SELF-IMPROVEMENT CAN SELF-REGRESS: THE RISE-AND-COLLAPSE FAILURE MODE OF LLM SELF-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-improvement can self-regress. In verifiable-reward code RL, continued optimization can first improve a language model and then erase those gains. We study this instability with Qwen-2.5-3B/7B on custom competitive-programming splits drawn from HumanEval, MBPP, and APPS, using binary CodeGrader reward and chained training campaigns. In a 200-step Qwen-2.5-7B REINFORCE diagnostic run, the *online* success signal rises from 25% to 81% within roughly 50 updates and later falls to nearly zero; checkpoint-continuation tests show that checkpoints from different parts of this trajectory lead to sharply different downstream outcomes. We therefore ask an operational question: When does regression become actionable, and where should the control loop intervene?

We separate two temporal quantities: a *within-campaign cliff*, measured by the loss from a campaign's online peak to its endpoint, and *cross-campaign endpoint persistence*, measured from the sequence of checkpoints passed forward through the chain. The distinction exposes a striking asymmetry. On the matched 7B held-out splits, the frozen base reaches 10.4% [8.1, 12.8], vanilla REINFORCE 11.8% [5.2, 18.3], and GRPO 20.7% [15.7, 25.1] over five seeds, yet the mean within-campaign peak-to-end gap remains nearly unchanged under REINFORCE and GRPO (17.6 vs. 16.5 points). A matched REINFORCE+leave-one-out-baseline (RLOO-style) control reaches 13.0% [8.5, 17.5] while retaining a 17.0-point mean peak-to-end gap. **GRPO raises the floor, but does not remove the cliff.** The RLOO control further shows that this endpoint gain is not reproduced by adding a simple baseline alone.

Control effectiveness is also regime-dependent. A simple campaign-end controller raises 3B end-of-chain pass@1 from 4.9% to 9.5% across five matched seeds (paired bootstrap difference [+0.4,+8.9] points), but provides no clear gain at 7B. A trajectory-informed peak-rollback/budget schedule reaches 22.2% [14.1,28.0] at 7B ($n = 3$). Turning the causal-prefix three-decline rule into a live controller raises 7B held-out end-of-chain pass@1 to 25.6% [20.3,30.4] over five seeds and reduces the mean peak-to-end gap from 17.6 to 6.2 points. The central lesson is that final performance is not a sufficient statistic for training stability: update-rule changes can raise the floor while same-campaign control directly suppresses the cliff. We advocate reporting local peak retention, failure geometry, endpoint persistence, and detection lead time alongside final pass@1 when evaluating self-improving RL systems.

1 INTRODUCTION

RLVR has become a standard tool for improving mathematical and code reasoning because correctness can be scored automatically. The usual mental model is monotone: a policy explores, receives verifiable reward, and gradually sharpens useful behavior. In practice, the optimization trajectory can be far less benign. Figure 1 shows a diagnostic Qwen-2.5-7B REINFORCE run on our HumanEval/MBPP/APPS competitive-programming testbed. Online CodeGrader success rises from 25% to 81% in roughly 50 updates, then reverses and approaches zero by step 200. More training first helps and then destroys the behavior the reward is intended to reinforce.

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.

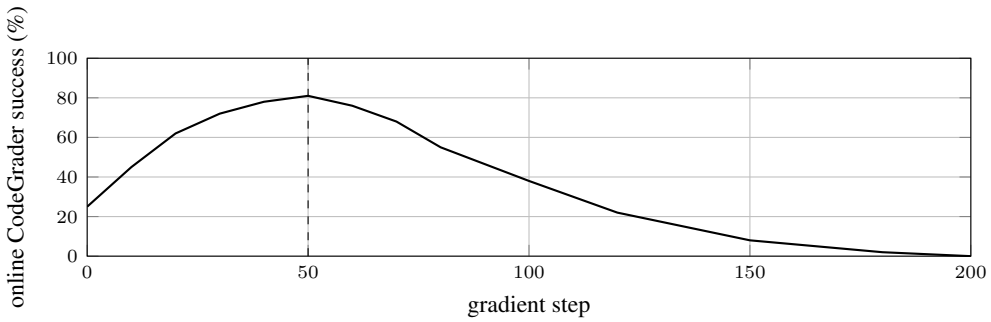

Figure 1: **Diagnostic rise-and-collapse.** One 200-step Qwen-2.5-7B REINFORCE run. The curve is a training-time CodeGrader signal, not the fixed held-out metric used for headline comparisons.

Recent work has revealed several forms of instability in iterative post-training, including self-improvement reversal, temporal forgetting, entropy and template collapse, and sensitivity to checkpoint selection (Wu et al., 2025; Li et al., 2026; Jin et al., 2026; Hao et al., 2026; Xu et al., 2026; Wang et al., 2026; Zheng et al., 2026; Nguyen et al., 2025). These results motivate a complementary operational question: *once regression exists, where in the training loop is there still enough information to act on it?* A controller can use a coarse signal after a campaign, exploit fine-grained timing from the trajectory, or alter the update rule itself. These interventions consume different information and act at different temporal resolutions. Treating them as interchangeable obscures why one can improve final performance while another changes the shape of the trajectory.

We decompose chained RL into two failure modes. **Within-campaign collapse** asks how much performance is lost between the best point reached in a campaign and its endpoint. **Between-campaign carryover** asks whether the checkpoint handed to the next campaign preserves gains across the chain. This distinction explains three findings in our testbed. First, a campaign-end controller can choose locally better checkpoints yet fail to improve 7B long-horizon performance. Second, the location of the within-campaign peak contains useful control information. Third, GRPO can nearly match a trajectory-informed REINFORCE schedule at the end of the chain while leaving the local peak-to-end drop almost unchanged.

