Don't Let Gains FADE: Breaking Down Policy Gradient Weights in RL

Juliette Decugis[1,2] Sean O'Brien[1] Francis Bach[2] Gabriel Synnaeve[1] Taco Cohen[1]
[1]FAIR at Meta [2]Inria, École Normale Supérieure

Problem

Question: which advantage to use, and when?

The REINFORCE update $r\,\nabla_\theta \log \pi_\theta(\tau)$ is noisy when estimated from a few rollouts, especially over long trajectories. Subtracting a baseline, $(r - b)\,\nabla_\theta \log \pi_\theta(\tau)$, keeps the gradient unbiased [1] and reduces its variance. The variance-optimal baseline is [2,3]

$$b^* = \frac{\mathbb{E}\left[r\,\|s\|^2\right]}{\mathbb{E}\left[\|s\|^2\right]}, \qquad s = \nabla_\theta \log \pi_\theta(\tau),$$

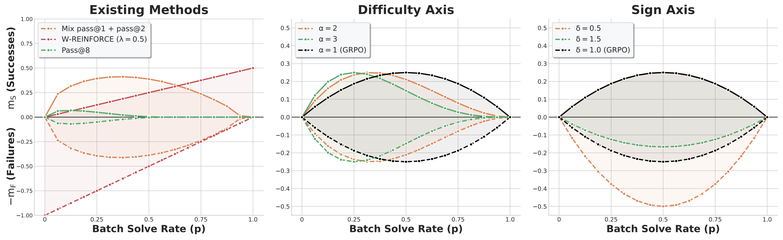
which reduces to the standard choice $b = \mathbb{E}[r]$ only under three assumptions that fail in reasoning RL:

- Successes and failures look alike $\Rightarrow$ they differ in entropy and length [4,5].
- We only care about accuracy (pass@1) $\Rightarrow$ we care about pass@k [6,7].
- Unbiasedness beats lower variance $\Rightarrow$ with very long sequences, trading some bias for lower variance can be worth it [8,9].

Advantages as gradient weights

We focus on advantages $A = c(p)\,(r - b)$, where $p := \mathbb{E}[r]$ is the prompt's solve rate over G rollouts. With $r \in \{0, 1\}$, the gradient splits into positive and negative updates:

$$g = \underbrace{c(p)}_{\text{which prompts}} \underbrace{\left[(1-b)\,p\,\nabla_S - b\,(1-p)\,\nabla_F\right]}_{\text{which sign}} = c(p)\,\nabla_\theta p.$$



Existing methods mix both axes (left); Power α isolates difficulty (middle), AsymGRPO sign (right).

What each advantage predicts

Method	$c(p)$	Sign	Entropy	pass@k slope
REINFORCE ($r \in \{0,1\}$)	1	positive only	collapses	falls
Dr. GRPO, RLOO	1	balanced	stable	falls
GRPO	$1/\sqrt{pq}$	balanced	stable	falls
MaxRL	$1/p$	balanced	stable	$\approx$ flat
Log-mean-exp	$\propto \frac{1}{1+p(e^\beta-1)}$	balanced	stable	falls
ReLU	q	positive only	collapses	falls
Pass@k	$k\,q^{k-1}$	positive only	collapses	flat to rising
W-REINFORCE	$1+\lambda$	failure-biased	rises, drifts	falls
AsymRL	1	failure-biased	mild drift	falls
CoRPO	1	failure-biased if $p<r_{\min}$	mild drift	falls
MC-GRPO	1	flips at $p=\frac{1}{2}$	self-correcting	falls
Power α	$q^{\alpha-1}$	balanced	stable	flat at $\alpha=2$
AsymGRPO	$q + p/\delta$	$m_S/m_F=\delta$	$\uparrow$ if $\delta<1$, $\downarrow$ if $\delta>1$	falls
FADE	adaptive	adaptive	held near H^*	flat

Yellow rows are our methods. Entropy and slope are predicted from each method's sign balance and focus.

Experimental setup

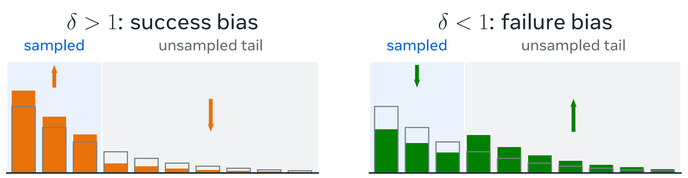
- **Models:** Qwen 2.5 7B [10], CWM 32B [11], Qwen3.5-35B-A3B [12].
- **RL Data:** 25k CodeContests [14] + TACO [15] problems, and R2E-Gym [16] for SWE.
- **Eval**: LiveCodeBench [17], AIME 2024/25 [18], SWE-bench Verified [19].

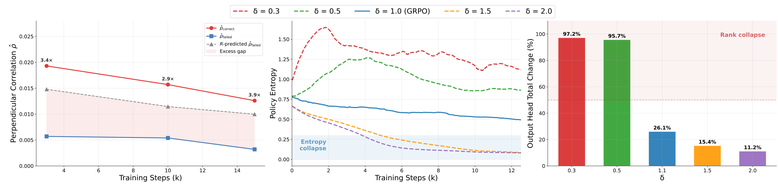
Sign axis: balance controls overfitting

Takeaway: Entropy is proportional to the fraction of positive to negative gradient mass.

