DIRECT SELF-EVOLVING OPTIMIZATION: EVOLVING LLMS WITHOUT CHALLENGER TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-evolving language models improve by generating tasks and learning from their own feedback, but adapting the task generator often requires a separate challenger-training loop. Can we generate tasks adapted to the current solver without explicitly training a challenger? We introduce **D**irect Self-**E**volving **O**ptimization (DEO), which replaces challenger parameter updates with solver-guided task sampling. The KL-regularized challenger objective defines an exponential tilt of a fixed base task distribution. DEO uses this distribution as a sampling target: a frozen LLM generates and mutates tasks, the solver scores them, and an approximate Metropolis selection rule refines the training pool. Only the solver is trained. Theoretically, for an idealized variant that samples exactly from the tilted distribution, and under regularity, local gradient-dominance, and initialization conditions, we show that DEO learns distributionally robust reasoning ability. In experiments, DEO achieves reasoning performance competitive with R-Zero while using over 50% less wall-clock training time, and improves reasoning accuracy over a no-walk ablation. Replacing the task generator with a frozen API-only LLM further improves the local solver, illustrating a capability enabled by removing challenger training.

1 INTRODUCTION

Self-evolving training offers a way to improve large language models (LLMs) by generating their own tasks and learning from the resulting feedback. Recent methods, including R-Zero (Huang et al., 2025), Absolute Zero Reasoner (AZR) (Zhao et al., 2025), and Agent0 (Xia et al., 2025), couple task proposal with task solving to construct an adaptive curriculum. A challenger seeks tasks that challenge the current solver, and the solver learns from the generated tasks during this interaction. R-Zero and Agent0 train separate models for these roles; AZR trains one model to perform both (but still two-stage training). These approaches demonstrate that pretrained models can improve without an externally supplied task–answer dataset for this stage of training.

Training the challenger, however, adds another optimization process to every self-evolving round. A separately trained challenger also requires its own training state and access to trainable model parameters. Yet the solver ultimately consumes the tasks produced by the challenger, rather than the challenger's parameters. This motivates our central question:

> *Can we directly generate tasks lying in the current solver's capability frontier without explicitly training a challenger?*

Our starting point is to view the challenger as a distribution over tasks. For a fixed solver, maximizing challenger reward with a KL penalty to a base task distribution has an exponential-tilt solution. This is a standard property of KL-regularized optimization (Rafailov et al., 2023): tasks with higher challenger reward receive greater relative probability. The closed form specifies a target distribution; it does not by itself make sampling tractable. It nevertheless suggests an alternative to learning a parameterized challenger: adapt the task samples directly. *An adaptive task distribution need not require updates to the generator's parameters.*

We develop this idea into **D**irect Self-**E**volving **O**ptimization (DEO). A frozen LLM generates an initial task pool and proposes mutations conditioned on the current tasks. The solver evaluates these

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.

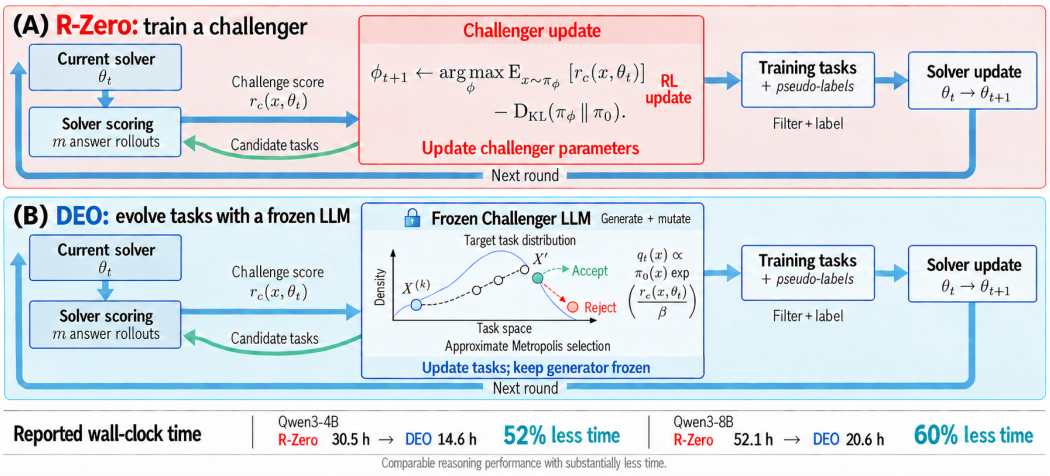


Figure 1: Comparison between traditional challenger training self-evolving method and our DEO. (A) Conventional self-evolving method updates both a task proposer and a solver. (B) DEO keeps the generator frozen and uses solver feedback to select task mutations; only the solver receives parameter updates. DEO uses more than half less wall-clock time than R-Zero while yielding competitive performance.

proposals, and a Metropolis-style selection rule guides the pool toward the current challenger target: if the mutated task looks more challenging to the current solver, we tend to accept it. Through this LLM-based mutation, we produce tasks that lie near the frontier of the current solver's capability. We then use the mutated tasks and solver-produced pseudo-labels to train the solver. Drawing on Metropolis–Hastings sampling (Metropolis et al., 1953; Hastings, 1970), this procedure uses LLM inference to adapt the curriculum while eliminating challenger parameter updates. Because the generator only needs inference access, DEO can also use a closed-source LLM as the task proposer and mutator.

We then ask, *what does DEO actually learn?* We analyze an idealized variant of DEO that samples exactly from the optimal challenger distribution. The analysis shows that DEO learns a model whose worst-case answer uncertainty over a KL neighborhood of the base task distribution is small (Theorem 1). If the initial model's modal answers are mostly correct, this worst-case answer concentration translates into a distributionally robust answer-accuracy bound (Corollary 1). These results explain the role of adversarial task sampling: it learns *distributionally robust reasoning ability*. They do not, however, imply uniform superiority on every evaluation domain.

We conduct experiments on mathematical reasoning. DEO remains competitive with R-Zero, which trains a challenger, while using 52%–61% less wall-clock time. Because DEO replaces challenger parameter updates with LLM inference and selection, it can also use a closed-source model as generator and mutator: replacing the open-source generator with Claude Haiku 4.5 further improves the solver's reasoning ability. Together, these results support solver-guided task sampling as a practical alternative to explicit challenger training.

2 Related Work

Self-evolving training. Self-generated data and feedback provide several routes to improving LLM parameters. Self-Instruct (Wang et al., 2022) generates and filters instruction data for fine-tuning, while STaR (Zelikman et al., 2022) iteratively trains on generated rationales that produce known correct answers. Evol-Instruct (Xu et al., 2023) rewrites seed instructions into more complex training examples, and Self-Rewarding Language Models (Yuan et al., 2024) use model-generated judgments for iterative preference optimization. These methods reduce the need for manual supervision, with different requirements for seed tasks, labels, and feedback.

An interacting-agent perspective on LLM training is discussed by Liu et al. (2024). Closest to our setting, R-Zero (Huang et al., 2025), AZR (Zhao et al., 2025), and Agent0 (Xia et al., 2025) adapt task generation to the evolving solver. R-Zero trains separate challenger and solver models; AZR trains a single model to propose and solve executor-verifiable tasks; and Agent0 co-evolves curriculum and executor agents with tool use. DEO shares their goal of constructing an adaptive curriculum, but replaces proposer parameter updates with solver-guided task sampling. LLM-based task rewriting itself has a precedent in Evol-Instruct; DEO instead grounds task selection in a regularized challenger objective and uses the current solver's evaluations to guide Metropolis-style refinement toward its induced task distribution.

Inference-time refinement and evolution. A complementary line of work improves outputs through repeated LLM inference. Self-Refine (Madaan et al., 2023) alternates answer generation, self-feedback, and revision, while Reflexion (Shinn et al., 2023) stores verbal reflections to guide subsequent attempts without updating model weights. Tree of Thoughts (Yao et al., 2023) searches over intermediate reasoning steps. Evolutionary approaches include Mind Evolution (Lee et al., 2025), which generates, recombines, and refines candidate responses, and FunSearch (Romera-Paredes et al., 2024), which pairs LLM-generated program variants with an automated evaluator. AlphaEvolve (Novikov et al., 2025) extends this approach to larger programs and algorithmic discovery, using LLM-proposed code edits, automated evaluators, and a population of candidate solutions. Its search optimizes programs against user-specified evaluation metrics. Related prompt optimization methods such as GEPA (Agrawal et al., 2025) use reflections on execution traces to evolve system prompts. DEO likewise uses a frozen LLM to propose revisions and feedback to select them, but the evolving objects are training tasks. Its walk constructs a curriculum for subsequent solver parameter updates, connecting inference-based search to self-evolving training.

3 Direct Self-Evolving Optimization

Existing self-evolving methods typically train two roles: a challenger (proposer) that generates a pool of training tasks, and a solver that tries to solve them. The challenger is trained to generate tasks that are more challenging for the solver, while the solver is trained to solve these tasks as correctly as possible.

3.1 From challenger training to a task-sampling target

Let π_θ be the solver policy parameterized by $\theta \in \mathbb{R}^d$, and let $\pi_0(x)$ be a fixed base task distribution induced by a frozen generator and its prompting protocol. Let $r_c(x, \theta)$ score the challenging-ness of task x to the current solver π_θ. In the existing framework such as Huang et al. (2025), the challenger training stage's objective is

$$\max_{\phi} \left\{ \mathbb{E}_{x\sim\pi_\phi} r_c(x, \theta_t) - \beta D_{\mathrm{KL}}(\pi_\phi \| \pi_0) \right\}, \tag{1}$$

where ϕ denotes the parameters of the challenger model (an LLM), θ_t is the solver at round t, and the KL term with coefficient $\beta > 0$ controls departure from the base task law. In words, the challenger is trained to place more probability on tasks with higher challenge scores. However, training this model can be costly. Inspired by DPO (Rafailov et al., 2023), we consider the following distributional formulation of equation 1:

$$q_t \in \arg\max_{q \ll \pi_0} \left\{ \mathbb{E}_{x\sim q} r_c(x, \theta_t) - \beta D_{\mathrm{KL}}(q \| \pi_0) \right\}, \tag{2}$$

The key observation is that q_t admits the following closed form:

$$q_t(x) = \frac{\pi_0(x) \exp(r_c(x, \theta_t)/\beta)}{Z_{\theta_t}}, \qquad Z_{\theta_t} = \mathbb{E}_{x\sim\pi_0} \exp(r_c(x, \theta_t)/\beta). \tag{3}$$

This follows from the Gibbs variational identity (Donsker & Varadhan, 1975). Thus q_t reweights the base distribution, giving exponentially more mass to tasks that are more challenging for the current solver θ_t. As the solver changes, the reward changes and hence q_t changes. If we can directly draw tasks from q_t, then we are able to obtain useful training samples without fitting a new generator at each round.

3.2 LLM-BASED METROPOLIS–HASTINGS SAMPLING

However, directly sampling from q_t is intractable. We thus propose a sampling step based on ***LLM-based mutation and Metropolis–Hastings-style selection***, LLM-MH for short. A frozen LLM (the generator inducing π_0, or even a closed-source model) first generates a seed pool. It then proposes revisions conditioned on each current task, while the frozen current solver evaluates the proposals. A Metropolis–Hastings-style selection rule uses these evaluations to accept or reject each revision.

In detail, each round starts with n_t seed tasks from the frozen LLM. For each task x, we prompt the same LLM to produce a structurally different task x' by some invertible operation:

$$x' \sim \pi_0(\cdot|x, \text{mutation prompt})$$

Denote this prompted proposal kernel by $K_{\mathrm{LLM}}(x' \mid x)$. Reversible task edits can make reverse moves possible, but do not guarantee symmetric proposal probabilities or reversibility with respect to π_0.

The current solver then draws m fresh responses to x'. We accept $x \leftarrow x'$ with probability

$$\hat{\alpha}(x, x') = \min\left\{1, \exp\left(\frac{\hat{r}_c(x', \theta_t) - \hat{r}_c(x, \theta_t)}{\beta}\right)\right\}. \tag{4}$$

where $\hat{r}_c$ estimates r_c from m solver rollouts. This is an approximate selection rule. Exact Metropolis–Hastings targeting q_t would use exact rewards and also include the factor $\pi_0(x')K_{\mathrm{LLM}}(x \mid x')/[\pi_0(x)K_{\mathrm{LLM}}(x' \mid x)]$, which cancels only for a π_0-reversible proposal. The practical walk omits this intractable factor and uses noisy scores and finitely many steps. The exact-sampling assumption in Section 4 is therefore an idealization, not a claimed consequence of running this walk longer. In our experiments, $\widehat{r}_c$ is a triangular agreement score with a batch-level BLEU repetition penalty, so acceptance is evaluated on the whole batch (Appendix E.3).

