural-prefix extraction and PRM rendering protocol, and the data splits. Appendix E reports the adapter configuration, the optimization settings, and the development-set selection procedure used for the three seeds, and Appendix F specifies the baseline constructions. Training used two NVIDIA A100 40GB GPUs. Anonymized source code is available in the supplementary material.

REFERENCES

Aakriti Agrawal, Souradip Chakraborty, Armin Saghafian, Nihal Sharma, Rizal Fathony, Nam H Nguyen, C. Bayan Bruss, Amrit Singh Bedi, and Furong Huang. The hidden bias of process reward models:prism for rewarding the right reasoning, 2026. URL https://arxiv.org/abs/2606.09078.

Anirudh Bharadwaj, Chaitanya Malaviya, Nitish Joshi, and Mark Yatskar. Flattery, fluff, and fog: Diagnosing and mitigating idiosyncratic biases in preference models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=tsfjjfhEz7.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. Alphamath almost zero: Process supervision without process. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL https://papers.nips.cc/paper_files/paper/2024/hash/30dfe47a3ccbee68cffa0c19ccb1bc00-Abstract-Conference.html.

Wei-Lin Chen, Zhepei Wei, Xinyu Zhu, Shi Feng, and Yu Meng. Do llm evaluators prefer themselves for a reason?, 2025a. URL https://arxiv.org/abs/2504.03846.

Zhi-Yuan Chen, Hao Wang, Xinyu Zhang, Enrui Hu, and Yankai Lin. Beyond the surface: Measuring self-preference in LLM judgments. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 1653–1672, Suzhou, China, November 2025b. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.86. URL https://aclanthology.org/2025.emnlp-main.86/.

Jie Cheng, Gang Xiong, Ruixi Qiao, Lijun Li, Chao Guo, Junle Wang, Yisheng Lv, and Fei-Yue Wang. Stop summation: Min-form credit assignment is all process reward model needs for reasoning. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL https://papers.nips.cc/paper_files/paper/2025/hash/be91eb86eb74efc055cff83e953f86ce-Abstract-Conference.html.

Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. CLUB: A contrastive log-ratio upper bound of mutual information. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1779–1788. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/cheng20b.html.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016. URL http://jmlr.org/papers/v17/15-239.html.

Jujie He, Tianwen Wei, Rui Yan, Jiacai Liu, Chaojie Wang, Yimeng Gan, Shiwen Tu, Chris Yuhao Liu, Liang Zeng, Xiaokun Wang, Boyang Wang, Yongcong Li, Fuxiang Zhang, Jiacheng Xu, Bo An, Yang Liu, and Yahui Zhou. Skywork-o1 open series, November 2024. URL https://doi.org/10.5281/zenodo.16998085.

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. In Joaquin Vanschoren and Sai-Kit Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Kaixuan Huang, Jiacheng Guo, Zihao Li, Xiang Ji, Jiawei Ge, Wenzhe Li, Yingqing Guo, Tianle Cai, Hui Yuan, Runzhe Wang, Yue Wu, Ming Yin, Shange Tang, Yangsibo Huang, Chi Jin, Xinyun Chen, Chiyuan Zhang, and Mengdi Wang. Math-perturb: Benchmarking llms' math reasoning abilities against hard perturbations, 2025a. URL https://arxiv.org/abs/2502.06453.

Zeyu Huang, Zihan Qiu, Zili Wang, Edoardo Ponti, and Ivan Titov. Post-hoc reward calibration: A case study on length bias. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=Iu8RytBaji.

IBM Granite Team. Granite-3.3-8b-math-prm-v2. Hugging Face model card, 2026. URL https://huggingface.co/ibm-granite/granite-3.3-8b-math-prm-v2.

Dawei Li, Renliang Sun, Yue Huang, Ming Zhong, Bohan Jiang, Jiawei Han, Xiangliang Zhang, Wei Wang, and huan liu. Preference leakage: A contamination problem in LLM-as-a-judge. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=grIvSXVJ65.

Wendi Li and Yixuan Li. Process reward model with q-value rankings. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=wQEdh2cgEk.

Zhuo Li, Pengyu Cheng, Zhechao Yu, Feifei Tong, Anningzhe Gao, Tsung-Hui Chang, Xiang Wan, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Eliminating inductive bias in reward models with information-theoretic guidance. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=57YfUhcYXd.

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

Zihan Liu, Yang Chen, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. AceMath: Advancing frontier math reasoning with post-training and reward modeling. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 3993–4015, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.206. URL https://aclanthology.org/2025.findings-acl.206/.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. Improve mathematical reasoning in language models by automated process supervision, 2024. URL https://arxiv.org/abs/2406.06592.

Sadegh Mahdavi, Muchen Li, Kaiwen Liu, Christos Thrampoulidis, Leonid Sigal, and Renjie Liao. Leveraging online olympiad-level math problems for LLMs training and contamination-resistant evaluation. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 42554–42578. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/mahdavi25a.html.

Ignavier Ng, Patrick Blöbaum, Siddharth Bhandari, Kun Zhang, and Shiva Kasiviswanathan. Debiasing reward models by representation learning with guarantees, 2025. URL https://arxiv.org/abs/2510.23751.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. LLM evaluators recognize and favor their own generations. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL https://papers.nips.cc/paper_files/paper/2024/hash/7f1f0218e45f5414c79c0679633e47bc-Abstract-Conference.html.

Qwen Team. Qwen2.5-math-prm-7b. Hugging Face model card, 2025. URL https://huggingface.co/Qwen/Qwen2.5-Math-PRM-7B.

RLHFlow Team. Llama3.1-8b-prm-mistral-data. Hugging Face model card, 2024. URL https://huggingface.co/RLHFlow/Llama3.1-8B-PRM-Mistral-Data. DeepSeek-data checkpoint: https://huggingface.co/RLHFlow/Llama3.1-8B-PRM-Deepseek-Data.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=A6Y7AqlzLW.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. PRMBench: A fine-grained and challenging benchmark for process-level reward models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25299–25346, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1230. URL https://aclanthology.org/2025.acl-long.1230/.

Rishabh Tiwari, Aditya Tomar, Udbhav Bamba, Monishwaran Maheswaran, Heng Yang, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. Reward under attack: Analyzing the robustness and hackability of process reward models, 2026. URL https://arxiv.org/abs/2603.06621.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

Koki Wataoka, Tsubasa Takahashi, and Ryokan Ri. Self-preference bias in llm-as-a-judge, 2025. URL https://arxiv.org/abs/2410.21819v2.

David Wu and Sanjiban Choudhury. Aligning llms with domain invariant reward models, 2025. URL https://arxiv.org/abs/2501.00911.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. Evaluating mathematical reasoning beyond accuracy. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39 (26):27723–27730, April 2025. ISSN 2159-5399. doi: 10.1609/aaai.v39i26.34987. URL http://dx.doi.org/10.1609/aaai.v39i26.34987.

Bin Xie, Bingbing Xu, Xueyun Tian, Yilin Chen, and Huawei Shen. Towards robust process reward modeling via noise-aware learning, 2026. URL https://arxiv.org/abs/2601.12748.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 73511–73525. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/yuan25c.html.

Hugh Zhang, Jeff Da, Dean Lee, Vaughn Robinson, Catherine Wu, William Song, Tiffany Zhao, Pranav Raja, Charlotte Zhuang, Dylan Slack, Qin Lyu, Sean Hendryx, Russell Kaplan, Michele Lunati, and Summer Yue. A careful examination of large language model performance on grade school arithmetic. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL https://papers.nips.cc/paper_files/paper/2024/hash/53384f2090c6a5cac952c598fd67992f-Abstract-Datasets_and_Benchmarks_Track.html.

Yifan Zhang, Ge Zhang, Yue Wu, Kangping Xu, and Quanquan Gu. Beyond bradley-terry models: A general preference model for language model alignment. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 76939–76965. PMLR, 2025a. URL https://proceedings.mlr.press/v267/zhang25de.html.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 10495–10516, Vienna, Austria, July 2025b. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.547. URL https://aclanthology.org/2025.findings-acl.547/.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. ProcessBench: Identifying process errors in mathematical reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*

(Volume 1: Long Papers), pp. 1009–1024, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.50. URL https://aclanthology.org/2025.acl-long.50/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL https://papers.nips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

Jiaru Zou, Ling Yang, Jingwen Gu, Jiahao Qiu, Ke Shen, Jingrui He, and Mengdi Wang. Reasonflux-prm: Trajectory-aware prms for long chain-of-thought reasoning in llms. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL https://papers.nips.cc/paper_files/paper/2025/hash/26618fb384d3873b8ef6ab292a69095b-Abstract-Conference.html.

Guide to the appendices.

Appendices A–C give related work, metric definitions, and data protocols; Appendix D supports the self-preference analysis in Section 2. Appendices E–F specify SPARC and its baselines. Appendix G reports complete component ablations, and Appendix H is an analysis of generalization beyond the calibration data.

A Extended Related Work

Verification and process rewards. The supervision target determines which aspects of reasoning a verifier learns: outcome verifiers judge complete solutions (Cobbe et al., 2021), whereas process supervision supplies intermediate correctness feedback (Lightman et al., 2024; Wang et al., 2024). ReasonEval separates step validity from redundancy (Xia et al., 2025), and ReasonFlux combines step-level and trajectory-level supervision for reasoning traces that include intermediate thinking and a final response (Zou et al., 2025). Label quality and ranking quality pose different training problems: noise-aware training addresses unreliable step labels (Xie et al., 2026), while PRISM uses contrastive hard negatives to reduce overcrediting of plausible but incorrect steps (Agrawal et al., 2026). A complementary formulation learns intermediate value rankings (Li & Li, 2025). These approaches improve reasoning assessment through supervision or ranking; our diagnostic asks whether verified-correct solutions to the same problem receive different scores depending on their source, and how this difference evolves throughout reasoning.

