# ACCEPTED BUT UNFINISHED: THE SALVAGE VALUE OF TRUNCATED ROLLOUTS IN RLVR

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

A policy trained with verifiable rewards can learn to place an answer its verifier accepts and then keep generating until the response cap. The reason is the default scoring path: the four open RLVR stacks we inspected grade whatever text is retained at the cap, so a truncated rollout is not scored zero but earns its *salvage value*, the mean reward paid to rollouts that hit the cap. This one unreported quantity governs what the policy learns, what the pass rate measures, and what the update does. *Learning:* in one default-grader run all $848$ easy-stratum rollouts are accepted while $11$ complete; in another, completed-and-correct output falls from $0.70$ to $0.015$ in a single update while two thirds of truncated outputs still earn full credit. *Measurement:* salvage value is exactly recoverable from three logged scalars, verified per difficulty stratum across five policy families; since it enters the pass rate as $q = a - (a - b)\tau$, the same policy is ranked differently at different caps without training ($44.7$ pp swing, rank correlation $0.60$). *Update:* salvage is a distinct credit term, the covariance of reward with policy score within truncated rollouts; a preregistered matched fork that pins truncated reward at the checkpoint's own salvage value finds the default-grader arm truncating more in four of six seeds. The published treatments of overlong outputs act on different terms of the same decomposition: a fixed high reward costs completion in all five seeds, masking costs accuracy in nine of twelve matched cells, and neither substitutes for the other. Salvage value is the coordinate along which acceptance and completion separate; it is readable from existing logs and belongs next to the pass rate.

## 1 INTRODUCTION

An answer can be accepted without its generation being complete. In reinforcement learning with verifiable rewards (RLVR), a policy generates responses, a programmatic verifier scores them, and group-relative rewards drive the update (Shao et al., 2024; Guo et al., 2025; Yu et al., 2025). A response budget $B$ ends generation when the policy has not stopped. If the verifier accepts an answer already present in that prefix, the rollout can still receive full credit. The training objective then rewards answer production without necessarily rewarding reliable termination.

Figure 1 shows the consequence in two default-grader runs, one that saturates and one that collapses within a single update and recovers. In both, reward paid to truncated outputs rises while the fraction of completed-and-correct responses falls toward zero, and reloaded checkpoints reveal accepted answers followed by continued generation to the cap. This is not an inability to solve the problem: *answer acceptance and completion have separated*, and we measure them separately: strict accuracy requires completion and verifier acceptance; prefix acceptance records what the grader recognises before forced termination. Neither metric alone describes the other.

We call the mean reward paid at truncation the *RLVR salvage value*, inspired by terminal payoffs in finite-horizon control (Bertsekas, 2017; Kamien & Schwartz, 1991). It is a delivered reward, not the counterfactual value of continuing an unfinished answer. Write $b$ for the population quantity and $b^\star$ for its empirical reading. If $N_T$ of $N$ rollouts truncate,

$$\underbrace{b^\star}_{\text{implied salvage value}} := \frac{1}{N_T} \sum_{i:\, s_i = \mathrm{T}} r_i = \frac{\bar{r} - \bar{r}_{\mathrm{strict}}}{\tau}, \qquad \tau = \frac{N_T}{N} > 0. \tag{1}$$



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.

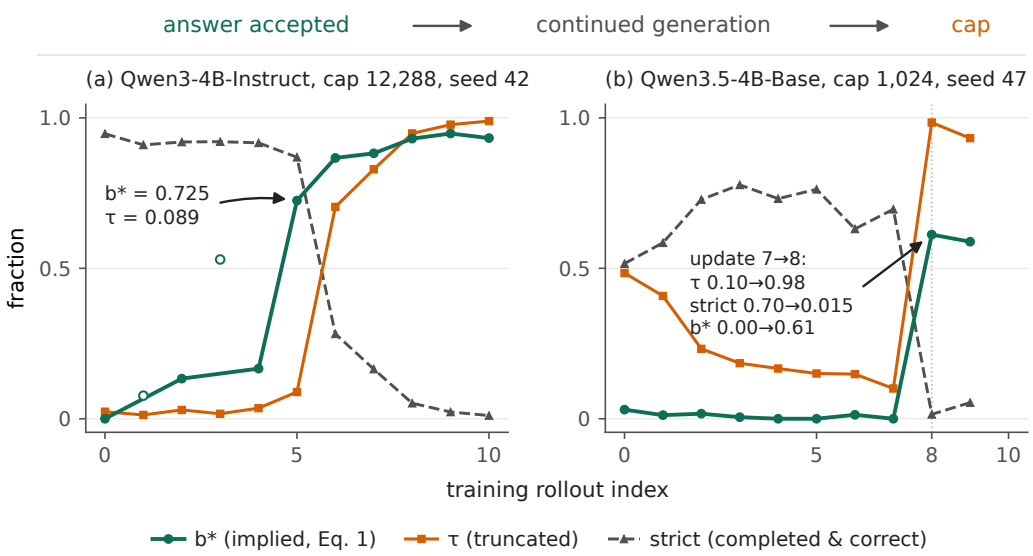



Figure 1: **Accepted answers need not produce completed responses.** The top strip illustrates the behaviour, not a sampled response. Below are all recorded training batches from two default-grader runs: Qwen3-4B-Instruct at cap 12288 (seed 42), which saturates, and Qwen3.5-4B-Base at cap 1024 (seed 47), which collapses between updates 7 and 8 and recovers. The implied salvage value is Eq. (1); strict accuracy is the completed-and-correct fraction; open $b^\star$ markers have $\tau \leq 0.02$ and are not joined. Left: $b^\star = 0.725$ at rollout 5 while $\tau = 0.089$, ending at $(0.933, 0.989)$. Right: $\tau$ jumps from 0.10 to 0.98 and $b^\star$ from 0.00 to 0.61 in one update; $b^\star$ is near zero beforehand because this policy rarely writes an accepted answer within 1024 tokens. Two same-seed arms with masked or pinned truncated reward show no excursion (Table S5). These are case trajectories, not failure rates. Batch-level readings mix within-stratum change with change in which samples truncate; Appendix S8 separates them. The third case, at cap 4096, is in Appendix S8; reloaded outputs are in Table 2.

Here $\bar{r}_{\text{strict}} = N^{-1} \sum_i r_i \mathbf{1}\{s_i = \text{C}\}$ is strict correctness when completed rewards are binary. All three logs use the same samples and weights. Recovery is an exact sample identity; estimating a population value still has sampling uncertainty. Fixed-reward runs provide a bookkeeping check with a known answer.

Existing recipes for overlong outputs vary reward assignment, loss masks and budgets (Table 1); the same decomposition places each on a different term, so their effects become comparable without assuming a common dose–response curve.

**Contributions.**

1. **Acceptance and completion come apart under the default grader, and we catch it in the act.** In three default-grader runs the policy learned to be accepted without finishing: two saturate at the cap, and one collapses within a single update from 0.70 to 0.015 completed-and-correct while 604 of 912 truncated easy-stratum rollouts still earn reward 1, then recovers. In the saturating run all 848 easy-stratum rollouts are accepted while 11 complete, and reloaded checkpoints show first-box positions moving from 0.72 to 0.07 of response length. Two arms launched from the collapsing run's seed on the same launcher, differing only in truncation handling, show no excursion (Section 5).
2. **The credit that drives this is measurable, and it is part of the score.** The default scoring paths of four open RLVR stacks grade retained text without using truncation status; the salvage value they pay is an unreported label. Equation (1) recovers it exactly from three logged scalars, and on the calibration arms returns the installed constant to $10^{-14}$ per difficulty stratum across five policy families (712 cells). Under the default grader positive salvage is common but family-dependent, paid in about half of low-truncation cells pooled and from 77% to 3% across five families, in the order of how often each writes an accepted answer before the cap. Because salvage enters the

| treatment | example | operation |
|---|---|---|
| default prefix grading | inspected scoring paths | reward depends on the text retained at the cap |
| overlong masking | Yu et al. (2025) | removes token losses; the reward baseline must be specified |
| soft overlong punishment | Yu et al. (2025) | adds a length-dependent penalty; total credit depends on the base reward |
| budget growth | Luo et al. (2025) | changes truncation exposure for the same policy |
| budget-dependent filtering | Liu et al. (2025d) | evaluates the filter at different generation caps |
| explicit zero at truncation | Liu et al. (2025b) | installs $b = 0$ instead of leaving credit to the parser |

Table 1: **The label "overlong handling" hides different operations.** A stated reward rule, a token-loss mask and a generation budget describe different parts of an implementation. DLER explicitly states zero truncation credit; this does not measure the evolving conditional reward of a default prefix grader.

pass rate as $q = a - (a - b)\tau$, the same policy receives different difficulty rankings at different caps without learning: a $44.7$ pp zero-training swing, rank correlation $0.60$ (Sections 3, 4).

3. **Salvage is a distinct term in the update, and removing it changes what the policy does.** The gradient decomposes as $\nabla P(C) + b\,\nabla P(T) + P(T)\nabla b$; the last term is the covariance of reward with policy score *within* truncated rollouts. A preregistered matched fork pins truncated reward at the checkpoint's own salvage value: the default-grader arm truncates more in four of six seeds, with the largest gaps where that variance is largest. A fixed high reward removes the term but costs completion in every one of five seeds (12–41 pp), and masking costs accuracy in nine of twelve matched cells ($-5.3$ pp, interval excluding zero); the two are not interchangeable (Sections 3, 5, 6).
4. **The published treatments of overlong outputs act on different terms, and the decomposition says which.** Substituting each recipe's reward rule and loss mask into the identity shows that an explicit zero, an installed constant, a soft length penalty, overlong masking and budget growth each remove, rescale or leave intact a different term. The masking residual $-\tau(a - b)\nabla\tau$ grows with how far the untreated policy has moved toward the cap; the one cell in which masking helped is the one whose untreated arm collapsed, and the published disagreement over masking sits on the same axis (Section 2; Appendix S2.2).

## 2 Related work

**RLVR, length and overlong handling.** Group-relative policy optimisation (Shao et al., 2024) and its descendants (Guo et al., 2025; Hu et al., 2025; Zeng et al., 2025; Lambert et al., 2024; Zhao et al., 2025) train under finite generation budgets. Liu et al. (2025c) analyse normalization-induced length and difficulty biases; we use their unnormalised advantages. Other work controls length explicitly (Kimi Team et al., 2025; Aggarwal & Welleck, 2025; Hou et al., 2025; Yeo et al., 2025). Our question is what credit already remains when the cap stops an output that the policy did not finish.

Yu et al. (2025) combine overlong masking with soft punishment; He et al. (2025) report no comparable masking benefit, and Luo et al. (2025) grow the budget during training. Liu et al. (2025d) report benefits at shorter budgets and degradation at a larger one. Liu et al. (2025b) explicitly assign zero credit at truncation. Read through the decomposition of Section 3, these are not variants of one treatment: a constant reward (including zero) removes the within-truncation credit $\mathbb{E}[T(r - b)z]$ and fixes $a - b$; a soft penalty keeps that credit and adds a length-graded term; masking with a retained group mean drops truncated tokens from the loss but keeps a residual $-\tau(a - b)\nabla\tau$; budget growth changes the truncated population, not the rule. The masking disagreement between Yu et al. (2025) and He et al. (2025) is consistent with that residual's dependence on how far the untreated policy has moved toward the cap; we organise the comparison rather than reanalyse their data (Table 1; substitutions in Appendix S2.2).

**Budget-dependent measurement.** Wachi et al. (2026) study a relative-budget ratio, Souza & Panisson (2026) show model rankings reversing across budgets, and Wu et al. (2025) identify budget

mismatch as an evaluation confound. Test-time computation can improve answers (Wei et al., 2022; Wang et al., 2023; Muennighoff et al., 2025; Snell et al., 2024; Liu et al., 2025a; Ghosal et al., 2025). Our frozen-policy scans measure how the budget also changes prompt-level difficulty, which curriculum methods consume (Bengio et al., 2009; Graves et al., 2017; Matiisen et al., 2017; Jiang et al., 2021). Ranking changes are compatible with heterogeneous test-time scaling, not a refutation of it.

**Time limits and terminal credit.** Pardo et al. (2018) distinguish task horizons from training-only truncation; bootstrapping is appropriate to the latter. Terminal costs in dynamic programming (Bertsekas, 2017) and salvage payoffs in control (Kamien & Schwartz, 1991) motivate our terminology, not a prescribed value for RLVR. Likewise, censored-data analysis (Klein & Moeschberger, 2005; Amemiya, 1984) distinguishes the selected observed population from a counterfactual continuation. We study the reward actually delivered at the cap, and the separation between accepted answers and normal completion that it can conceal.

## 3 The Budget, Written Down

A prompt is rolled out $K$ times under policy $\pi$ with generation capped at $B$ tokens. Let $s_i \in \{\mathrm{C}, \mathrm{T}\}$ record completion or truncation and $r_i$ the delivered reward. For fixed $(\pi, x, B)$ define

$$\tau_B = \Pr(s = \mathrm{T}), \qquad a_B = \mathbb{E}[r \mid s = \mathrm{C}], \qquad b_B = \mathbb{E}[r \mid s = \mathrm{T}]. \tag{2}$$

For binary rewards these conditional means are acceptance probabilities. Our runs use Dr. GRPO-style $A_i = r_i - \bar{r}$, without division by the group standard deviation (Liu et al., 2025c); Appendix S2.1 gives the derivations.

**Proposition 1 (What the pass rate measures)** *The mean reward (pass rate for a binary verifier) satisfies, exactly,*

$$q_B = (1 - \tau_B)a_B + \tau_B b_B = a_B - (a_B - b_B)\tau_B. \tag{3}$$

*It combines rewards on completed and truncated subpopulations, both selected by the budget. Neither conditional mean is a property of the prompt alone.*

**A terminal payoff, not a continuation value.** The salvage value is credit paid when generation stops at the cap, not the expected reward of continuing the prefix (Section 2). Zero credit at truncation can be the right task specification; a parser without a truncation flag cannot enforce it on its own.

**Population, empirical and installed values.** $b_B$ in Eq. (2) is the population conditional mean; $b^\star$ in Eq. (1) is its empirical reading. At the same checkpoint, budget, verifier and prompt distribution they describe the same quantity, not distinct intrinsic and implied values. Both can change during training. In an installed arm every truncated rollout instead receives $b^{\mathrm{set}}$, so $b^\star = b^{\mathrm{set}}$ whenever a truncated rollout is observed. A bare $b$ denotes the salvage value in the current context.

**From status to advantage.** For the population-centred advantage $A_i^\circ = r_i - q$, completion status predicts reward:

$$A_i^{\mathrm{st}} := \mathbb{E}[A_i^\circ \mid s_i] = \begin{cases} \tau(a - b), & s_i = \mathrm{C}, \\ -(1 - \tau)(a - b), & s_i = \mathrm{T}, \end{cases} \qquad A_i^\circ = A_i^{\mathrm{st}} + A_i^{\mathrm{res}}. \tag{4}$$

With the same-group sample mean and conditionally iid rollouts, $\mathbb{E}[A_i \mid s_i] = (K - 1)A_i^{\mathrm{st}}/K$; centring does not erase the contrast.

**Proposition 2 (The status component of the update)** *The population-centred weights have a unique $L^2$ projection onto status. Their status-induced variance is the second term in*

$$\mathrm{Var}(r) = \underbrace{\mathbb{E}[\mathrm{Var}(r \mid s)]}_{\textit{within-status variation}} + \underbrace{\tau(1 - \tau)(a - b)^2}_{\textit{status-induced}}. \tag{5}$$

*Writing $\hat{g} = K^{-1} \sum_i A_i^\circ \nabla_\theta \log \pi_\theta(y_i \mid x)$ gives a corresponding status/residual split. The status weights vanish at $\tau = 0, 1$ and at $b = a$; the reward endpoints $b = 0, 1$ alone do not imply vanishing. The gradient components need not be orthogonal.*

This is a variance of weights, not a gradient norm or a stability theorem. For fixed $a - b$, it peaks at $\tau = 1/2$ and vanishes at the budget extremes (Figure S1). With installed rewards and a fixed sampled group, $\partial A_i/\partial b = \mathbf{1}\{s_i = \mathrm{T}\} - \hat{\tau}_g$: only a common shift is removed (Appendix S2.1).

Masking and reward replacement are distinct interventions. Recentring on completed members removes the explicit completion-status contrast; masking tokens while retaining the original group mean does not. Our implementation uses the latter, so its empirical effect cannot be identified with setting $b = a$ (Appendix S2.1).

**Hypothesis 3 (Switch)** *Weakening completion-related credit can increase length when other stopping incentives are weak; its effect depends on the policy and on which baseline and loss mask are used.*

**The censoring interface, and what changes when nobody installs $b$.** For installed $b \in [0, 1]$, the reward encoding $\phi_b(W) = 0$, $\phi_b(T) = b$, $\phi_b(C) = 1$ distinguishes wrong, truncated and correct outcomes iff $0 < b < 1$. The midpoint maximises the smaller adjacent gap, a geometric property rather than an accuracy guarantee. Here $C$ denotes completed-and-correct, unlike the completion-status label above. The objective $J_b(\theta) = P_\theta(C) + bP_\theta(T)$ has sensitivity $\partial J_b/\partial b = \tau$ with the policy fixed. Under a fixed prefix-reading grader, $b(\theta) = \mathbb{E}_\theta[r \mid T]$ instead changes with the sampled prefixes:

$$\nabla J = \nabla P_\theta(C) + b(\theta)\nabla P_\theta(T) + P_\theta(T)\nabla b(\theta). \tag{6}$$

Let $z = \nabla_\theta \log \pi_\theta(y \mid x)$ (pseudocode: Algorithms 1–2). The last term equals $\mathbb{E}[\mathbf{1}_T(r - b)z]$: reward-dependent credit *within* truncated rollouts, not backpropagation through the grader. Installing a constant removes this within-truncation credit, but need not eliminate length growth or improve accuracy. A preregistered matched fork tests it: from one checkpoint, one arm keeps the default grader and one pins $b$ at that checkpoint's own $b^\star$, removing within-truncation reward variation. The default-grader arm truncates more in four of six seeds, and the two seeds in which within-truncation reward carries more than a quarter of batch reward variance show the two largest gaps (Appendix S15.3). The prefix probes in Section 5 examine a behaviour this credit can favour: placing an accepted answer before the cap while continuing to generate.

**Where else $b$ comes from.** The inspected default reward paths in verl, TRL, OpenRLHF and slime score decoded text without using truncation status to enforce zero credit; this describes those paths, not the frameworks' capacity to expose status. The DAPO reference grader reads the last three hundred characters (Table S2, Appendix S3). The mechanism is not specific to `\boxed`, and prefix acceptance alone guarantees neither $\nabla b \neq 0$ nor collapse: the reward must covary with the policy score within the truncated population (Appendix S2.1).

## 4 The Budget Manufactures the Difficulty It Measures

Before intervening on $b$, we measure the lever arm it acts through. By Eq. (3), $\partial q_B/\partial b = \tau_B$: the salvage value matters exactly as much as the budget censors. We therefore hold everything fixed (policy, prompts, verifier, sampler, sample count), vary only $B$, and read the iteration-0 evaluation so that no gradient step is taken. The main ladder uses Qwen3-4B-Instruct-2507 on 750 mathematical development prompts, 8 samples per prompt (hard stratum 75 prompts / 600 rollouts, defined once and reused throughout).

