Extended Abstract

Motivation Reinforcement learning is commonly used for language model reasoning, and reverse curriculum reinforcement learning (RCRL) is commonly used in robotics RL applications, but only one paper has explored the usage of RCRL for language model reasoning **?**. We believe that the application of RCRL to language model reasoning is promising, and this project builds on the proposed R^3 algorithm to achieve improvements surpassing their methods.

Methods We introduce R^4, recipes for reverse curriculum reinforcement learning for language model reasoning by experimenting with four key modifications: (1) FastWarm (allocating less training time to easier curriculum prompts and more to harder ones), (2) StartHard (beginning training at a harder difficulty), (3) FinalVanilla (replacing mixed-difficulty, curriculum final stages with vanilla prompts), and (4) GradMix (creating gradual rather than discrete difficulty transitions). We test and evaluate these modifications independently and compare them against baseline reverse curriculum methods. We then devise follow-up experiments that explore whether the best of our modifications can receive further augmentation when using SFT warmup.

Implementation We conduct experiments using Qwen2.5-Math-1.5B as our base model on 6400 training examples from the Orca-Math dataset, which provides detailed step-by-step reasoning chains suitable for reverse curriculum construction. Our training algorithm uses GRPO (Group Relative Policy Optimization) with binary rewards for final answer correctness. We evaluate our results on both answer accuracy and format accuracy on 1600 test examples.

Results Our best-performing modification, R^3 -FastWarm, achieves 65.4% answer accuracy, constituting 4.2% improvement over the original R^3 (61.2%). Surprisingly, vanilla RL achieves competitive performance at 65.1%, outperforming many complex reverse curriculum approaches. When applied after SFT initialization, all methods show substantial gains, with R^3 -FastWarm on SFT reaching 70.8% answer accuracy. The performance gap between different methods narrows significantly in the SFT setting, suggesting that strong prior training reduces the relative importance of curriculum design.

Discussion Our findings challenge **?**'s finding that reverse curricula universally outperform simpler approaches for mathematical reasoning. The strong performance of vanilla RL suggests that for reasoning tasks, consistent exposure to complete problems at their hardest difficulty may be more beneficial than naive graduated difficulty progression. However, the success of our FastWarm modification indicates that careful curriculum design can outperform vanilla methods. The convergence of performance across methods after SFT also highlights the importance of foundational reasoning capabilities.

Conclusion While reverse curriculum learning can be improved through our targeted modifications, the combination of strong supervised pre-training with efficient curriculum design (FastWarm) appears to provide the most effective approach for mathematical reasoning tasks. Our experiments demonstrate that the effectiveness of RCRL and R^3 can be improved upon, though the diminishing returns in SFT settings suggest that curriculum complexity becomes less critical when models possess strong foundational capabilities. These insights may provide valuable guidance for future work in reverse-curriculum-based reinforcement learning for language model reasoning.

Introducing R^4: Recipes for Language Model Reasoning through Reverse Curriculum Reinforcement Learning

Jack Hsieh
Department of Computer Science
Stanford University
jhsiehca@stanford.edu

Dillon Nguyen
Department of Computer Science
Stanford University
dillonkn@stanford.edu

Ethan Zhang
Department of Computer Science
Stanford University
ez26@stanford.edu

Abstract

Xi et al. (2024) propose an adaptation of reverse curriculum reinforcement learning (RCRL) for large language model reasoning. While their approach to generating examples of varying difficulty is highly intuitive, their exploration of how difficulties should be staged within a reverse curriculum is limited. In this work, we conduct a thorough investigation to develop more effective reverse curriculum design strategies. We undertake a thorough study to develop a more effectively structured curriculum could lead to better results. We apply four targeted modifications: FastWarm (allocating less training time to easier problems and more to harder problems), StartHard (starting training in high difficulty), FinalVanilla (ending training with a stage akin to the test dataset), and GradMix (gradated difficulty increases rather than discrete). We evaluate our approach using Qwen2.5-Math-1.5B on the Orca-Math dataset. Our best-performing modification, R^3 -FastWarm, achieves 65.4% answer accuracy, representing a 4.2 percentage point improvement over our R^3 implementation (61.2%). However, further experimentation shows that warming up with SFT equalizes methods, regardless of curriculum structure, indicating that strong foundational capabilities may be more important than intelligent curriculum design.

1 Introduction

Reinforcement Learning (RL) has proven highly effective for improving the reasoning capabilities of Large Language Models (LLMs) over a diverse range of tasks. Most methods involve performing RL over Chains-of-Thought (CoTs). However, most reasoning datasets lack **process supervision** that labels intermediate states (i.e. thoughts) with rewards. Process supervision is subjective, expensive to collect, and often infeasible for online learning. Instead, reasoning datasets typically provide final answers, allowing for only **outcome supervision** about whether a CoT ultimately culminated in a correct answer. Such supervision, while easier to collect and more suitable for online learning, leads to sparse rewards. This sparsity can make learning over CoTs difficult, especially when a task requires exploration at the start.