Our contributions are threefold. **(1) Target-metric rise-and-collapse.** We document a regime in which the optimized reward signal itself reverses under continued verifiable-reward training, and we show with checkpoint-continuation experiments that late checkpoints can be materially worse starting points for future training. **(2) A two-timescale diagnosis.** We separate local peak loss from cross-campaign carryover and quantify when the cliff becomes visible. In the 7B REINFORCE traces, most post-peak deterioration is concentrated in a single update and appears late in a 20-step campaign, making an endpoint-only signal intrinsically delayed. **(3) Different controls move different timescales.** At 7B, GRPO improves end-of-chain pass@1 from 11.8% to 20.7% but leaves the mean within-campaign peak-to-end gap essentially unchanged (17.6 vs. 16.5 points). Its endpoint sequence is more persistent, consistent with better carryover. This shows that final performance alone is insufficient to identify *how* a training intervention stabilizes a self-improving loop.

2 Related Work

Self-improvement regression and temporal forgetting. Wu et al. (2025) introduce self-improvement reversal: iterative post-training can raise pass@1 while harming diversity, harder-problem solving, and out-of-distribution generalization. Their key reversal is cross-capability. Our focus is complementary: the optimized training signal itself can reverse, and we study the temporal structure of that reversal. Li et al. (2026) show that individual reasoning problems solved at earlier checkpoints are often forgotten later and exploit this temporal diversity at inference time. Their unit of analysis is which problems are stored in which checkpoints; ours is the geometry of campaign-level optimization and the distinction between transient peaks and retained endpoints.

Collapse in RLVR. A fast-growing literature studies entropy loss during RLVR. Jin et al. (2026) analyze how clipping, off-policy updates, and data diversity affect entropy; Hao et al. (2026) derive

token-level entropy-change factors and propose adaptive token weighting; and Xu et al. (2026) interpret collapse through imbalanced on-policy entropy flow. Wang et al. (2026) show a different template-collapse failure mode in agentic RL, while Zheng et al. (2026) characterize prosperity-before-collapse under stale off-policy data. Our focus is complementary: we ask what the observed trajectory implies for *control placement*—when does the failure become visible, and what kind of intervention can still affect it?

Checkpoint selection and stopping. Final-checkpoint selection is known to be unreliable in unstable RL fine-tuning. Nguyen et al. (2025) rank checkpoints using uncertainty on difficult examples without repeated validation passes, while Temporal Sampling (Li et al., 2026) explicitly uses earlier checkpoints at decoding time. We use checkpoint and budget controls to isolate temporal resolution, enabling a direct comparison of coarse endpoint control, fine-grained trajectory timing, and an update-rule change within the same chained protocol.

3 Setup and Two-Timescale Metrics

Testbed. We train Qwen-2.5-3B-Instruct and Qwen-2.5-7B-Instruct (Yang et al., 2024) on competitive-programming problems drawn from HumanEval, MBPP, and APPS (Chen et al., 2021; Austin et al., 2021; Hendrycks et al., 2021). Correctness is scored by executing generated code. Unless otherwise noted, the headline setting is ten sequential campaigns with a nominal 20-gradient-step budget per campaign. Each campaign uses 50 training problems and 30 held-out evaluation problems; the checkpoint selected from campaign c initializes campaign $c+1$. Training samples 16 completions per prompt at temperature 0.7 with learning rate 10^{-6}. We use five seeds for the main REINFORCE and GRPO conditions, with separately reported three-seed trajectory-timing combinations.

Why chained campaigns? A single long run can reveal local instability but cannot separate transient improvement from retained improvement. Chaining forces every selected endpoint to become the next campaign's initialization. It therefore exposes a second failure mode that one-shot curves hide: even if a campaign reaches a useful transient policy, does enough of that gain survive for improvement to compound?

Online versus held-out metrics. We distinguish the training-time CodeGrader success $r_{c,t}$ from held-out checkpoint pass@1. The former provides the per-step trajectory and control signal; the latter measures end-of-chain checkpoint quality. Our held-out suites are custom competitive-programming slices drawn from the same benchmark sources, and all headline method comparisons use the same evaluation protocol.

For campaign c with online trajectory $r_{c,1:T_c}$, define

$$P_c = \max_t r_{c,t}, \qquad E_c = r_{c,T_c}, \qquad \mathrm{Gap}_c = P_c - E_c. \tag{1}$$

Gap_c is the **within-campaign loss**. To quantify whether the drop is cliff-like, define

$$\mathrm{Cliff}_c = \frac{\max_{t>t_c^*}(r_{c,t-1} - r_{c,t})_+}{P_c - E_c + \epsilon}, \tag{2}$$

where t_c^* is the first peak index. $\mathrm{Cliff}_c \approx 1$ means most of the post-peak deterioration occurs in one update. Across campaigns, endpoint persistence is summarized by the sequence of checkpoints handed forward through the chain; Figure 2 tracks the corresponding online endpoint sequence. A method can improve long-horizon performance by reducing Gap_c, by preserving stronger endpoints across campaigns, or by doing both.

