MATCHED COMPUTE IS NOT MATCHED SIGNAL: WHERE THE GRADIENT GOES IN GROUP-RELATIVE REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning from verifiable rewards is usually preceded by a supervised warm-up, and runs are compared at matched compute. But group-relative methods such as GRPO learn only from prompts whose sampled answers disagree: a group of rollouts that all earn the same reward contributes exactly zero gradient, however much it cost to generate. Two runs with the same budget can therefore receive very different amounts of learning signal. Prior work uses this to filter prompts during training; we make it exact and extend it to policies and prompt sets. We derive the gradient signal a group carries under GRPO, Dr.GRPO and RLOO and average it over the prompts. The result is a signal efficiency $\eta = c_G[a(1-a) - \mathrm{Var}_x(p)]$, where a is mean accuracy, p is a prompt's success probability and c_G depends only on the group size. A more accurate policy can therefore carry less signal. Across 32 runs given identical budgets, the generations that actually carried a gradient differed by a factor of 8.8. A warm-up trained on the problems its reinforcement-learning phase will use silences them: after 27 epochs, solution distillation solves 98.6% of them and η falls from 0.600 to 0.065. The same accounting explains why a base model barely improves: its gradient is real, but almost all of it comes from the format reward rather than from being right. Choosing prompts by $p(1-p)$, from a single sampling pass before training, beats a random set of the same size and budget on all 20 runs across four settings, a search task and GSM8K at 3B to 14B, by 2.2 to 7.8 points of held-out accuracy. The lead survives at twice the training horizon, though its signal advantage fades as the chosen prompts are solved, and reverses on every seed in the one setting where the random control climbs far enough toward $p = 1/2$. Two plausible variants of the rule each fail in some setting, by overshooting an optimum the accounting locates. What η does not do is rank warm-ups or predict how much a run will gain: it prices prompts, not checkpoints. Runs should therefore be compared by the generations that carry a gradient, not the generations spent. Code: anonymous.4open.science/r/signal-efficiency-rl-D5C1.

1 INTRODUCTION

Reinforcement learning from verifiable rewards has become a standard final stage for reasoning models, and it is almost always preceded by a supervised warm-up. A growing literature compares warm-ups at matched compute: fix the reinforcement-learning budget, vary what was distilled, and see which starting point ends higher (Gandhi et al., 2025; Cai et al., 2025; Li et al., 2025). Two assumptions are usually left implicit, that a warm-up should be as accurate as possible and that fixing the number of generations fixes the comparison.

Neither survives contact with how group-relative methods compute a gradient. GRPO and its relatives form the baseline for each completion from its own group of rollouts (DeepSeek-AI et al., 2025; Liu et al., 2025; Ahmadian et al., 2024), so a group whose completions earn the same reward has every advantage exactly zero: the prompt contributes no policy gradient, however many rollouts it consumed (fig. 1a). Matched compute therefore does not mean matched learning, and accuracy is a poor proxy for what a warm-up supplies, because a policy that is uniformly right or uniformly wrong is silent either way. That a single prompt's learnability tracks its reward variance is known

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.

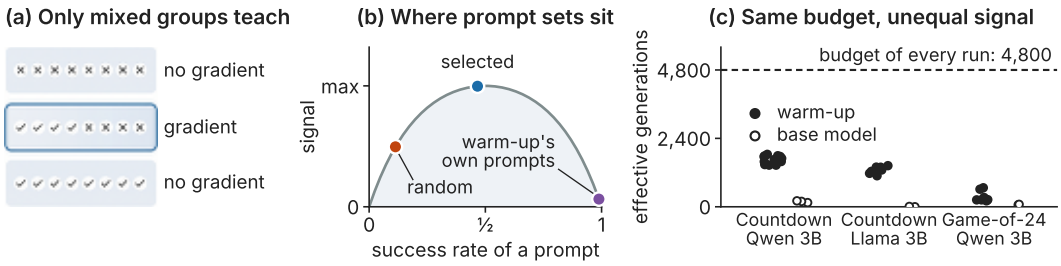


Figure 1: **Matched compute is not matched signal.** (a) A group whose $G = 8$ answers all agree gives exactly zero gradient (proposition 1); AI-generated schematic. (b) Signal against success rate, with three Countdown prompt sets at their measured starting pass rates. (c) Effective generations ($\eta \times 4800$) of every matched-compute run: the 32 warm-up runs (filled) span 209 to 1839 (8.8×), mostly between tasks, and in every setting sit above all 7 base-model runs (hollow, 0 to 208). (b) and (c) are plotted from stored run files.

and is used to filter prompts during training (Yu et al., 2025; Zheng et al., 2025; Jiang et al., 2025; Bae et al., 2026). What is missing is the aggregate a whole *policy* supplies over a prompt distribution, which one must score to choose a set before any rollout is spent.

We answer it exactly. For a binary verifier reward and a group of G rollouts of which k succeed, the total absolute advantage is $2\sqrt{k(G-k)}$ under GRPO and $2k(G-k)/(G-1)$ under RLOO, and averaging over prompts gives a policy-level signal efficiency $\eta = c_G[a(1-a) - \mathrm{Var}_x(p)]$ with $c_G = 4(G-1)/G$. The accuracy a warm-up is selected for therefore competes with the difficulty spread that supplies its signal, and the same accounting prices the group size (fig. 1b).

The accounting turns on one variable a budget is spent on, the prompt set, and we move it both ways with the weights held fixed: choosing prompts for signal, and letting a warm-up solve them in advance. One equation prices both; between them the two interventions move η by a factor of 57.

It also shows that matched compute is not matched. Across 32 runs with identical nominal budgets, the generations that carry a gradient range over a factor of 8.8 (fig. 1c), and a base-model arm receives almost none, so its flat curve is not an inability to learn but a budget it never received. What η does *not* do is rank warm-ups or predict how much a run will gain, which we establish as carefully as the positive results. Verifier-backed search-control distillation is the instrument for varying η at a fixed problem set, prompt and answer format, not the contribution.

Contributions. **An exact accounting.** The advantage mass of a group in closed form for each estimator (proposition 2), aggregated into a policy-level quantity that splits into a part accuracy fixes and a part it does not (theorem 1), with the group-size trade-off this forces (proposition 3) and a sandwich for the standard-deviation-normalized estimator used in practice (eq. (8)). All verified against brute-force computation. **Three consequences that contradict standard practice, each tested.** Accuracy and signal compete, so a warm-up trained on its reinforcement-learning problems destroys the gradient it was meant to supply: η falls to a third at 3B and a sixteenth at 7B, deepening with capacity and saturating by 7B (section 5.2). A shaping term's weight *cancels* on exactly the groups where correctness is silent (eq. (8)), so a base model's gradient is real, full-magnitude and almost entirely about format (section 5.3). And a chosen prompt set spends itself: its signal advantage must decay and eventually reverse (proposition 4), and reverses on every seed where the control climbs far enough. **A rule that wins, and two that fail.** Selecting prompts by $p(1-p)$ from one pre-training pass beats an equal-size random set at identical budget on *all twenty* runs across four settings (3B to 14B, a search task and GSM8K), and keeps a narrowed lead at twice the horizon. Two plausible variants each reverse somewhere; the accounting says why: they overshoot an optimum just below $p = 1/2$. We price it end to end, including the budget at which it stops paying.

2 Related Work

Our setting is reinforcement learning from verifiable rewards with a group-relative advantage: GRPO (Shao et al., 2024), scaled for reasoning by DeepSeek-AI et al. (2025); the bias-corrected

Dr.GRPO (Liu et al., 2025); the leave-one-out RLOO (Ahmadian et al., 2024); and open reproductions (Hu et al., 2025; Chen et al., 2025b; Gulcehre et al., 2023; Kazemnejad et al., 2024).

Learnability of a prompt is known; learnability of a policy is the open question. That a group whose rollouts agree contributes no gradient is established, and so is the quantity that follows. DAPO oversamples and discards such groups during training (Yu et al., 2025); GRESO predicts which prompts will be degenerate and skips them before rollout (Zheng et al., 2025); VCRL builds a curriculum from the group reward variance (Jiang et al., 2025); and Bae et al. (2026) derive a reverse-KL lower bound proportional to $p(1-p)$, maximized at $p = 1/2$, accounting for why intermediate-difficulty prompts are the informative ones. We take that as established and ask what this line does not: all of it scores a *prompt*, to decide what to train on next, whereas choosing a warm-up is choosing a *policy*, before any prompt is seen. Aggregating per-prompt learnability over the prompt distribution is not bookkeeping, because it produces $\eta = c_G[a(1-a) - \mathrm{Var}_x(p)]$, in which the accuracy a warm-up is selected for and the spread that supplies its signal are competing terms. Our additions are therefore the exact per-group mass for each estimator rather than a bound or proxy (proposition 2), that aggregation and its consequence for warm-up choice (theorem 1 and corollary 1), what it implies for the group size at fixed budget (proposition 3), and a measurement study of what the quantity does and does not predict. That line is active (Shi et al., 2025; Sun et al., 2025; Zhu et al., 2026) and chooses prompts *during* training. Closest in kind is Rajaraman et al. (2026), who prove that letting a model's own performance choose its problems improves both supervised and reinforcement fine-tuning. That result is asymptotic and adaptive; ours is an exact per-group accounting and a *static* rule fixed by one pass beforehand, weaker in principle and cheaper in practice, and we test it rather than bound it. Two concurrent preprints overlap with parts of this work and we mark the boundary rather than leave it to a reader. Sudalairaj et al. (2026) likewise profile a pool once under the initial policy and use the resulting success rates to filter and schedule, reporting a threefold reduction in total compute; they set the group size to the inverse of the success rate, which is the opposite of what proposition 3 prescribes. We ran that ladder and it does not settle the disagreement: the mass accounting is confirmed while outcomes stay flat across $G \in \{2, 4, 8, 16\}$ (appendix H). Su et al. (2026) run a controlled ablation of supervised and reinforcement-learning data overlap and reach the same qualitative conclusion as section 5.2 on one model and one domain. Neither derives the accounting, and neither reports the scale dependence or the falsifying variants that make our version an explanation rather than a recipe.

Entropy. A parallel line studies token-level policy entropy and its early collapse (Cui et al., 2025). That is a property of the distribution over strings, whereas signal efficiency is a property of the distribution over verifier *outcomes*, and only the latter reaches the group baseline; appendix G.4 finds the two oppositely ordered across our runs. The remaining context, on what supervision installs before reinforcement learning, on imitating search, and on geometry dependence, is in appendix D.

3 Group Signal Efficiency

3.1 Setup and Vanishing Groups

Group-relative policy-gradient methods draw G completions $o_1, \ldots, o_G \sim \pi_\theta(\cdot \mid x)$ per prompt, score them with a reward $r_i = R(x, o_i)$, and compare each with a baseline built from its group:

$$\hat{A}_i^{\mathrm{GRPO}} = \frac{r_i - \bar{r}}{\sigma_r + \epsilon}, \qquad \hat{A}_i^{\mathrm{Dr.GRPO}} = r_i - \bar{r}, \qquad \hat{A}_i^{\mathrm{RLOO}} = r_i - \frac{1}{G-1}\sum_{j \neq i} r_j, \tag{1}$$

with $\bar{r}, \sigma_r$ the group mean and standard deviation (DeepSeek-AI et al., 2025; Liu et al., 2025; Ahmadian et al., 2024). Prompt x contributes

$$g(x) \;=\; \frac{1}{G}\sum_{i=1}^{G} \hat{A}_i \, \nabla_\theta \log \pi_\theta(o_i \mid x) \tag{2}$$

to the policy gradient. We take $r_i \in \{0, 1\}$ from an exact verifier, write $k = \sum_i r_i$, and let $p_\pi(x)$ be the per-prompt success probability, so $k \sim \mathrm{Binomial}(G, p_\pi(x))$.

Proposition 1 (Vanishing groups). *If $r_1 = \cdots = r_G$ then $\hat{A}_i = 0$ for every i under all three estimators, hence $g(x) = 0$ exactly, independently of θ and of the common reward value.*

A prompt the policy always solves, and one it never solves, therefore contribute nothing, however many rollouts are spent on them. It motivates the dynamic sampling of Yu et al. (2025); our interest is what it implies for choosing the warm-up. Deferred proofs are in appendix A.

How much signal a group carries.

Proposition 2 (Advantage mass)**.** *With binary rewards and $0 < k < G$, the total absolute advantage $\Psi_G(k) = \sum_i |\hat{A}_i|$ is, as $\epsilon \to 0$,*

$$\Psi_G^{\mathrm{GRPO}}(k) = 2\sqrt{k(G-k)}, \qquad \Psi_G^{\mathrm{Dr.GRPO}}(k) = \frac{2k(G-k)}{G}, \qquad \Psi_G^{\mathrm{RLOO}}(k) = \frac{2k(G-k)}{G-1}. \tag{3}$$

Each is strictly increasing in $k(G-k)$, vanishes exactly at $k \in \{0, G\}$, and is maximized at $k = G/2$.

Since $\|g(x)\| \le \frac{1}{G}\sum_i |\hat{A}_i| \max_i \|\nabla_\theta \log \pi_\theta(o_i \mid x)\|$, the normalized $\Psi_G(k)/\Psi_G^{\max} \in [0,1]$ bounds the gradient a group can produce and is zero exactly when the group is degenerate. This concerns available signal, not learning: large mass does not make the gradient useful, and section 5.3 tests whether it is also predictive. Used later: $\hat{A}^{\mathrm{GRPO}}$ is invariant to $r \mapsto ar + b$ for $a > 0$, so a reward component constant within a group contributes nothing while one that varies does.