**The difficulty coordinate moves by $44.7$ pp, and the ranking moves with it.** Hard-stratum pass rate goes from $0.011$ to $0.458$ (Figure S2; Appendices S4–S6 give every cap, stratum and paired bootstrap interval) while the robust centre of the length distribution does not move. Completion rises $12.9\times$ and conditional correctness $3.7\times$, but the latter cannot be read as improved ability, because $a$ is defined on the subpopulation that finished, which grows from $6.9\%$ to $88.9\%$ of the group. Nor is it only the level: ranking the 750 prompts by pass rate at a 2k budget and again at 12k gives a rank correlation of .596, with 4–8% of prompts crossing $p = .5$ at each step. The coordinate is reordered, not rescaled, so *which* prompts are called hard is a function of $B$. This reveals heterogeneous budget sensitivity, not just a uniform level shift; it need not exclude test-time scaling. Souza & Panisson (2026) establish the same instability on the model axis; our unit is the prompt, which is what feeds a curriculum, so the instrument is inside the training loop and not only inside the report.

**The effect tracks released censoring, and it has a negative control.** Across five model variants and seven (policy, prompt-set) cells (Table S3, Appendix S7), the accuracy change tracks released censoring (Spearman $\rho = 0.79$, $n = 7$); this is not a fitted proportional law. Splitting at $\Delta\tau = 0.05$ separates the cells cleanly: five move hard-stratum accuracy by $0.18$–$0.45$, two are flat, and the two flat cells sit at *opposite* ends, one where the cap never binds (Llama-3.2-3B-Instruct, $\tau$ .060 $\rightarrow$ .042) and one where it never *stops* binding (Qwen3.5-4B-Instruct, $\tau$ .347 $\rightarrow$ .330). What looked like a property of the model is a property of where the swept range sits on the length distribution. The preregistered negative control makes the same point from the other side: on GSM8K truncation is pinned at $0.016$ and $0.007$ and accuracy moves by $+0.002$, bootstrap interval $[-0.003, +0.007]$ on $1319$ prompts, a detected zero. Three budget cells are re-runs of a lost early sweep; one misses its preregistered $\pm 2$ pp tolerance, at $2.33$ pp (Appendix S4). The same holds at 27B: a zero-training Qwen3.5-27B dense pilot on $96$ stratified prompts accepts $19$ of $59$ truncated responses at cap $4096$ ($b^\star = 0.322$) and $15$ of $39$ at $12288$ ($0.385$), so nonzero truncation credit is not a small-model artefact (Appendix S15).

**The same error inside a single policy.** Because the evaluation budget is bound to the training budget by default, two arms of an experiment are routinely compared through two different instruments. On one checkpoint, arm A scores $0.162$ on the hard stratum at $B = 4096$ and $0.440$ at $B = 12288$: one policy, $+27.8$ pp, entirely from the readout. Every comparison we report is therefore scored at one matched budget (Appendix S17).

## 5 When Answer Acceptance and Completion Separate

**Three default-grader cases: two terminal, one transient.** The default grader installs no constant reward for truncation, and in three runs the policy learned to be accepted without finishing. In the cap-$12288$ run, the recovered value reaches $0.725$ at training rollout $5$ while only $0.089$ of outputs truncate; it ends at $b^\star = 0.933$ and $\tau = 0.989$. At cap $4096$ it reaches $0.602$ at rollout $11$, when $\tau = 0.248$, and ends at $b^\star = 0.885$, $\tau = 1.000$. In E12's final training batch, all $848$ easy-stratum responses are accepted but only $11$ complete: acceptance is $100\%$ and strict accuracy $1.3\%$ on *the same samples*. The third case is a Qwen3.5-4B-Base run at cap $1024$ on the current launcher (seed $47$; Figure 1, Table S5). Between rollouts $7$ and $8$, training truncation jumps from $0.101$ to $0.984$, $b^\star$ from $0.000$ to $0.612$, and completed-and-correct mass from $0.696$ to $0.015$; of $912$ truncated easy-stratum responses, $604$ are accepted. Held-out strict accuracy at that checkpoint is $0.010$ with $0.987$ of responses truncated. The run recovers within two updates to $0.662$. Two arms launched from the same seed on the same launcher, differing only in how truncated rollouts are handled, do not show this excursion: with truncated tokens masked, or with the truncated reward pinned at $0.88$, truncation stays in $0.17$–$0.40$ across the same updates. *Drift* here means change in a conditional reward over training, not crossing a universal threshold; the cases establish that the separation occurs, in terminal and transient forms, not its frequency.

**Reading the generated outputs.** We reloaded six Qwen3-4B-Instruct seeds at cap $12288$ and sampled from one fixed prompt set; seeds $47$ and $56$ were read at two checkpoints. Between checkpoints $7$ and $15$, the accepted share of truncated outputs rises from $0.00$ to $0.36$ and from $0.59$ to $0.79$, respectively (Table 2). Their first boxed answers move from $0.72$ to $0.07$ and from $0.74$ to $0.64$ of response characters. The five available checkpoint-$15$ probe/ledger pairs have Spearman $\rho = 0.80$ across different prompt samples, an agreement check between two measurements of the same conditional reward.

At the collapsed $4096$ checkpoint all $1024$ probe responses truncate: $990$ contain a boxed answer and $0.92$ are accepted. At the $12288$ checkpoint, $950$ of $1024$ truncate, all $950$ contain a boxed answer and $0.93$ are accepted. Median first-box positions are $0.24$ and $0.23$ of response characters. The model has often produced an answer before the budget stops it. These are response positions, not token-cost estimates.

Other seeds show that this behaviour is not a sufficient failure condition: seed $52$ has first-box position $0.49$ and ledger $b^\star = 0.65$, yet ends at strict accuracy $0.916$. The remaining single-checkpoint positions are $0.76$, $0.85$ and $0.81$. Stratum-wise traces show both easy responses entering the truncated set and changes within strata (Appendix S8). The output behaviour is observed directly; its causal attribution remains distinct from the conditional-reward identity.

| seed | ckpt | $\tau$ | trunc. $n$ | has \boxed | accepted | ledger $b^\star$ | 1st box at | mean $L$ |
|---|---|---|---|---|---|---|---|---|
| 56 | 7 | 0.028 | 29 | 28/29 | 0.59 | 0.05 | 0.74 | 2823 |
| 56 | 15 | 0.204 | 209 | 205/209 | 0.79 | 0.57 | 0.64 | 5802 |
| 47 | 7 | 0.012 | 12 | 11/12 | 0.00 | – | 0.72 | 1905 |
| 47 | 15 | 0.357 | 366 | 366/366 | 0.36 | 0.22 | 0.07 | 5744 |
| 45 | 15 | 0.037 | 38 | 28/38 | 0.08 | 0.17 | 0.76 | 2126 |
| 46 | 15 | 0.077 | 79 | 44/79 | 0.27 | 0.35 | 0.85 | 3075 |
| 52 | 15 | 0.021 | 22 | 22/22 | 0.59 | 0.65 | 0.49 | 1801 |
| 53 | 15 | 0.014 | 14 | 8/14 | 0.00 | – | 0.81 | 2930 |
| 43† | 15 | 1.000 | 1024 | 990/1024 | 0.92 | 0.89 | 0.24 | 4096 |
| 42† | 7 | 0.928 | 950 | 950/950 | 0.93 | 0.88 | 0.23 | 11438 |

Table 2: **Answers can appear well before forced termination.** Each checkpoint produces 1024 new responses on a fixed prompt set. "Accepted" is the fraction of truncated responses receiving reward 1; "1st box at" is the median first-box character position divided by response length, among responses with a box. Completed-response medians span 0.87–0.98. Ledger readings come from training batches at the listed rollout indices, not these new responses; values with training $\tau < 0.02$ are suppressed. †The 4096 case is reloaded at checkpoint 15; the 12288 case at checkpoint 7, before its final training batch at index 10.

**Installing a reward instead of leaving it to the parser.** The fixed-reward sweep holds the cap at 12288 and compares $b \in \{0, 0.25, 0.5, 0.75, 1\}$ on Qwen3-4B-Instruct; a second seed is available at 0.5 (Figure 2). The sampled 0.25, 0.5 and 0.75 runs did not saturate. At $b = 1$, both this family and the Qwen3.5-4B-Base cap-1024 family reached complete truncation and zero strict accuracy. This is consistent with the objective making a never-terminating policy reward-optimal, without implying that optimization must select it.

Zero reward is not a demonstrated failure point: the two completed cap-12288, installed-$b = 0$ runs have held-out strict accuracies 0.9255 and 0.9145. We do not resolve an ordering among interior values (Figure S5, Appendix S11). The contrast supported here is between sampled reward implementations, not a universally optimal midpoint.

**A fixed high reward is not safe.** Five Qwen3.5-4B-Base seeds (43–47) at cap 1024 install $b = 0.88$ against a same-seed default arm. After ten rollouts, held-out strict accuracy falls in every seed, by 35.7, 13.2, 22.3, 40.9 and 11.8 pp (mean $-24.8$ pp, 95% $t$ interval $[-41.1, -8.5]$): 0.457 against 0.704. Hard-stratum truncation reaches 0.92–1.00 (mean 0.960) against 0.697. The installed value is recovered at every update; it removes within-truncation variation but truncation now pays close to a correct completion, and the policy learns to be truncated. The dose is fixed from seed 43's historical mean and held across seeds, not an adaptive $b = a$ (Appendix S15).

**Across seeds and controls.** The later default-grader cohort at cap 12288 contains 14 runs with 13 distinct seeds; none saturates, and its strict-accuracy distribution has sample sd 0.102, against 0.015 in the 9-seed fixed-0.5 replication cohort (Table S8, which also discusses what drives the spread). Truncation can rise and then reverse under a fixed reward (Table S6, Appendix S12), whereas the default cases accelerate toward the cap; count-matched placebos that also avert collapse recover less hard-stratum quality (Appendix S13). The fixed-reward construction removes within-truncation reward variation exactly; its effect on trained quality is the empirical question the rest of this section and Section 6 address.

## 6 Changing Truncation Treatment: A Paired Intervention

The installed-reward sweep changes $b$ directly. Here we study the field's other handle: masking truncated tokens while retaining their rewards in the group mean. This changes which trajectories receive updates, not simply the coefficient $b$ (Section 3; Appendix S2.1).

**The design.** Twelve cells, one per (policy, budget, seed): Qwen3-4B-Instruct at $B = 4096$ (untreated arm run for 16 updates, treated for 10, read at the matched update 7); Qwen3.5-4B-Base at 2048, 4096 and 8192, the 4096 cell being one trajectory of $10 + 14$ updates continued from its

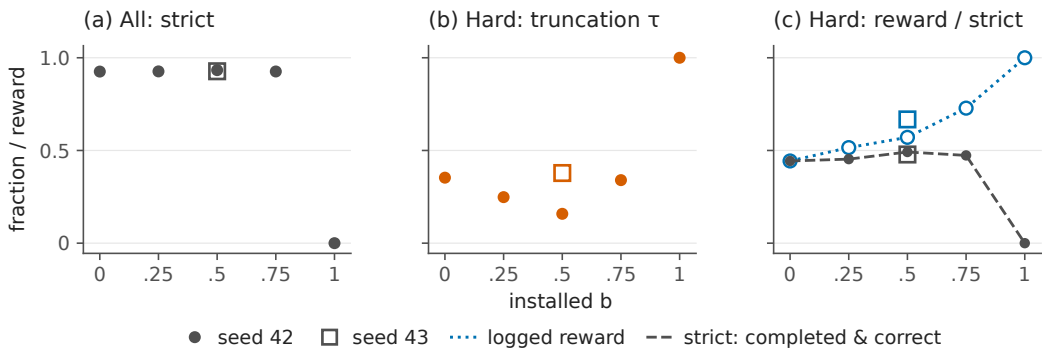



Figure 2: **Changing truncation credit changes both the objective and its reported score.** Installed-reward sweep at cap 12288, final evaluation after 16 training rollouts. Circles are seed 42; open squares are the additional seed 43 at $b = 0.5$. Left: strict accuracy on the full evaluation set. Centre: truncation on the hard stratum. Right: hard-stratum mean reward and completed-and-correct mass; these coincide at $b = 0$ but diverge as truncated outputs receive credit. No confidence band or fitted interior optimum is implied. Default-grader cases are shown separately in Figure 1, not treated as installed doses. Full values are in Table S9.

own checkpoints, and at 1024 with six seeds (42–47); Qwen3.5-4B-Instruct at 8192; R1-Distill-Qwen-7B at 12288. Within a pair the only deliberate difference is whether truncated rollouts enter the update: in all twelve cells the launch manifests of the two members agree bit-for-bit on the hash of the launcher, verifier, rollout monitor and training data. Treatment delivery is checked from the ledger: `removed_sample_count` is identically zero at every update in every untreated arm and non-zero at every update in every treated arm. Two further pairs, Qwen3.5-4B-Instruct and Llama-3.2-3B at 4096, are reported separately because full-set losses coexist with flat hard-stratum scores (Appendix S26). This twelve-cell cohort therefore does not estimate the effect over every run we conducted. The wide-budget Qwen3-4B cases, including M42/M43 and the separately collected default cohort, remain in Appendices S17 and S11; sharing a seed and cap alone does not make their endpoints an additional matched cell.

**Result: masking costs accuracy in nine of twelve cells.** Held-out strict accuracy falls in nine of twelve cells (Table 3), by 5.3 pp on average (twelve-cell 95% $t$ interval $[-8.9, -1.6]$ pp). Hard-stratum truncation rises in eight of twelve cells, by 10.5 pp on average (policy-equal weighting in Appendix S16). The one material exception is Qwen3-4B-Instruct at 4096: masking raised strict accuracy by 0.9 pp while cutting hard-stratum truncation by 40 points, the only cell where both signs reverse and the highest untreated truncation rate (0.745); two cap-1024 seeds gain under 2 pp with truncation essentially unchanged. The twelve span four policies, one at 7B, and budgets from 1024 to 12288. The treated arm masks tokens rather than assigning a new truncated reward; since truncated rewards remain in the group mean, surviving trajectories keep a baseline-dependent status term (Appendix S2.1), so this is an effect of masking as implemented.

**The cost is not an offset: it accumulates, and it starts at zero.** Both members of a pair were evaluated at the *same* update count throughout, so the difference is a trajectory (Figure S7, Appendix S18). In the six cells with a reading taken before any treated update, **six of six start at zero** (within 0.4 pp), so no baseline imbalance separates treated from untreated; and **the cost grows in five of six**, the final matched difference being worse than the first by 1.7 to 16.6 pp, while the Qwen3-4B-Instruct cell stays within a point of zero. A sign test on five of six does not reach significance ($p = 0.22$), so the accumulation is a shape we report, not a result we test; it does not by itself distinguish a deleted termination gradient from any other change to the effective update.

**What this does and does not settle.** It establishes an effect of the specified loss mask, in the regime where the untreated arm completed: every cell conditions on that, and where the untreated arm collapsed the same treatment rescued it (Appendix S17). It does not settle an optimal value, and no dose-response in exposure is detected ($t = 0.62$, $\rho = 0.23$ over twelve cells; Appendix S18),

| policy | $B$ | $\Delta$ strict | $\Delta$ hard $\tau$ | untreated hard $\tau$ |
|---|---|---|---|---|
| Qwen3.5-4B-Base[†] | 4096 | $-16.9$ | $+29.5$ | 0.545 |
| Qwen3.5-4B-Base | 2048 | $-11.7$ | $+47.5$ | 0.283 |
| Qwen3.5-4B-Base | 8192 | $-4.9$ | $+14.0$ | 0.412 |
| Qwen3.5-4B-Base (s42)[‡] | 1024 | $-5.7$ | $+36.7$ | 0.423 |
| Qwen3.5-4B-Base (s43) | 1024 | $+1.9$ | $-0.5$ | 0.717 |
| Qwen3.5-4B-Base (s44) | 1024 | $+1.5$ | $-1.5$ | 0.673 |
| Qwen3.5-4B-Base (s45) | 1024 | $-4.8$ | $-2.8$ | 0.725 |
| Qwen3.5-4B-Base (s46) | 1024 | $-7.1$ | $+2.2$ | 0.723 |
| Qwen3.5-4B-Base (s47) | 1024 | $-10.9$ | $+25.0$ | 0.652 |
| Qwen3.5-4B-Instruct | 8192 | $-2.0$ | $+7.7$ | 0.197 |
| R1-Distill-Qwen-7B | 12288 | $-3.4$ | $+9.2$ | 0.258 |
| Qwen3-4B-Instruct | 4096 | $+0.9$ | $-40.5$ | 0.745 |
| mean | | $-5.3$ | $+10.5$ | |
| direction | | $9/12$ | $8/12$ | |

Table 3: Treated minus untreated on the held-out evaluation set, in percentage points, at each arm's own budget. Treated arms mask truncated tokens but retain their rewards in the group mean; untreated arms apply no loss mask. Cell-level paired 95% CI $[-8.9, -1.6]$ pp. [†]One trajectory read at its end: a ten-update pair continued for fourteen more from its own checkpoints. The six cap-1024 cells are every same-seed (default, mask) pair we ran at that cap. [‡]The seed-42 pair predates the other five by one launcher revision (identical within the pair) and starts from a lower untreated truncation rate; seeds 43–47 share one revision and one evaluation schedule.

so it does not adjudicate the published disagreement over masking beyond the reporting point of Section 7.

## 7 Implications for Training and Evaluation

**Separate accepting an answer from finishing a response.** The case runs expose a failure of a combined success label: an answer can be accepted while the policy continues until forced termination. A task that only needs an extractable answer may count this as success; one that requires completed responses or controls cost cannot. Report verifier acceptance, completed-and-correct output and length separately; the salvage value locates where accepted credit remains when they separate.

**Specify the reward and the update, not just "overlong filtering".** Record the cap, truncation rate and conditional truncated reward, with the loss mask and its baseline samples. Constant replacement is an explicit control of reward variation within truncated outputs, not an accuracy improvement, and masking is not its substitute. These operational distinctions are the reusable contribution; the differing filtering results of Yu et al. (2025); Liu et al. (2025d) motivate controlled comparisons rather than a shared curve (protocol and budget-allocation analysis in Appendices S23, S20).

## 8 Limitations

Training uses 3–7B policies on mathematics with one stack and short windows. Three default-grader collapses (two terminal on earlier launcher revisions, one transient on the current one) establish occurrence, not frequency or a threshold. Probes and stratum traces constrain, but do not eliminate, selection effects. The masking estimate covers twelve configurations with three opposite-sign cells. We do not establish an optimal fixed reward or transfer beyond mathematics (Appendix S30 expands on power and scope).

**Conclusion.** Truncation need not erase the credit a verifier finds in a response. That salvage value is measurable from consistent logs, moves under the default grader, and is a distinct credit term whose removal, in a preregistered matched fork, lowered end-of-window truncation in four of six seeds. Accepted answers can coexist with failure to finish; make reward assignment and termination explicit, and evaluate both.

## Reproducibility Statement

Every number in this paper is recomputed from per-rollout ledgers retained on disk, without rerunning training, and the central instrument, Eq. (1), needs only three scalars that standard pipelines already log. Appendices S25 and S29 give the experimental protocol and the resolved launch configuration and the manifest hashes that confirm each sweep varies one scalar; Appendix S26 lists every completed arm, including those that do not support our claims; Appendix S28 lists failed and superseded runs; Appendix S27 pairs each assertion with its evidence. Preregistered predictions are reported whether or not they held. The captured ledgers and the scripts that recompute Eq. (1), the stratum decomposition, the recipe-to-term map and the endpoint tables from them are included as anonymised supplementary material; training code and launch scripts will be released with the camera-ready version.

## Ethics Statement

This work trains and evaluates language models on public mathematical benchmarks, with no human subjects, no personal data and no released weights. Its practical content is a reporting request. We are not aware of a dual-use concern specific to this work.

## AI Use Statement

Large language model assistants were used for two tasks: writing and debugging experiment and analysis code under author specification, and checking the manuscript for grammar and clarity. The research questions, conceptual framework, mathematical claims, experimental design and interpretation of results are the authors' own. No data in this paper is AI-generated. Every mathematical claim was derived by the authors; every quantitative claim was recomputed by the authors from the retained ledgers, with the two headline metrics computed by independent code paths (Appendix S24); and every bibliography entry was checked against its live record. The authors take full responsibility for the content of this work.