3.3 SOLVER LEARNING AND THE SELF-EVOLVING LOOP

For each generated task, obtain m responses from the current solver and form a pseudo-label $\widehat{z}_{t,i} = \mathcal{A}_m(y_{t,i,1:m})$, where $\mathcal{A}_m$ is an aggregation rule. The solver remains frozen during task generation and labeling. Supervision is derived from these responses, not from answers supplied by the task generator.

Let $r_s(x, \theta; z)$ denote the solver loss against fixed supervision z. Throughout, r_c is maximized and r_s is minimized. Given $\mathcal{D}_t = \{(x_{t,i}, \widehat{z}_{t,i})\}_{i=1}^{n_t}$, define

$$\widehat{L}_t(\theta) := \frac{1}{n_t}\sum_{i=1}^{n_t} r_s(x_{t,i}, \theta; \widehat{z}_{t,i}), \quad \widehat{J}_t(\theta) := \widehat{L}_t(\theta) + \tau_t \mathcal{R}_s(\pi_\theta, \pi_{\mathrm{ref},t}), \tag{5}$$

where the solver regularizer, reference, and coefficient schedule are independent choices from the challenger's regularization. The goal of solver training is to optimize equation 5 and find θ_{t+1} that satisfies

$$\widehat{J}_t(\theta_{t+1}) \le \inf_\theta \widehat{J}_t(\theta) + \varepsilon_{\mathrm{opt},t}. \tag{6}$$

where $\varepsilon_{\mathrm{opt},t}$ is the optimization error.

Algorithm 1 gives the DEO loop. After each update, θ_{t+1} defines the target and supervision for the next round. The generator requires only inference access throughout.

4 THEORETICAL ANALYSIS: IDEALIZED DEO LEARNS DISTRIBUTIONALLY ROBUST REASONING

What does the model learn from DEO? We answer this question by deriving a generalization bound for DEO. For ease of analysis, we assume that the tasks are sampled independently from the exact optimal challenger q_t instead of approximately sampling from it using LLM mutation and Metropolis selection. That is, at iteration t, $S_t \sim q_t^{n_t}$. The pseudo-labels are still generated by the current solver θ_t via finite rollout and majority voting. Let $P_\theta(a \mid x)$ be the canonical-answer probability, $p_\theta(x) = \max_a P_\theta(a \mid x)$, and $a_t(x)$ the population modal answer of solver θ_t. Instantiate

$$r_c(x, \theta) = 1 - p_\theta(x), \qquad r_s(x, \theta; z) = 1 - P_\theta(z \mid x). \tag{7}$$

Algorithm 1: Direct Self-Evolving Optimization (DEO)

Input: initial solver θ_0, frozen generator π_0, mutation kernel K_{LLM}, rounds T, pool sizes $\{n_t\}$, sweeps L_{w}, temperature β, rollouts m, aggregation $\mathcal{A}_m$.
Output: evolved solver θ_T.

Main routine

for $t = 0, \dots, T-1$ **do**
  Draw seed task pool with n_t tasks from frozen base model: $S_t \sim \pi_0^{n_t}$.
  $S_t \leftarrow$ LLM-MH $(S_t, \pi_{\theta_t}, K_{\mathrm{LLM}}, \beta, L_{\mathrm{w}}, m)$.
  ▷ Approximate the challenger target q_t;
  for $i = 1, \dots, n_t$ **do**
    Draw m responses $y_{t,i,1:m}$ from $\pi_{\theta_t}(\cdot \mid x_{t,i})$.
    $\widehat{z}_{t,i} \leftarrow \mathcal{A}_m(y_{t,i,1:m})$.
  $\mathcal{D}_t \leftarrow \{(x_{t,i}, \widehat{z}_{t,i})\}_{i=1}^{n_t}$.
  Train θ_{t+1} by minimizing $\widehat{J}_t$ in equation 5.
return θ_T.

Sampling subroutine

function LLM-MH$(S, \pi_\theta, K_{\mathrm{LLM}}, \beta, L_{\mathrm{w}}, m)$
  Estimate $\widehat{r}_c(x_i, \theta)$ for each $x_i \in S$ using m solver responses.
  for $\ell = 1, \dots, L_{\mathrm{w}}$ **do**
    for $i = 1, \dots, |S|$ **do**
      $x' \sim K_{\mathrm{LLM}}(\cdot \mid x_i)$.
      ▷ Prompt the frozen LLM to mutate the current task.
      Draw m responses from $\pi_\theta(\cdot \mid x')$.
      Compute $\Delta \leftarrow \widehat{r}_c(x', \theta) - \widehat{r}_c(x_i, \theta)$
      ▷ Finite roll-out estimation of the challenge score.
      $u \sim \mathrm{Uniform}(0, 1)$.
      if $u \le \min\{1, \exp(\Delta/\beta)\}$ **then**
        Replace x_i in S by x' and cache its score.
  return S.

The empirical solver uses

$$\widehat{J}_t(\theta) = \widehat{L}_t(\theta) + \frac{\|\theta - \theta_t\|_2^2}{2\eta_t}. \tag{8}$$

Thus the challenger seeks uncertainty, while the empirical solver minimizes disagreement with the majority of m old-solver responses. The theoretical instance uses KL challenger regularization and a decreasing proximal step size; it does not change the practical GRPO configuration.

Define

$$\Phi_\beta(\theta) = \beta \log \mathbb{E}_{\pi_0} e^{[1 - p_\theta(x)]/\beta}, \qquad \kappa_T = D_{\mathrm{KL}}(q_T \| \pi_0). \tag{9}$$

Write $\Phi_* = \inf_\theta \Phi_\beta(\theta)$ and $E_0 = \Phi_\beta(\theta_0) - \Phi_*$.

Initial answer separation. We assume that the initial solver satisfies, π_0-almost surely,

$$P_{\theta_0}(a_0(x) \mid x) - \max_{a \neq a_0(x)} P_{\theta_0}(a \mid x) \ge \gamma_0 > 0. \tag{10}$$

Our local result derives later-round separation by bounding total parameter movement, under the initialization and error-budget condition below.

Theorem 1 (Worst-case uncertainty bound)**.** *Run the idealized variant of Algorithm 1 in which LLM-MH is replaced by exact sampling, with equation 7–equation 8, n_t conditionally iid tasks per round from q_t at each iteration, m fresh labeling responses per task, and exact empirical minimization ($\varepsilon_{\mathrm{opt},t} = 0$). Assume at most $K \ge 2$ canonical answers, equation 10, and globally L-smooth fixed-answer losses with gradient norm at most $G > 0$. Assume conditional loss-class Rademacher complexity at most $C_{\mathcal{F}}/\sqrt{n_t}$ and the local PL inequality $\|\nabla \Phi_\beta(\theta)\|^2 \ge 2\mu[\Phi_\beta(\theta) - \Phi_*]$ on $\|\theta - \theta_0\| \le r_0 := \gamma_0/(4G)$. Fix $\delta \in (0, 1)$ and use $\eta_t = 18/[\mu(t+s)]$, with s and the initialization/sampling conditions specified in Appendix B.2. With probability at least $1 - \delta$, $\Phi_\beta(\theta_T) \le B_T$ for*

$$B_T := \Phi_* + \underbrace{\mathcal{O}\left(\frac{E_0}{T^2}\right)}_{\text{decaying optimization error}} + \underbrace{\widetilde{\mathcal{O}}\left(\frac{1}{T^2}\sum_{t=0}^{T-1}\frac{(t+1)^2 C_{\mathcal{F}}}{\sqrt{n_t}} + TKe^{-m\gamma_0^2/8}\right)}_{\text{statistical error}}. \tag{11}$$

The notation hides constants and confidence logarithms. Consequently,

$$\boxed{\sup_{D:\, D_{\mathrm{KL}}(D\|\pi_0)\le\kappa_T} \mathbb{E}_D[1-p_{\theta_T}(x)] \le b_T := \frac{1-e^{-\min\{B_T,1\}/\beta}}{1-e^{-1/\beta}}.} \tag{12}$$

The exact bound, including inexact-optimization errors, is given in equation 26.

DEO learns distributionally robust reasoning. The bound controls the final solver's answer uncertainty over the induced KL neighborhood of the base distribution π_0. The bound decreases in terms of the number of generated tasks and rollout budgets. The residual Φ_* accounts for the model class's irreducible uncertainty.

What drives the bound? At the current solver, disagreement with its old modal answer touches and upper-bounds uncertainty. Therefore

$$\mathbb{E}_{q_t}\, \nabla_\theta r_s(x,\theta;a_t(x))|_{\theta=\theta_t} = \nabla\Phi_\beta(\theta_t). \tag{13}$$

The advantage of sampling from the current challenger is that it follows the tilted uncertainty potential, whereas base-law sampling (sampling from base task distribution π_0) generally follows mean uncertainty. This is why DEO can yield distributionally robust guarantee. The proof is deferred to Appendix B.

An initial modal-answer correctness assumption further converts this concentration guarantee into a bound on sampled-answer error.

Corollary 1 (From concentration to answer accuracy)**.** *Under Theorem 1, let $a^*(x)$ be the correct answer, let $g(y)$ extract the answer of response y, and suppose $\Pr_{\pi_0}[a_0(x) \neq a^*(x)] \le \varepsilon_0$. Set $\Delta_T = B_T - \Phi_*$ and $b_* = (1-e^{-\Phi_*/\beta})/(1-e^{-1/\beta})$. On the same probability-$1-\delta$ event,*

$$\boxed{\begin{aligned}&\sup_{D:\, D_{\mathrm{KL}}(D\|\pi_0)\le\kappa_T} \Pr_{x\sim D,\, y\sim\pi_{\theta_T}(\cdot|x)}[g(y)\neq a^*(x)] \\ &\qquad \lesssim \min\left\{1,\ \varepsilon_0 + b_* + \sqrt{\frac{b_*}{2\beta}} + \Delta_T + \sqrt{\Delta_T}\right\}.\end{aligned}} \tag{14}$$

The explicit remainder is given in equation 49. If $\Phi_* = 0$, the bound is $\varepsilon_0 + O_\beta(T^{-1/2})$, capped at one. If $\varepsilon_0 = 0$, the sharper bound is $b_T \le b_* + O_\beta(\Delta_T)$.

Because the modal answer is preserved, the limiting error may exceed the initial modal-answer error ε_0: the guarantee describes robust sharpening of single-sample accuracy toward the initial majority-vote accuracy, not the acquisition of answers the initial model could not produce. The non-vanishing terms are an error certificate, not the minimum robust error over the model class, and hold over the induced radius κ_T rather than a fixed neighborhood. Proofs appear in Appendices B–C; approximate sampling is discussed in Appendix D.

5 Experiments

We instantiate DEO for *mathematical reasoning* and organize the evaluation around four questions: whether challenger training is necessary for competitive performance, how the methods compare in computational cost, whether task refinement adds value and whether DEO can use an API-only generator. Implementation details are in Appendix E.

Models and comparisons. We evaluate Qwen3-4B-Base, Qwen3-8B-Base, and OctoThinker-3B-Hybrid-Base through three self-evolving rounds, following the setup in Huang et al. (2025); Xia et al. (2025). We compare with R-Zero (Huang et al., 2025), which trains a challenger, and *no walk*, which samples tasks independently from the same frozen base model without LLM based Metropolis walk. DEO and no walk each start from 2000 candidates per round and share the labeling, filtering, and nominal solver-update configuration (Appendix F).

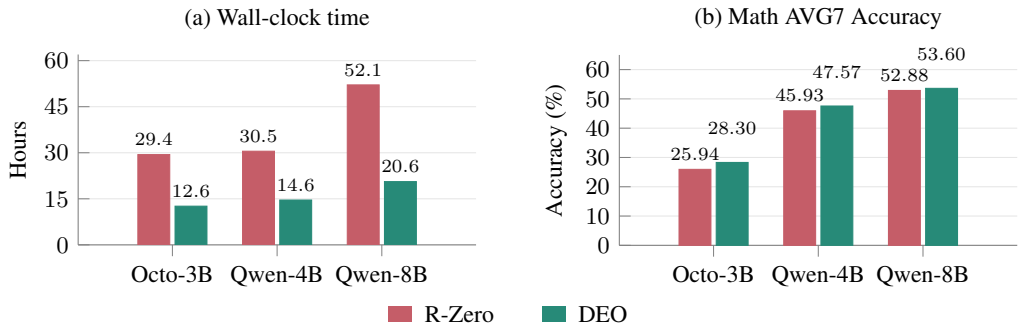


Figure 2: Wall-clock time and iteration-three AVG7 accuracy of DEO and R-Zero. DEO uses 57.1%, 52.1%, and 60.5% less wall-clock time on OctoThinker-3B, Qwen3-4B, and Qwen3-8B, respectively, while achieving competitive accuracy. The Qwen R-Zero runs use larger native question pools (Section 5.1).