Self-preference and familiarity. Authorship and familiarity offer related but distinct accounts of evaluator preferences. Generative judges can recognize and favor their own outputs (Panickssery et al., 2024), whereas the association between lower perplexity and more favorable judgments also extends to other models' responses (Wataoka et al., 2025). Preference leakage concerns students trained on synthetic data from a model related to the judge (Li et al., 2026a); it need not involve judging the evaluator's own outputs. Our own-base comparison instead links a scalar PRM to its language-model backbone and measures source-dependent scores at intermediate prefixes as well as completion. PPL-Cal operationalizes the familiarity account through a training-fitted score correction. Correctness and capability controls separate the measured preference from differences in generator answer accuracy.

Calibration and invariance. Length, sycophancy, and formatting are studied bias attributes in reward modeling (Li et al., 2026b). In PRMs, adversarial optimization can also exploit stylistic shortcuts and inflate rewards on invalid trajectories (Tiwari et al., 2026). Debiasing methods differ in what they modify: counterfactual training data target evaluator preferences (Bharadwaj et al., 2026), while latent-variable modeling seeks representations that retain intended preference factors (Ng et al., 2025). Post-hoc calibration estimates and subtracts length-related reward effects without retraining the reward model (Huang et al., 2025b). DIR combines preference learning with a variational mutual-information penalty between paired hidden representations and specified bias attributes (Li et al., 2026b). DIAL transfers preferences from labeled source data to unlabeled target data by reducing the Wasserstein distance between their representations (Wu & Choudhury, 2025). These objectives target attribute dependence or domain shift. SPARC instead penalizes mean score gaps within matched correct-trajectory groups at each reasoning-progress point and the change in those gaps between adjacent points. Group-mean calibration allows individual solutions to retain different scores. A shared LoRA adapter (Hu et al., 2022) couples these losses with terminal correctness protection. Our SafeScore-DIR baseline adapts the source-invariance principle to scalar PRM scores and adds terminal ranking protection; it is distinct from the original DIR formulation. Appendix F specifies the adaptations.

Evaluation beyond source parity. Source-gap reduction and reasoning utility answer different questions. ProcessBench measures reasoning-error identification (Zheng et al., 2025), while PRM-Bench assesses fine-grained process rewards (Song et al., 2025). AceMath-RewardBench evaluates selection among generated solutions (Liu et al., 2025), connecting scorer quality to the use of inference-time computation (Snell et al., 2025). Our comparisons hold evaluation examples and candidate pools fixed between the original and calibrated PRM. They therefore measure changes

in reasoning evaluation and candidate selection arising from the scorer, alongside the source-gap diagnostics.

B Score Gaps and Their Interpretation

Paired gaps. Following paired-score comparisons of evaluator preferences (Bharadwaj et al., 2026), we compare verified correct trajectories from the PRM's own base model and another model on the same problem x. For a trajectory with N natural steps, $y_m^{+,p}$ ends at step $\lceil pN \rceil$, with $p \in \{0.25, 0.50, 0.75, 1\}$. The native endpoint score gives

$$\mathrm{gap}_p(\theta) = s_\theta(x, y_{\mathrm{base}}^{+,p}) - s_\theta(x, y_{\mathrm{other}}^{+,p}), \qquad \mathrm{gap} = \mathrm{gap}_1. \tag{9}$$

Aggregation and score scale. Within group c, normalized pair weights w_j give each problem equal total weight. We report the exponentiated mean gap:

$$\mu_c = \sum_j w_j \, \mathrm{gap}_p^{(j)}, \qquad \mathrm{GAP}_p = \exp(\mu_c), \qquad \mathrm{GAP} = \mathrm{GAP}_1. \tag{10}$$

Parity is $1\times$; values above one favor own-base trajectories, and values below one favor other-model trajectories on average. Magnitudes retain the PRM's native score scale. Qwen and LLaMA subtract negative from positive logits, yielding log odds $\log[P_\theta(+ \mid x, y)/P_\theta(- \mid x, y)]$. For these interfaces,

$$\mathrm{GAP}_p = \exp\left(\sum_j w_j \log \frac{\mathrm{odds}_{\mathrm{base}}^{(j)}}{\mathrm{odds}_{\mathrm{other}}^{(j)}}\right) = \prod_j \left(\frac{\mathrm{odds}_{\mathrm{base}}^{(j)}}{\mathrm{odds}_{\mathrm{other}}^{(j)}}\right)^{w_j}. \tag{11}$$

This is a weighted geometric mean odds ratio. Skywork instead uses a raw scalar value-head output: exponentiating its gap has no odds-ratio interpretation. GAP magnitudes are consequently compared within a PRM.

Calibration and ranking. With normalized evaluation weights w_c across source groups,

$$\mathrm{GAP} = \exp\left(\sum_c w_c \mu_c\right), \qquad \mathrm{RMS}_{\mathrm{group}} = \sqrt{\sum_c w_c \mu_c^2}. \tag{12}$$

GAP retains the direction of the average difference; group RMS captures oppositely signed group means even when they cancel in the overall mean. Terminal ranking accuracy measures whether a correct answer scores strictly above an incorrect answer, counting ties as errors. Acc is the unweighted mean of A, B, and G accuracies. Source calibration and correctness discrimination therefore require separate measurements. Self-preference rate measures how often the own-base score is higher, giving half credit to ties; generator answer accuracy measures how often the generator answers correctly. Their populations are defined in Appendix D.2.

Statistical unit. Problem-bootstrap intervals retain all comparisons for each sampled problem; the GAP interval exponentiates the mean-gap interval endpoints. The four prefixes are repeated observations, not independent problems. Learned-method comparisons use matched evaluation rows across seeds 17, 29, and 43. Table captions specify when paired intervals also resample training seeds.

C Data Construction and Evaluation Details

The experiments use verified trajectories with model identities and natural step boundaries. Diagnostic populations and calibration partitions are specified separately below; input question counts precede PRM eligibility filtering.

C.1 Source Registry and Data Provenance

Table 3 maps every PRM to its documented base model. The registry distinguishes the PRM used for scoring from the model that generated a trajectory. Other models are the registered generators

other than the PRM's own base model, regardless of answer quality or correctness. Each raw record stores the dataset and problem identifier, model identifier, completion, final-outcome status, natural segment markers, and the renderer version used to produce PRM inputs. Filtering records distinguish answer-verification failures, missing step markers, and inputs that exceed a PRM's context limit.

Table 3: PRMs and their documented base models. Each mapping follows the checkpoint lineage reported in the corresponding paper or model card.

PRM	base model
Qwen-Math (Qwen Team, 2025)	Qwen2.5-Math-Instruct
Skywork (He et al., 2024)	Qwen2.5-Math-Instruct
PURE (Cheng et al., 2025)	Qwen2.5-Math Base
ReasonFlux (Zou et al., 2025)	Qwen-Instruct
RLHFlow Mistral-data (RLHFlow Team, 2024)	Llama-3.1-Instruct
RLHFlow DeepSeek-data (RLHFlow Team, 2024)	Llama-3.1-Instruct
Granite-Math (IBM Granite Team, 2026)	Granite-3.3-Instruct
ReasonEval (Xia et al., 2025)	WizardMath-7B

C.2 Final Outcome Verification

Final outcomes are verified with the benchmark-specific exact-answer checker or the released answer validator. A trajectory is marked correct only when its complete final answer passes that validator; otherwise it is marked incorrect or excluded when the validator cannot decide. The label applies to the complete trajectory, not to its intermediate prefixes. We use these final-answer labels to construct correct–correct calibration groups and terminal A/B/G pairs; the calibration groups do not require step-level correctness labels.

C.3 Natural-Prefix Extraction and PRM Rendering

We preserve the original segment order of every completion. For a trajectory with N natural steps, the prefix at progress p contains the first $\lceil pN \rceil$ steps. Progress is measured by step count, not token count. The $p = 1$ observation is the complete trajectory. Prefix extraction preserves segment text and uses only the completion's existing step markers. A prefix is eligible only when its distinct marker and complete input pass the target PRM's native template, tokenizer, and context-length checks. Overflow is recorded as an exclusion rather than resolved by truncation. The exact rendered input is cached with the relation manifest so that the Original PRM and the adapted PRM score the same text.

C.4 Dataset Statistics and Split Details

GSM8K (1,319 questions) and MATH500 (500 questions) supply complete-trajectory diagnostics and do not enter calibration training. MATH500 also supplies external candidate-selection problems. GSM1k (1,205 questions) and MATH-P (279 Simple and 279 Hard questions) are split by problem identifier using a deterministic hash partition with nominal proportions 70/10/20 for training, development, and test. Corresponding Simple and Hard perturbations of the same MATH-P problem remain in the same partition. All trajectories, prefixes, and comparison pairs inherit their problem's assignment. Filtering for correctness and valid PRM inputs is performed within these partitions, so the final eligible counts need not follow the nominal percentages exactly. Where a table reports MATH-P as a single entry, the two components are combined by weighting each component by the eligible count it contributes to that comparison, so the effective weight is set by the eligible counts of the analysis being reported and can differ between tables. The n shown for such an entry is the sum of the two components and is therefore not proportional to the effective weight; that entry has $n = 22$ at completion in Table 11 and $n = 34$ in Table 12.

LiveAoPS uses January–June 2024 (2,788 questions) for training, July–August (620 questions) for development, and September–December (1,280 questions) for the reported calibration evaluation. The diagnostic survey uses different windows from the calibration split. The final training split (January–June), development split (July–August), and reported evaluation window (September–

December) are disjoint, and the final configuration was frozen before the reported September–December metrics were computed. The September–December view was constructed by repartitioning an earlier July–October development / November–December test view, however, so some September–October questions appeared in earlier development comparisons. The reported results therefore reflect evaluation of the frozen final configuration on the September–December population, rather than selection on the reported evaluation metrics.

