COSTDESCENT: DECOUPLING COST FROM REWARDS FOR EFFICIENT TOOL-INTEGRATED REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool-integrated reasoning (TIR) expands the capabilities of language models but incurs substantial token and tool-use costs. Efficient TIR therefore seeks to retain task performance at reduced cost. Existing approaches often incorporate resource cost into the reward, so that lower cost can be rewarded at the expense of task success. Subsequent refinements temper this interaction but still keep both in one learning signal. Drawing on goal-conditioned RL (GCRL), we instead let task reward alone evaluate outcome, while resource cost enters policy optimization as a goal input. We introduce CostDescent, a goal-conditioned training framework that expresses these goals relative to a task-dependent policy-induced reference. From on-policy performance-cost outcomes, CostDescent constructs a task-aware target reference favoring lower-cost behavior, relabels achieved costs around it, and applies a goal-routed update. We establish sufficient conditions under which this update induces local cost descent. Across six mathematical-reasoning and knowledge-intensive benchmarks, CostDescent achieves the strongest overall task performance and tool productivity. These results establish task-aware cost descent as an effective approach to Efficient TIR and suggest pursuing behavioral targets outside the task reward in broader LLM reinforcement learning.

1 INTRODUCTION

Large language models increasingly extend their internal reasoning with external tools, including search engines, code interpreters, and structured knowledge sources (Yao et al., 2022; Jin et al., 2025; Song et al., 2025; Li et al., 2025e). This tool-integrated reasoning (TIR) enables models to acquire information and execute operations that are difficult to realize through parametric knowledge alone. The resulting capability, however, can require long model-generated traces and repeated tool interactions, increasing both token and tool-use costs. Because the same task can often be solved through trajectories with markedly different cost profiles, task performance alone does not characterize an effective TIR policy. Efficient TIR therefore seeks a favorable performance–cost trade-off: retaining strong task performance while reducing the resources consumed during reasoning.

Efficient TIR often incorporates resource use into RL rewards, using token length, tool use, or the necessity and quality of search decisions as efficiency signals (Wang et al., 2025; Wu et al., 2025; 2026). This lets standard policy optimization discourage unnecessary computation, but can also suppress reasoning or tool calls that aid task completion.

Recent methods temper this pressure by gating efficiency optimization on successful rollout groups or trajectories (Dai et al., 2025; Lee et al., 2026; Song & Zheng, 2025), modulating it by confidence (Wu et al., 2025; Huang et al., 2025; Qi et al., 2026), or adjusting cost-reward sensitivity through curricula (Chen et al., 2026b;c). These refinements temper the interaction between task outcome and efficiency, but continue to optimize them through a shared reward or advantage.

Inspired by goal-conditioned RL (GCRL), which represents a desired outcome as a condition for the policy (Schaul et al., 2015; Andrychowicz et al., 2017), we let task reward alone evaluate task outcome and supply resource cost to the policy as a goal, decoupling the two learning signals. Realizing cost descent through such goals presents two technical challenges: a) The cost goals must be task-dependent while remaining aligned with the evolving policy; b) The learning rule must use these goals to steer policy updates toward lower-cost, task-effective behavior over iterations.

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.

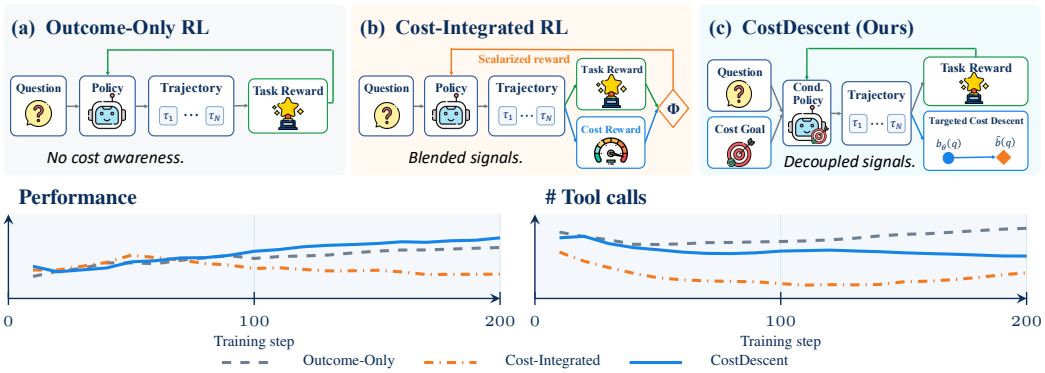


Figure 1: **Top:** Three RL paradigms for Tool-Integrated Reasoning (TIR). Existing approaches either omit the cost signal or combine task performance and cost into a single scalar. CostDescent instead treats cost as a separate goal and optimizes for targeted cost descent. **Bottom:** Stylized training dynamics of CostDescent. It reduces excessive tool-call costs while maintaining performance comparable to outcome-only RL, thereby improving tool productivity.

