THE MODEL IS ITS OWN CURRICULUM: INTERNAL DIFFICULTY ASSESSMENT FOR SELF-ADAPTIVE RL IN REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning can substantially improve language model reasoning, but sparse outcome rewards provide limited guidance for exploring complex reasoning trajectories. Curriculum learning offers a natural way to structure this exploration by adapting difficulty to the evolving policy. A key question, however, is at what granularity such learner-relative difficulty should be defined. While empirical pass rates provide a dynamic estimate of problem difficulty, the current policy's likelihood can further distinguish among correct reasoning trajectories for the same problem. We propose Self-Curriculum, which uses the negative length-normalized log-likelihood of correct trajectories as a learner-relative measure of trajectory difficulty. Self-Curriculum orders verified trajectories from easy to hard, periodically updates their difficulty as the policy evolves, and incorporates this signal into reward shaping. Across multiple mathematical reasoning benchmarks and model families, Self-Curriculum consistently outperforms standard GRPO. On Qwen3-8B, a curriculum based on dynamic pass rates already improves over GRPO, while Self-Curriculum yields further gains and also outperforms the evaluated external difficulty baselines at both the problem and trajectory levels. These results support policy likelihood as a fine-grained, learner-relative signal for adaptive curriculum learning under outcome supervision.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable reasoning capabilities (Chen et al., 2025; Ferrag et al., 2025; Li et al., 2025b; Wang et al., 2026; 2022), yet training them to reason via reinforcement learning (RL) faces a structural bottleneck: reward sparsity (Müller et al., 2025; Zhong et al., 2026). Without guidance, discovering complex, multi-step reasoning chains through stochastic exploration can be highly inefficient. Curriculum learning offers a natural way to organize this exploration process (Parashar et al., 2025; Rane, 2020; Soviany et al., 2022). Many recent reasoning curricula obtain difficulty from signals external to the learner, such as stronger models, correct demonstrations, or reasoning prefixes (Shi et al., 2025; Lotfi et al., 2026; Xi et al., 2024). Such signals can be useful, but they need not track the evolving state of the policy being optimized.

This motivates defining curriculum difficulty relative to the learner itself. A natural learner-relative signal is the model's empirical pass rate. Given multiple rollouts for a problem, the fraction of answer-correct trajectories directly estimates how solvable that problem is for the current policy and can be refreshed as the policy changes. However, pass rate remains a *problem-level* signal: every correct trajectory generated for the same problem receives the same difficulty, even when the current policy assigns very different probabilities to different valid reasoning paths. Thus, making difficulty learner-relative does not by itself resolve the question of what granularity is most useful for curriculum learning.

We therefore ask a more specific question: *does the learner's own trajectory likelihood provide useful curriculum information beyond problem-level success rate?* Consider two answer-correct trajectories for the same problem. One may consist largely of high-probability reasoning steps under the current policy, while another may traverse a substantially lower-probability path. Although both trajectories have identical outcome correctness and inherit the same problem-level pass rate, they occupy different regions of the learner's current probability landscape. We use negative length-normalized trajectory

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.

log-likelihood as an operational, learner-relative proxy for this distinction. Importantly, we do not assume that low likelihood is an intrinsic or semantic measure of reasoning difficulty; rather, it measures how improbable a particular verified trajectory is under the current policy.

Building on this observation, we propose Self-Curriculum, a learner-adaptive curriculum framework with three steps. (1) The model explores by sampling trajectories and retains problems with at least one answer-correct trajectory. (2) Each answer-correct trajectory is assigned a learner-relative difficulty $d(\tau) = -\ell_{\text{norm}}(\tau)$, where ℓ_{norm} is its length-normalized log-likelihood under the learner. These scores are normalized to $\hat{d}(\tau) \in [0, 1]$, and answer-correct trajectories are ordered from easier to harder according to the current policy, while incorrect trajectories from retained problems are preserved as negative samples. (3) A difficulty-aware reward introduces a relative preference among answer-correct trajectories with different learner-relative difficulty. Within each epoch, the likelihoods of remaining answer-correct trajectories are periodically rescored to refresh their ordering as the policy evolves. The difficulty signal requires no auxiliary teacher model or process-level difficulty annotation; final-answer supervision is used for correctness verification.

Our experiments isolate what makes the curriculum signal effective. A Dynamic Pass-Rate Curriculum shows that learner-adaptive problem-level pacing already improves over GRPO. On Qwen3-8B MATH, it improves GRPO from 65.59% to 68.46%, while Self-Curriculum reaches 71.29%, indicating additional value from trajectory-level likelihood. Controlled external-difficulty variants score the same student-generated training pool using a stronger teacher at either problem or trajectory granularity; both underperform the learner-relative signal. Together with component ablations, these comparisons separate the roles of learner adaptivity, trajectory granularity, and the source of the difficulty signal. Our contributions are as follows:

- **We separate learner adaptivity from trajectory granularity.** Success-Rate SPL and Dynamic Pass-Rate controls test whether learner-adaptive problem-level pacing is sufficient, while Self-Curriculum evaluates the additional value of trajectory-level likelihood.
- **We isolate the source of the difficulty signal.** Holding the student-generated and student-filtered training pool fixed, external problem and trajectory-level controls replace only the difficulty signal with estimates from a stronger teacher, enabling comparison with learner-relative difficulty.
- **We develop an adaptive trajectory-level curriculum for outcome-supervised reasoning.** Self-Curriculum combines verified self-generated trajectories, learner-relative ordering, periodic reranking, and difficulty-aware rewards without an auxiliary difficulty model or process-level difficulty annotations. It consistently improves over GRPO across benchmarks and model families.

2 Related Work

Curriculum and Self-Paced Learning for Reasoning. Curriculum learning organizes training from easier to harder examples to facilitate learning and exploration under sparse feedback (Parashar et al., 2025; Rane, 2020; Soviany et al., 2022). Recent curricula for LLM reasoning often derive difficulty from stronger models, correct demonstrations, or reasoning prefixes (Shi et al., 2025; Xi et al., 2024; Lotfi et al., 2026). Classical self-paced learning (SPL), in contrast, adapts the curriculum according to the learner's evolving state (Kumar et al., 2010). These approaches motivate learner-adaptive curricula, but difficulty is typically associated with training examples or problems rather than with multiple reasoning trajectories for the same problem. Our work focuses on this finer granularity: correct trajectories for the same problem can have different likelihoods under the current policy, providing a learner-relative signal for trajectory difficulty.

Learner-Adaptive Pacing in RLVR. Recent RLVR systems also adapt training according to the evolving policy. DAPO dynamically filters prompt groups based on their reward structure (Yu et al., 2025), GRESO predicts uninformative prompts before rollout (Zheng et al., 2025), and MoPPS models prompt success rates online to improve prompt selection (Qu et al., 2026). These methods adapt sampling or computation primarily at the prompt level. In contrast, Self-Curriculum characterizes learner-relative difficulty at the level of outcome-verified trajectories, allowing different correct solutions to the same problem to receive different curriculum scores.

Self-Generated Reasoning Data. Self-generated reasoning trajectories can reduce reliance on human-annotated rationales. STaR and rejection-sampling fine-tuning train on self-generated correct rationales (Zelikman et al., 2022), while related approaches improve verification, refinement, or

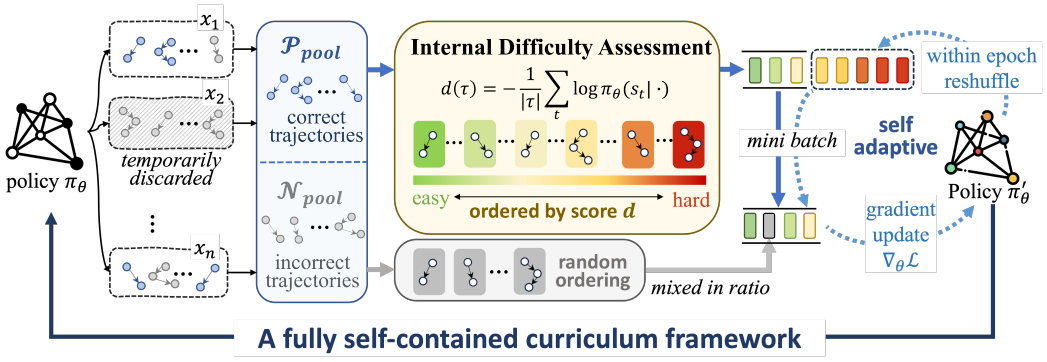

Figure 1: The Self-Curriculum pipeline. The model generates k trajectories per problem, assigns internal difficulty to correct ones, and sorts them into a queue. Positive samples are drawn in curriculum order, with periodic reshuffling to adapt to model progress. Negative samples are randomly sampled. Policy updates use precomputed advantages.

on-policy distillation of generated sequences (Hosseini et al., 2024; Dou et al., 2024; Agarwal et al., 2024). Our work builds on the use of outcome-verified self-generated trajectories, but focuses on how such trajectories can be organized according to their difficulty for the current learner.