Counting comparisons. A unique problem can contribute several source comparisons and several progress observations. We therefore distinguish unique problems, pair memberships, source groups, and seed-specific ranking pairs. In particular, three repetitions of the same ranking pair across training seeds are not three independent test questions.

D Additional Self-Preference Diagnostics

We provide the counts and numerical results underlying Section 2, followed by correctness controls and additional progress profiles.

D.1 Complete Cross-PRM Results

Qwen-Math reaches $\mathrm{GAP} = 48.859$ on GSM8K and 28.942 on MATH500. Table 4 gives the numerical values, pointwise intervals, and eligible counts for all PRMs from the original complete-trajectory comparison, which draws on every eligible problem of each benchmark rather than on the held-out calibration split.

The additional datasets use the natural-prefix protocol at $p = 1$, including its requirement of at least four retained natural steps. The pooled LiveAoPS 2024 set combines three windows of 2,788, 1,255, and 645 questions (January–June, July–October, and November–December 2024); only strictly verified correct trajectories enter these comparisons. Aggregation follows Appendix B. Intervals for the additional datasets use 10,000 problem-bootstrap draws; GSM8K and MATH500 retain the intervals from the complete-trajectory comparison. For ReasonFlux, we score the original complete trajectories with the same terminal-marker interface used on GSM8K and MATH500. Two of its 7,271 additional trajectories exceed the 4,096-token limit and are excluded; both affected problems retain other valid comparisons.

Table 4: **Own-base correct ↔ other-model correct.** Complete-trajectory self-preference across five benchmarks. MATH-P pools Simple and Hard; LiveAoPS pools the three 2024 windows. Problems carry equal weight within each source group, and groups are combined by normalized source weights. n counts eligible problems; intervals are pointwise 95% problem-bootstrap intervals. The counts cover every eligible problem of each benchmark, including those assigned to the calibration splits, and therefore do not match the held-out test results of Table 1.

Dataset	PRM	n	GAP	95% CI
GSM8K	Qwen-Math	1240	48.859	[43.591, 54.521]
	Skywork	1240	5.234	[4.923, 5.590]
	ReasonFlux	1132	2.332	[2.064, 2.619]
	RLHFlow Mistral-data	1027	12.898	[12.026, 13.829]
	RLHFlow DeepSeek-data	1027	19.466	[18.223, 20.766]
	PURE	760	2.289	[2.055, 2.534]
MATH500	Qwen-Math	372	28.942	[22.060, 38.001]
	Skywork	372	4.542	[3.887, 5.238]
	ReasonFlux	345	3.821	[2.912, 4.961]
	RLHFlow Mistral-data	202	4.454	[3.717, 5.450]
	RLHFlow DeepSeek-data	202	5.552	[4.464, 6.970]
	PURE	242	1.408	[1.125, 1.773]
GSM1k	Qwen-Math	609	7.433	[6.161, 8.964]
	Skywork	609	2.834	[2.604, 3.084]
	ReasonFlux	606	4.023	[3.425, 4.714]
	RLHFlow Mistral-data	574	1.672	[1.169, 1.983]
	RLHFlow DeepSeek-data	574	0.578	[0.467, 0.712]
	PURE	637	1.281	[1.160, 1.414]
MATH-P Simple&Hard	Qwen-Math	192	40.204	[24.739, 65.788]
	Skywork	192	2.356	[1.912, 2.922]
	ReasonFlux	179	2.525	[1.716, 3.693]
	RLHFlow Mistral-data	109	3.624	[2.311, 5.682]
	RLHFlow DeepSeek-data	109	7.725	[4.875, 12.224]
	PURE	156	1.934	[1.520, 2.459]
LiveAoPS 2024	Qwen-Math	1057	17.716	[13.978, 22.336]
	Skywork	1057	1.707	[1.566, 1.857]
	ReasonFlux	971	1.684	[1.414, 2.003]
	RLHFlow Mistral-data	381	2.424	[1.923, 3.076]
	RLHFlow DeepSeek-data	381	4.353	[3.438, 5.556]
	PURE	933	1.152	[1.052, 1.263]

D.2 Scoring Protocols for Figure 3

Self-preference rate. For a problem x and a generator m, let $\bar{s}_m(x)$ be the mean PRM score over the saved trajectories from m on x, and let $\mathcal{O}(x)$ be the other generators with a valid scored trajectory on x. The base–other score difference on that problem is

$$d(x) = \bar{s}_{\text{base}}(x) - \frac{1}{|\mathcal{O}(x)|} \sum_{m \in \mathcal{O}(x)} \bar{s}_m(x). \tag{13}$$

The self-preference rate averages this comparison over the eligible problems $\mathcal{P}$, giving half credit when the two sides coincide:

$$\text{SPR} = \frac{1}{|\mathcal{P}|} \sum_{x \in \mathcal{P}} \Big(\mathbb{1}\{d(x) > 0\} + \tfrac{1}{2} \mathbb{1}\{d(x) = 0\} \Big). \tag{14}$$

The all-outcome pool $\mathcal{P}$ requires a valid base-model score and at least one valid other-model score, regardless of correctness. The CC comparison instead uses only verified correct trajectories on both sides.

Base-model answer accuracy. The horizontal axis is the fraction of all dataset questions for which the base model's saved rollout has a verified correct final answer. For LiveAoPS it is the strictly

Table 5: **Own-base incorrect ↔ other-model correct.** Pair-weighted native terminal scores: each incorrect own-base answer is paired with every available correct other-model answer to the same problem. n counts problems and N counts answer pairs. Δ is Own minus Other; intervals are pointwise 95% problem-cluster bootstrap intervals (10,000 resamples). LiveAoPS covers January–June 2024. Native units differ across PRMs.

Dataset	PRM	n	N	Own	Other	Δ	95% CI
GSM8K	Qwen	33	72	3.34	1.65	+1.69	[+0.35, +3.34]
	Skywork	33	72	0.74	0.32	+0.42	[-0.43, +1.43]
	RLH-M	259	822	27.13	25.17	+1.96	[+1.76, +2.17]
	RLH-D	259	822	23.73	21.58	+2.15	[+1.91, +2.40]
	PURE	266	1045	0.31	1.81	-1.50	[-1.84, -1.17]
	ReasonFlux	156	447	6.04	7.67	-1.63	[-2.25, -1.02]
MATH500	Qwen	19	27	2.48	1.85	+0.63	[-1.96, +2.92]
	Skywork	19	27	0.20	0.59	-0.38	[-1.58, +0.84]
	RLH-M	145	384	24.88	24.13	+0.75	[+0.47, +1.01]
	RLH-D	145	384	20.85	20.30	+0.54	[+0.10, +0.97]
	PURE	81	191	1.51	1.89	-0.38	[-1.03, +0.29]
	ReasonFlux	65	111	0.83	5.33	-4.50	[-5.53, -3.45]
MATH-P-Simple	Qwen	34	63	7.38	5.03	+2.35	[+0.06, +4.69]
	Skywork	34	63	0.25	0.38	-0.13	[-1.42, +1.18]
	RLH-M	66	147	24.52	24.37	+0.15	[-0.82, +1.16]
	RLH-D	66	147	21.88	21.85	+0.03	[-1.32, +1.41]
	PURE	37	74	1.28	1.60	-0.32	[-1.02, +0.37]
	ReasonEval	136	380	1.88	1.54	+0.35	[-0.07, +0.78]
MATH-P-Hard	Qwen	33	46	7.75	4.41	+3.33	[+0.86, +5.80]
	Skywork	33	46	-0.73	-1.37	+0.64	[-0.54, +1.85]
	RLH-M	56	118	24.90	24.07	+0.82	[-0.27, +2.03]
	RLH-D	56	118	23.08	21.47	+1.61	[+0.67, +2.71]
	PURE	42	72	1.33	1.27	+0.06	[-0.46, +0.61]
	ReasonEval	87	178	1.66	1.11	+0.55	[+0.04, +1.06]
LiveAoPS	Qwen	273	438	5.71	4.41	+1.30	[+0.55, +2.07]
	Skywork	273	438	-0.57	0.52	-1.09	[-1.55, -0.64]
	RLH-M	390	812	24.71	24.56	+0.15	[-0.16, +0.46]
	RLH-D	390	812	22.43	22.23	+0.20	[-0.19, +0.59]
	PURE	319	563	0.80	1.38	-0.59	[-0.82, -0.36]
	ReasonEval	677	1640	1.22	1.43	-0.21	[-0.37, -0.04]

verified-correct fraction over the 2,788 questions of the January–June 2024 window. The resulting points for all six panels are reported in Appendix D.3, which also audits whether the same preference survives conditioning on final-answer correctness.

D.3 Capability and Final-Wrong Audits

Incorrect own-base versus correct other-model answers. Figure 3 and Table 7 use one set of saved model responses, whereas this analysis and Figure 8 use a separate set. The two describe different response populations and should not be combined. Each incorrect own-base answer is paired with every available correct other-model answer to the same problem. Table 5 weights these pairs equally and reports native mean scores, problem counts, and pair counts. Its pointwise 95% intervals use 10,000 problem-cluster bootstrap resamples, retaining every pair of a sampled problem.

Across the thirty PRM–dataset comparisons, incorrect own-base answers score higher in twenty, and in ten of those the pointwise interval lies wholly above zero. GSM8K and MATH500 score the complete answer as one terminal unit; MATH-P and LiveAoPS read the final natural-step score after the complete reasoning prefix. Each pair uses the same interface for both sources within its PRM and dataset.