To address these challenges, we present **CostDescent**, a cost space goal-conditioned training framework for efficient tool-integrated reasoning. For the first challenge, we introduce **Relative Cost Conditioning** and express an absolute cost goal as a policy-induced cost reference plus a signed offset supplied to the policy. The reference is the policy's expected cost at zero offset for each task, so the relative goal coordinates stay naturally aligned with the policy as it evolves. Then *Cost-Aware Rollout* collects on-policy performance–cost evidence under zero and exploratory goals. Our core mechanism, **Task-Aware Cost Descent**, addresses the second challenge by utilizing the on-policy evidence to define a target cost reference that favors lower-cost outcomes among task-effective trajectories, and to relabel achieved cost goals relative to that target. A *Goal-Routed Update* reinforces positive-advantage trajectories at their relabeled goals while retaining the rollout goals for other trajectories. Under sufficient local conditions, this positive-advantage update moves the policy-induced reference toward the target. The updated policy supplies the reference and evidence for the next iteration, forming a closed optimization loop.

We evaluate CostDescent on six mathematical-reasoning and knowledge-intensive benchmarks. CostDescent delivers the strongest overall results in both task performance and tool productivity. Ablations attribute this cost descent to the goal input, and attribution analysis shows how the policy responds to it. These results show that cost goals can improve the performance–cost trade-off without incorporating resource cost into the task reward.

Our contributions are threefold:

- Drawing on goal-conditioned RL, we optimize Efficient TIR in cost space: task reward alone evaluates the outcome, while resource cost enters policy optimization as a goal input rather than a reward term, decoupling the two learning signals.
- We develop CostDescent, a concrete framework for cost-space optimization through relative cost conditioning, cost-aware rollout, task-aware cost descent, and goal-routed updates. We characterize its local cost-descent direction and motivate behavioral targets outside the task reward in broader LLM reinforcement learning.
- Across six mathematical-reasoning and knowledge-intensive benchmarks, CostDescent achieves the strongest overall task performance and tool productivity, improving the performance–cost trade-off for Efficient TIR.

2 Related Work

Agentic Reinforcement Learning Agentic RL trains language models to reason through interactions with external tools. Building on ReAct's reasoning–acting interface (Yao et al., 2022), Search-

R1 and R1-Searcher learn search policies (Jin et al., 2025; Song et al., 2025), while ToRL and ToolStar enable code execution and heterogeneous tool use through RL (Li et al., 2025e; Dong et al., 2025). Subsequent work improves multi-turn training stability (Xue et al., 2026), credit assignment (Feng et al., 2026), and exploration (Dong et al., 2026). These advances strengthen tool-use capabilities; our focus is reducing their token and tool costs without adding cost to the reward.

Efficient Tool-Integrated Reasoning Efficient TIR commonly augments task rewards with signals for tool-use efficiency or search quality (Wang et al., 2025; Wu et al., 2025; 2026; Qi et al., 2026). ET-Agent schedules tool-use and reasoning-length rewards through a curriculum (Chen et al., 2026c), while EAPO adapts tool-cost shaping to problem difficulty (Chen et al., 2026b). Other approaches learn efficient behavior through supervised data, preference learning, or auxiliary on-policy supervision (Qian et al., 2025; Chen et al., 2026d;a). For reasoning models without tools, length-control methods regulate thinking tokens through budgets or mode selection (Han et al., 2025; Aggarwal & Welleck, 2025; li et al., 2026). CostDescent instead decouples cost from reward: token and tool costs enter optimization as a goal input and are reduced through task-aware cost descent.

Goal-Conditioned RL and Hindsight Relabeling Hindsight Experience Replay reuses trajectories under goals they actually achieve (Andrychowicz et al., 2017), and Hindsight Policy Gradients extends this principle to policy-gradient learning (Rauber et al., 2019). Hindsight relabeling also supports instruction following (Zhang et al., 2023) and interactive language agents (Gaven et al., 2024; Li et al., 2026). Complementary goal curricula adapt attempted goals to the agent's current capabilities (Florensa et al., 2018). Related return-conditioned methods derive commands from recorded experience (Srivastava et al., 2019; Chen et al., 2021). CostDescent applies hindsight to achieved token and tool costs while retaining the original task. Specifically, it relabels costs relative to a target reference estimated from the current rollout group's performance–cost outcomes. These relabeled goals direct positive-advantage updates toward lower-cost, task-effective behavior while rollout goals retain exploratory coverage.

3 Problem Formulation

3.1 Efficient Tool-Integrated Reasoning

Tool-integrated reasoning (TIR) generates an answer with an associated resource cost. Given a task instance q, a policy generates a trajectory τ comprising model-generated text, tool interactions, and a final answer. The task reward $R(q, \tau)$ evaluates the quality of this behavior, while its resource usage is described by

$$c(\tau) = \big(c_{\text{think}}(\tau), c_{\text{tool}}(\tau)\big) \in \mathbb{R}^2_{\geq 0}, \tag{1}$$

where c_{think} measures model-generated token usage and c_{tool} measures tool usage. Different trajectories can solve the same task with different cost profiles. Efficient TIR seeks task-effective reasoning behavior with lower resource use.

