Heuristics Considered Harmful: RL With Random Rewards Should Not Make LLMs Reason

Owen Oertell, Wenhao Zhan*, Gokul Swamy, Zhiwei Steven Wu, Kiante Brantley, Jason Lee, Wen Sun*

Recent work has shown that for particular combinations of base model and training algorithm, *reinforcement learning with random rewards* (RLRR) improves the performance of LLMs on certain math reasoning benchmarks. This result is surprising as the (expected) policy gradient is *exactly* zero for RLRR, as all policies look the same under a random reward function. In response, we use RLRR as a *diagnostic task* for evaluating how well different classes of RL algorithms follow this true policy gradient. First, we demonstrate that algorithms that follow the (natural) policy gradient (e.g. RLoo (Kool et al., 2019) or REBEL (Gao et al., 2024)) produce the expected behavior of performance staying flat with random rewards, only increasing when provided with ground-truth rewards. Second, we show that rather than holding steady, *heuristic* policy gradients like PPO (Schulman et al., 2017) and GRPO (Shao et al., 2024) can either increase or decrease the reasoning performance of the model considerably. Third, we demonstrate than on a didactic bandit problem — a problem that has nothing to do with LLMs or reasoning — GRPO exhibits a bias towards choices that were more likely under the base policy, while the vanilla REINFORCE policy gradient (Williams, 1992) has no such tendencies. Taken together, our results underscore the importance of the choice of RL algorithm when making claims about LLM reasoning and beyond.

Code: https://github.com/Owen-Oertell/heuristics_harmful

W&B Logs: https://wandb.ai/cornell-npg/random-rewards-reasoning

Introduction

Recent work from Shao et al. (2025) shows that for certain specific base models and settings (e.g., Qwen base models with a chat template), the RL algorithm GRPO (Shao et al., 2024) can improve the performance of the policy on certain mathematical reasoning tasks, even when provided with *random rewards*. This observation is surprising as, by construction, the reward function provides no information about the quality of the actual generations. A bit more formally, when performing RL with random rewards (RLRR), *all* policies look equally good. Thus, the true policy gradient is exactly *zero* in expectation, as there is no direction one can change the policy that would change performance (Sutton et al., 1999). One should therefore expect a flat training curve averaged over seeds, rather than a clear performance change.

In response, we ablate the dependence of this phenomenon on the choice of RL algorithm used for the policy update, using RLRR as a *diagnostic task* to elucidate the behavior of different kinds of RL algorithms. Specifically, we compare two classes of algorithms. The first, *(natural) policy gradients* (e.g. RLoo (Kool et al., 2019), REBEL (Gao et al., 2024)), can be interpreted as following an unbiased estimate of the gradient of the expected performance of the policy with respect to the policy parameters. The second, *heuristic policy gradients* (e.g. PPO (Schulman et al., 2017), GRPO (Shao et al., 2024)), are not always unbiased estimates of the true gradient due to heuristics like clipping or normalization. ***We***

provide evidence that the inclusion of these heuristics is what is leading to unexpected changes in policy performance on certain tasks.

We conduct experiments on the MATH dataset using Qwen2.5 MATH 7B models. We find that, as expected, RLoo / REBEL only change performance with the ground-truth reward and preserve policy performance with random rewards. In contrast, we find that the heuristic policy gradients PPO and GRPO exhibit the unexpected behavior of dramatically changing policy performance with random rewards. For example, GRPO improves the performance of the policy under one random seed but decreases it on another of the seeds we ran. ***Our results indicate that these heuristic policy gradients are not following the true policy gradient, and therefore should be used with caution when making and evaluating claims about LLM reasoning.***

We then attempt to further isolate the core phenomena of interest by performing experiments on a didactic bandit problem — i.e. a problem that has nothing to do with LLMs nor reasoning. Specifically, we construct a two-armed bandit problem where both arms are equally good in expectation. While the standard policy gradient REINFORCE (Williams, 1992) behaves as expected, we surprisingly see GRPO exhibit a bias towards whichever action was more likely under the base policy, echoing some findings in the appendix of Shao et al. (2025). These results imply that heuristic PGs can cause unexpected behavior, even on the simplest of RL problems.

Taken together, our results underscore the importance of the choice of RL algorithm when making or evaluating potentially surprising claims about LLM reasoning and beyond.