Radar score normalization. The radar panels in Figure 1(b) show Qwen-Math, ReasonEval, and the Mistral-data RLHFlow PRM across GSM8K, MATH500, GSM1k, MATH-P-Simple, MATH-P-

Hard, and LiveAoPS (January–June 2024). They use the incorrect-own-base versus correct-other-model pairs defined above. A problem with k eligible correct other-model answers contributes k pairs, repeating its own-base score in each pair. For a fixed PRM and benchmark, let b_j and o_j be the native terminal scores of pair j, and let B and O denote their respective means over the N pairs. We standardize both groups using their joint mean and population standard deviation:

$$\mu = \frac{B+O}{2}, \qquad \sigma^2 = \frac{1}{2N}\sum_{j=1}^{N}\left[(b_j-\mu)^2 + (o_j-\mu)^2\right],$$
$$z_{\text{own}} = \frac{B-\mu}{\sigma}, \qquad z_{\text{other}} = \frac{O-\mu}{\sigma} = -z_{\text{own}}. \tag{15}$$

Radial ticks denote transformed scores, rather than probabilities or GAP; the correct–correct GAP analysis in Section 2.1 uses a different population and aggregation. Table 6 gives the values underlying these panels. The broader PRM comparison is reported in Table 5.

Table 6: Scores underlying the radar panels in Figure 1(b). n counts problems and N counts answer pairs. Own and Other are the pair-weighted native terminal means B and O; σ is the population standard deviation of the pooled $2N$ scores. The standardized other-model mean is $-z_{\text{own}}$. LiveAoPS covers January–June 2024.

Dataset	PRM	n	N	Own	Other	σ	z_{own}
GSM8K	Qwen	33	72	3.342	1.649	3.630	+0.2332
	ReasonEval	143	412	3.603	3.109	1.630	+0.1515
	RLHFlow Mistral†	259	822	26.858	25.445	2.076	+0.3401
MATH500	Qwen	19	27	2.482	1.851	4.010	+0.0787
	ReasonEval	224	620	2.400	2.230	1.941	+0.0438
	RLHFlow Mistral	145	384	24.878	24.132	1.799	+0.2073
GSM1k	Qwen	56	149	6.307	7.595	3.991	-0.1613
	ReasonEval	160	508	2.933	2.363	1.528	+0.1864
	RLHFlow Mistral†	112	330	23.447	24.032	2.240	-0.1305
MATH-P-Simple	Qwen	34	63	7.384	5.031	5.181	+0.2270
	ReasonEval	136	380	1.884	1.535	2.024	+0.0862
	RLHFlow Mistral	66	147	24.518	24.369	2.749	+0.0271
MATH-P-Hard	Qwen	33	46	7.746	4.415	5.251	+0.3171
	ReasonEval	87	178	1.656	1.106	1.913	+0.1436
	RLHFlow Mistral	56	118	24.897	24.073	2.702	+0.1525
LiveAoPS	Qwen	273	438	5.708	4.411	4.912	+0.1320
	ReasonEval	677	1640	1.222	1.429	1.723	-0.0602
	RLHFlow Mistral	390	812	24.715	24.563	2.435	+0.0312

Figure 8 gives the complementary problem-level rate on GSM8K and MATH500. It compares each incorrect own-base score with the mean of its correct other-model scores and includes ties. Each problem has equal weight, unlike the pair-weighted means above. This rate is distinct from terminal ranking accuracy (Acc).

Answer accuracy uses the full question counts in Appendix C. Scored pools require parseable completions, valid native scores, and, beyond GSM8K/MATH500, at least four natural steps. GSM8K/MATH500 use seven generator policies, GSM1k/MATH-P eight, and LiveAoPS nine. Each correlation uses six PRMs: Qwen-Math, Skywork, PURE, both RLHFlow PRMs, and either ReasonFlux on GSM8K/MATH500 or ReasonEval on the other datasets. Invalid scores and context overflow are excluded.

For LiveAoPS, the nine generators produce 25,092 answers on the 2,788 January–June questions. Strict verification labels 4,099 correct and 11,857 incorrect; 9,136 are unresolved or generation-ineligible. The capability coordinate divides verified-correct answers by all questions. Scored comparisons use only resolved answers with valid natural segments and terminal scores, without relabeling unresolved answers as incorrect. The terminal manifests contain 3,396 correct and 9,621 incorrect trajectories; context limits exclude 14 correct-pool and 104 incorrect-pool PRM records. Using resolved-only accuracy instead of the verification pass rate changes Pearson correlation from 0.68 to 0.82, with Spearman correlation unchanged at 0.71.

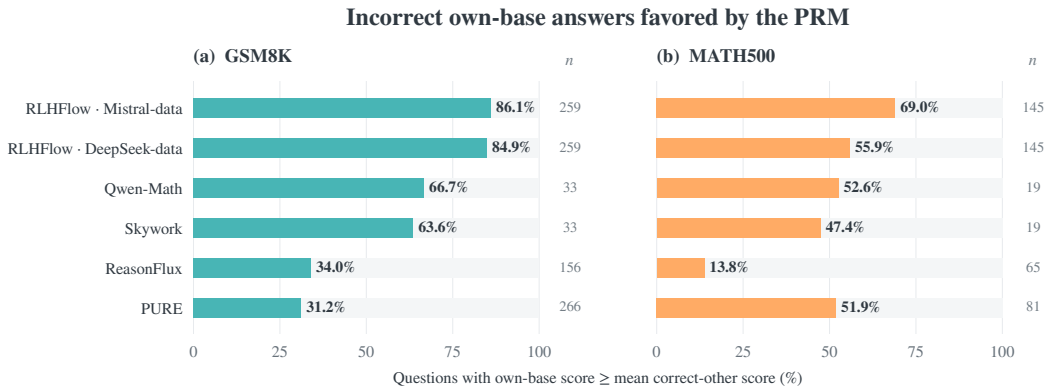


Figure 8: Problem-level preference for incorrect own-base answers at $p = 1$. n counts eligible questions.

Table 7 lists the PRM-level points plotted in Figure 3, from which its correlations are computed. These descriptive correlations motivate controlling for answer correctness. The matched-trajectory comparisons then measure residual self-preference among correct answers.

Table 7: The points plotted in Figure 3. Each row is one of the six PRMs in the corresponding panel. Base accuracy is the fraction of the base model's saved rollouts whose final answer is verified correct; the self-preference rate compares the own-base score with the mean score of the other models on the same problem, over all valid scored answers, giving half credit to ties. Both columns are percentages. The Pearson and Spearman correlations reported for each panel are computed from the underlying unrounded values, displayed here to one decimal. For LiveAoPS the base-model accuracy is a verification pass rate, with the resolved-answer variant in parentheses; ReasonFlux is replaced by ReasonEval on the last four datasets.

Dataset	PRM	Base acc.	Self-pref. rate	Dataset	PRM	Base acc.	Self-pref. rate
GSM8K	PURE	84.8	70.4	MATH-P-Simple	PURE	55.9	78.8
	Qwen-Math	95.4	98.2		Qwen-Math	53.8	94.4
	Skywork	95.4	97.1		Skywork	53.8	81.8
	ReasonFlux	86.1	82.4		ReasonEval	18.3	62.8
	RLHFlow DeepSeek-data	85.3	97.7		RLHFlow DeepSeek-data	26.2	77.8
	RLHFlow Mistral-data	85.3	98.7		RLHFlow Mistral-data	26.2	70.4
MATH500	PURE	56.4	73.4	MATH-P-Hard	PURE	29.7	83.4
	Qwen-Math	80.0	93.0		Qwen-Math	23.7	87.3
	Skywork	80.0	92.2		Skywork	23.7	71.4
	ReasonFlux	70.2	79.7		ReasonEval	8.6	65.0
	RLHFlow DeepSeek-data	48.0	84.6		RLHFlow DeepSeek-data	8.6	73.8
	RLHFlow Mistral-data	48.0	93.5		RLHFlow Mistral-data	8.6	70.0
GSM1k	PURE	79.8	61.9	LiveAoPS	PURE	29.9 (38.1)	70.6
	Qwen-Math	87.9	85.8		Qwen-Math	31.6 (41.2)	84.9
	Skywork	87.9	89.6		Skywork	31.6 (41.2)	74.9
	ReasonEval	75.4	91.1		ReasonEval	7.3 (10.1)	55.7
	RLHFlow DeepSeek-data	80.9	43.1		RLHFlow DeepSeek-data	11.4 (24.3)	74.9
	RLHFlow Mistral-data	80.9	45.2		RLHFlow Mistral-data	11.4 (24.3)	69.2

D.4 Per-Benchmark Progress Results

Figures 4 and 9 use correct trajectories from the base model and at least one other model. Aggregation follows Appendix B. The analysis uses all eligible problems in GSM1k and both MATH-P variants. LiveAoPS uses January–June 2024 and evaluates the Original PRMs, before SPARC adaptation.

The problem sets remain fixed across progress. One PURE LiveAoPS problem has fewer valid other-model comparisons at 75% and 100%; excluding it changes each GAP by less than 0.005. One MATH-P-Simple problem likewise has fewer comparisons at completion for Qwen-Math and Skywork. MATH-P-Hard retains both problems and comparison models throughout.

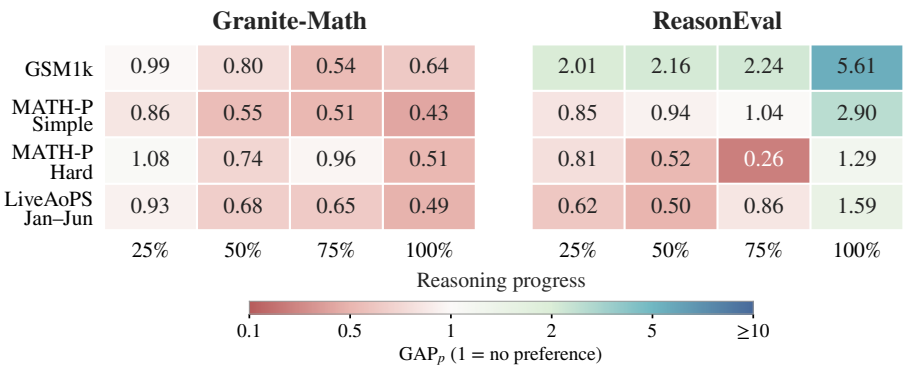


Figure 9: Granite-Math and ReasonEval show weak or reversed self-preference on several datasets and at several reasoning stages. See Appendix D.5 for training context and possible explanations.

