HALF THE ROLLOUTS, NO MEASURABLE COST: A SEEDED AUDIT OF DYNAMIC SAMPLING AND CREDIT ASSIGNMENT IN DAPO

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning for mathematical reasoning is routinely evaluated on AIME, where a single problem is worth over three points, and improvements of one to two points are commonly reported from a single training run. We ask whether such improvements are measurable, and answer with a pre-registered, seeded audit of DAPO on Qwen2.5-Math-7B. Three seeds of the unmodified baseline, read out at 128 samples per problem on AIME 2024 and 2025 pooled, span 1.96 points with a standard deviation of 0.99; across thirteen single-seed runs the rank correlation between the two AIME years is -0.03. Against that unit, we built nine variants: token-level credit-assignment methods that reweight advantages by pivotality, add contrastive step rewards, or importance-weight positions by future influence, and a prompt selector driven by a tracked difficulty estimate; none is individually distinguishable from the baseline under its three-seed prediction interval, and the modified-recipe runs sit on average about 1.5 points below it. A KL-targeted step-size controller, holding each variant at the baseline's late-phase KL, does not recover the largest harm, from high-amplification reweighting. The positive finding concerns the rollout budget: the 75% of prompts that DAPO's dynamic sampling discards are mostly surplus truncated once the training batch is filled; under a trained policy the mixed-group rate is 60–76%, far above the 25% implied by the discard statistic. Halving the generation batch trains the same 64 groups per step at half the rollouts and 24% less wall-clock, with a pooled accuracy difference of -0.18 points over three seeds each, inside the seed standard deviation on both sides.

1 INTRODUCTION

Group-relative policy optimisation (GRPO; Shao et al., 2024) and its DAPO refinements (Yu et al., 2025) have become the default recipe for reinforcement learning on verifiable mathematics. Much of the follow-on work changes how credit is assigned within a rollout: which tokens are amplified, which are down-weighted, what dense signal is added to the sparse outcome reward. The reported effects are typically one to two points on AIME 2024, a 30-problem benchmark on which one problem is worth 3.3 points, from one training run per method.

This paper is an audit of that practice, conducted on our own methods. Over several months we built and ran nine variants of DAPO on Qwen2.5-Math-7B (Yang et al., 2024b), eight token-level credit-assignment methods and one prompt selector, each pre-registered with its readout before any number existed. One of the nine, FIPO (Ma et al., 2026), is a published method, re-implemented at one seed; Dr. GRPO (Liu et al., 2025) enters separately as a one-flag ablation; the other eight are our own, so this is an audit of the practice as we followed it, not a replication study. None beat the baseline. Rather than report nine negative results, we asked what measurement would have been needed to know that in advance, and what survives it. The paper has one question, what a seeded measurement of a standard DAPO recipe shows, and three parts: Section 3 supplies the unit, the run-to-run variance any claim must exceed; Section 4 reports the one result that exceeds it, about the rollout budget; Section 5 reports what does not, with an instrument that rules out step size as the explanation.

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.

Contributions.

- **The measurement unit.** Three seeds of unmodified DAPO, at 128 samples per problem over three late checkpoints on AIME 2024 and 2025, have a pooled SD of 0.99 points (1.68 on AIME 2024). Three seeds bound that SD only loosely (95% interval 0.5–6.2), so single runs are compared to the baseline with a prediction interval from those seeds (± 4.9 points), inside which every run here falls. Across thirteen single-seed runs the rank correlation between the two AIME years is -0.03 (95% interval about ± 0.55), which near-equal effects under 30-problem noise predict; a contamination-free third year (AIME 2026) does not change the picture (Section 3).
- **Where the rollouts go.** DAPO generates four times the prompts it trains on and discards 75%. By direct count, most discarded groups are usable ones cut off once the training batch is full: 60–76% of generated groups have mixed outcomes under a trained policy. Halving the generation batch trains the identical 64 groups per step at 50% of the rollouts and 76% of the wall-clock, with a three-seed pooled accuracy difference of -0.18 points (95% interval -2.4 to $+2.1$). The contribution is the measured rate and the sizing rule it gives (generation batch $\times$ mixed rate $\geq$ training batch), whose predicted breaking point we confirm and which holds across three models (Section 4).
- **Credit assignment under a step-size control.** We give a five-slot taxonomy of where an update can be modified, observe that PPO's ratio clip is inert in this configuration, so a reweighting that changes the realised policy step meets no resistance, and introduce a KL-targeted step controller as an *instrument* that holds the realised KL at the baseline's late-phase value. Under it the reweighting methods still do not beat DAPO, and holding the step recovers none of the largest loss, a decline with amplification strength that only AIME 2025 shows (Section 5).

The methods are specified in full in the appendix, transcribed from the code that produced every number here.

2 Setup and evaluation protocol

Training. All runs fine-tune Qwen2.5-Math-7B on DAPO-Math-17k with the DAPO recipe of Yu et al. (2025) as implemented in an open RL framework (Sheng et al., 2025) with vLLM rollouts (Kwon et al., 2023): $G = 32$ responses per prompt, a training batch of 64 prompts, generation batch of 256 prompts, asymmetric clip $[0.2, 0.28]$, dual clip (Ye et al., 2020) at 10, token-mean loss aggregation, no KL penalty, prompts up to 2,048 tokens and responses up to 10,240 with overlong-response shaping over the last 2,048, learning rate 10^{-6} with Adam, eight mini-batch updates per step, on one node of eight 80GB accelerators. Dynamic sampling keeps only groups whose 32 outcomes are not all identical and regenerates until 64 such groups are available. Runs are capped at 400 generation rounds; the budget counts every generation batch, including the extra ones dynamic sampling requests, so the number of optimiser steps is 400 minus the extra batches: 379–395 for the Qwen2.5-Math-7B runs at batch 256 or 128, 400 for the general-7B run, 371 for the 1.5B run, and 296 for the 96-prompt run. The baseline objective and every variant's departure from it are written out in Appendix D.

Readout. Training-time validation uses 32 samples per problem ("mean@32"), whose noise we measured at ± 1.4 points by evaluating one checkpoint twice. Every number reported here instead uses 128 samples per problem at temperature 1.0, top-p 0.7, averaged over checkpoints 320, 350, and 380. We report three views: AIME 2024, AIME 2025, and the *pooled* score on the combined 60 problems, which is the mean of the two and costs nothing extra. AIME 2025 post-dates the base model's release; AIME 2024 does not. We do not claim contamination, but the two years are the only independent replication of each other this scale of experiment affords. A fourth view, AIME 2026 (30 problems, held seventeen months after the base model's release, answer key checked against two transcriptions), was evaluated last on the same checkpoints for every run and is reported as a separate column, not folded into the pooled score, so that every pooled number remains comparable to the pre-registered readouts.

Table 1: Three seeds of unmodified DAPO, mean@128 at steps 320/350/380. The last row is the standard deviation across seeds, which is the unit every other number in this paper is measured in.

DAPO seed	AIME 2024	AIME 2025	Pooled 60
1	33.48	18.91	26.20
2	31.08	17.40	24.24
3	34.31	16.64	25.48
mean	32.96	17.65	25.31
SD	1.68	1.16	0.99

Pre-registration. Each experiment's readout, comparison, and interpretation branches were written into a dated internal ledger before the run started and are reproduced in the appendix. The ledger was not deposited with a third party, so a reader cannot verify the dates. Where a pre-registered reading was later overturned by a control, both are reported.

3 SEED VARIANCE IS THE UNIT OF MEASUREMENT

Until its final week every comparison in this study was one run against one run, with a 0.2–0.4 point error bar across late checkpoints of the same run, which contains no run-to-run variance. We measured that variance by training the unmodified baseline three times, changing only the data-shuffle seed.

Seeds of the same recipe differ by up to 1.96 pooled points and 3.23 on AIME 2024 (Table 1), more than every effect any of our methods produced and more than most effects reported in the literature this paper responds to; the checkpoint error bar under-estimates the uncertainty by a factor of three to five. Figure 1 places every single-seed run against that band: all fourteen fall inside the three-seed prediction interval defined below, and two fall outside the narrower pooled-variance interval used as a sensitivity check.

The unit is itself uncertain. An SD from three seeds has two degrees of freedom and a 95% confidence interval of 0.52–6.2 points pooled. Even the lower end exceeds the checkpoint error bar, so the previous paragraph stands, but a rule that treats 0.99 as known does not, and we do not classify runs by z-scores. Whether a single run is consistent with the baseline is a prediction interval: a new baseline run differs from the three-seed mean by at most $t_{0.975,\nu}\, s\sqrt{1+1/3}$ with probability 0.95. With the three DAPO seeds alone ($\nu = 2$) the half-width is 4.9 points pooled and 3.1 on AIME 2026, and no run in this paper is outside either; that is the rule we use, because it involves no choice. A narrower interval follows if the halved-batch recipe of Section 4 is assumed to share DAPO's variance (its SD is 0.37; an F-test on $(2, 2)$ degrees of freedom does not reject equality, but has almost no power): pooling gives $s = 0.75$, $\nu = 4$, half-widths 2.40 and 1.48. Neither interval was pre-registered, so the pooled one is reported only as a sensitivity check (Appendix A.7); no claim below depends on it.

Rank agreement between years. Across the thirteen single-seed runs of Table 3 (eleven variant runs and the first seed of each baseline), the Spearman rank correlation between AIME 2024 and AIME 2025 accuracy is -0.03; AIME 2026 correlates $+0.21$ with 2024 and $+0.58$ with 2025. At $n = 13$ each value carries a 95% interval about ± 0.55 wide and moves with the run set (Appendix A.8), so the three are not distinguishable. We do not read this as the years measuring different things: if the runs' true accuracies are nearly equal, which is what the seeds say, each year's ranking is mostly sampling noise on thirty problems, two such rankings correlate near zero, and the run first on one year is as likely as any other to be last on the next. We recorded six such inversions over the study, each after a method had looked good on the year we were watching; they are the expected product of near-equal effects and 30-problem noise. The one hint of signal, uncorrected for three comparisons, is the positive agreement between 2025 and 2026, which 2024 does not share.

Reference and decision rule. The DAPO seed mean is the reference. A single-seed run is reported with its difference from the mean, also as $z = (x - 25.31)/0.99$ for readability, and is read as

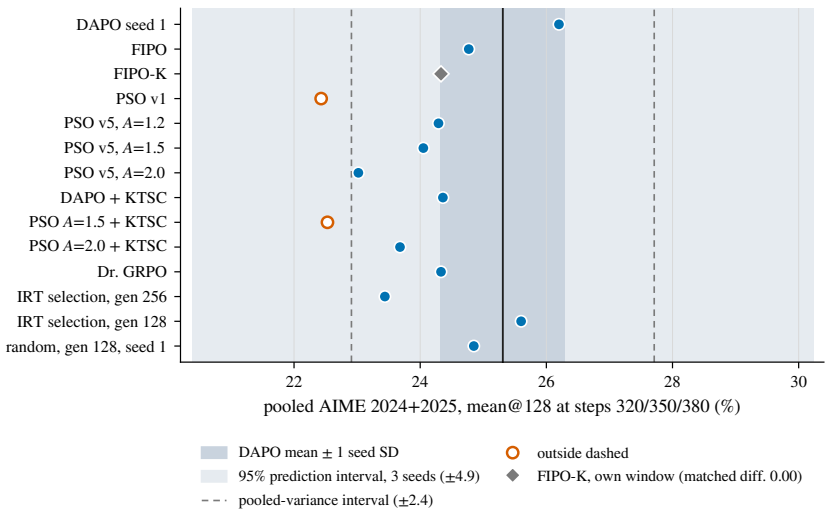

Figure 1: Every single-seed run in the study (Table 3), pooled AIME 2024+2025 mean@128, against the three-seed DAPO mean (line), with one seed SD (dark) and the 95% prediction interval from DAPO's three seeds (±4.9, light) shaded; every run lies inside it. Dashed lines mark the narrower pooled-variance interval (±2.4) used as a sensitivity check; the two hollow points outside it are PSO v1 ($A = 2.0$) and PSO $A = 1.5$ under the step controller. FIPO-K (diamond) is plotted at its own window (steps 200–280), where matched DAPO reads 24.33, so its difference from its own reference is 0.00 and it is not compared to the band.

"outside noise" only beyond the three-seed prediction interval (4.9 pooled, 3.1 on AIME 2026); exceedances of the sensitivity interval (2.40, 1.48) are reported as such. Where the twelve variant runs of Table 3 (every modified-recipe run, including the controlled runs and the Dr. GRPO ablation; Appendix A.7) are considered together, we count how many fall below the reference; they share DAPO seed 1's data order and several are dose levels or controlled twins of one method, so we give the count and a family-level version rather than a p-value that assumes independence. Only the generation-batch comparison of Section 4 has three seeds on both sides, and even there the interval is wide.

4 Where the rollouts go

4.1 What the discard statistic measures

DAPO's dynamic sampling exists because uniformly-correct and uniformly-wrong groups have zero advantage everywhere and contribute no gradient. With a generation batch of 256 and a training batch of 64, a full run retains 24.8% of generated prompts and discards 75.2%. We built a method on that number, a prompt-selection scheme that tracks each prompt's solve rate with a discounted Beta posterior and preferentially selects prompts likely to yield a mixed group (Appendix L), on the premise that most discarded prompts were problems the model could not yet solve at all.