While credit assignment and RLAIF methods aim to provide denser rewards, an alternative approach is to better leverage sparse outcome-based rewards. Reverse Curriculum Reinforcement Learning

(**RCRL**) starts training with "easy" start states near the end of the final outcomes and progressively rolls back the start states toward the beginning of each trajectory. This process induces an increasingly difficult training curriculum where as the policy improves, it encounters increasingly difficult tasks (i.e., start states farther from the final goal), potentially enabling outcome supervision alone to provide a more consistent training signal. Critically, RCRL is compatible with sparse rewards, potentially avoiding unintended behaviors/reward hacking often introduced by reward shaping. RCRL is therefore particularly well-suited for language reasoning tasks. However, existing work on RCRL for Language Models (LMs) is incipient, with only one work applying a basic version of RCRL to LMs. In this project, we explore more sophisticated design of reverse curricula recipes to outperform generic training methods.

2 Related Work

$\mathbf{R}^3$ **?**, the main paper on which our approach is based, introduces **R**easoning through **R**everse Curriculum **R**einforcement Learning ($\mathbf{R}^3$). While RCRL has been proposed and developed much earlier for robotics, this paper most critically was the first (and, to our knowledge, the only) to apply the technique to LLM reasoning capabilities. R^3 selects start states by randomly splitting each CoT into $M \approx 8$ segments and organizing the curriculum into M stages of increasing distance from the end of the CoT. RL is then performed on each stage in ascending order of distance from the end, followed by a final mixing stage across all starting states to improve generalization. **?** demonstrate the effectiveness of RCRL across math reasoning, logical reasoning, natural language inference, and reading comprehension. While R^3 provides a first foray into the adaptation for RCRL for LM reasoning tasks, recipes for constructing reverse curricula for the setting of language model reasoning remain underexplored. Moreover, the demonstrated improvements for math are quite modest, suggesting room for improvement via more targeted reverse curricula construction.

Forward-Backward Tao et al. (2024) introduces a reverse and forward curricula hybrid approach to curriculum learning. Key to their method is the insight that the reverse stage enables the policy to learn successful behavior from demonstrated start states, while the forward stage enhances generalization across the broader start state space. This approach has proven successful in domains such as robot manipulation, including tasks like key insertion and ring on peg with seven degree-of-freedom robots, where agents learned not only to avoid constraints but to exploit them effectively Florensa et al. (2018). Though we do not implement explicitly forward-backward curricula in our modifications, this paper served as a critical reference point in our design of variable curricula composition.

3 Methods

Approach We begin with several baseline experiments to assess to assess the effectiveness of R^3 against vanilla RL, and determine possible areas of improvement. Following these baseline experiments, we move to the main experimentation of our work: modifying R^3 . Specifically, we focused on implementing targeted adjustments to the composition and order of data stages during our reverse curriculum model training. After analyzing the outcomes of these main experiments, we investigated interesting trends with several follow-up experiments, focusing on augmenting the performance of the modifications that performed best during our main experiments.

3.1 Dataset Construction

For each reverse curriculum experiment, a unique, custom dataset is needed, as data must be pre-processed, formatted, and staged according to the experiment's goals. In addition to processing prompts to have partial CoTs of proper length, many of our experiments require several stages of varying difficulty compositions. For instance, R^3 , as mentioned in Related Works, demands a dataset with *M* stages of increasing difficulty followed by a stage of mixed difficulty **?**. To create these custom datasets, we write a universal dataset generator that constructs a dataset given a configuration.

Stage construction Formally, we specify a single data stage as $(D; b)$, where b is the number of batches[1] and $D \subseteq \{1, 2, \ldots, d_{\text{max}}\}$ is a set of difficulties mixed within the stage. To construct a data stage $(D; b)$, we first sample n train examples, including the question and the golden rationale, from the train set. For each train example, we randomly sample a difficulty $d \in D$. Then, with L being the number of newline-separated lines in the golden rationale for the train example, we append $\left\lfloor L \cdot \left(1 - \frac{d}{d_{\text{max}}}\right)\right\rfloor$ lines of the golden rationale to the end of the question to form the reverse curriculum prompt. Critically,

- The lower the difficulty, the more lines of the golden rationale are included in the prompt.
- Since $d \geq 1$, the last line containing the ground-truth answer of the golden rationale is never included in the prompt.
- For the hardest difficulty $d = d_{\text{max}}$, no lines from the golden rationale are included in the prompt. Using the dataset prompts as they are is therefore equivalent to this hardest stage.

We find that the golden rationales tend to have more than $d_{\text{max}} \leq 6$ newline-separated lines, allowing each stage to meaningfully differ.

> A train travels 290 km in 4.5 hours, then continues for another 400 km in 5.5 hours. After that, it travels 350 km in 7 hours, and finally completes a 480 km trip in 6 hours. Calculate the overall average speed of the train during the entire journey. To calculate the overall average speed of the train during the entire journey, we need to find the total distance traveled and the total time taken. Total distance traveled = 290 km + 400 km + 350 km + 480 km. Total distance traveled = 1520 km. Total time taken = 4.5 hours + 5.5 hours + 7 hours + 6 hours. Total time taken = 23 hours. Now, we can calculate the overall average speed using the formula: Average speed = Total distance traveled / Total time taken.
>
> {model completion here}

Figure 1: An example difficulty 1 prompt from R^3 with concatenated partial golden rationale, including the question and concatenated partial CoT, as well as an indicator signposting where model output begins. Note that this signposting does not exist in the actual datasets.