D.5 Training Context and Preference Direction

Figure 9 shows that own-base preference is not uniform across PRMs. Granite-Math is below parity in fifteen of sixteen cells. ReasonEval changes direction with the dataset and progress: its MATH-P-Hard GAP rises from 0.26 at 75% to 1.29 at completion, whereas its terminal GSM1k GAP reaches 5.61. Values below one indicate a preference for other-model answers, rather than source neutrality. Native score scales differ across PRMs (Appendix B), so these profiles establish differences in direction without ranking the models by absolute bias magnitude.

Granite-Math: a broader training distribution. The Granite-Math v2 model card documents three stages after Granite-3.3-8B-Instruct initialization: language-model fine-tuning on mathematical and code reasoning, reinforcement learning with verifiable rewards, and process supervision. The last stage uses responses from the Granite family, Phi-4, and Mixtral 8x22B to questions from MetaMathQA, MathInstruct, and NuminaMath (IBM Granite Team, 2026). This exposes the evaluator to successful reasoning outside its own generator family and changes its parameters beyond the original instruct checkpoint used as our own-base reference. A plausible consequence is that the learned correctness signal becomes less tied to that checkpoint's familiar response patterns. Such a shift can also favor external response patterns, consistent with negative mean gaps. The automatic labels follow the OmegaPRM approach, which locates errors using continuation rollouts (Luo et al., 2024); heterogeneous responses therefore do not make its supervision independent of the continuation policy.

ReasonEval: validity separated from redundancy. ReasonEval-7B replaces WizardMath-7B-V1.1's next-token prediction head with a three-class step classifier. It is trained on PRM800K, approximately 800,000 human step annotations over 75,000 GPT-4 solutions, with supervision at each step's final token (Xia et al., 2025). Its labels distinguish incorrect steps, correct but redundant steps, and correct steps that advance the solution. Human judgments supply a direct validity target rather than a continuation model's probability of eventually solving the problem. Separating redundancy also provides supervision about reasoning structure without requiring every valid step to advance the solution.

Let P_-, P_0, and P_+ be the predicted probabilities of these three classes, with logits z_-, z_0, and z_+. ReasonEval defines validity as $v = P_0 + P_+$. The endpoint score used in our GAP computation is therefore

$$s_{\mathrm{RE}} = \log \frac{v}{1-v} = \log \frac{P_0 + P_+}{P_-} = \log(e^{z_0} + e^{z_+}) - z_-. \tag{16}$$

Redistributing predicted probability between the two valid classes leaves this score unchanged when P_- is fixed. The training distinction remains: for label $t \in \{-, 0, +\}$, let $c = \mathbf{1}\{t \neq -\}$ and $u = P_+/v$. The three-class cross-entropy decomposes exactly as

$$-\log P_t = \mathrm{BCE}(c, v) + c\ \mathrm{BCE}(\mathbf{1}\{t = +\}, u)\,, \tag{17}$$

where $\mathrm{BCE}(a, b) = -a \log b - (1-a)\log(1-b)$. The identity follows from $P_+ = vu$, $P_0 = v(1-u)$, and $P_- = 1 - v$. Thus the objective learns validity and, conditional on validity, whether

a step makes progress. This can shape the shared representation differently from a binary validity objective, although it does not impose equal scores across generators. ReasonEval's scoring ablation supports including the neutral-class probability when detecting errors (Xia et al., 2025). Figure 9 uses the current endpoint, so the paper's minimum-over-steps aggregation is not responsible for the displayed differences.

Comparison with the other PRMs. The released Qwen-Math PRM uses Qwen-family responses and combines Monte Carlo labels with a Qwen critic through consensus filtering (Zhang et al., 2025b). Granite's mixture spans more generator families, while ReasonEval's human annotations remove the continuation policy from label construction. Neither distinction is a sufficient explanation by itself. The RLHFlow checkpoints also learn from external Mistral or DeepSeek generators despite their Llama initialization (RLHFlow Team, 2024). PURE likewise uses human-annotated PRM800K and merges its neutral and positive labels into one valid class (Cheng et al., 2025), yet its Figure 4 profile remains above parity. ReasonEval retains a separate redundancy learning objective, but its contribution to source preference is not isolated by these comparisons.

The training records point to response diversity and separate supervision for validity and reasoning utility as possible explanations for these preference profiles. Neither recipe explicitly enforces source parity. Matching final-answer correctness also permits differences in intermediate validity, redundancy, and reasoning completed at the same relative progress. Isolating the contribution of each training component would require holding the backbone, response pool, and score interface fixed.

E SPARC Implementation and Optimization

E.1 Adapter and Optimization Configuration

We train rank-8 query/value LoRA adapters with scaling $\alpha = 8$ and zero dropout, freezing all original PRM parameters including the reward head. AdamW uses learning rate 5×10^{-5} and BF16 arithmetic. All runs use three epochs and seeds 17, 29, and 43. The shared objective coefficients are $\lambda_{\text{drift}} = 1$, $\lambda_{\text{protect}} = 8$, and $\lambda_u = 0.01$; the margin floor, cap, and softplus temperature are $\rho = 0.5$, $\gamma = 4$, and $t = 0.25$. Qwen and Skywork use 262 updates per epoch (786 total), and LLaMA uses 379 per epoch (1,137 total). The adapter architecture, learning rate, training budget, ρ, t, and λ_u are held fixed throughout the Qwen configuration comparisons below. The latter three retain the uncapped control's values; the development comparisons vary the cap, drift term, and protection weight.

The regularizer in Equation 8 is defined on the effective weight updates rather than on the LoRA factor matrices separately:

$$\mathcal{R}(\phi) = \sum_{\ell} \|\Delta W_\ell(\phi)\|_F^2, \qquad \Delta W_\ell(\phi) = \frac{\alpha}{r_{\text{LoRA}}} B_\ell A_\ell, \tag{18}$$

where ℓ indexes the adapted projection matrices, A_ℓ and B_ℓ are the trainable low-rank factors, and $r_{\text{LoRA}} = 8$ is the adapter rank.

E.2 Development Selection and Fixed Budgets

Sequential configuration comparisons. We select the configuration on Qwen's development split, using the partitions in Appendix C. Each candidate is trained for three epochs with seeds 17, 29, and 43 and evaluated at update 786. Table 8 gives the configurations in evaluation order. We first compare a cap of $\gamma = 4$ with drift disabled or enabled at $\lambda_{\text{drift}} = 1$, keeping $\lambda_{\text{protect}} = 1$. We then evaluate the tighter cap $\gamma = 2$ with drift enabled. Because these candidates reduce terminal group RMS but lose A/G accuracy, we return to $\gamma = 4$ and increase the protection weight from 1 to 4 and then 8. The comparisons follow this sequence; they do not form a full parameter grid.

The initial training-only audit gives $L_{\text{correct}} = 16.38$ and $L_{\text{drift}} = 6.47$, motivating a unit drift coefficient as the enabled candidate. The cap of 4 retains a strong positive margin while relaxing extreme Original margins: under the Bradley–Terry link, $\sigma(4) \approx 0.982$. The smaller cap of 2 tests stronger relaxation. An earlier development comparison increased terminal calibration weight to

1.5, 2, or 3 times each earlier progress weight. None met its joint RMS and ranking criteria, so we retain equal total weights across progress levels.

Final selection rule. The initial rule requires no per-benchmark A/B/G decrease and at most a 5% increase in preterminal RMS; none of the five candidates meets that rule. After inspecting the development trade-offs, we revise the rule and compare the last candidate, $(\gamma, \lambda_{\text{drift}}, \lambda_{\text{protect}}) = (4, 1, 8)$, with the uncapped level-only control. The revised rule requires lower terminal group RMS on every benchmark and at most a 10% increase at each earlier progress. On each benchmark, B must not decrease, the absolute A–B accuracy gap must not increase, and G may decrease by at most five percentage points. Across the pooled seed-specific pairs, A and G may decrease by at most five and one percentage points, respectively, and the total number of correct A+B pairs must not decrease. This rule retains $(4, 1, 8)$ and is frozen before evaluating that candidate on the test split. It does not select solely for the lowest RMS: the selected configuration's mean normalized terminal RMS is 0.383, whereas the tighter-cap candidate reaches 0.199 with ranking regressions.

Development sample sizes. Two disjoint problem pools feed the development comparisons. The terminal calibration sets, used by the progress and terminal gap losses, contain 87, 13, 6, and 111 unique questions for GSM1k, MATH-P-Simple, MATH-P-Hard, and LiveAoPS, with eligible unique-problem counts 55/55/55, 13/13/12, 6/6/6, and 97/95/94 at 25/50/75% progress. The protection loss instead draws on a separate A/B/G ranking pool: per seed it selects 36, 16, and 12 questions for the first three benchmarks, giving 9/9/18, 4/4/8, and 3/3/6 pairs, and 137 or 144 questions for LiveAoPS, giving 36/32/69 for seeds 17 and 29 and 41/32/71 for seed 43. Each selected question contributes exactly one pair to one condition, so the two pools need not agree in size. Per-benchmark accuracies average the three seeds; the pooled selection checks sum correct and total pair counts. Repeated pairs across seeds are not additional independent questions.