Controls. We use three probes. **Campaign-end control (CEC)** is a scalar checkpoint controller: after a campaign finishes, it uses the observed end/peak ratio to decide whether to keep the endpoint, roll back to a peak checkpoint, or shorten/retry the next campaign. CEC tests whether a coarse post-hoc signal is sufficient. **Trajectory-informed budgeting (TB)** rolls forward the peak checkpoint and sets campaign $c+1$'s maximum steps to $\mathrm{clamp}(\mathrm{peak_step}(c) + 3, 8, 20)$. It therefore uses fine-grained timing from the previous campaign but is not a live detector for the current one. **Live causal-prefix control** monitors the current campaign online and, after three consecutive declines

Table 1: **What each intervention can observe and control.** The rows are not an accuracy ranking; they identify temporal resolution. The central hypothesis of the paper is that a controller cannot repair a failure that becomes visible only after its control window has closed.

Probe	Signal available	What it can change
Campaign-end control (CEC)	peak/end summary after campaign c	checkpoint or budget used by $c{+}1$
Trajectory-informed budget (TB)	peak location from campaign c	length and initialization of $c{+}1$
GRPO	group-relative rewards at each update	update direction inside every step

Table 2: **Checkpoint-continuation test.** Each row starts a new 20-step continuation from a checkpoint of the diagnostic run. "Start" and "end" are held-out pass@1 on this diagnostic slice, not the headline long-chain evaluation. The result shows that pre- and post-collapse checkpoints are not interchangeable initializations and that checkpoint timing materially changes subsequent training behavior.

Initialization	Start	End after continuation	Interpretation
step 50 (near peak)	0.63	**0.81**	gain retained
step 100 (degrading)	0.31	0.77	recovery possible
step 150 (collapsed)	0.41	0.00	continuation fails

exceeding 0.02 from the running peak, stops and rolls back to the running-peak checkpoint. Finally, **GRPO** uses group-relative policy optimization while keeping the chain and evaluation stack fixed (Shao et al., 2024; DeepSeek-AI, 2025).

Statistics and scope. Headline intervals are nonparametric bootstrap 95% intervals across seeds. Per-campaign trajectory statistics aggregate the observed campaign traces, with seed-level results reported for the principal endpoint comparisons.

Research questions. The experiments are organized around three questions rather than around a proposed controller. **RQ1: Is the observed regression locally consequential?** We test whether checkpoints before and after the cliff lead to different continuation outcomes. **RQ2: When is the failure visible relative to the available control window?** We measure peak-to-end gap, cliffness, and onset within the 20-step campaigns. **RQ3: Do interventions that improve final performance stabilize the same timescale?** We compare campaign-end control, trajectory-informed budgeting, and GRPO under the same chained protocol. This organization is important because two methods can end at similar pass@1 while changing different dynamical quantities.

4 Rise-and-Collapse Has an Actionability Window

The long diagnostic curve motivates the phenomenon, but the more important question is whether checkpoint timing changes downstream behavior. Starting fresh 20-step continuations from three checkpoints of the 200-step run gives a large separation. The step-50 checkpoint begins at held-out pass@1 0.63 and ends its continuation at 0.81; the step-100 checkpoint begins at 0.31 and recovers to 0.77; the step-150 checkpoint begins at 0.41 but ends at 0.00. Thus the position at which the trajectory is cut is a high-leverage decision for subsequent training.

The continuation test rules out a weak interpretation in which the late part of Figure 1 is merely a noisy training-time readout with no consequence for the learned policy. The checkpoints induce sharply different downstream trajectories under the same continuation budget. It does *not* prove irreversibility: the step-100 checkpoint recovers, and a different optimizer, data mixture, or longer budget could recover from step 150. The narrower claim is enough for control: by the time a policy has crossed far into the collapsed region, selecting that checkpoint can materially worsen the next optimization stage.

A KL stress test shows that the phenomenon persists even when reference-policy drift is constrained. Across three 50-step campaigns, no-KL REINFORCE reaches endpoints 0.06, 0.75, and 0.22 after near-unit peaks; fixed KL=0.05 ends at 0.07, 0.10, and 0.08; an increasing KL schedule ends at 0.00,

Table 3: **Reference-policy KL stress test on 7B REINFORCE.** Three 50-step campaigns on 30 held-out problems (so 1.00 can mean 30/30). Values are start / peak / end pass@1.

Setting	Campaign 1	Campaign 2	Campaign 3
no KL	.25 / 1.00 / .06	.63 / .96 / .75	.69 / 1.00 / .22
fixed KL=.05	.25 / 1.00 / .07	.56 / 1.00 / .10	.75 / .94 / .08
increasing KL	.31 / 1.00 / .00	.63 / 1.00 / .06	.63 / 1.00 / .00

Table 4: **Actionability diagnostics for 7B REINFORCE.** Campaign-level values summarize the observed 20-step traces; they are descriptive rather than independent statistical samples.

Diagnostic	Observed value
mean peak-to-end gap Gap_c	0.176
mean cliffness Cliff_c	≈ 0.78
typical collapse onset	$\approx 17/20$ steps
endpoint-only latency in same campaign	0 usable steps

0.06, and 0.00. Realized reference-policy KL remains bounded (median 0.021 under fixed KL and 0.016 under the increasing schedule), while the same entropy contraction and late gradient-norm increase accompany the cliff (Appendix D).

This stress test also motivates why we avoid describing the phenomenon as ordinary catastrophic forgetting. No task switch is required for the local drop: within each listed campaign, the objective and nominal task distribution remain fixed while the trajectory moves from peak to endpoint. Across campaigns the problem is different: the selected endpoint becomes the next initialization, so local over-optimization can be converted into long-horizon carryover loss. The two-timescale language keeps these mechanisms conceptually separate even when they appear in the same chain.