Intrinsic Signals and Outcome-Supervised RL. Reasoning RL commonly optimizes outcome-supervised objectives such as PPO and GRPO (Schulman et al., 2017; Shao et al., 2024), while recent work explores model-derived signals such as confidence, self-certainty, and consensus as additional sources of supervision (Li et al., 2025a; Zhao et al., 2026; Zuo et al., 2025). Self-Curriculum retains final-answer outcome verification as the correctness signal and uses policy likelihood only to estimate learner-relative trajectory difficulty for curriculum pacing and weighting, rather than as a substitute for outcome verification.

3 Method

We present the Self-Curriculum framework (Figure 1) across the following subsections.

3.1 Problem Formulation

We are given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ of problems and their ground truth answers. No reasoning process is provided; only final answer correctness is observable. Our goal is to train a policy π_θ that generates correct reasoning chains that yield the correct answer.

3.2 Self-Generated Trajectories and Data Filtering

At the beginning of each training epoch, we use the current model π_θ to perform k independent samplings for each problem x_i in the dataset, obtaining k reasoning trajectories $\{\tau_i^{(1)}, \tau_i^{(2)}, \ldots, \tau_i^{(k)}\}$.

Based on the correctness of the final answer, we divide these trajectories into two groups:

- **Positive set** ($\mathcal{P}_i = \{\tau_i^{(j)} \mid \text{answer}(\tau_i^{(j)}) = y_i\}$): trajectories with correct answers.
- **Negative set** ($\mathcal{N}_i = \{\tau_i^{(j)} \mid \text{answer}(\tau_i^{(j)}) \neq y_i\}$): trajectories with incorrect answers.

If all k samplings for a problem x_i fail to produce a correct answer ($\mathcal{P}_i = \emptyset$), no answer-correct trajectory is observed for that problem in the current rollouts, and the problem is temporarily excluded from training in this epoch. This focuses updates on problems for which the current policy has produced at least one answer-correct trajectory. As training progresses, previously filtered problems may yield correct answers in subsequent epochs and thus naturally enter the training process. Let $\mathcal{P}_{\text{pool}} = \bigcup_{i:\mathcal{P}_i \neq \emptyset} \{(x_i, \tau) \mid \tau \in \mathcal{P}_i\}$ denote the collection of all positive trajectories from retained problems, and $\mathcal{N}_{\text{pool}} = \bigcup_{i:\mathcal{P}_i \neq \emptyset} \{(x_i, \tau) \mid \tau \in \mathcal{N}_i\}$ denote the collection of all negative trajectories from the same retained problems.

3.3 Adaptive Internal Difficulty Assessment

For each answer-correct trajectory $\tau_i^{(j)} = (s_1, s_2, \ldots, s_{|\tau_i^{(j)}|})$ in the positive set, we define its length-normalized log-likelihood under the current policy as

$$\ell_{\text{norm}}(\tau_i^{(j)}) = \frac{1}{|\tau_i^{(j)}|} \sum_{t=1}^{|\tau_i^{(j)}|} \log p_\theta(s_t \mid x_i, s_{<t}). \tag{1}$$

We use its negative as the internal trajectory difficulty:

$$d(\tau_i^{(j)}) = -\ell_{\text{norm}}(\tau_i^{(j)}). \tag{2}$$

This quantity corresponds to the average token-level surprisal of the trajectory under the current model. Among correct trajectories, a lower difficulty indicates that the reasoning path is more probable under the current policy, whereas a higher difficulty indicates a less familiar trajectory. We therefore use it as a model-internal proxy for trajectory difficulty.

The score also admits a limited operational interpretation. For a trajectory of length L, $\ell_{\text{norm}}(\tau) = L^{-1} \log p_\theta(\tau \mid x)$, and therefore $p_\theta(\tau \mid x) = \exp(-L\, d(\tau))$. Under independent sampling from the current policy, the expected number of rollouts required to reproduce the exact token trajectory is $1/p_\theta(\tau \mid x) = \exp(L\, d(\tau))$. Thus, for equal-length trajectories, $d(\tau)$ is monotonic in the expected exact-trajectory reproduction time under the current policy. This interpretation concerns learner-relative generation difficulty rather than intrinsic semantic reasoning complexity; for trajectories of different lengths, exact reproduction is governed by the total negative log-likelihood $L\, d(\tau)$.

As the model parameters are updated, ℓ_{norm} changes accordingly. We periodically refresh the ordering of unused trajectories within an epoch and fully regenerate trajectories between epochs, allowing the curriculum to track the evolving policy at two timescales.

To bound the difficulty-dependent reward scale within each epoch's positive sample pool $\mathcal{P}_{\text{pool}}$, we apply min-max normalization:

$$\hat{d}(\tau_i^{(j)}) = \frac{d(\tau_i^{(j)}) - d_{\min}}{d_{\max} - d_{\min} + \epsilon}, \tag{3}$$

where $d_{\min}$ and $d_{\max}$ are the minimum and maximum difficulty scores in $\mathcal{P}_{\text{pool}}$, and $\epsilon = 10^{-5}$ prevents division by zero. The normalized internal difficulty $\hat{d} \in [0, 1]$ is then used for curriculum ordering and reward computation.

3.4 Curriculum Sampling Strategy Based on Difficulty Ranking

The core of curriculum learning lies in presenting training samples to the model in a reasonable order. This section elaborates on how we construct the curriculum sequence and arrange training based on adaptive difficulty scores.

3.4.1 Definition of Training Units

In our framework, the basic training unit is the problem trajectory pair $(x_i, \tau_i^{(j)})$ rather than the problem itself. This means that different correct trajectories of the same problem x_i are treated as independent training samples, each with its own difficulty score. The motivation for this design is that the same problem may correspond to multiple answer-correct reasoning paths that receive substantially different probabilities under the current policy. We treat this probability difference as a learner-relative pacing signal: higher-probability trajectories are presented earlier and lower-probability trajectories later. This ordering should not be interpreted as a claim that lower-probability trajectories are intrinsically more complex or semantically more sophisticated.

3.4.2 Difficulty Ranking and Selection of Positive Samples

Positive samples are presented in internal difficulty ascending order, while negative samples are randomly interleaved at the natural pool ratio. Specifically, we compute the normalized difficulty $\hat{d}(\tau_i^{(j)})$ for each positive sample using Equation 3 and sort them from low to high difficulty, forming the curriculum queue $\mathcal{P}_{\text{sorted}}$. During training, we draw positive samples in this order, so that the model first learns easy patterns and progressively tackles harder ones. Negative samples are randomly sampled from $\mathcal{N}_{\text{pool}}$ and mixed with positive samples such that, within each training batch, the ratio of negative to positive samples matches the natural ratio observed in the global pools ($|\mathcal{N}_{\text{pool}}| : |\mathcal{P}_{\text{pool}}|$).

3.4.3 Dynamic Curriculum Update

As training proceeds, the model parameters evolve, requiring two levels of adaptation. First, within each epoch, we periodically recompute the difficulty of remaining unused samples in $\mathcal{P}_{\text{sorted}}$ and reshuffle them accordingly. This brings previously hard samples forward as they become easier and postpones those that become harder. Importantly, the recomputed difficulty is used only to update the ordering of the remaining curriculum queue. The difficulty-aware rewards and the corresponding advantages are computed once at the beginning of the epoch and remain fixed throughout that epoch.

Second, across epochs, we discard all trajectories from the previous epoch and resample every problem k times using the updated model. The trajectory pools, difficulty scores, rewards, and advantages are therefore recomputed at the beginning of each new epoch, allowing the training data and curriculum to adapt to the model's evolving capabilities.