Curricula recipes We represent a reverse curriculum recipe as series of data stages, where d_{max} is inferred as the largest difficulty.[2] For example, we use

$$[(1 - 3; 40), (4; 80)]$$

to represent a reverse curriculum with two data stages:

1. The first stage contains 40 batches, with difficulties uniformly sampled from 1-3.
2. The second stage contains 80 batches, with the highest difficulty, 4.

In this example, the last stage is equivalent to non-curriculum prompts.

3.2 Baseline Experiments

Our baseline experiments include the following:

- **Base Model**: We evaluate our base model zero-shot on our eval dataset to establish a baseline of performance.
- **Supervised Fine-Tuning**: We run supervised fine-tuning for 10 epochs on the prompts and golden rationales of our train set.

[1] As discussed in our experiment details, we sample 32 prompts per batch. Thus, the number of examples in each stage is $32b$.

[2] We assume that the hardest difficulty (i.e., no partial rationales) is always worth including.

- **Vanilla**: We run 1 stage of 120 epochs of prompts without any golden rationale concatenation. The curriculum recipe is [(1; 120)].
- **Reverse Staged**: The model trains on 6 stages of increasing difficulty across 120 epochs. The curriculum recipe is [(1; 20), (2; 20), (3; 20), (4; 20), (5; 20), (6; 20)]. This is equivalent to running R^3 for $M = 6$ stages and without the final mixed stage.
- **Reverse Mixed:** The dataset is one single stage of 120 epochs containing prompts all difficulties, equivalent to running the last stage of R^3 for 120 epochs. The curriculum recipe is [1-6; 120].
- **R^3 :** We replicate R^3 with $M = 4$ to accommodate compute constraints. The curriculum recipe is [(1; 20), (2; 20), (3; 20), (4; 20), (1-4; 40)].

3.3 R^3 Modifications

After establishing our implemented baselines, we identified four potential areas for algorithmic improvement for reverse curriculum learning (see Results). We give our targeted modifications to the core reverse curriculum recipes.

1. **Vanilla Final Stage (FinalVanilla)**: Vanilla RL training demonstrates consistent high performance during our baseline experimentation, outperforming our Reverse Mixed implementation. We theorize that replacing the last stage of R^3 , which is typically a double-sized Reverse Mixed stage, with a pure vanilla prompt stage will better resemble the evaluation data and match the high performance in our vanilla baseline model.
2. **Fast Warm-Up (FastWarm)**: Given that the base model is already capable of answering and formatting many questions correctly, we hypothesize that allocating less training time to easier prompts and more time to harder prompts will improve performance by training the model in its optimal zone of difficulty while still receiving a brief warm-up with the easier difficulties.
3. **Start with Harder Problems (StartHard)**: Similarly to FastWarm, this alteration to R^3 bypasses the easier first half of difficulties and begins training with shorter partial rationales attached. This configuration posits that the easy half of stages is not necessary for the model to converge on desired behavior and that devoting all time to the harder stages is more efficient. Note that this modification is mutually exclusive with Fast Warm-Up, as Fast Warm-Up requires at least some time spent in easier stages.
4. **Gradual Mixing (GradMix)**: Instead of each stage being a discrete increase in difficulty, this modification creates a gradated increase in difficulty. For instance, consider [(1-2; 30), (2-3; 30), (3-4; 30), (4-5; 30)]. Stage 1 has prompts of difficulty 1 and 2, and Stage 2 has prompts of difficulty 2 and 3, thereby making the difficulty increase from Stage 1 to Stage 2 less discrete. We speculate that doing so will stabilize model learning across stages and improve final performance with this more supported structure.

We also consider **Mixed-Vanilla**: one stage of mixed difficulty prompts, followed by an equally sized stage of vanilla prompts. The curriculum recipe is [(1-4; 60), (4; 60)].

To explore how the modifications affect performance of R^3 , we run the following experiments with the following dataset configurations:

- **R^3 -FastWarm:** [(1-2; 10), (3-4; 30), (5-6; 80)]
- **R^3 -StartHard:** [(3; 30), (4; 30), (5; 30), (6; 30)]
- **R^3 -GradMix**: [(1-3; 40), (2-4; 40), (3-5; 40)]
- **R^3 -FinalVanilla**: [(1; 20), (2; 20), (3; 20), (4: 60)]
- **R^3 -FinalVanilla-FastWarm**: [(1-2; 10), (3; 20), (4; 30), (5; 30), (6; 30)]
- **R^3 -FinalVanilla-StartHard**: [(3; 30), (4; 30), (5; 30), (6; 30)]
- **R^3 -FinalVanilla-GradMix**: [(1-3: 30), (2-4; 30), (3-5: 30), (6: 30)]

Given that vanilla RL performed well in our initial baseline experiments, we aim to measure the impact of training with FinalVanilla, hence the testing of the other 3 modifications in the presence and absence of FinalVanilla.