The premise was wrong, and the method's own instrumentation showed it. The order of events matters: the truncation was found after the fact, in the diagnostics of a failed method, and is exploratory; everything after it (the random-selection control at 128, the second and third seeds on both sides, the prediction at 96, the AIME 2026 readout) was fixed before its numbers existed. To score prompts we had to count, for every generated group, whether its outcomes were mixed. Under the trained policy, over steps 100 onward and across the three seeds at generation batch 128, the per-step count ranges from 55% to 81% of all groups, and 80% of steps fall between 60% and 76%, the interdecile range we quote as the plateau. The trainer keeps the first 64 mixed groups it finds and discards the rest, mixed or not; with 256 prompts at a 70% mixed rate it finds about 180 and throws away 116 of them. The 24.8% was $64/256$. Consistently, the baseline never needed a second generation batch

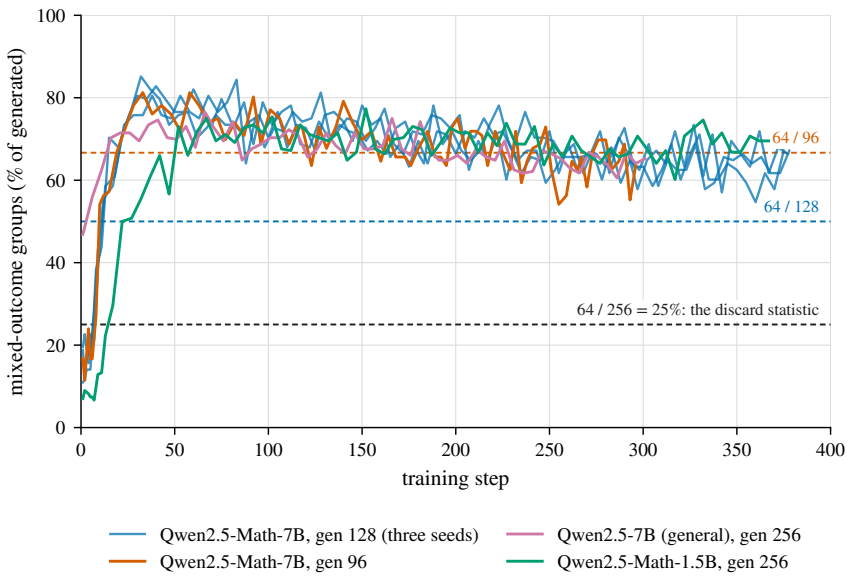

Figure 2: Fraction of generated groups with mixed outcomes, counted directly at every generation, for every run that carried the counter, all with uniformly random prompt selection. Dotted lines mark the fraction at which a generation batch of 256, 128, or 96 prompts yields the 64 mixed groups the trainer needs. Three models of different size and pretraining settle in overlapping bands (interdecile ranges over steps 100 onward: 60–76%, 63–71%, 66–73%); only the first 20–50 steps differ.

Table 2: Generation batch 256 (DAPO) versus 128, three seeds each, mean@128 at steps 320/350/380; $\pm$ is the SD across seeds. After step 20 no run needed a second generation batch, so prompts/step is the nominal batch (each prompt is 32 rollouts). Wall-clock is per training step over steps 100–300, averaged over the three seeds on each side (a single-run pairing in Appendix L read -30%).

Generation batch	AIME 2024	AIME 2025	Pooled 60	AIME 2026	Prompts/step	min/step
256 (DAPO)	32.96 ± 1.68	17.65 ± 1.16	25.31 ± 0.99	14.54 ± 0.62	256	11.2
128	33.81 ± 1.05	16.46 ± 1.02	25.13 ± 0.37	14.35 ± 0.21	128	8.5
difference	$+0.85$	-1.19	-0.18	-0.19	-50%	-24%

after its first seven steps in any seed, and no seeded run at either batch size needed one after step 20. Figure 2 shows the measured rate over training for every run that carried the counter.

4.2 Halving the generation batch

If the mixed rate is 70%, 128 prompts yield about 90 mixed groups against the 64 the trainer needs, so one generation batch of half the size should fill every step. We ran this configuration three times with uniformly random prompt selection (the difficulty tracker disabled, so that the only change from DAPO is the generation batch size), and the baseline three times.

The pooled difference of -0.18 points is smaller than either side's SD (Table 2), but three seeds per side leave it wide: the 95% confidence interval (Welch, $\nu \approx 2.5$) is -2.4 to $+2.1$ points. As two one-sided tests, the 90% interval -1.8 to $+1.4$ establishes equivalence at a margin of 2 points and fails at 1.5; a cost of about a point is not excluded, and we do not claim it is. The two AIME years move in opposite directions by about a point each, inside their own SDs; with two seeds per side they had looked like a rank inversion, and the third seed put them back inside noise. On AIME 2026 the difference is -0.19 (95% interval -1.6 to $+1.2$). The design settles what the readout cannot: the trained batch is the same, the first 64 mixed groups found from the same prompt distribution, so only surplus generation is removed, which is also why the wall-clock saving (24%) is smaller

than the rollout saving (50%). The two savings differ in kind: halving the rollouts follows from the batch sizes and holds on any hardware; the wall-clock figure depends on how much of a step this infrastructure spends generating. "No measurable cost" in our title means exactly this: the saving is a design fact, and the accuracy cost, if any, is below what three seeds per side can measure. Equal accuracy is therefore close to expected and is not the finding; the finding is the measured rate that makes the argument applicable, since the discard statistic implies the opposite. The check is still not a formality, because the two settings draw on the prompt stream at different rates, so their data schedules over a run differ.

Breaking point of the halving. The arithmetic predicts a breaking point: 96 prompts at a 60–76% mixed rate yield 58–73 mixed groups against 64 required, so at 96 the second generation batch should start to appear. We pre-registered that prediction and ran one seed at 96. After step 20 the second batch fired at 26% of steps, against 0% at 128 in all three seeds, giving 121 generated prompts per step (Appendix A, Figure 3). The saving holds down to the batch size the arithmetic predicts. This run is a mechanism point; its single-seed accuracy is in the same appendix.

4.3 The rate across model size and specialisation

To test whether the mixed rate, and with it the halving, is a property of Qwen2.5-Math-7B or of the training regime on this data, we ran the same recipe with the counter on two further models of the same family, at generation batch 256, one seed each, random selection: Qwen2.5-Math-1.5B, a fifth of the size, and Qwen2.5-7B (Yang et al., 2024a), the general base model. They start from very different places (the general model at 47%, because its outputs are less confident and so more varied within a group; the small math model at 7%, needing two to four generation batches per step for its first twenty steps), and all three settle in the same band (Figure 2; Table 4 in Appendix A.5: interdecile plateaus 60–76%, 63–71%, 66–73%). On this data the plateau does not move with model size or with mathematical specialisation, so the halving transfers across those two axes with the same constant. It is a property of how hard the prompts are for the policy being trained, so a harder dataset, or a model family whose solve rates differ from Qwen2.5's, could sit lower, and the sizing rule rather than the constant is what we expect to carry over; we did not test a non-Qwen model. We had pre-registered the opposite expectation for the 1.5B model; the early readings supported it and the full trajectory overturned it. Accuracy, by contrast, differs by a factor of two across the three models (25.31, 13.71, 12.05 pooled); the plateau reflects the filter and says nothing about how much the model learns. The tracker itself saved no compute and its effect on accuracy does not survive the seeds (Appendix A.10); the surplus mixed groups that remain at 128 could be buffered rather than dropped (Appendix A.11).

5 Credit assignment under a step-size control

5.1 Five places to intervene

A policy-gradient update on a group of G trajectories can be modified in five places, in the order the quantities are computed: the prompt distribution (before generation), the per-token reward (before the advantage), the advantage itself, the importance weight (inside the loss), and the step size (after the loss, on the optimiser). Over the study we built methods in every slot (Table 6 in Appendix B); their definitions are in Appendices D–L.

5.2 The clip does not bind

PPO's clipped ratio (Schulman et al., 2017) is the only trust-region mechanism in the DAPO objective, and in our configuration it does nothing. Each 64-prompt step is one epoch of eight mini-batch updates (8 prompts, 256 responses each) at learning rate 10^{-6}; the first update of a step has ratio exactly 1 and the next seven move it very little. Over 400 baseline steps the clipped fraction of tokens is 6–10×10^{-4} and the inter-quartile range of the ratio is $[0.99999991,\ 1.0]$, below single-precision resolution. This follows from the update schedule and learning rate and is a property of this configuration. The advantage, by contrast, is pinned: with $G = 32$ and std-normalisation a group with one correct answer has $\max|\hat{A}| = 5.480$, and the logged maximum equals 5.480 at 198 of 200 late steps. In TRPO's bound (Schulman et al., 2015) $\max|A|$ multiplies the KL term; the

bound is vacuous at $\gamma = 1$ over thousands of tokens, so we take it only as a pointer to what a trust region would watch, and here that quantity is a constant set by the rarest-success groups the filter admits.

A method in the top block of Table 6 can therefore change how far the policy moves per step with nothing in the objective pushing back, and it does: PSO v5 at $A = 1.5$ realises about eight times DAPO's per-token KL, with a gradient norm of 0.4–5.2 against 0.27–0.34 (Appendix K.1). The cause is not advantage scale, to which Adam is nearly invariant and which PSO v2 and later hold at mean one per response; the candidates are gradient concentration on the up-weighted tokens, heavy-tailed weights, and their interaction with gradient clipping and Adam's slow second moment, which we did not isolate. What matters is the confound: a comparison of such a method against DAPO mixes what it did to the credit with what it did to the step.

5.3 KL-TARGETED STEP CONTROL AS A MEASUREMENT INSTRUMENT

To separate them we hold the step fixed. Adam passes the learning rate through to the update whatever the gradient's scale, and the realised per-step KL scales as η^2 for a fixed update direction, so a multiplicative controller on the learning rate,

$$m_{t+1} = m_t \cdot \mathrm{clip}\Big((\delta/\hat{\kappa}_t)^{g/2},\ 0.7,\ 1.43\Big), \qquad \eta_t = m_t\,\eta_0, \tag{1}$$

with $\hat{\kappa}_t$ a rolling median of the logged KL estimate, gain $g = 0.25$, and $\delta = 1.124 \times 10^{-4}$ equal to the baseline's own median KL over steps 181–380, holds any variant's realised KL at the baseline's late-phase value. That is not DAPO's own trajectory: early in training DAPO's KL is well below δ, so the controlled DAPO run takes 2–3$\times$ DAPO's learning rate for its first hundred steps, and the matched pair below is "PSO at δ" against "DAPO at δ". The controller's design and measured tracking are in Appendix K; it is applied to DAPO and to the PSO variants alike.

5.4 RESULTS

Table 3 collects every completed run; runs stopped early because training-time accuracy collapsed (PSO v6, the three CSR gatings, PSO+CSR v3) are listed at their peak mean@32 in Appendix A.

Position relative to the baseline. Every run in Table 3 lies inside the three-seed prediction interval (4.9 points) on the pooled view; under the sensitivity interval (2.40) two fall outside, PSO v1 and PSO $A = 1.5$ under the controller. On the first DAPO seed alone several runs had looked like 1–2 point deficits at "3–5σ" on the checkpoint error bar; the seeds say those sigmas were not the right unit. Being indistinguishable one at a time does not make the group neutral: ten of the eleven standard-window variant runs sit below the DAPO mean on the pooled view (FIPO-K, read at its own window, is a tie at 0.00 and is left out), and their mean is 23.86 (-1.45; -1.55 over the nine methods' own runs). A sign test treating the eleven as independent would give $p = 0.01$, but they are not independent (Section 3): collapsing correlated runs into five method families gives five of five below, $p = 0.06$, and the shared data order is a dependence no grouping removes, so we read the count as descriptive. It is carried by AIME 2025 (ten of eleven below, -2.3); on AIME 2024 the split is eight of eleven and on AIME 2026 five of eleven, with the variants' mean 0.12 above the baseline. On AIME 2026 the sensitivity interval places two runs above the baseline and one below; we report them without interpreting them (Appendix A.12). In summary, no variant run is distinguishable on its own; as a group they sit slightly below the baseline on the pooled view, and that deficit belongs to one AIME year more than to the methods.

The amplification sweep. The two runs outside the sensitivity interval are PSO v1 (-2.88) and PSO v5 at $A = 1.5$ under the controller (-2.78); PSO v5 at $A = 2.0$ sits at its edge (-2.29). On AIME 2025 the PSO v5 sweep is monotone in the perturbation strength: from the DAPO three-seed mean, $17.65 \to 17.20 \to 14.77 \to 13.34$ as $\mathrm{std}(w)$ goes $0 \to 0.048 \to 0.119 \to 0.239$, a span of 4.3 points in the harmful direction (5.6 against DAPO seed 1, which shares the variants' data order but is also the highest seed on this year). AIME 2024 could not see it (the same four runs span 2.1 points with no ordering), nor could AIME 2026 (13.65, 13.16, 14.67); Appendix A.9 plots both years. This is a single-year pattern of the kind Section 3 warns against: the sweep was pre-registered on AIME 2024, where it does not appear, AIME 2025 was evaluated afterwards, and only

Table 3: Every completed single-seed run, mean@128 at steps 320/350/380, against the three-seed DAPO reference, grouped by slot. z is the pooled difference from the DAPO mean in units of the DAPO seed SD (0.99); no run is outside the three-seed prediction interval (4.9 pooled, 3.1 on AIME 2026); bold marks a difference outside the narrower pooled-variance sensitivity interval (2.40 pooled, 1.48 on AIME 2026). r_{24}, r_{25} are ranks among the thirteen standard-window runs. A is PSO's amplification, $\mathrm{std}(w)$ the SD of its weights. FIPO-K is read at its own window (steps 200–280), which predates AIME 2026; its z is against DAPO at that window (24.33), hence 0.00, and it is excluded from the rank columns and from the counts in the text.