## References

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025. URL https://arxiv.org/abs/2503.04697.

Takeshi Amemiya. Tobit models: A survey. *Journal of Econometrics*, 24(1–2):3–61, 1984.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2009.

Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. Athena Scientific, 4 edition, 2017.

Soumya Suvra Ghosal, Souradip Chakraborty, Avinash Reddy, Yifu Lu, Mengdi Wang, Dinesh Manocha, Furong Huang, Mohammad Ghavamzadeh, and Amrit Singh Bedi. Does thinking more always help? mirage of test-time scaling in reasoning models. *arXiv preprint arXiv:2506.04210*, 2025. URL https://arxiv.org/abs/2506.04210.

Alex Graves, Marc G. Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *International Conference on Machine Learning (ICML)*, 2017.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, et al. Skywork open reasoner 1 technical report. *arXiv preprint arXiv:2505.22312*, 2025. URL https://arxiv.org/abs/2505.22312.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. Thinkprune: Pruning long chain-of-thought of LLMs via reinforcement learning. *arXiv preprint arXiv:2504.01296*, 2025. URL https://arxiv.org/abs/2504.01296.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025. URL https://arxiv.org/abs/2503.24290.

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4940–4950, 2021. URL https://proceedings.mlr.press/v139/jiang21b.html.

Morton I. Kamien and Nancy L. Schwartz. *Dynamic Optimization: The Calculus of Variations and Optimal Control in Economics and Management*. North-Holland, 2 edition, 1991.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1.5: Scaling reinforcement learning with LLMs. *arXiv preprint arXiv:2501.12599*, 2025. URL https://arxiv.org/abs/2501.12599.

John P. Klein and Melvin L. Moeschberger. *Survival Analysis: Techniques for Censored and Truncated Data*. Springer, 2 edition, 2005.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024. URL https://arxiv.org/abs/2411.15124.

Runze Liu, Junqi Gao, Jian Zhao, Kaiyan Zhang, Xiu Li, Biqing Qi, Wanli Ouyang, and Bowen Zhou. Can 1b LLM surpass 405b LLM? rethinking compute-optimal test-time scaling. *arXiv preprint arXiv:2502.06703*, 2025a. URL https://arxiv.org/abs/2502.06703.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Yejin Choi, et al. DLER: Doing length pEnalty Right — incentivizing more intelligence per token via reinforcement learning. *arXiv preprint arXiv:2510.15110*, 2025b. URL https://arxiv.org/abs/2510.15110.

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. *arXiv preprint arXiv:2503.20783*, 2025c. URL https://arxiv.org/abs/2503.20783.

Zihan Liu, Zhuolin Yang, Yang Chen, Chankyu Lee, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. AceReason-Nemotron 1.1: Advancing math and code reasoning through SFT and RL synergy. *arXiv preprint arXiv:2506.13284*, 2025d. URL https://arxiv.org/abs/2506.13284.

Michael Luo, Justin Tan, Colin Wong, Xiaoxiang Shi, William Y. Tang, et al. DeepScaleR: Surpassing O1-Preview with a 1.5b model by scaling RL. Project page and model card, 2025. URL https://huggingface.co/agentica-org/DeepScaleR-1.5B-Preview.

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher-Student curriculum learning. *arXiv preprint arXiv:1707.00183*, 2017. URL https://arxiv.org/abs/1707.00183.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025. URL https://arxiv.org/abs/2501.19393.

Fabio Pardo, Arash Tavakoli, Vitaly Levdik, and Petar Kormushev. Time limits in reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *PMLR*, pp. 4042–4051, 2018. URL https://arxiv.org/abs/1712.00378.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

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. *arXiv preprint arXiv:2409.19256*, 2024. URL https://arxiv.org/abs/2409.19256.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024. URL https://arxiv.org/abs/2408.03314.

Rodrigo Guedes de Souza and Alison R. Panisson. Who thinks best depends on how long you let them: Budget-dependent rankings in LLM evaluation. *arXiv preprint arXiv:2608.12150*, 2026. URL https://arxiv.org/abs/2608.12150.

Akifumi Wachi, Hirota Kinoshita, Shokichi Takakura, Rei Higuchi, and Taiji Suzuki. A relative-budget theory for reinforcement learning with verifiable rewards in LLM reasoning. *arXiv preprint arXiv:2602.01523*, 2026. URL https://arxiv.org/abs/2602.01523.

Xu Wan, Speed Zhu, Jianwei Cai, Guang Chen, XiMing Huang, Wiggin Zhou, and Mingyang Sun. The shadow price of reasoning: Economic perspective on optimal budget allocation for LLMs. *arXiv preprint arXiv:2606.03092*, 2026. URL https://arxiv.org/abs/2606.03092.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Fang Wu, Aaron Tu, Weihao Xuan, Heli Qi, Xu Huang, Qingcheng Zeng, Shayan Talaei, Yijia Xiao, Peng Xia, Xiangru Tang, et al. Position: The hidden costs and measurement gaps of reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2509.21882*, 2025. URL https://arxiv.org/abs/2509.21882.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. Demystifying long chain-of-thought reasoning in LLMs. *arXiv preprint arXiv:2502.03373*, 2025. URL https://arxiv.org/abs/2502.03373.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025. URL https://arxiv.org/abs/2503.14476.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. SimpleRl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025. URL https://arxiv.org/abs/2503.18892.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, et al. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*, 2025. URL https://arxiv.org/abs/2505.03335.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, et al. SGLang: Efficient execution of structured language model programs. *arXiv preprint arXiv:2312.07104*, 2023. URL https://arxiv.org/abs/2312.07104.

# S1 NOTATION

| symbol | meaning |
|---|---|
| $B$ | response budget (generation cap), in tokens |
| $K$ | rollouts per prompt during training (16; evaluation uses 8) |
| $s \in \{\mathrm{C}, \mathrm{T}\}$ | completed / truncated at $B$ |
| $\tau_B$ | $\Pr(s = \mathrm{T})$; the truncation rate at budget $B$ |
| $a_B$ | $\Pr(r = 1 \mid s = \mathrm{C})$; correctness given completion |
| $b$ | the score the reward assigns to a truncated rollout; a design variable |
| $b^\star$ | $b$ recovered from a log, $(\bar{r} - \bar{r}_{\mathrm{strict}})/\tau$ (Sec. 3) |
| $b_B$ | $\Pr(r = 1 \mid s = \mathrm{T})$; correctness among cut-off rollouts |
| $q_B$ | observed pass rate, $= (1 - \tau_B)a_B + \tau_B b_B$ |
| $A_i$ | group-relative advantage, $r_i - \bar{r}$ |
| $\tau_0$ | truncation rate at the first rollout, before any training |

# S2 PROOFS

Throughout the appendix $b$ denotes the salvage value of a truncated rollout and $b^\star$ the implied salvage value recovered by Eq. (1); the symbol is used in preference to the phrase only for compactness in tables and proofs.

**Proposition 1.** By the law of total expectation and Eq. (2), $\mathbb{E}[r] = \Pr(s = \mathrm{C})\,\mathbb{E}[r \mid s = \mathrm{C}] + \Pr(s = \mathrm{T})\,\mathbb{E}[r \mid s = \mathrm{T}] = (1 - \tau)a + \tau b$; rearranging, $(1 - \tau)a + \tau b = a - (a - b)\tau$. □

The conditional mean is defined separately for each policy, budget, checkpoint and prompt distribution. Its empirical reading $b^\star$ under a default grader can drift (Table S4); readings on different scopes are not different intrinsic and implied properties of the same population.

**Proposition 2.** The law of total variance gives $\mathrm{Var}(r) = \mathbb{E}[\mathrm{Var}(r \mid s)] + \mathrm{Var}(\mathbb{E}[r \mid s])$. The conditional mean is a two-point random variable equal to $a$ with probability $1 - \tau$ and $b$ with probability $\tau$, whose variance is $\tau(1 - \tau)(a - b)^2$. Since $A = r - \bar{r}$ with $\bar{r} = \mathbb{E}[r]$, $\mathbb{E}[A^2] = \mathrm{Var}(r)$. □

With an in-group sample mean and $K$ iid rollouts conditional on the prompt, $\mathbb{E}[A_i^2] = (K - 1)\mathrm{Var}(r)/K$ and $\mathbb{E}[A_i \mid s_i] = (K - 1)(\mathbb{E}[r_i \mid s_i] - q)/K$. The status variance of these conditional sample-centred weights is therefore $((K - 1)/K)^2\tau(1 - \tau)(a - b)^2$. These are weight moments, not gradient norms: multiplication by different policy scores need not preserve the orthogonality of the reward projection. Loss masking alone does not recenter the rewards.

## S2.1 WHAT GRPO REMOVES, AND WHAT TRUNCATION CREDIT PRESERVES

All expectations below condition on a fixed prompt and budget. The score-function statements concern an on-policy sequence-level update without clipping, KL terms or length-dependent loss weighting. They are not claims that the implemented multi-step optimizer exactly ascends the expected reward.

**Selective rewards survive centring.** Let $T_i = \mathbf{1}\{s_i = \mathrm{T}\}$ and $c_i$ be completed-and-correct credit, so $r_i(b) = c_i + bT_i$. Holding the sampled trajectories fixed gives

$$A_i(b) = c_i - \bar{c} + b(T_i - \hat{\tau}_g), \qquad \frac{\partial A_i}{\partial b} = T_i - \hat{\tau}_g, \qquad \hat{\tau}_g = K^{-1}\sum_j T_j.$$

For any truncated/completed pair, the derivative of their advantage difference is one. Only groups with identical status have zero derivative for every sample. At population level a common baseline disappears because $\mathbb{E}[z] = 0$, whereas $\mathbb{E}[bTz] = b\nabla\tau$, where $z = \nabla_\theta \log \pi_\theta(y \mid x)$. Under conditionally iid sampling the same-group baseline multiplies this expected update by $(K - 1)/K$, rather than cancelling it.

**The policy-dependent term is within-truncation credit.** For a fixed reward function $r(y)$ and a fixed truncation rule, assuming differentiation under the expectation is valid and $\tau > 0$,

$$\tau \nabla b = \nabla \mathbb{E}[Tr] - b\nabla\tau = \mathbb{E}[T(r-b)z] = \tau \operatorname{Cov}(r, z \mid T).$$

There is no differentiation through a discrete grader. The score-function update can reinforce higher-reward prefixes within the truncated population. A prefix-accepting parser is not sufficient for a nonzero term: if every truncated rollout earns the same constant, the covariance is zero. Nor does a nonzero term guarantee increasing length. Fixing $b$ removes this within-truncation covariance but not the truncation-status term.

**Why the masking baseline matters.** Write $C = 1 - T$ for completion, $a = \mathbb{E}[r \mid C]$ and $q = (1-\tau)a + \tau b$. If a mask drops truncated tokens but keeps the original reward mean, its population sequence-level update is

$$g_{\text{mask}} = \mathbb{E}[C(r-q)z] = (1-\tau)\nabla a - \tau(a-b)\nabla\tau.$$

The second term remains. Recentring the completed samples at $a$, or replacing all truncated rewards with the current $a$ while treating that value as stop-gradient, gives $\mathbb{E}[C(r-a)z] = (1-\tau)\nabla a$ instead. For a finite group the exact recentering uses its own completed-sample mean; an all-truncated group needs an explicit convention. Neither a historical batch-average $a$ nor a fixed constant such as $0.88$ implements this adaptive groupwise rule. Token weighting and optimizer normalization can introduce further differences. The paired masking experiment therefore tests that implementation, not a pure intervention on the reward coefficient.

**Reward sensitivity is not instability.** For a fixed policy, $\partial J_b/\partial b = \tau$ measures the reward leverage. It does not measure gradient strength. If only a stopping logit changes, $\tau = \operatorname{sigmoid}(\ell)$ with $a, b$ fixed, then $\partial J/\partial \ell = \tau(1-\tau)(b-a)$. A positive completion gap still opposes truncation, even when it becomes small. With shared parameters the full gradient is $(1-\tau)\nabla a + (b-a)\nabla\tau + \tau\nabla b$; the two other terms can affect length. Thus $b$ crossing $0.5$, or even $b < a$, is not a model-independent stability criterion.

### S2.2 Where each published recipe acts

The three-term form $\nabla J = \nabla P(C) + b\,\nabla P(T) + P(T)\nabla b$, with $P(T)\nabla b = \mathbb{E}[T(r-b)z]$, has one term that recipes can remove exactly (the within-truncation covariance), one contrast they can rescale ($a - b$, which sets the weight $\tau(1-\tau)(a-b)^2$ of completion status) and one population they can move (the truncated set itself, through $B$). Each published treatment of overlong outputs acts on a different one of these, and reading them off is a matter of substituting the recipe's reward rule and loss mask into the expectation. `recipe_decomposition_check.py` evaluates every row below on a synthetic population with two orthogonal score directions, one that moves the stop decision and one that moves "answer early, then continue"; the columns report which term survives under which direction.

| recipe | reward rule / mask | within-$T$ credit | status contrast | truncated set |
|---|---|---|---|---|
| default prefix grader | $r = c(1-T) + pT$ | **present** | $a - b(\theta)$, drifts | fixed by $B$ |
| installed constant | $r = c(1-T) + b_{\text{set}}T$ | removed | $a - b_{\text{set}}$, fixed | fixed by $B$ |
| explicit zero (Liu et al., 2025b) | $b_{\text{set}} = 0$ | removed | $a$, maximal | fixed by $B$ |
| soft penalty (Yu et al., 2025) | $r - \lambda \max(0, L - B_0)$ | **present** | enlarged, $L$-graded | fixed by $B$ |
| masking (Yu et al., 2025) | drop $T$ from loss; keep $q$ | out of loss | residual $-\tau(a-b)\nabla\tau$ | fixed by $B$ |
| budget growth (Luo et al., 2025) | $B \to B'$ | rule unchanged | $a, b, \tau$ all move | **changes** |

Three consequences follow without any fitted quantity. First, the explicit zero of Liu et al. (2025b) and an installed interior constant are the same operation on the within-truncation term and differ only in the status contrast they leave behind; neither is a stability guarantee, and our sweep finds both $b = 1$ arms saturating and the $0.88$ arm losing completion. Second, the soft penalty of Yu et al.

(2025) does *not* remove the within-truncation credit — prefix acceptance still varies across truncated rollouts and still covaries with the policy — it adds a length-graded pressure on top of it. Third, the two masking results that disagree in the literature, a benefit in Yu et al. (2025) and none in He et al. (2025), differ on the two quantities masking's residual $-\tau(a-b)\nabla\tau$ depends on: how much of the batch truncates and how far $a$ sits from $b$ at the time of masking. Where truncation is common and the untreated policy is sliding toward the cap, masking has a large term to remove and $a-b$ to protect; where the policy already terminates, the same operation has little to act on. Our own paired cells show the same dependence (Section 6): the one comparison in which the untreated arm was collapsing is the one in which masking helped. We state this as a reading of their published configurations through the identity, not as a reanalysis of their data.

**Exact bookkeeping still has statistical and scope uncertainty.** Equation (1) exactly recovers the sample mean on truncated rollouts. For iid binary rewards its conditional standard error is approximately $\sqrt{b(1-b)/N_T}$; clustered prompts require a cluster-aware uncertainty estimate. Rounded logs incur error amplified by $1/\tau$. Furthermore, a batch-level $b = \sum_x \Pr(x \mid T) b_x$ can change through within-prompt rewards or through the mixture $\Pr(x \mid T)$. Fixed-prompt probes reduce this ambiguity but do not by themselves remove selection within the truncated set. A matched-prefix continuation intervention would test the behavioural mechanism more directly.

**The usable-group term (this is Eq. (7)).** For an idealised filter that recentres rewards on retained completions, a group of $K$ rollouts retains $\#\mathrm{C} \sim \mathrm{Binomial}(K, 1-\tau)$ members. If $\#\mathrm{C} \le 1$, its recentered advantages vanish. Retaining at least two completions is necessary, not sufficient, for a nonzero update; identical retained rewards still give zero. This is not the baseline-preserving mask used in our experiments. The probability of at least two completions is $1-\tau^K - K(1-\tau)\tau^{K-1}$, and the expected fraction of the group's samples that were censored — the material available for denoising — is $\tau$. Their product is

$$\tau \, \Pr[\#\mathrm{C} \ge 2] \;=\; \tau \left(1 - \tau^K - K(1-\tau)\tau^{K-1}\right), \tag{7}$$

which is zero at both $\tau = 0$ and $\tau = 1$ and has an interior maximum ($\tau^\star \approx 0.76$ at our training group size $K = 16$).

*Eq. (7) is a design score, not an exact expectation.* It multiplies two marginals — the censoring mass $\tau$ and the probability that a group is usable — and we use it as a proxy for how much denoising material a masking rule can act on. The exact unconditional expectation of the censored fraction retained in a usable group is

$$\mathbb{E}\left[\tfrac{K-C}{K}\,\mathbf{1}\{C \ge 2\}\right] \;=\; \tau - \tau^K - (K-1)(1-\tau)\tau^{K-1},$$

which is likewise zero at both ends with an interior maximum ($\tau^\star \approx 0.75$ at $K = 16$). Every qualitative claim we make — zero benefit at both ends, an interior optimum, and a $\tau^\star$ far above the initial truncation rate of every arm we ran ($\tau_0 \le 0.10$) — holds for either expression; only the location of the maximum moves. □

**The knob's optimum depends on group size.** Equation (7) is closed form, so $\tau^\star$ can be tabulated without running anything. This locates the optimum of the idealised proxy, not of held-out accuracy, training stability or the baseline-preserving mask used in our runs. It is not a recommendation to target that truncation rate.

Table S1: Proxy optimum: the $\tau \Pr[\#\mathrm{C} \ge 2]$ design score of Eq. (7) against group size $K$. It is a product of two marginals, not the exact retained-censored expectation derived in Sec. S2.

| $K$ | 4 | 6 | 8 | 10 | 16 | 20 | 24 |
|---|---|---|---|---|---|---|---|
| $\tau^\star$ | 0.49 | 0.58 | 0.64 | 0.68 | 0.76 | 0.79 | 0.82 |

For this proxy, larger groups retain at least two completions at higher truncation rates, so its optimum moves right. Whether masking helps requires the actual baseline rule, reward diversity and held-out outcomes, not comparison to this table alone.

**Proposition 4 (When is one shared cap optimal?)** *Fix a total token budget $C$ and a finite cap grid. For prompt $i$ let $p_i(B)$ and $k_i(B)$ be its pass rate and expected generated length at cap $B$, both*

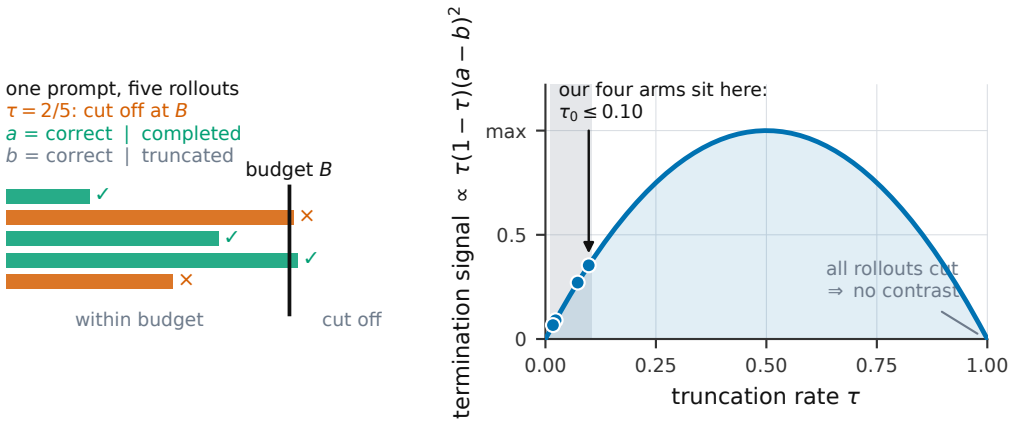



Figure S1: **One identity, two readings.** *Left*: a truncated rollout is a censored observation; the pipeline records $q_B = (1-\tau)a + \tau b$. Bar length is what the rollout wrote, colour and tick mark correctness, and one of the two rollouts that overrun the cap is correct — that is $b > 0$. *Right*: the scale $\tau(1-\tau)(a-b)^2$ against $\tau$ — zero at both ends, so the budget stops constraining termination both when it is loose and when it is saturated; the shaded band holds all four arms of the matched-budget audit ($\tau_0 \leq 0.10$).