The 7B 20-step REINFORCE traces expose why endpoint-only control is limited. Across 47 campaigns with a defined post-peak drop, the mean cliffness statistic is about 0.78: most of the peak-to-end deterioration is concentrated in the largest single update. Collapse onset—the first post-peak step at which success falls below 30% of the peak—occurs near step 17 in the 20-step campaigns. An endpoint-only controller therefore has essentially no usable latency inside that campaign.

The reward cliff becomes visible late in the campaign, while optimization diagnostics begin to move earlier. Around the pre-cliff region, policy entropy contracts by 18%, gradient norm rises to 1.9×, response length changes by only 2.7%, and compile-failure rate changes by 1.1 percentage points. The combination localizes the late-stage transition to optimization dynamics rather than a simple shift in response length or code validity, and motivates earlier predictive control from trajectory-level signals.

5 Which Intervention Moves Which Timescale?

Table 5 summarizes the end-of-chain held-out pass@1 results and connects endpoint performance to the two trajectory components defined above. The frozen-base row evaluates the unmodified instruction-tuned checkpoints on the exact held-out campaign splits used by the five headline seeds, providing a matched reference for net end-of-chain change.

5.1 Campaign-end control is regime-limited

At 3B, CEC improves the point estimate from 4.9% to 9.5%; the paired bootstrap interval from five matched seeds is $[+0.4, +8.9]$ points, with four of five seed differences positive. This shows that rollback can help when naive carryover is extremely fragile. At 7B, the same controller reaches 13.8% versus 11.8% for naive REINFORCE, with overlapping intervals. This is informative precisely because local checkpoint selection is high leverage: repeatedly making the decision only after each campaign is still insufficient to solve the 7B chain.

Table 5: **End-of-chain held-out pass@1 (%).** Ten sequential campaigns. Bootstrap 95% intervals are across seeds when available. Small-n rows are descriptive.

Update	Control signal	3B	7B
Frozen base	none	4.6 [3.4, 5.8] (n=5)	10.4 [8.1, 12.8] (n=5)
REINFORCE	none	4.9 [2.1, 9.5] (n=5)	11.8 [5.2, 18.3] (n=5)
REINFORCE	campaign end (CEC)	9.5 [6.3, 12.7] (n=5)	13.8 [2.8, 27.3] (n=5)
REINFORCE	trajectory timing (TB)	8.2 (n=5)	22.2 [14.1, 28.0] (n=3)
REINFORCE	live causal-prefix	–	**25.6 [20.3, 30.4]** (n=5)
GRPO	none	6.8 (n=5)	20.7 [15.7, 25.1] (n=5)
REINFORCE + LOO baseline	none	–	13.0 [8.5, 17.5] (n=5)
GRPO	trajectory timing	–	17.0 [0.0, 28.1] (n=3)

Table 6: **7B within-campaign trajectory diagnostics.** Descriptive averages over naive 20-step campaign traces from five seeds.

Metric	REINFORCE	GRPO
campaign traces	47	50
mean peak P_c	0.343	0.369
mean end E_c	0.166	0.204
mean gap $P_c - E_c$	**0.176**	**0.165**
$\Pr(\mathrm{Gap}_c > 0.2)$	36%	38%
$\Pr(\mathrm{Gap}_c > 0.3)$	11%	6%

5.2 Fine-grained peak timing contains useful control information

At 7B, TB reaches 22.2% [14.1, 28.0] over three REINFORCE seeds, compared with 11.8% for naive training. This gain is accompanied by a clear timing signal: the *location* of the peak inside campaign c carries information about how long campaign $c + 1$ should run, whereas the coarse end/peak ratio alone does not. A prefix-only stopping rule—walking each trajectory in time order and triggering after three consecutive declines of more than 0.02 from the running peak—provides a same-campaign control signal. Deployed live over five 7B chains, this rule reaches 25.6% [20.3,30.4] held-out end-of-chain pass@1 and reduces the mean online peak-to-end gap from 0.176 to 0.062. The corresponding retrospective causal-prefix quantity is roughly 34–35% on the online traces, compared with 37.8–47.9% for a hindsight best-checkpoint selector. Together, these results show that moving the control loop from one-campaign-delayed timing to the current trajectory converts the observed actionability window into a direct reduction of the local cliff.

At 3B the pattern is different: TB reaches 8.2%, slightly below the 9.5% CEC point estimate. This cautions against a simple "finer timing is always better" conclusion. Low-latency control helps only when the signal being transferred across campaigns is stable enough to act on.

5.3 GRPO raises the floor without removing the cliff

At 7B, naive GRPO reaches 20.7% [15.7,25.1], nearly matching the 22.2% REINFORCE+TB point estimate without a checkpoint controller. Endpoint performance alone might suggest that GRPO solved the same instability. The trajectories show otherwise.

To test whether the REINFORCE–GRPO difference is merely a no-baseline versus baseline effect, we add a matched leave-one-out-baseline control under the same 10-campaign protocol. The RLOO-style condition reaches 13.0% [8.5,17.5] held-out end-of-chain pass@1 ($n = 5$), compared with 11.8% [5.2,18.3] for vanilla REINFORCE and 20.7% [15.7,25.1] for GRPO. Its mean within-campaign peak-to-end gap is 0.170, essentially the same as REINFORCE (0.176) and GRPO (0.165). Thus a leave-one-out baseline alone does not reproduce the GRPO endpoint gain, while all three estimators retain a similar local cliff. This strengthens the separation between endpoint quality and local trajectory stability.