Evaluation. The primary comparison reports iteration-three accuracy on MATH-500 (Hendrycks et al., 2021; Lightman et al., 2023), GSM8K (Cobbe et al., 2021), AMC (MAA, n.d.b), Minerva (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), and the 2024 and 2025 editions of AIME (MAA, n.d.a). AVG7 is their unweighted mean; HARD5 averages the last five benchmarks. All methods use the same archived boxed-answer grading protocol. Additional Omni-MATH (Gao et al., 2024) results are reported in Appendix F.2. We additionally measure transfer to MMLU-Pro (Wang et al., 2024), SuperGPQA (M-A-P Team et al., 2025), and BBEH (Kazemi et al., 2025) at all three scales, with their mean denoted AVG3. Table entries are percentages. The evaluation protocol, repeated-question handling, and checkpoint conventions are detailed in Appendix F.

5.1 DEO COMPETES WITHOUT TRAINING A CHALLENGER, USING HALF THE TRAINING TIME

Table 1 compares each pretrained checkpoint with the three methods after the same number of self-evolving rounds. DEO reaches AVG7 scores of 47.57 at 4B and 53.60 at 8B, exceeding R-Zero by 1.64 and 0.72 percentage points. The corresponding HARD5 gains are 1.80 and 0.99 points. On OctoThinker-3B, DEO reaches 28.30 AVG7 versus 25.94 for R-Zero, with MATH-500 improving from 50.20 to 52.60. Thus competitive aggregate performance does not require updating the challenger's parameters in these experiments. The advantage is not uniform over rounds: at 4B, R-Zero leads in iterations one and two (Table 6). On the same eight-H100 node, DEO reduces wall-clock time by 52–61% (Figure 2). No walk takes 5.9, 4.9, and 6.2 hours through iteration three on OctoThinker-3B, Qwen3-4B, and Qwen3-8B, respectively.

General-domain transfer. At 4B, DEO improves AVG3 from 30.13 to 32.17, above no walk's 31.89 but below R-Zero's 32.59. At 8B, DEO attains the highest AVG3, 35.04, versus 34.62 for no walk and 34.30 for R-Zero. DEO has the highest BBEH score at both Qwen scales, while R-Zero leads MMLU-Pro and, on OctoThinker-3B, AVG3. Relative transfer gains thus depend on the model and benchmark.

5.2 LLM MUTATION REFINES THE QUESTION POOL

The no-walk comparison holds the candidate-pool and solver-update configurations fixed. In Table 1, DEO improves AVG7 over no walk by 1.89 points at 4B, 1.82 points at 8B, and 0.79 points on OctoThinker-3B. The Qwen HARD5 gains are 2.47 points at both scales, while no walk retains a slight OctoThinker-3B HARD5 edge (14.18 versus 14.15). The robustness guarantee of Section 4 does not imply dominance on every benchmark. The walk adds generation and solver scoring (up to $L_{\mathrm{w}} = 5$ proposals per task), so this comparison is matched in pool size and solver-update configuration, not total compute.

The trajectory of LLM mutation Figure 3 illustrates three archived trajectories from an instrumented Qwen3-4B run with the same fixed-temperature recipe. The frozen LLM introduces an addi-

Table 1: Base checkpoints and iteration-three results (accuracy, %). DEO and no walk use 2000 initial candidates; the Qwen R-Zero runs retain their native larger pools. Implementation settings are in Appendix E. Base denotes the pretrained checkpoint before self-evolution. Bold marks the best score in each column within each base-model block, including ties.

Method	MATH 500	GSM8K	AMC	Minerva	Olympiad Bench	AIME 2024	AIME 2025	AVG7	HARD5	MMLU-Pro	SuperGPQA	BBEH	AVG3
OctoThinker-3B-Hybrid-Base													
Base	44.60	61.49	17.42	19.12	14.67	**6.67**	0.00	23.42	11.58	8.20	5.33	0.97	4.83
No walk	48.40	73.24	25.00	**27.21**	**18.67**	0.00	0.00	27.50	**14.18**	17.73	9.70	3.30	10.24
R-Zero	50.20	73.09	12.50	**27.21**	15.26	3.33	0.00	25.94	11.66	**19.10**	10.47	**4.00**	**11.19**
DEO	**52.60**	**74.75**	**25.08**	26.47	15.85	3.33	0.00	**28.30**	14.15	17.97	**10.83**	3.70	10.83
Qwen3-4B-Base													
Base	72.60	81.90	48.40	36.80	39.10	**10.00**	**16.60**	43.63	30.18	55.13	27.07	8.20	30.13
No walk	75.80	91.51	52.58	47.43	39.11	**10.00**	3.33	45.68	30.49	58.73	28.67	8.27	31.89
R-Zero	74.00	**91.70**	47.30	**51.80**	40.40	9.60	6.70	45.93	31.16	**59.37**	**29.53**	8.87	**32.59**
DEO	**76.80**	91.43	**57.34**	45.96	**41.48**	6.67	13.33	**47.57**	**32.96**	58.10	28.50	**9.90**	32.17
Qwen3-8B-Base													
Base	70.00	88.93	52.50	43.38	40.74	10.00	3.33	44.13	29.99	60.23	31.33	9.87	33.81
No walk	81.20	93.78	**64.53**	**54.41**	45.19	10.00	13.33	51.78	37.49	62.17	**32.17**	9.53	34.62
R-Zero	**81.40**	93.86	62.81	53.68	45.04	16.67	**16.67**	52.88	38.97	**62.83**	31.63	8.43	34.30
DEO	81.20	**94.16**	62.81	**54.41**	**45.93**	**20.00**	**16.67**	**53.60**	**39.96**	62.23	**32.17**	**10.73**	**35.04**

tional trigonometric composition in a derivative problem (A), changes modular exponentiation into a multiplicative-order problem requiring a minimality argument (B), and replaces prime-divisibility conditions with congruence solution families before adding another constraint (C). The recorded uncertainty scores increase from 0.22 to 0.44, 0 to 0.89, and 0.22 to 0.89, respectively; the last accepted transition in C leaves the score at 0.89. These selected examples show how refinement can add reasoning requirements while increasing the solver's measured uncertainty.

LLM mutation increases in-band questions. Table 2 reports the archived Qwen3-4B counts under the training agreement-band criterion defined in Appendix E.5. In short, in-band counts how many generated tasks fall in the target uncertainty range (the "sweet spot"), excluding extremely hard and extremely easy questions. Pooling rounds one through three, the in-band fraction is 50.72% for no walk and 68.95% for DEO, with 357, 388, and 349 additional in-band candidates, respectively. The advantage remains after filtering: 3682 retained tasks in total versus 2892 for no walk. This shows that solver-guided refinement increases the operational yield of the candidate pool with a frozen generator. These counts use the same cached responses that drove selection, so they measure yield under the walk's own score, not independently verified difficulty or label correctness.

Table 2: Question counts for Qwen3-4B, iterations 1–3. In-band counts precede the pseudo-label and format checks; retained counts follow them. DEO and no walk each start with 2000 candidates.

	In-band			Retained		
Method	Iter 1	Iter 2	Iter 3	Iter 1	Iter 2	Iter 3
No walk	1029	970	1044	977	913	1002
DEO	1386	1358	1393	1246	1208	1228

5.3 DEO WORKS WITH AN API-ONLY GENERATOR

Removing challenger training permits a task generator whose parameters are inaccessible. We replace the local Qwen3-4B generator with Claude Haiku 4.5 for seed generation and mutation. The Qwen3-4B solver still supplies the scores and pseudo-labels; Claude's answers are never training targets. The task budget and training recipe are unchanged (Appendix F). At iteration three, the Claude-generator run reaches 49.60 AVG7 and 35.77 HARD5, compared with 47.57 AVG7 and 32.96 HARD5 for the local-generator DEO run. Gains are concentrated in Minerva, OlympiadBench, and AIME 2024; MATH-500 is unchanged and AMC decreases. Thus a frozen API-only model can improve the task distribution used to train a local solver.

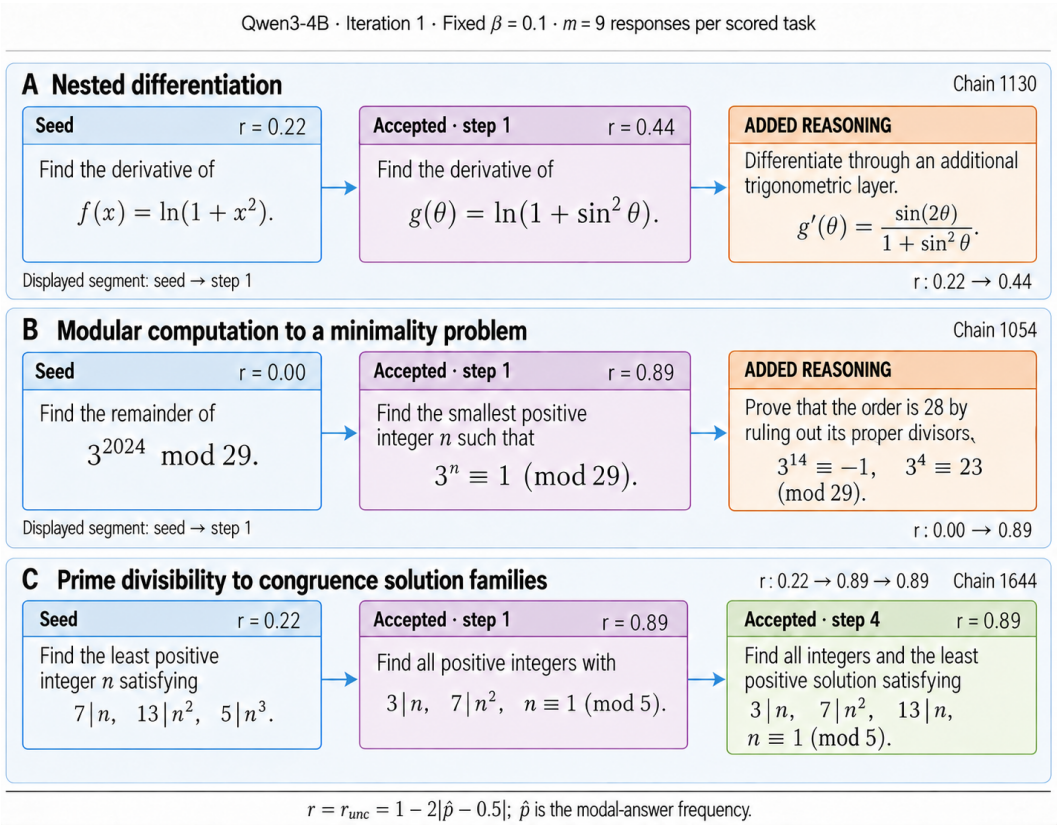


Figure 3: Accepted LLM mutations add reasoning requirements (Qwen3-4B, iteration one). A and B show the seed and first accepted mutation, with the added reasoning explained on the right; C shows both accepted mutations. Each shown mutation raises the uncertainty score and is accepted with probability one.

Table 3: DEO with a Qwen3-4B versus Claude Haiku 4.5 generator/mutator; Qwen3-4B solver, iteration three (%). Bold: better score, including ties. For the Claude one, the pseudo-labels for training were still produced by last-round solver θ_t. The Claude generator+mutator can notably boost the performance, while the R-Zero type of method cannot leverage that.

Generator	MATH 500	GSM8K	AMC	Minerva	Olympiad Bench	AIME 2024	AIME 2025	AVG7	HARD5
Qwen3-4B	**76.80**	91.43	**57.34**	45.96	41.48	6.67	**13.33**	47.57	32.96
Claude	**76.80**	**91.51**	52.50	**50.37**	**42.67**	**20.00**	**13.33**	**49.60**	**35.77**

6 Conclusion

DEO uses the challenger objective to define a sampling target and approximates it with LLM mutation and Metropolis selection, training only the solver. Idealized DEO provably learns distributionally robust reasoning; in practice DEO matches challenger training at about half the wall-clock time, and can use an API-only generator. *Limitations:* The guarantees cover idealized DEO only, while in practical algorithm we use LLM based MH walk to simulate the sampling process. Hence there is a gap between theory and the algorithm we actually use. In addition, the walk operates per question within a round and has no cross-iteration memory, whereas R-Zero-style challengers evolve continuously across iterations. Future work includes memory mechanisms for the walk and applying DEO to agent training. (See more discussion in Appendix A).

AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic data sets, assisting in the writing of proofs and implementing methods. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, cleaning and reformating dataset, supporting qualitative and thematic data analysis, interpreting results. Assisting with translation is not applicable to this work. Additionally, we used generative AI tools for creating or modifying scientific figures or images and editing a research paper to improve readability,. We have reviewed all AI-assisted work. We checked LLM-generated figures and polishing; LLM-generated code was verified and tested for correctness by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. *arXiv preprint arXiv:2507.19457*, 2025. URL https://arxiv.org/abs/2507.19457.

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. URL https://arxiv.org/abs/2110.14168.

Monroe D Donsker and SR Srinivasa Varadhan. Asymptotic evaluation of certain markov process expectations for large time, i. *Communications on pure and applied mathematics*, 28(1):1–47, 1975.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. Omni-MATH: A universal olympiad level mathematic benchmark for large language models, 2024. URL https://arxiv.org/abs/2410.07985.