3.5 Curriculum Reinforcement Learning Training

3.5.1 Difficulty-Aware Reward Design

We incorporate internal difficulty into the reward to introduce a relative preference among correct trajectories within the same problem group:

$$R(\tau_i^{(j)}) = \begin{cases} 1 + \alpha \cdot \hat{d}(\tau_i^{(j)}), & \text{if } \tau_i^{(j)} \text{ is correct,} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Here, α controls the strength of difficulty-based reward shaping. Rather than uniformly increasing the gradient magnitude of every hard trajectory, this term differentiates correct trajectories within the same comparison group: lower-likelihood correct trajectories receive higher rewards than easier correct alternatives. After group-relative normalization, this preference is most relevant when multiple correct trajectories with different difficulty levels occur within the same group.

3.5.2 Policy Optimization with Precomputed Advantages

We adopt the group-relative advantage computation of GRPO (Shao et al., 2024). For each problem x_i, its k sampled trajectories $\{\tau_i^{(1)}, \ldots, \tau_i^{(k)}\}$ form a comparison group. The advantage of each trajectory is computed by standardizing rewards within the group:

$$\hat{A}(\tau_i^{(j)}) = \frac{R(\tau_i^{(j)}) - \text{mean}\big(\{R(\tau_i^{(1)}), \ldots, R(\tau_i^{(k)})\}\big)}{\text{std}\big(\{R(\tau_i^{(1)}), \ldots, R(\tau_i^{(k)})\}\big) + \epsilon_A}, \tag{5}$$

where ϵ_A is a small constant for numerical stability. These advantages are computed once at the beginning of each epoch and remain fixed throughout that epoch. Subsequent difficulty reshuffling changes only the order in which trajectories are presented, not their rewards or precomputed advantages. After advantage computation, each pair $(x_i, \tau_i^{(j)})$ is stored together with its group-derived advantage $\hat{A}(\tau_i^{(j)})$ and treated as an independent optimization sample. Positive trajectories are inserted into the curriculum queue $\mathcal{P}_{\text{sorted}}$, while negative trajectories are drawn from $\mathcal{N}_{\text{pool}}$.

We optimize the following GRPO-style token-level clipped surrogate objective:

$$\mathcal{J}_{\text{clip}} = \frac{1}{|\mathcal{B}|} \sum_{(x,\tau)\in\mathcal{B}} \frac{1}{|\tau|} \sum_{t=1}^{|\tau|} \min\Big(\rho_t \hat{A}, \text{clip}(\rho_t, 1-\epsilon, 1+\epsilon)\hat{A}\Big), \tag{6}$$

where $\rho_t = \frac{\pi_\theta(s_t|x,s_{<t})}{\pi_{\text{old}}(s_t|x,s_{<t})}$ and π_{old} denotes the policy used to generate the trajectories at the beginning of the current epoch.

Standard GRPO computes group-relative advantages jointly from multiple responses sampled for the same prompt. We compute these advantages before curriculum batching and store them with the corresponding trajectories. This decouples subsequent optimization minibatch composition from the original response groups, allowing trajectories to be reordered according to the curriculum while retaining their group-derived advantages.

Table 1: Main results on GSM8K, MATH, and NuminaMath-CoT hard subsets.

Model	Method	GSM8K	MATH	AMC/AIME	Olympiads
Llama-3.1-8B-Instruct	Original	62.09	39.67	1.03	9.68
	GRPO	88.07	37.48	9.80	12.47
	RFT	82.19	40.35	13.74	11.73
	AdaBack	90.98	39.82	11.07	13.53
	Self-Curriculum	**93.17**	**42.79**	**17.21**	**16.55**
Qwen3-8B	Original	83.71	59.18	16.67	14.93
	GRPO	92.90	65.59	32.35	22.55
	RFT	90.33	66.17	27.63	23.89
	AdaBack	94.01	67.13	33.36	25.46
	Self-Curriculum	**96.72**	**71.29**	**38.27**	**27.42**

Table 2: Main results on other subsets of NuminaMath-CoT.

Model	Method	AoPS	CN K12	ORCA	Syn AMC	Syn Math
Llama-3.1-8B-Instruct	Original	8.22	27.26	63.71	17.71	47.94
	GRPO	10.96	30.93	64.99	20.14	48.45
	RFT	14.06	29.91	67.14	18.52	48.77
	AdaBack	13.21	31.09	64.71	19.42	49.48
	Self-Curriculum	**15.92**	**34.70**	**69.52**	**21.84**	**50.35**
Qwen3-8B	Original	3.01	45.73	75.08	17.27	55.31
	GRPO	15.07	51.92	80.39	23.74	61.60
	RFT	15.33	48.15	83.31	24.35	59.23
	AdaBack	17.81	53.85	81.23	27.34	61.86
	Self-Curriculum	**19.62**	**55.63**	**84.07**	**29.16**	**63.19**

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate on three widely used mathematical reasoning benchmarks: GSM8K (elementary school math) (Cobbe et al., 2021), MATH (high school competition level math) (Hendrycks et al.), and NuminaMath-CoT (covering AMC/AIME, AoPS forum, Chinese K12, Olympiad problems, and synthetic data) (Li et al., 2024).

Baselines and Controls. We compare Self-Curriculum with the original checkpoint, GRPO (Shao et al., 2024), rejection-sampling fine-tuning (RFT) (Yuan et al., 2023), and AdaBack (Lotfi et al., 2026). To disentangle the effects of learner adaptivity, signal granularity, and signal source, we additionally evaluate Success-Rate SPL, Dynamic Pass-Rate Curriculum, External Problem Difficulty, and External Trajectory Likelihood on Qwen3-8B in Section 4.3. Full dataset, baseline, and implementation details are provided in Appendix A.

4.2 Main Results

We evaluate all methods on Llama-3.1-8B-Instruct and Qwen3-8B. Tables 1 and 2 report the complete benchmark results.

Comparison with Baselines. Self-Curriculum achieves the strongest overall performance across both model families and all reported benchmark subsets. AdaBack is a particularly relevant curriculum baseline because it also attempts to control training difficulty, but it relies on externally provided correct reasoning prefixes. Self-Curriculum instead leaves the generated reasoning trajectory unchanged and organizes training according to the learner's own likelihood over answer-correct trajectories. The consistent advantage over AdaBack suggests that learner-relative trajectory difficulty can provide an effective curriculum signal without requiring external process supervision. RFT also uses self-generated correct trajectories, but optimizes them through supervised fine-tuning without explicit curriculum progression. Its lower overall performance indicates that simply collecting correct self-generated trajectories does not account for the gains of the full method.

Performance on Challenging Subsets. The improvements over GRPO are observed across both model families and remain especially clear on MATH and AMC/AIME, while gains are also maintained on Olympiad problems. This pattern is consistent with curriculum guidance being particularly useful when correct reasoning trajectories are harder to discover under sparse outcome supervision. Table 2 provides a broader test across AoPS, Chinese K12, ORCA, synthetic AMC, and synthetic mathematics subsets. Self-Curriculum remains the strongest method throughout these heterogeneous subsets, suggesting that the improvement is not driven by a single benchmark style or source.

4.3 Core Analysis: Which Learner-Relative Signal?

The main results establish that Self-Curriculum is effective, but they do not by themselves identify which property of the curriculum signal is responsible for the improvement. We therefore conduct targeted controls on Qwen3-8B with GSM8K and MATH. These experiments address two questions. First, is it sufficient to use a learner-adaptive signal at the problem level? Second, does trajectory-level likelihood provide additional information beyond problem-level solvability? Table 3 summarizes the results, with detailed implementations provided in Appendix A.2.

Table 3: Targeted curriculum and difficulty-signal controls on Qwen3-8B.

Method	GSM8K	MATH
Original	83.71	59.18
GRPO	92.90	65.59
Success-Rate SPL	94.17	66.94
Dynamic Pass-Rate	94.63	68.46
External Difficulty	93.47	65.31
External Traj. Likelihood	93.06	65.83
Self-Curriculum	**96.72**	**71.29**

Learner-Relative Problem-Level Controls. Success-Rate SPL uses the current policy's empirical problem-level success rate for easy-to-hard pacing with binary rewards, without trajectory-level likelihood or reward shaping. Dynamic Pass-Rate provides a closer control by using the current policy's pass rate as a dynamically refreshed problem-level difficulty shared by all answer-correct trajectories from the same problem. Both outperform standard GRPO, indicating that learner-adaptive problem-level pacing is useful on its own. Self-Curriculum further outperforms both, suggesting that trajectory-level learner-relative likelihood captures information beyond problem-level solvability.