*non-decreasing in $B$. Assume (a) every prompt has an affordable level, (b) any exchange demoting one prompt and promoting another is available, and (c) the optimum is interior, using all of $C$ with no prompt blocked at a boundary. Then a single shared cap is optimal only if the marginal returns $\Delta p_i/\Delta k_i$ coincide across prompts; whenever they differ, promoting the higher-return prompt and demoting the lower-return one strictly increases mean pass rate at non-increased total cost.*

**Proposition 4.** Let the cap grid be $B_1 < \cdots < B_m$ and write, for prompt $i$ and level $j$, $p_{ij} = p_i(B_j)$ and $k_{ij} = k_i(B_j)$, both non-decreasing in $j$. Consider the relaxation in which each prompt chooses a level subject to $\sum_i k_{i,j(i)} \leq C$. Suppose an optimum assigns level $u$ to prompt $i$ and level $v$ to prompt $i'$, with marginal returns $\rho_i = (p_{i,u+1} - p_{i,u})/(k_{i,u+1} - k_{i,u})$ and $\rho_{i'} = (p_{i',v+1} - p_{i',v})/(k_{i',v+1} - k_{i',v})$, and suppose $\rho_i > \rho_{i'}$. Promote $i$ by one level and demote $i'$ by one level. The change in total cost is $\Delta k = (k_{i,u+1} - k_{i,u}) - (k_{i',v} - k_{i',v-1})$ and the change in total accuracy is $\Delta p = (p_{i,u+1} - p_{i,u}) - (p_{i',v} - p_{i',v-1})$; whenever $\Delta k \leq 0$ the exchange is feasible and strictly improves accuracy. By scaling the exchange, a strict improvement exists whenever $\rho_i \neq \rho_{i'}$ and the grid permits it. Hence at an interior optimum all active marginal returns are equal, and a single shared cap — which imposes the same *level* but not the same marginal return — is optimal only in the degenerate case where marginal returns coincide across prompts. □

The premise is exactly what the ladder rules out: $1 - \tau_B$ is a CDF in $B$, so different prompts sit at different quantiles of it and their $\Delta p/\Delta k$ differ. The allocation problem is a multiple-choice knapsack, NP-hard in general; we solve it greedily by marginal return, which is the standard Lagrangian relaxation, and we bootstrap the entire rule so that the reported interval covers the decision procedure rather than a fixed partition.

## S3 Where truncation reward comes from

**How the inspected default scoring paths use status.** Read from the main branches on 2026-09-21 (Table S2). The question is whether the inspected scorer uses truncation status to override prefix reward, not whether the framework can expose that status through an optional argument or sample object.

Section 3 noted that the score given to a rollout that did not finish is set in two places, not one: it is the scalar $b$ on the training side, and on the verification side it is fixed by what the grader reads, since a grader that cannot see an answer still has to decide what to record. The two are the same variable. Writing the mapping down lets a practitioner locate their own configuration on the $b$ axis without running anything.

| framework | scoring function receives | truncation status |
|---|---|---|
| verl | `data_source, solution_str, ground_truth, extra_info` | not passed |
| verl, DAPO grader | `solution_str, ground_truth, strict_box_verify`; scores `solution_str[-300:]` | not passed |
| TRL | `prompts, completions, completion_ids` | computed from the last token, logged as a clipped ratio, not passed |
| OpenRLHF | `queries, prompts, labels` | not passed |
| slime | `args, sample`; default graders read `sample.response` | not read |

Table S2: **Default scoring paths do not enforce zero credit solely from truncation.** "Not passed" describes the listed signature; "not read" describes use of the sample object, which can contain additional state. This does not claim the frameworks lack status-aware extensions. TRL's optional `mask_truncated_completions` is off by default; the DAPO reference grader scores the trailing three hundred characters.

| what the grader reads | a truncated rollout scores | effective $b$ |
|---|---|---|
| requires completion before reading the answer | 0 | $b = 0$ |
| scores whatever prefix was produced | verifier acceptance | $0 \leq b \leq 1$ |
| reads format only | format match | low-signal $b$ |
| ignores content | coin flip | $b$, no signal |

The first row is an explicit zero-at-truncation rule. The second describes the inspected default prefix graders: its conditional mean is policy-dependent and can lie anywhere in $[0, 1]$. A content-independent stochastic reward is different from an installed constant: its conditional variance need not vanish. The $b = 1$ experiment installs deterministic full credit at truncation, not a coin-flip reward.

Two consequences we do not develop further. Because the read set is a design choice, two groups running "the same" RLVR recipe can be at different $b$ without either having set $b$; and reporting which read set was used, alongside $\tau$, belongs in the protocol of Section 7. We treat the verification side as a source of variation we do not manipulate and make no empirical claim about it beyond the mapping itself. In particular, we do not study grader fidelity as a quantity in its own right — whether a verifier can be satisfied without completion, how that interacts with curriculum statistics, or how to audit it without labels. Those are the subject of separate work; here the read set enters only as one of the two places the salvage value is set.

## S4 Full budget ladder, both repeats

Two independent runs per budget, Qwen3-4B-Instruct-2507, hard stratum ($n = 600$), zero training.

| $B$ | hard $q$ | | hard $1 - \tau$ | | median length (all) | |
|---|---|---|---|---|---|---|
| | run 1 | run 2 | run 1 | run 2 | run 1 | run 2 |
| 1024 | 0.0050 | 0.0167 | 0.058 | 0.080 | 774 | 722 |
| 2048 | 0.0200 | 0.0167 | 0.128 | 0.138 | 758 | 744 |
| 3072 | 0.0400 | 0.0350 | 0.170 | 0.183 | 763 | 782 |
| 4096 | 0.0633 | 0.0550 | 0.217 | 0.233 | 768 | 769 |
| 8192 | 0.3667 | 0.3583 | 0.673 | 0.645 | 765 | 760 |
| 12288 | 0.4550 | 0.4617 | 0.873 | 0.905 | 774 | 767 |

Three of the six budget cells came from an early sweep whose run directories were lost; we re-ran them against a preregistered $\pm 2$ pp tolerance, recovering $0.0633$, $0.3667$, $0.4550$ against historical $0.072$, $0.390$, $0.455$. Two of the three fall inside the tolerance; the 8k cell misses it, at 2.33 pp. We report the miss rather than widening the band after the fact: it does not change the ladder's direction.

The two repeats disagree most at $B = 1024$ ($0.005$ against $0.017$), which is 3 against 10 correct rollouts out of 600. This is why the headline is the conservative $42\times$ computed on repeat means, and why the most conservative pairing of the two repeats still gives $27\times$.

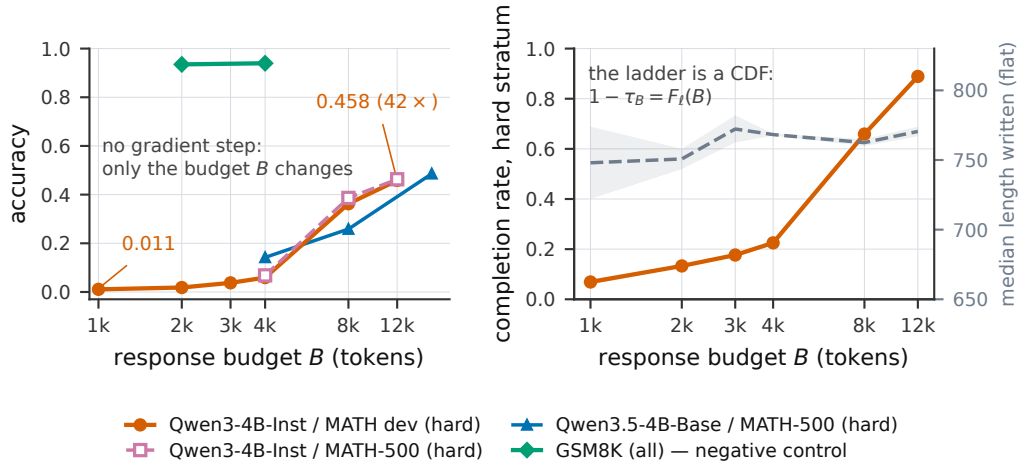



Figure S2: **Zero-training budget ladder.** No gradient step is taken; only $B$ changes. **Left:** hard-stratum accuracy rises from $0.011$ to $0.458$, by $44.7$ pp; the band spans two independent repeats per budget. We quote the difference rather than the $42\times$ ratio, because the denominator is 3–10 correct of 600 and the two repeats at the bottom rung differ by $3.3\times$ themselves. The same holds on MATH-500 and on a second policy; on GSM8K, whose length distribution sits far below every budget tested, the ladder is flat ($\Delta = +0.002$, interval $[-0.003, +0.007]$). **Right:** completion is the CDF $1 - \tau_B = F_\ell(B)$ of the response-length distribution. The median length written stays in 722–782 tokens while the mean rises $693 \to 1709$; the mean is itself censored (a truncated rollout contributes exactly $B$), the median is insensitive to that tail.

## S5 Budget ladders, stratified

Proposition 1 predicts that the completion factor, not the conditional factor, carries the budget dependence. The cleanest display is per stratum: for each stratum we report the pass rate $R$ and the truncation rate $T$ at every cap on disk. All cells are iteration-0 (zero training).

| family | stratum | cap 1 | | cap 2 | | cap 3 | |
|---|---|---|---|---|---|---|---|
| | | $R$ | $T$ | $R$ | $T$ | $R$ | $T$ |
| Qwen3.5-4B (4/8/16k) | hard ($n$=75) | 0.097 | 0.828 | 0.223 | 0.717 | 0.418 | 0.493 |
| | mid ($n$=40) | 0.284 | 0.672 | 0.566 | 0.375 | 0.681 | 0.216 |
| | easy ($n$=635) | 0.887 | 0.121 | 0.945 | 0.053 | 0.970 | 0.024 |
| | all ($n$=750) | 0.776 | 0.221 | 0.852 | 0.137 | 0.900 | 0.082 |
| Qwen3-4B-Inst (2/3k) | hard ($n$=75) | 0.020 | 0.872 | 0.040 | 0.830 | – | – |
| | mid ($n$=40) | 0.131 | 0.831 | 0.303 | 0.656 | – | – |
| | easy ($n$=635) | 0.900 | 0.121 | 0.962 | 0.049 | – | – |
| | all ($n$=750) | 0.771 | 0.234 | 0.835 | 0.160 | – | – |
| E5-mask arm (4/8k) | hard ($n$=75) | 0.272 | 0.445 | 0.423 | 0.133 | – | – |
| | mid ($n$=40) | 0.681 | 0.156 | 0.716 | 0.038 | – | – |
| | easy ($n$=635) | 0.963 | 0.009 | 0.966 | 0.003 | – | – |
| | all ($n$=750) | 0.879 | 0.060 | 0.899 | 0.018 | – | – |

Two readings. The truncation rate falls monotonically with the cap in every stratum, which is the CDF. And the budget sensitivity is concentrated in the hard stratum ($T$ from $0.828$ to $0.493$) while the easy stratum is nearly saturated at every cap ($T$ from $0.121$ to $0.024$): the budget is not a uniform difficulty knob, it is a per-prompt one. That heterogeneity is the premise of Proposition 4.

## S6 Paired bootstrap intervals

All ladder jumps and matched-budget arm contrasts, with paired bootstrap intervals over prompts (clustered by prompt, so the resampling unit is the prompt and not the rollout). “ratio” is the fold change in the hard stratum between the two caps.

| family | stratum | jump | Δ (95% CI) | ratio |
|---|---|---|---|---|
| ladfix Qwen3-4B-Inst | hard ($n$=75) | 4096 → 12288 | +0.392 [+0.308, +0.473] | 7.2× |
| ladfix Qwen3-4B-Inst | all ($n$=750) | 4096 → 12288 | +0.067 [+0.053, +0.081] | – |
| MATH-500 Qwen3-4B-Inst | hard ($n$=44) | 4096 → 12288 | +0.398 [+0.284, +0.517] | 7.7× |
| MATH-500 Qwen3.5-4B | hard ($n$=44) | 4096 → 16384 | +0.293 [+0.193, +0.395] | 3.6× |
| R1-Distill-Qwen-7B | hard ($n$=75) | 4096 → 8192 | +0.177 [+0.122, +0.235] | 4.0× |
| capladder Qwen3-4B-Inst | all ($n$=750) | 2048 → 3072 | +0.064 [+0.052, +0.076] | – |
| E5 mask arm | hard ($n$=75) | 4096 → 8192 | +0.152 [+0.092, +0.212] | 1.6× |
| E5 no-mask arm | hard ($n$=75) | 4096 → 8192 | +0.167 [+0.117, +0.217] | 1.6× |
| **GSM8K (negative control)** | all ($n$=1319) | 2048 → 4096 | **+0.002** [−0.003, +0.007] | 1.0× |

The negative control is the row that matters: on 1319 paired prompts the interval contains zero, so the ladder is absent exactly where the completion channel never engages. Every other row excludes zero by a wide margin.

Matched-budget contrasts between the rollout-allocation arms (all at $B = 8192$):

| contrast | Δ hard (95% CI) | verdict |
|---|---|---|
| A vs B | +0.003 [−0.073, +0.078] | not separable |
| A vs C | −0.020 [−0.095, +0.057] | not separable |
| B vs C | −0.023 [−0.085, +0.042] | not separable |

At the training budget (4096) the same three arms are separated by up to 18 pp, so the intervals above are the quantitative form of the claim in Section S19: what looked like a curriculum effect is within the resolution of the instrument.

## S7 Coverage matrix

Table S3: **Coverage matrix** (all seven cells; cited in Section 4). $\Delta\tau$ is the truncation rate released by the swept cap range; Δhard the resulting hard-stratum change. All cells are iteration 0, averaged over repeats.

| policy | prompt set | cap range | $\Delta\tau$ | Δhard |
|---|---|---|---|---|
| Qwen3-4B-Instruct | MATH dev | 1024–12288 | 0.380 | 0.448 (42×) |
| Qwen3.5-4B-Base | MATH dev | 1024–16384 | 0.399 | 0.432 (16×) |
| Qwen3-4B-Instruct | MATH-500 | 4096–12288 | 0.102 | 0.396 (7.2×) |
| Qwen3.5-4B-Base | MATH-500 | 4096–16384 | 0.116 | 0.318 (3.5×) |
| R1-Distill-Qwen-7B | MATH dev | 4096–8192 | 0.161 | 0.179 (4.2×) |
| Qwen3.5-4B-Instruct | MATH dev | 8192–16384 | 0.018 | 0.007 (1.08×) |
| Llama-3.2-3B-Instruct | MATH dev | 1024–2048 | 0.017 | 0.015 (1.06×) |

## S8 Per-update traces of the two collapsed arms

Table S4 gives the implied salvage value and training truncation rate, update by update, for both default-grader arms that saturated.

**Stratum-wise reward and changing selection.** Figure S3 recovers $b_x = (q_x - (1 - \tau_x)a_x)/\tau_x$ separately for the fixed easy/mid/hard strata, retaining every nonempty truncated subset. Because the batch value $b^\star$ is the truncation-count-weighted mean of the $b_x$, a rise in $b^\star$ can come from the $b_x$ themselves or from which stratum supplies the truncated samples. The two are separable only when every stratum has a nonempty truncated subset at both endpoints; otherwise $b_x$ is undefined at the base update and any split depends on the weighting order. `strata_decomposition.py` reports both orderings.

A43 rollouts $10 \to 11$ satisfy that condition, and there the movement is not a reweighting artefact: *every* stratum rises on its own truncated subset — easy $0.455 \to 0.947$ ($n = 22 \to 133$), mid $0.000 \to 0.481$ ($n = 10 \to 27$), hard $0.018 \to 0.149$ ($n = 56 \to 94$) — so the within-stratum term

dominates under both admissible weightings ($+0.358$ and $+0.261$, against composition $+0.120$ and $+0.216$) and $b^\star$ moves $0.125 \to 0.602$. Four further adjacent pairs in the ledger show all three strata rising simultaneously. Selection is present, but it is not the whole of the movement in the case where the question can be asked.

E12 rollouts $4 \to 5$ do not satisfy it: no easy sample truncates at rollout $4$, so the easy $b_x$ has no base value, and the split is order-dependent (composition $-0.107$ or $+0.446$). We therefore do not read that pair as evidence for either account. What it does show is unconditional: at rollout $5$, $60$ easy samples truncate and $57$ are accepted, while the hard stratum has $28$ truncated samples at mean reward $0.214$. Hence the batch figure $0.725$ at global $\tau = 0.089$ mixes a high easy-stratum reading with a low hard one, and low global truncation alone does not license reading $b^\star$ as one population's reward. The stratified $b_x$, not the batch $b^\star$, is the interpretable coordinate; $b^\star$ remains a summary that is exact by construction (Eq. 1) and useful as an alarm.

Two properties survive all of this. First, the stratified reading is not a weaker measurement. Across every installed-$b$ training arm on disk — 26 arms, five policy families (Qwen3-4B-Instruct, Qwen3.5-4B, Qwen3.5-4B-Base, Llama-3.2-3B-Instruct, DeepSeek-R1-Distill-Qwen-7B), six installed values and four caps — $b_x$ returns the installed constant to within $1.8 \times 10^{-14}$ in all 712 stratum-update cells with at least five truncated samples (`allrun_readback2.py`). Eq. (1) is exact *within* a stratum, on every family we trained, and not only on the batch. Second, positive salvage under the default grader is not confined to the collapse episodes, but it is strongly family-dependent. Among default-grader stratum-update cells with at least 10 truncated samples and global $\tau < 0.15$ (`allrun_strata.py`): Qwen3-4B-Instruct pays nonzero truncated reward in $313/406$ cells at cap 12288 ($77\%$) and $52/63$ at cap 4096 ($83\%$); Qwen3.5-4B-Base in $44/142$ across caps 1024–8192 ($31\%$); Llama-3.2-3B-Instruct in $15/85$ ($18\%$); DeepSeek-R1-Distill-7B in $5/95$ ($5\%$); Qwen3.5-4B in $1/37$ ($3\%$). Pooled, $432$ of $830$ cells ($52\%$). The family ordering matches how often each policy writes an accepted answer before the cap (Table 2 and Appendix S3): the salvage value is a property of the policy–grader pair, not of the budget alone, and a policy that rarely places its answer early pays little at truncation even under the same parser. Neither analysis identifies a within-prompt causal mechanism; a stratum's conditional prompt/response mixture can still change.

At E12 rollout 10, all $848$ easy responses receive reward 1, of which only 11 complete and 837 truncate. Thus verifier acceptance is $1.000$ and completed-and-correct mass is $11/848 = 0.0130$ on exactly the same stratum and batch, independently of any cross-stratum reweighting.