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. doi: 10.1093/biomet/57.1.97. URL https://doi.org/10.1093/biomet/57.1.97.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024. URL https://arxiv.org/abs/2402.14008.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. *arXiv preprint arXiv:2103.03874*, 2021. URL https://arxiv.org/abs/2103.03874.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-Zero: Self-evolving reasoning LLM from zero data. *arXiv preprint arXiv:2508.05004*, 2025. URL https://arxiv.org/abs/2508.05004.

Mehran Kazemi, Bahare Fatemi, Hritik Bansal, John Palowitch, Chrysovalantis Anastasiou, Sanket Vaibhav Mehta, Lalit K. Jain, Virginia Aglietti, Disha Jindal, Peter Chen, Nishanth Dikkala, Gladys Tyen, Xin Liu, Uri Shalit, Silvia Chiappa, Kate Olszewska, Yi Tay, Vinh Q. Tran, Quoc V. Le, and Orhan Firat. BIG-Bench extra hard. *arXiv preprint arXiv:2502.19187*, 2025. URL https://arxiv.org/abs/2502.19187.

Kuang-Huei Lee, Ian Fischer, Yueh-Hua Wu, Dave Marwood, Shumeet Baluja, Dale Schuurmans, and Xinyun Chen. Evolving deeper LLM thinking. *arXiv preprint arXiv:2501.09891*, 2025. URL https://arxiv.org/abs/2501.09891.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 2022. URL https://arxiv.org/abs/2206.14858.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023. URL https://arxiv.org/abs/2305.20050.

Yang Liu, Peng Sun, and Hang Li. Large language models as agents in two-player games. *arXiv preprint arXiv:2402.08078*, 2024.

M-A-P Team, Xinrun Du, Yifan Yao, Kaijing Ma, et al. SuperGPQA: Scaling LLM evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739*, 2025. URL https://arxiv.org/abs/2502.14739.

MAA. MAA invitational competitions: American invitational mathematics examination (AIME). Mathematical Association of America, n.d.a. URL https://maa.org/maa-invitational-competitions/. Official competition information; accessed September 24, 2026.

MAA. American mathematics competitions (AMC). Mathematical Association of America, n.d.b. URL https://maa.org/student-programs/amc/. Official competition information; accessed September 24, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*, 2023. URL https://arxiv.org/abs/2303.17651.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. doi: 10.1063/1.1699114. URL https://doi.org/10.1063/1.1699114.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. AlphaEvolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025. URL https://arxiv.org/abs/2506.13131.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.neurips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6. URL https://doi.org/10.1038/s41586-023-06924-6.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*, 2023. URL https://arxiv.org/abs/2303.11366.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-Instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*, 2022. URL https://arxiv.org/abs/2212.10560.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024. URL https://arxiv.org/abs/2406.01574.

Peng Xia, Kaide Zeng, Jiaqi Liu, Can Qin, Fang Wu, Yiyang Zhou, Caiming Xiong, and Huaxiu Yao. Agent0: Unleashing self-evolving agents from zero data via tool-integrated reasoning. *arXiv preprint arXiv:2511.16043*, 2025. URL https://arxiv.org/abs/2511.16043.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*, 2023. URL https://arxiv.org/abs/2304.12244.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023. URL https://arxiv.org/abs/2305.10601.

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

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. *arXiv preprint arXiv:2203.14465*, 2022. URL https://arxiv.org/abs/2203.14465.

Andrew Zhao, Yiran Wu, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/9837dc00ff67d176373268ed48042d49-Abstract-Conference.html.

APPENDIX

CONTENTS

A Limitations **14**

B Proof of Theorem 1 **14**

B.1 Definitions and objective compatibility 14

B.2 Initial margin and the localization budget 15

B.3 Uniform sampling error and finite-rollout labels 16

B.4 Relating empirical ERM to a comparison function 17

B.5 Descent along the empirical iterates 17

B.6 Closing the initial-margin induction 18

B.7 Unrolling the recursion and proving the risk bound 19

B.8 Scope without PL 20

C Proof of Corollary 1 **20**

D Extension to Approximate Question Sampling **21**

E Mathematical-Reasoning Implementation **21**

E.1 Concrete challenger reward 22

E.2 Answer statistics and repetition 22

E.3 LLM mutation and the implemented walk 23

E.4 Batch target and approximation scope 23

E.5 Format filtering and GRPO 24

F Experimental Protocol and Supplementary Results **24**

F.1 Experimental Protocol 24

F.2 Additional evaluation on Omni-MATH 25

F.3 First-three-round trajectories 26

G Prompt Templates **27**

G.1 Initial task generation 27

G.2 Solver responses 27

G.3 LLM-based task mutation 28

G.4 Training-data format check 29

G.5 Mathematical-answer grading 29

G.6 General-domain evaluation 30

A LIMITATIONS

Theory versus implementation. Theorem 1 and Corollary 1 cover idealized DEO: exact iid sampling from q_t, the monotone reward $1 - p_\theta$, fresh labeling responses, and a proximal L2 solver update. The implemented walk omits the base-density/proposal ratio, uses noisy scores and finitely many sweeps, uses a triangular score with a repetition penalty and an agreement filter, reuses walk responses as labels, and trains with GRPO. Appendix D shows how a sampler error d_t would enter the bound, but we do not estimate d_t for the LLM walk. The theorem also relies on local PL and a localization budget that we cannot verify for LLMs.

What the guarantee certifies. Supervision comes from the solver's own majority vote, and the proof keeps the initial modal answer fixed. The accuracy bound therefore cannot fall below the initial modal-answer error ε_0. It describes robust sharpening of single-sample accuracy toward majority-vote accuracy, not the discovery of answers the initial model could not produce. The robustness radius κ_T is induced by the final model rather than fixed in advance.

Label and task quality. Pseudo-labels are unverified majority answers. Higher solver uncertainty can reflect ill-posed or ambiguous tasks as well as harder ones. The in-band statistics in Section 5.2 reuse the responses that drove selection, so they are not an independent measure of difficulty or correctness.

Design. The walk operates per question within a round and has no cross-iteration memory, whereas R-Zero-style challengers evolve continuously across iterations. Future work includes memory mechanisms for the walk and applying DEO to agent training.

B PROOF OF THEOREM 1

B.1 DEFINITIONS AND OBJECTIVE COMPATIBILITY

Let g extract a valid canonical answer, with a fixed tie-breaking rule. The answer law and the empirical plurality rule used in the analysis are

$$P_\theta(a \mid x) = \Pr_{y \sim \pi_\theta(\cdot|x)}[g(y) = a], \qquad a_t(x) \in \arg\max_a P_{\theta_t}(a \mid x), \tag{15}$$

$$\widehat{z}_{t,i} = \mathcal{A}_m(y_{t,i,1:m}) \in \arg\max_a \sum_{j=1}^{m} \mathbf{1}\{g(y_{t,i,j}) = a\}. \tag{16}$$

Labels are fixed when optimizing the candidate solver. Its loss against the old population mode is

$$r_s(x, \theta; a_t(x)) = 1 - P_\theta(a_t(x) \mid x), \qquad L_t(\theta) = \mathbb{E}_{x \sim q_t} r_s(x, \theta; a_t(x)). \tag{17}$$

The empirical L2 instance sets

$$\mathcal{R}_s(\pi_\theta, \pi_{\theta_t}) = \tfrac{1}{2}\|\theta - \theta_t\|_2^2, \qquad \tau_t = \eta_t^{-1}, \tag{18}$$

in a fixed parameterization. This is separate from the KL regularizer on the challenger's question law.

The KL-optimal challenger has the explicit exponential-tilt form

$$q_\beta^\theta(x) = \frac{\pi_0(x) e^{[1-p_\theta(x)]/\beta}}{Z_\theta}, \qquad Z_\theta = \mathbb{E}_{\pi_0} e^{[1-p_\theta(x)]/\beta}, \quad q_t = q_\beta^{\theta_t}. \tag{19}$$

For later use, define worst-case uncertainty by

$$\mathrm{Unc}_\kappa(\theta) := \sup_{D:\, D_{\mathrm{KL}}(D\|\pi_0) \le \kappa} \mathbb{E}_{x \sim D}[1 - p_\theta(x)]. \tag{20}$$

The exact tilt induces the radius and achieves the risk

$$\kappa_\beta(\theta) = \frac{\mathbb{E}_{q_\beta^\theta}[1 - p_\theta(x)] - \Phi_\beta(\theta)}{\beta}, \tag{21}$$

$$\mathrm{Unc}_{\kappa_\beta(\theta)}(\theta) = \mathbb{E}_{q_\beta^\theta}[1 - p_\theta(x)] = \Phi_\beta(\theta) + \beta\kappa_\beta(\theta). \tag{22}$$

We prove these identities below. The ball is induced by the final model; it need not be fixed over training, and $q_\beta^{\theta_T}$ need not equal the last training distribution q_{T-1}.

All iterates in this proof are the empirical iterates of the idealized (exact-sampling) variant of Algorithm 1. Write $\Phi = \Phi_\beta$, $c_\theta(x) = r_c(x, \theta)$ and $g_t = \nabla\Phi(\theta_t)$. The losses evaluated against population majority labels and their proximal minimizers below are comparison quantities in the proof, not additional updates performed by the algorithm.

Since $\max_a P_\theta(a \mid x) \geq P_\theta(a_t(x) \mid x)$, the definitions in equation 7 give

$$\begin{aligned} r_c(x, \theta) &\leq r_s(x, \theta; a_t(x)) && \text{for every } \theta, \\ r_c(x, \theta_t) &= r_s(x, \theta_t; a_t(x)), \\ \nabla_\theta r_c(x, \theta_t) &= \nabla_\theta r_s(x, \theta; a_t(x))|_{\theta=\theta_t}\,. \end{aligned} \tag{23}$$

The last equality holds when the old modal answer is locally unique and the answer probabilities are differentiable. Thus the touching upper bound and gradient alignment follow from the specified objectives. The upper bound is valid even outside the initial neighborhood; later we prove that the actual iterates preserve the initial modal answer.

Regularity. Both r_s and r_c lie in $[0, 1]$. Assume that each fixed-answer loss $r_s(x, \cdot; z)$ has an L-Lipschitz gradient of norm at most G, uniformly in x, z. It follows that $r_c(x, \cdot)$ is G-Lipschitz; on the initial-margin neighborhood below it is also differentiable. All question-wise conditions hold π_0-almost surely. Work in $\mathbb{R}^d$ with $\beta > 0$ fixed and $L + G^2/\beta > 0$.

The current optimal challenger therefore gives

$$\begin{aligned} \nabla L_t(\theta_t) &= \mathbb{E}_{q_t}\left[\nabla_\theta r_s(x, \theta; a_t(x))|_{\theta=\theta_t}\right] \\ &= \mathbb{E}_{q_t}\nabla_\theta r_c(x, \theta_t) = \nabla\Phi_\beta(\theta_t). \end{aligned} \tag{24}$$

Sampling from q_t aligns the expected solver-update gradient with the exponentially tilted uncertainty potential. Sampling from π_0 instead generally aligns only with mean uncertainty under the base law.

B.2 Initial margin and the localization budget

For completeness, we retain inexact empirical optimization in the proof; the two-term display in Theorem 1 specializes to $\varepsilon_{\mathrm{opt},t} = 0$. Fix $\delta \in (0, 1)$ and define

$$\begin{aligned} \delta_m^0 &:= \min\{1, (K-1)e^{-m\gamma_0^2/8}\}, \\ e_t &:= \frac{4C_{\mathcal{F}}}{\sqrt{n_t}} + 2\sqrt{\frac{\log(2T/\delta)}{2n_t}} + 2\delta_m^0 + \varepsilon_{\mathrm{opt},t}. \end{aligned} \tag{25}$$

Choose $H = L + G^2/\beta$, $s \geq \max\{3, 36H/\mu\}$, and $\eta_t = 18/[\mu(t+s)]$. The sampling/optimization budget is $e_t \leq \zeta/(t+s)^2$ for some $\zeta > 0$. Set $C_0 = (s-1)E_0 + 3\zeta$ and $\Lambda_T = (T+s-2)(T+s-1)$. The exact finite-sample bound is

$$\begin{aligned} B_T &:= \Phi_* + \frac{(s-2)(s-1)E_0}{\Lambda_T} + \frac{3}{\Lambda_T}\sum_{t=0}^{T-1}(t+s-1)(t+s)e_t \\ &\leq \Phi_* + \frac{C_0}{T+s-1}. \end{aligned} \tag{26}$$

For fixed s, the first excess term is $O(E_0/T^2)$. The weighted finite-task errors are $\widetilde{O}(T^{-2}\sum_{t<T}(t+1)^2/\sqrt{n_t})$, and the common finite-rollout error contributes $O(TKe^{-m\gamma_0^2/8})$. These give equation 11 for exact empirical ERM. An inexact solver adds precisely $3\Lambda_T^{-1}\sum_{t<T}(t+s-1)(t+s)\varepsilon_{\mathrm{opt},t}$; it also consumes the same error and localization budgets.