External Problem and Trajectory Controls. To isolate the source of the difficulty signal, both external controls use the same student-generated and filtered trajectory pool as Self-Curriculum. External Problem Difficulty uses a stronger teacher's empirical problem-level success rate, whereas External Trajectory Likelihood scores the retained student trajectories using the teacher's length-normalized likelihood. Thus, these comparisons vary the difficulty signal without changing the training examples.

Both external controls underperform the learner-relative trajectory signal. Together with the problem-level controls above, this suggests that the gains depend on both trajectory-level granularity and alignment of the difficulty signal with the current learner. We therefore treat trajectory likelihood as an operational learner-relative pacing signal rather than an intrinsic measure of semantic difficulty.

4.4 Robustness, Efficiency, and Scaling

Robustness Across Independent Training Runs. The main results average repeated evaluations of fixed trained checkpoints over multiple decoding seeds, which reduces inference-time sampling variance but does not capture training variability. We therefore independently train Qwen3-8B with three different training seeds. As shown in Table 4, Self-Curriculum consistently outperforms GRPO and AdaBack on both GSM8K and MATH across independent training runs, with comparable variation across methods.

Table 4: Training-seed robustness on Qwen3-8B (mean ± standard deviation over three runs).

Method	GSM8K	MATH
Original	83.71	59.18
GRPO	92.76 ± 0.91	65.71 ± 0.62
AdaBack	93.84 ± 1.03	67.42 ± 0.43
Self-Curriculum	$\mathbf{96.58 \pm 0.76}$	$\mathbf{71.16 \pm 0.58}$

Computational Overhead and Budget Control. Self-Curriculum incurs 8% and 11% end-to-end training overhead on GSM8K and MATH, respectively, due to periodic trajectory rescoring. Increasing the GRPO group size from $k = 10$ to $k = 11$ yields only modest gains and does not close the gap to Self-Curriculum with $k = 10$. Full wall-clock, expanded-budget, and resampling results are provided in Appendix B; while not a strict equal-wall-clock comparison, these controls suggest that the gains are not simply explained by modest additional computation.

Table 5: Scaling results on Qwen3-14B across mathematical reasoning benchmarks.

Model	Method	GSM8K	MATH	AMC/AIME	Olympiads
Qwen3-14B	Original	84.53	65.06	20.51	19.17
	GRPO	94.24	68.90	38.72	23.87
	Self-Curriculum	**97.39**	**72.31**	**40.12**	**28.91**

Larger Model Scale. We additionally evaluate the method on Qwen3-14B to test whether the effect is specific to the 8B setting. As shown in Table 5, Self-Curriculum remains stronger than GRPO across the core GSM8K, MATH, AMC/AIME, and Olympiad evaluations. The remaining NuminaMath-CoT subsets show the same overall pattern and are reported in Appendix C. Comparing the evaluated 8B and 14B settings, the improvement remains clear at the larger scale, although the relative gap varies by benchmark. Because we evaluate only these model scales, we treat this experiment as evidence of transfer across scale rather than as a claim about a general scaling trend.

4.5 Adaptive Curriculum Dynamics

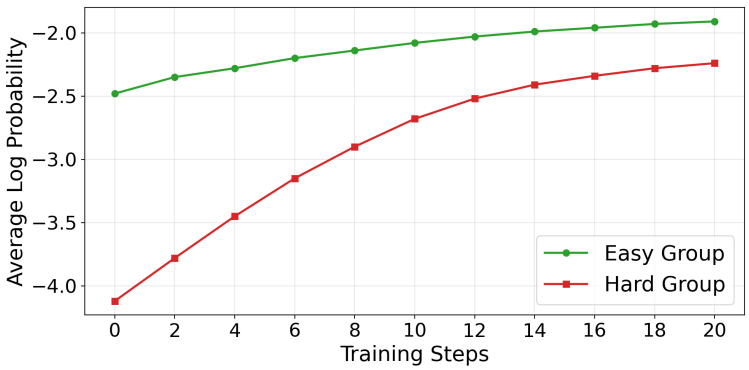

Figure 2: Evolution of fixed epoch-1-correct trajectories on Qwen3-8B/MATH.

To examine whether learner-relative difficulty evolves with the policy, we track a fixed set of trajectories throughout training. At the first epoch, we collect all answer-correct trajectories and split them evenly into easy and hard groups according to their initial learner-relative difficulty (higher vs. lower likelihood). At each subsequent epoch, we recompute the likelihood of the same trajectories under the updated policy. Holding trajectory content fixed ensures that the observed changes reflect shifts in the learner's probability landscape rather than newly sampled trajectories.

Figure 2 shows that the likelihood of both groups increases as training proceeds, indicating that previously discovered correct trajectories become more probable under the updated model. More importantly, the initially hard group exhibits a steeper increase than the easy group. This provides empirical evidence that trajectories judged as difficult by the initial learner can move toward greater familiarity as training progresses, which is the behavior required for an adaptive curriculum signal.

The result does not imply that likelihood is an intrinsic measure of logical or semantic difficulty. Rather, it supports the narrower interpretation that likelihood tracks a learner-dependent notion of familiarity that changes during optimization.

This analysis tracks trajectories from problems that are already solvable at the beginning of training. Complementarily, Appendix E tracks the expansion of the solvable problem set and shows that newly solvable problems continue to emerge across epochs.

4.6 Ablation Studies

We conduct targeted ablation studies on Qwen3-8B using GSM8K and MATH to disentangle the contributions of the main components in Self-Curriculum. Table 6 summarizes the results. **Random Order** retains the same student-side trajectory filtering as Self-Curriculum but removes difficulty-based ordering, allowing us to examine whether filtering alone accounts for the improvement. **Static Order** further introduces learner-relative difficulty ranking, but computes the ordering only at the beginning of each epoch without subsequent reranking, isolating the contribution of curriculum ordering from dynamic adaptation. Finally, **Self-Curriculum** ($\alpha = 0$) retains the adaptive curriculum procedure while removing the difficulty-aware reward incentive, allowing us to assess the additional contribution of reward shaping.

Table 6: Targeted ablations on Qwen3-8B.

Method	GSM8K	MATH
GRPO	92.90	65.59
Random Order	93.19	66.77
Static Order	94.02	69.23
Self-Curriculum ($\alpha = 0$)	94.46	70.10
Self-Curriculum	**96.72**	**71.29**

Effect of Curriculum Ordering. Random Order improves over standard GRPO, suggesting that restricting training to problems for which the current policy has produced at least one answer-correct trajectory is already useful in this setting. Static Order provides a further improvement, showing that the ordering of retained trajectories contributes beyond filtering alone. This comparison supports the curriculum component independently of the within-epoch adaptive update.

Effect of Dynamic Reranking. The full method improves over Static Order, which uses the same initial learner-relative ranking but leaves that ranking unchanged within the epoch. This result supports periodically recomputing the likelihood of unused trajectories as the policy evolves. The detailed interval study in Appendix D further shows that performance is relatively stable around the chosen update frequency, while very infrequent updates move closer to the static setting.

Effect of Difficulty-Aware Reward Shaping. Setting $\alpha = 0$ removes the difficulty-dependent reward preference while retaining the curriculum ordering mechanism. Its lower performance on both benchmarks indicates that reward shaping provides an additional contribution beyond ordering. Because rewards are standardized within each prompt group, the difficulty term does not simply scale the gradient for every hard trajectory. Instead, it changes the relative preference among correct trajectories when a group contains correct responses with different learner-relative difficulties. Sensitivity to α is reported in Appendix F.

Together, these ablations indicate that the performance of Self-Curriculum is not attributable to a single component. Student-side filtering establishes a trainable set of currently solvable problems, difficulty ordering determines the progression through retained trajectories, dynamic reranking adapts that progression as the learner changes, and reward shaping introduces an additional preference among answer-correct trajectories.