3.4 Follow-Up Experiments

After finding vanilla RL, mixed stages, and FastWarm trials to produce the best results across our experiments, we run five additional experiments on top of our SFT model to examine their performance after SFT gives the model exposure to golden rationales. Doing so negates the golden rationale exposure that reverse curriculum learning provides, so these experiments intend to show whether leveling the exposure playing field through SFT leads to interesting results. Specifically, we run the following experiments on a model that has undergone 10 epochs of SFT:

- **Vanilla on SFT**: [(1; 120)]
- **Mixed-Vanilla on SFT**: [(1-4; 60), (4; 60)]
- **$\mathbf{R}^3$ on SFT**: [(1; 20), (2; 20), (3; 20), (4; 20), (1-4; 40)]
- **$\mathbf{R}^3$ -FastWarm on SFT**: [(1-2; 10), (3-4; 30), (5-6; 80)]
- **$\mathbf{R}^3$ -FinalVanilla-FastWarm on SFT**: [(1-2; 10), (3; 20), (4; 30), (5; 30), (6; 30)]

4 Experimental Setup

4.1 Experiment Details

We invested a significant amount of code and effort in designing a suitable experimental pipeline given our compute constraints. We detail our choices here.

Base Model We adopt **Qwen2.5-Math-1.5B** as our base model, a model pretrained on high-quality English and Chinese mathematical text. (Yang et al., 2024). We select the base (non-instruct) variant rather than the already fine-tuned Qwen2.5-Math-1.5B-Instruct model, as we seek to measure the impact of RL fine-tuning from scratch.

We find that the 1.5B model strikes a suitable balance: strong enough to achieve meaningful reward signals, yet not so strong that it leaves little room for improvement through RL fine-tuning. Crucially, the model is just small enough to be trained on the single NVIDIA L40S GPUs available to us.

Datasets For our experiments, we use the **Orca-Math** dataset, gathering a subset consisting of 6,400 training and 1,600 evaluation examples (Mitra et al., 2024). A key advantage of Orca-Math is that it provides verified step-by-step rationales generated by GPT-4-Turbo. These rationales are significantly longer and more detailed than many other datasets, making them more suitable for use in constructing reverse curricula. Indeed, the included chain-of-thought annotations were specifically designed for distillation, aligning well with our training objective.

RL Details We train our models for 120 GRPO steps using a single NVIDIA L40S GPU, with both training and inference performed on the same device. For rollout generation, we use vLLM, a highly optimized inference engine (Kwon et al., 2023). As a result, inference was comparable in speed to log-probability computation, so we adopt an on-policy RL approach. While we explored off-policy GRPO and standard REINFORCE, we found that REINFORCE with a mean baseline (similar to RLOO) consistently performed best within our training budget.

For each RL step, we sample 32 distinct prompts from the active data stage and generate a group of 8 completions, yielding 256 rollouts per batch. We then perform an on-policy update. We use binary rewards: 1.0 if the rollout marks the correct final answer, and 0.0 otherwise. Rewards are then z-normalized within each group to compute advantages.

An initial hyperparameter sweep showed that a constant learning rate of 2e-5 provided stable and effective learning across runs. We used a constant learning rate schedule, as learning rate annealing minimizes the impact of the later, most critical stages in reverse curricula.

Metrics We evaluate **format accuracy** and **answer accuracy** on the 1600 evaluation examples every 10 GRPO steps. To better isolate reasoning ability from formatting, we also compute the answer-format ratio (AFR), defined as $\text{AFR} = \frac{\text{answer accuracy}}{\text{format accuracy}}$. Since a correct answer requires correct formatting, AFR is bounded above by 1 and reflects the proportion of identifiable answers that are indeed correct.

In addition, we track the mean response length and the mean train reward, which serves as a proxy for training accuracy due to the binary reward function. For comparison across runs, we report metrics corresponding to the checkpoint with highest evaluation accuracy.

5 Results

Table 1: Peak accuracy across baselines and our R^3 modifications

Method	Answer Accuracy	Format Accuracy	Answer-Format Ratio
Base Model	0.463	0.765	0.605
Vanilla RL	**0.651**	0.931	**0.699**
Reverse Staged	0.629	0.940	0.669
Reverse Mixed	0.626	**0.948**	0.660
R^3	0.612	0.943	0.649
R^3 -FastWarm	**0.654**	**0.954**	0.686
R^3 -StartHard	0.637	0.941	0.677
R^3 -GradMix	0.628	0.937	0.670
R^3 -FinalVanilla	0.643	0.948	0.678
R^3 -FinalVanilla-FastWarm	0.648	0.946	0.685
R^3 -FinalVanilla-StartHard	0.644	0.942	0.684
R^3 -FinalVanilla-GradMix	0.636	0.941	0.676
Mixed-Vanilla	0.653	0.941	**0.694**

Table 2: Peak accuracy across baselines and our R^3 methods on top of SFT.

Method	Answer Accuracy	Format Accuracy	Answer-Format Ratio
SFT	0.680	0.965	0.705
Vanilla on SFT	**0.709**	0.976	**0.726**
R^3 on SFT	0.701	**0.978**	0.717
R^3 -FastWarm on SFT	**0.708**	0.973	**0.728**
R^3 -FinalVanilla-FastWarm on SFT	0.701	0.973	0.720
Mixed-Vanilla on SFT	0.702	**0.975**	0.720