Run	AIME 2024	AIME 2025	Pooled	AIME 2026	z	r_{24}	r_{25}
DAPO, 3-seed mean (SDs in Table 1)	32.96	17.65	25.31	14.54	—	—	—
DAPO seed 1	33.48	18.91	26.20	15.06	+0.90	3	1
Importance weight							
FIPO	33.09	16.46	24.77	15.38	−0.55	5	5
FIPO-K (steps 200–280)	34.51	14.16	24.33	—	0.00	—	—
Advantage							
PSO v1, A=2.0	30.49	14.37	22.43	13.07	**−2.91**	12	11
PSO v5, A=1.2, $\mathrm{std}(w)$ 0.048	31.39	17.20	24.29	13.65	−1.03	11	3
PSO v5, A=1.5, $\mathrm{std}(w)$ 0.119	33.32	14.77	24.05	13.16	−1.27	4	9
PSO v5, A=2.0, $\mathrm{std}(w)$ 0.239	32.69	13.34	23.02	14.67	−2.31	6	12
Step size							
DAPO + KTSC	31.97	16.75	24.36	**17.20**	−0.96	9	4
PSO A=1.5 + KTSC	30.44	14.61	22.53	**13.01**	**−2.81**	13	10
PSO A=2.0 + KTSC	34.94	12.42	23.68	14.05	−1.65	1	13
Dr. GRPO (no σ)	32.66	16.01	24.33	**16.34**	−0.99	7	6
Prompt distribution							
IRT selection, gen 256	31.74	15.14	23.44	14.99	−1.89	10	8
IRT selection, gen 128	32.57	18.63	25.60	15.77	+0.29	8	2
Random selection, gen 128, seed 1	34.34	15.36	24.85	14.58	−0.46	2	7

the $A = 2.0$ controlled contrast below was pre-registered on that year; the group deficit above is also concentrated on AIME 2025, so part of the sweep may be that year's general deficit. The remaining evidence is the direction: the two runs outside the sensitivity interval and the ordering of the sweep agree, and no reading points the other way.

Matched-step comparison. The controller was built to decide this. The pre-registered interpretation: if controlled PSO lands at controlled DAPO's level, PSO's deficit was step size; if at its uncontrolled twin's level, the deficit is in the credit assignment. At $A = 1.5$, PSO at δ reads 22.53 against DAPO at δ at 24.36, a matched gap of 1.83 points in the same direction as the uncontrolled gap. That gap is between two single runs, whose difference has an SD of about 1.4, so it is inside noise; the comparison shows that holding the step did not bring PSO back (the controlled run is also below its uncontrolled twin, 24.05), not that a matched harm was measured. At $A = 2.0$ the pre-registered contrast was the AIME 2025 difference from the uncontrolled twin, with ≥ 2 points meaning "step size explains the dose harm" and ≈ 0 meaning "the reweighting itself"; it read -0.92. That design could not have separated its branches, which are only 1.2 SD apart for a difference of two single runs; pre-registration fixed how we would read the number but could not give the test more power. Holding the realised KL at the baseline's late-phase level recovers none of the loss at either amplification, within the resolution two single runs allow. DAPO under the controller itself trails uncontrolled DAPO by 0.95 points, inside the interval, and Dr. GRPO (Liu et al., 2025), which halves the realised step instead of holding it, lands within 0.03 of it (Appendix A.6).

6 Related work

Rollout efficiency. GRESO (Zheng et al., 2025) predicts from reward history which prompts will yield uniform outcomes and skips them before rollout; Sun et al. (2025) select prompts of moderate adaptive difficulty and replay recent rollouts; both report large savings. Our tracker is close in spirit to both and added nothing measurable over random selection at the right batch size. We read our

finding as a baseline for this line of work: in DAPO's own recipe much of the saving needs no predictor, estimator, or buffer, only a generation batch sized to the measured mixed rate.

Variance and credit assignment. Henderson et al. (2018) and Agarwal et al. (2021) established for control that seed variance routinely exceeds reported improvements and that point estimates from few runs mislead; Hochlehnert et al. (2025) document evaluation-side variance on reasoning benchmarks and recommend many samples per problem, which we adopt. The general message is theirs; what is new here is the size of the training-side variance for this RLVR recipe at 128 samples per problem, the prediction-interval rule built from it, and the finding that rankings across three AIME years, one contamination-free, are mostly noise at this effect size. Yue et al. (2025) argue that RL with verifiable rewards sharpens rather than extends a base model's capability, consistent with the narrow band our variants occupy. Liu et al. (2025) identify the std-normalisation bias in GRPO; we measure what removing it does to the realised step and to accuracy. Ma et al. (2026) reweight positions by their influence on the future KL of the trajectory; our FIPO and FIPO-K runs are a single-seed re-implementation at our batch size and step budget (Appendix I), not a reproduction claim, testing FIPO only in the setting of its authors' own ablation on the same Qwen2.5-Math-7B backbone; the headline claim of longer chains of thought on Qwen2.5-32B is outside what we ran and is not contradicted here. Process reward models (Lightman et al., 2024) address dense credit with learned supervision; the variants audited here attempt it from the outcome signal alone. The prompt-difficulty tracker is a discounted conjugate form of the item-response idea (Lord, 1980). Guo et al. (2025) report that distillation outperforms RL on small models; our audit is consistent with RL on a 7B model having a ceiling that credit assignment does not move.

7 Limitations and conclusion

One dataset, one model family, about 380 training steps, sixty problems per benchmark pair, three seeds. The seed SD is for this recipe on this model and is a floor, since we varied only the shuffle seed; three seeds bound it only to within a factor of six, which is why every single-run conclusion is stated through a prediction interval and the strongest statement about the variants as a class is a count. The mixed-rate mechanism was replicated on two further models, but their accuracy readouts are single-seed, and the seed variance of the credit-assignment methods themselves was not measured. The 96-prompt run is single-seed and reads lowest of the three budgets (Appendix A); whether that is a cost of the smaller budget or an early readout is unresolved, and the saving we claim as measured is the halving to 128. The rollout finding is a property of DAPO's filter-and-truncate implementation, a default that over-provisions generation, and of the mixed rate under this policy; a harder dataset with a lower plateau would move the breaking point, and the arithmetic of Section 4 says where. The negative results are for our implementations of the ideas, not for the ideas in general.

For a reader planning a similar experiment, our advice is concrete. Run the baseline three times before running the method once, and compare with a prediction interval, not a z-score on a three-seed SD. Report both AIME years and their pool. Pre-register the readout. And measure what the filter discards before building a method to rescue it.

Reproducibility statement

The appendix gives the exact form and hyper-parameters of every algorithm together with the pre-registered readout of each experiment. The training recipe, generation settings, and evaluation protocol are given in Section 2 and Appendix D. All numbers are mean@128 at named checkpoints and can be recomputed from the evaluation logs. The full code is available anonymously at `https://anonymous.4open.science/r/rrb-D899/`: the training framework, every run script named in Table 3 and Table 2, the step-size controller, the prompt selector and its mixed-group observer, the evaluation script, and the builders for the AIME 2025 and AIME 2026 evaluation sets. The same code is attached as supplementary material.

AI use statement

In this work we used a large language model as a writing assistant to aid and polish the text of this paper and its appendix: clarity, grammar, phrasing, and the wording of captions and headings. All text was reviewed and edited by the authors. Generative AI tools were not used to generate,

fabricate, or alter any experimental result; every number reported was produced by the described training and evaluation runs and read from their logs. We take responsibility for the final content of this work.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: MathArena as an evaluation platform for mathematics with LLMs. In *AI for Math Workshop at the International Conference on Machine Learning (ICML)*, 2026. arXiv:2605.00674.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *AAAI Conference on Artificial Intelligence*, 2018.

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. In *Conference on Language Modeling (COLM)*, 2025.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning (ICML)*, pp. 267–274, 2002.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *ACM SIGOPS Symposium on Operating Systems Principles (SOSP)*, 2023.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-like training: A critical perspective. In *Conference on Language Modeling (COLM)*, 2025.

Frederic M Lord. *Applications of Item Response Theory to Practical Testing Problems*. Lawrence Erlbaum, 1980.

Chiyu Ma, Shuo Yang, Kexin Huang, Jinda Lu, Haoming Meng, Shangshang Wang, Bolin Ding, Soroush Vosoughi, Guoyin Wang, and Jingren Zhou. FIPO: Eliciting deep reasoning with future-KL influenced policy optimization. *arXiv preprint arXiv:2603.19835*, 2026.

MathArena. MathArena: AIME 2026 problems and answers. `https://huggingface.co/datasets/MathArena/aime_2026`, 2026.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, 2015.

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, Y Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems (EuroSys)*, pp. 1279–1297, 2025.

Yifan Sun, Jingyan Shen, Yibin Wang, Tianyu Chen, Zhendong Wang, Mingyuan Zhou, and Huan Zhang. Improving data efficiency for LLM reinforcement fine-tuning through difficulty-targeted online data selection and rollout replay. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2.5-Math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024b.

Deheng Ye, Zhao Liu, Mingfei Sun, Bei Shi, Peilin Zhao, Hao Wu, Hongsheng Yu, Shaojie Yang, Xipeng Wu, Qingwei Guo, et al. Mastering complex control in MOBA games with deep reinforcement learning. In *AAAI Conference on Artificial Intelligence*, pp. 6672–6679, 2020.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Haizhong Zheng, Yang Zhou, Brian R. 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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

A Full results and pre-registered readouts

A.1 Runs without a plateau

Runs stopped early because training-time accuracy had collapsed are listed at their peak mean@32, since they have no late checkpoints to read out at mean@128: PSO v2 25.42% (step 110); PSO v6 (signed pivotality) 18.75% (step 60), against 24.2–26.7% for PSO v5 at the same step, a deficit far outside noise that established the absolute value in the pivotality score as necessary; CSR 17.40% (step 80); PSO+CSR v2 26.35% (step 130); PSO+CSR v4 21.88% (step 240); PSO+CSR v3 diverged near step 14 as its dynamic-sampling retry count grew without bound.

A.2 FIPO-K at matched steps

FIPO-K's plateau lies earlier than the others (steps 200–280) and declines thereafter. Over that window, mean@128 on AIME 2024 / AIME 2025 / pooled: FIPO-K $34.51/14.16/24.33 \pm 0.09$; FIPO $32.59/15.11/23.85 \pm 0.17$; DAPO at the same steps $32.19/16.46/24.33 \pm 0.20$. FIPO-K exceeds DAPO by 2.32 on AIME 2024 and trails it by 2.30 on AIME 2025, cancelling to $+0.01$ pooled. A secondary claim that FIPO-K converges faster also fails: DAPO is at 24.33 over the same window and continues to 26.20 at its own plateau, while FIPO-K declines.

A.3 Per-seed values for the generation-batch comparison

Generation batch 128 with random selection, seeds 1/2/3, pooled: 24.85 / 25.55 / 24.99 (AIME 2024: 34.34 / 34.48 / 32.60; AIME 2025: 15.36 / 16.62 / 17.39). Seed 2 ended at step 379 and was read out at $320/350/379$. Mixed-group counts per 128 generated, from every generation after step 20: 74–108, 70–106, and 73–109 across the three seeds (55–85%); over steps 100 onward,

70–104, 70–100, and 73–100 (55–81%), with 80% of generations between 77 and 97 (60–76%), the interdecile range quoted in the main text.

A.4 THE GENERATION-BATCH-96 MECHANISM POINT

Pre-registered prediction: at a 60–76% mixed rate, 96 prompts yield 58–73 mixed groups against 64 required, so the second generation batch should appear at a visible fraction of steps after step 20, where it never appeared at 128 (the prediction was registered with the slightly higher range measured at the time; the arithmetic is unchanged). Outcome: the run ended at step 296 (its extra batches count against the 400-round budget: 401 generation batches over 296 steps); after step 20 the second batch fired at 26.1% of 276 steps, mean 1.26 batches, 121 prompts per step; mixed rate 54–79% from step 100. Because the run ended before step 320, its accuracy readout uses its last three checkpoints, 280/290/296, and is single-seed: AIME 2024 30.34, AIME 2025 16.43, pooled 23.39, AIME 2026 15.10. The pooled value is 1.9 points below the late-window DAPO three-seed mean (25.31) and 1.7 below DAPO seed 1 read at its own step 280 (25.10), inside the three-seed prediction interval; it is the lowest of the three budgets, but a single seed read 40–100 steps earlier than every other row cannot separate an earlier readout from a real cost.

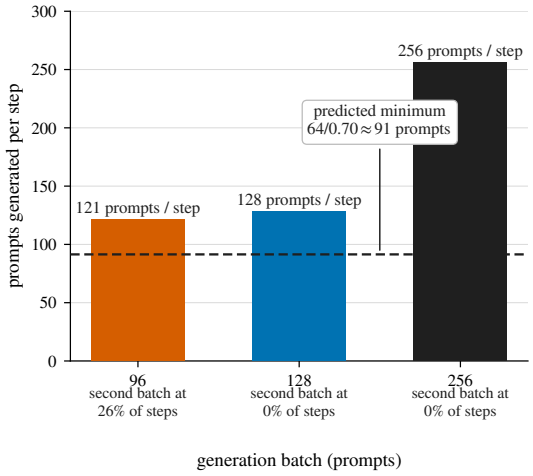

Figure 3: Prompts generated per step at three generation batch sizes, with the fraction of steps that needed a second batch (after step 20). The dashed line is the minimum the measured ≈ 0.70 mixed rate predicts. At 128 the second batch never fires; at 96 it fires at a quarter of steps.