3.2 The expected signal is the outcome variance

Define the *signal efficiency* of a policy as the expected normalized advantage mass,

$$\eta_G(\pi) \;=\; \mathbb{E}_{x\sim\mathcal{D}}\, \mathbb{E}_{k\sim\mathrm{Bin}(G,p_\pi(x))}\left[\frac{\Psi_G(k)}{\Psi_G^{\max}}\right] \;\in\; [0,1], \tag{4}$$

so a budget of N nominal generations delivers $\eta_G(\pi)\,N$ generations' worth of gradient. Arms given equal N match in nominal compute but not effective compute; the warm-up sets the conversion rate.

Theorem 1 (Signal is outcome variance)**.** *Normalize by $\Psi_G^{\max} = \Psi_G(G/2)$, the mass at the continuous maximizer of $k(G-k)$, which for even G, the case of every experiment here, is the largest an attainable group carries (appendix A records why the attainable maximum is the wrong normalizer at odd G). Then for the mean-baseline estimators, every $G \ge 2$ and every $p \in [0,1]$,*

$$\mathbb{E}_{k\sim\mathrm{Bin}(G,p)}\left[\frac{\Psi_G(k)}{\Psi_G^{\max}}\right] \;=\; c_G\, p(1-p), \qquad c_G = \frac{4(G-1)}{G}, \tag{5}$$

so, with $a = \mathbb{E}_x[p_\pi(x)]$ the mean accuracy,

$$\eta_G(\pi) \;=\; c_G\Big[\underbrace{a(1-a)}_{\text{fixed by accuracy}} \;-\; \underbrace{\mathrm{Var}_x\big(p_\pi(x)\big)}_{\text{set by the warm-up}}\Big]. \tag{6}$$

Proof. $\mathbb{E}[k(G-k)] = G\mathbb{E}[k] - \mathbb{E}[k^2] = G(G-1)p(1-p)$, and both mean-baseline masses are proportional to $k(G-k)$; expectations over x with $\mathbb{E}_x[p^2] = \mathrm{Var}_x(p) + a^2$ give eq. (6). □

GRPO's mass involves $\sqrt{k(G-k)}$ and is not exactly proportional to $p(1-p)$, but stays within a bounded factor, and all three estimators order individual groups identically, so the mechanism is not an artifact of one algorithm (appendix B). That $p(1-p)$ governs a *prompt*'s learnability is established (Yu et al., 2025; Zheng et al., 2025; Jiang et al., 2025; Bae et al., 2026); what eq. (6) adds is the aggregation to a *policy*, which is what one must score to choose a warm-up, and which splits the quantity into a part accuracy fixes and a part it does not.

Accuracy is the wrong objective for a warm-up. Since $0 \le \mathrm{Var}_x(p) \le a(1-a)$, with the upper limit attained exactly when every $p_\pi(x) \in \{0,1\}$:

Corollary 1 (Accuracy does not determine signal)**.** *At any fixed mean accuracy $a \in (0,1)$, signal efficiency ranges over $[0, c_G\, a(1-a)]$: maximal when every prompt has the same success probability, exactly zero when outcomes are deterministic per prompt. Hence there exist π, π' with $a(\pi) > a(\pi')$ yet $\eta_G(\pi) = 0 < \eta_G(\pi')$.*

A warm-up that has memorized which prompts it can solve, and solves exactly those every time, offers reinforcement learning nothing to learn from however accurate it is. We call $\mathrm{Var}_x(p_\pi(x))/[a(1-a)] \in [0,1]$ the *polarization* of a policy and report it alongside η.

What a fixed budget buys at each group size. Everything so far takes G as given. It is itself a choice, and once η is written down the trade-off is forced: N generations buy N/G groups, so a larger group earns more signal per group out of strictly fewer groups.

Proposition 3 (Budget allocation)**.** *Fix a per-step generation budget N and let $G \mid N$. For the mean-baseline estimators the expected total normalized advantage mass and the expected number of non-degenerate groups are*

$$M_G \;=\; \frac{N}{G}\,\eta_G(\pi) \;=\; 4N\,\frac{G-1}{G^2}\,\mathbb{E}_x\big[p(1-p)\big], \qquad m_G \;=\; \frac{N}{G}\,\mathbb{E}_x\big[\rho_G(p)\big], \tag{7}$$

with $\rho_G(p) = 1 - p^G - (1-p)^G$. Then M_G is strictly decreasing in G for $G \geq 2$, and m_G is non-increasing, with $\rho_2(p)/2 = \rho_3(p)/3 = p(1-p)$ identically and strict decrease for every $G \geq 3$. Both are therefore maximized at $G = 2$.

Relative to $G = 2$ the same budget delivers 0.44 of the advantage mass at $G = 8$ and 0.12 at $G = 32$. This is the *quantity* of signal, not the quality of an individual advantage estimate, which is what larger groups buy; the two pull opposite ways and only one is usually costed (appendix H).

3.3 A CHOSEN SET SPENDS ITSELF

Everything so far scores a policy at an instant. A prompt set, though, is chosen once and trained on for a whole run, so what matters is its trajectory, and the accounting already fixes its shape.

Proposition 4 (Selection front-loads and self-liquidates)**.** *Let $p_t(x)$ be the success probability of prompt x after t steps, suppose training makes progress, so $t \mapsto p_t(x)$ is non-decreasing, and write $\tau(x) = \inf\{t : p_t(x) \geq \frac{1}{2}\}$ for the time the prompt crosses the mode. Because $s(p) = p(1-p)$ increases on $[0, \frac{1}{2}]$ and decreases on $[\frac{1}{2}, 1]$: (i) each prompt's signal $s(p_t(x))$ is unimodal in t, non-decreasing before $\tau(x)$ and non-increasing after; (ii) ranking prompts by $s(p_0)$ ranks them by $|\operatorname{logit} p_0|$, so taking the top selects the* earliest *crossings; and (iii) hence the selected set peaks no later than any set drawn further below the mode, and declines while that set still climbs, so their difference in η is largest at the outset and reverses if training passes the second set's crossing.*

Proof. Immediate from the monotonicity of s on each side of its mode and linearity of expectation. □

A set selected to maximize $p(1-p)$ is placed just under the mode and crosses it almost at once: measured p_0 is 0.466, 0.443 and 0.496 across our three settings. Whether a reversal is predicted at all turns on where the *control* starts, and our settings separate on exactly that. On Countdown it starts far below the mode (0.113, 0.129) and climbs toward it, so at 3B the curves cross inside the first quarter, curated η falling $0.42 \to 0.25$ against the control's $0.32 \to 0.41$, unanimously over five seeds. On GSM8K the control starts *above* the mode (0.620), moving away from the peak rather than toward it, so no crossing is predicted and none occurs (appendix G.1). A prediction that picks out the one setting where a reversal happens is doing more than describing it. It also predicts the shape of the *outcome*, since what a run banks is the accumulated gap: front-loaded, narrowing as the horizon grows. Monotone progress is an idealization, but only its direction is used and aggregate pass rates are monotone in every run we report.

Two regimes, and why the decoupling has to be constructed. Where a policy sits on the curve matters. When success probabilities are uniformly small, $p(1-p) \approx p$ and $\eta \approx c_G a$, so signal and accuracy barely separate and corollary 1 bites only at appreciable accuracy or heterogeneous difficulty; that is why the decoupling must be constructed rather than observed (appendix E), and why section 5.3 finds the two collinear at $\rho = 0.94$. The a here is accuracy under the sampling distribution, not the greedy accuracy conventionally reported.

The estimator we actually use, and a hazard it carries. Theorem 1 is exact for the mean-baseline estimators; GRPO divides by the group standard deviation, so its mass goes as $\sqrt{k(G-k)}$ and no such identity holds for the quantity we report. It is bounded, which suffices. On $k(G-k) > 0$ the integer $X = k(G-k)$ obeys $G - 1 \leq X \leq G^2/4$, so $2X/G \leq \sqrt{X} \leq X/\sqrt{G-1}$ and

$$c_G\,p(1-p) \;\leq\; s_G^{\mathrm{GRPO}}(p) \;\leq\; 2\sqrt{G-1}\,p(1-p), \tag{8}$$

placing the ratio to the mean-baseline value in $[1, \kappa_G]$, $\kappa_G = G/(2\sqrt{G-1})$, which is 1.51 at $G = 8$; on our checkpoints it is 1.13 to 1.29. Equation (6) therefore transfers up to a known factor, and the rule of section 5.1 transfers exactly, since $p(1-p)$ and $\sqrt{p(1-p)}$ rank prompts identically.

The same scale invariance carries a hazard we have not seen stated. A shaping term is conventionally given a small weight so it cannot dominate the task reward. Under GRPO that fails on exactly the groups where it matters: if a group is degenerate in correctness its rewards are a constant plus λ times the shaping term, and since $(r - \bar{r})/\sigma_r$ is invariant to positive rescaling, λ cancels and the group carries the *full* mass $2\sqrt{k'(G-k')}$ for any $\lambda > 0$. At $G = 8$ a group with no correct rollout but a format term of weight 10^{-6} splitting it four to four carries exactly the mass of a genuinely four-of-eight correct group; under the mean-baseline estimators the contribution scales with λ as intended. This is why a base model's format improves while its accuracy does not (section 5.3).

4 Instrument, Setup, and Measurement

To exercise corollary 1 we need warm-ups differing in signal efficiency while the problems, answer format and supervised budget are held fixed. Verifier-backed search-control distillation provides that: from one offline tree search per problem we serialize supervision targets differing only in how much of the search's failure structure they expose, which moves η over a wide range.

Geometries, the verifier, and four targets. A geometry specifies an initial state, legal moves, a deterministic transition, a terminality test, and an exact verifier V_g accepting only a complete valid solution; we use five, generated programmatically from a fixed seed so every label and reward is exact (Countdown, Game-of-24, a 4×4 Sudoku, an arithmetic chain, N-Queens). A node is *solvable* if its subtree contains a state accepted by V_g, and at a frontier node the controller emits one decision from $\mathcal{A} = \{\textsc{Expand}, \textsc{Backtrack}, \textsc{Commit}, \textsc{Abandon}\}$, defined against solvability rather than a learned value (algorithm 1). Because the state space is finite within the node budget, solvability is *decided* rather than estimated, which separates these labels from values distilled from Monte Carlo Tree Search. The verifier and not the model must supply them: asked to choose between Expand and Abandon at nodes whose oracle label is known, base models agree with the verifier at chance ($\kappa = 0.021, 0.023, 0.091$ at 1.5B, 3B, 7B; appendix K). From the same trees we serialize four targets sharing the problem set, prompt and answer format: *typed control* interleaves the four tags with the moves; *backtrack-only* keeps just Backtrack and Commit, reproducing Dixit et al. (2026); *untyped trace* shows the identical exploration with the tags removed, in the style of Stream-of-Search (Gandhi et al., 2024); and *solution-only* shows the winning path alone. They differ threefold in length, 52 tokens against 174, which is the lever on η.

Training, and what is held fixed. The primary model is Qwen2.5-3B, with 1.5B and 7B for the diagnostic above, a 7B replication in section 5.2, a second family (Llama-3.2-3B) and a second geometry (Game-of-24). Both stages use rank-32 adapters in bfloat16 on one H200. Supervised fine-tuning runs three epochs on 600 problems unless the epoch count is the manipulated variable (section 5.2). Reinforcement learning is GRPO (TRL 0.24) against the exact verifier plus a small format term, 150 steps at $G = 8$ and 32 generations per step, 4800 in all, at 320 tokens, temperature 1.0, KL coefficient 0.04. Supervised fine-tuning uses AdamW at 10^{-4}. The reinforcement-learning rate is 10^{-6} in sections 5.3 and 5.4, a default we inherited that appendix G.5 shows bounds how far accuracy can move; sections 5.1 and 5.2 use 2×10^{-5} and 10^{-5}, chosen by a pilot on the *control* arm. Within a setting all arms share everything but the supervised starting weights; the interventions of sections 5.1 and 5.2 instead hold the weights fixed and vary one element of the budget, and say which. Evaluation is on 256 held-out problems every thirty steps, three seeds per setting unless stated, reporting means and population standard deviations over seeds. The search experiments use Countdown with three and four numbers, where checkpoints reach non-trivial accuracy and traces fit the budget. The format term needs care, since proposition 2 says a component constant within a group cancels while one that varies does not, and it varies far more often inside the trace arms' groups than the solution arm's. **Every η we quote is therefore the correctness-only quantity**, computed with GRPO's standard-deviation-normalized mass, the trainer's own estimator. Equation (6) is thus exact for the mean-baseline mass and an ordering for the numbers we report, which suffices throughout because $p(1-p)$ and $\sqrt{p(1-p)}$ rank prompts identically (appendix C).

Table 1: Prompt selection by one pre-training sampling pass, against a random control. Within each setting the warm-up, the 292-prompt set size, the budget, the held-out set and the five seeds are identical and only the rule differs. η is measured on a *second* independent pass, so it describes what each set contains rather than what selection hoped. The two rules that fail, and all budgets and pool sizes, are in appendix G.1.