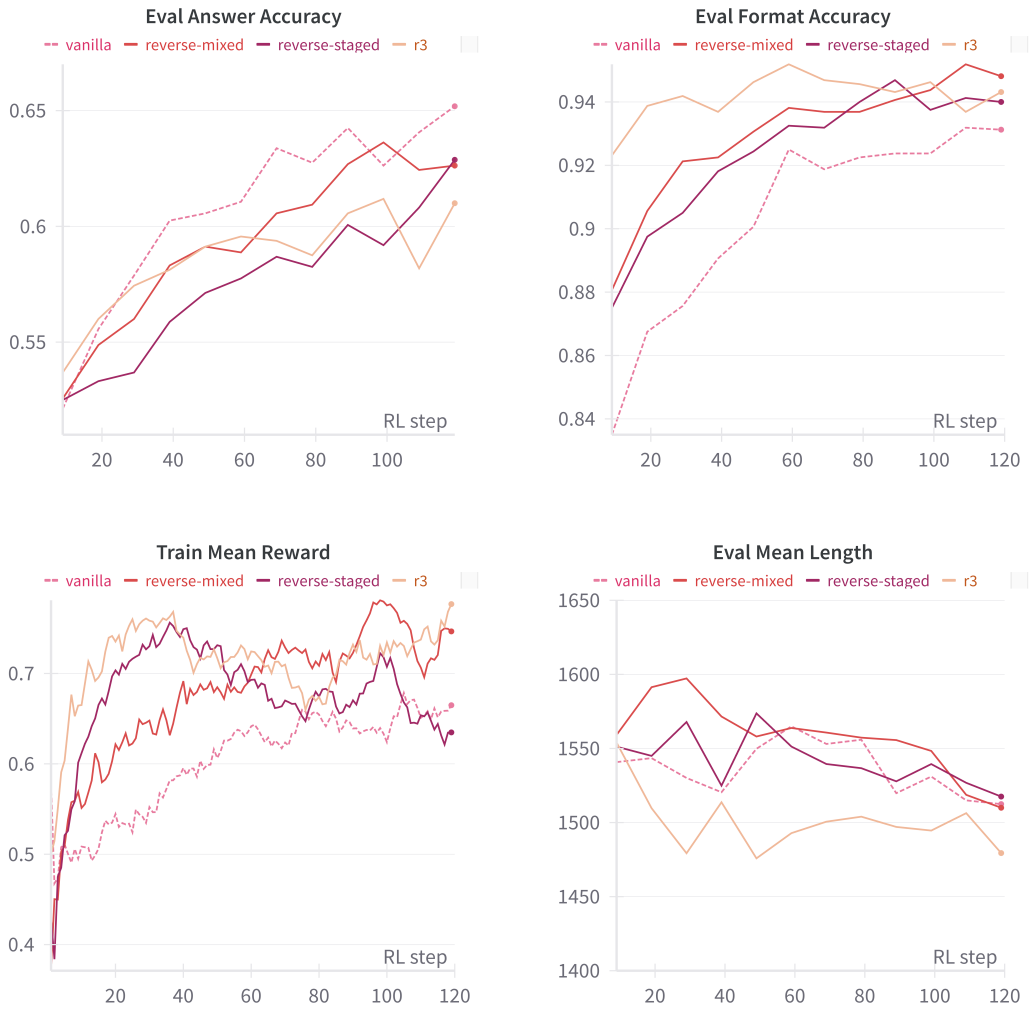

Figure 1: Baselines, corresponding to the top half of Table 1

Baselines We find that our baseline reverse curriculum methods learns formatting faster than vanilla RL and consistently demonstrates superior formatting accuracy. These findings are consistent with the presence of "easy" curriculum prompts that primarily train the policy to correctly format answers. However, Vanilla RL eventually converges a nearly comparable formatting accuracy. Similarly, we find that response lengths are mostly comparable toward the end of training, though notably, R^3 produces the shortest outputs. Across all methods, manual inspection of outputs and checking lengths reveal that the model acquires proficient general reasoning patterns across techniques.

We also find that the mean train reward is significantly higher and stabilizes faster throughout when using the reverse curriculum, which validates that the reverse curriculum provides an initially easier and more stable problem difficulty than a vanilla curriculum. However, we find that our baseline reverse curriculum methods generally underperform on the evaluation set compared to Vanilla RL. While lower performance in the earlier stages is to be expected, as the policy is not yet trained on full examples, our reverse curriculum baselines fail to recover the original performance of vanilla RL — at least, within our training budget. In agreement with **?**, we find that among the reverse curriculum baselines, R^3 performs best.

The high train rewards suggest that the less difficult stages of data may be too easy. While these easy problems help the policy learn formatting, they may detract from the policy reaching harder examples, which are needed to achieve highest performance on the evaluation dataset. Our 4 modification strategies aim to increase the difficulties of the staged dataset while preserving formatting benefits.