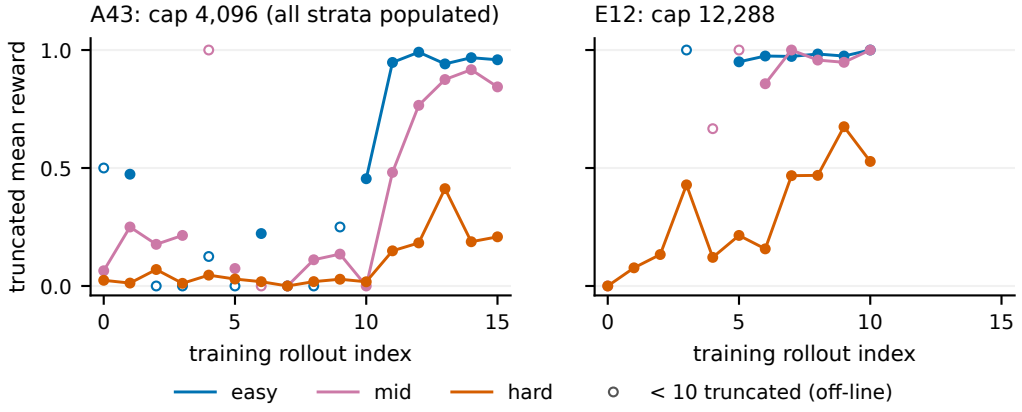



Figure S3: **Within-stratum salvage rises on its own, and batch-level drift also contains selection.** Conditional truncated reward is recovered separately in each existing difficulty stratum from the native training ledger. **Left (A43)** is the identified case: every stratum has truncated samples throughout the rise, and all three rise together at rollouts $10 \to 11$, so the movement is not only reweighting. **Right (E12)** has no truncated easy samples before rollout $5$, so its jump admits no order-free split and we do not rest on it. Every nonempty observation is shown; open markers have fewer than 10 truncated samples and are excluded from the connecting line, which joins only readings at or above that threshold. Gaps have no truncated samples. These are descriptive conditional means, not fixed-composition trajectories or confidence intervals.

| update | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *default grader,* $B = 12288$ *held* | | | | | | | | | | | | | | | | |
| $b^\star$ | .00 | – | .13 | – | .17 | **.73** | .87 | .88 | .93 | .95 | .93 | | | | | |
| $\tau$ | .02 | .01 | .03 | .02 | .04 | .09 | **.70** | .83 | .95 | .98 | .99 | | | | | |
| *default grader,* $B = 4096$ *held* | | | | | | | | | | | | | | | | |
| $b^\star$ | .04 | .17 | .08 | .04 | .06 | .04 | .08 | .00 | .04 | .10 | .13 | **.60** | .85 | .90 | .90 | .89 |
| $\tau$ | .11 | .13 | .11 | .10 | .09 | .10 | .09 | .09 | .07 | .08 | .09 | **.25** | .83 | .97 | 1.0 | 1.0 |

Table S4: Per-update implied salvage value and training truncation rate in the two default-grader arms that saturated. Every entry is one logged update; $b^\star$ is left blank where $\tau < 0.02$ and the quotient is undefined. Bold marks the first update at which $b^\star$ exceeds one half and the first at which $\tau$ exceeds one fifth. Neither launch configuration names a salvage value. Training truncation, logged once per update, is distinct from the hard-stratum evaluation truncation reported at a matched budget ($0.989$ and $0.992$ for the $12288$ arm; $0.147$ and $0.158$ for the stable $b = 0.5$ arm). The two arms ran under earlier revisions of the launcher than the multi-seed arms of Table S9; the revisions differ only in switches that were off in every arm reported here (Appendix S28). Table S5 gives a third case on the current launcher.

| arm | quantity | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| default | $\tau$ | .48 | .41 | .23 | .19 | .17 | .15 | .15 | .10 | **.98** | .93 |
| | $b^\star$ | .03 | .01 | .02 | .01 | .00 | .00 | .01 | .00 | **.61** | .59 |
| | completed∧correct | .52 | .59 | .73 | .78 | .73 | .76 | .63 | .70 | **.02** | .05 |
| mask | $\tau$ | .49 | .45 | .22 | .24 | .23 | .25 | .29 | .22 | .17 | .28 |
| | completed∧correct | .51 | .55 | .74 | .74 | .69 | .72 | .63 | .72 | .71 | .62 |
| $b_{\text{set}} = 0.88$ | $\tau$ | .49 | .48 | .27 | .22 | .27 | .22 | .32 | .27 | .29 | .40 |
| | completed∧correct | .51 | .52 | .70 | .75 | .70 | .75 | .64 | .71 | .69 | .57 |
| held-out strict, default | | .455 | .707 | – | .748 | – | .686 | – | **.010** | – | .662 |
| held-out strict, mask | | .451 | .701 | – | .698 | – | .708 | – | .704 | – | .552 |
| held-out strict, $0.88$ | | .454 | .674 | – | .730 | – | .706 | – | .631 | – | .544 |

Table S5: **A transient default-grader excursion on the current launcher, and two same-seed arms that do not show it.** Qwen3.5-4B-Base, cap $1024$, seed $47$, ten training updates; the three arms share seed, launcher revision and data order and differ only in truncation handling. Training quantities are per-update batch values from the ledger (completed∧correct is $(1-\tau)a$); held-out strict accuracy is evaluated at updates $0, 1, 3, 5, 7, 9$. Bold marks the excursion. In the default arm's update-$8$ batch, $1008$ of $1024$ rollouts truncate and $632$ are accepted; $604$ of the $912$ truncated easy-stratum rollouts receive reward $1$. The default arm's $b^\star$ is $0$ or near it before the excursion because this policy rarely writes a boxed answer before the cap (Table 2 pattern), so the jump in $b^\star$ and in $\tau$ arrive together. The mask and $0.88$ arms enter Table 3 and Section 5 as their seed-$47$ cells.

The transition in E12 is the single most important dynamical observation in the paper, so we also give the raw trace. Mean generated tokens per rollout, truncation rate, and millions of tokens consumed per update.

| update | 0 | 1 | 2 | 3 | 4 | 5 | **6** | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| mean length | 1643 | 1461 | 1924 | 1899 | 2407 | 2711 | **8981** | 10253 | 11676 | 12022 | 12160 |
| $\tau$ | .023 | .013 | .029 | .017 | .035 | .089 | **.704** | .829 | .948 | .978 | .989 |
| tokens (M) | 1.68 | 1.50 | 1.97 | 1.94 | 2.47 | 2.78 | **9.20** | 10.50 | 11.96 | 12.31 | 12.45 |

Two things to read off it. The first six updates move mean length by $+65\%$; the seventh moves it by $3.3\times$ in one step. And the fraction of groups in which every sample is wrong stays at or below $6.3\%$ throughout, so the collapse is not the "dead group" regime in which gradient vanishes for lack of a correct sample: groups retain contrast while length runs to the cap. We report this as a ruled-out explanation, not as an identified cause. Figure S4 gives the length, truncation and token-cost traces behind this paragraph.

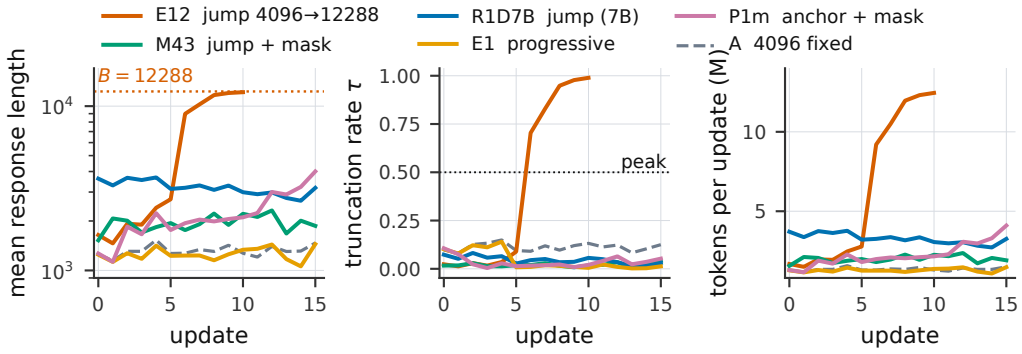



Figure S4: **A held** 12288 **cap under the default grader, and the ways out of it.** Mean rollout length (log scale), global truncation rate and tokens per update. The arm that trains at $B = 12288$ from the first update drifts for six updates and then, *in a single update*, moves from 2711 to 8981 mean tokens and from $\tau = 0.089$ to 0.704, saturating at the cap. Every other route to the same final budget stays flat. The collapse costs $8\times$ the tokens per update of the stable arms.

## S9 The intrinsic collapse, update by update

The cap-4096 terminal case of Table S4 never widens its budget: it trains at a fixed 4096 for all sixteen updates, with no mask and no schedule, so its collapse is not a hazard of aggressive widening. Its full training truncation trace is

```
0.113 0.132 0.109 0.103 0.094 0.096 0.086 0.086 0.074 0.078 0.086
                    | 0.248 0.828 0.974 0.998 1.000
```

with strict accuracy 0.0 at the final update. The first eleven updates are flat and even declining; the transition occupies three updates. Its seed-42 twin at the identical setting does not collapse. The comparison to the 12288 arm is one of timescale only: update six there against update twelve here; both held their cap from the first update.

## S10 The per-update regression in full

The sign of a per-update regression of $\Delta\tau$ on $\tau(1-\tau)$ separates outcomes on every arm; we treat it as description rather than mechanism (the regressor is the status-variance factor of Eq. (5), but the fit is on batch means and does not identify a gradient). The fitted table follows.

Table S6: **Feedback coefficient.** $\beta > 0$ means the status term amplifies truncation; $\beta < 0$ means it damps it. The sign separates the outcomes on every arm we have.

| arm | $\alpha$ | $\beta$ | $R^2$ | outcome |
|---|---|---|---|---|
| default grader (E12, 12288 held) | $+0.037$ | $+\mathbf{0.968}$ | 0.11 | collapse |
| default grader (seed 43, 4096 held) | $-0.150$ | $+\mathbf{2.401}$ | 0.45 | collapse |
| $b = 1.0$ | $-0.011$ | $+\mathbf{1.837}$ | 0.75 | hacking |
| $b = 0.25$ | $+0.018$ | $-0.536$ | 0.14 | stable |
| $b = 0.5$ | $+0.030$ | $-1.174$ | 0.49 | stable |
| $b = 0.5$ (second seed) | $+0.033$ | $-1.392$ | 0.17 | stable |
| mask (DAPO-style) | $+0.013$ | $-0.297$ | 0.08 | stable, drifting |
| mask (second seed) | $+0.024$ | $-1.255$ | 0.49 | stable |

This analysis is descriptive: masking changes loss support and the baseline-dependent status term, so a fitted coefficient does not directly measure the theoretical term. Moreover, $\Delta\tau$ is positive by construction on any arm whose truncation grows, so the regression partly restates the outcome it is meant to explain. What survives both is the contrast in fit quality, which says something the outcome does not — the $b = 1$ arm approaches saturation smoothly ($R^2 = 0.75$) while the drifting default-grader arm arrives in a single update ($R^2 = 0.11$).

## S11 THE FIXED-REWARD SWEEP IN FULL

**Every Qwen3-4B-Instruct arm at cap 12288, by grader.** Table S8 lists every completed sixteen-update arm of Qwen3-4B-Instruct at the 12288 cap, together with the two default-grader arms that collapsed. It is the population behind three numbers in the main text: the thirteen default-grader seeds none of which collapsed, the nine installed $b = 0.5$ seeds whose terminal spread is compared with theirs, and the peak $b^\star$ range. Seed 45 was run twice under the default grader on two nodes; both are listed and the spread statistics use one of them.

**The 25 arms on which Eq. (1) is exact.** Table S7 lists the original 25-arm calibration cohort whose manifests install constant truncation rewards (`trunc_beta_rm`). The subsequent fixed-0.88 follow-up is reported separately in Appendix S15. For each, $b^\star$ was recomputed from the three logged scalars at every update with $\tau > 0.02$; the range column is the minimum and maximum over those updates. In all 25 it equals the installed constant to three decimals at every readable update. Eq. (1) is an identity, so this is a check of the ledger's bookkeeping rather than a statistical test; it establishes that the three scalars are computed on the same sample set and that the default-grader readings of Table S4 can be trusted to the same precision.

| policy | cap | $b^{\text{set}}$ | seeds | readable updates | $b^\star$ range |
|---|---|---|---|---|---|
| Qwen3-4B-Instruct | 12288 | 0 | 42, 45 | 11, 8 | [0.000, 0.000] |
| Qwen3-4B-Instruct | 12288 | 0.25 | 42 | 10 | [0.250, 0.250] |
| Qwen3-4B-Instruct | 12288 | 0.5 | 42–50, 55, 56 (11) | 6–16 | [0.500, 0.500] |
| Qwen3-4B-Instruct | 12288 | 0.75 | 42 | 12 | [0.750, 0.750] |
| Qwen3-4B-Instruct | 12288 | 1.0 | 42 | 15 | [1.000, 1.000] |
| Qwen3.5-4B-Base | 1024 | 0, .25, .5, .75, 1 | 42 (each) | 10 (each) | $= b^{\text{set}}$ (each) |
| Qwen3.5-4B-Base | 12288 | 0.5 | 42 | 16 | [0.500, 0.500] |
| R1-Distill-Qwen-7B | 8192 | 0.5 | 42 | 16 | [0.500, 0.500] |
| Llama-3.2-3B-Instruct | 2048, 12288 | 0.5 | 42 (each) | 5 (each) | [0.500, 0.500] |

Table S7: Original installed-$b$ calibration cohort (25 in total: $2 + 1 + 11 + 1 + 1$ Qwen3-4B-Instruct, $5 + 1$ Qwen3.5-4B-Base, 1 R1-Distill-Qwen-7B, 2 Llama-3.2-3B). A "readable" update passes our reporting filter $\tau > 0.02$; the quotient itself is defined whenever $\tau > 0$. On every readable update, $b^\star$ equals $b^{\text{set}}$ to three decimals.

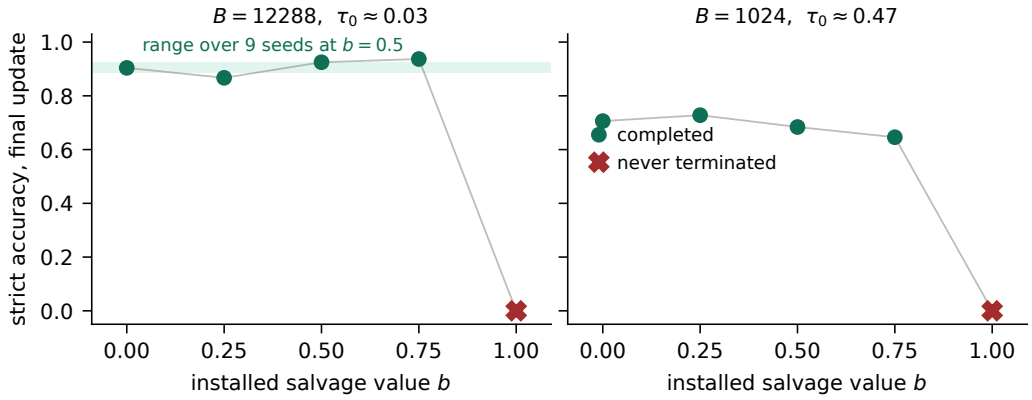



Figure S5: **The axis has one bad end, and a flat middle we do not order.** Each point is one completed arm at its final logged update, with $b$ installed in the grader. Left: the low-exposure family, where only about three per cent of first-rollout samples truncate, so there is little for $b$ to weight. Right: the high-exposure family at fifteen times that rate. In both, the interior does not order monotonically and the upper endpoint goes to exactly zero. **The shaded band is the observed range over the nine-seed fixed-$b = 0.5$ replication cohort of Table S8 and is drawn only to give scale** — it is not a confidence interval for a treatment difference, and we do not use it to claim the interior values are equivalent. The connecting line is a visual aid, not a fit. This figure is the reason the paper makes no claim about which interior value to prefer (Section 5).

**Default reward is not installed zero reward.** The collapsed E12 arm uses the default prefix grader and ends at training-batch $b^\star = 0.933$. The installed-$b = 0$ arm assigns zero to every truncated

| grader | seed | updates | final $\tau$ | final strict | peak $b^\star$ |
|---|---|---|---|---|---|
| default | 42 | 16 | 0.054 | 0.907 | 0.33 |
| default | 43 | 16 | 0.033 | 0.926 | 0.16 |
| default | 45 | 16 | 0.051 | 0.905 | 0.36 |
| default | 45 (second node) | 16 | 0.039 | 0.927 | 0.28 |
| default | 46 | 16 | 0.095 | 0.911 | 0.52 |
| default | 47 | 16 | 0.145 | 0.571 | 0.43 |
| default | 48 | 16 | 0.004 | 0.931 | 0.52 |
| default | 49 | 16 | 0.077 | 0.879 | 0.50 |
| default | 50 | 16 | 0.113 | 0.873 | 0.29 |
| default | 51 | 16 | 0.063 | 0.914 | 0.68 |
| default | 52 | 16 | 0.069 | 0.916 | 0.65 |
| default | 53 | 16 | 0.005 | 0.926 | 0.03 |
| default | 55 | 16 | 0.005 | 0.930 | 0.00 |
| default | 56 | 16 | 0.190 | 0.753 | 0.66 |
| default, collapsed | 42 | 11 | 0.989 | 0.045 | 0.95 |
| default, collapsed (cap 4096) | 43 | 16 | 1.000 | 0.000 | 0.90 |
| installed $b = 0.5$ | 44 | 16 | 0.093 | 0.907 | 0.50 |
| installed $b = 0.5$ | 45 | 16 | 0.074 | 0.889 | 0.50 |
| installed $b = 0.5$ | 46 | 16 | 0.089 | 0.911 | 0.50 |
| installed $b = 0.5$ | 47 | 16 | 0.060 | 0.888 | 0.50 |
| installed $b = 0.5$ | 48 | 16 | 0.029 | 0.909 | 0.50 |
| installed $b = 0.5$ | 49 | 16 | 0.036 | 0.923 | 0.50 |
| installed $b = 0.5$ | 50 | 16 | 0.073 | 0.923 | 0.50 |
| installed $b = 0.5$ | 55 | 16 | 0.027 | 0.885 | 0.50 |
| installed $b = 0.5$ | 56 | 16 | 0.059 | 0.913 | 0.50 |

Table S8: **Replication cohorts and the separately collected case runs.** The first block contains 14 default runs at cap 12288, with 13 distinct seeds; the second-node repeat of seed 45 is not included in the seed-level summary. These runs did not saturate. The two collapsed cases are shown separately and are not added to that denominator. Final $\tau$ and peak $b^\star$ are training-batch quantities; final strict accuracy is held-out evaluation. Peaks use the reporting rule $\tau > 0.02$, which is not the mathematical definition domain. The 13-seed default summary is mean 0.872, sample sd 0.102; the 9-seed fixed-0.5 replication cohort is mean 0.905, sd 0.015. The original fixed-0.5 sweep seeds 42/43 appear in Table S9, not this replication cohort. Low default endpoints are seeds 47/56; peaks above 0.6 occur in 51/52/56. The sets differ: 47 peaks at 0.43, while 51/52 end at high strict accuracy.

rollout instead. They are different reward implementations even when both launch configurations lack masking. The default cases and the installed-$b = 1$ failures both exhibit high truncated credit, but their observation does not establish a failure threshold or a common rate.

The following summary distinguishes training exposure from final evaluation: $\tau_0$ is the first training-batch truncation rate, whereas $\tau_{\text{end}}$ and $(1 - \tau)a$ are hard-stratum evaluation quantities. Strict accuracy and mean response length are full-set evaluation quantities, as in Table S9. These columns must not be substituted for one another in the reward identity.

Two readings. Across the interior, $b = 0.5$ reaches the highest completed-and-correct mass and the lowest final truncation, and $b = 0.25$ is close behind; we do *not* claim 0.5 is the optimum, only that every interior value we tried is stable. A third interior point, $b = 0.75$, is stable as well — sixteen updates, final hard-stratum truncation 0.340, strict accuracy 0.926, $(1 - \tau)a = 0.473$ — so the interior now has three points on seed 42 and a second seed at 0.5, and none of them fails. And the last row is the reward-hacking endpoint in its purest form: strict accuracy exactly zero, mean length pinned at the cap, and a logged accuracy of 1.000 that Sec. S24 explains.