Write $\Phi = \Phi_\beta$, $E_t = \Phi(\theta_t) - \Phi_*$, and $\mathcal{B}_0 = \{\theta : \|\theta - \theta_0\| \leq r_0\}$, where $r_0 = \gamma_0/(4G)$. The global fixed-answer gradient bound implies that, for $\theta \in \mathcal{B}_0$ and any $a \neq a_0(x)$,

$$\begin{aligned} P_\theta(a_0(x) \mid x) - P_\theta(a \mid x) &\geq \gamma_0 - 2G\|\theta - \theta_0\| \\ &\geq \gamma_0/2. \end{aligned} \tag{27}$$

This establishes unique, unchanged modal answers throughout a fixed neighborhood without assuming anything about future iterates. Since the tilt density relative to π_0 is strictly positive, the same statement holds almost surely under every ideal challenger distribution.

Theorem 1 assumes the PL inequality on $\mathcal{B}_0$, not an a priori margin condition along the trajectory. With $H = L + G^2/\beta$, $s \geq \max\{3, 36H/\mu\}$, $C_0 = (s-1)E_0 + 3\zeta$, its explicit additional condition is

$$\boxed{\frac{72\sqrt{2HC_0}}{\mu\sqrt{s-2}} + 4\sqrt{\frac{18\zeta}{\mu(s-1)}} \leq \frac{\gamma_0}{4G}.} \tag{28}$$

It requires sufficiently small initial excess potential and error budget relative to the initial answer separation. It is a sufficient, conservative local-basin condition, not a consequence of initial margin alone. Its constants are independent of the horizon when E_0, ζ, s are fixed. The empirical optimizer is still unconstrained in $\mathbb{R}^d$; no projection, frozen-label training rule, or extra population update is added. Labels are regenerated from the current solver at every round.

Finite sampling budgets. For a prescribed T, put $A_{T,\delta} = 2C_{\mathcal{F}} + \sqrt{\log(2T/\delta)/2}$. For $\zeta > 0$, the following sufficient choices ensure $e_t \leq \zeta/(t+s)^2$:

$$\begin{aligned} n_t &\geq \frac{36A_{T,\delta}^2(t+s)^4}{\zeta^2}, \\ m &\geq \frac{8}{\gamma_0^2}\log_+\left(\frac{6(K-1)(T+s)^2}{\zeta}\right), \\ \varepsilon_{\mathrm{opt},t} &\leq \frac{\zeta}{3(t+s)^2}. \end{aligned} \tag{29}$$

Here $\log_+(u) = \max\{0, \log u\}$, and positive integer sample sizes are rounded upward. A common n can use the largest displayed task budget. The common rollout count m remains fixed within a finite-horizon run; it must increase with the horizon in an asymptotic family of runs. These are sufficient theoretical budgets, not the settings used in the experiments. The finite-m term is an expected label-error contribution; the proof does not require all sampled labels to be correct simultaneously.

Why decreasing steps alone are insufficient. The crude bound $\|\theta_{t+1} - \theta_t\| \leq G\eta_t$ for exact empirical proximal minimization preserves a neighborhood if $\sum_t \eta_t$ is small, but then it does not yield a vanishing optimization term through the PL contraction. Our schedule has $\sum_t \eta_t = \infty$. It is the decay of the update direction, together with decreasing statistical errors, that makes the actual path length finite. This is proved next rather than assumed.

B.3 Uniform sampling error and finite-rollout labels

Let $\mathcal{F}_{t,m} = \{(x,z) \mapsto r_s(x,\theta;z) : \theta \in \mathbb{R}^d\}$. Its sampling law is a fresh question from q_t together with its m-response pseudo-label. Conditional on the pre-round history, let $\mathfrak{R}_{n_t}(\mathcal{F}_{t,m})$ denote its expected Rademacher complexity. Under the stated capacity bound, use the deterministic envelope

$$\varepsilon_{n,t} = 2C_{\mathcal{F}}/\sqrt{n_t} + \sqrt{\frac{\log(2T/\delta)}{2n_t}}. \tag{30}$$

The theorem assumes $\mathfrak{R}_{n_t}(\mathcal{F}_{t,m}) \leq C_{\mathcal{F}}/\sqrt{n_t}$ uniformly over rounds and admissible histories. This is a capacity assumption; bounded loss alone does not imply it.

For the following one-step estimates, condition on a history with $\theta_t \in \mathcal{B}_0$. The frozen old solver and q_t are then fixed. Each pair $(x_{t,i}, \widehat{z}_{t,i})$ is sampled independently from the law specified in Theorem 1. Let

$$\widetilde{L}_{t,m}(\theta) := \mathbb{E}_{x\sim q_t,\, \widehat{z}\sim H_{t,m}(\cdot|x)} r_s(x,\theta;\widehat{z}),$$

where $H_{t,m}$ is the law of the m-rollout pseudo-label. A conditional uniform deviation bound for the bounded loss class gives

$$\sup_\theta |\widehat{L}_t(\theta) - \widetilde{L}_{t,m}(\theta)| \leq \varepsilon_{n,t}$$

with conditional failure probability at most δ/T. A union bound yields a simultaneous event for all pre-exit rounds with probability at least $1-\delta$. The stopping argument below proves that no exit occurs. Independence across rounds is not required.

For the theoretical disagreement loss, let $a_t(x)$ denote the population modal answer of the old solver. For any competing answer a, define the m independent variables $W_j = \mathbf{1}\{g(y_j) = a\} - \mathbf{1}\{g(y_j) = a_t(x)\}$. They lie in $[-1,1]$ and have mean at most $-\gamma_0/2$. A bounded-sum tail inequality gives $\Pr(\sum_j W_j \geq 0) \leq e^{-m\gamma_0^2/8}$. Taking a union bound over the $K-1$ competitors, including ties as possible errors, gives

$$\Pr[\widehat{z} \neq a_t(x)] \leq \delta_m^0 := \min\{1, (K-1)e^{-m\gamma_0^2/8}\}. \tag{31}$$

Since $r_s \in [0,1]$, replacing the population modal label by $\widehat{z}$ changes its expected loss by at most δ_m^0, uniformly in x, θ. Consequently, with $L_t(\theta) = \mathbb{E}_{q_t} r_s(x, \theta; a_t(x))$,

$$\sup_\theta |\widehat{L}_t(\theta) - L_t(\theta)| \leq \varepsilon_{n,t} + \delta_m^0 =: b_t. \tag{32}$$

This controls disagreement with the old solver's modal answer, without assuming that this answer is correct.

B.4 Relating empirical ERM to a comparison function

Define the comparison function and its minimizer

$$F_t(\theta) := L_t(\theta) + \frac{\|\theta - \theta_t\|_2^2}{2\eta_t}, \qquad z_t := \arg\min_\theta F_t(\theta). \tag{33}$$

The loss L_t is L-smooth and hence L-weakly convex. Thus F_t is M-strongly convex with $M = \eta_t^{-1} - L \geq 1/(2\eta_t) > 0$. The bounded loss and coercive quadratic imply that z_t exists and is unique. The empirical algorithm does not compute z_t.

The same quadratic appears in F_t and $\widehat{J}_t$. Comparing the actual empirical optimizer to z_t on the simultaneous deviation event gives

$$\begin{aligned} F_t(\theta_{t+1}) &\leq \widehat{J}_t(\theta_{t+1}) + b_t \\ &\leq \widehat{J}_t(z_t) + \varepsilon_{\mathrm{opt},t} + b_t \\ &\leq F_t(z_t) + 2b_t + \varepsilon_{\mathrm{opt},t}. \end{aligned}$$

Therefore

$$F_t(\theta_{t+1}) - F_t(z_t) \leq e_t := 2\varepsilon_{n,t} + 2\delta_m^0 + \varepsilon_{\mathrm{opt},t}. \tag{34}$$

Strong convexity also yields $\|\theta_{t+1} - z_t\|_2^2 \leq 2e_t/M$.

B.5 Descent along the empirical iterates

For $q_\beta^\theta = \pi_0 e^{c_\theta/\beta}/Z_\theta$, direct substitution gives the Gibbs identity

$$\mathbb{E}_D c_\theta - \beta D_{\mathrm{KL}}(D\|\pi_0) = \Phi(\theta) - \beta D_{\mathrm{KL}}(D\|q_\beta^\theta). \tag{35}$$

Boundedness and the gradient envelope justify differentiation under the expectation at each visited iterate. The derived identity equation 23 gives

$$g_t = \mathbb{E}_{q_t} \nabla c_{\theta_t} = \mathbb{E}_{q_t}\left[\nabla_\theta r_s(x, \theta; a_t(x))|_{\theta=\theta_t}\right] = \nabla L_t(\theta_t).$$

For any candidate $\theta' = \theta_t + \Delta$, the tilt also gives

$$\Phi(\theta') - \Phi(\theta_t) = \beta \log \mathbb{E}_{q_t} e^{[c_{\theta'}(x) - c_{\theta_t}(x)]/\beta}.$$

Since $c_{\theta'} - c_{\theta_t} \in [-G\|\Delta\|_2, G\|\Delta\|_2]$, the inequality $\log \mathbb{E} e^{s(W - \mathbb{E}W)} \leq s^2(b-a)^2/8$ for $W \in [a,b]$ implies

$$\begin{aligned} \Phi(\theta') - \Phi(\theta_t) &\leq \mathbb{E}_{q_t}[c_{\theta'} - c_{\theta_t}] + \frac{G^2}{2\beta}\|\Delta\|_2^2 \\ &\leq L_t(\theta') - L_t(\theta_t) + \frac{G^2}{2\beta}\|\Delta\|_2^2. \end{aligned} \tag{36}$$

The touching upper bound justifies the second line even when the modal answer changes between iterates.

Write $\Delta_t = \theta_{t+1} - \theta_t$ and $\Delta_t^* = z_t - \theta_t$, and set

$$A := \frac{1}{2\eta_t} - \frac{G^2}{2\beta}, \qquad B := \frac{1}{\eta_t} + L, \qquad M := \frac{1}{\eta_t} - L.$$

The step-size condition gives $A \geq 1/(4\eta_t)$, $B \leq 3/(2\eta_t)$ and $M \geq 1/(2\eta_t)$. Since $F_t(\theta_{t+1}) \leq F_t(\theta_t) + e_t$, equation 36 yields

$$\Phi(\theta_{t+1}) \leq \Phi(\theta_t) - A\|\Delta_t\|_2^2 + e_t.$$

The first-order condition at the comparison point is $\nabla L_t(z_t) = -\Delta_t^*/\eta_t$. Smoothness then gives

$$\|g_t\|_2^2 \leq B^2\|\Delta_t^*\|_2^2 \leq 2B^2\|\Delta_t\|_2^2 + 4B^2 e_t/M.$$

Consequently,

$$\begin{aligned}\Phi(\theta_{t+1}) &\leq \Phi(\theta_t) - \frac{A}{2B^2}\|g_t\|_2^2 + (1 + 2A/M)e_t \\ &\leq \Phi(\theta_t) - \frac{\eta_t}{18}\|g_t\|_2^2 + 3e_t,\end{aligned}$$

using $A/(2B^2) \geq \eta_t/18$ and $2A/M \leq 2$. Substituting equation 34 gives

$$\begin{aligned}\Phi(\theta_{t+1}) \leq \Phi(\theta_t) &- \frac{\eta_t}{18}\|\nabla\Phi(\theta_t)\|_2^2 \\ &+ 6\varepsilon_{n,t} + 6\delta_m^0 + 3\varepsilon_{\mathrm{opt},t}.\end{aligned} \tag{37}$$

B.6 Closing the initial-margin induction

The preceding one-step bounds apply whenever $\theta_t \in \mathcal{B}_0$; they do not require θ_{t+1} to be in the ball. We now prove that exit cannot occur, which avoids assuming the desired future margins. Let $\tau_{\mathrm{exit}} = \inf\{t : \theta_t \notin \mathcal{B}_0\}$. The conditional uniform-deviation argument applies on each pre-exit history. A union bound over these histories gives an event of probability at least $1 - \delta$ on which all the required pre-exit bounds hold. Work on this event.

For $t < \tau_{\mathrm{exit}}$, local PL and equation 37, with $\eta_t = 18/[\mu(t+s)]$, give

$$E_{t+1} \leq \left(1 - \frac{2}{t+s}\right) E_t + 3e_t. \tag{38}$$

If $e_t \leq \zeta/(t+s)^2$, elementary induction gives

$$E_t \leq \frac{C_0}{t+s-1}, \qquad C_0 = (s-1)E_0 + 3\zeta, \tag{39}$$

up to and including a potential first exit: indeed, for $k = t + s$,

$$\left(1 - \frac{2}{k}\right)\frac{C_0}{k-1} + \frac{3\zeta}{k^2} \leq \frac{C_0}{k},$$

because $C_0 \geq 3\zeta$.