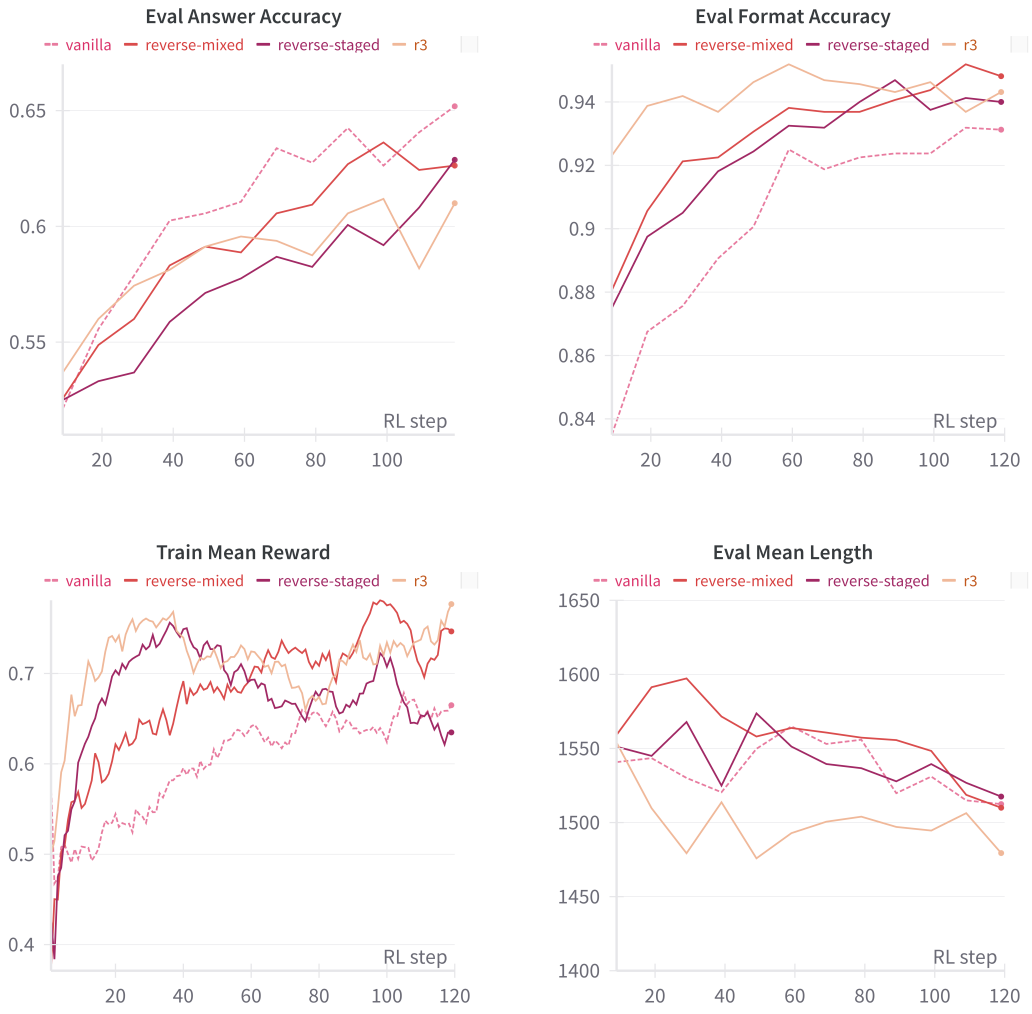

Figure 2: Baselines and our methods, corresponding to the full table of Table 1

R^3 Modifications As our modifications increase the average difficulty of the provided data, we see that the policy learns formatting somewhat slower than R^3 but still significantly faster than RL, indicating that the formatting of benefits of a reverse curriculum are preserved to a degree. Similarly, we observe a decrease in the average training reward, validating that our staged datasets are indeed more difficult than our baselines, though still easier than the vanilla curriculum.

We find that all but one of our modifications improve performance over our reverse curriculum baselines. Two of our methods, FastWarm and Mixed-Vanilla, outperform our vanilla RL baseline. Overall, we identify FastWarm and StartHard as the most effective strategies for increasing the difficulty of our staged datasets by an appropriate amount, while still retaining formatting benefits.