One bookkeeping note, because the ledger is meant to be checkable: the seed-43 interior arm finished after the numbers were first frozen, and an earlier reading of it ($\tau = 0.362$, strict 0.901, taken at update seven of sixteen) circulated in a draft. The row above is the completed run.

Table S9: **One scalar, five installed values, plus the default arm that collapsed.** At the final evaluation, **strict and mean $L$ are full-set quantities**; $\tau$, $(1-\tau)a$ and "acc." are hard-stratum quantities. "acc." is the logged mean reward, $(1-\tau)a+\tau b$, comparable within a row and meaningless across rows with different $b$ (Sec. S24); on every installed row it reverse-solves to exactly $b^{\text{set}}$. **The first row uses the unmodified grader**; its batch-level training $b^\star$ ended at 0.933, while its hard-stratum evaluation entry reverse-solves to $b = 0.43$: the grader accepts a truncated prefix less often on hard prompts than on the batch, which is the stratum dependence Eq. (1) is defined per. Shaded rows and the last row have an installed $b^{\text{set}}$. The $b = 0$ row is seed 42; the additional seed 45 completes at full-set strict accuracy 0.9145 (Appendix S28).

| $b$ | schedule | hard $\tau$ | full $L$ | full strict | hard strict | hard reward |
|---|---|---|---|---|---|---|
| default ($b^\star \to 0.93$) | cap held, collapsed | 0.992 | 12160 | 0.045 | 0.003 | 0.432 |
| 0 | cap held | 0.353 | 4869 | 0.925 | 0.443 | 0.443 |
| 0.25 | cap held | 0.248 | 2616 | 0.926 | 0.453 | 0.515 |
| 0.50 | cap held (seed 43) | 0.378 | 5042 | 0.927 | 0.478 | 0.667 |
| 0.50 | cap held | **0.158** | **2420** | **0.933** | **0.492** | 0.571 |
| 0.75 | cap held | 0.340 | 3438 | 0.926 | 0.473 | 0.728 |
| 1.00 | cap held | 1.000 | 12288 | **0.000** | 0.000 | 1.000 |

| arm | upd | $\tau_0$ | $\tau_{\text{end}}$ | strict | $(1-\tau)a$ | mean $L$ last |
|---|---|---|---|---|---|---|
| default, 12288 held, collapsed ($b^\star \to 0.93$) | 11 | 0.023 | 0.992 | 0.045 | 0.003 | 12160 |
| default, 4096 held, seed 42 | 16 | 0.098 | 0.688 | 0.861 | – | 1468 |
| default, 4096 held, seed 43, collapsed ($b^\star \to 0.89$) | 16 | 0.113 | 1.000 | **0.000** | 0.000 | 4096 |
| $b = 0.25$ | 16 | 0.023 | 0.248 | 0.927 | 0.453 | 3293 |
| $b = 0.50$ (seed 42) | 16 | 0.027 | 0.158 | **0.933** | **0.492** | 2674 |
| $b = 0.50$ (seed 43) | 16 | 0.015 | 0.378 | 0.927 | 0.478 | – |
| $b = 1.00$ | 16 | 0.026 | 1.000 | **0.000** | 0.000 | 12288 |
| *second policy family: Llama-3.2-3B-Instruct, cap* 12288 *held, seed* 42 | | | | | | |
| $b = 0$ | 16 | 0.043 | 0.030 | 0.708 | 0.393 | 504 |
| $b = 0.50$ | 16 | 0.047 | 0.040 | **0.739** | **0.455** | 628 |

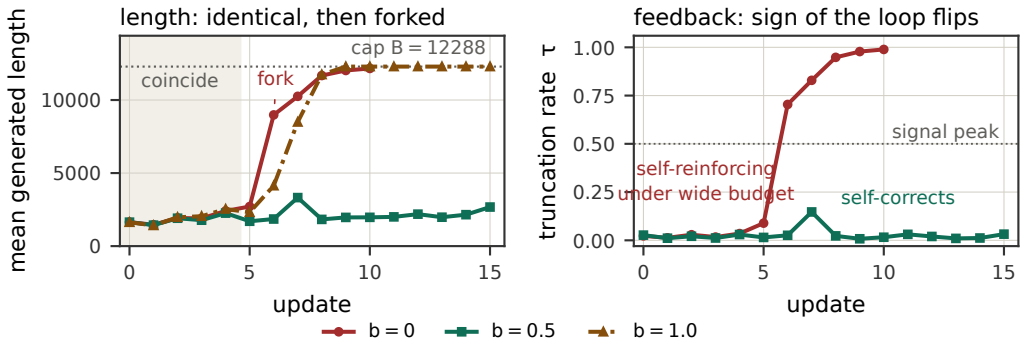



Figure S6: **One scalar, two feedback regimes.** *Left*: mean generated length at a cap held from the first update; the curves coincide for five updates then fork. *Right*: the training truncation rate. The default arm changes from 0.089 to 0.704 in one rollout and does not reverse over its observed window. The separate E3 budget-narrowing run reaches 0.883 strict accuracy; it is not a restart of the saturated checkpoint. In the installed-0.5 run, truncation reverses from 0.147 to 0.023. These trajectories do not establish an absorbing state or a universal critical point.

## S12 Length and truncation traces under the censored reward

Figure S6 shows the per-update traces behind Section 5: the two arms differ only in $b$, share the cap, pool, schedule and seed, and diverge only after five updates of near-identical behaviour.

## S13 COUNT-MATCHED PLACEBOS

Masking removes roughly half the rollouts on hard prompts, so the obvious objection is that its benefit is generic subsampling rather than anything about truncation. We ran two count-matched placebos on the collapsed arm: one deletes the same number of rollouts chosen uniformly at random, the other deletes preferentially the longest *completed* ones. Neither is targeted at censoring. Both avert collapse (strict accuracy $0.808$ and $0.844$ against $0.045$ for the untreated collapsed arm), which is why we say collapse is a fragile critical phenomenon that almost any perturbation displaces. What placebos do not do is recover quality: completed-and-correct on the hard stratum they reach $0.138$ and $0.185$, against $0.368$–$0.467$ for the censoring-targeted arms and $0.492$ for $b = 0.5$. Breaking the feedback loop is easy; breaking it without discarding the signal is not, and that is the sense in which the treatment should be aimed at censoring.

## S14 UNEXECUTED PRELIMINARY REWARD TEST

To separate the $(a - b)$ factor in Eq. (5) from $\tau$ itself, we designed an intervention that changes $b$ without touching the length dynamics: pay truncated rollouts a reward $\beta$ instead of $0$, with $\beta \in \{0.5, 1.0\}$, holding policy, budget (12288), seed (42) and masking fixed and comparing against E12. Written before any result existed, the prediction was that raising $\beta$ shrinks $(a - b)$ and therefore collapses *earlier*: the first collapse update should move from 6 (measured) to 4–5 at $\beta = 0.5$ and 3–4 at $\beta = 1.0$, with the falsification condition that $\beta = 1.0$ not collapsing at all would invalidate the $(a - b)$ factor.

These preliminary launch attempts exited non-zero before producing rollouts. They supply no test of the proposed collapse-time prediction, which is not a consequence of the variance identity alone. They are distinct from the completed installed-reward sweep and calibration arms in Table S7; no failed launch is counted as a completed experimental arm.

## S15 SUPPLEMENTARY SCALE AND FIXED-REWARD EXPERIMENTS

### S15.1 ZERO-TRAINING CALIBRATION OF A DENSE 27B MODEL

We evaluate the official Qwen3.5-27B checkpoint at revision `fc05daec18b0a78c049392ed2e771dde82bdf654`, with all eleven weight shards verified against their published hashes. From the same development pool we choose 32 prompts in each pre-existing easy/mid/hard stratum by sorting the SHA256 of `q27pilot-v1:uid`. These 96 prompts are fixed before generation and reused at caps $1024$, $4096$ and $12288$, with two sampling seeds ($42$, $43$), yielding $576$ responses. This is a balanced pilot, not a population-weighted score on the original $750$-prompt distribution. Stratum labels are inherited, not re-estimated for 27B.

The official chat template is used with thinking mode disabled, temperature $0.6$, top-$p = 0.95$, no top-$k$ restriction and repetition penalty $1$. SGLang $0.5.13$ serves BF16 weights with tensor parallelism $2$, a $16384$-token context and at most $32$ concurrent requests. The longest templated prompt is $1783$ tokens, so the requested output caps fit the context. The unchanged deepscaler verifier scores each returned text. A response is truncated exactly when the engine records a length finish; all $241$ such records have output length equal to the requested cap. Every sample, seed, prompt identifier and finish reason is retained. The run generated $1{,}376{,}273$ output tokens in about $85$ minutes including loading on two H20 GPUs; no weights were updated.

| cap | truncated / 192 | accepted trunc. | mean reward | strict acc. | $b^\star$ |
|---|---|---|---|---|---|
| 1024 | 143 | 0 | 0.240 | 0.240 | 0.000 |
| 4096 | 59 | 19 | 0.734 | 0.635 | 0.322 |
| 12288 | 39 | 15 | 0.786 | 0.708 | 0.385 |

Table S10: Dense-27B inference-only pilot, all $96$ balanced prompts with two samples per cap. Accepted-truncated counts are included in mean reward but excluded from strict accuracy. Conditional reward is computed directly from these counts and independently recovered using Eq. (1). Both sample seeds are generation replicates, not independently trained policies.

Table S10 shows nonzero credit on $19/59$ and $15/39$ truncated outputs at the two larger caps; all these accepted truncated responses contain a boxed answer. At the smallest cap, none of the $143$ truncated outputs is accepted. On the 32 hard prompts (64 responses per cap), strict accuracy is $0.047$, $0.250$ and $0.391$; truncation is $0.906$, $0.672$ and $0.469$. This extends the measurement phenomenon to a larger dense model, not the training-drift mechanism or its frequency. It is kept separate from the earlier five-model/seven-cell coverage analysis because its sampling weights, sample count and inference configuration differ.

### S15.2 A FIXED HIGH REWARD COMPARED WITH DEFAULT GRADING AND TOKEN MASKING

The follow-up uses Qwen3.5-4B-Base, seed $43$, cap $1024$, context $3072$, ten training rollouts of $64$ prompt groups with $16$ responses each, learning rate $8 \times 10^{-6}$ and no group-standard-deviation normalization. It compares with the existing seed-43 default and token-mask runs at the same cap and update indices. Each evaluation uses the same $750$ prompts and $8$ samples per prompt. The fixed value $0.88$ is the rounded arithmetic mean of the historical default arm's ten training-batch completion-conditional accuracies ($0.881893$). It is chosen before this follow-up, not adapted during training and not equal to each prompt group's $a$.

The three manifests agree on their recorded training hyperparameters and the hashes of the launcher, verifier, rollout monitor and its entry point, model configuration, training data and evaluation data. The fixed arm instead selects a custom reward callback; tracking names also differ. Initialisation source and checkpoint marker match, but a marker is not a hash of every weight tensor. Historical hook code and all runtime state were not fully frozen, so this is not a simultaneous randomized multi-seed comparison. Treatment delivery is independently checked: the default and fixed arms remove zero samples at every update; the mask arm removes 102–520. The fixed arm has $b^\star = 0.88$ at all ten training readings and all six evaluations, to numerical precision.

| eval rollout index | default strict | token-mask strict | fixed-0.88 strict |
|---:|---|---|---|
| 0 | 0.4533 | 0.4567 | 0.4575 |
| 1 | 0.6953 | 0.6753 | 0.5698 |
| 3 | 0.7338 | 0.7163 | 0.6292 |
| 5 | 0.7207 | 0.6948 | 0.4983 |
| 7 | 0.6603 | 0.7050 | 0.3578 |
| 9 | 0.6718 | 0.6903 | 0.3148 |
| final full $\tau$ | 0.1955 | 0.1955 | 0.6787 |
| final hard $\tau$ | 0.7167 | 0.7117 | 0.9967 |

Table S11: All six matched evaluations of the fixed-high-reward follow-up and its historical comparators. Index 0 is the initial evaluation; the last reading follows ten training rollouts. Strict accuracy is full-set completed-and-correct mass. The fixed reward is also applied during evaluation, so its final mean reward $0.91206$ must not be substituted for strict accuracy $0.31483$.

Relative to default, the seed-43 fixed arm ends $35.7$ pp lower despite starting $0.42$ pp higher; its difference in changes is $-36.12$ pp. Four replication seeds with the same installed $0.88$ (Section 5) end $13.2$, $22.3$, $40.9$ and $11.8$ pp (seeds 44–47: $0.592$, $0.521$, $0.311$, $0.544$ against defaults $0.724$, $0.744$, $0.720$, $0.662$) below their own default arms, with hard-stratum truncation $0.922$, $0.932$, $0.988$ and $0.962$; their masking arms end $+1.5$, $-4.8$, $-7.1$ and $-10.9$ pp against default, and with the seed-42 and seed-43 pairs make up the six cap-1024 cells of Table 3. Masking instead ends $1.85$ pp higher than default, from a $0.33$ pp initial difference. Table S11 retains all scheduled readouts, rather than selecting a checkpoint that maximizes the contrast. The result shows a direct fixed-reward intervention with severe loss of completion quality, and different outcomes from masking in this setting. It does not validate masking as an implementation of $b = a$ or identify a universal reward threshold. It rules out treating an interior fixed reward as sufficient for reliable completion. The $0.88$ arms are not pooled into the original 25-arm calibration cohort.

### S15.3 PREREGISTERED MATCHED-FORK TEST OF THE WITHIN-TRUNCATION TERM

The decomposition of Section 3 attributes to the default grader a term, $P(T)\nabla b = \mathbb{E}[T(r-b)z]$, that an installed constant removes exactly. The paper shows the term is algebraically present and that $b^\star$ moves; it has not shown that removing the term, while holding the reward mean fixed at the moment

of removal, changes what the policy does next. The following design isolates that term and nothing else. It was written, and this subsection was compiled into the manuscript, before any arm finished.

**Design.** For each seed $s$ in the cap-12288 default-grader cohort (Table S8) whose saved `iter_0000007` checkpoint has at least 15 truncated samples in its training batch, two arms are launched from the same weights, the same seed, the same dataset cursor (the parent's `global_dataset_state_dict_7`) and the same launcher, both continuing rollouts 8–15:

- **control**: the default prefix grader. $b(\theta)$ continues to move with the policy; the within-truncation term is present.
- **pinned**: every truncated rollout receives the constant $b_{\text{set}} = b^\star_s(7)$, read from the parent's own ledger at rollout 7 by Eq. (1). At the fork batch the two arms share the batch reward mean and status contrast $a - b$; the within-truncation reward variance in the pinned arm is zero by construction, so $\mathbb{E}[T(r-b)z] = 0$ there. The pinned value is a historical batch scalar: it does not enforce per-prompt or later-update mean matching, so the local gradient difference is $(b - b_{\text{set}})\nabla\tau + \tau\nabla b$, of which only the second term is removed by construction (Appendix S2.1).

Both arms reload weights without optimizer state (`--finetune`); they lose it identically, so neither is the parent's literal continuation but the pair is matched. Installed values: $s$=42 : 0.2800, $s$=45 : 0.1250, $s$=46 : 0.4717, $s$=50 : 0.0556, $s$=52 : 0.2941, $s$=56 : 0.0484.

**Preregistered read-out.** Primary: training truncation rate at rollout 15, control minus pinned, per seed, together with the control arm's $b^\star$ trajectory (the pinned arm's is constant by construction and serves as the bookkeeping check). Secondary: held-out strict accuracy and hard-stratum completed-and-correct mass at rollout 15. Tertiary, post hoc: first-box position on reloaded outputs. *Interpretation rule*, fixed in advance: a consistent sign of the primary difference across at least three seeds is reported as evidence that the within-truncation term has a behavioural consequence; a mixed or null pattern is reported as such and Hypothesis 3 remains a hypothesis. No seeds are added after results are read, and no seed is dropped for any reason other than a launcher failure recorded in Appendix S28.

**Results.** All six preregistered pairs completed; no seed was added or dropped. The pinned arm returns its installed constant at every update (maximum deviation $4.4 \times 10^{-5}$, from batches with as few as 10 truncated samples), so the bookkeeping check passes: the pinned arm carries no within-truncation reward variation at any update. On the primary read-out, control truncation at rollout 15 exceeds pinned truncation in **four of six seeds**, which meets the preregistered rule and is reported, as that rule specifies, as evidence that the within-truncation term has a behavioural consequence. Table S12 gives every seed. The secondary read-out, held-out strict accuracy, has no consistent sign

| seed | $b_{\text{set}}$ | ctrl $\tau_{15}$ | pinned $\tau_{15}$ | diff | ctrl strict | pinned strict | ctrl $b^\star_{8\to15}$ |
|---|---|---|---|---|---|---|---|
| 42 | 0.280 | 0.040 | 0.091 | −0.051 | 0.919 | 0.890 | 0.09 → 0.32 |
| 45 | 0.125 | 0.061 | 0.064 | −0.004 | 0.916 | 0.899 | 0.00 → 0.13 |
| 46 | 0.472 | 0.071 | 0.048 | +0.023 | 0.911 | 0.915 | 0.21 → 0.34 |
| 50 | 0.056 | 0.082 | 0.073 | +0.009 | 0.853 | 0.911 | 0.29 → 0.37 |
| 52 | 0.294 | 0.010 | 0.001 | +0.009 | 0.935 | 0.931 | 0.06 → 0.00 |
| 56 | 0.048 | 0.111 | 0.013 | +0.099 | 0.889 | 0.935 | 0.15 → 0.54 |

Table S12: Preregistered matched-fork read-out at rollout 15, eight updates after the fork. "diff" is control minus pinned training truncation. Strict is held-out completed-and-correct accuracy at the final evaluation. The last column is the control arm's recovered $b^\star$ at rollouts 8 and 15; the pinned arm's equals $b_{\text{set}}$ throughout.

across seeds (three of each). The control arm's $b^\star$ rises over the eight updates in five of six seeds; the conditional reward the pinned arm holds fixed is moving in the arm that keeps it (the gradient contribution also depends on the policy score and is not read from $b^\star$).

*Post hoc, not preregistered.* The decomposition assigns the within-truncation term a weight of $\tau\, b(1-b)$ in the batch reward variance $q(1-q)$. Averaged over rollouts 8–15 of each control arm, that share is 2.5% (s52), 5.0% (s45), 8.8% (s50), 10.7% (s42), 26.2% (s56) and 26.9% (s46). The two seeds in which the term carries the most signal are the two with the largest positive primary

difference ($+0.099$, $+0.023$); the two in which it carries the least are the two nearest zero. This ordering is what the decomposition predicts and is offered as a consistency check, not as a second test.

## S16 MATCHED-BUDGET TABLE, BOTH SEEDS

**Policy-equal weighting of Table 3.** Qwen3.5-4B-Base contributes nine of the twelve matched cells. Averaging within each of the four policies and weighting policies equally gives a mean of $-2.8$ pp with 95% $t$ interval $[-7.7, +2.2]$ (sample sd, 3 degrees of freedom); this changes the estimand and is not a cluster-robust interval for the twelve-cell mean of $-5.3$ pp, whose own interval is $[-8.9, -1.6]$. Within Qwen3.5-4B-Base alone, seven of nine cells are negative (mean $-6.5$ pp).