The mean peak-to-end gap is about 17 points for both update rules, and the fraction of campaigns losing more than 0.2 from peak to endpoint is essentially unchanged. Using $E_c = P_c - \mathrm{Gap}_c$, the $+3.8$-point shift in mean online endpoint decomposes into roughly $+2.6$ points from a higher peak and only $+1.1$ from a smaller local gap (rounding explains the residual). Thus GRPO's online gain is not primarily a closing of the cliff. Across campaign positions, the mean online endpoints have

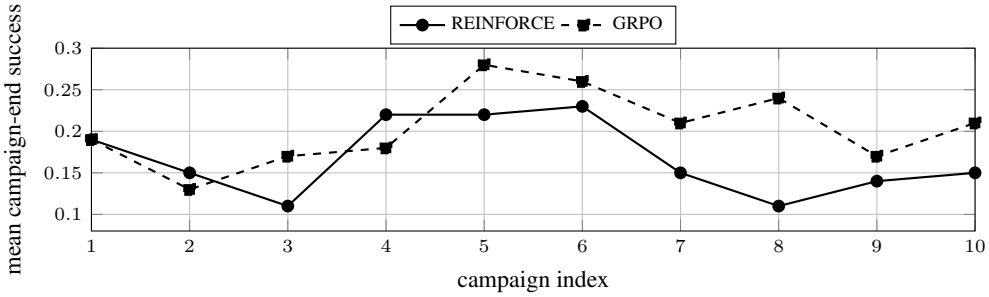

Figure 2: **Online endpoint-retention proxy at 7B.** Mean campaign-end online success over five naive chains. GRPO endpoints are more persistent although its within-campaign gap remains close to REINFORCE's.

descriptive least-squares slopes of -0.38 percentage points/campaign for REINFORCE and $+0.53$ for GRPO (Figure 2). GRPO therefore leaves the local cliff largely intact while producing a more persistent online endpoint sequence.

This result separates two questions that are often conflated. *Did the optimizer reach a better final checkpoint?* Yes: GRPO clearly raises the 7B endpoint in our chains. *Did it remove local rise-and-collapse?* No: the peak-to-end gap remains. The distinction matters because it changes what future stabilization should target. An entropy or optimizer intervention may improve carryover while a trajectory predictor or restart policy is still needed for single-campaign cliffs.

The GRPO+TB run further illustrates the persistence of the local failure. Two of three matched seeds improve over naive GRPO by 2.7–3.7 points, while a third suffers a final-campaign collapse to 0, producing a 17.0% mean. Combining trajectory timing with GRPO therefore does not automatically remove the cliff.

6 What the Decomposition Explains

The value of the two-timescale view is not another taxonomy; it is that it makes several otherwise puzzling results consistent with one another.

Similar final scores can hide different stabilization mechanisms. REINFORCE+TB and naive GRPO have similar 7B end-of-chain point estimates (22.2% and 20.7%), yet their evidence comes from different temporal signals. TB uses the previous campaign's peak timing to avoid repeatedly spending a full budget after the empirically useful region. GRPO changes every update. Table 6 then shows that GRPO's local peak-to-end gap remains close to REINFORCE's. The natural interpretation is not that one intervention is "better" in general, but that final pass@1 conflates at least two stabilization objectives: preserving useful states within a campaign and transporting useful states across campaigns.

Why endpoint rollback can be locally right but globally insufficient. Table 2 says that checkpoint selection is high leverage in a single continuation. Yet the 7B campaign-end controller does not clearly separate from naive training after ten campaigns. There is no contradiction. Rolling back prevents one bad endpoint from being inherited, but it does not prevent the next campaign from generating another cliff. A post-hoc controller can repeatedly repair the consequence without changing the process that creates it. This is why the actionability window matters more than the label attached to the controller.

Why the 3B and 7B results need not order methods consistently. At 3B, the naive endpoint is very low (4.9%), and coarse rollback raises the point estimate to 9.5%; trajectory-informed budgeting is 8.2%. At 7B, coarse endpoint control changes little, while timing and GRPO give much larger point estimates. The two scales suggest a signal-to-control matching effect: when the observed trajectory is weak or noisy, coarse rollback can protect against poor endpoints; when the trajectory signal is stronger, finer timing becomes more useful.

Table 7: **Minimal stability report for chained RL.** No single row is sufficient. Together they separate local retention, failure geometry, long-horizon carryover, and whether a controller receives the signal in time to act.

Quantity	Question answered	Failure it distinguishes
held-out end-of-chain pass@1	where did training finish?	overall utility
$P_c - E_c$	how much local peak was lost?	within-campaign retention
Cliff_c	was loss gradual or abrupt?	slope vs. cliff
held-out endpoint sequence	do useful endpoints persist?	between-campaign carryover
detection lead time	was the signal available early enough?	observable vs. actionable failure

A reproducible pre-cliff phenotype. Instrumentation around the 7B REINFORCE cliff reveals a consistent change in optimization dynamics before endpoint failure. From the local peak to the pre-cliff region, policy entropy contracts by 18% on average and the gradient norm rises to 1.9$\times$ its peak-region level, while mean response length changes by only 2.7% and compile-failure rate by 1.1 points. The same qualitative signature remains visible in the KL stress test: realized reference-policy KL stays bounded (median 0.021 under fixed KL and 0.016 under the increasing schedule), while entropy contraction and the late gradient-norm increase persist. Thus proximity to the reference policy and concentration of the current policy capture different aspects of the trajectory, explaining why the tested KL schedules can constrain drift without removing the cliff.