5 Conclusion

We propose Self-Curriculum, which organizes outcome-verified self-generated trajectories using learner-relative trajectory likelihood. Our controlled experiments show that learner-adaptive problem-level pacing is already beneficial, while trajectory-level learner-relative likelihood provides additional gains beyond problem-level solvability signals. Comparisons with external difficulty signals further suggest that aligning the curriculum with the current learner's own probability distribution is useful for training. Across the reported benchmarks and model families, Self-Curriculum consistently improves over GRPO without requiring an auxiliary difficulty model or process-level difficulty annotations. These results support learner-relative trajectory likelihood as an effective signal for constructing adaptive curricula from a model's own generated experience.

AI USE STATEMENT

Generative AI tools were used solely for language polishing to improve the clarity and readability of the manuscript. They were not used for scientific decision-making, experimental design, data generation, or result analysis. All AI-assisted edits were carefully reviewed by the authors, who take full responsibility for the final content of the paper.

ETHICS STATEMENT

This study uses publicly available reasoning datasets and involves no human subjects. Self-Curriculum does not require additional process-level human annotations or human-provided difficulty labels beyond the existing final-answer supervision used for correctness verification. The auxiliary non-mathematical experiments reported in the appendix likewise use existing benchmark supervision rather than newly collected human data. All models used, including the teacher model employed in our comparative analysis, are publicly released checkpoints, and we do not foresee direct negative societal impacts from this work. As with other methods that improve the reasoning capabilities of language models, we acknowledge the general dual-use nature of such advances and encourage responsible downstream deployment. A comprehensive description of our use of LLMs is provided in the AI Use Statement above.

REPRODUCIBILITY STATEMENT

We have taken several steps to facilitate independent verification of our results. The core method, including the internal difficulty formulation, the normalization procedure, and the curriculum sampling and reward design, is fully detailed in Section 3. Experimental setup, datasets, and baseline implementations are described in Appendix A. Key hyperparameters and hardware specifications are documented in Appendix A.3; dataset statistics and splits are given in Appendix A.1. The construction of the dynamic pass-rate and external difficulty controls used in Section 4.3 is described to enable replication. We will release our code, training scripts, and evaluation pipelines upon publication.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Zi-Yi Dou, Cheng-Fu Yang, Xueqing Wu, Kai-Wei Chang, and Nanyun Peng. Re-rest: Reflection-reinforced self-training for language agents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 15394–15411, 2024.

Mohamed Amine Ferrag, Norbert Tihanyi, and Merouane Debbah. Reasoning beyond limits: Advances and open problems for llms. *ICT express*, 2025.

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 *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. V-star: Training verifiers for self-taught reasoners. In *Conference on Language Modeling*, 2024.

M. Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta (eds.), *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper_files/paper/2010/file/e57c6b956a6521b28495f2886ca0977a-Paper.pdf.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024.

Pengyi Li, Matvey Skripkin, Alexander Zubrey, Andrey Kuznetsov, and Ivan Oseledets. Confidence is all you need: Few-shot rl fine-tuning of language models, 2025a. URL https://arxiv.org/abs/2506.06395.

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, et al. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025b.

Aryo Lotfi, Nicolas Baldwin, Samy Bengio, Mehrdad Farajtabar, Emmanuel Abbe, Robert West, et al. Rl for reasoning by adaptively revealing rationales. In *International Conference on Learning Representations*, volume 2026, pp. 118599–118620, 2026.

Henrik Müller, Lukas Berg, and Daniel Kudenko. Using incomplete and incorrect plans to shape reinforcement learning in long-sequence sparse-reward tasks. *Neural Computing and Applications*, 37(23):18851–18866, 2025.

Shubham Parashar, Shurui Gui, Xiner Li, Hongyi Ling, Sushil Vemuri, Blake Olson, Eric Li, Yu Zhang, James Caverlee, Dileep Kalathil, et al. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning. *arXiv preprint arXiv:2506.06632*, 2025.

Yun Qu, Qi Wang, Yixiu Mao, Vincent Tao Hu, Björn Ommer, and Xiangyang Ji. Can prompt difficulty be online predicted for accelerating rl finetuning of reasoning models?, 2026. URL https://arxiv.org/abs/2507.04632.

Sunayana Rane. *Learning with curricula for sparse-reward tasks in deep reinforcement learning*. PhD thesis, Massachusetts Institute of Technology, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

Taiwei Shi, Yiyang Wu, Linxin Song, Tianyi Zhou, and Jieyu Zhao. Efficient reinforcement finetuning via adaptive curriculum learning. *arXiv preprint arXiv:2504.05520*, 2025.

Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. Curriculum learning: A survey. *International Journal of Computer Vision*, 130(6):1526–1565, 2022.

Peng-Yuan Wang, Tian-Shuo Liu, Chenyang Wang, Ziniu Li, Yidi Wang, Shu Yan, Chengxing Jia, Xu-Hui Liu, Xinwei Chen, Jiacheng Xu, et al. A survey on large language models for mathematical reasoning. *ACM Computing Surveys*, 58(8):1–35, 2026.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Zhiheng Xi, Wenxiang Chen, Boyang Hong, Senjie Jin, Rui Zheng, Wei He, Yiwen Ding, Shichun Liu, Xin Guo, Junzhe Wang, et al. Training large language models for reasoning through reverse curriculum reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 54030–54048, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=OU9nFEYR2M.

Haizhong Zheng, Yang Zhou, Brian Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 124321–124346. Curran Associates, Inc., 2025. doi: 10.52202/085713-4143. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/b42fb82fc3ffb88b872c4714f093875d-Paper-Conference.pdf.

Wenlin Zhong, Chengyuan Liu, Yiquan Wu, Bovin Tan, Changlong Sun, Yi Wang, Xiaozhong Liu, and Kun Kuang. P2s: Probabilistic process supervision for general-domain reasoning question answering. *arXiv preprint arXiv:2601.20649*, 2026.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, Biqing Qi, Youbang Sun, Zhiyuan Ma, Lifan Yuan, Ning Ding, and Bowen Zhou. Ttrl: Test-time reinforcement learning, 2025. URL https://arxiv.org/abs/2504.16084.

A Experimental Details

A.1 Dataset Details

We conduct our experiments on three widely used mathematical reasoning benchmarks: GSM8K, MATH, and NuminaMath-CoT. The statistics of the training and test sets are presented in Table 7.

GSM8K (Cobbe et al., 2021). GSM8K (Grade School Math 8K) is a dataset of 8,500 high-quality grade school math problems. Each problem requires 2 to 8 steps to solve, involving basic arithmetic operations. We use the standard training split for training and the test split for evaluation. The evaluation metric is exact match accuracy.

MATH (Hendrycks et al.). MATH is a dataset comprising 12,500 competition-level mathematics problems. It covers seven subjects: algebra, counting and probability, geometry, intermediate algebra, number theory, prealgebra, and precalculus. We use the standard training split for training and the test split for evaluation.

NuminaMath-CoT (Li et al., 2024). NuminaMath-CoT is a large-scale collection of mathematical problems with chain-of-thought annotations, sourced from diverse origins including competition forums, K12 curricula, and synthetic generation. The dataset covers a wide spectrum of difficulty levels, from elementary problems to olympiad-level challenges. We construct our experimental split by selecting 8,000 problems for training and 2,000 for testing from the following seven source subsets:

- **cn_k12:** Problems collected from Chinese K12 mathematics curricula, covering grades 1 - 12.
- **synthetic_math:** Synthetically generated mathematical problems with step-by-step solutions.
- **orca_math:** Mathematical problems from the Orca-Math source.
- **olympiads:** Problems sourced from mathematical olympiad collections.
- **synthetic_amc:** Synthetically generated AMC-style problems.
- **aops_forum:** Problems sourced from the Art of Problem Solving forum.
- **amc_aime:** Problems sourced from AMC and AIME competitions.

Detailed statistics for the three datasets used in our experiments are presented in Table 7.

Table 7: Dataset information with train and test set sizes.

Dataset / Subset	Train	Test
GSM8K	7,473	1,319
MATH	7,500	5,000
NuminaMath-CoT	8,000	2,000
cn_k12	2,594	624
synthetic_math	1,566	388
orca_math	1,427	357
olympiads	1,375	377
synthetic_amc	584	139
aops_forum	278	73
amc_aime	176	42

A.2 Baseline Method Details

We compare Self-Curriculum with main training baselines and controlled difficulty-signal variants.