We also need an *upper* bound on movement, not merely descent. Smoothness of L_t in equation 36 gives, for every $h \in \mathbb{R}^d$,

$$\Phi(\theta_t + h) \leq \Phi(\theta_t) + \langle g_t, h\rangle + \frac{H}{2}\|h\|^2.$$

Taking $h = -g_t/H$ and using the global lower bound Φ_* yields $\|g_t\| \leq \sqrt{2HE_t}$. This argument uses a touching upper model; it does not assume that Φ is globally smooth across modal ties. The first-order condition for the comparison minimizer z_t implies

$$\|z_t - \theta_t\| \leq \eta_t\big(\|g_t\| + L\|z_t - \theta_t\|\big) \leq \eta_t\|g_t\| + \tfrac{1}{2}\|z_t - \theta_t\|.$$

Together with $\|\theta_{t+1} - z_t\| \le 2\sqrt{\eta_t e_t}$, this gives

$$\|\theta_{t+1} - \theta_t\| \le 2\eta_t\sqrt{2HE_t} + 2\sqrt{\eta_t e_t}. \tag{40}$$

Substituting the error and potential envelopes and summing any prefix,

$$\begin{aligned}\sum_{t<j}\|\theta_{t+1} - \theta_t\| &\le \frac{36\sqrt{2HC_0}}{\mu}\sum_{t\ge 0}\frac{1}{(t+s)\sqrt{t+s-1}} \\ &\quad + 2\sqrt{\frac{18\zeta}{\mu}}\sum_{t\ge 0}(t+s)^{-3/2} \\ &\le \frac{72\sqrt{2HC_0}}{\mu\sqrt{s-2}} + 4\sqrt{\frac{18\zeta}{\mu(s-1)}} \le r_0.\end{aligned}$$

The integral bounds use $s \ge 3$. If a first exit $j \le T$ existed, all terms in this prefix would obey the bounds, contradicting $\|\theta_j - \theta_0\| > r_0$. Hence every empirical iterate stays in $\mathcal{B}_0$. Equation equation 27 now proves mode preservation and gap at least $\gamma_0/2$ for all rounds. The statistical and descent bounds therefore hold throughout the run.

B.7 Unrolling the recursion and proving the risk bound

For $\Lambda_T = (T+s-2)(T+s-1)$, the products in equation 38 telescope:

$$\prod_{j=t+1}^{T-1}\left(1 - \frac{2}{j+s}\right) = \frac{(t+s-1)(t+s)}{\Lambda_T}.$$

Consequently the fully explicit finite-sample bound is

$$\begin{aligned}\Phi(\theta_T) \le \Phi_* &+ \frac{(s-2)(s-1)E_0}{\Lambda_T} \\ &+ \frac{3}{\Lambda_T}\sum_{t=0}^{T-1}(t+s-1)(t+s)[2\varepsilon_{n,t} + 2\delta_m^0 + \varepsilon_{\mathrm{opt},t}].\end{aligned} \tag{41}$$

This is bounded by B_T in equation 26 under the assumed complexity envelope. Equation equation 39 gives $B_T \le \Phi_* + C_0/(T+s-1)$ as well, by applying the same scalar recursion to its deterministic right-hand side. No population optimization step or output averaging is used. We next establish the induced adversarial set and its risk. The Gibbs identity shows that, for any D with $D_{\mathrm{KL}}(D\|\pi_0) \le \kappa_\beta(\theta)$,

$$\mathbb{E}_D c_\theta \le \Phi(\theta) + \beta D_{\mathrm{KL}}(D\|\pi_0) \le \Phi(\theta) + \beta\kappa_\beta(\theta) = \mathbb{E}_{q_\beta^\theta} c_\theta.$$

The last equality follows directly from $\log(q_\beta^\theta/\pi_0) = c_\theta/\beta - \log Z_\theta$. Since q_β^θ itself is feasible, it attains the worst-case value. This proves equation 21 and equation 22.

For $r \in [0,1]$, concavity and the values at zero and one give

$$1 - e^{-r/\beta} \ge r(1 - e^{-1/\beta}).$$

Furthermore,

$$\mathbb{E}_{q_\beta^\theta} e^{-c_\theta/\beta} = \frac{1}{Z_\theta} = e^{-\Phi(\theta)/\beta}.$$

Taking expectations in the preceding scalar inequality therefore proves

$$\mathrm{Unc}_{\kappa_\beta(\theta)}(\theta) \le \frac{1 - e^{-\Phi(\theta)/\beta}}{1 - e^{-1/\beta}}. \tag{42}$$

The right-hand side increases with $\Phi(\theta)$. Since $0 \le \Phi(\theta_T) \le \min\{B_T, 1\}$, substitution gives equation 12 on the same probability-$1-\delta$ event. This completes the proof of Theorem 1. □

B.8 SCOPE WITHOUT PL

Before a possible exit, the one-step argument without PL gives only

$$\sum_{t<T} \eta_t \|\nabla\Phi(\theta_t)\|^2 \le 18[\Phi(\theta_0) - \Phi_*] + 54 \sum_{t<T} e_t, \tag{43}$$

if all these iterates are separately known to remain in the neighborhood. It does not establish last-iterate convergence of Φ or preservation from the initial margin alone. Theorem 1 explicitly retains local PL and the localization budget for both conclusions.

C PROOF OF COROLLARY 1

Modal-answer preservation. Theorem 1, proved using only the initial gap and the explicit localization condition, already gives $a_T(x) = a_0(x)$, π_0-almost surely. No new small-step condition or per-round margin assumption is needed for this corollary. It also does not require every finite-rollout pseudo-label to equal the population mode.

Transferring uncertainty to truth. Define the initial error set $A_0 = \{x : a_0(x) \neq a^*(x)\}$ and the final sampling error under a question law D by

$$\mathrm{Err}_D(\theta_T) := \Pr_{x\sim D,\, y\sim\pi_{\theta_T}(\cdot|x)}[g(y) \neq a^*(x)] = \mathbb{E}_D[1 - P_{\theta_T}(a^*(x) \mid x)].$$

Mode preservation yields the pointwise inequality

$$1 - P_{\theta_T}(a^*(x) \mid x) \le 1 - p_{\theta_T}(x) + \mathbf{1}\{x \in A_0\}. \tag{44}$$

For every D with $D_{\mathrm{KL}}(D\|\pi_0) \le \kappa_T$, Pinsker's inequality (with natural logarithms) gives

$$D(A_0) \le \pi_0(A_0) + \mathrm{TV}(D, \pi_0) \le \varepsilon_0 + \sqrt{\kappa_T/2}.$$

Taking expectations in equation 44 and applying Theorem 1 on its simultaneous event proves

$$\sup_{D:\, D_{\mathrm{KL}}(D\|\pi_0)\le\kappa_T} \mathrm{Err}_D(\theta_T) \le \min\{1, \varepsilon_0 + b_T + \sqrt{\kappa_T/2}\}. \tag{45}$$

This intermediate bound transfers uncertainty to truth. If $\varepsilon_0 = 0$, finite KL implies $D \ll \pi_0$, hence $D(A_0) = 0$ for every feasible D. The sharper error bound is then b_T.

An explicit radius bound. By equation 22, $\Phi_\beta \ge 0$, and the theorem,

$$\beta\kappa_T = \mathbb{E}_{q_\beta^{\theta_T}}[1 - p_{\theta_T}(x)] - \Phi_\beta(\theta_T) \le b_T. \tag{46}$$

Consequently, a bound involving only B_T, β, and ε_0 is

$$\inf_{D:\, D_{\mathrm{KL}}(D\|\pi_0)\le\kappa_T} \mathbb{E}_D P_{\theta_T}(a^*(x) \mid x) \ge \left[1 - \varepsilon_0 - b_T - \sqrt{b_T/(2\beta)}\right]_+. \tag{47}$$

Separating the irreducible term from the vanishing error. For $v \ge 0$, define

$$g_\beta(v) := \frac{1 - e^{-\min\{v,1\}/\beta}}{1 - e^{-1/\beta}}, \qquad C_\beta := \frac{1}{\beta(1 - e^{-1/\beta})}.$$

Then $b_T = g_\beta(B_T)$ and $b_* = g_\beta(\Phi_*)$, since $0 \le \Phi_* \le 1$. For $0 < v < 1$, $g'_\beta(v) = e^{-v/\beta}/[\beta(1 - e^{-1/\beta})] \le C_\beta$, and g_β is constant above one. It is therefore nondecreasing and C_β-Lipschitz on $[0, \infty)$. With $\Delta_T = B_T - \Phi_* \ge 0$, this gives

$$b_T \le b_* + C_\beta\Delta_T. \tag{48}$$

Combining equation 45, equation 46, and $\sqrt{a+b} \le \sqrt{a} + \sqrt{b}$ for $a, b \ge 0$ yields

$$\sup_{D:\, D_{\mathrm{KL}}(D\|\pi_0)\le\kappa_T} \mathrm{Err}_D(\theta_T) \le \min\left\{1,\ \varepsilon_0 + b_* + \sqrt{\frac{b_*}{2\beta}} + C_\beta\Delta_T + \sqrt{\frac{C_\beta\Delta_T}{2\beta}}\right\}. \tag{49}$$

This proves equation 14. Its remainder is explicit in the task counts, labeling rollouts, and optimization errors through $\Delta_T = B_T - \Phi_*$ in equation 26. Under the theorem's budgets, $\Delta_T \le C_0/(T+s-1)$, so the remainder is at most

$$\frac{C_\beta C_0}{T+s-1} + \sqrt{\frac{C_\beta C_0}{2\beta(T+s-1)}} = O_\beta(T^{-1/2}).$$

For $\Phi_* = 0$, $b_* = 0$ and only ε_0 remains in the limiting upper bound. For $\varepsilon_0 = 0$, the earlier bound $\mathrm{Err}_D(\theta_T) \le b_T$ removes the distribution-shift term entirely; then equation 48 gives $b_* + C_\beta \Delta_T$, with an $O_\beta(1/T)$ remainder under the same budgets. These are bounds on the induced KL set for each T, not on a fixed-radius set, and $b_* + \sqrt{b_*/(2\beta)}$ is a consequence of the optimal uncertainty potential, not an optimal true-error value.

Under the base question law itself, no distribution-shift term is needed: $\mathrm{Err}_{\pi_0}(\theta_T) \le \min\{1, \varepsilon_0 + b_T\}$. Here ε_0 bounds the initial *population modal-answer* error, not its single-response error. The corollary concerns sampling accuracy, not a change in modal predictions, and it does not certify the practical triangular-reward GRPO implementation. □

D Extension to Approximate Question Sampling

Theorem 1 assumes the exact optimal question law while retaining finite-m labels and finite-n_t empirical training. However in practice we can only estimate p_θ with finite roll-outs. Here we examine the bound under this realization. For a sampler law $\widetilde{q}_t$ with $\mathrm{TV}(\widetilde{q}_t, q_t) \le d_t$, retain conditional iid questions, fresh independent labels, and $\widetilde{q}_t \ll \pi_0$ so that the initial almost-sure margin applies to its support. Bounded losses add at most d_t to equation 32; replace e_t by $e_t + 2d_t$ everywhere, including the error schedule and localization condition. Under these augmented conditions,

$$B_T^{\mathrm{approx}} := B_T + \frac{6}{\Lambda_T} \sum_{t=0}^{T-1} (t+s-1)(t+s) d_t. \tag{50}$$

The complexity is evaluated under the actual question–label law. Both q_T and the induced radius still refer to the ideal challenger of the final model. Sampler error consumes the same localization budget; it cannot simply be added to the final bound while ignoring its effect on preservation. Constant nonzero sampler error is not covered by the vanishing-error asymptotic rate.

Expected finite-rollout reward. For the theoretical uncertainty reward $r_c = 1 - p_\theta$ in equation 7, define $\rho_{\theta,m}(x) = \mathbb{E}[1 - \max_a \widehat{P}_m(a \mid x)]$, where $\widehat{P}_m$ uses m independent valid answers. Convexity of the maximum and the multinomial frequency error give

$$\begin{aligned} 0 \le r_c(x,\theta) - \rho_{\theta,m}(x) &\le \mathbb{E}\|\widehat{P}_m - P_\theta\|_\infty \\ &\le \sqrt{\frac{1 - \sum_a P_\theta(a \mid x)^2}{m}} \le \frac{1}{\sqrt{m}}. \end{aligned}$$

If a sampler exactly targets $q_t^{(m)} \propto \pi_0 e^{\rho_{\theta_t,m}/\beta}$, the oscillation of its log-density ratio with q_t is at most $h = 1/(\beta\sqrt{m})$, implying

$$d_t \le \tanh\left(\frac{1}{4\beta\sqrt{m}}\right). \tag{51}$$

To see this, a density ratio $w \in [a,b]$ of mean one with $b/a \le e^h$ satisfies $\mathrm{TV} \le (b-1)(1-a)/(b-a)$ by convexity of $|w-1|$. Maximizing over $a \le 1 \le b$ gives $\tanh(h/4)$.