Setting	Prompt set	η	Final	Gain	vs. control
Countdown, 3B	random	0.309	0.193	+0.050	—
	top $p(1-p)$	0.800	0.216	+0.082	+0.0227 (5/5)
Countdown, 7B	random	0.409	0.251	+0.040	—
	top $p(1-p)$	0.869	0.273	+0.053	+0.0219 (5/5)
GSM8K, 7B-Instruct	random	0.438	0.776	+0.085	—
	top $p(1-p)$	0.865	0.855	+0.166	+0.0784 (5/5)
GSM8K, 14B-Instruct	random	0.422	0.861	+0.130	—
	top $p(1-p)$	0.856	0.892	+0.160	+0.0312 (5/5)

Measuring signal efficiency. Before any reinforcement learning we replay what GRPO would see on its first pass, at the same prompts, group size, temperature and reward, and from the resulting groups estimate

$$\hat{\eta} \;=\; \frac{1}{n}\sum_{x}\frac{\Psi_G(k_x)}{\Psi_G^{\max}}, \tag{9}$$

which is unbiased for η because each prompt contributes one honest draw of k from the policy's own binomial. Two finite-sample corrections are easy to miss: the plug-in $s_G(\hat{p})$ is biased down by roughly 13% by Jensen's inequality, and the sample variance of $\hat{p}$ overstates $\mathrm{Var}_x(p)$ by more than a factor of two at $G = 8$ unless corrected for binomial noise (appendix B). During training we recompute the same quantity from the group reward vectors at every step, which costs nothing and gives the effective-compute trajectories of section 5.3.

5 Results

A reinforcement-learning budget is spent on one thing, the prompt set, and two decisions about it are usually made by default. Equation (6) prices both: prompts near $p = 1/2$ raise the signal a budget can carry, and a warm-up that has already solved them drives $a \to 1$ and destroys it. We take them in that order, then scope the quantity: what it explains exactly, and what it cannot predict.

5.1 Choosing the prompt set

Equation (5) says a prompt's mass is proportional to $p(1-p)$, so a set chosen to maximize it should convert a fixed budget into more progress (fig. 1b); prior work exploits it online, during training (Yu et al., 2025; Zheng et al., 2025; Jiang et al., 2025). We select 292 prompts by four rules, holding the warm-up, set size, budget, held-out set and five seeds fixed (table 1); the control is random.

The rule eq. (5) prescribes beats that control on *every one of twenty* runs across four settings: $+0.023$ and $+0.022$ on Countdown at 3B and 7B, and $+0.078$ and $+0.031$ on GSM8K with Qwen2.5-7B- and 14B-Instruct. Those twenty are not twenty trials of the rule: seeds within a setting share one drawn control, so the unit randomized is the draw. Redrawing the 3B control four more times, the selected set beats all five, $p = 0.031$; the correction matters, 43% of the control's variance lying between draws with a spread exceeding the effect (appendix G.1).

What that is worth, in the currency it is paid in. An accuracy difference at a fixed budget is a compute saving only if the control has stopped improving, and ours had not: 275 and 400 steps rather than 200 take it from 0.1930 to 0.2055 to 0.2141. The lead is a gain rather than a head start, which needs the control given the same horizon: at 400 steps the curated arm reaches 0.2281 against 0.2141, $+0.0141$ on five of five seeds, narrowing without closing. What selection does not buy is compute. Priced as the pass budget a setting affords, $C^\star = n_{\mathrm{ctrl}}(a_c) - n_t$ with n_{ctrl} what the control needs to reach a_c unaided, the cheaper pass earns $C^\star = 2764$ and costs 2400: $1.04\times$,

bootstrap $[0.86, 1.45]$, a wash, while the full pass loses outright at $0.36\times$ (appendix G.1). Because $C^\star$ belongs to the setting, not our rule, it is the bar any selection method must clear, whether it buys its estimate with a pass or reads it from rollouts a run spends (Yu et al., 2025; Zheng et al., 2025).

Realized signal does not mediate it, and the run mean hides why. At 3B the winning arm ends with *less* in-run η than the control it beats, 0.30 against 0.38, which contradicts a conversion-rate reading and is what proposition 4 requires (fig. 2). Its gain is front-loaded, $+0.070$ in the first quarter against $+0.012$ across the other three, where the control's is $+0.037$ then $+0.013$: in that first quarter it has both the higher η and nearly twice the gain, and its η then collapses *because* those prompts have been solved, since the aggregates bound at least 0.339 of its groups unanimously *correct* against at least 0.442 of the control's unanimously *wrong*, and the control ends with the *higher* live fraction (appendix G.1), so selection defers the degenerate budget rather than avoiding it.

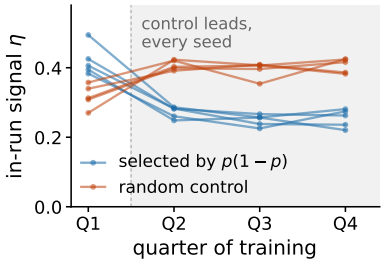


Figure 2: Proposition 4 in the data, each seed drawn. The selected set leaves the mode of $s(p)$ as the control climbs toward it, so from Q2 on the control carries *more* signal, on every seed.

Refreshing a spent set works once and does not replicate. Proposition 4 suggests a free intervention: a run knows which prompts went degenerate, so replace them. On Countdown it gains $+0.0148$ on all five seeds; on GSM8K it is null, with 47 of 292 prompts spent against Countdown's 161; and refreshing repeatedly *loses* to never refreshing, as offline scores go stale and the set overshoots the optimum. The mechanism is real, the recipe not established, and in appendix G.3 three arms differing in η by half rank in reverse on accuracy.

The rule discriminates, which is why we believe it. Weighting by remaining headroom, $p(1-p)^2$, gives the largest gain at 3B and on GSM8K yet reverses at Countdown 7B, where dropping only the provably silent prompts is null too. Only the quantity the theory names survives everywhere; all four rules sit in one table. Against that weakest variant rather than the control, the full rule is ahead by $+0.022$ at 7B and on GSM8K and 0.009 behind at 3B: what the theory buys over merely discarding silent prompts is real but small and not uniform. Because training moves p, the best *starting* point is strictly below the mode, and at each setting's horizon $p^\star \in [0.46, 0.49]$, so aiming at $1/2$ is near-optimal throughout (appendix G.2). The selected set lands within 0.05 of $p^\star$ everywhere and wins on every seed; the variants aim lower, land 0.13 to 0.29 past it, and account for every failure.

5.2 Losing the prompt set to the warm-up

The same equation runs the other way. When one generator supplies both stages, the prompts reinforcement learning trains on *are* what the warm-up was supervised on, so training it longer drives $a(1-a)$ to zero on exactly the set the next stage will use. Sweeping the supervised budget over 1, 3, 9 and 27 epochs (table 3), solution distillation at 27 epochs solves 98.6% of those prompts and 89.8% on *every* rollout, so η falls from its peak of 0.600 at nine epochs to 0.065, and its 4800 generations deliver 312. Its held-out accuracy, the highest in the sweep at 0.369, gives no hint of it.

Holding that checkpoint fixed and swapping its prompts for fresh problems of the same difficulty isolates the cause: in-training η is 0.196 on the fresh prompts against 0.061 on its own, and at 7B, where the model absorbs its supervision more completely, 0.175 against 0.011. Measured from 1.5B to 32B the always-solved fraction runs 0.921 to 0.986, so the collapse is no small-model artifact and saturates by 7B. Whether it costs accuracy needs two corrections, for the inherited optimizer setting and a difference in target distribution; corrected for both, the overlapping arm *loses* 0.013 against a matched fresh arm's $+0.006$, positive on four of five seeds, not excluding zero, and null at 14B (appendix G.5). The gradient loss is large and certain, the accuracy cost small and not established.

5.3 What the accounting explains, and what it cannot predict

It explains the base-model bound exactly. The base model's flat curve is usually read as an inability to emit a parseable answer. The measurement corrects that: its correctness signal is an

order of magnitude below every warm-up, 0.060 against 0.31 to 0.35, and 83% of its groups are degenerate on correctness while the format term varies in 95% of them, so by eq. (8) they carry *full* mass whatever its weight. Its gradient is real and almost none is about being right.

It does not rank warm-ups and does not predict the gain. Over 32 runs spanning four settings, two model families and two geometries, supervised accuracy predicts area under the curve almost perfectly ($R^2 = 0.980$), a warning that the summary restates where a warm-up started, and predicts the *gain* not at all ($R^2 = 0.001$). Adding η raises that to 0.242, the largest move any candidate makes and nominally significant ($p = 0.018$), which we still decline to credit: with arm and setting effects removed η reaches only $p = 0.055$, and none of nine candidates survives correction for having tried nine (appendix G.4). Sampled accuracy and η are collinear here at $\rho = 0.94$, as the theory predicts when $\eta \approx c_G a$. That supervised quality fails to predict outcome is established at greater scale by Kang et al. (2025); we add not a better predictor but a reason, since the Pass@k that does predict it reads the same per-prompt success distribution η is derived from. Within a setting the four arms begin at $\eta = 0.31$ to 0.35 yet end differently: η identifies the regime, not the warm-up to pick.

And nominal compute is not effective compute. Every run in the matched-compute comparison spends exactly 4800 generations, of which the fraction carrying a correctness gradient spans $8.8\times$, 209 to 1839, one Llama base run receiving exactly zero (table 2). It also reclassifies Game-of-24: the arms look close, but in effective compute every one is starved at 4.7% to 13.4% of budget (appendix F).

5.4 The instrument, and what it did not show

Verifier-backed control distillation is how we obtain warm-ups differing in η at a fixed problem set, prompt and answer format, and is not itself the contribution. At three supervised epochs typed control ends 0.027 above solution distillation and 0.023 above a backtrack-only ablation (Dixit et al., 2026), but with three seeds neither separates ($p = 0.09$, $p = 0.23$): the ordering is suggestive, the ablation underpowered rather than null. The ranking is in any case a property of the supervised budget, typed control leading by 0.025 at three epochs and trailing by 0.115 at nine, since solution distillation improves faster per epoch on targets a third the length; what survives is that the *gain* favours typed control at every budget. In four further appendix manipulations (appendices E, F, J and L) signal quantity lands where eqs. (6) and (7) say but does not alone determine the outcome.

6 Discussion and conclusion

We studied how much gradient signal a policy supplies to group-relative reinforcement learning, deriving it exactly for GRPO, Dr.GRPO and RLOO as a signal efficiency η. We found that a run's budget and its signal differ widely. Identically budgeted runs differed 8.8-fold in gradient-carrying generations, a warm-up trained on its own reinforcement-learning problems turned 4800 generations into 312, and a base model's gradient came almost entirely from the format reward. We proposed selecting prompts by $p(1-p)$ from one sampling pass, and it beat a same-size, same-budget random set on all twenty runs across four settings. We also found its limit, since η does not rank warm-ups or predict a run's gain. It prices prompts, not checkpoints.

We recommend four practices, each of which changed one of our own conclusions. Report how much a run gains, not only where it ends; compare warm-ups at more than one supervised budget, since their ranking flipped between three and nine epochs; measure accuracy at the sampling temperature; and count the generations that carry a gradient, not those spent. We used search-control distillation only as an instrument for varying η, and its backtrack-only form, which reproduces Dixit et al. (2026), did not separate from the full typed vocabulary at three seeds.

Our models span 1.5B to 32B with low-rank adapters, mostly on synthetic puzzles with exact labels and modest accuracies. The cheaper selection pass breaks even on compute, and re-selecting prompts mid-run lost to a fixed set, so we offer an accounting and a static rule, not a scheduler (appendix G.3). The theory also predicts that the best target difficulty shifts with training length by about 0.03 in p, finer than our prompt sets, spaced 0.12 apart, could resolve (appendix G.2). Matched compute is not matched signal, and the gap is measurable in advance.

AI USE STATEMENT

During the preparation of this manuscript, the authors used AI tools to assist with document formatting, LaTeX syntax, and technical code maintenance. In particular, an AI coding agent, Antigravity, was used to support refactoring and organizing the experimental codebase for release as supplementary material, with the aim of improving clarity and reproducibility for reviewers.

ETHICS STATEMENT

This study runs mainly on synthetic reasoning geometries generated programmatically from fixed seeds, together with GSM8K (Cobbe et al., 2021), a public human-authored benchmark used under its MIT licence and not redistributed. It involves no human subjects, no personal or sensitive data, and no release of a scraped dataset, so the usual concerns around consent, privacy, and dataset documentation do not arise. The models fine-tuned here are small open-weight language models trained to solve closed-form arithmetic and constraint puzzles against an exact verifier, and the contribution is a measurement of how supervision shapes a subsequent reinforcement-learning phase rather than a deployable system. We report negative and regime-dependent outcomes faithfully throughout.