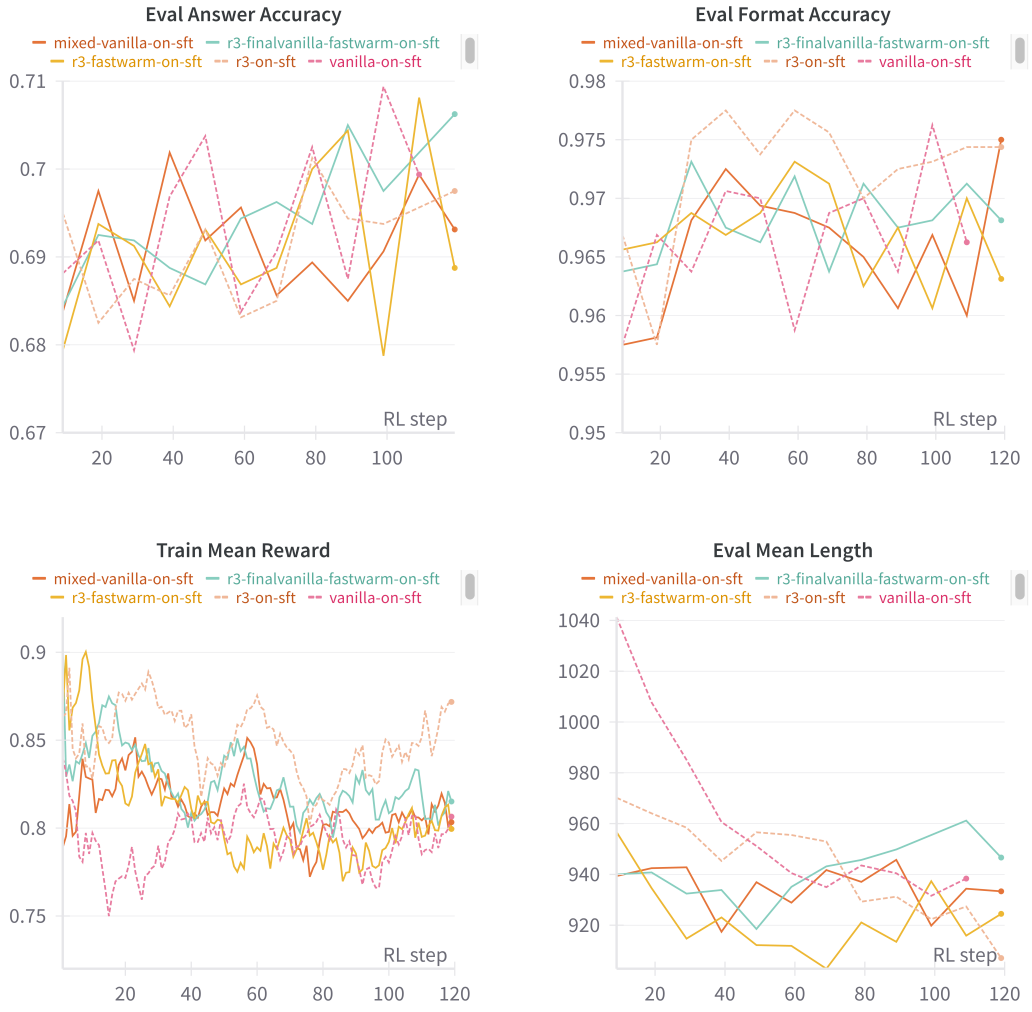

Figure 3: Best baselines + methods atop SFT, corresponding to the full table of Section 5

Methods atop SFT We find that SFT is highly effective at leveraging the rationales, providing a stronger baseline on which to perform RL for all methods. We find generally that RL offers small gains atop SFT, and reverse curriculum methods do not offer improvements over vanilla RL. We attribute this to the likely possibility that SFT ingrains strong formatting behavior and instills foundation reasoning capabilities on its own, reducing the added effectiveness of reverse curriculum learning.

6 Discussion

Our experimental results reveal that the positive impact of reverse curricula learning on language reasoning may not be simple. The most striking finding is that vanilla RL achieves competitive performance (65.1% answer accuracy) that rivals or exceeds more sophisticated reverse curriculum methods. This challenges the original R^3 work's suggestion that complex curricula universally outperform simpler approaches for mathematical reasoning tasks. We hypothesize that for reasoning tasks, consistent exposure to complete problem-solving trajectories may be more beneficial than graduated difficulty progression, as it allows the model to learn end-to-end reasoning patterns without the potential confusion introduced by partial demonstrations.

While we focus on a multi-step math reasoning setting of comparable difficulty to **?**, the greatest benefits of a reverse curriculum may emerge in more challenging, long-CoT tasks. Due to compute constraints, we were unable to explore such long-context domains. Future work should investigate strategies for constructing reverse curricula in these settings, such as agentic or coding tasks.

Among our modifications, FastWarm consistently emerges as the most effective strategy, improving R^3 performance by 4.2 percentage points. This success validates our hypothesis that spending less time on problems the model can already solve and more on challenging cases is crucial for effective curriculum design. However, our follow-up experiments reveal that supervised fine-tuning acts as an equalizer, substantially improving all methods while narrowing performance differences. This suggests that SFT provides crucial foundational reasoning and formatting capabilities that reduce the relative importance of sophisticated reverse curriculum design. Our results suggest that curriculum design for mathematical reasoning should prioritize time efficiency over complexity, but that such work may not be as fruitful when golden rationales exist for SFT.

Our compute constraint was a persistent challenge throughout the project. We originally had a completely different experimental setup for the milestone and presentation that we completely replaced due to slow training and poor performance. Our prior implementations are detailed in the appendix.

7 Conclusion

This work systematically examines the effectiveness of reverse curriculum reinforcement learning for mathematical reasoning in language models, addressing fundamental limitations in existing approaches through targeted algorithmic modifications. Our evaluation reveals that adjusting data composition recipes is key to constructing successful reverse curricula, with our FastWarm modification outperforming the original R^3 by several percentage points. These improvements, however, come with an important caveat: vanilla RL achieves surprisingly competitive performance, often matching or exceeding more sophisticated reverse curriculum approaches. This finding questions the necessity of complex curricula for mathematical reasoning tasks and suggests that simpler training strategies may be more effective than previously recognized.

Another key insight from our research concerns the interaction between curriculum design and supervised fine-tuning. When reverse curriculum methods are applied after SFT initialization, we observe substantial improvements across all approaches, but performance differences between methods narrow significantly. Moreover, SFT offers the most significant increase in performance from leveraging golden rationales of the methods we examine. This pattern suggests that strong prior training reduces the relative importance of complex curriculum structures, with practical implications for researchers working with limited computational resources.

Our contributions extend beyond algorithmic improvements to reshape broader understanding of curriculum learning in language models. While we demonstrate that targeted refinements can meaningfully enhance RCRL effectiveness, we simultaneously reveal the competitive nature of simpler training strategies, highlighting the need for a more cautious understanding and approach to curriculum design.