A.5 SECOND AND THIRD MODELS

Table 4: Mixed-group rate for three models under the same recipe. "Start" is the rate at step 1; "plateau" is the interdecile range (10th to 90th percentile) of the per-step rate over steps 100 onward, with the full range in parentheses; "2nd batch" is the fraction of steps after 20 that needed a second generation batch at 256. The last column is pooled mean@128 (three seeds for Qwen2.5-Math-7B, one seed for the others).

Model	Start	Reaches 60% by	Plateau	2nd batch	Pooled
Qwen2.5-Math-7B	17–20%	step $\approx$ 20	60–76% (55–81)	0%	25.31
Qwen2.5-7B (general)	47%	step $\approx$ 5	63–71% (61–75)	0%	13.71
Qwen2.5-Math-1.5B	7%	step $\approx$ 45	66–73% (60–77)	0%	12.05

Qwen2.5-Math-1.5B, generation batch 256, random selection, one seed, ended at step 371. Mixed groups of 256: 18–34 (7–13%) over steps 1–11, 57–169 over steps 13–50, 154–198 (60–77%) thereafter. Generation batches per step: mean 2.45 (max 4) over steps 1–20, exactly 1 from step 21 onward. Training-time AIME 2024 mean@32 reached 14.0% at step 371, roughly 0.4× the 7B models. Its mean@128 readout at steps 320/350/370 is AIME 2024 13.79, AIME 2025 10.31,

pooled 12.05, AIME 2026 12.09. Qwen2.5-7B (general base), same configuration: mixed rate 47% at step 1, 61–75% from step 100; one generation batch per step after step 20. The run ended at step 400; training-time AIME 2024 mean@32 reached 17.7%. Its mean@128 readout at steps 320/350/380 is AIME 2024 16.68, AIME 2025 10.74, pooled 13.71, AIME 2026 8.41. The general base model therefore lands at about half the accuracy of the math-specialised model of the same size under the identical recipe, while its mixed-group plateau is the same; the plateau is not a proxy for accuracy.

A.6 Runs that moved the step trajectory

DAPO under the controller reads 24.36 pooled, 0.95 below uncontrolled DAPO, in a direction the controller's trajectory explains: DAPO's natural KL is far below the late-phase δ early in training, so the controller raised the learning rate 2–3$\times$ for the first hundred steps. Removing the std-normalisation instead (Dr. GRPO) halves the realised KL step, to a median of $0.52\,\delta$, and lands at 24.33, within 0.03 of the constant-δ run. Both are inside the prediction interval, but the agreement across two opposite manipulations of the step is the one pattern in the audit we would test next with seeds.

A.7 Counting convention

Nine methods were built (Section 1; Table 6). Table 3 reads them out on twelve single-seed runs of modified recipes, which the main text calls the variant runs; they include the two controlled PSO runs, the controlled DAPO run and the Dr. GRPO ablation, which change the step rather than the credit. The table also carries one seed each of DAPO and of the random 128-prompt recipe. FIPO-K's row is read at a different window (steps 200–280) against DAPO at that window, so it is compared only to its matched reference (difference 0.00), has no AIME 2026 value, and is excluded from the counts, the rank statistics, and the 22 readings (11 pooled, 11 on AIME 2026) of the sensitivity check; the eleven remaining variant runs are the standard-window variant runs. Under the pooled-variance sensitivity interval (2.40 pooled, 1.48 on AIME 2026), five of those 22 readings fall outside (two pooled, three on AIME 2026) against about one expected by chance; under the three-seed interval, none does. Figure 1 plots all fourteen runs, with FIPO-K marked separately.

A.8 Rank scatter across years

Figure 4 plots, for the thirteen standard-window runs, the rank on AIME 2024 against the rank on AIME 2025. The correlations of Section 3 move with the run set: -0.21, $+0.30$, $+0.55$ (2024–2025, 2024–2026, 2025–2026) on the eleven variant runs alone, and -0.09, $+0.25$, $+0.38$ on all seventeen single-seed runs, against -0.03, $+0.21$, $+0.58$ on the thirteen; one of the six inversions counted in the main text was later dissolved by a third seed (Section 4). The scatter has no visible structure, which is what near-equal true accuracies under 30-problem sampling noise predict (Section 3).

A.9 The amplification sweep on both years

Figure 5 plots the PSO v5 sweep of Section 5 against the measured spread of its weights on AIME 2025 and AIME 2024, with the controlled runs marked.

A.10 Contribution of the tracker

The tracker saved no compute, and its effect on accuracy does not survive the seeds: with it enabled the run reads 23.44 pooled at 256 and 25.60 at 128, against 25.13 for the three random runs at 128, all inside the interval. Its greedy share re-draws prompts already observed to be mixed, which serves the objective it was given and not the one that matters, because the filter it was built to help was never the binding constraint. It is reported here as the experiment that exposed the truncation.

A.11 Keeping the surplus

Halving discards less, but at 128 about a quarter of the mixed groups found are still dropped each step. The obvious further step, which we did not run, is to keep them in a buffer for the next update,

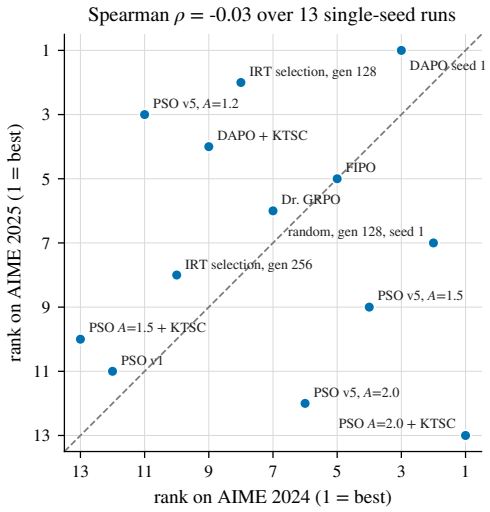

Figure 4: Rank of each single-seed run on AIME 2024 against its rank on AIME 2025 (FIPO-K omitted because its readout window differs). The dashed line is agreement. With near-equal true accuracies both rankings are mostly noise, and this scatter is what that predicts; the run ranked first on 2024 is last on 2025.

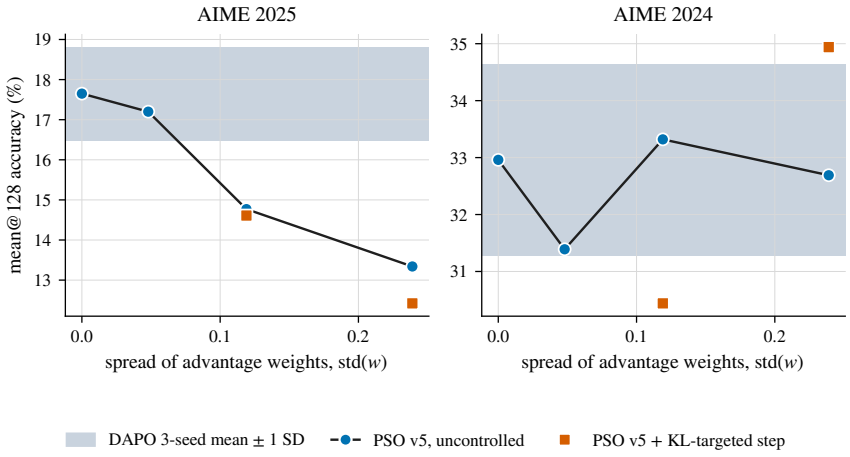

Figure 5: PSO v5 accuracy against the measured spread of its advantage weights, $\mathrm{std}(w)$, with $\mathrm{std}(w) = 0$ being the DAPO three-seed mean. Shaded: one baseline seed SD. On AIME 2025 the loss is monotone in the dose and the controlled runs (squares) do not recover it; on AIME 2024 the same runs show no ordering.

as rollout replay does (Sun et al., 2025). Section 5 suggests it would be cheap: the realised policy change per step is about 10^{-4} in KL and the ratio clip almost never binds, so a group one step stale is nearly on-policy. A second alternative is to generate incrementally and stop as soon as 64 mixed groups are in hand, which removes the surplus exactly rather than approximately; it requires a rollout engine that can be interrupted mid-batch and gives up some of the throughput of a single large batch, which is why we did not run it in this framework. The fixed-batch sizing rule is the static counterpart of that streaming scheme, needs no change to the trainer, and is the baseline either extension should be measured against.

A.12 AIME 2026 UNDER THE PREDICTION-INTERVAL RULE

On the contamination-free view no run is outside the three-seed prediction interval (3.1 around 14.54). Under the pooled-variance sensitivity interval (1.48), two runs are *above* the baseline, DAPO

under the controller (+2.66) and Dr. GRPO (+1.80), and one below (PSO $A = 1.5$ under the controller, -1.53). Neither of the two is above the baseline on either other year; the controlled DAPO is ninth of thirteen on AIME 2024. Three of the five sensitivity-interval exceedances in 22 readings fall on this view, more than chance alone gives, but that interval rests on a gen-128 SD of 0.21 from runs that solve four to five of thirty problems. We report these readings and do not interpret them, as with the negative exceedances.

A.13 AIME 2026 PER-SEED VALUES

DAPO seeds 1/2/3: 15.06 / 13.86 / 14.70 (per-checkpoint 14.69, 14.95, 15.55; 13.78, 13.46, 14.35; 14.53, 15.05, 14.53). Generation batch 128 seeds 1/2/3: 14.58 / 14.32 / 14.16 (per-checkpoint 14.35, 14.84, 14.56; 14.35, 14.24, 14.37; 14.04, 14.19, 14.24). Source: the MathArena transcription of the 2026 contest (MathArena, 2026; Dekoninck et al., 2026), all thirty answers cross-checked against an independent transcription; one third-party page disagreed on one answer and was itself corrupted elsewhere, which is why two agreeing sources were required.

A.14 TABLE 5: EVERY COMPLETED RUN

Table 5 lists every completed run on all four views, one row per run, including the runs of the other two models and the 96-prompt run that the main text reports only by pointer.

Table 5: All completed runs, mean@128 at steps 320/350/380 unless stated, four views. Qwen2.5-Math-7B unless stated; single-seed unless marked. FIPO-K's window has no AIME 2026 readout.

Run	AIME 2024	AIME 2025	Pooled 60	AIME 2026
DAPO seed 1	33.48	18.91	26.20	15.06
DAPO seed 2	31.08	17.40	24.24	13.86
DAPO seed 3	34.31	16.64	25.48	14.70
gen 128, random, seed 1	34.34	15.36	24.85	14.58
gen 128, random, seed 2 (320/350/379)	34.48	16.62	25.55	14.32
gen 128, random, seed 3	32.60	17.39	24.99	14.16
FIPO	33.09	16.46	24.77	15.38
FIPO-K (200–280)	34.51	14.16	24.33	—
PSO v1	30.49	14.37	22.43	13.07
PSO v5 A=1.2	31.39	17.20	24.29	13.65
PSO v5 A=1.5	33.32	14.77	24.05	13.16
PSO v5 A=2.0	32.69	13.34	23.02	14.67
DAPO + KTSC	31.97	16.75	24.36	17.20
PSO A=1.5 + KTSC	30.44	14.61	22.53	13.01
PSO A=2.0 + KTSC	34.94	12.42	23.68	14.05
Dr. GRPO	32.66	16.01	24.33	16.34
IRT selection, gen 256	31.74	15.14	23.44	14.99
IRT selection, gen 128	32.57	18.63	25.60	15.77
gen 96, random (steps 280/290/296)	30.34	16.43	23.39	15.10
Qwen2.5-Math-1.5B, gen 256 (steps 320/350/370)	13.79	10.31	12.05	12.09
Qwen2.5-7B (general), gen 256	16.68	10.74	13.71	8.41

A.15 PRE-REGISTERED READOUTS AND THEIR OUTCOMES

Each experiment's readout was fixed before its numbers existed. The ones whose outcomes are used in the main text:

- *Controlled PSO,* $A = 1.5$. Branches: lands at DAPO's level (step size), above DAPO (reweighting helps), at the twin's level (credit assignment). Outcome: 22.53 against controlled DAPO 24.36 and twin 24.05; third branch.
- *Controlled PSO,* $A = 2.0$. Contrast: AIME 2025 versus the uncontrolled twin; $\geq +2$ points means step size explains the dose harm, ≈ 0 means the reweighting. Outcome: -0.92.

- *IRT selection.* Mechanism gate: mixed fraction must rise materially above the 24.8% baseline. Outcome: it did (72–78%), and the $\epsilon = 1$ control then showed the same fraction without the tracker, which overturned the premise.
- *Random selection at gen 128 ($\epsilon = 1$).* Readings: rollouts per step $\approx 2\times$ the tracked run means the tracker's contribution is the saving; $\approx$ equal means the tracker is not needed. Outcome: 128.0 versus 128.4; second reading.
- *Second and third DAPO seeds.* Reading: $|\Delta| \geq 1.5$ pooled between seeds reclassifies every single-run difference below 1.5 as noise. Outcome: 1.96.

B Formal specification of every variant

The remainder of the appendix gives the exact mathematical form of each algorithm, transcribed from the implementation that produced the reported numbers, organised by the slot in which it intervenes. Section numbers below are referenced from the main text.

Table 6: Where each variant intervenes. Methods above the line redistribute credit *within* rollouts; the two below change the step or the prompts and leave credit untouched.