**Reload cohort.** All arms scored at a single evaluation budget $B = 12288$ on the full 750-prompt set (6,000 rollouts; hard stratum $n = 600$). Readings were verified by an independent script that reads the monitor's native token ledger rather than any report file; the four groups have structurally identical evaluations ($750 \times 8$), and neither the verifier nor the evaluation monitor changed during the period spanning all evaluations.

| arm | seed | hard $q$ | hard completion | hard $\tau$ | hard $a$ | all |
|---|---|---|---|---|---|---|
| A | 42 | 0.4400 | 0.7283 | 0.2717 | 0.5561 | 0.9215 |
| M42 | 42 | 0.4950 | 0.5167 | 0.4833 | – | 0.9257 |
| M43 | 43 | 0.5083 | 0.7433 | 0.2567 | 0.6390 | 0.9330 |
| E1 | 42 | 0.4317 | 0.9417 | 0.0583 | 0.4531 | 0.9150 |
| E1 | 43 | 0.4100 | 0.9517 | 0.0483 | – | 0.8802 |
| E2 | 42 | 0.4083 | 0.9350 | 0.0650 | 0.4260 | 0.9120 |
| E2 | 43 | 0.4967 | 0.8633 | 0.1367 | – | 0.9268 |

The seed spread visible here is the basis of the variance claim in Section S17: $|\Delta|$ across seeds is 1.3 pp for the mask recipe (0.4950/0.5083), 2.2 pp for the progressive schedule (0.4317/0.4100) and 8.8 pp for the two-update anchor (0.4083/0.4967).

The per-budget rungs of the zero-training ladder of Section 4, with the completion factor and conditional correctness separated, are in Table S13.

Table S13: **Zero-training budget ladder**, Qwen3-4B-Instruct-2507, hard stratum ($n = 600$). Means of two independent repeats. $q = (1 - \tau)a + \tau b$ holds with an implied $b$ between 0.01 and 0.11.

| $B$ | $q$ | $1 - \tau$ | $a$ | median length hard | median length all |
|---|---|---|---|---|---|
| 1024 | 0.011 | 0.069 | 0.137 | 1024 | 748 |
| 2048 | 0.018 | 0.133 | 0.119 | 2048 | 751 |
| 3072 | 0.038 | 0.177 | 0.175 | 3072 | 772 |
| 4096 | 0.059 | 0.225 | 0.223 | 4096 | 768 |
| 8192 | 0.363 | 0.659 | 0.532 | 6861 | 763 |
| 12288 | 0.458 | 0.889 | 0.501 | 6707 | 771 |
| $1024 \rightarrow 12288$ | | 12.9× | 3.7× | 1.03× (all) | |

## S17 WHAT ACTUALLY BUYS ACCURACY AT A MATCHED BUDGET

In our setup the evaluation budget is bound to the training budget, so endpoints read at each arm's own training budget are read with different instruments and cannot be compared. The size of the error: arm A's own checkpoint scores 0.162 on the hard stratum at $B = 4096$ and 0.440 at $B = 12288$ — one policy, $+27.8$ pp, purely from the instrument. Everything below is scored at one matched budget, with the full table in Appendix S16.

**Widening alone buys nothing we can detect.** E1 and E2 reach the wide budget safely and at their own budget look 5 pp better (0.911/0.914 against 0.861). At the matched budget they are 0.432 and 0.408 against A's 0.440: no gain, if anything a small loss. The progressive schedule does not earn the wide budget; it only avoids being destroyed by it.

**A sign reversal we owe the reader an account of.** Measured against arm A, all four censoring-intervention runs come out ahead on the hard stratum: $+5.3$, $+6.8$, $+10.7$ and $+12.3$ pp, replicated in two independently constructed recipes (anchor-plus-mask seeds differ by $1.7$ pp, mask-only by $1.5$ pp). With $75$ hard prompts these are one to two standard errors each, so at most a trend. **An earlier draft explained this away by calling A a collapsed baseline. That is wrong and we correct it here:** A completed all sixteen updates with hard-stratum $1 - \tau = 0.728$ and $b \approx 0.13$, which is a healthy arm, not the collapse discussed in Appendix S28. So the reversal against the $-5.3$ pp of Section 6 is real and is not explained by baseline identity.

What distinguishes the two comparisons is that they are not the same contrast. Section 6 is *paired*: treated and untreated differ in one flag with seed, policy, budget and pool held. The rows here are *unpaired* — A, M and P1m differ in budget schedule as well as in masking, and A is evaluated at a cap far above its own training budget ($0.162$ at its own budget against $0.440$ at $B = 12288$, the $+27.8$ pp instrument effect of Appendix S24). A cross-schedule difference of 6–12 pp is therefore not comparable to a within-configuration difference of $-5.3$ pp, and we do not average or reconcile them. The paired estimate is the one we interpret; these rows we report because they exist and because they disagree, and we flag the disagreement rather than harmonising it. Conditional correctness is where the unpaired advantage sits ($0.768$ and $0.619$ against $0.556$) while completion rates bracket A's, so whatever is happening acts through $a$ and not through $1 - \tau$.

**The switch, tested: one half holds, one half misses.** A pre-registered sibling pair, each run with and without masking on the same pool (predictions filed in advance: instruct $\Delta \geq +8$ pp, base $\Delta \leq -5$ pp). **The base prediction held and the instruct prediction failed**: the base arm lost $5.5$ pp against a predicted $\leq -5$, while the instruct arm moved $-1.5$ pp against a predicted $\geq +8$, missing on direction as well as magnitude. What the miss shows is narrower than the prediction was: where the untreated run already terminates, masking has nothing to rescue, and we do not read $-1.5$ pp as a null. Crucially the base penalty is *flat* in $\tau$: four pairings over $\tau_0 \in [0.22, 0.84]$ give $-8.0$, $-9.5$, $-10.3$, $-8.0$ pp (sd $0.99$), where a dosage effect would have varied roughly fourfold. The sign is set by whether the policy can stop, not by how much censoring it sees. Per-arm traces are in Figure S8.

**What the switch buys.** It turns a contested recipe into a measurement: read whether the policy terminates on its own before deciding what masking can do for it. We state this as a reading aid, not a predictive rule — the preregistered instruct prediction failed on direction (Hypothesis 3), and the effect depends on the policy and on which baseline and loss mask are used. In that light masking is a *rescue operation rather than a performance upgrade* — worth a great deal in the one comparison where the untreated arm collapsed ($+88$ pp), worth approximately nothing once the untreated run is stable ($-1.5$ pp, $+0.2$ pp, $-3.4$ pp), and, as the paired design of Section 6 shows, negative on average ($-5.3$ pp over twelve matched cells, nine of twelve negative) once the untreated arm is not in danger. Because E1/E2 make the wide budget usable without beating the tight-budget baseline, what benefit there is belongs to the censoring channel rather than to the budget. Partial credit acts on the same channel without discarding tokens, but it is not a free substitute: Section 5 reports that both installed-$b = 1$ arms saturate and that the fixed-$0.88$ arms lose 12–41 pp in five seeds strict accuracy, so replacing the reward has consequences of its own.

## S18 Does censoring exposure moderate the intervention?

If the cost of the intervention of Section 6 comes from losing the termination gradient, it should be largest where truncation is common. It is not, in these data. Over the twelve main-table pairs, regressing $\Delta$ strict (pp) on the untreated arm's final hard-stratum truncation rate gives a slope of $+5.4$ pp per unit $\tau$ (se $8.7$, $t = 0.62$, $\mathrm{df} = 10$); the rank statistic is $\rho = +0.23$. On the original seven-pair subset the slope was $+1.5$ ($t = 0.10$, $\rho = -0.14$). No moderation by exposure is detected. So the twelve pairs establish a direction and not a dose-response, and the decisive evidence for a sign reversal along the budget axis remains Liu et al. (2025d) rather than our data.

Each arm was evaluated between two and six times; the pair difference that enters this regression is read at both members' final logged evaluation, a rule fixed before the analysis, and the accumulation result of Section 6 likewise reads only the first and last matched evaluation of each cell.

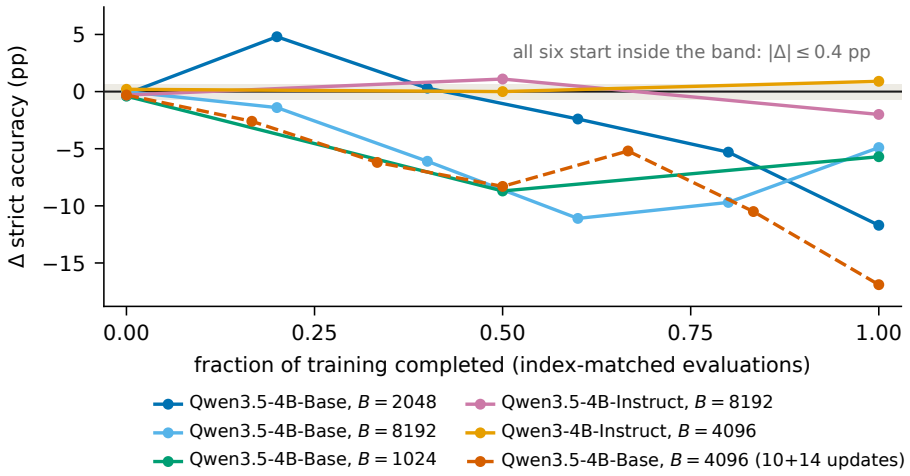



Figure S7: **The cost of masking accumulates, and it starts at zero.** Each line is one matched cell, both members read at the *same* update count; the whole trajectory is plotted, so no checkpoint is selected. All six begin inside the shaded band ($|\Delta| \leq 0.4$ pp); five end worse than they began; the Qwen3-4B-Instruct cell stays within a point of zero. The dashed trajectory is the Qwen3.5-4B-Base cell at 4096, one run of ten updates continued for fourteen more from its own checkpoints. Non-monotone excursions within a trajectory are expected at this evaluation noise; the claim is about the endpoints.

## S19 Rollout-allocation arms

Three arms trained identically at $B = 4096$: **A** samples uniformly, **B** spends extra rollouts on low-support prompts, **C** targets prompts with high estimated truncation hazard. Each scored at its training budget and at a common $B = 8192$ that none of them saw in training.

| | | evaluated at $B = 4096$ | | evaluated at $B = 8192$ | |
|---|---|---|---|---|---|
| arm | | hard | all | hard | all |
| A | uniform | 0.162 | 0.868 | 0.392 | **0.917** |
| B | extra rollouts | **0.343** | 0.871 | 0.385 | 0.879 |
| C | hazard-targeted | 0.285 | 0.873 | 0.382 | 0.886 |
| spread | | 18.1 pp | 0.5 pp | 1.0 pp | 3.8 pp |

At the training budget the arms are separated by 18.1 pp on the hard stratum and B looks clearly best; at the matched budget the gap is 1.0 pp and on the full set the ordering reverses with A on top. We do not treat this as a result about curriculum methods — it is one comparison at 75 prompts, and seed-to-seed spread on that stratum reaches 6.3 pp, so neither the 18 pp nor the 1 pp is a stable quantity. We report it because it is the clearest illustration of what an unmatched budget does to a comparison, and because a reader is entitled to see it. It is deliberately not one of the paper's contributions. The accuracy and truncation readouts for the architecture-matched sibling pair are in Figure S8.

## S20 The budget is allocatable

Because $1 - \tau_B$ is a CDF in the budget, prompts sit at different quantiles of it, so a shared cap leaves accuracy on the table. Off-line (no training, only the per-prompt ledgers already on disk), re-allocating the same total token spend per prompt beats every single cap we tested across four model–prompt settings: $+1.9$ to $+4.6$ pp at 2.5–9.9% *fewer* tokens, with bootstrap 95% intervals that exclude zero, and exactly zero where the probe carries no information. The allocation table is offline and in Appendix S21 (Table S14); nothing downstream depends on it.

We claim no priority for the allocation itself. Per-query budget allocation under a global token constraint has been posed and solved more completely elsewhere: Wan et al. (2026) derive an

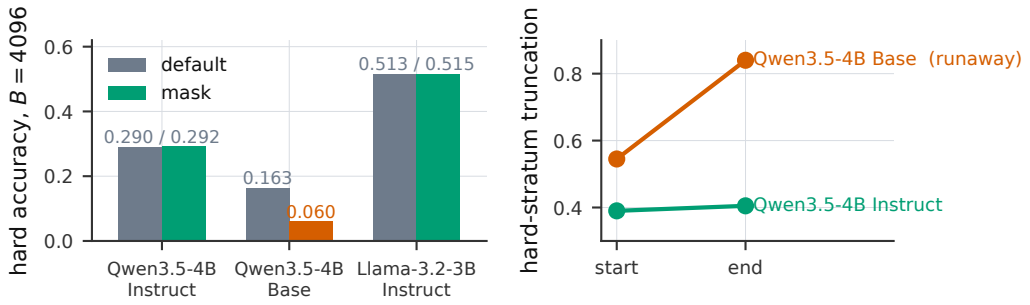



Figure S8: **The switch.** *Left:* hard accuracy at $B = 4096$ for architecture-matched siblings and a cross-family instruct policy; masking is null on both instruct policies and harmful by 10.3 pp on the base policy. *Right:* hard-stratum truncation over the same continuation; the base policy runs away ($0.545 \rightarrow 0.840$) while the instruct sibling stays flat. Both readouts are at each arm's own budget — the matched-budget version of the same experiment is the switch test in Appendix S17.

optimal policy from a shadow price that equalises marginal utility, including the decision to abandon unsolvable queries, and report larger gains than ours in the scarce regime. We include this section for a narrower reason: it is the prediction the coordinate change makes. If $1 - \tau_B$ is a CDF evaluated at the cap, then a shared cap is a shared quantile applied to prompts with different length distributions, and re-allocating must help wherever those distributions differ, and must return nothing where they do not, which is the zero we report. Read it as a test of the framing, not as a method.

## S21 The budget-allocation exercise in full

**Input.** For each (model, prompt set) with at least three caps on disk, the iteration-0 ledger of each cap gives, *per prompt*, the accuracy at that cap and the tokens actually generated at that cap. No model of cost or accuracy is used; both are logged.

**Rule.** Sort the cap grid. Start every prompt at the cheapest cap. Repeatedly buy the single cheapest available upgrade in accuracy-per-token until the total budget is exhausted. Set the budget to the total cost of the middle cap, so the comparison is against every single cap including the best one affordable at that same spend.

**Intervals.** Resample prompts with replacement, 2000 repetitions, and recompute *the whole allocation and the best single cap* in each repetition. The interval is therefore on the decision rule, not on a fixed partition — a partition chosen once and then bootstrapped would understate the uncertainty.

Table S14: Per-prompt allocation against every shared cap (full form of the result quoted in Appendix S20).

| setting | best single cap | allocated | Δ accuracy (95% CI) | Δ tokens |
|---|---|---|---|---|
| Qwen3-4B-Inst / MATH-500 (4/8/12k) | 0.918 | **0.937** | +0.019 [+0.009, +0.022] | −6.2% |
| Qwen3.5-4B / MATH-500 (4/8/16k) | 0.858 | **0.901** | +0.042 [+0.032, +0.050] | −2.5% |
| Qwen3.5-4B / MATH dev (4/8/16k) | 0.851 | **0.890** | +0.038 [+0.030, +0.046] | −4.6% |
| Qwen3-4B-Inst / MATH dev (1/2/3k) | 0.771 | **0.817** | +0.046 [+0.036, +0.055] | −9.9% |

Four settings, four strict Pareto improvements, four intervals excluding zero. The largest gain is at the low-cap end, which is what Proposition 4 predicts: when most prompts are budget-starved, the dispersion in marginal returns is largest.

**What this does and does not establish.** It establishes that, at equal or lower spend, allocating budget per prompt beats every shared cap on the ladders we have, offline and without training. It does *not* establish that a trained policy would behave the same way, because $p_i(B)$ and $k_i(B)$ are re-measured after every update in a real run and the allocation would have to be refreshed. We report the offline result as the constructive consequence of the measurement claim, and flag the online version as open.

## S22 A BUDGET-INVARIANT DIFFICULTY COORDINATE

Because $1 - \tau_B$ is the CDF in $B$ of the policy's response-length distribution, sweeping the budget recovers, for every prompt, the budget at which that prompt's solution completes. We call this the *required budget*. Unlike the pass rate it is defined without reference to any particular $B$, it is monotone in difficulty by construction, and it makes the "hard" stratum a statement about solution length rather than about an accident of the instrument.

We state it as a coordinate rather than a result because we have not built a curriculum on top of it and compared one. What we can say is that the quantities it needs — completion and generated length, per prompt — are already logged by the pipeline, so the coordinate is available at no cost, and that Section 4 establishes the premise: the ladder is a CDF, so different prompts do sit at different quantiles of it.

## S23 THE FOUR-LINE AUDIT PROTOCOL

Each item is addable to an existing pipeline without a new run, and each is a direct consequence of an argument in the main text. Algorithm 1 is the instrument: it consumes three per-batch scalars that every stack in Table S2 already logs and returns the salvage value, with the identity check that guards against mixing scalars from different sample sets. Algorithm 2 is the reward path into which every treatment in Table 1 is a substitution; the default grader is the case in which none of its three switches is set.

**Algorithm 1** Salvage read-out from the training ledger (Eq. 1)

**Require:** per-batch logged scalars $\bar{r}$ (mean reward), $\bar{r}_{\text{strict}}$ (mean completed-and-correct), $\tau$ (truncated fraction), all over the same $N$ rollouts
**Ensure:** salvage value $b^\star$, or UNDEFINED if no rollout truncated

**if** $\tau = 0$ **then**
  **return** UNDEFINED ▷ no truncated set; Eq. (1) has no denominator
$a \leftarrow \bar{r}_{\text{strict}}/(1-\tau)$ ▷ correctness among completed rollouts
$b^\star \leftarrow (\bar{r} - \bar{r}_{\text{strict}})/\tau$ ▷ mean reward among truncated rollouts
**assert** $0 \le b^\star \le 1$ and $|\bar{r} - (1-\tau)a - \tau b^\star| < \epsilon$ ▷ fails iff the three scalars were not computed on the same samples
**return** $b^\star$ ▷ per stratum: apply to that stratum's three scalars

**Algorithm 2** Reward and loss-mask path for one rollout, with the three switches of Table 1

**Require:** rollout $y$ of length $L$, cap $B$, verifier $v(\cdot) \in \{0,1\}$ on retained text; switches: installed constant $b_{\text{set}} \in [0,1] \cup \{\perp\}$, mask flag $m \in \{0,1\}$, soft penalty $\lambda \ge 0$ with onset $B_0 < B$

$T \leftarrow \mathbf{1}[L \ge B]$ ▷ truncation status is available here in every inspected stack
**if** $T = 0$ **then**
  $r \leftarrow v(y)$ ▷ completed: graded as usual
**else if** $b_{\text{set}} = \perp$ **then**
  $r \leftarrow v(y)$ ▷ **default**: prefix graded, $r$ varies within $T$; $b(\theta) = \mathbb{E}[v \mid T]$
**else**
  $r \leftarrow b_{\text{set}}$ ▷ installed constant: within-$T$ variance is zero; $b_{\text{set}} = 0$ is DLER
$r \leftarrow r - \lambda \max(0, L - B_0)$ ▷ soft penalty (Yu et al., 2025): applied on top, does not remove within-$T$ variance
**emit** $(r,\ \text{in_loss} = 1 - mT)$ ▷ mask: $r$ still enters the group mean; $y$'s tokens leave the loss

1. **Report $\tau$ beside every accuracy number.** A pass rate is a product, and $1 - \tau$ is a property of the instrument rather than of the problem (Section 4).
2. **Compare arms at a matched evaluation budget.** In our setup this alone moved one checkpoint by 27.8 pp — larger than most reported RLVR gains.
3. **Choose $b$ deliberately and report it.** The default is not $b = 0$ but $b = b(\theta)$. Fixing $b$ controls reward variation, not training stability; the fixed $b = 0.88$ follow-up loses completion quality (Section 5).
4. **Allocate budget per prompt where the ladder is steep.** It beat every shared cap we tested, at lower token cost (Appendix S20).