This particular bound assumes the exponential of the expected finite-rollout reward and independent labeling responses. Exponentiating one noisy score, reusing selection responses as labels, filtering the question law, or using a diversity-coupled batch needs further analysis. These conditions are not an error certificate for the practical walk in Appendix E.

E Mathematical-Reasoning Implementation

This appendix instantiates the general algorithm of Section 3 for the mathematical-reasoning experiments. The concrete reward, LLM mutation instructions, finite-step acceptance rule, and solver

training choices are collected here. This practical configuration differs from the ideal-sampling $1-p$ and L2 instance analyzed in Section 4.

Table 4: Configuration of the local-generator DEO runs. The agreement rollouts and GRPO training rollouts serve different purposes.

Setting	Value
Base models	Qwen3-4B/8B-Base; OctoThinker-3B-Hybrid-Base
Candidate pool per round	$N = 2000$, freshly generated
Mutation sweeps	$L_{\mathrm{w}} = 5$
MH temperature	Fixed $\beta = 0.1$
Agreement rollouts per evaluated question	$m = 9$
Agreement sampling	Temperature 1.0, top-$p = 1.0$, top-$k = 40$
Solver response limit	4096 tokens
BLEU distance threshold / repetition weight	0.5 / $\lambda_{\mathrm{rep}} = 10$
Training agreement filter	$\widehat{p} \in [0.3, 0.8]$
Solver training	GRPO; 5 training responses per prompt
Solver KL reference / coefficient	Original base model / 0.01
Additional sampling interventions	None
Reported checkpoints	Iterations 1, 2 and 3

E.1 Concrete challenger reward

For each question x, draw m responses from the frozen current solver. Let $\widehat{p}(x)$ be the largest frequency of an extracted answer, divided by m. We use

$$\widehat{r}_c(x, \theta) = 1 - 2\left|\widehat{p}(x) - \tfrac{1}{2}\right|. \tag{52}$$

For a pool $X = (x_1, \ldots, x_N)$, the score used for selection is

$$\widehat{r}_c(X, \theta) = \sum_{i=1}^{N} [\widehat{r}_c(x_i, \theta) - \lambda_{\mathrm{rep}} r_{\mathrm{rep}}(x_i; X)]_+, \qquad r_{\mathrm{rep}}(x_i; X) = \frac{|\mathcal{N}_i(X)|}{N}, \tag{53}$$

where $\mathcal{N}_i(X)$ contains the BLEU-based neighbors of x_i, including itself. We use $m = 9$ and $\lambda_{\mathrm{rep}} = 10$. The same notation r_c applies to individual questions and complete pools; hats indicate cached finite-rollout estimates. The answer-extraction and neighbor-count definitions follow below.

E.2 Answer statistics and repetition

The frozen pretrained model π_{base} induces the fixed seed question law π_0 through its generation prompt, predefined topic sampling, and output parsing. During each walk, the current solver is frozen. For responses $\xi = (y_1, \ldots, y_m)$, let $g(y) = \perp$ denote an invalid answer extraction and define, for $a \neq \perp$,

$$c_a = \sum_{j=1}^{m} \mathbf{1}\{g(y_j) = a\}, \qquad \widehat{p} = \frac{\max_{a \neq \perp} c_a}{m}, \qquad \widehat{a} \in \arg\max_{a \neq \perp} c_a. \tag{54}$$

If no answer is extracted, set $\widehat{p} = 0$ and $\widehat{a} = \perp$. Invalid responses remain in the denominator m. Answers are grouped by extracted string, without full mathematical-equivalence clustering. Thus $\widehat{p}$ is empirical answer agreement, not verified correctness. The stored pseudo-label comes from the current solver, not from the mutator's proposed solution.

The symmetric neighbor matrix thresholds smoothed sentence-BLEU distances at 0.5. Its row counts include self-neighbors and give

$$r_{\mathrm{rep}}(x_i; X) = |\mathcal{N}_i(X)|/N. \tag{55}$$

These are local neighbor counts, not transitive cluster sizes. Clipping in equation 53 occurs before the sum. A mutation changes the selected question's neighbor row and column and the penalties of all affected neighbors. Unchanged questions retain their uncertainty estimates. The reported settings are $N = 2000$, $m = 9$, $L_{\mathrm{w}} = 5$, $\beta = 0.1$, and $\lambda_{\mathrm{rep}} = 10$.

Algorithm 2: LLM-based mutation and Metropolis selection

Input: initial pool X^0, frozen current solver π_θ, frozen LLM mutation kernel K_{LLM}, temperature β, sweeps L_{w}, rollout count m.
Output: final pool X and cached $(\widehat{p}_i, \widehat{a}_i)_{i=1}^N$.

Set $X \leftarrow X^0$; draw m responses per question and cache $(\widehat{p}_i, \widehat{a}_i, \widehat{r}_c(x_i, \theta))$.
for $\ell = 1, \dots, L_{\mathrm{w}}$ **do**
  Draw a random permutation σ of $\{1, \dots, N\}$.
  for $k = \sigma(1), \dots, \sigma(N)$ **do**
    Prompt the frozen LLM with x_k to sample $x'_k \sim K_{\mathrm{LLM}}(\cdot \mid x_k)$.
    if x'_k is parseable **then**
      Draw m fresh solver responses to x'_k; compute its answer statistics.
      Form $X' \leftarrow X[k \leftarrow x'_k]$ and compute $\Delta \leftarrow \widehat{r}_c(X', \theta) - \widehat{r}_c(X, \theta)$.
      Accept X' and its statistics with probability $\min\{1, e^{\Delta/\beta}\}$; otherwise retain the cached state.
return X and its cached answer statistics.

E.3 LLM MUTATION AND THE IMPLEMENTED WALK

Each round starts with N seed questions from the frozen base LLM. For each current question x_k, we prompt this LLM to produce a structurally different question x'_k by choosing one of five operations: generalize the setting, compose interacting conditions, invert the task, change the requested quantity, or exchange a mathematical concept for its dual. The prompt discourages mere number substitution. Thus mutation is *conditional text generation by an LLM*, rather than a learned challenger update. Denote this prompted proposal kernel by $K_{\mathrm{LLM}}(x'_k \mid x_k)$.

The current solver draws m fresh responses to x'_k; these provide both its uncertainty score and its proposed pseudo-label. Let X' replace x_k by x'_k. We accept with probability

$$\widehat{\alpha}(X, X') = \min\left\{1, \exp\left(\frac{\widehat{r}_c(X', \theta_t) - \widehat{r}_c(X, \theta_t)}{\beta}\right)\right\}. \tag{56}$$

The score difference includes all affected repetition penalties. Acceptance replaces the question and its cached answer statistics; rejection retains the old state. Unchanged questions reuse their responses. We perform $L_{\mathrm{w}} = 5$ randomly ordered sweeps at fixed $\beta = 0.1$.

This rule is motivated by the KL-optimal batch tilt $Q_t(X) \propto \Pi_0(X) \exp(r_c(X, \theta_t)/\beta)$, where $\Pi_0 = \pi_0^{\otimes N}$ and $r_c(X, \theta_t) = \mathbb{E}\widehat{r}_c(X, \theta_t)$. The implemented walk is approximate: it uses noisy scores, omits the base-density/proposal ratio, and runs finitely many steps. Appendix E.4 gives the exact target and acceptance rule. No challenger training, CD correction, or temperature adaptation is used.

E.4 BATCH TARGET AND APPROXIMATION SCOPE

Write $\widehat{r}_c(X, \theta; \xi)$ when the rollout randomness needs to be explicit. The ideal batch reward is its finite-rollout expectation,

$$r_c(X, \theta) = \mathbb{E}_\xi \widehat{r}_c(X, \theta; \xi). \tag{57}$$

It is not the triangular function applied to population probabilities. With $\Pi_0 = \pi_0^{\otimes N}$, the batch challenger objective and optimizer are

$$\mathcal{J}_t(Q) = \mathbb{E}_Q r_c(X, \theta_t) - \beta D_{\mathrm{KL}}(Q \| \Pi_0), \tag{58}$$

$$Q_t^*(X) = \frac{\Pi_0(X) e^{r_c(X, \theta_t)/\beta}}{Z_t}, \qquad Z_t = \mathbb{E}_{\Pi_0} e^{r_c(X, \theta_t)/\beta}. \tag{59}$$

Indeed,

$$\mathcal{J}_t(Q) = \beta \log Z_t - \beta D_{\mathrm{KL}}(Q \| Q_t^*). \tag{60}$$

This optimizes over unrestricted batch laws. The repetition term couples questions, so the optimum generally does not factorize.

Let $\mathcal{K}$ be the batch proposal kernel induced by coordinate selection and K_{LLM}. Exact Metropolis–Hastings acceptance would require

$$\alpha^*(X, X') = \min\left\{1, \frac{\Pi_0(X')\mathcal{K}(X \mid X')}{\Pi_0(X)\mathcal{K}(X' \mid X)} e^{[r_c(X',\theta_t) - r_c(X,\theta_t)]/\beta}\right\}. \tag{61}$$

The implementation omits the base-density/proposal ratio and uses cached noisy scores. The omitted factor cancels if $\mathcal{K}$ is reversible with respect to Π_0; this is not established for the prompted LLM mutator. Symmetry of $\mathcal{K}$ alone is insufficient. Also, $\mathbb{E}e^{\widehat{r}_c/\beta} \neq e^{\mathbb{E}\widehat{r}_c/\beta}$ in general, and a finite walk need not mix. These distinctions prevent interpreting the implemented walk as exact sampling from equation 59. No CD penalty, U-statistic correction, adaptive temperature, memory, or mutation bandit is used in the reported configuration.

The frozen LLM uses the standard V1 prompt: it chooses one of generalize, compose, invert, change-objective, and dualize, conditioned on the current question. Proposals are generated in batches for throughput, with their score changes processed against the current pool. The prompt requests a number, algebraic expression, or finite-set answer, disallows proof and yes/no questions, and rejects mere number swapping as a mutation strategy. The exact generation, mutation, solver, and evaluation prompts are given in Appendix G.

E.5 Format filtering and GRPO

The final dataset contains only pairs with $\widehat{p} \in [0.3, 0.8]$ and a nonempty pseudo-label that pass the implemented format checks. The code applies parsing/regex checks and an LLM `VALID`/`INVALID` judge for template text, prompt leakage, corrupted output, non-math or unsupported question types, and missing answers. The judge is explicitly instructed not to solve the problem; it is not a correctness verifier. For $m = 9$, the band keeps modal counts 3 through 7. The budget $N = 2000$ is measured before filtering, so the retained training size varies by round.

The GRPO implementation maximizes

$$R_{\mathrm{GRPO}}(x, y; \widehat{a}) = 0.9\,\mathbf{1}\{\mathrm{grade}(g(y), \widehat{a}) = 1\} + 0.1\, r_{\mathrm{format}}(y), \tag{62}$$

where the grader checks answer equivalence and r_{format} checks the inherited reasoning/final-answer format. Equivalently, it minimizes the response-averaged loss

$$0.9\, r_s(x, \theta; \widehat{a}) + 0.1\, \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[1 - r_{\mathrm{format}}(y)], \tag{63}$$

with $r_s(x, \theta; \widehat{a})$ interpreted using the same answer-equivalence grader. Here $r_s(x, \theta; \widehat{a}) = 1 - \Pr_{y\sim\pi_\theta(\cdot|x)}[\mathrm{grade}(g(y), \widehat{a}) = 1]$. KL regularization has coefficient 0.01 and uses the original base model as reference; this differs from the previous-iterate L2 instance of the theorem. Training draws fresh responses against the stored pseudo-labels and returns the checkpoint specified in the experimental protocol.

F Experimental Protocol and Supplementary Results

F.1 Experimental Protocol

Models and question generation. We use Qwen3-4B-Base, Qwen3-8B-Base and OctoThinker-3B-Hybrid-Base. For the local-generator runs, the generator and initial solver use the same checkpoint. DEO starts each round with $N = 2000$ parsed candidate questions and applies five mutation sweeps at a fixed $\beta = 0.1$. Initial questions are sampled with temperature 1.0 and top-$p = 0.95$; the standard V1 mutator uses temperature 1.1. Both generation calls have a 1536-token output limit. The generator is frozen at the original pretrained checkpoint. The challenger reward and BLEU repetition penalty are defined in Appendix E, with no additional intervention. Table 4 gives the shared settings. Section 5.3 additionally replaces the 4B generator and mutator with Claude while retaining the Qwen solver.