Slot	Method	What changes
advantage	PSO v1 / v2 / v5 / v6	scale $\hat{A}_{i,t}$ by a per-position pivotality weight (hard cut-off; soft via $p_t / \max p$; soft via rank; signed)
token reward	CSR, PSO+CSR	add a dense contrastive step reward, signed by outcome; three gatings
importance weight	FIPO (Ma et al., 2026), FIPO-K	reweight positions by discounted future log-probability drift; fixed or Kalman-adapted gain
step size	KTSC	multiply the learning rate to hold the realised per-step KL at a target
prompt distribution	IRT selection	select prompts by tracked probability of a mixed group

Conventions. The subsections that follow give each method's exact form, the hyper-parameters used, and the measurements that verify its mechanism. Accuracy comparisons quote mean@128 at steps $320/350/380$ against the three-seed DAPO mean (25.31 pooled; Table 1) and are read with the three-seed prediction interval of Section 3; the mixed-group rate is quoted as the interdecile range over steps 100 onward (60–76%) unless a full range is stated.

C Setup and notation

All methods operate on groups of sampled responses to the same prompt, and all of them modify one of two objects: the *token-level reward* before advantages are computed, or the *advantage* after.

For a prompt q we sample G responses $o_1, \ldots, o_G$ from the rollout policy $\pi_{\theta_{\text{old}}}$. In every run reported here $G = 32$. Responses are padded to a common length L, and we write

$$m_{i,t} \in \{0, 1\} \quad \text{response mask: 1 if token } t \text{ of response } i \text{ is real,} \tag{2}$$

$$\ell_{i,t} = \log \pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t}) \quad \text{rollout log-probability of the token actually generated,} \tag{3}$$

$$r_{i,t} \in \mathbb{R} \quad \text{token-level reward (nonzero only at the final token),} \tag{4}$$

$$R_i = \sum_{t=1}^{L} r_{i,t} \quad \text{outcome reward of response } i. \tag{5}$$

The reward manager assigns $R_i = +1$ for a verified-correct answer and $R_i = -1$ for an incorrect one, with an additional soft length penalty for responses that enter the overlong buffer, so in practice $R_i \in [-2, 1]$.

Responses split into a correct and a wrong set,

$$\mathcal{C} = \{\, i : R_i > 0 \,\}, \qquad \mathcal{W} = \{\, i : R_i \le 0 \,\}, \tag{6}$$

and every method in this paper is a no-op on a group where $\mathcal{C} = \emptyset$ or $\mathcal{W} = \emptyset$—with no contrast available there is nothing to compute. Such groups are in any case removed upstream by DAPO's dynamic sampling (Appendix D).

Where each method intervenes. Writing the training step as a pipeline,

$$\underbrace{\{o_i\}}_{\text{rollout}} \longrightarrow \underbrace{r_{i,t}}_{\text{reward}} \xrightarrow{\text{CSR}} \underbrace{\hat{A}_{i,t}}_{\texttt{compute_advantage}} \xrightarrow{\text{PSO}} \underbrace{\mathcal{L}(\theta)}_{\text{policy loss}}, \tag{7}$$

CSR edits $r_{i,t}$ *before* advantages exist, while PSO rescales $\hat{A}_{i,t}$ *after* they have been computed. The two are therefore composable, which is what the PSO+CSR variants exploit.

D Baseline: GRPO with DAPO modifications

D.1 Group-relative advantage

GRPO replaces the learned value function with a group baseline. The outcome rewards of the group are standardised,

$$\hat{A}_i = \frac{R_i - \mu}{\sigma}, \qquad \mu = \frac{1}{G}\sum_{j=1}^{G} R_j, \qquad \sigma = \sqrt{\frac{1}{G-1}\sum_{j=1}^{G}(R_j - \mu)^2}, \tag{8}$$

where σ is the unbiased (Bessel-corrected) standard deviation, as in the implementation's default `torch.std`; and the resulting scalar is broadcast unchanged to every token of that response:

$$\hat{A}_{i,t} = \hat{A}_i \quad \text{for all } t \text{ with } m_{i,t} = 1. \tag{9}$$

Equation 9 is the object every method in this paper attacks. It states that a 2,000-token derivation and the single token where the argument actually went wrong receive *identical* credit. This is the "coarse-grained credit assignment" that pivotal-token and dense-reward methods set out to fix.

A useful consequence of equation 8: in a group of G responses where exactly one is correct, $\hat{A}_i = \sqrt{(G-1)^2/G} = 5.480$ for $G = 32$ (the population standard deviation would give $\sqrt{G-1} \approx 5.57$; the logged value is 5.480, which is how the convention was confirmed). Rare-success groups therefore dominate the gradient, which matters when reading the advantage statistics later.

D.2 Policy objective

With importance ratio

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}, \tag{10}$$

the token-mean, asymmetrically clipped DAPO objective is

$$\mathcal{J}(\theta) = \mathbb{E}\left[\frac{1}{\sum_i |o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\min\Big(\rho_{i,t}\hat{A}_{i,t},\ \text{clip}(\rho_{i,t},\, 1-\epsilon_{\text{low}},\, 1+\epsilon_{\text{high}})\,\hat{A}_{i,t}\Big)\right], \tag{11}$$

with $\epsilon_{\text{low}} = 0.2$, $\epsilon_{\text{high}} = 0.28$, no KL penalty ($\beta = 0$), and a dual-clip guard (Ye et al., 2020) at $c = 10$ on negative-advantage tokens.

D.3 Dynamic sampling

DAPO keeps only groups whose outcomes are mixed,

$$\text{std}\left(\{\mathbb{1}[R_i > 0]\}_{i=1}^{G}\right) > 0, \tag{12}$$

regenerating until the training batch is full. Uniformly-correct groups carry no gradient ($\hat{A}_i = 0$ for all i), and uniformly-wrong groups likewise.

This filter is stated explicitly because it is a shared precondition of every method below and, in the configuration used throughout, the trainer discards roughly 75% of generated prompts. Every method in this paper redistributes credit *within* the groups that are kept; none of them changes which prompts survive. That 75% is dominated by truncation, not by the filter: the generation batch is four times the training batch, and once 64 mixed groups are found the remaining mixed groups are discarded too. Direct counts over every generated group put the mixed fraction at 60–76% (interdecile range over steps 100 onward; Section 4 of the main text), so most discarded prompts were mixed.

E THE CONTRASTIVE SIGNAL

All PSO and CSR variants are built from one quantity. Let

$$\bar{\ell}_t^{\mathcal{C}} = \frac{\sum_{i\in\mathcal{C}} m_{i,t}\,\ell_{i,t}}{\max\big(\sum_{i\in\mathcal{C}} m_{i,t},\,1\big)}, \qquad \bar{\ell}_t^{\mathcal{W}} = \frac{\sum_{i\in\mathcal{W}} m_{i,t}\,\ell_{i,t}}{\max\big(\sum_{i\in\mathcal{W}} m_{i,t},\,1\big)}, \tag{13}$$

be the mask-weighted mean rollout log-probability at position t over correct and over wrong responses. The **contrastive signal** is their difference,

$$\boxed{c_t \;=\; \bar{\ell}_t^{\mathcal{C}} - \bar{\ell}_t^{\mathcal{W}}} \tag{14}$$

Reading c_t. Each response contributes the log-probability of *its own* generated token, so c_t is not a comparison of two distributions over a shared token. It is a *confidence gap*:

- $c_t > 0$: at position t the correct responses were more confident in what they wrote than the wrong ones were in what they wrote. The successful trajectory is committing decisively here.
- $c_t < 0$: the *wrong* responses were the confident ones. The failing trajectory is committing decisively—to an error.

Both regimes are informative, and whether to keep the sign is the question separating v5 from v6 (Appendix F.4).

Validity mask. Positions valid in at least one response of either set,

$$u_t = \mathbb{1}\big[\textstyle\sum_{i\in\mathcal{C}} m_{i,t} + \sum_{i\in\mathcal{W}} m_{i,t} > 0\big], \qquad n_{\text{valid}} = \sum_t u_t, \tag{15}$$

define the support on which c_t is meaningful. PSO v1 used $m_{i_0,t}$—the mask of an arbitrary first response of the group—in place of u_t, and an unmasked mean in place of equation 13; both were corrected in v2 and retained thereafter.

F PIVOTAL STEP OPTIMIZATION (PSO)

PSO leaves rewards untouched and rescales the broadcast advantage equation 9 position by position:

$$\tilde{A}_{i,t} \;=\; \hat{A}_i \cdot \tilde{w}_{i,t}. \tag{16}$$

All variants share the same two-stage construction—select a pivotal set, then map it to weights—and differ only in how they do it.

F.1 STAGE 1: SELECTING PIVOTAL POSITIONS

Define per-position **pivotality**

$$p_t \;=\; \begin{cases} |c_t| \cdot u_t & \text{v1, v2, v5} \\ \max(c_t,\,0) \cdot u_t & \text{v6 (Appendix F.4)} \end{cases} \tag{17}$$

Take the top ρ fraction by p_t, where $\rho = \texttt{pso_top_ratio}$:

$$k = \max\big(1, \lfloor \rho\, n_{\text{valid}} \rfloor\big), \qquad \tau = k\text{-th largest value of } \{p_t\}, \qquad \mathcal{P} = \{\, t : p_t \geq \tau \ \wedge\ u_t = 1 \,\}. \tag{18}$$

Throughout, $\rho = 0.2$. Stage 1 is *identical* across v1, v2 and v5; only v6 changes it, through equation 17.

F.2 Stage 2: mapping the pivotal set to weights

This is where the variants separate. Let $A = \texttt{pso_amplify}$.

v1—hard cutoff. Every pivotal position receives the full amplification, every other position receives none:

$$w_t = \begin{cases} A & t \in \mathcal{P} \\ 1 & \text{otherwise,} \end{cases} \qquad \tilde{w}_{i,t} = w_t. \tag{19}$$

No normalisation is applied, so the mean weight over valid positions is $1 + \rho(A-1)$, i.e. v1 inflates the effective learning rate by that factor.

v2—soft, max-normalised. The intent was to grade the amplification by how pivotal a position is:

$$s_t = \frac{p_t \cdot \mathbb{1}[t \in \mathcal{P}]}{\max_{u \in \mathcal{P}} p_u}, \qquad w_t = 1 + (A-1)\, s_t. \tag{20}$$

v5—soft, rank-normalised. Identical to equation 20 except that the scale is the position's *rank* inside the pivotal set rather than its value:

$$\boxed{s_t = \frac{\operatorname{rank}_{\mathcal{P}}(p_t)}{|\mathcal{P}|} \in (0, 1], \qquad w_t = 1 + (A-1)\, s_t} \tag{21}$$

where $\operatorname{rank}_{\mathcal{P}}$ assigns 1 to the smallest and $|\mathcal{P}|$ to the largest p_t within $\mathcal{P}$. The most pivotal position still receives the full A, so v5 preserves v2's stated semantics; only the distribution over the rest of the set changes.

Why equation 20 fails and equation 21 does not. The distribution of p_t is strongly right-tailed: a couple of positions dominate the rest by an order of magnitude. Dividing by $\max_{u \in \mathcal{P}} p_u$ therefore crushes almost the whole pivotal set toward zero. Concretely, if the top-20% pivotality values are

$$(10.0,\ 1.2,\ 0.9,\ 0.7,\ 0.5,\ 0.4,\ 0.3,\ 0.3,\ 0.2,\ 0.2),$$

then

$$s^{\text{v2}} = (1.00,\ 0.12,\ 0.09,\ 0.07,\ 0.05,\ 0.04,\ 0.03,\ 0.03,\ 0.02,\ 0.02), \quad \overline{s^{\text{v2}}} = 0.15,$$

$$s^{\text{v5}} = (1.0,\ 0.9,\ 0.8,\ 0.7,\ 0.6,\ 0.5,\ 0.4,\ 0.3,\ 0.2,\ 0.1), \quad \overline{s^{\text{v5}}} = 0.55.$$

Under v2 only a single token is meaningfully amplified: the method selects the top 20% of positions and then effectively uses the top 2%. Measured on the actual runs, $\bar{s}$ was 0.15 for v2 against 0.50 for v5 (Table 10), and the nominal setting $A = 2.0$ behaved like an effective amplification of roughly 1.15.

Under v5, s_t takes the values $1/k, 2/k, \ldots, 1$ over the k pivotal positions, so $\mathbb{E}[s_t \mid t \in \mathcal{P}] = (k+1)/(2k)$, which is 0.55 in the ten-position example above and 0.5006 at the measured pivotal set sizes (Table 10), close enough to $1/2$ that the intended A is realised. The closed form for the induced spread, which matches the measurements, is

$$\operatorname{std}(w) = (A-1)\sqrt{\tfrac{\rho}{3} - \tfrac{\rho^2}{4}} \ \overset{\rho=0.2}{=}\ 0.238\,(A-1). \tag{22}$$

F.3 PER-TRAJECTORY GRADIENT NORMALISATION

v2, v5 and v6 (but not v1) renormalise the weights within each response so that the mean weight over its valid tokens is exactly one:

$$\tilde{w}_{i,t} = \frac{w_t^{(i)}}{\dfrac{1}{|\{t : m_{i,t} = 1\}|} \displaystyle\sum_{t\,:\,m_{i,t}=1} w_t^{(i)}}, \qquad w_t^{(i)} = \begin{cases} w_t & m_{i,t} = 1 \\ 1 & \text{otherwise.} \end{cases} \tag{23}$$

Consequently PSO from v2 onward is a pure *redistribution* of gradient within a trajectory: it cannot change the total learning signal that trajectory contributes, only where along the sequence it lands. v1, lacking equation 23, changes both.