A Quartet of Policy Gradients

We compare four policy gradient algorithms in this note. The first, RLoo (Kool et al., 2019), is the REINFORCE policy gradient with a leave-one-out estimator for the baseline:

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \mathbb{E}_{y\sim\pi_\theta(x)}[r(x,y)] \approx \frac{1}{k}\sum_i^k \nabla_\theta \log \pi_\theta(y_i|x)\left(r(x,y_i) - \frac{1}{k-1}\sum_{j\neq i} r(x,y_j)\right)$$

Observe that this is an *unbiased* estimate of the vanilla / true policy gradient.

The second, REBEL (Gao et al., 2024), can be seen as a generalization of the Natural Policy Gradient (Kakade, 2002). REBEL solves the following regression problem at each round:

$$\arg\min_\theta \mathbb{E}_{\mathcal{D}}\left(\frac{1}{\eta}\ln\frac{\pi_\theta(y|x)}{\pi_{t-1}(y|x)} - \ln\frac{\pi_\theta(y'|x)}{\pi_{t-1}(y'|x)} - r(x,y) + r(x,y')\right)^2$$

As argued by Gao et al., 2024, this can be seen as a principled, regression-based approximation of the idealized online mirror descent update that NPG also attempts to approximate. In particular, if one solves this regression problem exactly, one will exactly implement the OMD update. If one solves this regression problem with one step of the Gauss-Newton method, it recovers the natural policy gradient update. Gao et al., 2024 also analyze how regression error translates to policy performance.

We now introduce our two *heuristic* policy gradient algorithms. The first, PPO (Schulman et al., 2017), uses clipped importance weights to enable a greater degree of off-policyness than vanilla REINFORCE, at the cost of potentially not being equal to the true policy gradient after multiple gradient steps on the same batch of data:

$$\nabla_\theta \mathbb{E}_{y\sim\pi_{t-1}}\left[\min\left(\frac{\pi_\theta(y|x)}{\pi_{t-1}(y|x)}A^{t-1}(x,y), \text{clip}\left(\frac{\pi_\theta(y|x)}{\pi_{t-1}(y|x)}, 1-\epsilon, 1+\epsilon\right)A^{t-1}(x,y)\right)\right]$$

The second, GRPO (Shao et al., 2024), simplifies the above update by removing the need for a learned advantage estimate / critic by using an advantage estimate similar to RLoo:

$$\hat{A}_{i,t} = \frac{r(x,y_i) - \text{mean}(r(x,y_1),\ldots,r(x,y_k))}{\text{std}(r(x,y_1),\ldots,r(x,y_k))}$$

GRPO still uses clipping, which means there are situations in which it will not match the true policy gradient in expectation. There are additional differences in terms of how the KL divergence to the reference policy is estimated between standard PPO and GRPO implementations — please see the above papers for more details.

LLM Reasoning Experiments

We begin by outlining our experimental setup. As in Shao et al. (2025), we use the Qwen2.5 MATH 7b model (Qwen Team, 2024). Since this is not an instruct fine-tuned model, we abstain from using a chat template. We train on the MATH dataset (Hendrycks et al., 2021) and evaluate on the commonly used MATH 500 test set (Lightman et al., 2023) We train for 1000 steps, or around 10 epochs. All models were trained with a constant learning rate of 1e-7 and $n = 16$ (16 rollouts per prompt) with a batch size of 64 prompts. Runs are averaged over 3 random seeds, other than GRPO which we averaged over 4 random seeds due to large variance.

Finding #1: RLoo and REBEL Preserve Performance under Random Rewards while Heuristic Policy Gradients like PPO and GRPO Don't.

In Figure 1, we observe that RLoo and REBEL do not learn under random rewards, as expected. This indicates that they are following the true policy gradient. In contrast, the heuristic policy gradients (GRPO, PPO) produce unexpected behavior: PPO improves performance on all three seeds, while GRPO improves it on one seed and decreases it on another. Together, these results indicate a disconnect between the true policy gradient and the updates the heuristic policy gradients make. We note that while this disconnect shook out in favor of improved performance on this particular combination of base model / task, there is no reason this should be true in general. ***An algorithm that behaves unexpectedly well in one setting can perform unexpectedly poorly in another, perhaps more important, setting.*** We therefore advocate for caution when interpreting results generated via a heuristic PG rather than the more principled RLoo / REBEL.