REPRODUCIBILITY STATEMENT

The geometries, verifiers, search teacher, and serialization targets are described in section 4 and algorithm 1; the estimator definitions and the exact quantity we measure are given in section 3 and eq. (9); and the complete experimental configuration, including base models, adapter rank, the matched reinforcement-learning budget, the held-out splits, the seed counts, and the difficulty regime, is in section 4. Because the problems are generated programmatically from fixed seeds rather than downloaded, the data pipeline is reconstructable from the seed and the difficulty range, and the verifier itself serves as the evaluation oracle, so there is no annotation ambiguity. The construction of the matched-accuracy prompt sets, which is the step where the intervention of appendix E could silently go wrong, is given in full in appendix E. Every table in the paper is generated from stored per-run result files by two scripts, `analysis/tables.py` and `analysis/tables_signal.py`, rather than transcribed by hand, and all thirteen rebuild byte-identically from the committed artifact bundle alone. Statistics quoted only in prose, the t and R^2 values and the agreement coefficients, are computed by the analysis modules named beside them. The complete experimental codebase accompanies this submission, both as supplementary material and anonymously at `https://anonymous.4open.science/r/signal-efficiency-rl-D5C1`. It comprises the core library and its geometries, the supervised and reinforcement-learning entry points, the measurement and prompt-carving tools, every analysis module, the per-experiment run scripts, and the test suite of appendix B. It includes `analysis/verify_paper.py`, which pairs each reported number with a function recomputing it from the raw per-run result files at a stated tolerance and reports a claim whose inputs are absent rather than passing it silently; it currently checks eighty-five claims. Result files and checkpoints are too large to attach and are not included, so the analysis modules reproduce the reported numbers once runs have been executed; the released scripts regenerate those runs end to end.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.

Zachary Ankner, M. Paul, Brandon Cui, Jonathan Chang, and Prithviraj Ammanabrolu. Critique-out-loud reward models. *ArXiv*, abs/2408.11791, 2024.

Sanghwan Bae, Jiwoo Hong, Min Young Lee, Hanbyul Kim, JeongYeon Nam, and Donghyun Kwak. Online difficulty filtering for reasoning oriented reinforcement learning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 700–719, 2026.

Maciej Besta, Nils Blach, Aleš Kubíček, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, H. Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In *AAAI Conference on Artificial Intelligence*, pp. 17682–17690, 2024.

Siddhant Bhambri, Upasana Biswas, and Subbarao Kambhampati. Interpretable traces, unexpected outcomes: Investigating the disconnect in trace-based knowledge distillation. *ArXiv*, abs/2505.13792, 2025.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *ArXiv*, abs/2407.21787, 2024.

Hongyi Cai, Junlin Wang, Xiaoyin Chen, and Bhuwan Dhingra. How much backtracking is enough? exploring the interplay of sft and rl in enhancing llm reasoning. *ArXiv*, abs/2505.24273, 2025.

Liang Chen, Xueting Han, Li Shen, Jing Bai, and Kam-Fai Wong. Beyond two-stage training: Cooperative sft and rl for llm reasoning. *ArXiv*, abs/2509.06948, 2025a.

Zhipeng Chen, Yingqian Min, Beichen Zhang, Jie Chen, Jinhao Jiang, Daixuan Cheng, Wayne Xin Zhao, Zheng Liu, Xu Miao, Yang Lu, Lei Fang, Zhongyuan Wang, and Ji-Rong Wen. An empirical study on eliciting and improving r1-like reasoning models. *ArXiv*, abs/2503.04548, 2025b.

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

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, R. Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633–638, 2025.

Aradhya Dixit, Tianxi Liang, and Jai Telang. Project aletheia: Verifier-guided distillation of backtracking for small language models. *ArXiv*, abs/2601.14290, 2026.

Tao Feng, Yicheng Li, Chenglin Li, Hao Chen, Fei Yu, and Yin Zhang. Teaching small language models reasoning through counterfactual distillation. In *Conference on Empirical Methods in Natural Language Processing*, pp. 5831–5842, 2024.

Yuqian Fu, Tinghong Chen, Jiajun Chai, Xihuai Wang, Songjun Tu, Guojun Yin, Wei Lin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. Srft: A single-stage method with supervised and reinforcement fine-tuning for reasoning. *ArXiv*, abs/2506.19767, 2025.

K. Gandhi, Ayush K Chakravarthy, Anikait Singh, nathan lile, and Noah D. Goodman. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *ArXiv*, abs/2503.01307, 2025.

Kanishk Gandhi, Dorsa Sadigh, and Noah D. Goodman. Strategic reasoning with language models. *ArXiv*, abs/2305.19165, 2023.

Kanishk Gandhi, D. Lee, Gabriel Grand, Muxin Liu, Winson Cheng, Archit Sharma, and Noah D. Goodman. Stream of search (sos): Learning to search in language. *ArXiv*, abs/2404.03683, 2024.

Xinyu Guan, L. Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. rstar-math: Small llms can master math reasoning with self-evolved deep thinking. *ArXiv*, abs/2501.04519, 2025.

Caglar Gulcehre, T. Paine, S. Srinivasan, Ksenia Konyushkova, L. Weerts, Abhishek Sharma, Aditya Siddhant, Alexa Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, A. Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (rest) for language modeling. *ArXiv*, abs/2308.08998, 2023.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and H. Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *ArXiv*, abs/2503.24290, 2025.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. *ArXiv*, abs/2310.01798, 2023.

Guochao Jiang, Wenfeng Feng, Guofeng Quan, Chuzhan Hao, Yuewei Zhang, Guohua Liu, and Hao Wang. Vcrl: Variance-based curriculum reinforcement learning for large language models. *arXiv preprint arXiv:2509.19803*, 2025.

Can Jin, Yang Zhou, Qixin Zhang, Hongwu Peng, Di Zhang, Marco Pavone, Ligong Han, Zhang-Wei Hong, Tong Che, and Dimitris N. Metaxas. Your reward function for rl is your best prm for search: Unifying rl and search-based tts. *ArXiv*, abs/2508.14313, 2025a.

Hangzhan Jin, Sitao Luan, Tianwei Ni, Sicheng Lyu, Guillaume Rabusseau, Reihaneh Rabbany, D. Precup, and Mohammad Hamdaqa. Rl fine-tuning heals ood forgetting in sft. *ArXiv*, abs/2509.12235, 2025b.

Feiyang Kang, Michael Kuchnik, Karthik Padthe, Marin Vlastelica, Ruoxi Jia, Carole-Jean Wu, and Newsha Ardalani. Quagmires in SFT-RL post-training: When high SFT scores mislead and what to use instead. *arXiv preprint arXiv:2510.01624*, 2025.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms. *ArXiv*, abs/2410.01679, 2024.

Joongwon Kim, Anirudh Goyal, Liang Tan, Hanna Hajishirzi, Srini Iyer, and Tianlu Wang. Astro: Teaching language models to reason by reflecting and backtracking in-context. *ArXiv*, abs/2507.00417, 2025.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M. Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal M. P. Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. *ArXiv*, abs/2409.12917, 2024.

Lucas Lehnert, Sainbayar Sukhbaatar, DiJia Su, Qinqing Zheng, Paul Mcvay, Michael Rabbat, and Yuandong Tian. Beyond a*: Better planning with transformers via search dynamics bootstrapping. *ArXiv*, abs/2402.14083, 2024.

Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Eric Tang, Sumanth Hegde, Kourosh Hakhamaneshi, Shishir G. Patil, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Llms can easily learn to reason from demonstrations structure, not content, is what matters! *ArXiv*, abs/2502.07374, 2025.

Wendi Li and Yixuan Li. Process reward model with q-value rankings. *ArXiv*, abs/2410.11287, 2024.

H. Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, I. Sutskever, and K. Cobbe. Let's verify step by step. *ArXiv*, abs/2305.20050, 2023.

Alexis Limozin, Eduard Durech, Torsten Hoefler, Imanol Schlag, and V. Pyatkin. Sft-then-rl outperforms mixed-policy methods for llm reasoning. *ArXiv*, abs/2604.23747, 2026.

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*, abs/2503.20783, 2025.

Ximing Lu, Seungju Han, David Acuna, Hyunwoo Kim, Jaehun Jung, Shrimai Prabhumoye, Niklas Muennighoff, M. Patwary, M. Shoeybi, Bryan Catanzaro, and Yejin Choi. Retro-search: Exploring untaken paths for deeper and efficient reasoning. *ArXiv*, abs/2504.04383, 2025.

Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Yanhao Li, Bin Cui, and Wentao Zhang. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. *ArXiv*, abs/2506.07527, 2025.

Yiran Ma, Zui Chen, Tianqiao Liu, Mi Tian, Zhuo Liu, Zitao Liu, and Weiqi Luo. What are step-level reward models rewarding? counterintuitive findings from mcts-boosted mathematical reasoning. *ArXiv*, abs/2412.15904, 2024.

L. Markeeva, Sean McLeish, Borja Ibarz, Wilfried Bounsi, Olga Kozlova, Alex Vitvitskyi, Charles Blundell, Tom Goldstein, Avi Schwarzschild, and Petar Veličković. The clrs-text algorithmic reasoning language benchmark. *ArXiv*, abs/2406.04229, 2024.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke S. Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In *Conference on Empirical Methods in Natural Language Processing*, pp. 20275–20321, 2025.

Jingwei Ni, Yihao Liu, Xinpeng Liu, Yutao Sun, Mengyu Zhou, Pengyu Cheng, Dexin Wang, Xiaoxi Jiang, and Guanjun Jiang. Trace2skill: Distill trajectory-local lessons into transferable agent skills. *ArXiv*, abs/2603.25158, 2026.

Andreas Opedal, Francesco Ignazio Re, Abulhair Saparov, Mrinmaya Sachan, Bernhard Scholkopf, and Ryan Cotterell. Learning to reason efficiently with a* post-training. *ArXiv*, abs/2605.24597, 2026.

Bo Pang, Deqian Kong, Silvio Savarese, Caiming Xiong, and Yingbo Zhou. Reasoning curriculum: Bootstrapping broad llm reasoning from math. *ArXiv*, abs/2510.26143, 2025.

Simon Park, Simran Kaur, and Sanjeev Arora. How does rl post-training induce skill composition? a case study on countdown. *ArXiv*, abs/2512.01775, 2025.

Zhenting Qi, Mingyuan Ma, Jiahang Xu, L. Zhang, Fan Yang, and Mao Yang. Mutual reasoning makes smaller llms stronger problem-solvers. *ArXiv*, abs/2408.06195, 2024.

Tian Qin, David Alvarez-Melis, Samy Jelassi, and Eran Malach. To backtrack or not to backtrack: When sequential search limits model reasoning. *ArXiv*, abs/2504.07052, 2025.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. *ArXiv*, abs/2407.18219, 2024.

Nived Rajaraman, Audrey Huang, Miro Dudik, Rob Schapire, Dylan J. Foster, and Akshay Krishnamurthy. Learning to reason with curriculum i: Provable benefits of autocurriculum. In *Annual Conference on Computational Learning Theory (COLT)*, pp. 5518–5555, 2026.

Bilgehan Sel, Ahmad Al-Tawaha, Vanshaj Khattar, Ruoxi Jia, and Ming Jin. Algorithm of thoughts: Enhancing exploration of ideas in large language models. *ArXiv*, abs/2308.10379, 2023.

Kulin Shah, Nishanth Dikkala, Xin Wang, and Rina Panigrahy. Causal language modeling can elicit search and reasoning capabilities on logic puzzles. *ArXiv*, abs/2409.10502, 2024.

Rulin Shao, S. Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, S. Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Y. Tsvetkov, Hanna Hajishirzi, Pang Wei Koh, and Luke S. Zettlemoyer. Spurious rewards: Rethinking training signals in rlvr. *ArXiv*, abs/2506.10947, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *ArXiv*, abs/2402.03300, 2024.

Guni Sharon. Tree of thoughts as a classical heuristic search problem: Formal foundations and design patterns. *ArXiv*, abs/2605.28566, 2026.

Taiwei Shi, Yiyang Wu, Linxin Song, Tianyi Zhou, and Jieyu Zhao. Efficient reinforcement finetuning via adaptive curriculum learning. *arXiv preprint*, 2025.

C. Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *ArXiv*, abs/2408.03314, 2024.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations of large language models on reasoning and planning tasks. *ArXiv*, abs/2402.08115, 2024.

Xiaole Su, Kasey Zhang, and A. Lyu. SFT-GRPO data overlap as a post-training hyperparameter for autoformalization. *arXiv preprint arXiv:2604.13515*, 2026.

Shivchander Sudalairaj, Kai Xu, Akash Srivastava, and Giorgio Giannone. sGPO: Trading inference FLOPs for training efficiency in RLVR. *arXiv preprint arXiv:2606.08854*, 2026.

Yifan Sun, Jingyan Shen, Yibin Wang, Tianyu Chen, Zhendong Wang, Mingyuan Zhou, and Huan Zhang. Improving data efficiency for llm reinforcement fine-tuning through difficulty-targeted online data selection and rollout replay. *arXiv preprint*, 2025.

Karthik Valmeekam, Matthew Marquez, S. Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models - a critical investigation. *ArXiv*, abs/2305.15771, 2023.

Boshi Wang, Xiang Yue, Yu Su, and Huan Sun. Grokking of implicit reasoning in transformers: A mechanistic journey to the edge of generalization. *Advances in Neural Information Processing Systems 37*, 2024a.

Chaojie Wang, Yanchen Deng, Zhiyi Lyu, Liang Zeng, Jujie He, Shuicheng Yan, and Bo An. Q*: Improving multi-step reasoning for llms with deliberative planning. *ArXiv*, abs/2406.14283, 2024b.

Yangzhen Wu, Zhiqing Sun, Shanda Li, S. Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for llm problem-solving. In *International Conference on Learning Representations*, 2025.