F.4 V6—SIGNED PIVOTALITY

v6 is identical to v5 in every respect except equation 17: it keeps the sign of c_t and amplifies only positions where the correct responses were the confident ones,

$$p_t = \max(c_t, 0) \cdot u_t. \tag{24}$$

Because clamping can leave fewer than k strictly positive entries—in which case τ would fall to zero and equation 18 would select every position—the pivotal set additionally requires $p_t > 0$:

$$\mathcal{P} = \{\, t : p_t \geq \tau \;\wedge\; u_t = 1 \;\wedge\; p_t > 0 \,\}. \tag{25}$$

Top-k selection and the rank map are otherwise untouched, so at equal A the intervention *strength* matches v5 and only the *location* of the amplification differs—a strength-matched controlled comparison. This is confirmed by the step-1 diagnostics in Table 10, which agree with v5.

G CONTRASTIVE STEP REWARD (CSR)

CSR converts the sparse outcome reward into a dense per-token reward, acting *before* advantages are computed. Normalise the contrastive signal by its own spread over valid positions,

$$\hat{c}_t = \frac{c_t}{\sigma_c + \varepsilon}\, u_t, \qquad \sigma_c = \mathrm{std}\left(\{c_t : u_t = 1\}\right), \qquad \varepsilon = 10^{-8}, \tag{26}$$

and add it to the token reward with a sign set by the response's own outcome:

$$\boxed{r_{i,t} \;\longleftarrow\; r_{i,t} \;+\; \alpha\, \mathrm{sgn}(R_i)\, \hat{c}_t\, m_{i,t}} \tag{27}$$

with $\alpha = \texttt{csr_alpha}$. The modified rewards then flow into equation 8 as usual, so unlike PSO the perturbation passes *through* the group standardisation.

The interference in equation 27. Consider a position with $c_t < 0$, i.e. one where the wrong responses were the confident ones. For a wrong response, $\mathrm{sgn}(R_i) = -1$ and the added term $\alpha \cdot (-1) \cdot \hat{c}_t > 0$ is a *bonus*—CSR rewards the failing trajectory at the position where it went confidently astray. For a correct response at the same position, $\mathrm{sgn}(R_i) = +1$ gives a *penalty*. Both directions oppose what the outcome reward is trying to do, and this is the mechanism the PSO+CSR variants below attempt to remove.

H PSO + CSR COMBINATIONS

All three combinations run CSR on the rewards and then PSO on the resulting advantages, in that order:

$$r_{i,t} \xrightarrow{\text{CSR}} r'_{i,t} \xrightarrow{\textit{equation 8}} \hat{A}_i \xrightarrow{\text{PSO}} \tilde{A}_{i,t}. \tag{28}$$

The PSO stage is v1-style equation 19 in all of them. They differ only in which positions and which responses the CSR term touches.

v2—unrestricted. Equation 27 applied as written, to every position and every response.

v3—pivotal-token gated. The CSR term is restricted to the same pivotal set PSO amplifies, leaving all other positions at their outcome reward:

$$r_{i,t} \;\longleftarrow\; r_{i,t} + \alpha\, \mathrm{sgn}(R_i)\, \hat{c}_t\, \mathbb{1}[t \in \mathcal{P}]\, m_{i,t}. \tag{29}$$

Both correct and wrong responses are still modified, so the sign interference described above survives; it is merely confined to 20% of positions.

v4—correct-trajectory gated. The sign factor is removed by restricting the bonus to correct responses only, leaving wrong ones entirely untouched:

$$r_{i,t} \;\longleftarrow\; \begin{cases} r_{i,t} + \alpha\, \hat{c}_t\, m_{i,t} & R_i > 0 \\ r_{i,t} & \text{otherwise.} \end{cases} \tag{30}$$

Wrong responses are then suppressed by GRPO and PSO alone, with no CSR term able to work against them.

I FIPO: Future-Influence Policy Optimization

PSO and CSR act on the advantage and the reward respectively. FIPO acts on neither: it multiplies the advantage by a weight derived from the *policy's own drift*, and it is the published method (Ma et al., 2026) this project set out to reproduce; what we report is a single-seed re-implementation at our batch size and step budget, not a reproduction claim. It is specified here because Appendix A reports its accuracy, and because FIPO-K (Appendix J) is defined as a modification of it.

I.1 The future-influence weight

Let $z_{i,t}$ denote the per-token log-ratio already appearing in equation 11, clamped for stability,

$$z_{i,t} \;=\; \mathrm{clip}\big(\log \rho_{i,t},\, -20,\, 20\big) \;=\; \mathrm{clip}\big(\log \pi_\theta(o_{i,t}) - \log \pi_{\theta_{\text{old}}}(o_{i,t}),\, -20,\, 20\big). \tag{31}$$

Tokens above the dual-clip threshold contribute no gradient in any case, so FIPO removes them from the accumulation via a participation mask,

$$\pi_{i,t} = \mathbb{1}\big[z_{i,t} \le \log c\big], \qquad \tilde{z}_{i,t} = z_{i,t}\, m_{i,t}\, \pi_{i,t}, \tag{32}$$

with $c = 10$ the dual-clip constant. The *future influence* at position t is the exponentially discounted sum of all drift from t onward,

$$F_{i,t} \;=\; \sum_{k \ge t} \gamma^{\,k-t}\, \tilde{z}_{i,k}, \qquad \gamma = 2^{-1/\tau}, \tag{33}$$

equivalently the backward recursion $F_{i,t} = \tilde{z}_{i,t} + \gamma F_{i,t+1}$. The decay rate τ is a hyperparameter, set to 32 in our runs; $\gamma = 2^{-1/32} = 0.97857$, so influence halves every 32 tokens.

The weight applied to the advantage is the exponential of equation 33, clipped for stability:

$$w_{i,t}^{\text{FIPO}} \;=\; \mathrm{clip}\big(\exp(F_{i,t}),\, 1-\epsilon_F,\, 1+\epsilon_F\big), \qquad \epsilon_F = 0.2, \tag{34}$$

followed by a safety rule that prevents already-suppressed negative-advantage tokens from being amplified,

$$w_{i,t}^{\text{FIPO}} \leftarrow \mathrm{clip}\,\big(w_{i,t}^{\text{FIPO}},\, 0.8,\, 1.0\big) \quad \text{where} \quad \hat{A}_{i,t} < 0 \;\wedge\; \rho_{i,t} > 3. \tag{35}$$

The objective is then equation 11 with $\hat{A}_{i,t}$ replaced by $w_{i,t}^{\text{FIPO}} \hat{A}_{i,t}$, the weight treated as a constant (`detach`) so no gradient flows through it.

Interpretation. A position is up-weighted when the tokens that follow it moved in the direction the update favours, and down-weighted otherwise. The discount encodes a fixed assumption about how far into the future a token's influence extends.

Cost. As written, equation 33 is evaluated as a matrix product against a decay matrix $\Gamma_{tk} = \gamma^{k-t}\mathbb{1}[k \ge t]$, computed in column chunks. This is $O(L^2)$ in the response length.

I.2 THE MECHANISM IS LARGELY NEUTRALISED BY ITS OWN CLIPPING

The clip in equation 34 is not a formality. Measured over a full run:

mean of w^{FIPO}	1.0015
mean of $\exp(F)$ before clipping	1.0075
maximum of $\exp(F)$ before clipping	5.11–6.20
fraction of tokens at a clip boundary	9.9%

The raw weight reaches a factor of five, but $\epsilon_F = 0.2$ confines the applied weight to $[0.8, 1.2]$, and the realised average intervention is 0.15% away from the identity. Roughly one token in ten is at a boundary, i.e. has its future-influence signal truncated. Most of what equation 33 computes is therefore removed again by equation 34.

I.3 τ CANNOT BE CHOSEN

The published ablation over $\tau \in \{8, 32, 128, 256\}$ (Ma et al., 2026, Table 6) ranks $\tau = 256$ first on AIME 2024 (42.0%) and last on AIME 2025 (16.0%), while $\tau = 128$ leads on AIME 2025 (21.0%). That ablation is run on the same backbone we use, and our reimplementation follows its 7B setting exactly: future-KL clip $[0.8, 1.2]$, i.e. $\epsilon_F = 0.2$ in equation 34, and safety threshold 3.0 in equation 35. The ordering inverts between the two benchmarks. A fixed horizon is therefore wrong somewhere: no single γ serves both. It is the observation FIPO-K starts from.

J FIPO-K: ADAPTIVE GAIN

J.1 FUTURE-INFLUENCE AS A FILTER

Equation 33 is a first-order infinite-impulse-response filter with a fixed pole γ. Read as an estimator, it is the steady-state Kalman filter for a random-walk latent state observed in noise, and γ is the *Kalman gain*—fixed, and therefore optimal only if the observation noise is constant along the sequence.

There is no reason for it to be constant. FIPO-K replaces the scalar γ with a per-position gain,

$$\gamma_{i,t} \;=\; \mathrm{clip}\!\left(\frac{Q}{Q + R_{i,t}},\; \gamma_{\min},\; \gamma_{\max}\right), \qquad \gamma_{\min} = 0.5,\;\; \gamma_{\max} = 0.9995, \tag{36}$$

where $R_{i,t}$ estimates the observation noise ahead of position t as the variance of the drift over a forward window,

$$R_{i,t} \;=\; \mathrm{Var}\left\{\tilde{z}_{i,k} : t < k \le t + W,\; \pi_{i,k} m_{i,k} = 1\right\}, \qquad W = 64, \tag{37}$$

falling back to the per-response variance where fewer than two valid observations lie ahead. Both equation 37 and the backward recursion are computed by prefix sums in $O(L)$.

The reading is: *where the future is consistent, look far; where it is noisy, look only nearby.*

J.2 CALIBRATION REMOVES THE HYPERPARAMETER

Q is the process noise. Rather than introducing it as a new tunable, it is set per micro-batch so that the *median* gain equals FIPO's fixed value:

$$Q \;=\; \frac{\gamma^{\star}\ \mathrm{median}_{i,t}(R_{i,t})}{1 - \gamma^{\star}}, \qquad \gamma^{\star} = 2^{-1/\tau}, \tag{38}$$

which follows by inverting equation 36 at the median. Consequently τ is *eliminated* rather than replaced: the typical horizon is pinned to FIPO's, and only its allocation across positions changes. Homoscedastic R recovers FIPO exactly, so FIPO is a special case—a property verified numerically both in unit tests and in training, where step 1 (at which $\log \pi_\theta \equiv \log \pi_{\theta_{\text{old}}}$, hence $\tilde{z} \equiv 0$ and R is degenerate) produced a gain standard deviation of exactly zero.

Everything downstream of F—equation 34, equation 35, and the objective—is unchanged, so any difference in outcome is attributable to the gain.

J.3 LOG-SPACE EVALUATION

With γ varying, equation 33 is no longer a Toeplitz matrix product. Writing $G_{i,m} = \sum_{j<m} \log \gamma_{i,j}$, the recursion has the closed form

$$F_{i,t} \;=\; e^{-G_{i,t}} \sum_{k \geq t} e^{G_{i,k}}\, \tilde{z}_{i,k}, \tag{39}$$

a suffix sum computable in $O(L)$—an asymptotic improvement on FIPO's $O(L^2)$.

Equation 39 requires care. Over a chunk of length n with γ near $\gamma_{\min}$, G falls by $n \log \gamma_{\min}$; at $n = 777$ and $\gamma_{\min} = 0.5$ this is $G \approx -539$, so $e^{-G} = \infty$ and $e^{G} = 0$, and their product is `NaN`. The implementation therefore caps the chunk length from the observed maximum $|\log \gamma|$ so that no exponent exceeds 60, and the unit tests include this case explicitly.

J.4 MEASURED BEHAVIOUR

The calibration holds throughout training ($\mathrm{median}(\gamma) = 0.9785694$ against $\gamma^\star = 0.9785721$), and the gain varies ($\mathrm{std}(\gamma) \approx 0.04$), so the mechanism is neither inert nor collapsed to FIPO.

One consequence of equation 36 deserves recording because it affects how the method should be reported. The effective horizon $1/(1-\gamma)$ is a convex function of γ, so pinning the *median* gain does not pin the mean horizon: the observed mean was ≈ 400 tokens against FIPO's fixed 46.7, the difference carried by a minority of positions at $\gamma_{\max}$. The controlled quantity is the median horizon, and that is what should be quoted.

Finally, FIPO-K reduces the clipping loss identified in Appendix I.2: the fraction of tokens at a weight boundary falls from 9.9% to 3.6–5.0%. The adaptive gain does suppress the future-influence blow-ups that the fixed gain produced. The accuracy result of Appendix A therefore cannot be attributed to a malfunctioning mechanism: the gain behaved as designed, and accuracy did not improve.

K KL-TARGETED STEP CONTROL (KTSC)

Every method above redistributes credit: it changes $\hat{A}_{i,t}$ or the token rewards that produce it. KTSC changes nothing about credit. It restores the constraint that TRPO imposed and PPO discarded—a bound on how far the policy is allowed to move per update—and it exists because that constraint turns out to be entirely absent in our setting.

K.1 MOTIVATION: THE CLIP DOES NOT BIND

TRPO solves a constrained problem,

$$\max_{\theta}\; L_{\theta_{\text{old}}}(\theta) \quad \text{subject to} \quad \overline{D}_{\mathrm{KL}}(\theta_{\text{old}}, \theta) \leq \delta, \tag{40}$$

and enforces equation 40 directly: it takes a candidate step, *measures* the realised KL, and shrinks the step until the constraint holds. PPO replaced this line search with the cheap surrogate already written in equation 11 — clipping $\rho_{i,t}$ to $[1-\epsilon_{\text{low}},\, 1+\epsilon_{\text{high}}]$. The premise is that if no individual token's probability ratio moves far, the policy as a whole has not moved far either.