## S24 WHICH METRIC WE REPORT, AND WHY

A rule-based verifier returns a scalar for the text it receives. A truncated response can already contain an accepted answer, so default prefix grading need not return zero. Under an installed reward rule, truncated rollouts instead receive the specified constant $b$. More generally $b$ is their conditional mean, and the logged mean reward satisfies:

$$\overline{r} \;=\; (1-\tau)\,a \;+\; \tau\,b, \tag{8}$$

with $a$ the correctness rate among completed rollouts. We verified this identity against the ledger rather than assuming it: on the two installed-$b$ arms of the original sweep ($b^{\text{set}} = 0.5$ and $1.0$ at cap $12288$; these are *not* the $\beta$ arms of Appendix S14, which never produced rollouts) and for both the global and the hard-stratum scopes, the logged `accuracy` equals the right-hand side of Eq. (8) to within $10^{-5}$. The per-update exactness of Eq. (1) itself is checked on all $25$ installed-$b$ arms in Table S7.

The consequence is that `accuracy` is *not comparable across arms that differ in* $b$. The $\beta = 1.0$ arm reports hard-stratum accuracy $1.000$ while its strict accuracy is exactly $0$: every rollout runs to the cap, every one is scored correct, and the policy has learned nothing except not to stop. Reading that column as a performance number would invert the conclusion. We therefore report, everywhere in this paper,

- `strict_accuracy`: the fraction of rollouts that both completed and were correct. It is $b$-free by definition, since a truncated rollout can never contribute to it.
- $(1-\tau)\,a$: the completed-and-correct mass on a stratum. This is a second, independently computed path to the same quantity, and we checked that the two agree on every arm (they are separate code paths in the ledger).

We report $\tau$ beside every number so that a reader can reconstruct either column from the other. Where a published number of ours is quoted elsewhere under a different convention, it is the strict-accuracy value that should be used.

## S25 EXPERIMENTAL PROTOCOL

**Policy and data.** Primary policy Qwen3-4B-Instruct-2507, served and trained with a standard RLVR stack (Zheng et al., 2023; Sheng et al., 2024). Development set: $750$ mathematical prompts, $8$ samples per prompt per evaluation ($6{,}000$ rollouts), strata fixed once ($75$ hard, $40$ mid, $635$ easy) and reused by every experiment. Held-out prompt set: MATH-500, $500$ prompts $\times 8$, hard stratum $44$ prompts ($352$ rollouts). Negative control: GSM8K test split, $1319$ prompts $\times 8$ — the same $n$ that the bootstrap interval in Section 4 and Table S14 are computed on. An earlier version of this appendix reported $659$ here, which was a transcription error and is corrected.

**Optimisation.** GRPO with KL coefficient $0$ and no standard-deviation normalisation (Dr. GRPO), group size $16$ for training rollouts and $8$ per prompt for evaluation. evaluation budget is bound to the training budget; matched-budget scores are obtained by separate evaluation-only runs that load a checkpoint and set the budget independently. Every arm records a token ledger per update and per evaluation, and every number in this paper is read from those ledgers rather than from a summary file.

**Compute.** The primary arms plus ladders and controls generated $438$ M tokens in total on $8\times$ H20 ($96$ GB) nodes. The most expensive single arm is the one that collapses (E12, $12.45$ M tokens in its final update); the cheapest arm reaching the same budget stably is M43 ($1.91$ M). Across the whole study, the arms that fail are consistently the most expensive ones, which we note because it is the opposite of the usual accuracy–compute trade-off.

**Reproducibility artefacts.** Run directories, launch manifests (which record the source-code hash of every launch) and token ledgers are retained for all arms listed in Appendix S26, so every cell of every table in this paper can be recomputed from disk.

## S26 EVERY ARM WE RAN

This table exists so a reader can check for selective reporting; it lists all training arms whose runs completed, including the ones that do not support our claims, with numbers and no commentary. Each arm is read at its own final update and its own training budget.

| arm | policy | budget / schedule | censoring | upd | strict | hard $q$ | $\tau$ |
|---|---|---|---|---|---|---|---|
| A | Qwen3-4B-Inst | 4096 | default | 16 | 0.861 | 0.162 | 0.109 |
| E5 | Qwen3-4B-Inst | 4096 | mask | 10 | 0.886 | 0.190 | 0.077 |
| B | Qwen3-4B-Inst | 4096 + extra | default | 16 | 0.871 | 0.343 | 0.030 |
| B-s43 | Qwen3-4B-Inst | 4096 + extra | default | 16 | 0.864 | 0.315 | 0.035 |
| C | Qwen3-4B-Inst | 4096 + targeted | default | 16 | 0.873 | 0.285 | 0.051 |
| E12 | Qwen3-4B-Inst | 12288 held | default | 11 | 0.045 | 0.432 | 0.954 |
| M42 | Qwen3-4B-Inst | 12288 held | mask | 16 | 0.888 | 0.493 | 0.089 |
| M43 | Qwen3-4B-Inst | 12288 held | mask | 16 | 0.927 | 0.502 | 0.030 |
| E1 | Qwen3-4B-Inst | 4096→8192→12288 | default | 16 | 0.911 | 0.425 | 0.008 |
| E1-s43 | Qwen3-4B-Inst | same | default | 16 | 0.878 | 0.400 | 0.005 |
| E2 | Qwen3-4B-Inst | 4096×2→12288 | default | 16 | 0.914 | 0.428 | 0.007 |
| E2-s43 | Qwen3-4B-Inst | same | default | 16 | 0.926 | 0.492 | 0.017 |
| E3 | Qwen3-4B-Inst | 12288×4→4096 | default | 16 | 0.883 | 0.248 | 0.077 |
| P1m | Qwen3-4B-Inst | 4096×2→12288 | mask | 16 | 0.922 | 0.547 | 0.055 |
| R1D7B | R1-Distill-Qwen-7B | 12288 held | default | 16 | 0.902 | 0.402 | 0.038 |
| R1D7B-m | R1-Distill-Qwen-7B | 12288 held | mask | 16 | 0.867 | 0.330 | 0.060 |
| q35i-noop | Qwen3.5-4B-Inst | 4096 | default | 8 | 0.815 | 0.290 | 0.059 |
| q35i-mask | Qwen3.5-4B-Inst | 4096 | mask | 8 | 0.683 | 0.292 | 0.067 |
| q35c-noop | Qwen3.5-4B-Base | 4096 | default | 15 | 0.779 | 0.163 | 0.127 |
| q35c-mask | Qwen3.5-4B-Base | 4096 | mask | 15 | 0.622 | 0.060 | 0.331 |
| ll-noop | Llama-3.2-3B-Inst | 4096 | default | 8 | 0.757 | 0.513 | 0.009 |
| ll-mask | Llama-3.2-3B-Inst | 4096 | mask | 8 | 0.733 | 0.515 | 0.013 |

Two rows deserve a flag. `q35i-mask` and `ll-mask` lose 13.2 pp and 2.4 pp on the *full* set while being flat on the hard stratum, so the decline sits in the easy region; we have no explanation and do not use these numbers. `q35c-mask` is the base-policy arm of the switch test in Appendix S17 and is the only arm in which masking is harmful on the hard stratum by more than 2 pp.

Table S15: **Matched-budget audit (main-text Table).** Every arm scored at one evaluation budget $B = 12288$ (6,000 rollouts; hard $n = 600$). $b$ is implied by Eq. (3) so every row can be checked against the identity.

| arm | seed | hard stratum at $B = 12288$ | | | | own budget |
|---|---|---|---|---|---|---|
| | | $q$ | $1-\tau$ | $a$ | $b$ | (as reported) |
| A | 42 | 0.440 | 0.728 | 0.556 | 0.13 | 0.162 |
| E1 | 42 | 0.432 | 0.942 | 0.453 | 0.09 | 0.425 |
| E2 | 42 | 0.408 | 0.935 | 0.426 | 0.15 | 0.428 |
| M42 | 42 | 0.493 | 0.497 | 0.742 | 0.25 | 0.493 |
| M43 | 43 | 0.508 | 0.743 | 0.639 | 0.13 | 0.502 |
| P1m | 42 | 0.547 | 0.625 | 0.768 | 0.18 | 0.547 |
| P1m | 43 | 0.563 | 0.892 | 0.619 | 0.11 | 0.563 |

## S27 CLAIMS LEDGER

The table below is the reference we used for every claim in the paper: what we assert, what supports it, and the strongest form we believe the evidence licenses. Nothing in the main text is stated more strongly than the third column allows.

## S28 RUNS THAT DID NOT PRODUCE A USABLE NUMBER

Reported in full because a ledger that lists only confirming runs is not a ledger.

**Launcher revisions.** Every launch manifest records a hash of the launcher script. Three revisions appear in this paper. The two saturating default-grader arms of Table S4 ran the two earlier ones; every multi-seed arm, every installed-$b$ arm and every matched pair ran the current one. We diffed the three.

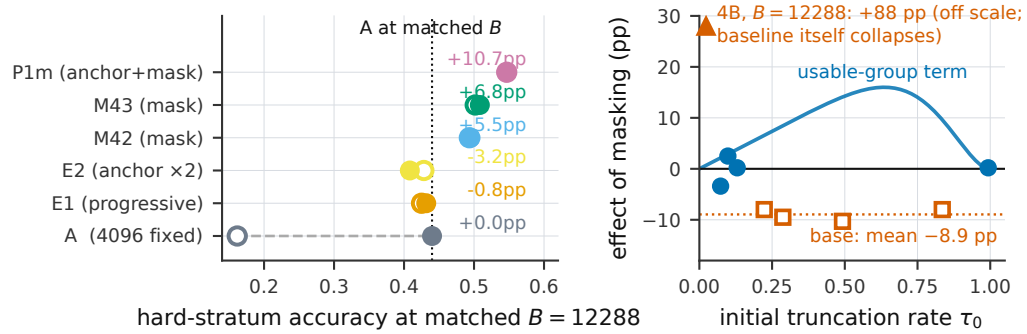



Figure S9: **Left:** hollow markers are each arm at its own training budget, filled markers the same checkpoint re-scored at $B = 12288$; the dotted rule is arm A at the matched budget, so each $\pm$pp is read against it. Seed spread at the matched budget is 1.3 pp for the two mask arms, 2.2 pp for the progressive pair and 8.8 pp for the anchor pair. **Right:** the effect of masking against $\tau_0$. Blue points are self-terminating policies; the curve is the usable-group term of Eq. (7), shown as a necessary condition and not as a fitted gain. Open squares are policies that cannot terminate on their own: they sit at $-8.0$ to $-10.3$ pp (mean $-8.9$ pp, the dotted rule), with no trend in $\tau_0$ across $[0.22, 0.84]$.

The later revisions add a switch for GRPO standard-deviation normalisation (default off, matching the earlier hard-coded flag), a switch for an alternative masking filter (off unless requested), an option to skip saving optimiser state, and a configurable model-arguments file. None of these changes the listed training flags for the default-grader Qwen3-4B configuration. This code comparison does not establish identical runtime behaviour or rule out all revision-related effects. The later seed-42 default run completed whereas the earlier case saturated; the cases are not counted as independent replications under one frozen executable.

**Why the per-update regression is descriptive only.** The regression of $\Delta\tau$ on $\tau(1-\tau)$ reported in Section 5 is reported as a summary of trajectory shape, not as evidence for the mechanism, for two reasons that a reader is entitled to press. First, by the masked-update expression of Appendix S2.1, $g_{\text{mask}} = (1-\tau)\nabla a - \tau(a-b)\nabla\tau$, the status weights differ between recentered masking and our baseline-preserving token mask, so a coefficient fitted on masked arms does not isolate the unmasked status term. Second, and more seriously, there is a circularity: $\Delta\tau$ is by construction positive on arms whose truncation grows and near zero on arms where it does not, so any regressor correlated with the level of $\tau$ will inherit the sign of "did this run collapse". We therefore use the regression to describe *how* the collapse proceeds — smoothly in the $b = 1$ arm ($R^2 = 0.75$) against a single-update jump in the default-grader arm ($R^2 = 0.11$) — and rely on the degeneracy argument of Section 3, which is algebraic, for *why*.

One clarification on run bookkeeping. The $b = 0.25$ cell *does* have a complete run of sixteen updates; the two rows above are failed *launch attempts* on particular machines, and the completed run is the one reported in Appendix S11. The $b = 0.00$ and $b = 0.75$ rows likewise refer to an early mid-run snapshot, not to the runs we report: on disk both have sixteen training ledgers and a final evaluation at rollout 15, and those completed ledgers are what Table S9 uses.

The one arm we genuinely cannot complete is the collapsed run `escape_E_budget12288`, which stops at update eleven: it collapsed, so there is nothing further to read. Its launch manifest names no custom reward function, so it was graded by the default prefix grader, and its ledger $b^\star$ reached 0.933 (Table S4); it is not a $b = 0$ arm and does not enter the installed-$b$ sweep. The installed $b = 0$ arms at this cap (`truncbeta0.00` and `Beta00_s45`) both produced sixteen training rollouts. Their final *training-batch* strict rewards are 0.9043 and 0.9102; their final *held-out full-set* strict accuracies are 0.9255 and 0.9145, respectively. These use different prompts and denominators, not inconsistent measurements.

## S29 Launch configuration and reproducibility

Every arm in this paper is launched from a script that writes a manifest recording the resolved configuration together with content hashes of the artifacts it consumes. The manifest is what we used

| claim | evidence | strongest licensed form |
|---|---|---|
| Observed difficulty depends on the budget | zero-training ladders, six budgets × two repeats, five policies, three prompt sets; GSM8K negative control | **strong**: established |
| Evaluation at an unmatched budget misleads | zero-training hard-stratum pass rate moves by $44.7$ pp ($0.011 \rightarrow 0.458$) when only the evaluation budget changes; a *trained* checkpoint at matched evaluation caps moves by $27.8$ pp ($0.162 \rightarrow 0.440$) | **strong**: established |
| Fixed rewards have different training consequences | original $b \in \{0, 0.25, 0.5, 0.75, 1\}$ sweeps and fixed-$0.5$ seed cohort; five same-seed $b = 0.88$ pairs at cap $1024$ | **observed**: both $b = 1$ arms saturate; original sampled interior arms do not. The five $0.88$ arms have hard truncation $0.92$–$1.00$ and lose 12–41 pp strict accuracy against their own default arms (mean $-24.8$, interval $[-41.1, -8.5]$). No universal safe interval or optimal interior value is established |
| Reward encoding and within-truncation credit | algebraic: $0 < b < 1$ distinguishes the three fixed reward levels; fixing $b$ removes reward variation within truncated outputs | **exact** for the algebra, not a stability theorem. The $0.88$ follow-up shows that three distinct levels need not preserve completion |
| Answer acceptance can separate from completion | reloaded checkpoints (Table 2); at E12 rollout 10 all 848 easy responses are accepted while 11 complete, on one stratum and batch | **established as possible**: two case runs plus reload probes; frequency across policies and tasks is not estimated |
| Collapse does not require budget widening | both default-grader arms that collapsed held their cap from the first update (12288 and 4096); in each, $b^\star$ drifted above one half before saturation | **moderate**: one policy, two arms. $b^\star$ is not a classifier: a high reading alone does not predict collapse, and we state no threshold |
| Salvage is measurable per stratum, not only per batch | $b_x$ returns the installed constant to $10^{-14}$ in 712 stratum-update cells over 26 arms and five families; all three strata rise together in the identified A43 pair; $432/830$ low-$\tau$ default cells have $b_x > 0$, from $77\%$ (Qwen3-4B-Instruct) to $5\%$ (R1-Distill-7B) (Appendix S8) | **exact** for the identity within a stratum; **observed** for the joint rise. Batch-level movement is not decomposed where a stratum has no truncated samples, and no within-prompt causal mechanism is identified |
| $\tau_0$ alone does not order which policies survive | the 7B arm that survived started at a higher $\tau_0$ than a 4B arm that collapsed (Appendix S16) | **counterexample only**: rules out $\tau_0$ as a sufficient predictor |
| Masking has context-dependent effects | twelve matched cells and historical near-saturation rescues | **scoped**: nine of twelve main-table cells lose accuracy, interval $[-8.9, -1.6]$ pp; the three gains are $+0.9$, $+1.5$ and $+1.9$ pp. Neither benefit nor harm is universal |
| Per-prompt allocation beats every shared cap | four offline settings, bootstrap intervals on the whole decision rule | **scoped**: strictly offline, iteration-0 ladders; the online version is open |

| run | what happened | status |
|---|---|---|
| `truncbeta0.25` launch attempt (node A) | launcher exited non-zero before the first training rollout | failed, superseded |
| `truncbeta0.25` launch attempt (node B) | second attempt; log contains only the chain-start line | failed, superseded |
| `truncbeta0.00`, `truncbeta0.75` | first snapshot of these two arms was taken mid-run (10 and 4 of 16 updates); both later completed all 16 and the completed ledgers are what Table S9 reports | superseded by the completed runs |
| $\tau$-matched 2×2: instruct side | preregistered prediction was $\Delta \geq +8$ pp; observed $-1.5$ pp | **prediction failed**, reported |
| `p3_beta` | launcher failure; superseded by `p3b_beta_fix` | failed, rerun succeeded |
| `Prog_gradual_s43` & ladder cells | endpoint mismatch or incomplete evaluation | excluded from tables |

to verify that the $b$ sweep varies one variable: for the $\beta$ arms and the $b = 0$ arm they are matched on `schedule_treatment` (`A_uniform_no_extra`), `rollout_max_response_len` (12288), `rollout_max_prompt_len` (2048), rollout seed (42), schedule seed (42), and the schedule manifest's `sha256`; the one field that differs, `rollout_max_context_len`, takes the value $14336 = 2048 + 12288$ and is therefore not binding.

| quantity | value |
|---|---|
| policy | Qwen3-4B-Instruct-2507 |
| training rollouts per prompt | 16 |
| evaluation samples per prompt | 8 |
| optimizer | GRPO, no standard-deviation normalisation (Dr. GRPO) |
| KL | on-policy distillation disabled, KL coefficient 0 |
| response cap | 12288 unless stated; prompt cap 2048 |
| sampling | temperature 0.6, top-$p$ 0.95 |
| rollout batch / global batch | 64 / 1024 |
| verifier | DeepScaleR-style programmatic checker |
| prompt pool | fixed once, reused by every arm in the paper |
| strides | hard / mid / easy = 75 / 40 / 635 prompts, fixed before any budget experiment |

The artifacts themselves — run directories, launch manifests and per-rollout token ledgers — are retained. Every number in the paper is recomputable from the ledgers without rerunning training, and the two headline metrics are available as separate code paths that we cross-checked against each other (Sec. S24).

## S30 Extended limitations

*Where the evidence sits.* The hard stratum is 75 prompts with 8 samples each (600 rollouts), a standard error of roughly 4–6 pp on the stratum and 1–3 pp on the full 750-prompt set; seed-to-seed differences on it reach 8.8 pp. At a matched budget the three allocation arms A (0.440), E1 (0.432) and E2 (0.408) are not separated at that resolution, and the gains of M42 (+5.5), M43 (+6.8) and P1m (+10.7) are each about 1.5–2 standard errors, so the masking comparison is read as a direction reproduced by three arms built two independent ways rather than as a magnitude. The preregistered fork fixes six seeds in advance: four of six lie in the predicted direction with a 95% interval that contains zero, and the secondary readout splits three and three. Its control arms sit at $\tau_{15} \in [0.010, 0.111]$, below the collapse regime treated in Section 5, so the fork addresses whether the term moves truncation at all, not how large the effect is once the cap binds throughout.

*Two choices that set the reading.* Across arms we compare the unconditional hard-stratum pass rate, because the arms differ in completion rate by up to 32 pp and their conditional subsets are therefore differently selected; $a$ is reported in Table S15 as a decomposition, not as a cross-arm contrast. The gain from masking and the gain from partial credit are for the same reason neither additive nor interchangeable — removal changes which comparisons a group supports, re-scoring changes a weight. E5 is read at update 7 of 10 against arm A at 15 of 16, which is the conservative direction: less training favours the arm being compared against, so the +2.5 pp masking effect at $B = 4096$ in Figure S9 is a lower bound and enters no argument that depends on its size.