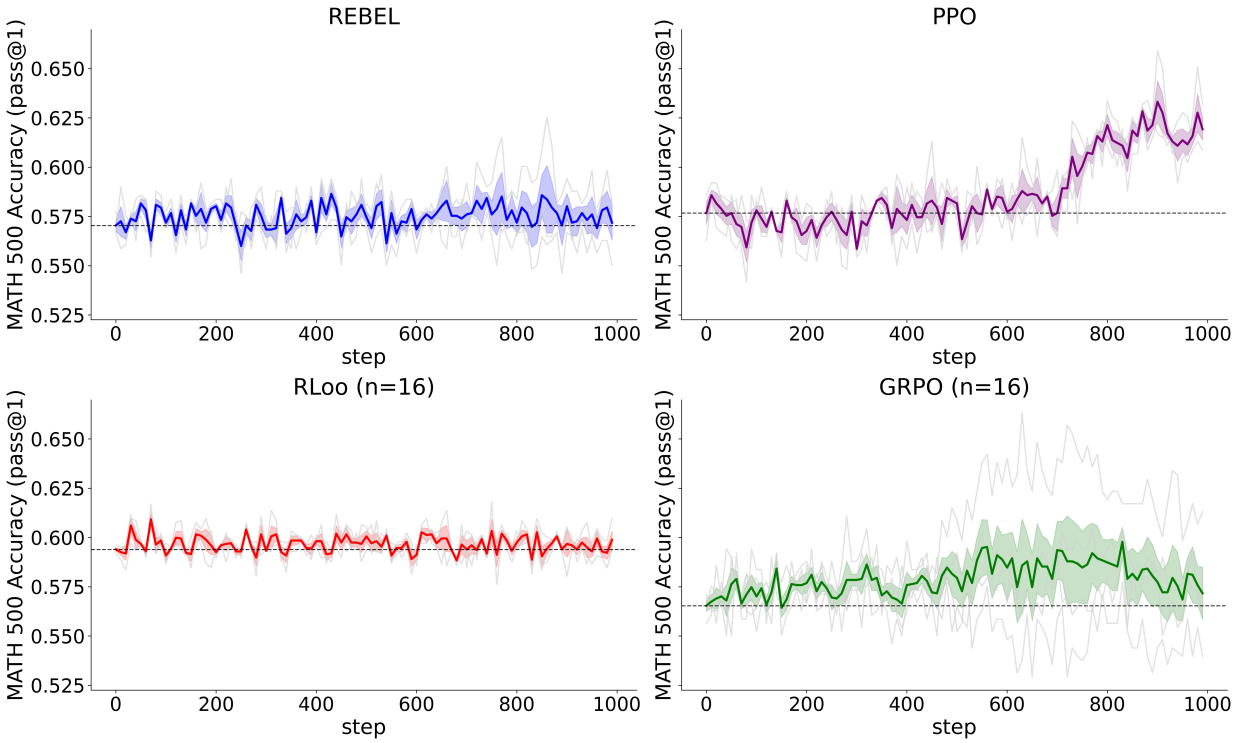

Figure 1: The accuracy of the model under random rewards for REBEL, RLoo, GRPO, and PPO. While REBEL and RLoo produce the expected behavior of preserving policy performance, GRPO and PPO instead increase (and, less commonly, decrease) policy performance. This indicates they are not following the true policy gradient, which is identically zero for this problem. Standard errors are calculated over at least three seeds for all algorithms.

Note: we observe that under training GRPO learns under all seeds (i.e., reward of the training set increases, see the linked W&B logs) and it is possible that the difference is due to the greedy sampling (temperature = 0) done at test time whereas sampling is stochastic in training (temperature = 1.0).

As a sanity check, we use the same set of hyperparameters for RLoo and REBEL under ground-truth rewards and observe increased performance, with REBEL achieving an impressive 73%.