Our findings open several promising avenues for future investigation. The generalizability of our modifications to domains beyond mathematical reasoning presents a compelling research direction, particularly in areas where golden rationales are scarce and expert-iteration-style approaches would be necessary. Similarly, applying our FastWarm principles to more challenging reasoning domains, such as formal theorem proving with verification languages like Lean, could yield insights into scaling curriculum methods where both reasoning precision and answer accuracy are required.

Additionally, examining how curriculum design interacts with different foundational model capabilities remains an important open question. Understanding whether our findings about the diminishing returns of complex curricula hold across models with varying reasoning abilities could inform more targeted training strategies and resource allocation decisions in future work.

8 Team Contributions

- **Group Member 1: Jack Hsieh** set up Dr. GRPO/vLLM, developed the final universal reverse curriculum constructor, helped formulate modification strategies, and ran half of experiments. Originally, he wrote code adapting to R3 before these were deprecated in favor
- **Group Member 2: Dillon Nguyen** helped formulate modification strategies, wrote curricula recipes, helped prepared data and RL execution jobs, and ran the other half of the experiments. Originally, for our milestone, he wrote data generation code, an SFT script, and a wrapper script for Dr. GRPO library, in addition to running forward curricula scripts.
- **Group Member 3: Ethan Zhang** wrote the original curriculum construction scripts and custom dataloaders. Originally he was responsible for tagging the data with R^3 -Hindsight but instead focused on constructed the base constructor script for all our experiments instead of building on top of the algorithm.

All members contributed to writing the report.

Changes from Proposal: Many of our proposed modifications were ideas taken from the continuous environment of robotics, and upon attempting implementation, we discovered that these techniques were difficult/inappropriate to translate to the discrete and less regenerable domain of language. As such, we pivoted our modifications to focus specifically on the staging of data.

References

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. 2021. Training Verifiers to Solve Math Word Problems. arXiv:2110.14168 [cs.LG] https://arxiv.org/abs/2110.14168

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. 2018. Reverse curriculum generation for reinforcement learning. https://arxiv.org/abs/1707.05300

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. arXiv:2309.06180 [cs.LG] https://arxiv.org/abs/2309.06180

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding R1-Zero-Like Training: A Critical Perspective. arXiv:2503.20783 [cs.LG] https://arxiv.org/abs/2503.20783

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-Math: Unlocking the potential of SLMs in Grade School Math. arXiv:2402.14830 [cs.CL] https://arxiv.org/abs/2402.14830

Stone Tao, Arth Shukla, Tse kai Chan, and Hao Su. 2024. Reverse Forward Curriculum Learning for Extreme Sample and Demonstration Efficiency in Reinforcement Learning. arXiv:2405.03379 [cs.LG] https://arxiv.org/abs/2405.03379

Zhiheng Xi, Wenxiang Chen, Boyang Hong, Senjie Jin, Rui Zheng, Wei He, Yiwen Ding, Shichun Liu, Xin Guo, Junzhe Wang, Honglin Guo, Wei Shen, Xiaoran Fan, Yuhao Zhou, Shihan Dou, Xiao Wang, Xinbo Zhang, Peng Sun, Tao Gui, Qi Zhang, and Xuanjing Huang. 2024. Training Large Language Models for Reasoning through Reverse Curriculum Reinforcement Learning. arXiv:2402.05808 [cs.AI] https://arxiv.org/abs/2402.05808

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-Math Technical Report: Toward Mathematical Expert Model via Self-Improvement. arXiv:2409.12122 [cs.CL] https://arxiv.org/abs/2409.12122

A Further Implementation Details

Originally, we attempted to work upon the codebase provided by Xi et al. (2024). However, the codebase was extremely difficult and slow to work with, and so we opted to implement our baselines from scratch for control. We then wrote a significant amount of code for the milestone and presentation for leveraging external library for GRPO. However, despite all our efforts, their implementation was impractically slow and unwieldly. As such, we decided to instead perform RL by modifying code written previously for another class. We emphasize that our main code contribution is not the re-used code; we are simply using this code to perform GRPO at a tolerable speed. Our code contributions are the constructions of our datasets, logging, and recipe scripts.

Our initial experiments used the smaller Qwen2.5-0.5B Base model. However, its limited instruction-following and reasoning abilities resulted in poor reward signals throughout training, with performance below 5% for all fine-tuning methods. At the time, our use of an off-the-shelf GRPO implementation poorly utilized memory efficiency, necessitating use to smaller models. By replacing this with our custom GRPO implementation, we were able to scale up to the 1.5B model for our main experiments.

In our initial experiments, we used the GSM8K dataset (Cobbe et al., 2021). GSM8K is a well-known benchmark for grade-school math problems with several key features: (1) it is of manageable size that fits within our compute and model constraints; (2) the difficulty is non-trivial but accessible enough to sanity-check; and (3) it includes golden rationales for each solution. However, we found that these rationales are relatively short and involve few reasoning steps, limiting their utility in constructing a reverse curriculum.

In early experiments, we applied a partial reward (0.2) for correct answer formatting, but this was removed in main experiments, as the larger 1.5B model reliably learns proper formatting regardless, and eliminating the partial reward reduces the risk of reward hacking.

While our initial experiments used the DeepSeek-r1-zero prompt template, we found that directly using the question as the prompt and answer format `\boxed\{answer\}` is simpler and works just as well, in agreement with prior work (Liu et al., 2025).