1. Original The original checkpoint without additional task-specific training, used as an unadapted reference baseline.

2. Reinforcement Learning Baseline

- **GRPO (Group Relative Policy Optimization) (Shao et al., 2024):** A GRPO-style baseline without curriculum learning. The model samples k trajectories for each problem, and advantages are computed via within-group standardization using the same group-relative clipped objective described in our method. This baseline lacks difficulty-based curriculum ordering and reward shaping.

3. Self-Supervised Fine-Tuning Baseline

- **RFT (Yuan et al., 2023):** An iterative rejection-sampling fine-tuning baseline that uses the same self-generated trajectory filtering mechanism as Self-Curriculum but replaces reinforcement learning with supervised fine-tuning. Specifically, for each training epoch, the current model samples $k = 10$ trajectories per problem, retains those with correct answers, and directly fine-tunes the model on these trajectories using cross-entropy loss. This baseline evaluates whether curriculum ordering and RL optimization provide advantages beyond supervised imitation of self-generated correct trajectories.

4. External Curriculum Baseline

- **AdaBack (Lotfi et al., 2026):** A forward prompting curriculum learning method that relies on correct reasoning prefixes to control difficulty. When the model fails to generate a correct answer, AdaBack dynamically provides correct reasoning prefixes, with shorter prefixes indicating harder tasks. This baseline represents curriculum learning approaches for reasoning that depend on

external process supervision. We re-implement AdaBack in the GRPO framework, with difficulty determined by prefix length rather than learner-relative trajectory log-likelihood.

5. Learner-Relative Problem-Level Curriculum Control

- **Dynamic Pass-Rate Curriculum:** For each problem x_i, the current policy generates $k = 10$ trajectories. Let $c_i^{(j)} \in \{0, 1\}$ indicate whether trajectory j produces the correct final answer. We estimate the empirical pass rate as

$$p_i^{\text{pass}} = \frac{1}{k} \sum_{j=1}^{k} c_i^{(j)}$$

and define problem difficulty as $d_i^{\text{pass}} = 1 - p_i^{\text{pass}}$. All answer-correct trajectories from the same problem inherit this difficulty. The pass rate is recomputed from fresh rollouts at each full trajectory-resampling point, while the remaining training pipeline is kept unchanged. This control tests whether the gains of Self-Curriculum can be explained by learner-adaptive problem-level pacing alone.

6. Success-Rate Self-Paced Learning Control

- **Success-Rate SPL:** The current policy generates $k = 10$ trajectories per problem and uses empirical problem-level success rate as the easiness signal. Problems with higher success rates are introduced earlier, and a fixed easy-to-hard pacing schedule progressively expands the active problem set. Success rates and the active curriculum are refreshed using trajectories generated by the evolving policy. This baseline uses binary correctness rewards and does not use trajectory-level likelihood ordering, within-epoch likelihood reranking, or difficulty-aware reward shaping.

7. External Difficulty-Signal Controls Both external controls use the same student-generated and student-filtered training pool as Self-Curriculum and replace only the source of the curriculum difficulty signal with estimates from the fixed Qwen3-32B teacher. For **External Problem Difficulty**, we estimate the teacher's problem-level success rate from $k = 10$ rollouts and use $d_{\text{ext}} = 1 - p_{\text{ext}}$; all retained correct trajectories for the same problem inherit this value. For **External Trajectory Likelihood**, each retained student trajectory τ is scored by the teacher using length-normalized log-likelihood and assigned $d_{\text{ext}}(\tau) = -\ell_{\text{ext}}(\tau)$. Student-side filtering, curriculum construction, reranking, and reward weighting otherwise follow the corresponding Self-Curriculum pipeline.

A.3 Implementation Details

A.3.1 Hardware and Training Setup

All experiments were conducted on a compute cluster equipped with 8 NVIDIA A100 (80GB) GPUs. For the Llama-3.1-8B-Instruct and Qwen3-8B models, we use 4 GPUs for training with distributed data parallelism. For the Qwen3-14B model, we use 8 GPUs.

A.3.2 Evaluation Protocol

For evaluation sets containing fewer than 500 examples, we evaluate each trained checkpoint using 10 different decoding seeds and report the mean accuracy across the 10 runs. For evaluation sets containing at least 500 examples, we evaluate each trained checkpoint using 5 different decoding seeds and report the mean accuracy across the 5 runs. These are inference-time decoding seeds for a fixed trained checkpoint and are distinct from the independent *training* seeds reported in Section 4.4.

The model is instructed to place its final answer in a predefined answer format. We extract the answer using the corresponding format-based parser and compare it with the reference answer using the same procedure for all methods. The AMC/AIME evaluation entries in this paper are drawn directly from the corresponding `amc_aime` subset of our NuminaMath-CoT split rather than from a separately constructed year-specific benchmark.

A.3.3 HYPERPARAMETERS

We adopt the following hyperparameters for all Self-Curriculum experiments:

- **Number of sampling trajectories per problem (k):** 10
- **Difficulty incentive coefficient (α):** 0.5
- **GRPO clipping parameter (ϵ):** 0.2
- **Batch size:** 256
- **Number of training epochs:** 20
- **Difficulty reshuffling interval:** Every 40 training steps
- **Maximum trajectory length:** 2048 tokens

B ADDITIONAL ROBUSTNESS AND EFFICIENCY ANALYSES

This appendix provides detailed efficiency, rollout-budget, trajectory-freshness, and auxiliary generalization analyses that complement the robustness results in Section 4.4.

B.1 MEASURED END-TO-END OVERHEAD

The main additional cost of Self-Curriculum is periodically rescoring unused correct trajectories for reranking. Under the same hardware and training configuration, the measured end-to-end wall-clock ratios are:

Table 8: Measured end-to-end training time relative to GRPO on representative Qwen3-8B runs.

Method	GSM8K	MATH
GRPO	$1.00\times$	$1.00\times$
Self-Curriculum	$1.08\times$	$1.11\times$

The measured overhead is 8% on GSM8K and 11% on MATH. The following expanded-budget and resampling analyses further separate rollout budget from trajectory freshness.

B.2 EXPANDED-BUDGET GRPO CONTROL

Self-Curriculum incurs additional rescoring cost beyond standard GRPO. To test whether its gains can be explained simply by a moderately larger rollout budget, we increase the GRPO group size from $k = 10$ to $k = 11$, corresponding to approximately 10% more rollout samples, while retaining $k = 10$ for Self-Curriculum.

Table 9: Expanded-rollout-budget GRPO control on Qwen3-8B.

Method	GSM8K	MATH
Original	83.71	59.18
GRPO ($k = 10$)	92.90	65.59
GRPO ($k = 11$)	93.26	65.91
Self-Curriculum ($k = 10$)	**96.72**	**71.29**

As shown in Table 9, increasing the GRPO rollout budget produces only a modest improvement, while Self-Curriculum retains a clear advantage. This is a rollout-budget control rather than a strict equal-wall-clock comparison; the measured end-to-end overhead is reported in Table 8.

B.3 Full Trajectory Resampling Frequency

Within an epoch, Self-Curriculum periodically rescales neither rewards nor advantages; it only recomputes likelihoods for unused correct trajectories and updates their ordering. Full trajectory resampling is a separate operation that discards the remaining queue and regenerates trajectories from the updated policy. To probe trajectory-content staleness, we vary the number of full resampling operations per epoch while keeping the number of policy-update steps fixed.

Table 10: Sensitivity to full trajectory-resampling frequency on Qwen3-8B. More frequent resampling uses additional rollout computation and is therefore not compute-matched.

Setting	GSM8K	MATH
Original	83.71	59.18
GRPO	92.90	65.59
Self-Curriculum, 1 resample/epoch	96.72	71.29
Self-Curriculum, 2 resamples/epoch	96.89	71.52
Self-Curriculum, 4 resamples/epoch	**96.94**	71.63
Self-Curriculum, 8 resamples/epoch	96.91	**71.68**

As shown in Table 10, performance remains close across the tested resampling frequencies. Relative to once-per-epoch resampling, the largest observed improvement is 0.22 points on GSM8K and 0.39 points on MATH, despite the extra rollout cost. In the evaluated setting, once-per-epoch full resampling combined with within-epoch likelihood reranking therefore provides a favorable trade-off between freshness and cost. This result should not be interpreted as establishing that policy staleness is negligible in other training regimes.