In our configuration (one epoch of eight mini-batch updates of 8 prompts, 256 responses, per 64-prompt step at learning rate 10^{-6}) the premise fails: the clip is reached by fewer than one token in a thousand. This is what that update schedule produces, and configurations with more updates per rollout or a larger learning rate would engage the clip more; we report it because it determines what the objective can and cannot resist. Measured across full runs of DAPO, FIPO-K and PSO v5:

The interquartile range of ρ is $[0.99999991,\, 1.0]$. Single-precision machine epsilon is 1.2×10^{-7}: the median token's ratio differs from 1 by *less than the floating-point resolution of the comparison*, while the clip boundary sits at 1.28. The value $\epsilon = 0.2$ was calibrated for continuous-control benchmarks with small networks, many epochs per batch, and correspondingly large per-step movement. At $\mathrm{lr} = 10^{-6}$, one epoch, and a $152{,}064$-token vocabulary, it engages only beyond the 99.9th percentile of token ratios (Table 7: $p_{99.9} = 1.21$ against a boundary of 1.28).

Table 7: The clip is inert. Ratio percentiles are over valid response tokens; $\overline{\mathrm{KL}}$ is the logged per-token `ppo_kl`; $\|\nabla\|$ is the gradient norm before clipping at 1.0.

	DAPO	FIPO-K	PSO v5 ($A = 1.5$)
upper clip fraction	6–10×10^{-4}	6–10×10^{-4}	1–3×10^{-3}
dual-clip fraction (max)	2.8×10^{-5}	1.4×10^{-5}	6.6×10^{-4}
$\overline{\mathrm{KL}}$ per token	1.0–1.8×10^{-4}	10^{-5}–10^{-4}	2.5×10^{-4}–2.3×10^{-3}
$\|\nabla\|$	0.27–0.34	0.23–0.38	0.4–5.2
$\max \lvert\hat{A}\rvert$	5.480 (198/200 steps)	5.480 (390/394)	7.6–8.1
ρ at $p_{25}/p_{50}/p_{75}$	1.0000 / 1.0000 / 1.0000		
ρ at $p_{99.5}/p_{99.9}$	1.12 / 1.21		
ρ maximum	2.7–12		

Two consequences follow. First, ρ is a one-sample estimate of a distributional quantity: with 152k possible next tokens, the ratio at the *sampled* token says little about $D_{\mathrm{KL}}(\pi_{\text{old}} \| \pi_\theta)$. Second, and more concretely, PSO v5 at $A = 1.5$ takes roughly eight times DAPO's step ($\overline{\mathrm{KL}}$ mean 1.19×10^{-3} against 1.51×10^{-4}) with nothing opposing it. The larger step is not a matter of advantage scale: Adam is nearly invariant to a uniform rescaling of the gradient, and PSO v2 and later renormalise their weights to mean one within each response. Its gradient norm before clipping is 0.4–5.2 against DAPO's 0.27–0.34 and grows over training (Appendix K.6), so the candidates are a gradient concentrated on the up-weighted tokens, heavy-tailed weights, and the interaction of a growing norm with clipping at 1.0 and with Adam's slowly adapting second-moment estimate; we did not isolate which. Any accuracy difference between PSO and DAPO is therefore confounded: it may be the reweighting, or it may be the unregulated step size. KTSC exists to separate those two, and it does so on the learning rate, which Adam passes through directly, so the separation does not depend on the mechanism.

K.2 TRPO's bound read through GRPO

TRPO's Theorem 1 states

$$\eta(\tilde{\pi}) \;\geq\; L_\pi(\tilde{\pi}) \;-\; \frac{4\varepsilon\gamma}{(1-\gamma)^2}\,\alpha^2, \qquad \varepsilon = \max_{s,a} \big|A_\pi(s,a)\big|, \tag{41}$$

so the admissible step shrinks in proportion to the *largest* advantage in the batch. At $\gamma = 1$, the setting of token-level language-model RL, the coefficient diverges, and finite-horizon forms of the bound are vacuous at horizons of thousands of tokens; we use the theorem as a pointer to the quantity a trust region would track, not as a bound. In GRPO that quantity is not free to vary. Writing k for the number of correct responses in a group of n with binary rewards, the standardised advantage of equation 8 is

$$\hat{A}_{\text{correct}} = \sqrt{\frac{(n-k)(n-1)}{k\,n}}, \qquad \hat{A}_{\text{wrong}} = -\sqrt{\frac{k\,(n-1)}{(n-k)\,n}}, \tag{42}$$

with the unbiased σ of equation 8; at $k = 1$, $n = 32$ this is exactly 5.480—the value observed in 198 of 200 DAPO steps (the population convention would give $\sqrt{(n-k)/k} = 5.568$). Dynamic sampling (Appendix D.3) keeps mixed groups, so a $k = 1$ or $k = 31$ group is present in essentially every batch and $\max|\hat{A}|$ is pinned.

Two separate arguments therefore point at the same groups. TRPO's equation 41 says the trust region should tighten where $|A|$ is largest; Kakade & Langford's advantage-estimation-error term (Kakade & Langford, 2002)—which TRPO explicitly sets aside[1]—says it should tighten where $\hat{A}$ is least reliable. In GRPO both are the same function of k, maximal at $k = 1$, where a single lucky rollout out of 32 determines a 5.48 advantage that shifts by 30% if one sample changes. Reliability and influence are inversely correlated, and equation 11 contains no term that responds to either.

[1] "Our theory ignores estimation error for the advantage function … we omit them for simplicity."

K.3 THE CONTROLLER

KTSC restores the line search as a one-step-lag feedback loop on the learning rate. Writing m_t for the learning-rate multiplier applied at step t and κ_t for the realised per-token KL measured under it,

$$\boxed{m_{t+1} \;=\; \mathrm{clip}\left(m_t \cdot \left(\frac{\delta}{\hat{\kappa}_t}\right)^{g/2},\; \tfrac{1}{c},\; c\right), \qquad \mathrm{lr}_t = \mathrm{lr}_{\text{base}} \cdot m_t} \tag{43}$$

with $\hat{\kappa}_t$ a robust statistic of recent κ (Appendix K.5), gain g, per-step clamp c, and m additionally bounded globally to $[m_{\min}, m_{\max}]$.

Why the exponent is $g/2$. KL is second order in the step size. Expanding around the current parameters, the linear term vanishes because D_{KL} is minimised at $\Delta = 0$, leaving

$$D_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\theta+\Delta}) \;\approx\; \tfrac{1}{2}\,\Delta^\top F \Delta, \qquad \Delta = \eta\, g_\theta \;\;\Longrightarrow\;\; \kappa \;\propto\; \eta^2, \tag{44}$$

with F the Fisher information. Correcting κ by a factor therefore requires correcting η by its square root, which is the $1/2$; the free parameter g then scales how much of that correction is applied per step.

Stability. Let $u = \log \eta$. By equation 44, $\log \kappa = 2u + \text{const}$, and equation 43 in log space becomes

$$u_{t+1} - u^\star \;=\; (1-g)\,(u_t - u^\star), \qquad u^\star \text{ the } \eta \text{ satisfying } \kappa = \delta, \tag{45}$$

so the loop gain is exactly g and converges geometrically for $0 < g < 2$ when κ_t is measured without delay. A statistic with lag L replaces this with the characteristic equation $z^{L+1}(z-1) + g = 0$, whose stability limit is

$$g_{\text{crit}} \;=\; 2\sin\!\left(\frac{\pi}{2(2L+1)}\right) \;=\; 2,\;\; 1,\;\; 0.618,\;\; 0.445 \quad \text{for } L = 0, 1, 2, 3. \tag{46}$$

δ is not a hyperparameter. It is set to the baseline's own realised KL—DAPO's median `ppo_kl` over steps 181–380, $\delta = 1.124 \times 10^{-4}$. Under this choice equation 43 means "hold this algorithm at DAPO's late-phase step size." It is not DAPO's own trajectory: DAPO's KL is well below δ early in training, so the controlled DAPO run takes 2–3$\times$ the base learning rate over its first hundred steps (Appendix K.5). A controlled variant therefore differs from controlled DAPO in credit assignment and in nothing else, and both differ from DAPO in their early step. This is the same device used elsewhere in this study to make comparisons interpretable: FIPO-K pins its median gain to FIPO's fixed γ, and the PSO variants pin the pivotal fraction at 0.20.

K.4 ESTIMATING THE REALISED KL

The true D_{KL} would require summing over the full vocabulary at every position. It is estimated from the sampled token instead. With $r = \pi_\theta / \pi_{\theta_{\text{old}}}$ evaluated at the sampled token, two standard estimators are available:

$$k_1 = -\log r, \qquad k_3 = (r-1) - \log r. \tag{47}$$

Both are unbiased for $D_{\mathrm{KL}}(\pi_{\text{old}} \| \pi_\theta)$ under sampling from π_{old}—the $(r-1)$ term has zero mean—but only k_3 is non-negative pointwise. The logged `ppo_kl` is k_1, which is why *negative* KL readings appear in the training logs; they are an artifact of a signed estimator at small drift, not a negative divergence.

KTSC controls on k_1 because δ was derived from DAPO's k_1 history and the target and the controlled variable must be the same statistic. A capped k_3, computed post-update over a strided slice of the batch, is logged in parallel as a diagnostic. The cap is necessary: k_3 grows like e^ρ, and a single token with $\rho \approx 18$—observed—sends the uncapped batch mean to 1.24×10^3.

K.5 TWO VERSIONS, AND WHY THE FIRST FAILED

v1: exponential moving average. The first implementation used $\hat{\kappa}_t = \mathrm{EMA}_\beta(\kappa)$ with $\beta = 0.7$, $g = 0.5$, δ set to DAPO's *mean* 1.5×10^{-4}, per-step clamp $c = 2$ and floor $m_{\min} = 0.05$. It ran 65 steps and collapsed.

The cause is a property of PSO in its own right: **its KL is heavy-tailed**. Roughly one step in ten produces a spike 10–60× the median (the p_{90}/p_{10} ratio over a full run is 55 for PSO against 11 for DAPO). At steps 57 and 58 two spikes landed consecutively, 5.5×10^{-3} and 9.1×10^{-3}, i.e. 37× and 60× the target. The EMA rose by a factor of 27; the controller halved the learning rate on five successive steps and reached the floor at step 62. The raw KL returned to 2.3×10^{-5} at step 59, but the EMA decayed slowly, so the run spent roughly 15 further steps at $1/20$ of the base learning rate on account of a disturbance that had already passed.

The error comes from the controller's structure. *A lagged controller cannot prevent a spike that has already occurred; it can only penalise the ordinary steps that follow it.*

v2: regulate the steady state, leave spikes to a per-sample mechanism. The redesign separates the two jobs. KTSC controls the *typical* step size and makes no attempt to react to individual excursions.

Table 8: KTSC v1 → v2.

	v1	v2	rationale
$\hat{\kappa}$	$\mathrm{EMA}_{0.7}$	median of last 5	one or two spikes cannot move a median
δ	mean 1.5×10^{-4}	median 1.124×10^{-4}	target the statistic being controlled
g	0.5	0.25	median lags $L = 2$; $g_{\text{crit}} = 0.618$ by equation 46
$m_{\min}$	0.05	0.2	a 20× cut stops learning instead of regulating it
c	2	$1/0.7$	

The gain reduction is forced by equation 46: a median over 5 samples lags about $(5-1)/2 = 2$ steps, giving $g_{\text{crit}} = 0.618$, and $g = 0.5$ sits at 81% of that—badly underdamped. At $g = 0.25$ the loop is at 40% of critical.

In simulation against the exact v1 failure (a plant held at 1.5δ with 40× and 60× spikes on consecutive steps), v1's settings collapse to the floor while v2 moves the multiplier by 1×10^{-4}; on a static plant v2 settles at the analytic optimum with no residual oscillation.

K.6 Measured behaviour

Under live conditions the spike robustness holds. In steps 60–103 of the controlled PSO run the controller absorbed six spikes—8.6, 93.7, 5.7, 9.5, 11.6 and 6.0 times δ, one of them larger than anything v1 encountered—without approaching the floor. The one weakness observed is a window effect: when three spikes fell inside the same 5-sample window the median itself moved, and the multiplier dropped 40% for five steps before recovering, which argues for a longer window in any subsequent version.

Two measurements should be separated from the controller's own behaviour.

The step size is held. Over steps 40–58 the uncontrolled PSO twin ran at a median 1.94δ against the controlled run's 1.03δ, a 1.9× reduction for a 2.1× cut in learning rate. The quadratic law equation 44 would give 4.4× for a fixed update direction; the shortfall is consistent with the direction changing as the step shrinks.

PSO's step size grows during training; DAPO's does not. The uncontrolled twin's median k_1, in units of δ:

steps	40–47	64–71	80–87	96–103
twin $\mathrm{median}(k_1)/\delta$	1.45	1.64	3.62	**10.91**

a sevenfold rise over sixty steps, accompanied by a gradient norm of 3.2 — i.e. norm clipping engaging on essentially every step. DAPO's remains flat near 1δ through step 380. The pivotal

amplification therefore *compounds* as the policy sharpens, and equation 11 does not react to it. The controller is consequently tracking a rising target rather than settling, which accounts for its multiplier falling from 0.92 to 0.36 over the same interval.