Transfer and checkpoint policy. LLaMA and Skywork reuse the Qwen-selected objective coefficients, floor, cap, and temperature with their own PRM rendering and training data. The matched component ablations use fixed final checkpoints: 786 updates for Qwen and Skywork, and 1,137 for LLaMA. These comparisons do not select intermediate checkpoints or retune the objective for each test benchmark.

Table 8: Hyperparameter selection for Qwen-Math on the development set. Five candidate configurations and the uncapped level-only control they replace, all evaluated on the Qwen development split with seeds 17, 29, and 43. Residual gap is the mean terminal group-RMS ratio to the control (1.00 is the control; lower is better). Ranking accuracies A, B, and G are percentages, aggregated over benchmarks by pair count. All candidates substantially reduce terminal group RMS, with decreases in A or G relative to the control. Bold marks the configuration retained under the revised selection rule; gap reduction alone does not identify it, since the $\gamma = 2$ candidate reaches the lowest ratio while its G falls 8.3 points on MATH-P-Simple and 6.7 points on LiveAoPS.

Configuration			Residual gap	Ranking accuracy (%)		
γ	λ_{drift}	λ_{protect}	($\times$ control)	A	B	G
uncapped level-only control			1.000	94.5	50.0	93.1
4	0	1	0.315	88.3	56.9	92.4
4	1	1	0.328	88.9	56.2	92.4
2	1	1	0.199	88.3	56.9	88.2
4	1	4	0.352	89.5	54.9	92.8
4	**1**	**8**	**0.383**	**89.5**	**55.6**	**92.8**

E.3 Group Averaging and Gradient Estimation

We use the group means μ_c and problem-balanced weights defined in Appendix B.

Drift groups use the same problem-balanced averaging and require distinct natural prefix markers at both endpoints.

The group weights ω_c split mass equally across progress points, then benchmark families, then benchmarks within each family, and then other models within each benchmark. The drift weights ω_h follow the same hierarchy with adjacent intervals in place of progress points. Each set of group weights sums to one.

L_{protect} gives equal total weight to A, B, and G. Within each condition K, $\mathbb{E}_K$ averages equally over benchmark families, then benchmarks, then the selected problems within each benchmark. Each problem contributes one frozen terminal pair assigned to one condition. For G, fixed seeds select eligible correct and incorrect trajectories without restricting their source models; A and B retain their source constraints.

We evaluate Acc by Equation 7 for each benchmark and seed, then average across seeds.

The calibration losses square full group means, whereas each training update uses minibatches. Let $\overline{\text{gap}}_c$ be the mean of samples drawn with replacement from group c, with sampling probabilities proportional to w_j. It is an unbiased estimate of μ_c, but its square satisfies

$$\mathbb{E}\big[\overline{\text{gap}}_c^2\big] = \mu_c^2 + \text{Var}(\overline{\text{gap}}_c). \tag{19}$$

At fixed adapter parameters, squaring one minibatch mean would therefore penalize pairwise variation as well as the group mean. We instead draw two independent minibatches with replacement using the same weights. Stopping gradient propagation through the first mean gives

$$\mathbb{E}\Big[2\,\text{stopgrad}\Big(\overline{\text{gap}}_c^{(1)}\Big)\,\nabla_\phi \overline{\text{gap}}_c^{(2)}\Big] = 2\mu_c \nabla_\phi \mu_c = \nabla_\phi \mu_c^2. \tag{20}$$

Independence makes this an unbiased gradient estimate of the squared group mean. Sampling groups according to ω_c gives the gradient of L_{correct}. The drift loss uses the same estimator with $\text{gap}_{p_{k+1}} - \text{gap}_{p_k}$ in place of gap_p and group sampling weights ω_h; together the correct-pair and adjacent-step losses give the gradient of L_{process}.

F Baseline Construction

The baselines compare correctness-only adaptation (BT-LoRA), familiarity correction (PPL-Cal, motivated by Wataoka et al. 2025), and source invariance with correctness protection (SafeScore-DIR, adapting Li et al. 2026b). The latter two adapt these principles to scalar PRM scores and the same frozen trajectories used by SPARC.

Shared notation and adaptation capacity. Let $s_0(x, y^p)$ and $s_\phi(x, y^p)$ be the original and adapted PRM scores at progress $p \in \{0.25, 0.50, 0.75, 1\}$. The Original PRM uses s_0 without fitting. BT-LoRA and SafeScore-DIR update rank-8 query/value LoRA adapters with scaling 8 and zero dropout; the pretrained parameters and reward head remain frozen. PPL-Cal instead fits scalar correction coefficients and leaves all neural parameters unchanged. The methods use the same problem partitions and native score interfaces. Training uses existing trajectories and their verified final-answer labels.

F.1 BT-LoRA

Terminal pairs. BT-LoRA uses the same selected training problems and pair budget as the source-balanced manifest. For each problem it draws a verified correct–incorrect terminal pair (x_j, y_j^+, y_j^-) without prescribing trajectory sources. Pair selection is fixed separately for seeds 17, 29, and 43. An epoch covers the selected pairs once, using four terminal pairs per optimizer update. Unlike the source-balanced manifest, which supplies one A pair, one B pair, and two G pairs per update, BT-LoRA treats all four as generic G comparisons.

Ranking objective. Let $\nu_j(\phi) = s_\phi(x_j, y_j^+) - s_\phi(x_j, y_j^-)$ and $P_\phi(y_j^+ \succ y_j^- \mid x_j) = \sigma(\nu_j(\phi))$, where $\sigma(z) = (1 + e^{-z})^{-1}$ is the Bradley–Terry preference model (Zhang et al., 2025a). For Qwen and Skywork, the weighted ranking objective is

$$J_{\text{BT}}(\phi) = 8 \sum_j w_j \log(1 + e^{-\nu_j(\phi)}) + 0.01\mathcal{R}(\phi), \qquad \sum_j w_j = 1. \tag{21}$$

The weights balance benchmark families, benchmarks within a family, and selected pairs within a benchmark. The effective-weight regularizer $\mathcal{R}$ is defined in Equation 18. The LLaMA implementation retains an original-margin protection objective on these same generic terminal pairs:

$$J_{\mathrm{BT}}^{\mathrm{L}}(\phi) = \sum_j w_j \,\mathrm{sp}_t(\rho + \mathrm{sp}_t(\nu_{0,j} - \rho) - \nu_j(\phi))^2 + \lambda_u \mathcal{R}(\phi), \tag{22}$$

where $\nu_{0,j}$ is the original margin, $\mathrm{sp}_t(v) = t\log(1 + e^{v/t})$, $\rho = 0.5$, $t = 0.25$, and $\lambda_u = 0.01$. All three BT-LoRA implementations omit correct–correct source calibration and adjacent-progress drift.

F.2 PPL-CAL

Familiarity signal. For a prefix y^p ending at a natural step boundary, let $\mathcal{T}_p$ index its response-content tokens. Using the frozen evaluator's retained causal language-model head, we compute

$$q_p(x, y) = -\frac{1}{|\mathcal{T}_p|} \sum_{t \in \mathcal{T}_p} \log P_0(y_t \mid x, y_{<t}), \qquad \mathrm{PPL}_p(x, y) = \exp(q_p(x, y)). \tag{23}$$

The native renderer supplies context. NLL averages response-content tokens through the entire prefix, excluding prompt tokens and scoring markers.

Training-only regression. Let $\Delta s_{i,p}$ and $\Delta q_{i,p}$ be the own-base minus other-model score and NLL differences for same-problem correct–correct pair i. A separate regression is fitted at each of the four progress points:

$$(\widehat{a}_p, \widehat{b}_p) = \arg\min_{a,b} \sum_{i \in \mathcal{D}_{\mathrm{CC},p}^{\mathrm{train}}} w_{i,p}(\Delta s_{i,p} - a - b\Delta q_{i,p})^2. \tag{24}$$

Weights sum to one within each progress point. Each benchmark family receives equal mass, divided equally among its benchmarks, then external sources, then pair memberships in each source cell. We solve these weighted least-squares regressions in closed form.

Deployment. The calibrated score for any trajectory is

$$s_{\mathrm{PPL}}(x, y^p) = s_0(x, y^p) - \widehat{b}_p q_p(x, y). \tag{25}$$

The intercept describes residual pairwise source differences and is not subtracted from individual scores. Inference uses the same rule for every source, with no source label or reference trajectory. At completion, the correction uses $\widehat{b}_1$; all coefficients are fitted only on training pairs.

F.3 SAFESCORE-DIR

SAFESCORE-DIR combines a scalar source adversary with original-margin protection and projection of conflicting gradients. Qwen and Skywork use conditional CLUB; LLaMA uses cross-entropy reversal, as specified below.

Source direction and classifier fitting. For a calibration cell c defined by benchmark, external source, and progress, let $\delta_{i,c} = s_\phi(x_i, y_{i,\mathrm{base}}^{+,p}) - s_\phi(x_i, y_{i,\mathrm{other}}^{+,p})$. We balance the left/right source orientation in every sampled cell:

$$b_i \in \{0, 1\}, \qquad z_i = (2b_i - 1)\delta_{i,c}, \qquad P(b_i = 0 \mid c) = P(b_i = 1 \mid c) = \tfrac{1}{2}. \tag{26}$$

Here $b_i = 1$ places the own-base trajectory on the left. A classifier q_ψ predicts this direction from z_i. Its cross-entropy loss is

$$L_{\mathrm{clf}}(\psi; \phi) = -\frac{1}{n} \sum_i \log q_\psi(b_i \mid \mathrm{stopgrad}(z_i), c_i). \tag{27}$$

The classifier takes five updates with the PRM scores detached, then remains fixed while the adapter receives the source-removal gradient.