A concrete target for follow-up work. A stronger stabilization method should report a pair of quantities rather than only final pass@1: a local retention statistic such as $\mathbb{E}[P_c - E_c]$ and the held-out endpoint sequence across campaigns. It should also report detection lead time relative to the observed cliff. Under this evaluation, a method that raises final performance by lifting all campaign endpoints is distinguishable from one that actually suppresses the cliff. That distinction is the main practical output of this study.

7 A Minimal Reporting Protocol for Training Stability

The experiments suggest that a single end-of-training number is an incomplete stability report for self-improving RL. We therefore propose a small set of *diagnostics*, not a new benchmark or optimizer, that can be computed from traces many RLVR pipelines already log. Their purpose is to separate "the final model is better" from "the training process is more stable."

Separate online control signals from fixed evaluation. A per-step reward trace provides the temporal resolution needed for intervention, while fixed held-out evaluation measures checkpoint quality on the campaign evaluation split. We therefore use the online CodeGrader trace for geometry and timing and held-out end-of-chain pass@1 for the headline method comparisons. Reporting both signals makes the control timescale and retained utility explicit.

Report causal-prefix and hindsight quantities separately. A best checkpoint chosen after inspecting the whole trajectory answers "how much performance appeared at any time?" It does not answer whether that performance was recoverable online. A causal-prefix rule must make its decision from information available up to the current step. The gap between these two quantities measures remaining control headroom. In our 7B traces, the prefix-only three-decline rule reaches roughly 34–35%, below the 37.8–47.9% hindsight maxima but well above naive end-of-chain REINFORCE. This gap is more informative for controller design than a hindsight maximum alone.

Normalize timing by the intervention's actual resolution. A warning that arrives at step 17 can be early in a 200-step run and useless in a 20-step campaign if the only action happens after the campaign ends. "Early" is therefore not an absolute step number; it is lead time relative to the next available intervention. We recommend reporting detection lead time together with the controller's action frequency. This converts trajectory analysis into an operational question: how many optimizer

updates remain between the first reliable warning and the next point at which the system can stop, roll back, change data, or modify the update?

Why this matters for self-improvement claims. Self-improvement is longitudinal. A checkpoint that scores well once is less useful if the next training stage systematically erases it. Conversely, an optimizer that modestly improves every campaign endpoint can compound into a strong long-horizon gain even if its local trajectories still oscillate. Reporting both within-campaign retention and between-campaign carryover makes these two forms of progress visible. We view this reporting protocol as the main reusable artifact of the paper: it can be applied to future RLVR studies independent of whether their root cause is entropy collapse, off-policy staleness, reward design, or something else.

8 Discussion and Future Directions

Temporal decomposition. Our central contribution is a *temporal decomposition* of rise-and-collapse under a common chained RLVR protocol. Prior work asks why entropy or reasoning collapses, which examples are forgotten, or how to choose a better checkpoint. Our measurements ask when the failure becomes actionable and show that two interventions with similar end-of-chain scores can operate through different mechanisms: one shows a more persistent online endpoint sequence while leaving the local cliff intact.

Control resolution matters. The controller experiments expose a direct relationship between temporal resolution and the failure component being stabilized. Campaign-end rollback acts after the local cliff, trajectory-informed budgeting transfers timing information to the next campaign, and the live causal-prefix controller acts inside the current campaign. The latter reaches 25.6% held-out end-of-chain pass@1 and reduces the mean local gap to 6.2 points, showing that the actionability window identified by the trajectory analysis can be converted into same-campaign control.

Estimator control. We explicitly test a matched leave-one-out-baseline control alongside REINFORCE and GRPO to isolate the effect of adding a simple variance-reduction baseline. The RLOO-style condition reaches 13.0% [8.5,17.5] at 7B and retains an approximately 17-point local peak-to-end gap. It does not recover the 20.7% GRPO endpoint. Together, the three conditions separate the effect of a simple leave-one-out baseline from the larger endpoint shift under GRPO.

Future work: predicting the cliff. The live prefix controller establishes that same-campaign intervention can preserve gains once deterioration becomes visible. A next step is to move the warning earlier by forecasting the cliff while performance is still near its peak. Promising signals include policy-entropy contraction and entropy flow, changes in cross-prompt input dependence or output diversity, degeneration of reward/advantage variance, and increasing repetition or template concentration, with response length and code validity/execution statistics as inexpensive complementary diagnostics. Among these, entropy-change and input-dependence signals are especially promising candidates for converting the measured actionability window into predictive control.

Cross-family check. The main controlled study uses Qwen-2.5 at two scales on execution-verified code, with REINFORCE, a matched leave-one-out-baseline control, and GRPO. A Gemma-3-4B run reproduces the qualitative rise-and-collapse signature ($0\% \rightarrow 32.8\% \rightarrow 0\%$), extending the observed trajectory pattern beyond Qwen.

Measurement. We use the per-step CodeGrader signal to characterize trajectory geometry and held-out end-of-chain pass@1 for final checkpoint quality. Reporting both exposes distinctions that are invisible from endpoint accuracy alone.

Conclusion. Continued verifiable-reward training can first improve and then erase its own gains. In our chained code-RL testbed, GRPO raises the 7B final score while leaving the local peak-to-end gap nearly intact: *the floor moves, but the cliff remains*. A live causal-prefix controller acts on the complementary timescale, raising held-out end-of-chain pass@1 to 25.6% while reducing the local gap from 17.6 to 6.2 points. Stability should therefore be judged from both within-campaign peak loss and what persists across campaigns, rather than from the final checkpoint alone.