Violet Xiang, Charles Burton Snell, K. Gandhi, Alon Albalak, Anikait Singh, Chase Blagden, Duy Phung, Rafael Rafailov, nathan lile, Dakota Mahan, Louis Castricato, Jan-Philipp Franken, Nick Haber, and Chelsea Finn. Towards system 2 reasoning in llms: Learning how to think with meta chain-of-thought. *ArXiv*, abs/2501.04682, 2025.

Tong Yang, Yu Huang, Yingbin Liang, and Yuejie Chi. Agentic transformers provably learn to search via reinforcement learning. *ArXiv*, abs/2606.00183, 2026.

Xiao-Wen Yang, Xuan-Yi Zhu, Wen-Da Wei, Ding-Chu Zhang, Jiejing Shao, Zhi Zhou, Lan-Zhe Guo, and Yu-Feng Li. Step back to leap forward: Self-backtracking for boosting reasoning of language models. *ArXiv*, abs/2502.04404, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale. *ArXiv*, abs/2503.14476, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *ArXiv*, abs/2504.13837, 2025.

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*, abs/2503.18892, 2025.

Charlie Zhang, Graham Neubig, and Xiang Yue. On the interplay of pre-training, mid-training, and rl on reasoning language models. *ArXiv*, abs/2512.07783, 2025a.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. Rest-mcts*: Llm self-training via process reward guided tree search. *ArXiv*, abs/2406.03816, 2024a.

Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b. *ArXiv*, abs/2406.07394, 2024b.

Yunxiang Zhang, Muhammad Khalifa, Lajanugen Logeswaran, Jaekyeom Kim, Moontae Lee, Honglak Lee, and Lu Wang. Small language models need strong verifiers to self-correct reasoning. *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 15637–15653, 2024c.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. *ArXiv*, abs/2501.07301, 2025b.

Rosie Zhao, Alexandru Meterez, S. Kakade, Cengiz Pehlevan, Samy Jelassi, and Eran Malach. Echo chamber: Rl post-training amplifies behaviors learned in pretraining. *ArXiv*, abs/2504.07912, 2025.

Haizhong Zheng, Yang Zhou, Brian R. Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. *arXiv preprint arXiv:2506.02177*, 2025.

Haodong Zhu, Yangyang Ren, Yanjing Li, Sheng Xu, Haiguang Liu, Linlin Yang, and Baochang Zhang. Kalman meets curriculum: Efficient dynamic prompt selection for adaptive rl finetuning. *arXiv preprint*, 2026.

A Deferred proofs

The three proofs below are elementary and are moved here for space; theorem 1 and proposition 4 keep theirs inline, being the two the argument turns on. Every statement is additionally verified numerically in appendix B.

The normalizing constant at odd G. Theorem 1 normalizes by $\Psi_G(G/2)$, the mass at the continuous maximizer of $k(G-k)$, which no attainable group reaches when G is odd. Normalizing by the attainable maximum $\Psi_G(\lfloor G/2 \rfloor)$ instead replaces $c_G = 4(G-1)/G$ by $4G/(G+1)$ and breaks the strict monotonicity of proposition 3 across $G = 2 \to 3$, so the continuous maximizer is the convention that keeps the group-size result true. Every experiment reported uses even G, where the two coincide, and no reported number depends on the choice.

Vanishing groups (proposition 1) Each estimator subtracts a baseline that is an average of the group's rewards; when all are equal that baseline equals r_i, so every numerator vanishes, and GRPO's denominator is positive.

Advantage mass (proposition 2) For GRPO, $\bar{r} = k/G$ and $\sigma_r = \sqrt{k(G-k)}/G$, so a success has $\hat{A} = \sqrt{(G-k)/k}$ and a failure $-\sqrt{k/(G-k)}$, summing to $2\sqrt{k(G-k)}$; the others follow by the same substitution.

Budget allocation (proposition 3) M_G follows from eq. (5) and $\frac{d}{dG}\frac{G-1}{G^2} = \frac{2-G}{G^3} < 0$ for $G > 2$. For m_G, writing $b = 1-p$ and using $p^3 + b^3 = (p+b)^3 - 3pb(p+b) = 1 - 3pb$ gives $\rho_3 = 3p(1-p) = \frac{3}{2}\rho_2$, so the first two rungs coincide; appendix B verifies the remainder in exact rational arithmetic.

B NUMERICAL VERIFICATION OF THE THEORETICAL CLAIMS

Every claim in section 3 is checked computationally against brute-force evaluation of the estimator definitions, rather than only by hand. The checks below form a test suite that runs in seconds and is part of the released code; we list them because several of them constrain how the quantities may be estimated in practice, and one of them corrected an error we would otherwise have made.

The identities. For every group size $G \leq 16$ and every k, the closed forms of eq. (3) agree with the total absolute advantage computed directly from eq. (1) to within 10^{-15}, for all three estimators. The advantage mass of a constant-reward group is exactly zero for all three, for any common reward value, confirming proposition 1. The GRPO mass is invariant to any affine reparametrization $r \mapsto ar + b$ with $a > 0$ in the limit $\epsilon \to 0$; with the $\epsilon = 10^{-4}$ used in practice the deviation is $O(10^{-3})$, three orders of magnitude below the effects we report. The closed form of eq. (5) matches the binomial expectation to within 10^{-15} for $G \in \{2, 4, 8, 16, 32\}$ across $p \in (0, 1)$, and the decomposition of eq. (6) holds to within 10^{-15} over 3000 randomly drawn policies at three group sizes.

How far the result is estimator-independent. All three estimators are strictly increasing in $k(G-k)$, so they order individual groups identically and vanish on exactly the same set; we verified zero inversions over all pairs of k at $G \in \{4, 8, 16\}$. They are not, however, interchangeable once expectations are taken over a heterogeneous prompt set, because $\sqrt{\cdot}$ and the identity are different concave transforms: over 2000 random pairs of policies, GRPO and RLOO disagree on which has more signal in 3.5% of comparisons, all of them near-ties. We therefore claim identical ordering at the level of groups and near-identical ordering at the level of policies, not exact equivalence.

A related consequence is worth recording because it has a practical edge. Because GRPO divides by the group standard deviation, its expected signal grows like $\sqrt{p(1-p)}$ rather than $p(1-p)$, which compresses differences between policies. On our measured checkpoints, a warm-up with 1.48 times the sampled accuracy of another shows only 1.13 times its GRPO signal but 1.21 times its mean-baseline signal. Standard-deviation normalization makes the effective learning signal less sensitive to policy quality than the mean-baseline estimators are.

The budget-allocation rungs, and why floating point cannot check them. The mass rate $4(G-1)/G^2$ of proposition 3 matches a direct sum over groups to within 10^{-15} at $G \in \{2, 4, 8, 16, 32\}$ and is strictly decreasing. The companion rate $\rho_G(p)/G$ needed more care, and checking it naively produced a false negative we nearly reported. Evaluating $1 - p^G - (1-p)^G$ subtracts two quantities that agree to many digits when p is small, so the result is dominated by cancellation error and the sequence appears to increase at $p \lesssim 10^{-3}$. In exact rational arithmetic over 199 values of p and $G \leq 32$ the sequence never increases, and the only equality is between $G = 2$ and $G = 3$, where $\rho_2/2 = \rho_3/3 = p(1-p)$ identically because $p^3 + (1-p)^3 = 1 - 3p(1-p)$. The decrease is strict from $G = 3$ onward. We record this because the same cancellation would affect anyone estimating ρ directly at low pass rates.

Estimating η from finitely many samples. Two finite-sample facts constrain the measurement and are easy to get wrong.

First, the realized advantage mass of eq. (9) is unbiased for η, because each prompt contributes one honest draw of k from the policy's own binomial; in a 20000-prompt simulation its bias is -0.0003. The natural-looking alternative of substituting $\hat{p} = k/G$ into s_G is biased downward by Jensen's inequality, with a bias of -0.064 on the same simulation, roughly 13% of the true value. For the mean-baseline estimators the realized quantity has the exact form $\hat{\eta} = 4\,\hat{p}(1-\hat{p})$.

Second, the sample variance of $\hat{p}$ is *not* an estimate of $\mathrm{Var}_x(p)$. Since $\hat{p}$ is itself binomial,

$$\mathbb{E}\big[\mathrm{Var}_x(\hat{p})\big] \;=\; \mathrm{Var}_x(p) \;+\; \frac{\mathbb{E}[p(1-p)]}{G}, \tag{10}$$

so at $G = 8$ the naive estimate can exceed the truth by more than a factor of two and would make every policy look far more polarized than it is. Using $\mathbb{E}[\hat{p}(1-\hat{p})] = \mathbb{E}[p(1-p)](G-1)/G$ to estimate the second term recovers the true spread: on a simulation with $\mathrm{Var}_x(p) = 0.0134$, the naive estimate

is 0.0315 and the corrected one is 0.0132. All polarization values reported for trained checkpoints use the corrected estimator; the set-construction figures in appendix E are the raw pass-rate statistics the selection actually consumed, and are larger than their corrected values by up to $1.9\times$.

C IMPLEMENTATION NOTES

The four decisions are defined against solvability rather than a learned value: EXPAND grows a solvable non-terminal node, COMMIT accepts a verified terminal, ABANDON prunes a node whose subtree contains no solution, and BACKTRACK returns to a still-solvable ancestor after a branch is abandoned. The teacher records at most a fixed number of abandoned siblings per node, which shortens a trace without affecting the exactness of any emitted label.

Algorithm 1 Verifier-backed control labeling for a solvable problem

Input: geometry g, problem p, node budget B
compute SOLVABLE(s) for nodes via memoized exhaustive search bounded by B
$s \leftarrow$ initial state; emit EXPAND at s
while s is not terminal **do**
  for each move m in canonical order from s **do**
    $s' \leftarrow \text{apply}(g, s, m)$
    if SOLVABLE(s') **then**
      emit EXPAND on (m, s'); $s \leftarrow s'$; **break**
    else
      emit ABANDON on (m, s'); emit BACKTRACK to s
    end if
  end for
end while
emit COMMIT; **return** typed decision sequence and verified solution

Which quantity the reported η is. The format term varies inside 15% of solution-distillation groups against 77 to 86% of the trace arms', whose long completions are more often truncated before an answer, so a signal figure that included it would compare arms on how often they were cut off. Every η in the paper is therefore correctness-only. It is also GRPO's standard-deviation-normalized mass rather than the mean-baseline quantity eq. (5) is exact for, whose expectation grows like $\sqrt{p(1-p)}$; on our checkpoints the two differ by 16 to 49%, the gap widening as accuracy falls, which is the bounded discrepancy of eq. (8). Appendix J reports the answer-emission rate and the accuracy conditional on emitting one, so the two failure modes stay separable.

A silent numerical failure worth reporting. With the CUDA build used here, scaled dot-product attention produces non-finite gradients on left-padded batches containing fully masked rows. The failure is silent: the first optimizer step writes NaN into every adapter parameter, training continues to report a finite loss of exactly zero, and the run only dies much later inside sampling with an opaque device-side assertion about a probability tensor. Bisecting across attention backends isolates it, and FlashAttention-2 and the eager implementation both produce finite, closely agreeing gradients. Every run reported here uses FlashAttention-2, and the trainer aborts on a non-finite gradient norm so that a poisoned run cannot be mistaken for a real result. We report this because a run affected by it would produce a plausible-looking flat learning curve rather than an obvious crash.

Adapter merging. Merging a supervised adapter into the base weights leaves the merged model carrying a stale adapter configuration. If that configuration survives, the trainer stacks a second adapter on top of it and no gradient reaches the newly initialized parameters. We delete it explicitly before the reinforcement-learning stage, and enable input gradients so that gradient checkpointing reaches the adapter through a frozen embedding.

D ADDITIONAL RELATED WORK

What supervised fine-tuning installs before reinforcement learning. The line closest to our hypothesis argues that supervision installs structure rather than correct content: priming cognitive behaviors such as verification and backtracking is what enables self-improving reinforcement learning (Gandhi et al., 2025); trace correctness has minimal impact and the value of search-trace supervision is as a warm-up (Cai et al., 2025); structure, not content, is what models learn from demonstrations (Li et al., 2025); and a small curated set of long traces can suffice (Muennighoff et al., 2025). Two-stage supervised-then-reinforcement pipelines outperform mixed alternatives (Limozin et al., 2026; Fu et al., 2025; Chen et al., 2025a; Zhang et al., 2025a; Ma et al., 2025). This literature is largely descriptive about *why* structure helps. We supply a mechanism that is measurable in advance and makes falsifiable predictions, including the prediction that the effect must vanish when signal efficiency is near zero for every arm, which appendix I tests deliberately.

Imitating search, and the verifier premise. A second line trains models to imitate the search process rather than its solution. Stream-of-Search serializes an entire search including dead ends and improves it with reinforcement learning, beating optimal-path supervision on Countdown (Gandhi et al., 2024); Searchformer distills A* dynamics (Lehnert et al., 2024); causal language modeling can elicit search on logic puzzles (Shah et al., 2024); and meta chain-of-thought targets the procedure rather than the answer (Xiang et al., 2025; Yang et al., 2026; Opedal et al., 2026; Gandhi et al., 2023). A third distills the value or policy found by Monte Carlo Tree Search (Guan et al., 2025; Zhang et al., 2024a; Qi et al., 2024; Zhang et al., 2024b; Wang et al., 2024b; Lu et al., 2025; Feng et al., 2024), with process reward models that are themselves noisy and hard to train (Lightman et al., 2023; Zhang et al., 2025b; Li & Li, 2024; Ma et al., 2024; Jin et al., 2025a; Bhambri et al., 2025; Ankner et al., 2024). Because language models do not reliably self-correct without external feedback (Huang et al., 2023; Stechly et al., 2024; Valmeekam et al., 2023; Zhang et al., 2024c), our labels come from an exact verifier rather than the model, and backtracking is supervised directly (Kumar et al., 2024; Qu et al., 2024; Yang et al., 2025; Kim et al., 2025). The nearest prior work, Project Aletheia, performs verifier-guided distillation of backtracking into small models and reports that explicit verification behavior triggers rarely at inference (Dixit et al., 2026); our backtrack-only arm reproduces that signal exactly. We use these four supervision targets as an instrument: a convenient way to produce warm-ups that differ while the problems, prompt, and answer format are held fixed.