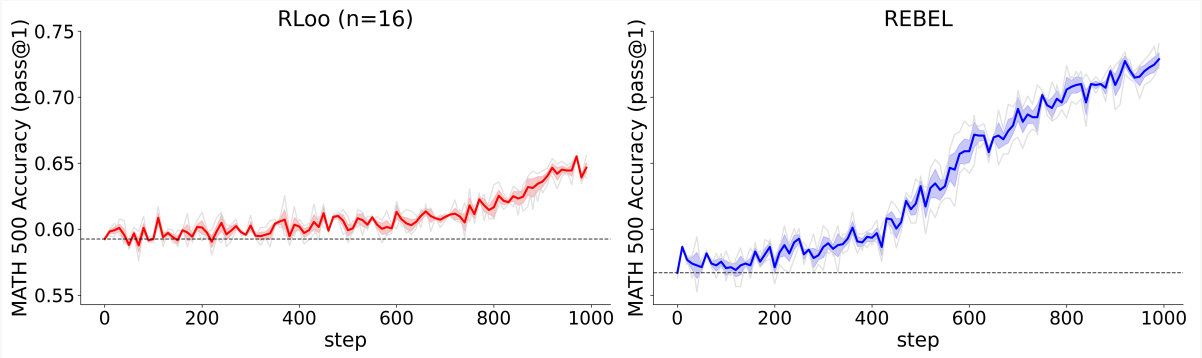

Figure 2: The accuracy of the model under ground-truth rewards for REBEL and RLoo. As expected for this sanity check, we see improved performance for both algorithms.

We note that in Figure 11 (a) of their paper, Shao et al. (2025) observe a related phenomena, where turning off GRPO clipping eliminates the unexpected performance change. However, it is unclear whether this modified algorithm would be able to learn with the ground-truth reward signal, which the above sanity check confirms for our setup. We view our results as bolstering their observation and generalizing it to other algorithms beyond GRPO without clipping.

Didactic Bandit Experiments

The preceding set of experiments includes several confounders due to the complexities of LLMs and reasoning. To further investigate the effect of policy gradient heuristics in the RLRR setting, we consider a minimal setup: a simple, two-armed bandit problem with random rewards.

Specifically, we allow the learner to pick between two arms. It then receives a random reward, uniformly sampled from $\{0, 1\}$. We parameterize the learner's policy with a single parameter z, with $\sigma(z)$ being the probability of predicting arm 0. The probability of picking arm 1 is therefore $1 - \sigma(z)$. For simplicity, we use $(r_i - 0.5)/0.5$ as the (normalized) advantage function, plugging in the exact mean (0.5) and standard deviation (0.5). The expected behavior for this problem is to stay at the initial policy, as all policies (including the initial policy) are optimal. We compare two algorithms in this setting, the vanilla REINFORCE algorithm and GRPO, below.

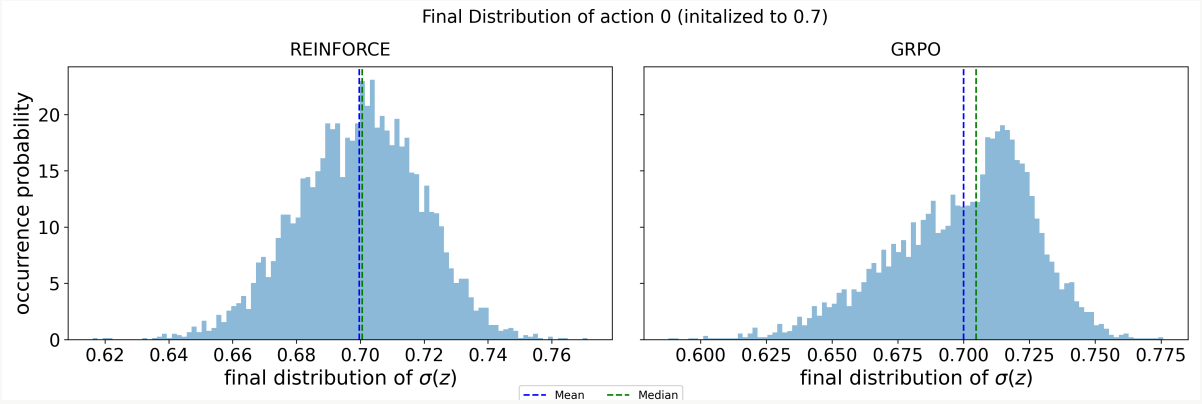

Figure 3: The final distribution of $\sigma(z)$ for the two armed bandit setting after training on random rewards for 500 steps over many random seeds. Notice that that with vanilla REINFORCE (left), the distribution remains close to normal and centered around the initialization 0.7, whereas GRPO (right) shows a clear, unexpected shift to the right.

Finding #2: GRPO Exhibits an Unexpected Bias Towards More Likely Actions Under the Base Policy, while REINFORCE Does Not.