REFERENCES

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mane. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Federico Cassano, John Gouwar, Daniel Nguyen, Sydney Nguyen, Luna Phipps-Costin, Donald Pinckney, Ming-Ho Yee, Yangtian Zi, Carolyn Jane Anderson, Molly Q. Feldman, Arjun Guha, Michael Greenberg, and Abhinav Jangda. MultiPL-E: A scalable and polyglot approach to benchmarking neural code generation. *IEEE Transactions on Software Engineering*, 2023.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Xinzhu Chen, Xuesheng Li, Zhongxiang Sun, and Weijie Yu. Beyond high-entropy exploration: Correctness-aware low-entropy segment-based advantage shaping for reasoning LLMs. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 32970–32984. Association for Computational Linguistics, 2026.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 6621–6642. PMLR, 2024.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, 2017.

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

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 10835–10866. PMLR, 2023.

Zhezheng Hao, Hong Wang, Haoyang Liu, Jian Luo, Jiarui Yu, Hande Dong, Qiang Lin, Can Wang, and Jiawei Chen. Rethinking entropy interventions in RLVR: An entropy change perspective. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2026.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring coding challenge competence with APPS. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Jaeeun Jang, Hansle Lee, and Sangmin Kim. A few bad apples spoil the bunch: Preventing global entropy collapse driven by a small set of tokens in LLM reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 13134–13154. Association for Computational Linguistics, 2026.

Renren Jin, Pengzhi Gao, Yuqi Ren, Zhuowen Han, Tongxuan Zhang, Wuwei Huang, Wei Liu, Jian Luan, and Deyi Xiong. Revisiting entropy in reinforcement learning for large reasoning models. In *Findings of the Association for Computational Linguistics: ACL 2026*. Association for Computational Linguistics, 2026.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, et al. Kimi k1.5: Scaling reinforcement learning with LLMs. *arXiv preprint arXiv:2501.12599*, 2025.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114:3521–3526, 2017.

Yuetai Li, Zhangchen Xu, Fengqing Jiang, Bhaskar Ramasubramanian, Luyao Niu, Bill Yuchen Lin, Xiang Yue, and Radha Poovendran. Temporal sampling for forgotten reasoning in LLMs. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 28309–28327. Association for Computational Linguistics, 2026.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Remi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with AlphaCode. *Science*, 378:1092–1097, 2022.

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. *arXiv preprint arXiv:2305.20050*, 2023.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. *arXiv preprint arXiv:2305.01210*, 2023.

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*, 2025.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pp. 6467–6476, 2017.

Manh Nguyen, Dung Nguyen, Dai Do, Svetha Venkatesh, and Hung Le. Uncertainty-guided checkpoint selection for reinforcement finetuning of large language models. *arXiv preprint arXiv:2511.09864*, 2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023.

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

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize from human feedback. In *Advances in Neural Information Processing Systems*, 2020.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

Zihan Wang, Chi Gui, Xing Jin, Qineng Wang, Licheng Liu, Kangrui Wang, Shiqi Chen, Linjie Li, Zhengyuan Yang, Pingyue Zhang, Yiping Lu, Jiajun Wu, Fei-Fei Li, Lijuan Wang, Yejin Choi, and Manling Li. RAGEN-2: Reasoning collapse in agentic RL. In *International Conference on Machine Learning*, 2026.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

Ting Wu, Xuefeng Li, and Pengfei Liu. Progress or regress? self-improvement reversal in post-training. In *International Conference on Learning Representations*, 2025.

Can Xie, Ruotong Pan, Xiangyu Wu, Zhang Yunfei, Jiayi Fu, Tingting Gao, and Guorui Zhou. Unlocking exploration in RLVR: Uncertainty-aware advantage shaping for deeper reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 19057–19076. Association for Computational Linguistics, 2026.

Huimin Xu, Shuai Zhao, Xiaobao Wu, and Anh Tuan Luu. Understanding and preventing entropy collapse in RLVR with on-policy entropy flow optimization. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17759–17771. Association for Computational Linguistics, 2026.

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*, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, 2025.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

Eric Zelikman, Jesse Mu, Noah D. Goodman, and Yuhuai Wu. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, 2022.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 3987–3995. PMLR, 2017.

Xiaoyun Zhang, Xiaojian Yuan, Di Huang, Wang You, Chen Hu, Jingqing Ruan, Kejiang Chen, and Xing Hu. Revisiting entropy regularization: Adaptive coefficient unlocks its potential for LLM reinforcement learning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 18005–18020. Association for Computational Linguistics, 2026.

Haizhong Zheng, Jiawei Zhao, and Beidi Chen. Prosperity before collapse: How far can off-policy RL reach with stale data on LLMs? In *International Conference on Learning Representations*, 2026.

Wei Zhu, Jian Zhang, Lixing Yu, Kun Yue, and Zhiwen Tang. Dissecting failure dynamics in large language model reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 8893–8914. Association for Computational Linguistics, 2026.

AI USE DISCLOSURE

Generative AI tools were used for language editing and manuscript restructuring. All scientific claims, experimental results, citations, and final wording were reviewed and approved by the authors.

A ADDITIONAL EXPERIMENTAL DETAILS

The 200-step run in Figure 1 is a continuous diagnostic for long-horizon trajectory shape, while the headline comparisons in Table 5 use ten sequential campaigns with a nominal 20-step budget. Each campaign uses 50 training problems and 30 held-out evaluation problems. A separate 50-step setting is used for the reference-policy stress test. For each condition, marginal bootstrap intervals resample seeds with replacement and recompute the mean; paired intervals resample matched seed-level differences.