Conditional CLUB for Qwen and Skywork. For a sampled cell with n_c observations, the categorical CLUB estimate (Cheng et al., 2020) is

$$\widehat{I}_c = \frac{1}{n_c}\sum_i \log q_\psi(b_i \mid z_i, c) - \frac{1}{n_c^2}\sum_{i,j} \log q_\psi(b_j \mid z_i, c), \qquad L_{\mathrm{src}} = \frac{1}{|\mathcal{C}_B|}\sum_{c\in\mathcal{C}_B} \widehat{I}_c. \tag{28}$$

The second term breaks the correspondence between a score difference and its orientation label within the same cell. Cells are sampled with the training macro weights, and the adapter minimizes L_{src} with ψ fixed. The classifier concatenates the scalar difference with a learned 16-dimensional cell embedding and uses a 128-unit ReLU hidden layer.

Original-margin protection. For a terminal pair, let ν_0 be its original correct–incorrect margin and ν_ϕ its adapted margin. Define $\mathrm{sp}_t(v) = t\log(1 + e^{v/t})$. Qwen and Skywork use the following smooth floor and cap:

$$u(\nu_0) = \rho + \mathrm{sp}_t(\nu_0 - \rho), \qquad T_{\mathrm{SD}}(\nu_0) = u(\nu_0) - \mathrm{sp}_t(u(\nu_0) - C), \tag{29}$$

with $\rho = 0.5$, $C = 2$, and $t = 0.25$. As $t \to 0$, this approaches $\min\{C, \max\{\rho, \nu_0\}\}$. This nested smoothing defines the baseline target, which differs from SPARC's target in Equation 5. For each terminal condition $K \in \{A, B, G\}$, the protection loss is

$$L_K = \mathbb{E}_K\left[\mathrm{sp}_t(T_{\mathrm{SD}}(\nu_0) - \nu_\phi)^2\right]. \tag{30}$$

Conflict projection. For Qwen and Skywork, let $g_{\mathrm{src}} = 8\nabla_\phi L_{\mathrm{src}}$ and $g_K = \frac{8}{3}\nabla_\phi L_K$. A condition is active when at least one sampled pair has $\nu_\phi < T_{\mathrm{SD}}(\nu_0)$. Starting with $v = g_{\mathrm{src}}$, we cycle over active conditions and apply

$$v \leftarrow v - \frac{\min\{0, \langle v, g_K\rangle\}}{\|g_K\|_2^2} g_K \quad (\|g_K\|_2 > 0). \tag{31}$$

Projection removes source-gradient components that increase protection loss to first order. We cycle until the inner-product checks pass within numerical tolerance, for at most 32 cycles. AdamW receives

$$g_{\mathrm{update}} = \sum_{K\in\{A,B,G\}} g_K + r_k v + \lambda_u \nabla_\phi \mathcal{R}(\phi), \qquad r_k = \min\{1, k/80\}, \tag{32}$$

where k starts at one and the source ramp lasts 80 updates.

LLaMA implementation. The LLaMA scorer uses the same balanced scalar differences with a 256-unit classifier that does not receive a cell embedding. Its source objective uses cross-entropy gradient reversal (Ganin et al., 2016):

$$L^{\mathrm{L}}_{\mathrm{clf}} = -\frac{1}{n}\sum_i \log q_\psi(b_i \mid z_i), \qquad g^{\mathrm{L}}_{\mathrm{src}} = -\nabla_\phi L^{\mathrm{L}}_{\mathrm{clf}}. \tag{33}$$

The classifier minimizes this cross-entropy loss on detached scores, whereas the adapter maximizes it. Its margin target is the uncapped smooth floor $T_{\mathrm{L}}(\nu_0) = \rho + \mathrm{sp}_t(\nu_0 - \rho)$, using the same $\rho = 0.5$ and $t = 0.25$. Let L^{L}_K denote Equation 30 with this target and set $g^{\mathrm{L}}_{\mathrm{guard}} = \nabla_\phi \frac{1}{3}\sum_K L^{\mathrm{L}}_K$. LLaMA applies Equation 31 once with $v = g^{\mathrm{L}}_{\mathrm{src}}$ and $g_K = g^{\mathrm{L}}_{\mathrm{guard}}$, giving $\widetilde{g}^{\mathrm{L}}_{\mathrm{src}}$. Its update is

$$g^{\mathrm{L}}_{\mathrm{update}} = g^{\mathrm{L}}_{\mathrm{guard}} + \widetilde{g}^{\mathrm{L}}_{\mathrm{src}} + \lambda_u \nabla_\phi \mathcal{R}(\phi). \tag{34}$$

A zero protection gradient leaves the source gradient unchanged; the numerical denominator is floored at 10^{-12}. This implementation uses no source ramp.

Optimization settings. The adapter optimizer is AdamW with learning rate 5×10^{-5}, BF16 arithmetic, gradient-norm clipping at 1, and $\lambda_u = 0.01$. Training spans three epochs for seeds 17, 29, and 43, corresponding to 786 updates for Qwen and Skywork and 1,137 for LLaMA. Source classifiers use AdamW at 10^{-4} with five updates per adapter step. The Qwen and Skywork source and protection coefficients are both 8; the LLaMA coefficients are both 1. SAFESCORE-DIR uses no SPARC group-mean calibration or drift loss. At inference the auxiliary classifier is discarded, leaving a single adapted PRM with its native score interface.

Reported checkpoints. The Qwen and LLaMA SAFESCORE-DIR evaluations use development-selected checkpoints at updates 131 and 379, respectively. Skywork uses update 786. Selection uses development calibration and terminal ranking statistics; these reported checkpoints can precede the final training update.

G COMPONENT ABLATIONS

We report all four progress points for the variants defined in Section 4.4. Within each PRM, the adapter capacity, seeds, update budget, and evaluation comparisons are fixed. Original PRM is the unadapted reference. The control without step calibration retains terminal calibration only and drops drift; the process-loss control removes both calibration terms. The remaining controls remove drift, terminal protection, or the margin cap individually. Qwen also replaces the squared group mean by the mean squared pairwise gap.

The progress profiles distinguish terminal parity from calibration throughout reasoning. A/B/G accuracies show whether reduced gaps retain correctness discrimination, particularly when protection is removed. The mean-squared-gap control compares matching individual scores with removing systematic group differences.

Table 9 reports one row per variant and aggregates the four benchmarks. Let b index GSM1k, MATH-P-Simple, MATH-P-Hard, and LiveAoPS, and $p \in \{25, 50, 75, 100\}$ the progress points. Two weight vectors control the aggregation. Progress gaps and RMS use w_b^{gap}, the number of calibration questions behind the progress metric (184, 26, 8, and 197 for Qwen and Skywork; 192, 19, 5, and 96 for LLaMA). Ranking accuracies use w_b^{acc}, the number of questions contributing to A, B, or G, counted once per question across conditions and seeds (87, 19, 20, and 361 for Qwen; 119, 37, 29, and 411 for LLaMA; 87, 19, 20, and 372 for Skywork). Both weight vectors are evaluation-side counts and are unrelated to the development pool sizes given in Appendix E.2: the two sets coincide only for the GSM1k entry, which is 87 in both. Gaps are combined in log space; RMS and ranking accuracies are combined as weighted arithmetic means:

$$\mathrm{GAP}_p = \exp\left(\frac{\sum_b w_b^{\mathrm{gap}} \log \mathrm{GAP}_{b,p}}{\sum_b w_b^{\mathrm{gap}}}\right), \qquad \mathrm{RMS} = \frac{\sum_b w_b^{\mathrm{gap}} \mathrm{RMS}_b}{\sum_b w_b^{\mathrm{gap}}}, \qquad \bar{A} = \frac{\sum_b w_b^{\mathrm{acc}} A_b}{\sum_b w_b^{\mathrm{acc}}}, \tag{35}$$

with the same weighting for B and G, and the final column is their mean, $\bar{\mathrm{Acc}} = \frac{1}{3}(\bar{A} + \bar{B} + \bar{G})$. The same aggregation produces the coordinates of Figure 7. Because $\bar{A}$, $\bar{B}$, $\bar{G}$, and $\bar{\mathrm{Acc}}$ average per-benchmark macro-accuracies, they remain accuracies in percent rather than rates over $\sum_b w_b^{\mathrm{acc}}$.

Table 9: Capped-drift component ablations. Each row aggregates the four benchmarks as defined in Equation 35. GAP (×) is at 25/50/75/100% progress, RMS is terminal group RMS, and A/B/G are terminal ranking accuracies (%) for the three conditions in Equation 4; Acc is their mean. Adapted rows use three seeds and fixed final checkpoints. GAP closest to 1× is preferred.