Surprisingly, even on such a toy problem, we see GRPO exhibit a bias towards the arm that had a higher probability under the base policy, echoing the reasoning of Shao et al. (2025), who argue that clipping can induce a shift towards what is already likely under the policy. In Fig. 3, we plot the distribution of $\sigma(z)$ after updating for 500 steps. The initial parameter is set such that $\sigma(z) = 0.7$ (i.e., the initial distribution has higher probability of picking action 0). In this figure, we see that that although the mean of the distribution of $\sigma(z)$ remains close to the initial value, for GRPO, the distribution's median shifts towards the action that has higher probability under the base model. REINFORCE, in contrast, keeps both statistics close to their initial values.

Finding #3: Smaller Clipping ϵ in GRPO Leads to Less Skew.

We ablate the clipping threshold, ϵ , in the above experiment and find that as we scale down ϵ , the skew induced by the GRPO update decreases. This corresponds to making the update increasingly on-policy. In the limit of $\epsilon \to 0$, we would recover the standard policy gradient (up to the baseline). We would therefore match the REINFORCE results above, which don't show a change in the policy. These results provide evidence that for this particular bandit problem, it is the clipping which is causing the unexpected skewing of the distribution. We note that Shao et al. (2025) ablates clipping in the LLM reasoning context, observing that the unexpected shifts disappear when clipping is removed. However, our results show this phenomena occurs even on problems far simpler than LLM reasoning. We leave a more thorough investigation of this phenomena on other problems (both small-scale and large-scale) to future work.

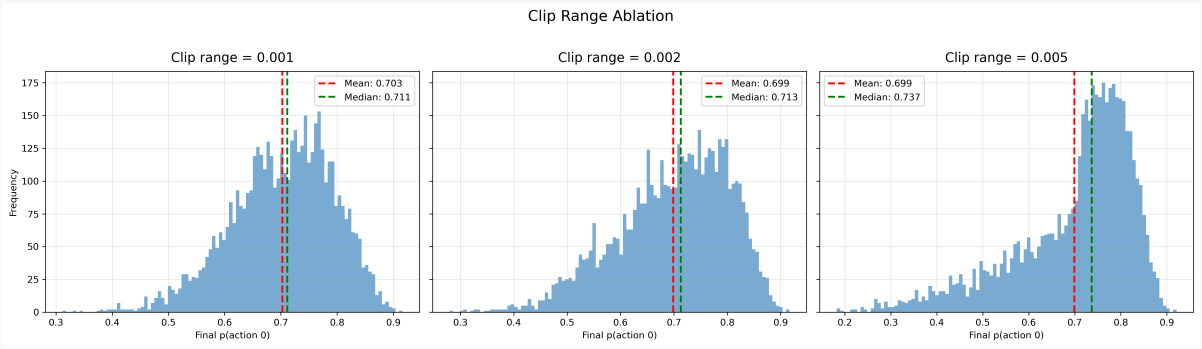

Figure 4: We ablate the clipping threshold, ϵ in the GRPO formulation. As ϵ is decreased (i.e. we approach vanilla REINFORCE), we see that the skew in the distribution decreases.

Conclusion

By using the RLRR setting as a testbed to evaluate four RL algorithms, we see clear patterns emerge. RLoo and REBEL — which have clear interpretations as following the true policy gradient — exhibit the expected behavior in all cases. In contrast, the heuristic policy gradients GRPO and PPO can lead to drastic policy changes, even when the provided reward function incentivizes none. We see the same story play out on a simple didactic bandit problem, which has nothing to do with LLMs nor reasoning, indicating a fundamental disconnect between expectation and reality for heuristic policy gradients. We therefore recommend caution when making or evaluating claims about LLM reasoning and beyond with GRPO and PPO, ideally using algorithms like RLoo or REBEL instead.

Citation

If you found our work useful, feel free to cite it as follows:

```
@misc{oertell2025heuristicsconsideredharmful,
    title={Heuristics Considered Harmful: RL With Random Rewards Should Not Make LLMs Reason},
    author={Owen Oertell and Wenhao Zhao and Gokul Swamy and Zhiwei Steven Wu and Kiante Brai
    year={2025},
    url={https://fuchsia-arch-d8e.notion.site/Heuristics-Considered-Harmful-RL-With-Random-Rewai
}
```