This is the trust-region argument of Appendix K.1 in a single measurement, and it is independent of whether the controlled run ultimately scores higher.

K.7 Readout

The pre-registered readout is the mean@128 accuracy at steps $320/350/380$ on AIME 2024 and AIME 2025 pooled, for the controlled PSO run against two fixed references: its uncontrolled twin and DAPO under the same controller. The pre-registered interpretation is Table 9; the readouts are:

run	AIME 2024	AIME 2025	pooled
DAPO, three-seed mean	32.96	17.65	25.31
DAPO + KTSC	31.97	16.75	24.36
PSO A=1.5 + KTSC	30.44	14.61	22.53
PSO A=1.5, uncontrolled	33.32	14.77	24.05
PSO A=2.0 + KTSC	34.94	12.42	23.68
PSO A=2.0, uncontrolled	32.69	13.34	23.02

The matched comparison at $A = 1.5$, PSO and DAPO on the same controller with the same δ, separates by 1.83 points pooled in DAPO's favour, and the controlled PSO run is also below its uncontrolled twin. Both gaps are between two single runs, whose difference has an SD of about 1.4 points, and both lie inside the three-seed prediction interval, so what the comparison establishes is that holding the step did not bring PSO back to controlled DAPO's level, and no more than that. At $A = 2.0$ the pre-registered contrast was the AIME 2025 difference between the controlled run and its uncontrolled twin, with $\geq +2$ points assigned to "step size explains the dose harm" and ≈ 0 to "the reweighting itself"; it read -0.92. The two branches are only 1.2 SD apart for a difference of two single runs on AIME 2025, so the design could not separate them, and the reading is the same as at $A = 1.5$: holding the realised KL at DAPO's late-phase level did not recover the loss, within that resolution. On AIME 2024 alone the controlled $A = 2.0$ run is above its twin (34.94 against 32.69), one of the rank inversions between the two years discussed in Section 3.

The controller is not neutral on DAPO itself: the controlled DAPO run reads 0.95 points below the three-seed mean, inside the interval. Its multiplier sat at 2–3 for the first hundred steps because DAPO's natural KL is far below the late-phase δ there, so holding a late-phase statistic constant from the start changes the early trajectory; the realised statistic tracked δ throughout, so the instrument is accurate and a constant target is simply a different schedule from DAPO's rising one. A one-line alternative to the controller is to drop the group-std division (Dr. GRPO): the largest advantage falls from 5.480 to 1.9375 and the realised KL step halves (median $0.52\,\delta$) with no controller at all. That run reads 32.66 / 16.01 / 24.33, within 0.03 of the controlled DAPO run. The runs that moved DAPO's step trajectory in either direction therefore landed 0.9–1.0 points below the three-seed mean, inside the interval; we record the agreement and do not read it as an effect.

Table 9: Pre-registered interpretation, fixed before the readout.

Controlled PSO lands at	Conclusion
DAPO's level	PSO's deficit was step size, not credit assignment
above DAPO	the reweighting helps; step size was masking it
the twin's level	the deficit is in the credit assignment itself

The controlled DAPO run is what licenses reading the controlled PSO run as "PSO at DAPO's late-phase step size" rather than "PSO at some smaller step size": its multiplier settled near 1 once DAPO's natural KL reached δ. The comparison between the two controlled runs is one single run against another and is read with that resolution.

L IRT-GUIDED PROMPT SELECTION

Every method so far operates on prompts that have already been rolled out. This one changes which prompts get rolled out at all. It is the only entry in this paper that touches the axis identified at the end of Appendix D.3: dynamic sampling discards 75.2% of generated prompts, and, on the premise the method was designed on, most of those discards were problems the model could not yet solve, the ones that never contribute a gradient. The premise was wrong, as the method's own observer showed (Appendix L.5); the design is given here as it was built, and the measurement follows.

L.1 MOTIVATION: THE FILTER IS AN ORACLE WE QUERY BLINDLY

A group carries gradient if and only if its outcomes are mixed: a uniformly correct or uniformly wrong group has $\sigma = 0$ in equation 8 and $\hat{A}_i = 0$ throughout. Dynamic sampling therefore generates $G = 32$ responses per prompt *just to find out* whether the group will be mixed, then discards three quarters of the generated prompts. Meanwhile the surviving distribution drifts: the mean correct fraction among kept groups rises from 0.06 to 0.58 over training, i.e. the batch fills with problems the model already half-solves.

The idea is to predict mixedness before generating. This requires a persistent identity for prompts across the run, which the dataset happens to provide: DAPO-Math-17k contains $17{,}398$ unique prompts, each repeated ≈ 103 times, interleaved. A 400-step run draws $102{,}400$ prompts, so each unique prompt is visited ≈ 5.9 times—and every visit already produces a 32-sample estimate of its solve rate as a byproduct of the rollouts, whether or not the filter keeps the group. *The counts are already available at no extra cost; only the bookkeeping was missing.*

L.2 CONNECTION TO ITEM RESPONSE THEORY

The structure is that of psychometric testing: a latent ability (the policy) meets items of latent difficulty (the prompts), producing binary outcomes. Computerised adaptive testing (CAT) selects the next item to maximise Fisher information, which for binary items peaks where the success probability is near $1/2$—the prompts most likely to produce mixed groups. The correspondence is exact: in GRPO the gradient *exists* only where outcomes are mixed, so information-maximising item selection and gradient-seeking prompt selection are the same criterion. Two adaptations are needed: the ability drifts (the policy improves), and the selection objective is $\Pr[\text{mixed}]$ rather than Fisher information.

L.3 THE TRACKER: A DISCOUNTED-BETA FILTER

Each prompt j carries a Beta belief over its current solve rate p_j. A visit at global step t rolls G responses and observes k correct—a Binomial observation—giving the conjugate update

$$\alpha_j \leftarrow \alpha_j + k, \qquad \beta_j \leftarrow \beta_j + (G - k). \tag{48}$$

Because p_j drifts as the policy improves, evidence ages. Before a belief that is Δt steps old is used or updated, it is discounted toward the prior with half-life H:

$$\lambda = 2^{-\Delta t/H}, \qquad \alpha_j \leftarrow \alpha_0 + \lambda(\alpha_j - \alpha_0), \qquad \beta_j \leftarrow \beta_0 + \lambda(\beta_j - \beta_0), \tag{49}$$

with $(\alpha_0, \beta_0) = (1, 1)$ and $H = 150$. Equation equation 49 is the conjugate analogue of a Kalman filter's process noise; the original design sketch used the Kalman form directly, but on a probability whose mass piles up near zero—which is where most of the discarded prompts live—the Gaussian is a poor approximation and the Beta form is exact.

The selection score is the probability that a fresh group of G comes out mixed, and under a Beta belief it has a closed form:

$$\Pr[\text{mixed}] = 1 - \mathbb{E}[p^G] - \mathbb{E}[(1-p)^G] = 1 - \prod_{i=0}^{G-1} \frac{\alpha + i}{\alpha + \beta + i} - \prod_{i=0}^{G-1} \frac{\beta + i}{\alpha + \beta + i}. \tag{50}$$

No quadrature or sampling is needed. The score's behaviour does the right things automatically: an unseen prompt under the $\mathrm{Beta}(1, 1)$ prior scores $1 - 2/(G{+}1) = 0.9394$ at $G = 32$, so fresh

prompts are attractive and exploration needs no separate bonus; a single all-wrong visit drops the score to 0.492 and a second to 0.329, so hopeless prompts are demoted fast; and equation 49 revives them as the policy improves, which is when a formerly hopeless prompt becomes worth retrying.

L.4 SELECTION

The training loop is untouched; the dataloader is wrapped. Incoming prompts accumulate in a candidate pool of $4\times$ the generation batch. Each round scores the pool with equation 50 and emits a batch of the usual size: a fraction $1 - \epsilon$ taken greedily by score with duplicate prompts excluded (the dataset's ≈ 103 copies would otherwise waste slots), and a fraction $\epsilon = 0.25$ drawn uniformly from the remainder, which maintains exploration and anchors the distribution. Unselected rows remain pooled; beyond twice the target size the lowest scorers are evicted—their evidence lives in the tracker, and further copies keep arriving from the stream. Per-step compute is identical to the baseline: the same number of prompts is rolled out; only their identity changes.

Outcome observation hooks the reward function, which in the DAPO loop receives every generated group *before* filtering—the unbiased place to watch. Rewards, advantages, loss, and optimiser are all stock DAPO.

L.5 MEASURED BEHAVIOUR AND READOUT

The pre-registered readout is the pooled 60-problem mean@128 at steps $320/350/380$ against DAPO, with a mechanism gate: the mixed-group fraction of generated groups had to rise materially above the 24.8% implied by the discard statistic, since a selector that does not move the acceptance rate is not doing anything, whatever the accuracy says. Unit tests cover the closed form equation 50 against Monte Carlo, the discounting, hashing invariances, deduplication, pool eviction, and checkpoint round-trips.

The gate passed, and what it revealed. With the tracker selecting from a 256-prompt generation batch, the mixed fraction of generated groups rose from 17% at step 1 (tracker empty) to 76% by step 24 and stayed at 72–78% thereafter; a second generation batch was needed only at steps 1–7. The pre-registered control, which sets $\epsilon = 1$ so that every prompt is drawn uniformly and the tracker never influences selection, showed the same thing without the tracker: under random selection the mixed fraction is 60–76% (interdecile over steps 100 onward, Section 4). The "24.8% retained, 75.2% discarded" figure that motivated this method was the ratio of trained to generated prompts, $64/256$, fixed by the generation-batch configuration: once 64 mixed groups are found the trainer discards the rest, mixed or not. It was never a measurement of how many groups are uniformly correct or wrong. The population the method was designed to rescue, prompts the model never solves, was small, not three quarters of the data.

Compute. At a 128-prompt generation batch (pool held at $1{,}024$) the tracked run generated 128.4 groups per step and needed a second generation batch at 2 of 361 steps after step 20; the random control generated 128.0 and needed none. The tracker therefore contributes nothing to the compute saving; the saving is the halved generation batch, available because the mixed fraction under the trained policy is far above 25%. In this single-run pairing the 128-prompt runs took 7.9 minutes per step against 11.1–11.5 for the 256-prompt runs (-30%); over three seeds on each side the main text reports -24% (Table 2). The tracked 128-prompt run's own mixed fraction was 59–65% over its last 150 steps, lower than under random selection because its greedy share re-draws prompts already observed to be mixed and narrows the distribution it samples; it still exceeded the 64 groups the trainer needs.

Accuracy. Against the three-seed DAPO mean (25.31 pooled), the tracked run at the full 256-prompt batch reads 31.74 / 15.14 / 23.44 (-1.87), the tracked run at 128 reads 32.57 / 18.63 / 25.60 ($+0.29$), and the random control at 128 reads 34.34 / 15.36 / 24.85 (-0.46), the best AIME 2024 figure of any run in the study beside one of the lower AIME 2025 figures. All three are inside the three-seed prediction interval. Selection statistics offer an explanation for the 256-prompt tracked run's position: its selected median score stayed at the prior value throughout, so the greedy share (192 of 256 prompts) kept re-drawing prompts already observed to be mixed and narrowed the training distribution, while at 128 the greedy share is half as large. That explanation does not,

however, account for the random control landing below the tracked run at the same batch, and three runs on the same code path span 15.1–18.6 on AIME 2025, which is the size of run-to-run noise (Section 3). In sum, item-response-style selection bought no compute and no accuracy that the seeds can see; what it exposed, the truncation, is the configuration finding of Section 4, whose accuracy side was then settled with three seeds on each side.

M MEASURED DIAGNOSTICS

The quantities below are logged at every training step and confirm that each variant does what its equations say. All are averaged over valid positions.

Table 10: Step-1 weight statistics, all at $A = 2.0$, $\rho = 0.2$ unless noted. $\overline{s}$ is the mean rank/max scale over the pivotal set; $\mathrm{std}(w)$ is the spread of the pre-normalisation weights.

Variant	$\overline{s}$	$\overline{w}$	$\mathrm{std}(w)$	pivotal fraction
PSO v1 (hard cutoff)	—	1.199	≈ 0.40	0.20
PSO v2 (max-normalised)	0.15	1.030	0.080	0.1998
PSO v5 (rank-normalised)	0.5006	1.100	0.2386	0.1999
v5 at $A = 1.5$	0.5006	1.0500	0.1191	0.1999
v5 at $A = 1.2$	0.5006	1.0200	0.0476	0.1999
PSO v6 (signed), at $A = 1.5$	0.50	1.050	0.119	0.1999

Three points follow.

First, the $\mathrm{std}(w)$ column tracks equation 22 to three decimals across all three values of A, so the closed form can be used to set the intervention strength directly.

Second, the pivotal fraction is 0.20 for every variant. Stage 1 is unchanged; v2 and v5 differ only in stage 2, which is what makes their comparison controlled.

Third, v2's $\mathrm{std}(w) = 0.080$ against v1's ≈ 0.40 shows the soft weighting introduced in v2 weakened the perturbation fivefold while appearing to leave A untouched.

Dynamic-sampling yield. Measured over a full run, the filter of Appendix D.3 retains 24.8% of generated prompts and 75.2% are discarded—but, as Appendix L establishes, mostly by truncation to the training batch rather than as uniformly correct or uniformly wrong; the non-mixed fraction proper is 24–40% over steps 100 onward (the complement of the 60–76% interdecile mixed rate). The mean fraction of correct responses among *surviving* groups rises from 0.06 early in training to 0.58 by step 320, i.e. the training distribution drifts steadily toward problems the model already partially solves.