B.4 Generalization Beyond Mathematical Reasoning

As an auxiliary generalization check, we evaluate Self-Curriculum on HotpotQA and StrategyQA, two non-mathematical benchmarks involving multi-hop or multi-step reasoning. Because these experiments were introduced as a targeted robustness check rather than part of the original full benchmark matrix, we compare against GRPO and report them in the appendix. F1 is used for both benchmarks, and trainable methods are reported as mean $\pm$ standard deviation over three independent runs.

Table 11: Auxiliary non-mathematical reasoning results. Trainable methods are mean $\pm$ standard deviation over three independent training runs.

Model	Method	HotpotQA	StrategyQA
Llama-3.1-8B-Instruct	Original	53.32	77.38
	GRPO	60.58 ± 0.49	86.29 ± 0.63
	Self-Curriculum	$\mathbf{64.71 \pm 0.37}$	$\mathbf{90.11 \pm 0.54}$
Qwen3-8B-Instruct	Original	61.08	93.71
	GRPO	63.73 ± 0.85	95.35 ± 0.32
	Self-Curriculum	$\mathbf{66.25 \pm 0.72}$	$\mathbf{97.28 \pm 0.44}$

Table 11 shows that Self-Curriculum also improves over GRPO on both non-mathematical reasoning benchmarks. We treat these results as auxiliary evidence rather than claiming comprehensive cross-domain validation.

C Additional Model Scaling Results

Section 4.4 reports Qwen3-14B results on the core benchmarks. Table 12 gives the remaining NuminaMath-CoT subsets.

Table 12: Performance of Qwen3-14B on other subsets of NuminaMath-CoT.

Model	Method	AoPS	CN K12	ORCA	Syn AMC	Syn Math
Qwen3-14B	Original	11.72	47.12	77.11	24.46	60.82
	GRPO	19.18	53.37	81.51	30.53	62.89
	Self-Curriculum	**22.19**	**56.32**	**85.26**	**32.72**	**65.08**

Across these additional subsets, Self-Curriculum also remains above GRPO, complementing the core 14B results in the main paper.

D Difficulty Reshuffling Interval Analysis

As discussed in Section 3.4.3, the difficulty of a trajectory is defined as $d(\tau_i^{(j)}) = -\ell_{\text{norm}}(\tau_i^{(j)})$, which depends on the current model parameters θ. Since θ is continuously updated during training, the difficulty scores of the same trajectories evolve dynamically. A trajectory that is difficult for the initial model may become easier after several gradient updates and should therefore be moved forward in the curriculum queue to reflect its reduced difficulty. Conversely, if its likelihood decreases, the corresponding trajectory may be postponed.

This dynamic nature necessitates periodic re-evaluation of difficulty scores for pending trajectories. We refer to this operation as *difficulty reshuffling*. The reshuffling interval determines the trade-off between curriculum adaptivity and computational overhead. A very short interval would incur high computational cost due to frequent likelihood recomputation, while a very long interval would reduce adaptivity.

To investigate the impact of the reshuffling interval on final performance, we conduct experiments on the Qwen3-8B model using the MATH benchmark. We compare five settings:

- **No Reshuffling:** Difficulty is computed only once at the beginning of each epoch and remains fixed throughout training. This serves as the baseline for non-adaptive curriculum.
- **Reshuffle every 20 steps:** High-frequency adaptation with aggressive curriculum updates.
- **Reshuffle every 40 steps:** Moderate-frequency adaptation.
- **Reshuffle every 60 steps:** Low-frequency adaptation.
- **Reshuffle every 80 steps:** Very low-frequency adaptation, approaching the non-adaptive limit.

Table 13: Ablation study on difficulty reshuffling interval on Qwen3-8B (MATH benchmark).

Reshuffling Interval	MATH Accuracy (%)
No Reshuffling	69.23
Every 20 steps	71.13
Every 40 steps	71.29
Every 60 steps	70.47
Every 80 steps	70.29

Results and Analysis. As shown in Table 13, all tested adaptive reshuffling settings outperform the non-adaptive baseline (No Reshuffling, 69.23%), supporting the benefit of periodic difficulty re-evaluation in this setting. Among the tested intervals, reshuffling every 40 steps achieves the best performance (71.29%), matching the full Self-Curriculum result reported in the main experiments.

Performance peaks at the 40-step interval among the tested settings. Reshuffling every 20 steps yields slightly lower performance (71.13%) than the 40-step setting. We hypothesize that overly frequent reshuffling may introduce curriculum instability because the model undergoes relatively few gradient updates between successive difficulty evaluations.

Reshuffling every 60 steps (70.47%) and every 80 steps (70.29%) produces lower performance approaching the non-adaptive setting. This suggests that overly infrequent updates can leave the curriculum ordered according to increasingly stale difficulty estimates.

The best-performing tested interval of 40 steps appears to strike a useful balance between adaptivity and stability in this setting. After 40 training steps, the model has processed approximately 10,240 samples (batch size 256×40 steps).

Practical Recommendation. Based on these experiments, a 40-step reshuffling interval is a reasonable default for our 8B mathematical-reasoning setup. Other model scales, datasets, or optimization regimes may require retuning this interval.

E Expansion of the Solvable Problem Set

This appendix complements the main analysis by examining whether the model can learn to solve problems that it initially cannot handle. While Section 4.5 examines likelihood changes on a fixed set of initially solvable trajectories, here we investigate expansion of the model's solvable problem set using the entire training set.

For each epoch, we sample $k = 10$ trajectories for every problem in the MATH training set and consider a problem as "solved" in that epoch if at least two of its sampled trajectories yield the correct answer. We then compute the cumulative proportion of problems that have been solved at least twice up to and including each epoch. The requirement of at least two correct trajectories (rather than one) provides a more stringent criterion for considering a problem as genuinely mastered, reducing the likelihood of counting lucky guesses.

Figure 3 presents the results using the Qwen3-8B model. At epoch 1, approximately 61.2% of problems have been solved at least twice, which is slightly higher than the single generation test accuracy of 59.2% reported in Table 1 due to the advantage of multiple sampling attempts. As training progresses, the cumulative solved proportion increases steadily, reaching 73.0% by epoch 20. This represents a relative improvement of over 19% in the number of problems the model can successfully tackle.

Critically, new problems continue to be solved even in late epochs. Between epoch 15 and epoch 20, the model still adds approximately 0.5 percentage points to the cumulative solved proportion, demonstrating that the capability frontier is still expanding and has not saturated. This provides direct evidence that our curriculum successfully enables the model to gradually tackle problems that were previously beyond its reach.

The gradually decelerating curve reflects the natural learning dynamics: the easiest problems are mastered early, while harder problems require more training steps and are solved later, consistent with the curriculum learning hypothesis.

F Difficulty Incentive Coefficient Analysis

The difficulty-aware reward function introduces a hyperparameter α that controls the strength of the difficulty incentive. For correct trajectories, the reward is defined as $R = 1 + \alpha \cdot \hat{d}(x, \tau)$, where $\hat{d}(x, \tau) \in [0, 1]$ is the normalized difficulty. A larger α amplifies the reward difference between hard and easy samples.

To investigate the impact of α on model performance, we conduct experiments on the Qwen3-8B model using the GSM8K and MATH benchmarks. We vary α from 0 to 1.0 and report the accuracy results in Table 14.