Geometry dependence and reinforcement learning from base models. Reasoning can also be searched at inference time (Besta et al., 2024; Sel et al., 2023; Sharon, 2026; Brown et al., 2024; Snell et al., 2024; Wu et al., 2025), and the value of a search behavior depends on problem structure: backtracking helps Sudoku and hurts Countdown at fixed compute (Qin et al., 2025), reinforcement learning can heal supervised out-of-distribution forgetting (Jin et al., 2025b), and whether learned reasoning generalizes depends on the structure being learned (Park et al., 2025; Pang et al., 2025; Ni et al., 2026; Wang et al., 2024a; Markeeva et al., 2024). We build the full cross-geometry transfer matrix over three seeds and report where it is and is not predictable. Finally, whether reinforcement learning creates capability or amplifies what is already present is actively debated (Yue et al., 2025; Zhao et al., 2025; Shao et al., 2025), and its success from a base model depends strongly on that model's identity (Zeng et al., 2025). Both bear directly on our replication across families, where the signal a base model can supply differs sharply.

E CONSTRUCTING PROMPT SETS WITH MATCHED ACCURACY AND UNMATCHED SIGNAL

This appendix records how the sets of appendix E were built, because the construction is where the experiment can go wrong and the failure is not visible in the result.

Choice of substrate, and a failure mode it creates. The intervention is run from a single supervised checkpoint, and which one matters more than it appears. Our first choice was solution distillation at nine epochs, selected because its completions are short enough that the format term is saturated and none of its groups are format-only, so its reward is effectively pure correctness and the manipulation is not diluted. The construction succeeded on that checkpoint and the experiment

still could not decide anything, because solution distillation is the one arm in this study whose accuracy does not move under reinforcement learning in any setting or at any supervised budget. With nothing learning there is nothing for signal to modulate. We report the result anyway, in appendix E, because a policy that converts 62% of its budget into gradient and does not improve is itself the sharpest statement of the necessary-but-not-sufficient reading. Anyone repeating this should first check that the substrate responds to reinforcement learning at all.

Pool. The three sets are drawn from 1800 Countdown problems generated from a fresh seed at difficulty 0 (three numbers), each solved by exhaustive search so that only solvable instances enter, and each checked against a canonical key, the sorted multiset of numbers with the target, so that no problem in the pool appears in the supervised set, the reinforcement-learning set, or the evaluation set of the main experiments. A random 256 are set aside as the common held-out evaluation set *before* any success probability is measured, so the evaluation set cannot be selected on. The remaining 1544 form the construction pool.

Two sampling passes, doing different jobs. We measure each pool prompt with 16 rollouts at the temperature and completion length reinforcement learning uses, twice, under independent seeds. The two passes agree closely in aggregate (0.1891 and 0.1911 mean accuracy), which is the check that 16 rollouts is enough to estimate the pool, though not enough to estimate any single prompt. Selection uses pass A alone and description uses pass B alone. This matters more than it might appear: choosing prompts with extreme $\hat{p}$ is a winner's-curse problem, since a prompt reaches the extreme partly because the policy is extreme on it and partly because 16 samples were lucky. Selecting and reporting on the same pass would inherit the whole of that optimism. Concretely, the selection pass puts the uniform-to-polarized η ratio at 4.0 and the independent pass puts it at 2.37; the smaller number is the one we report and the one the runs actually experience.

Selection. Each set has 300 prompts and the three are disjoint. The polarized set is built first because it needs the scarce extremes: sorting the pool by $\hat{p}$ and taking the h largest together with the $300 - h$ smallest, h is chosen so the mean lands on the pool mean, which makes it the maximum-variance set at that mean. The uniform set takes the 300 prompts nearest the target and then walks its mean back onto the target by swapping the selected prompt furthest on the wrong side for the nearest unselected one on the other, which costs the least variance per unit of correction. The intermediate set is fitted last, from what remains, as a blend of extremes and middle chosen to land halfway between the other two in variance.

What the construction achieves. Verified on the independent pass, the three sets have mean success probabilities of 0.197, 0.198 and 0.191, variances of 0.020, 0.071 and 0.096, and signal efficiencies of 0.572, 0.350 and 0.241 at $G = 8$, with the non-degenerate fraction falling from 0.690 to 0.408 to 0.293. Accuracy is matched to within 0.007 while η spans a factor of 2.37, which is what corollary 1 says should be possible and what no manipulation of the warm-up itself produced for us. Repeating the whole construction from a typed-control warm-up at the same supervised budget gives η of 0.571, 0.395 and 0.234 at mean success probabilities of 0.2015, 0.1942 and 0.2067; there the residual accuracy gap runs -0.005, so the polarized set is if anything marginally the easier one and a uniform advantage cannot be attributed to easier prompts.

F EFFECTIVE COMPUTE ACROSS THE STUDY

Table 2: The same nominal budget of 4800 generations, converted into the generations that actually carried a gradient toward correctness. Percentages are of the nominal budget. Arms that a matched-compute protocol calls equal differ here by up to two orders of magnitude, and one Llama base-model run received exactly zero.

Warm-up	Qwen 3B / Countdown	Llama 3B / Countdown	Qwen 3B / Game-of-24
Typed control	1688 (35.2%)	1315 (27.4%)	288 (6.0%)
Backtrack-only	1805 (37.6%)	1367 (28.5%)	226 (4.7%)
Solution-only	1467 (30.6%)	1356 (28.3%)	641 (13.4%)
Untyped trace	1626 (33.9%)	1131 (23.6%)	238 (5.0%)
Base (RL only)	181 (3.8%)	5 (0.1%)	77 (1.6%)

G THE SUPERVISED-BUDGET SWEEP IN FULL

Table 3: Supervised-budget sweep on Countdown with Qwen2.5-3B, two seeds. The right-hand columns are measured during reinforcement learning: the pass rate on the *training* prompts and the correctness-only signal efficiency, which converts the nominal budget into effective generations; the fraction of *groups* that were informative is ρ, a different and larger quantity. Solution distillation at 27 epochs has the highest final accuracy in the table and the lowest signal efficiency in the study.

Warm-up	Epochs	Start	Final	Gain	Train pass rate	$\bar{\eta}$
Typed control	1	0.090	0.090	+0.000	0.010	0.051
	3	0.180	0.209	+0.029	0.143	0.364
	9	0.223	0.244	+0.021	0.240	0.501
	27	0.342	0.346	+0.004	0.604	0.614
Solution-only	1	0.098	0.098	+0.000	0.026	0.120
	3	0.186	0.184	-0.002	0.089	0.315
	9	0.361	0.359	-0.002	0.309	0.600
	27	0.367	0.369	+0.002	0.986	0.065

Table 4: The interventions in full. Weights, reward, budget and held-out set are identical within each block; only how the budget is spent differs. The pass rate is measured on the training prompts during reinforcement learning, so the matched accuracy of the difficulty-spread block is verified inside the training loop and not only in the pre-measurement.

Condition	Train pass	$\bar{\eta}$	Eff. gens.	Start	Final	Gain
Prompt overlap, 27-epoch solution warm-up						
Disjoint prompts	0.260	0.196	1256	0.312	0.316 ± 0.008	+0.004
Supervised-on prompts	0.987	0.061	387	0.309	0.314 ± 0.006	+0.005
Difficulty spread, typed warm-up						
Uniform difficulty	0.206	0.625	3998	0.221	0.236 ± 0.014	+0.014
Intermediate	0.189	0.401	2564	0.225	0.220 ± 0.005	-0.005
Polarized	0.203	0.262	1677	0.203	0.223 ± 0.004	+0.020
Difficulty spread, solution warm-up						
Uniform difficulty	0.200	0.618	3956	0.225	0.232 ± 0.005	+0.007
Intermediate	0.189	0.369	2362	0.228	0.224 ± 0.011	-0.004
Polarized	0.193	0.260	1664	0.232	0.223 ± 0.004	-0.009

Table 5: Signal along the supervised ladder, measured on the 292 difficulty-0 prompts each warm-up was supervised on and on 292 disjoint prompts of the same difficulty, at the group size and sampling temperature reinforcement learning uses. Always-solved is the fraction of prompts correct on every one of eight rollouts, which contributes exactly zero gradient by proposition 1. The left block is concave in accuracy, as eq. (6) requires; the right-hand ratio is the overlap effect, and it crosses one between 27 and 54 epochs of typed control and between 9 and 27 of solution distillation. Accuracies read 1.000 to three decimals are 0.9996 and 0.9991.

		On its own supervised prompts			Disjoint, difficulty-matched		
Warm-up	Epochs	Accuracy	Always-solved	η	Accuracy	η	η ratio
Typed control	1	0.015	0.000	0.075	—	—	—
	3	0.238	0.003	0.574	—	—	—
	9	0.390	0.024	0.697	0.181	0.458	1.52
	27	0.702	0.274	0.612	0.248	0.428	1.43
	54	1.000	0.997	0.002	0.313	0.187	0.01
	81	0.999	0.993	0.005	0.334	0.184	0.02
Solution-only	1	0.038	0.000	0.182	—	—	—
	3	0.146	0.000	0.478	—	—	—
	9	0.440	0.041	0.735	0.177	0.464	1.58
	27	0.989	0.932	0.048	0.256	0.181	0.27

G.1 Prompt selection in full

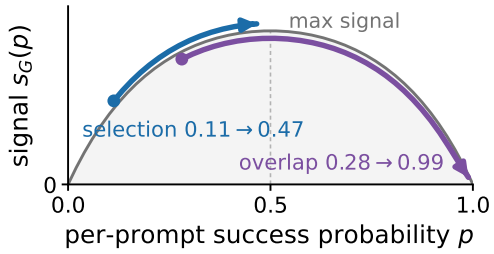


Figure 3: Both experiments move one quantity, in opposite directions, without touching the weights. Selection carries a prompt set from the control's 0.113 to 0.466, just under the peak of $s_G(p)$; overlap carries an overtrained warm-up from fresh prompts (0.281) onto its own (0.989), past the peak into the vanishing tail. Values are starting pass rates; the tracks are lifted off the curve so neither hides the other. This is the detailed form of fig. 1b, adding the fresh-prompt starting point of the overlap experiment.

Table 6: All four selection rules, five seeds each. Two of them fail: weighting by remaining headroom reverses at 7B, and dropping only the silent prompts is null there. Only the rule eq. (5) names survives all three settings. Columns as in table 1, with the realized in-run η and training pass rate added.

		η		Train pass			
Model	Prompt set	before	in-run	rate	Start	Final	vs. control
Countdown, 3B	Random (control)	0.309	0.381	0.15	0.143	0.193	—
	Top $p(1-p)$	0.800	0.299	0.68	0.134	0.216	$+0.0227$ (5/5, $p = 0.012$)
	Top $p(1-p)^2$	0.740	0.390	0.57	0.142	0.232	$+0.0391$ (4/5, $p = 0.012$)
	Drop silent prompts	0.589	0.452	0.35	0.142	0.224	$+0.0312$ (5/5, $p = 0.004$)
Countdown, 7B	Random (control)	0.409	0.392	0.16	0.211	0.251	—
	Top $p(1-p)$	0.869	0.559	0.55	0.220	0.273	$+0.0219$ (5/5, $p = 0.004$)
	Top $p(1-p)^2$	0.764	0.474	0.41	0.218	0.227	-0.0234 (1/5, $p = 0.952$)
	Drop silent prompts	0.631	0.568	0.26	0.213	0.250	-0.0008 (2/5, $p = 0.579$)
GSM8K, 7B-Instruct	Random (control)	0.438	0.366	0.80	0.692	0.776	—
	Top $p(1-p)$	0.865	0.546	0.79	0.689	0.855	$+0.0784$ (5/5, $p = 0.004$)
	Top $p(1-p)^2$	0.827	0.616	0.70	0.693	0.860	$+0.0836$ (5/5, $p = 0.004$)
	Drop silent prompts	0.565	0.448	0.81	0.694	0.833	$+0.0564$ (5/5, $p = 0.012$)
GSM8K, 14B-Instruct	Random (control)	0.422	0.261	0.86	0.731	0.861	—
	Top $p(1-p)$	0.856	0.393	0.87	0.732	0.892	$+0.0312$ (5/5, $p = 0.004$)

The budget is 6400 generations at Countdown 3B and 4800 at Countdown 7B and GSM8K, so blocks of table 1 are comparable within themselves and not across. On GSM8K there is no warm-up at all: the instruct model is the policy. The Countdown pool holds 1544 candidates. The GSM8K pool holds 2400. An earlier carve held 571, because the exclusion filter keyed on the serialized specification and a GSM8K specification is only its numeric answer, so every training item whose answer coincided with any evaluation answer was dropped; the treatment and control sets then shared 141 of 292 prompts, which made that comparison conservative rather than leaky. The question-level overlap between the two splits is in fact zero. Keying the exclusion on the question instead leaves the treatment arm unmoved ($0.8548 \to 0.8540$, two independently carved sets agreeing to 0.0008) and returns the control to 0.7764 from 0.8296, which is the whole of the difference between the $+0.025$ that carve reported and the $+0.078$ here. The design was three seeds, and two more were added at all four conditions after the three-seed result to lower the exact floor from $1/20$ to $1/252$; all five are reported.