Solver training and comparisons. The native training jobs use the inherited verl GRPO configuration with 64 prompts per rollout batch, five responses per prompt, learning rate 10^{-6} and a configured maximum of 20 training steps per round. The runner exports and evaluates the

`global_step_15` checkpoint, and uses that checkpoint to initialize the next round. Thus the configured 20-step job budget should not be confused with the checkpoint index. Experiments run on an eight-H100 AzureML node, with separate resources for generation, solver scoring, training and evaluation. The no-walk ablation also generates 2000 candidates per round and uses the same solver labeling, filtering and training recipe, but sets $L_{\mathrm{w}} = 0$. This comparison holds the candidate-pool and solver-update configurations fixed; it does not equalize the additional generation and scoring cost of the walk.

We also include the archived R-Zero (Huang et al., 2025) runs for all three base models. R-Zero trains a separate challenger with RL and uses solver-produced modal answers to supervise solver training. Its solver uses the same nominal GRPO update budget and exported checkpoint index; we compare iterations one through three under the same evaluation protocol. The 2000-candidate restriction applies to DEO and no walk: the Qwen R-Zero runs retain their native, larger question-generation pools. For example, its first three 4B training sets contain 3939, 4312 and 4376 post-filter questions. R-Zero is therefore a method-level baseline, whereas no walk is the matched-pool ablation. This comparison does not establish equal total compute or attribute every difference solely to the walk.

Evaluation. For mathematical benchmarks, we report the first three iterations from the archived runs using the same Claude Haiku boxed-answer grading protocol. AVG7 is the unweighted mean over MATH-500, GSM8K, AMC, Minerva, OlympiadBench, AIME 2024, and AIME 2025; HARD5 averages the last five. All entries are percentages. Main-table means and the differences discussed in the text are calculated from the reported per-benchmark percentages and rounded after calculation. Omni-MATH is evaluated separately in Appendix F.2 and is excluded from AVG7 and HARD5. The evaluation loader repeats AMC and AIME questions 32 times; these repetitions are not additional independent problems. We therefore report the observed scores without treating repeated rows as independent samples for statistical significance. These mathematical comparisons use a common grading protocol without checkpoint selection beyond iteration three.

For general-domain transfer, we evaluate all three base models on MMLU-Pro, SuperGPQA and BBEH using fixed stratified subsets (target size 3000 per benchmark; seed 0), shared across checkpoints. Generation is greedy with at most 4096 new tokens. Multiple-choice answers are graded by option letter, and BBEH uses normalized exact match; this evaluation does not use an LLM judge. AVG3 is the unweighted mean of the three accuracies. This is our project's evaluation protocol, rather than a reproduction of R-Zero's original general-domain evaluation harness.

Closed-source generation. The Claude run uses Claude Haiku 4.5 for both seed generation and mutation, with the Qwen3-4B solver unchanged. The API uses a generation temperature capped at 1.0, whereas the local V1 mutator uses 1.1. Both runs retain the same candidate budget, solver-scoring and labeling roles, filtering, training, and evaluation conventions. Claude responses do not supply the solver's target labels.

Candidate pools and retained datasets. The 2000-task budget is measured before the final filter. The archived DEO datasets contain 1246, 1208, and 1228 retained training tasks in the first three 4B rounds, and 1520, 1508, and 1510 at 8B. These are dataset sizes, not counts of distinct tasks guaranteed to be consumed by the fixed-length solver-training job.

F.2 Additional evaluation on Omni-MATH

We additionally evaluate the base and iteration-three checkpoints on Omni-MATH (Gao et al., 2024), using a fixed 2000-question subset and the same harness across all models and methods. These scores are reported separately from the seven-benchmark comparison in the main text and do not enter AVG7 or HARD5.

Table 5 shows similar accuracy across the trained methods, with model-dependent rankings: no walk leads at 4B, DEO leads at 8B, and R-Zero leads on OctoThinker-3B.

Table 5: Omni-MATH accuracy (%) on the fixed 2000-question subset. All trained methods use iteration three. Bold marks the best trained-method score in each row.

Model	Base	No walk	R-Zero	DEO
OctoThinker-3B	9.50	12.80	**13.10**	12.80
Qwen3-4B	22.00	**25.50**	24.75	24.80
Qwen3-8B	23.55	26.55	26.65	**27.00**

F.3 First-three-round trajectories

Table 6 gives the archived trajectories. The iteration-three rows agree with the primary comparison; intermediate rankings can differ.

Table 6: DEO, R-Zero and the no-walk ablation through iteration three. DEO and no walk use 2000 initial candidates per round; R-Zero uses its native larger question pool. DEO uses fixed $\beta = 0.1$, five sweeps and plain noisy acceptance, without CD or auxiliary interventions. Bold marks the best score among the three methods at each scale and iteration.

		AVG7 (↑)			HARD5 (↑)		
Model	Method	Iter 1	Iter 2	Iter 3	Iter 1	Iter 2	Iter 3
4B	No walk	45.49	47.73	45.68	30.02	32.84	30.49
	R-Zero	**46.46**	**48.13**	45.93	**31.38**	**33.96**	31.16
	DEO	45.09	47.71	**47.57**	29.02	33.18	**32.96**
8B	No walk	51.91	50.64	51.78	38.04	36.28	37.49
	R-Zero	51.96	51.70	52.88	37.88	37.68	38.97
	DEO	**52.01**	**53.39**	**53.60**	**38.22**	**40.02**	**39.96**

G PROMPT TEMPLATES

We give the prompt templates used in the reported experiments. The panels separate message roles, following the presentation of R-Zero (Huang et al., 2025). Text is reproduced from the implementation, with line wrapping adjusted for readability. Placeholder meanings are given below each panel; chat messages are serialized with the corresponding model's chat template. The seed-generation and V1 mutation templates are shared by the local and Claude generators. No walk uses the same seed-generation and solver prompts, without mutation.

G.1 INITIAL TASK GENERATION

Seed-generation prompt

System Message:

You are an expert competition-math problem setter. FIRST, in your private scratch-pad, think step-by-step to design a brand-new, non-trivial problem. Aim for a medium-to-hard competition level.

CRITICAL RULES:
1. The final answer MUST be a SPECIFIC NUMBER, ALGEBRAIC EXPRESSION, or FINITE SET.
2. DO NOT generate "Prove that", "Show that", "Justify", or "Explain why" questions.
3. DO NOT generate questions that ask for True/False or Yes/No answers.
4. Ensure the problem has exactly one unambiguous final answer.
5. LIMIT YOUR SCRATCH-PAD THINKING TO UNDER 50 WORDS! Do not write out the full proof.

THEN, output the problem and answer exactly in this format:
`<question>`
[Write the full problem statement here on one or more lines]
`</question>`
`\boxed{final_answer}`

User Message:

Generate one new, challenging reasoning question now. YOU MUST STRICTLY FOCUS ON: **`{topic}`**.

Placeholder. `{topic}` is sampled uniformly from: Algebra: Polynomial roots and coefficients; Geometry: Triangle centers and circles; Geometry: 3D spatial geometry and volume; Geometry: Coordinate geometry and loci; Number Theory: Diophantine equations; Number Theory: Modular arithmetic and congruences; Combinatorics: Probability and expected value; and Calculus: Limits and derivatives. The generator's boxed answer is not used as the solver's training label.

G.2 SOLVER RESPONSES

Shared solver prompt

System Message:

Please reason step by step, and put your final answer within `\boxed{}`.

User Message:

`{problem}`

Use. `{problem}` is the question being scored, used for GRPO training, or evaluated on a mathematical benchmark. The same system message is used for the $m = 9$ scoring responses, modal-answer pseudo-labeling, fresh training responses, and benchmark evaluation. Training compares fresh answers with the stored pseudo-label; the pseudo-label is not inserted into this prompt.

G.3 LLM-BASED TASK MUTATION

The standard V1 template below is used throughout the reported DEO runs. It asks the frozen LLM to select one structural mutation.

Task-mutation prompt (V1)

System Message:

You are an expert competition-math problem setter. I will provide a seed problem.
Your task is to generate a STRUCTURALLY DIFFERENT problem by applying ONE of the following mutation strategies.

MUTATION STRATEGIES (pick exactly ONE that is NOT already exemplified by the seed):
[A] GENERALIZE — Lift the structure: replace a specific constant with a parameter, OR raise the dimension (2D → 3D, single-variable → multivariable, single equation → system of equations).
[B] COMPOSE — Add a NEW non-trivial second condition that interacts with the existing structure (NOT just a range bound like "with $x > 0$"). The two conditions together must create a real interaction.
[C] INVERT — Given the original answer or output, ask the reader to recover an input or precondition.
[D] CHANGE_OBJECTIVE — Change WHAT is being asked: e.g. "find x" → "count integer solutions", "compute" → "find the smallest n such that ...", "find value" → "find sum of all such values".
[E] DUALIZE — Swap to a dual concept: sum↔product, max↔min, area↔perimeter, gcd↔lcm, addition↔multiplication, distance↔angle.

CRITICAL RULES:
1. DO NOT just swap numbers. If the only edit is a digit change, you have FAILED — restart with another strategy.
2. DO NOT pick the strategy already exemplified by the seed. (e.g. if seed is already 3D, don't pick GENERALIZE→add dimension.)
3. The final answer MUST be a SPECIFIC NUMBER, ALGEBRAIC EXPRESSION, or FINITE SET.
4. NO "Prove that", "Show that", "Justify", "True/False", or "Yes/No" questions.
5. LIMIT scratch-pad reasoning to UNDER 50 WORDS.

Output format (STRICT — all three tags required):
`<strategy>{A|B|C|D|E}</strategy>`
`<question>`
[the NEW mutated problem statement]
`</question>`
`\boxed{final_answer}`

User Message:

Here is the seed problem:
`{seed}`

Pick ONE mutation strategy from `{A, B, C, D, E}` and apply it now. Remember: number-swapping is FAILURE.

Placeholder. `{seed}` is the current accepted question at the selected pool coordinate. Each proposal receives this question alone: solver responses, numerical scores, and weakness-memory notes are not included. Solver scoring and the acceptance decision occur after the proposal is generated. On rejection, the current question is retained for the next mutation attempt.

G.4 TRAINING-DATA FORMAT CHECK

After parsing and band filtering, the implementation uses the following completion prompt to screen the question and its solver-produced pseudo-label.

Format-check prompt

Raw Completion Prompt:

You are a strict validity filter for competition-math problems.

Mark a problem as INVALID ONLY IF you are CONFIDENT one of these specific issues applies:
- Contains placeholder/template text like "[the NEW mutated problem statement]", "final_answer", or "[Write the full problem statement]".
- Contains visible prompt leakage, role tags (Assistant:, User:), HTML/XML tags (<question>, <strategy>), or markdown code fences (```).
- Contains corrupted Unicode garbage, randomly-mixed unrelated languages, or web/chat artifacts.
- Is clearly NOT a math problem.
- Asks for a proof, justification, true/false, yes/no, or open-ended explanation.
- The proposed boxed answer is missing or empty.

When in doubt, the problem is VALID. Do NOT solve the problem. Do NOT reject for being hard, unusual, or having minor formatting quirks.

Problem:
<<<
{q}
>>>
Proposed boxed answer: {ans}

Verdict ("VALID" or "INVALID"):

Placeholders and decoding. {q} is the parsed question and {ans} is the solver's modal-answer pseudo-label. This is sent to the solver endpoint as a raw completion, with temperature 0 and a six-token output limit. It checks format and question type, not mathematical correctness.

G.5 MATHEMATICAL-ANSWER GRADING

For the reported seven-benchmark results, the boxed-answer regrader uses Claude Haiku 4.5 (`claude-haiku-4-5-20251001`), with temperature 0.1 and a four-token output limit. No system message is supplied.

Boxed-answer regrading prompt

User Message:

Hi, there is an answer: {model_answer},and the ground truth answer is: {reference_answer},please check whether the answer is correct or not, and return the **only**Yes or No.

Placeholders and use. {model_answer} is the last boxed answer in the model response, or the literal string `(NO BOXED ANSWER FOUND IN MODEL RESPONSE)` if none is found. {reference_answer} is the last boxed reference answer, falling back to the reference text. Only responses scored incorrect by the initial grader are rechecked; an affirmative verdict promotes their score. The regrader receives these answer strings, not the question or the full model reasoning. This evaluation judge is separate from the training-data format check above.

G.6 General-domain evaluation

MMLU-Pro, SuperGPQA and BBEH use the shared solver system message from Appendix G.2. Their user messages are shown below.

General-domain evaluation prompts

User Message: MMLU-Pro / SuperGPQA:

`{question}`

Options:
`{options}`

Answer with the letter of the single correct option, and put that letter within `\boxed{}`.

User Message: BBEH:

`{question}`

Put your final answer within `\boxed{}`.

Placeholders and use. `{question}` is the benchmark question (the input field for BBEH). `{options}` expands to all supplied choices, one per line, in the form `A. option text`, `B. option text`, and so on. Generation is greedy, with a 4096-token output limit. Multiple-choice outputs are graded by option letter and BBEH by normalized exact match; neither uses the boxed-answer LLM regrader. Omni-MATH uses the shared solver prompt with its problem statement and no additional user-message suffix.