PRM		GAP (×)					Ranking (%)			
PRM	Variant	25%	50%	75%	100%	RMS	A	B	G	Acc
Qwen	Original PRM	6.46	9.62	13.06	52.14	4.336	96.4	25.9	89.9	70.71
	Full SPARC	1.30	1.44	1.56	1.58	0.567	93.8	48.4	91.4	77.88
	w/o drift loss	1.14	1.27	1.30	1.63	0.651	93.8	48.0	91.4	77.75
	w/o process loss	2.75	3.76	4.12	4.01	1.524	94.9	45.9	90.6	77.13
	w/o cap	2.47	3.03	3.82	6.90	2.184	95.6	41.1	92.0	76.23
	w/o step calib.	1.94	2.49	2.62	1.17	0.404	93.0	49.9	90.7	77.87
	w/ mean squared gap	1.28	1.40	1.50	1.70	0.619	93.8	45.5	92.3	77.21
	w/o protect loss	1.01	1.01	1.07	0.89	0.267	78.5	52.3	75.0	68.61
LLaMA	Original PRM	1.37	1.63	1.98	1.83	1.953	78.7	48.5	66.6	64.59
	Full SPARC	1.19	1.10	1.25	1.16	1.557	80.4	69.5	81.6	77.16
	w/o drift loss	1.19	1.14	1.34	1.43	1.533	81.3	68.8	81.9	77.31
	w/o process loss	1.51	1.44	1.77	1.39	1.728	78.4	70.0	83.5	77.30
	w/o cap	1.21	1.11	1.31	1.67	1.993	80.4	66.6	80.3	75.74
	w/o step calib.	1.48	1.49	1.78	1.46	1.535	78.8	68.8	81.5	76.36
	w/o protect loss	0.86	0.94	0.94	1.29	0.866	62.8	55.8	66.3	61.65
Skywork	Original PRM	2.12	1.99	2.89	4.90	1.687	89.5	65.2	92.4	82.38
	Full SPARC	1.45	1.27	1.45	1.57	0.502	89.1	70.6	91.9	83.89
	w/o step calib.	1.83	1.66	2.21	1.21	0.362	87.4	71.4	91.6	83.46
	w/o drift loss	1.40	1.22	1.40	1.58	0.544	88.4	70.2	91.9	83.52
	w/o process loss	2.02	1.89	2.71	3.41	1.316	89.5	68.7	92.6	83.62
	w/o cap	1.48	1.26	1.46	1.80	0.695	88.1	69.5	91.5	83.03
	w/o protect loss	1.06	0.94	0.96	0.97	0.121	78.8	69.0	85.9	77.87

G.1 Per-Benchmark Progress Gaps

Equation 35 combines the four benchmarks into the single values of Table 9. Table 10 gives the underlying per-benchmark gaps for the Original PRM and Full SPARC. Its twelve rows are the 48 cells that Section 4.2 averages into the headline mean-absolute-log-gap reduction.

Table 10: Progress-resolved terminal gap (×) for the Original PRM and Full SPARC, per benchmark. Each cell shows Original → Full. These are the 48 cells behind the reduction reported in Section 4.2, which is computed within each PRM before the three PRM-level reductions are averaged; the single-row values of Table 9 instead combine benchmarks with the weights of Equation 35. Lower is closer to parity. Rows follow the benchmark order of Table 1.

PRM	Benchmark	25%	50%	75%	100%
Qwen	GSM1k	3.82 → 0.99	5.59 → 1.18	8.10 → 1.32	8.69 → 1.02
	MATH-P-Simple	6.43 → 1.07	15.12 → 1.32	20.08 → 2.05	162.96 → 2.24
	MATH-P-Hard	4.19 → 1.11	43.92 → 3.09	79.36 → 1.92	3525.26 → 4.33
	LiveAoPS	10.73 → 1.73	14.16 → 1.70	17.92 → 1.74	201.52 → 2.18
LLaMA	GSM1k	0.85 → 1.10	1.16 → 1.12	1.34 → 1.27	1.44 → 1.25
	MATH-P-Simple	5.09 → 1.53	3.35 → 0.97	3.01 → 0.91	7.53 → 1.20
	MATH-P-Hard	3.42 → 1.73	2.50 → 0.88	4.45 → 0.86	2.51 → 0.88
	LiveAoPS	2.62 → 1.29	2.72 → 1.11	3.80 → 1.33	2.19 → 1.00
Skywork	GSM1k	1.33 → 1.04	1.54 → 1.09	1.51 → 0.93	4.68 → 1.21
	MATH-P-Simple	2.37 → 1.42	2.70 → 1.39	4.16 → 1.72	6.66 → 2.27
	MATH-P-Hard	9.24 → 4.30	10.40 → 4.20	40.74 → 7.44	9.32 → 3.35
	LiveAoPS	3.03 → 1.90	2.27 → 1.37	4.54 → 2.00	4.79 → 1.86

H Analysis

We evaluate whether Qwen calibration generalizes to an other-model source or a benchmark family excluded from training. Both experiments use the source-balanced construction.

H.1 Generalization to an Unseen Other Model

The held-out experiment removes the selected other model from every calibration group, terminal A/B/G relation, and training trajectory set. The final capped-drift variant is retrained from the remaining sources, and trajectories from the held-out model are used only to evaluate the frozen checkpoint on the test split. Table 11 reports GAP and group RMS at all four progress points, together with terminal Acc. Group RMS decreases, and all twelve progress-wise gaps move toward parity, so the calibration extends to a source that never entered training. On MATH-P the terminal Acc instead falls from 72.2 to 63.9. That comparison is the smallest of the three, and the drop traces to a single A pair: MATH-P has four A pairs at completion, against 9 for GSM1k and 33 for LiveAoPS. We therefore read it as sampling variation at this pair count rather than as a failure of calibration on the held-out source.

Table 11: Qwen generalization to the excluded LLaMA generator: progress-resolved GAP, group RMS, and terminal ranking. Each cell shows Original → Full SPARC; intervals are paired 95% intervals for the change in group RMS. MATH-P-Simple and MATH-P-Hard are pooled as in the main table; pooled group-RMS intervals combine the two memberships. Mean Acc averages the A, B, and G relations at completion, so it is reported once per benchmark. MATH-P-Hard has no B pair containing the held-out source, so the pooled B relation draws on MATH-P-Simple only.

Benchmark	Progress	n	GAP	Group RMS	Δ RMS [95% CI]	Mean Acc (%)
GSM1k	25%	75	3.575 → 1.215	1.274 → 0.198	-1.076 [-1.396, -0.607]	75.0 → 77.8
	50%	75	7.090 → 1.579	1.959 → 0.458	-1.501 [-1.824, -1.165]	
	75%	75	7.290 → 1.392	1.986 → 0.333	-1.653 [-2.003, -1.304]	
	100%	157	6.566 → 1.333	1.882 → 0.292	-1.590 [-1.892, -1.295]	
MATH-P Simple&Hard	25%	19	4.112 → 0.935	1.419 → 0.133	-1.286 [-2.198, -0.708]	72.2 → 63.9
	50%	19	16.608 → 1.921	3.100 → 1.017	-2.083 [-2.918, -1.271]	
	75%	19	7.381 → 1.357	2.002 → 0.421	-1.581 [-2.297, -0.849]	
	100%	22	49.343 → 1.819	4.011 → 0.627	-3.384 [-4.845, -1.858]	
LiveAoPS	25%	67	5.982 → 1.513	1.789 → 0.418	-1.371 [-1.751, -0.960]	61.1 → 63.5
	50%	65	6.020 → 1.339	1.795 → 0.297	-1.499 [-1.942, -0.958]	
	75%	64	16.432 → 1.801	2.799 → 0.595	-2.204 [-2.708, -1.681]	
	100%	77	80.355 → 2.400	4.386 → 0.878	-3.508 [-4.046, -3.008]	

H.2 Generalization to Excluded Benchmark Families

We train three Qwen leave-one-family-out folds, omitting GSM1k, MATH-P (Simple and Hard together), or LiveAoPS from the calibration and terminal training comparisons. Each fold uses seeds 17, 29, and 43 and the original held-out test split. Table 12 reports GAP and group RMS at all four progress points, together with terminal Acc. The paired intervals for the RMS change lie below zero at every benchmark and progress point, and all twelve progress-wise gaps move toward parity. Acc rises from 77.4 to 79.8 on GSM1k, 71.3 to 80.6 on pooled MATH-P, and 68.8 to 71.8 on LiveAoPS. The pooled MATH-P comparison draws on 34 eligible test problems at completion, which limits the problem coverage of that comparison.

Table 12: Qwen calibration on excluded benchmark families: progress-resolved GAP, group RMS, and terminal ranking. Each fold omits one family from training and is evaluated on that family; MATH-P-Simple and MATH-P-Hard are pooled as in the main table under the fold that omits MATH-P. Each cell shows Original → Full SPARC; intervals are paired 95% intervals for the change in group RMS. Mean Acc averages the A, B, and G relations at completion. n counts eligible problems at that progress point.

Benchmark	Progress	n	GAP	Group RMS	Δ RMS [95% CI]	Mean Acc (%)
GSM1k	25%	129	3.816 → 1.116	1.341 → 0.150	-1.190 [-1.427, -0.787]	77.4 → 79.8
	50%	129	5.590 → 1.331	1.737 → 0.299	-1.438 [-1.713, -1.133]	
	75%	128	8.105 → 1.569	2.116 → 0.479	-1.637 [-1.941, -1.320]	
	100%	184	8.694 → 1.199	2.668 → 0.251	-2.417 [-2.639, -2.179]	
MATH-P Simple&Hard	25%	30	5.778 → 1.269	1.993 → 0.622	-1.372 [-2.275, -0.293]	71.3 → 80.6
	50%	30	19.742 → 2.104	3.263 → 1.127	-2.136 [-3.009, -1.494]	
	75%	30	28.314 → 2.543	3.660 → 1.088	-2.572 [-3.291, -1.001]	
	100%	34	327.112 → 3.258	6.475 → 1.526	-4.948 [-5.927, -3.600]	
LiveAoPS	25%	176	10.728 → 4.259	2.469 → 1.532	-0.936 [-1.122, -0.764]	68.8 → 71.8
	50%	172	14.158 → 5.010	3.057 → 1.973	-1.084 [-1.248, -0.794]	
	75%	166	17.917 → 5.207	3.277 → 2.064	-1.213 [-1.535, -1.020]	
	100%	197	201.521 → 9.091	5.540 → 2.277	-3.264 [-3.666, -2.857]	

I LIMITATIONS

Despite the improvements reported above, limitations remain; neither affects the matched comparisons, which hold the problem, the correctness label, and the progress level fixed. First, like most prior work in this area, we evaluate primarily on mathematical reasoning; extending the analysis to code or open-ended writing would be valuable. Second, our calibration acts on the score gap between sources rather than on the content of a response, so a difference driven purely by surface style would be absorbed into the calibrated term rather than separated from it. Formalizing such style-induced score differences, and proving their removal, would require an explicit model of response style; we leave this as future work.