The two settings also fail in opposite ways, which is why the same rule works in both for different reasons: Countdown's degenerate prompts are ones the policy never solves, while on GSM8K the instruct model *always* solves 30.3% of the pool, so there selection is mostly removing prompts that have nothing left to teach.

Which way a group was wasted. η and ρ say how much of a run carried gradient, not how the rest failed, and the two ways are opposites: a group whose rollouts all succeed is a prompt the policy has outgrown, one whose rollouts all fail is a prompt it cannot yet reach, and the remedies differ. Writing f_0 and f_1 for the all-fail and all-pass fractions, ρ for the live fraction and m for the mean pass rate, $f_0 + f_1 + \rho = 1$ and $m = f_1 + \rho\,\mathbb{E}[k/G \mid \text{live}]$, and a live group has $k/G \in [1/G, (G-1)/G]$, so $m - \rho(G-1)/G \le f_1 \le m - \rho/G$ with f_0 taking the complement. The bound settles the question whenever it excludes zero. The curated arm, mean pass rate 0.680 against $\rho = 0.391$, has $f_1 \ge 0.339$: even if every live group passed every rollout they could account for only 0.391 of the mean, so the rest must come from unanimously correct groups. The random control, mean pass rate 0.153 against $\rho = 0.462$, has $f_0 \ge 0.442$. Both spend about half their generations on groups that contribute nothing, for opposite reasons. `analysis/waste.py` computes it, exactly where the per-group counts were recorded and by this bound otherwise.

The mediation result of section 5.1 is 3B-specific, and proposition 4 says why. At Countdown 7B the control starts at $p_0 = 0.129$, well below the mode, so its signal climbs as predicted and the gap narrows from 0.21 to 0.10, but 200 steps do not carry it far enough to cross. On GSM8K the control starts at $p_0 = 0.620$, on the *far* side of the mode, so it is moving away from the peak rather than toward it and its signal falls throughout ($0.52 \to 0.28$) alongside the selected set's ($0.86 \to 0.42$); a crossing is not predicted there and does not occur. The one setting that shows a reversal is the one

whose control begins below the mode with room to climb, which is what the proposition asks for and not something we could have arranged after the fact.

Redrawing the control. The twenty runs behind table 1 are four settings by five reinforcement-learning seeds, and within a setting every seed trains on the same two prompt sets: the control was drawn once, at carve seed 0. The unit randomized for this comparison is therefore the draw, not the run, and a run-level sign test assumes twenty independent trials where there are four. Rather than argue about the denominator we redrew the Countdown 3B control four more times (carve seeds 1 to 4, everything else identical, two seeds each; pairwise overlap between draws is 15 to 22%, as expected when sampling 292 of 1544). The five control draws reach 0.1930, 0.1855, 0.1855, 0.1719 and 0.2051, and the selected set at 0.2156 beats all five, exact one-sided sign test $p = 0.031$. The exercise is worth its cost for what it reveals about the design as much as for the confirmation: 43% of the control's variance sits between draws rather than between seeds, and the draw-to-draw spread of 0.033 is larger than the 0.027 effect, so a study reporting one control draw and many seeds is quoting a precision it does not have. `analysis/draws.py`.

Pricing the method end to end. The control ladder that $C^\star$ is measured against is three separate runs of the random set from the same warm-up, at 200, 275 and 400 steps: 0.1930, 0.2055 and 0.2141, with population standard deviations 0.0122, 0.0090 and 0.0173 over five seeds each. They must be separate runs rather than readings off one curve, because the learning rate decays linearly over `max_steps`, so a 400-step run at generation 6400 has had a gentler decay than a 200-step run at its end and the two are not comparable mid-trajectory. Reading a horizon off a longer run's curve is exactly the error that produced the speedup this paragraph replaces. Two accounting conventions, both conservative against us: a selection generation is charged as a training generation, though it runs no backward pass and so costs strictly less; and the pass is charged in full to one run, though one pass chose four sets and served five seeds each. Amortized over R runs from one checkpoint the cheap pass gives 1.13, 1.45 and $1.52\times$ at $R = 1, 5, 20$ and the full pass 0.36, 1.05 and $1.63\times$, but that is an argument about how a study is organized rather than a measurement, so it stays out of the headline. Intervals come from a seed bootstrap that resamples control seeds within each horizon and curated seeds within the arm and rebuilds the curve and its inversion on every draw, so they carry the noise in both the target and the curve it is inverted against; a draw that puts a later horizon below an earlier one is repaired by taking the running maximum, which can only make the control look faster and the saving smaller. Where a curated accuracy exceeds the control's last measured point, as the full pass's 0.2156 exceeds 0.2141 at 12800, n_{ctrl} is a lower bound and is marked as one rather than extrapolated. An arm trained to the end of the ladder cannot be priced at all: n_{ctrl} is censored there, so $C^\star \geq 0$ is true and vacuous, and the persistence comparison above is reported instead as what such an arm can support. `analysis/breakeven.py` recomputes all of it from the raw curve files.

G.2 Why the surviving rule survives

The static accounting says a prompt is worth $p(1-p)$ and is therefore most valuable at $p = 1/2$. That is the right answer for the next step and the wrong one for a set trained on for a whole run, because training moves p: a prompt at the mode is solved fastest and goes silent fastest. Parameterizing a prompt's success probability in logit space, $p = \sigma(z)$, the expected-reward gradient is $\mathrm{d}\mathbb{E}[r]/\mathrm{d}z = \sigma'(z) = p(1-p)$, so under a mean-baseline estimator the flow is autonomous, $\dot{z} = \alpha\,\sigma'(z)$. Two consequences follow. The cumulative mass a prompt supplies telescopes, $\int_0^T \sigma'(z(t))\,\mathrm{d}t = (z_T - z_0)/\alpha$, so a prompt is worth exactly the progress it makes. And the starting point maximizing that is *not* the mode: differentiating through the flow, the optimum $p^\star$ solves $\int_0^T \sigma'(z(s))\,(1 - 2\sigma(z(s)))\,\mathrm{d}s = 0$, which because $\sigma' > 0$ and the trajectory is increasing can hold only for a trajectory starting below the mode and ending above it.

The optimum therefore depends on the setting only through how far a run actually travels in logit space, which every run records. Reading that travel off the *control* arm of each setting, so that a rule which trains faster is not scored against an easier target, gives $p^\star = 0.490$, 0.493 and 0.462 at Countdown 3B, Countdown 7B and GSM8K: a spread of 0.031 across settings whose horizons differ severalfold.

That explains the discrimination result. Aiming at $1/2$ is within 0.04 of the true optimum everywhere, so the static rule is robust by construction, and the sets it produces sit 0.023, 0.050 and 0.034 from $p^\star$ and win on every seed in all three. The variants aim deliberately lower: 0.132, 0.197 and 0.135 past the optimum for $p(1-p)^2$, and 0.288, 0.250 and 0.150 for dropping only silent prompts, and between them they account for every failure in the study. We put no weight on the magnitude: across the nine (setting, rule) cells the correlation between distance from $p^\star$ and the gain is $r = -0.38$, which at $n = 9$ is not evidence. The claim is the qualitative one, that the rule which never overshoots is the one that never fails.

It also retires a null of ours honestly rather than leaving it as a negative. We tested whether the best target drifts with the horizon and found nothing; $p^\star$ moves by 0.031 across these settings, while the prompt sets in that experiment were spaced 0.12 apart, so the design had roughly a quarter of the resolution the effect required and could not have detected it either way. `analysis/optimum.py` recomputes all of it, and `scd/dynamics.py` carries the derivation with its numerical checks.

G.3 Refreshing a spent set

Proposition 4 says a selected set spends itself, which suggests replacing the spent prompts, and both facts that requires are already paid for: the run rolls out $G = 8$ on every prompt it visits and the trainer records the outcome per prompt, while the offline pass measured the candidates it did not select. A prompt counts as spent on its *last* observed group rather than its pooled rate, since one visited early at $k = 4$ and solved later pools to a healthy value while carrying nothing under the current policy; one never visited is kept, absence of an observation not being an observation. Three arms share segment 1 exactly and are chained identically, so they share a starting policy and a learning-rate schedule: *hold* keeps the set, *refresh* replaces spent prompts by highest offline $p(1-p)$, and *churn* replaces the same count at random, which is what distinguishes the rule from mere novelty.

Countdown, one refresh at the midpoint. 161 of 292 prompts were spent, 109 of them by being solved. Held-out accuracy is 0.2039 (hold), 0.2188 (refresh), 0.2156 (churn); training pass rate 0.70, 0.48, 0.36; in-run η 0.24, 0.35, 0.29. Refresh beats hold by $+0.0148$ on five of five seeds (sign $p = 0.031$), landing the set almost exactly at the mode while churn overshoots past it. Refresh beats churn by only $+0.0031$ on three of five, which we report as unresolved rather than negative: the paired standard error is 0.0057, so the design resolves about 0.011. The pool is why, and it is a property of Countdown rather than of the method, the selected set having already taken 292 of the 396 candidates above $p(1-p) = 0.15$.

GSM8K: null, and the manipulation says why. The same design at 7B-Instruct gives $+0.0036$ on two of five. Only 47 of 292 prompts were spent, against Countdown's 161, and the refresh moved the training pass rate by 0.004 (0.791 to 0.787). There was almost nothing to replace. The difference is not the model but the shape of the set's progress: Countdown's prompts polarize into solved and unsolved, while GSM8K's improve together, so a rule that removes individually saturated prompts has little to act on.

Repeated refreshing loses to holding. Starting from a random set and refreshing three times over a run, which is the online form the method is positioned against and the closest we come to Zheng et al. (2025), the rule *loses* to never refreshing at all: -0.0055 on two of five seeds, and random refilling loses by -0.0094 on one of five. The intervention worked and the outcome did not follow. The cause is staleness, which the design anticipated: the refill ranking comes from the initial pass, and prompts it scored near $1/2$ are easier once the policy has moved, so the refreshed set overshoots to a training pass rate of 0.566 while a random refill undershoots to 0.239, and the untouched set sits nearest the optimum by accident. Getting current estimates for unused candidates is exactly the probe a free method refuses to buy.

These three arms are also the study's sharpest dissociation between signal and outcome. They were manipulated to differ in η by half, and their accuracy ordering is its exact reverse: hold $\eta = 0.345$ at 0.2055, rule 0.363 at 0.2000, random refill 0.513 at 0.1961, with $r = -0.35$ over the fifteen runs. Per seed the reversal holds in three of five, so we call it suggestive rather than established; what is

not in doubt is that a large deliberate gain in signal bought no accuracy. We therefore report the mechanism of proposition 4 as confirmed and the recipe built on it as not established.

G.4 Everything we tried against the gain

Token entropy, and eight other candidates. Token-level entropy, whose early collapse is well documented (Cui et al., 2025), appears to be the best predictor of the gain we have ($\rho = -0.46$) and is not one: it is nearly determined by which target was distilled, so conditioning on that drops it to -0.12 and reverses its sign in three of four arms. Of eight candidate statistics of the pre-training pass-rate distribution, none survives multiplicity correction (appendix G.4).

Table 7: Candidate predictors of the reinforcement-learning gain, over the 32 runs that have both a pre-training per-prompt pass-rate measurement and a completed curve. Each is a statistic of the same measured distribution $\{p_i\}$ on the run's own prompt set, except the last. The partial column removes additive arm and setting effects, which is the version a practitioner faces, since the arm is chosen rather than measured; p is two-sided at $\mathrm{df} = 25$ and *none* of these survives a Bonferroni correction over the nine candidates ($p < 0.0056$).

Candidate	Marginal ρ	Partial ρ	p
$\mathbb{E}[p(1-p)^2]$ (signal × headroom)	+0.082	+0.435	0.016
Informative fraction $\mathbb{E}[\rho_G(p)]$	+0.167	+0.383	0.038
$\mathbb{E}[p(1-p)\mathbb{1}(p \le 1/2)]$	+0.071	+0.380	0.040
Fraction with $0 < p < 1$	+0.089	+0.379	0.041
$\eta \propto \mathbb{E}[p(1-p)]$	+0.212	+0.358	0.055
Fraction with $p = 0$	-0.086	-0.348	0.063
Sampled accuracy $\mathbb{E}[p]$	+0.296	+0.170	0.390
Headroom $\mathbb{E}[1-p]$	-0.296	-0.170	0.390
Token entropy at step 1	-0.455	-0.122	0.540

Two things in this table are worth more than the ordering. The first is that token entropy inverts under conditioning. Marginally it is the strongest candidate here, $\rho = -0.455$, which invites the reading that a collapsed policy is one with nothing left to learn. But entropy at the first step is very nearly determined by which target was distilled, reading 0.0652, 0.0627 and 0.0652 across the three seeds of typed control on Qwen/Countdown, and the four target families differ in mean gain. Conditioning on the family removes the effect ($\rho = -0.122$, $p = 0.54$) and the within-family correlations are *positive* in three of four arms, $+0.54$, $+0.25$ and $+0.85$ against -0.18. It is a Simpson reversal, and the marginal number should not be quoted.