Results and Analysis. Compared to the baseline with $\alpha = 0$ (no difficulty incentive), all positive α values improve performance on both benchmarks, confirming that difficulty-aware rewards are beneficial. More importantly, the method is robust to the choice of α: values between 0.3 and 0.7 consistently achieve strong results, with only a small performance gap. The optimal value varies slightly across benchmarks. $\alpha = 0.7$ gives the best GSM8K accuracy while $\alpha = 0.5$ leads on MATH, but the differences are marginal. This insensitivity suggests that our difficulty normalization and

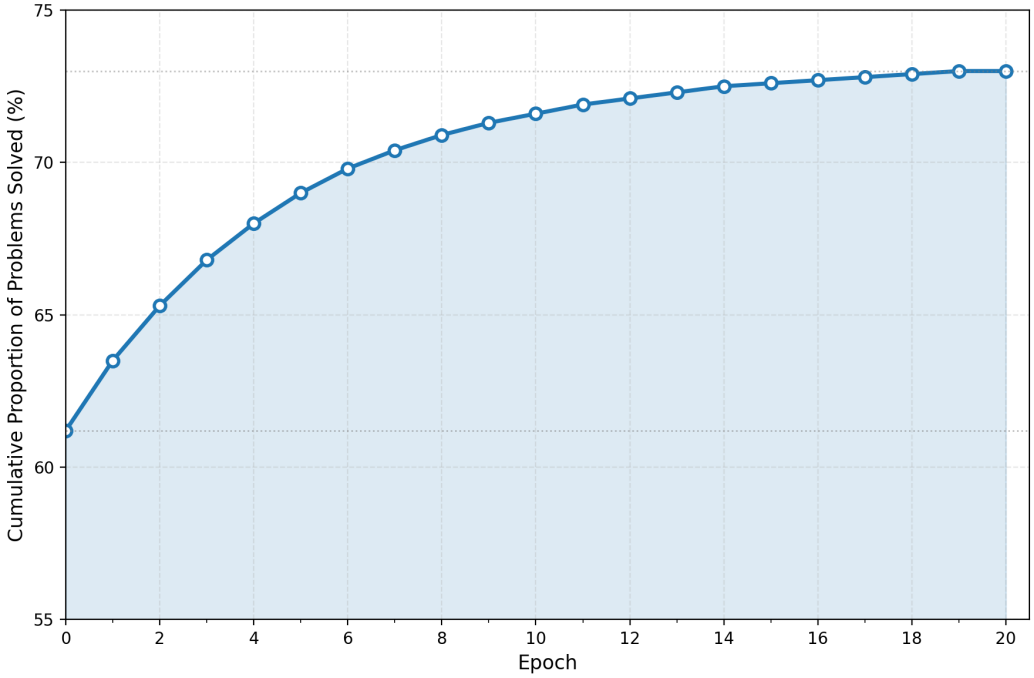

Figure 3: Cumulative proportion of training problems solved at least twice over training epochs on MATH using Qwen3-8B.

Table 14: Ablation study on the difficulty incentive coefficient α using Qwen3-8B.

α	GSM8K (%)	MATH (%)
0.0	94.46	70.10
0.3	96.31	71.07
0.5	96.72	**71.29**
0.7	**96.84**	70.96
1.0	96.48	70.85

reward design are well behaved, and practitioners can safely set $\alpha = 0.5$ without extensive tuning. We adopt $\alpha = 0.5$ for all main experiments.

G Case Study: Internal vs. External Difficulty Divergence

To illustrate why internal difficulty based on the student's own trajectory log-likelihood outperforms external difficulty from a stronger teacher model, we conducted a case study analysis.

G.1 Problem Statement

Consider the following problem:

Problem Example

Find all real numbers x such that $\sqrt{x+5} - \sqrt{x-3} = 2$.

G.2 Comparing Internal vs External Difficulty Assessments

We compare three difficulty assessments for two different correct reasoning trajectories generated by the student model (Qwen3-8B) at epoch 5:

- **Internal Difficulty:** $d(\tau) = -\ell_{\text{norm}}(\tau)$, computed by the student model itself.
- **External Problem Difficulty:** $d_{\text{ext}} = 1 - p_{\text{ext}}$, where p_{ext} is the teacher's problem-level accuracy over its sampled responses.
- **External Trajectory Likelihood:** $d_{\text{ext}}(\tau) = -\ell_{\text{ext}}(\tau)$, where $\ell_{\text{ext}}(\tau)$ is the teacher's length-normalized log-likelihood of the same trajectory.

G.3 Two Reasoning Trajectories

Trajectory A (Conventional solution). The model solves the problem by isolating the square root and squaring both sides:

Step 1: Add $\sqrt{x-3}$ to both sides: $\sqrt{x+5} = 2 + \sqrt{x-3}$

Step 2: Square both sides: $x + 5 = 4 + 4\sqrt{x-3} + (x-3)$

Step 3: Simplify: $x + 5 = x + 1 + 4\sqrt{x-3}$

Step 4: Cancel x: $5 = 1 + 4\sqrt{x-3}$

Step 5: Solve: $4 = 4\sqrt{x-3} \to \sqrt{x-3} = 1 \to x - 3 = 1 \to x = 4$

Trajectory B (Alternative solution). The model uses a substitution method that is less common but more elegant:

Step 1: Let $a = \sqrt{x+5},\ b = \sqrt{x-3}$. Then $a - b = 2$.

Step 2: Note that $a^2 - b^2 = (x+5) - (x-3) = 8$.

Step 3: Since $a^2 - b^2 = (a-b)(a+b) = 2(a+b),$ we have $2(a+b) = 8 \to a + b = 4$.

Step 4: Solve $a - b = 2$ and $a + b = 4 \to a = 3,\ b = 1$.

Step 5: From $b = \sqrt{x-3} = 1 \to x - 3 = 1 \to x = 4$.

G.4 Analysis

Table 15: Comparison of internal vs external difficulty assessments for two trajectories.

Assessment Method	Trajectory A	Trajectory B
Internal Difficulty (Student)	0.32 (Easy)	0.78 (Hard)
External Problem Difficulty	0.20 (Easy)	0.20 (Easy)
External Trajectory Likelihood	0.41 (Medium)	0.44 (Medium)

The results are shown in Table 15. This example is intended to illustrate how learner-relative and teacher-relative signals can disagree; it is not evidence that either signal universally measures semantic reasoning difficulty.

External problem difficulty assigns the same value to both trajectories by construction because it is defined at problem granularity. It therefore cannot distinguish two answer-correct trajectories for the same prompt, even when the student assigns them different probabilities.

External trajectory likelihood provides trajectory-level scores, but in this example the teacher assigns similar scores to the two student-generated trajectories. The learner-relative signal separates them more strongly, reflecting a difference between the teacher's and student's probability landscapes.

Learner-relative trajectory likelihood therefore induces a different ordering for this example. We interpret the difference only as a measure of relative generation probability under the student policy. Surface form, tokenization, notation, and other non-semantic factors may also contribute to the likelihood gap.

The aggregate experiments in Section 4.3, rather than this individual example, provide the primary evidence for the usefulness of learner-relative trajectory likelihood. The case study simply visualizes one possible source of disagreement between problem-level, teacher-relative trajectory-level, and learner-relative trajectory-level signals.

H Detailed Limitations and Scope

Despite the promising results, there are still some limitations that can be addressed in future work:

Efficiency of Problem Filtering. In our current approach, problems for which the model does not generate any correct solution across all k sampling attempts are temporarily filtered out from training in the current epoch. The proportion of such problems naturally decreases as training progresses, but sample utilization efficiency in the early stages remains suboptimal. Developing more sophisticated strategies for handling currently unsolved problems, such as curriculum-aware problem decomposition or adaptive difficulty adjustment, could further improve training efficiency.

Interpretation and Theoretical Understanding of Learner-Relative Difficulty. Our experiments show that learner-relative trajectory likelihood outperforms the evaluated dynamic pass-rate and external difficulty alternatives in our setting, but this does not imply that negative log-likelihood is an intrinsic or semantic measure of reasoning difficulty. Likelihood can also be affected by surface form, tokenization, trajectory length normalization, and the multiplicity of valid reasoning paths. We therefore use it as an operational pacing signal relative to the current policy. Its theoretical properties remain underexplored, including how the signal evolves under policy updates, when its ordering is stable, and how it relates to notions of sample complexity and learnability. Formalizing these properties is an important direction for future work.

Computation, Staleness, and Formal Guarantees. Periodic likelihood rescoring introduces measurable extra computation: in our representative Qwen3-8B runs, end-to-end training time is $1.08\times$ GRPO on GSM8K and $1.11\times$ on MATH. Within an epoch, only the ordering of unused trajectories is refreshed; difficulty-aware rewards and precomputed advantages remain fixed until the next full trajectory resampling. This creates two adaptation timescales and does not eliminate all policy-staleness effects. We do not provide a convergence guarantee, an optimality guarantee for the induced curriculum, or formal robustness guarantees under arbitrary distribution shift or catastrophic forgetting.