Under Adam, pushing a sampled token down lifts every unsampled token by the same step. AsymGRPO scales failures by $1/\delta$, so $m_S/m_F = \delta$ for every prompt:

$$A_i^{\text{AsymGRPO}} = \begin{cases} A_i, & A_i \geq 0 \\ A_i/\delta, & A_i < 0 \end{cases}$$





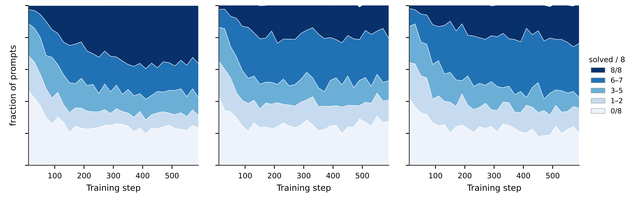
Qwen 2.5 7B: successes are more correlated (left); $\delta > 1$ collapses entropy (middle); $\delta < 1$ gives a rank-1 update (right).

Difficulty axis: who learns fastest decides pass@k

Takeaway: easy prompts learn fastest and flatten pass@k; balance learning speed across difficulties.

Pass@$k = \mathbb{E}_x\left[1 - (1 - p_x)^k\right]$ [20] probes prompts near $p \approx 1/k$. With a power-law hard tail it follows a scaling law [21,22] whose slope b decides who wins at large k:

$$1 - \text{pass@}k \;\propto\; k^{-b}.$$



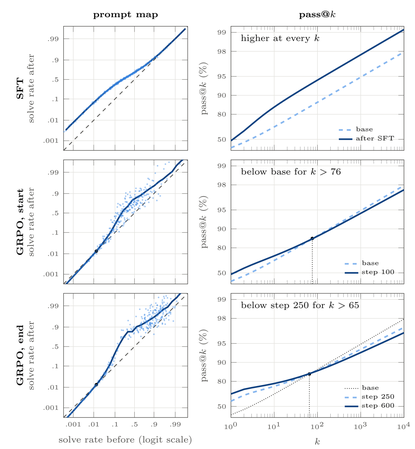
SWE prompts by solve rate: Dr. GRPO leaves the hardest behind; Power α=2 and MaxRL move them.

A predictive law for pass@k

Each prompt's log-odds $\ell = \ln \frac{p}{1-p}$ moves at speed $u_c(p) = c(p)\,p\,(1-p)$ per step. Over a window of training:

$$\ell_{\text{after}} = \underbrace{\Lambda}_{\text{shift}} + \underbrace{\nu}_{\text{stretch}}\,\ell_{\text{before}} + \underbrace{\sigma z}_{\text{noise}}$$

The shift Λ lifts every k and the noise σ only the level, so the stretch ν alone sets the slope, $b \to b/\nu$, and it is predicted from u_c with no free parameters.

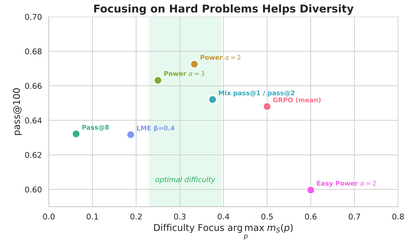


Simulated prompt maps and pass@k: SFT lifts the hardest prompts and raises every k; Dr. GRPO moves the middle and falls below the base at large k.

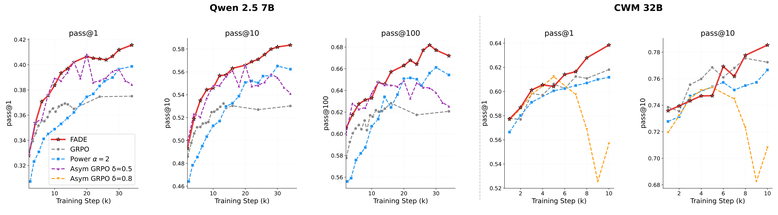
FADE: Focal Advantage with Dynamic Entropy

FADE adapts both axes online:

1. Focus on hard problems first, then relax towards medium ones as prompts get easier.
2. Target a set entropy H^* using the gradient sign: add failure bias only when entropy drops below it.



$$A_i = (1-\bar{r})^{\alpha-1} \begin{cases} r_i - \bar{r} & r_i \geq \bar{r} \\ \dfrac{r_i - \bar{r}}{\delta} & r_i < \bar{r} \end{cases} \qquad \alpha = \tfrac{1-\hat{p}}{\hat{p}},\;\; \delta = \text{clip}\left(1 + \hat{H} - H^*,\, 0.3,\, 1\right)$$



LiveCodeBench: best pass@100 at $p \in [0.25, 0.4]$ (top); FADE vs. the best static advantages (bottom).

References

[1] Williams. REINFORCE. Mach. Learn., 1992.
[2] Greensmith et al. Variance reduction for policy gradients. JMLR, 2004.
[3] Weaver & Tao. The optimal reward baseline. arXiv, 2013.
[4] Zhu et al. Negative reinforcement in LLM reasoning. 2025.
[5] Arnal et al. Asymmetric REINFORCE. 2026.
[6] Tang et al. Inference-time objectives with RL. 2025.
[7] Chen et al. Pass@k training. 2025.
[8] Shao et al. DeepSeekMath (GRPO). 2024.
[9] Andrychowicz et al. What matters in on-policy RL. ICLR, 2021.
[10] Hui et al. Qwen2.5-Coder. 2024.
[11] FAIR CodeGen. CWM. 2025.
[12] Qwen Team. Qwen3.5. 2026.
[13] Schulman et al. PPO. 2017.
[14] Li et al. AlphaCode (CodeContests). Science, 2022.
[15] Li et al. TACO. 2023.
[16] Jain et al. R2E-Gym. 2025.
[17] Jain et al. LiveCodeBench. ICLR, 2025.
[18] OpenAI. Learning to reason with LLMs. 2024.
[19] Jimenez et al. SWE-bench. 2024.
[20] Chen et al. Evaluating LLMs trained on code. 2021.
[21] Brown et al. Large language monkeys. 2024.
[22] Schaeffer et al. How do LLM monkeys get their power laws? ICML, 2025.