The second is that the direction of the conditioning is opposite for the pass-rate statistics: they are weak marginally and stronger once arm and setting are removed, because those effects are large and are not what the statistics are about. The nominal winner, $\mathbb{E}[p(1-p)^2]$, weights each prompt by the headroom it has left rather than symmetrically about $p = 1/2$ as η does, which is the asymmetry one would predict for gain on a held-out set given that pushing an almost-solved prompt higher is the memorization of section 5.2. We tested it for that reason and it does come first, at $\rho = 0.435$ against η's 0.358. We do not claim it: with 32 runs the two are well inside each other's uncertainty, five candidates sit between 0.35 and 0.44, and nothing survives multiplicity correction. What the table supports is the narrower statement made in section 5.3, that no function of the pre-training pass-rate distribution we have found predicts the gain, and that the one quantity that appears to is an artifact of conditioning.

G.5 DOES THE LOSS OF SIGNAL COST ACCURACY?

Table 8: The overlap intervention with two corrections the version in section 5.2 lacks. The learning rate is raised from the 10^{-6} the study inherited to 2×10^{-5}, without which held-out accuracy cannot move far enough for any manipulation to register; and the fresh prompts are drawn from the same pool but matched to the supervised set's target histogram, without which the fresh arm trains on the evaluation distribution and the supervised arm does not. Typed control at 54 epochs, five seeds, 6400 generations each; the manipulation is a 57-fold difference in pre-training signal and eightfold during training.

Reinforcement-learning prompts	Train pass	η	Start	Final	Gain
Its own supervised prompts	0.995	0.0195	0.3672	0.3539	-0.0133
Matched fresh prompts	0.341	0.1558	0.3648	0.3711	+0.0063

Two things had to be fixed before this question could be asked at all, and both are worth recording because either would have produced a confident wrong answer.

The first is that the outcome was not free to move. Every reinforcement-learning run in this study used a learning rate of 10^{-6} with a KL coefficient of 0.04 on rank-32 adapters, inherited as a default and never swept, and the held-out curves plateau within roughly sixty steps and then sit still. Holding the warm-up, prompts, evaluation set, budget and seed fixed and changing only the optimizer, 2×10^{-5} takes the gain from $+0.043$ to $+0.074$, and dropping the KL term as well to $+0.094$; over three seeds the raised setting averages $+0.064$ against the inherited setting's $+0.025$. An outcome null measured inside a channel about 0.02 wide says little about signal and a good deal about the optimizer, so what follows uses 2×10^{-5} with the KL term kept.

The second is that the two prompt sets differed in more than overlap. Both are difficulty-0 three-number Countdown sharing no problem, but the supervised set comes from the original generator and the fresh set from a later pool, giving target means of 14.9 against 18.9, and the held-out evaluation set is drawn from that pool as well, so the fresh arm was training on the evaluation distribution. Uncorrected, that arm leads by $+0.027$ on five of five seeds with complete separation and an exact one-sided rank-sum $p = 0.0040$, which we initially took at face value. What refuted it was the nine-epoch rung: there in-training η is equal between the arms, 0.419 against 0.410, since η is endogenous once the policy is genuinely moving, and yet the fresh arm still led by $+0.026$. An effect that holds constant while its supposed cause varies ninefold is not caused by it. Matching the fresh set to the supervised set's target histogram removes about two fifths of the gap.

What remains we report as unresolved. Over five seeds the arm training on its own supervised prompts *loses* 0.013 of held-out accuracy, negative on four seeds with the fifth exactly zero, while the matched arm gains 0.006; the difference is $+0.017$, positive on four of five, with an exact one-sided $p = 0.064$ and a 95% interval of ± 0.032 that includes zero. The most defensible reading is the within-arm one, which involves no comparison between sets and so cannot be confounded by their provenance: spending a full budget on prompts a warm-up has already memorized, where 99.5% of groups are degenerate, leaves the policy slightly worse on held-out problems than when it began. Reusing supervised problems as reinforcement-learning prompts is not merely wasteful but mildly harmful.

We resist the stronger reading that this establishes η as controlling outcomes. A 57-fold manipulation of signal buys a difference of 0.017 that five seeds cannot separate from zero, a long way from the conversion-rate account, and it is the same qualitative conclusion the rest of the paper reaches by other routes. What the two corrections change is the strength of that conclusion rather than its direction: the earlier nulls were open to the objection that nothing could have moved the outcome, and that objection is now answered on its own terms, with the channel widened by a factor of 2.6 and the distributions matched.

Where the memorization appears along the ladder. Nothing is visibly wrong until well past the conventional operating point, and then almost the entire prompt set goes silent at once: the always-solved fraction moves 0.000, 0.003, 0.024, 0.274, 0.997 across 1, 3, 9, 27 and 54 epochs of typed control, and 0.000, 0.000, 0.041, 0.932 for solution distillation. This is not ordinary overfitting. As

typed control's always-solved fraction rises from 2.4% to 99.7%, its accuracy on *fresh* problems of the same difficulty rises too, $0.181 \to 0.248 \to 0.313$, so the checkpoint with least left to learn from its own prompts is the strongest one we have on new ones; what collapses is the informativeness of a particular pairing between the two stages, not the policy. Because both prompt sets in table 5 are difficulty-0 and equal in size, the supervised-versus-disjoint comparison is matched by construction, which the earlier version of this measurement was not: it used the mixed three- and four-number reinforcement-learning prompt set against a difficulty-0 fresh set, and so confounded overlap with difficulty. The numbers here supersede it, and they agree with the intervention of section 5.2 where the two overlap, solution distillation at 27 epochs being the checkpoint that intervention holds fixed.

H SPENDING A FIXED BUDGET: THE GROUP-SIZE LADDER

Proposition 3 says total advantage mass falls as $4(G-1)/G^2$ at a fixed generation budget, so enlarging the group buys estimate quality out of quantity. Holding generations per step at 32 and varying $G \in \{2, 4, 8, 16\}$, the accounting is confirmed to within a tenth while the gains, $+0.004$ to $+0.016$ over two seeds, do not follow it. We report this as unresolved rather than as an inversion: at two seeds the ladder is flat within noise, and the two currencies disagree, since total mass M_G falls with G while effective generations $N\bar{\eta}_G$ rise with it, because c_G does. Which one prices a budget is exactly the question proposition 3 answers and the experiment does not settle (appendix F).

I BOUNDARIES: REPLICATION AND TRANSFER

Across a second model family and a second geometry the two quantities come apart again: what reinforcement learning *adds* is ordered by search structure, typed control gaining most in all three settings ($+0.025$, $+0.047$, $+0.023$), while where a run *ends* also depends on where supervision left the checkpoint, by which measure solution distillation wins the other two. Reporting only one is how the same experiment supports either conclusion. Cross-geometry transfer is absent and unpredictable (appendix L).

J THE HEAD-TO-HEAD IN DETAIL

Table 9: Anchor setting: Countdown with Qwen2.5-3B at matched compute, three seeds. Reinforcement-learning quantities on the left, diagnostics on the right. Gain is final minus the accuracy after the first optimizer step. $\bar{\eta}$ is the mean correctness-only signal efficiency over training; the entropy and truncation columns are the quantities that would support the competing explanations of section 5.4 if they varied, which they do not.

	Reinforcement learning				Diagnostics			
Warm-up	AUC	Start	Final	Gain	$\bar{\eta}$	η end/start	Entropy ratio	Truncation
Typed control	0.200 ± 0.007	0.188	0.212 ± 0.007	**+0.025**	0.352	0.92	1.01	$0.61 \to 0.58$
Backtrack-only (Aletheia-style)	0.188 ± 0.016	0.182	0.189 ± 0.015	+0.007	0.376	1.00	1.00	$0.63 \to 0.61$
Solution-only	0.179 ± 0.008	0.190	0.185 ± 0.007	-0.005	0.306	0.99	1.01	$0.02 \to 0.02$
Untyped trace	0.111 ± 0.024	0.096	0.116 ± 0.027	+0.020	0.339	1.00	1.03	$0.69 \to 0.68$
Base, reinforcement only	0.000	0.000	0.000	+0.000	0.038	0.93	0.98	$0.35 \to 0.34$

Generation budget. Re-running the head-to-head at 512 tokens rather than 320, with the supervised adapters held identical, tests whether the trace arms are limited by room to finish. They are not. Typed control's answer-emission rate rises from 0.386 to 0.518 while its accuracy conditional on answering falls from 0.347 to 0.265; the rollouts that newly complete are correct about 3% of the time, against 34.7% for those that completed before. Traces needing more than 320 tokens are essentially never right, so the trace arms' high conditional accuracy is a selection effect, easy problems finishing quickly, rather than capability awaiting room. The typed-minus-solution gap is 0.027 at 320 tokens and 0.023 at 512, so the comparison is not an artifact of the budget.

Prompt curation against a size-matched control. Dropping the prompts on which the warm-up is degenerate raises effective compute from 0.32 to 0.65 of nominal for typed control and from

0.28 to 0.51 for solution distillation. Compared against a random subset of the same size, which separates removing silent prompts from simply making more passes over fewer problems, the paired differences over three seeds are $+0.043$, 0.000, 0.000 for typed control and $+0.008$, -0.008, $+0.016$ for solution distillation. We report this as a null. Our curation is static, computed once from the warm-up, whereas the dynamic sampling of Yu et al. (2025) recomputes it as the policy moves; a static approximation may simply go stale, and the doubling of effective compute it achieves is spent late in training where the accuracy curve has largely flattened.

K THE SELF-JUDGMENT CEILING IN DETAIL

We sample frontier nodes whose oracle label is known, balance EXPAND against ABANDON so that a constant response scores $\kappa = 0$, and ask the base model to choose. Averaged over three seeds, Cohen's κ is 0.021 at 1.5B, 0.023 at 3B and 0.091 at 7B. The aggregate hides a geometry dependence that matters for the rest of the paper: on the high-branching and constraint-satisfaction geometries κ stays near zero even at 7B, while the low-branching arithmetic chain becomes judgeable at 7B ($\kappa = 0.37$). A model that cannot tell a solvable node from a dead one cannot supply its own control labels, which is why the teacher here is an exact verifier rather than the model itself, and the geometry dependence foreshadows the transfer result below.

The geometry fingerprints computed from the same search statistics are distinct and structured: Countdown and Game-of-24 correlate at $+0.56$, and both anti-correlate with the deep constraint-satisfaction geometry at -0.66. That structure exists but does not predict transfer, as the next section reports.

L CROSS-GEOMETRY TRANSFER IN DETAIL

A caveat on N-Queens instance diversity. The five geometries are not equally diverse at the instance level, and one is much less so than we assumed. An N-Queens instance is determined by its board size, with the remaining field in the specification controlling only the order in which the teacher explores; the 600 training prompts and 256 evaluation problems therefore contain three distinct tasks, $N \in \{5, 6, 7\}$, rather than 600 and 256. Accuracies on this geometry are consequently averages over three problems with many samples each, so their sampling error is far larger than a binomial over 256 would suggest, and a policy that finds one valid placement per board size solves the whole set: a reinforcement-learning run on a pool of this geometry reaches 1.000 held-out accuracy within ninety steps. The other four geometries are diverse at the instance level (600, 600, 600 and 1800 distinct tasks). Nothing in the paper turns on this, and we checked: dropping N-Queens moves the transfer contrast from 0.232 against 0.021 to 0.202 against 0.029, still an order of magnitude, and its role elsewhere is as one geometry among five in aggregate diagnostics. We report it because per-geometry numbers on N-Queens should be read as three problems, not as a held-out set, and because it is the reason the prompt-selection experiment of section 5.1 could not be replicated on a second geometry here.

We train one typed-control controller per geometry per seed and evaluate every controller on every geometry, with the base model as the lower bound and the in-domain controller as the upper bound. Averaged over three seeds, the in-domain lift on the four geometries where a controller helps is 0.275, against a mean off-diagonal transfer of 0.012, a factor of twenty-three. On the replication settings, solution distillation reaches a higher final accuracy than typed control on Llama-3.2-3B (0.152 against 0.127) and on Game-of-24 (0.063 against 0.035), in both cases because typed control starts far lower, while typed control has the larger reinforcement-learning gain in every setting. The law we set out to find is not there: the Spearman correlation between failure-landscape feature similarity and transfer is -0.07 over twenty off-diagonal pairs, and a sign classifier built on feature similarity reaches an area under the curve of 0.46, which is chance. We report the matrix as the artifact and the law as a measured negative, now with three seeds rather than the single seed this result previously rested on.

Conditioning the controller on those features rather than on surface tokens does not rescue transfer either. Across three seeds the held-out difference between a feature-conditioned and a token-conditioned controller is -0.017, and in-domain the feature-conditioned variant is far worse, falling from 0.231 to 0.072 on Countdown, 0.263 to 0.035 on the arithmetic chain, and 0.604 to 0.000 on

N-Queens. Encoding the features verbosely inline roughly doubles the sequence length and spends capacity on reproducing feature values, which degrades the policy. A prompt-side or head-based encoding remains untested.