B CAMPAIGN-END CONTROL BASELINE

The campaign-end controller provides the coarse-resolution comparator. It observes a campaign after completion, computes the endpoint-to-peak ratio, and selects among carrying the endpoint, rolling back to a peak checkpoint, or using a shortened retry/pilot budget. The defining constraint is temporal: it cannot affect the campaign whose collapse generated the signal. On the 3B 10-campaign REINFORCE chains, matched seed differences relative to naive are $+11.1, +10.2, +1.7, +1.7, -1.6$ points, yielding the paired bootstrap interval $[+0.4, +8.9]$. At 7B, campaign-end control has no clear separation from naive REINFORCE.

C LEAVE-ONE-OUT BASELINE CONTROL

The leave-one-out-baseline control uses the same Qwen-2.5-7B model, 10 sequential 20-step campaigns, prompts, sampling configuration, learning rate, and evaluation pipeline as the headline REINFORCE and GRPO chains. The only intended change from vanilla REINFORCE is the per-group leave-one-out baseline used to center the policy-gradient advantage. The control reaches 13.0% [8.5,17.5] held-out end-of-chain pass@1 over five seeds and has a mean online peak-to-end gap of 0.170.

D REFERENCE-POLICY KL STRESS TEST

	Campaign 1			Campaign 2			Campaign 3		
Method	Start	Peak	End	Start	Peak	End	Start	Peak	End
No KL	.25	1.00	.06	.63	.96	.75	.69	1.00	.22
KL=0.05	.25	1.00	.07	.56	1.00	.10	.75	.94	.08
Adaptive KL	.31	1.00	.00	.63	1.00	.06	.63	1.00	.00

Across the tested coefficient schedules, reference-policy regularization does not remove the observed rise-and-collapse pattern.

Optimization diagnostics. Across the fixed-KL and increasing-KL runs, median realized reference-policy KL is 0.021 and 0.016, respectively. Around the observed cliffs, policy entropy contracts while gradient norm rises, whereas response length and compile-failure rate remain comparatively stable. This separates reference-policy proximity from the within-policy concentration associated with the late-stage cliff.

E LIVE CAUSAL-PREFIX CONTROL

We deploy the same three-decline prefix rule used in the retrospective trace analysis as a live controller in five Qwen-2.5-7B REINFORCE chains. At each step, the controller compares the current online

score with the running peak; after three consecutive declines exceeding 0.02, it terminates the campaign and rolls back to the running-peak checkpoint before continuing the chain. End-of-chain held-out pass@1 is 25.6% [20.3,30.4], and the mean online peak-to-end gap falls from 0.176 under vanilla REINFORCE to 0.062.

F GRPO + Trajectory-Timing Run

Three matched Qwen-2.5-7B chains combine GRPO with the same trajectory-informed budget rule. The controller uses the one-campaign-delayed $\mathrm{peak_step}(c)+3$ rule. End-of-chain held-out pass@1 is 0.00 vs. 11.28 for matched naive GRPO on seed s17, 28.08 vs. 25.35 on s23, and 22.82 vs. 19.13 on s42. These runs further distinguish one-campaign-delayed timing from the same-campaign live controller used in the main control experiment.

G Cross-Family Pilot

A short-chain C++ condition in the original experiment set provides a cross-language sanity check. A single Gemma-3-4B pilot uses the same 20-step REINFORCE/CodeGrader setup. Online success starts at 0%, peaks at 32.8% near step 15, and ends at 0%. This provides an initial cross-family reproduction of the qualitative Qwen signature.

H Extended Related Work

Self-training, reasoning RL, and reward optimization. Our setting sits within a broader line of iterative self-training and self-rewarding methods (Zelikman et al., 2022; Yuan et al., 2024; Chen et al., 2024; Rafailov et al., 2023), verifiable-reward reasoning and large-scale RL systems (Cobbe et al., 2021; Lightman et al., 2023; Shao et al., 2024; DeepSeek-AI, 2025; Yu et al., 2025; Liu et al., 2025; Kimi Team et al., 2025), and the classical policy-gradient/RLHF lineage (Sutton & Barto, 2018; Williams, 1992; Schulman et al., 2017; Christiano et al., 2017; Stiennon et al., 2020; Ouyang et al., 2022; Bai et al., 2022). Reward-model over-optimization provides a related example in which continued optimization of a proxy can reduce true utility (Gao et al., 2023; Amodei et al., 2016).

Collapse diagnostics, code evaluation, and forgetting. Recent 2026 work studies RLVR collapse through entropy, token-level credit assignment, exploration, off-policy staleness, template collapse, and localized reasoning failures (Jin et al., 2026; Hao et al., 2026; Xu et al., 2026; Zhang et al., 2026; Jang et al., 2026; Xie et al., 2026; Chen et al., 2026; Wang et al., 2026; Zheng et al., 2026; Zhu et al., 2026; Li et al., 2026). Our code testbed builds on established execution-based benchmarks and evaluation methodology (Chen et al., 2021; Austin et al., 2021; Hendrycks et al., 2021; Li et al., 2022; Liu et al., 2023; Cassano et al., 2023; Wang et al., 2023). Finally, our within-task self-regression framing is distinct from task-switch continual-learning methods such as EWC, SI, and GEM (Kirkpatrick et al., 2017; Zenke et al., 2017; Lopez-Paz & Ranzato, 2017).