3.2 Cost Goals for Efficient TIR

Goal-conditioned RL represents a family of behaviors through an explicit goal input (Schaul et al., 2015; Andrychowicz et al., 2017). We adopt this interface for resource cost. An absolute cost goal $g^{\text{abs}} \in \mathbb{R}^2$ specifies a desired cost profile and enters the policy as a goal input, while $R(q, \tau)$ alone evaluates task quality. The task reward thus determines which behavior is reinforced, and the cost goal indexes the resource profile of that behavior.

To let resource use adapt to task difficulty and tool demand at inference, we represent task-to-cost routing by a *cost reference* $b : \mathcal{Q} \to \mathbb{R}^2$, where $b(q)$ anchors the absolute cost goal for task q. The reference alone does not identify an efficient resource profile, so the learning rule must favor lower-cost goals while preserving task quality.

4 Methodology

We present **CostDescent** as an iterative optimization procedure in cost space (Fig. 2). *Relative Cost Conditioning*(§4.1) defines task-dependent goals around the policy-induced cost reference,

while *Cost-Aware Rollout*(§4.2) gathers on-policy performance–cost evidence. *Task-Aware Cost Descent*(§4.3) uses this evidence to construct a target reference favoring lower-cost, task-effective outcomes and relabels achieved costs relative to it. *Goal-Routed Update*(§4.4) routes positive-advantage trajectories to relabeled goals and others to rollout goals. We establish the optimization direction in cost space under sufficient local conditions.

4.1 Relative Cost Conditioning

For Efficient TIR, appropriate absolute cost goals depend on the task. An intuitive approach pairs an absolute-cost-conditioned policy with a separate cost-reference router $b_\phi(q)$, which supplies a task-dependent reference during RL rollouts and inference. Yet policy updates change the cost realized at that reference, allowing the router and policy to drift out of alignment.

We instead derive the reference from the policy itself. Let $\pi_\theta(\tau \mid q, g)$ denote a policy family with a two-dimensional goal input g. We take $g = 0$ as the default condition. The policy's expected-cost map and corresponding policy-induced cost reference are defined as

$$\mu_\theta(q, g) := \mathbb{E}_{\tau \sim \pi_\theta(\cdot|q,g)}[c(\tau)], \qquad b_\theta(q) := \mu_\theta(q, 0). \tag{2}$$

The mapping $q \mapsto b_\theta(q)$ acts as an implicit router whose reference remains synchronized with the policy that realizes it.

We then express absolute targets in *relative cost-goal coordinates*:

$$g = g^{\mathrm{abs}} - b_\theta(q), \tag{3}$$

Each coordinate of $g \in \mathbb{R}^2$ specifies a signed cost offset in its axis's units. With the policy's behavior at $g = 0$ defining the task-dependent origin, this relative parameterization retains the expressivity of aligned absolute-cost conditioning.

Proposition 1 (Expressive sufficiency under aligned reference routing, informal)**.** *Every relative-cost-conditioned policy with finite expected cost at $g = 0$ admits an aligned absolute-cost representation that preserves trajectory distributions at corresponding goals $g^{\mathrm{abs}} = b_\theta(q) + g$.*

Appendix B.1 gives the formal statement and proof.

We implement this interface using SVD-LoRA (Zhong et al., 2024), which injects g through affine goal-dependent low-rank updates. At deployment, $g = 0$ removes the goal-dependent terms, and the remaining update can be merged into the backbone without additional inference-time computation. Appendix A.2 gives the full parameterization.

4.2 Cost-Aware Rollout

To collect performance–cost evidence across cost goals, we split each rollout group evenly between trajectories sampled at $g = 0$ and trajectories sampled at exploratory relative goals (Fig. 2(a)). The former records the current default-policy behavior; the latter probes how the policy behaves under different cost goals. The exploratory goals are drawn from a zero-centered uniform box whose half-width along each cost dimension tracks an exponential moving average of the corresponding achieved-cost standard deviation over preceding batches. We denote the goal used to sample τ_i by g_i^{roll}; full sampling details are given in Appendix A.3.

4.3 Task-Aware Cost Descent

To optimize the policy-induced cost reference for lower-cost, task-effective behavior, we derive its target from on-policy performance–cost outcomes.

Task-Aware Target Construction For each task q, let $\{\tau_i\}_{i=1}^G$, with $G > 1$, be a group of trajectories sampled from the old policy. Each trajectory yields an achieved cost $c_i = c(\tau_i)$ and task reward $R_i = R(q, \tau_i)$, from which we compute GRPO (Shao et al., 2024) advantage A_i. Together, these observations determine a *target cost reference* $\hat{b}(q)$, held fixed during the corresponding update.

Let S_i indicate acceptance under a task-performance criterion, and order c_i lexicographically by tool cost and then token cost. Each accepted trajectory j defines $\mathcal{B}_j = \{i : c_i \preceq c_j\}$; their average

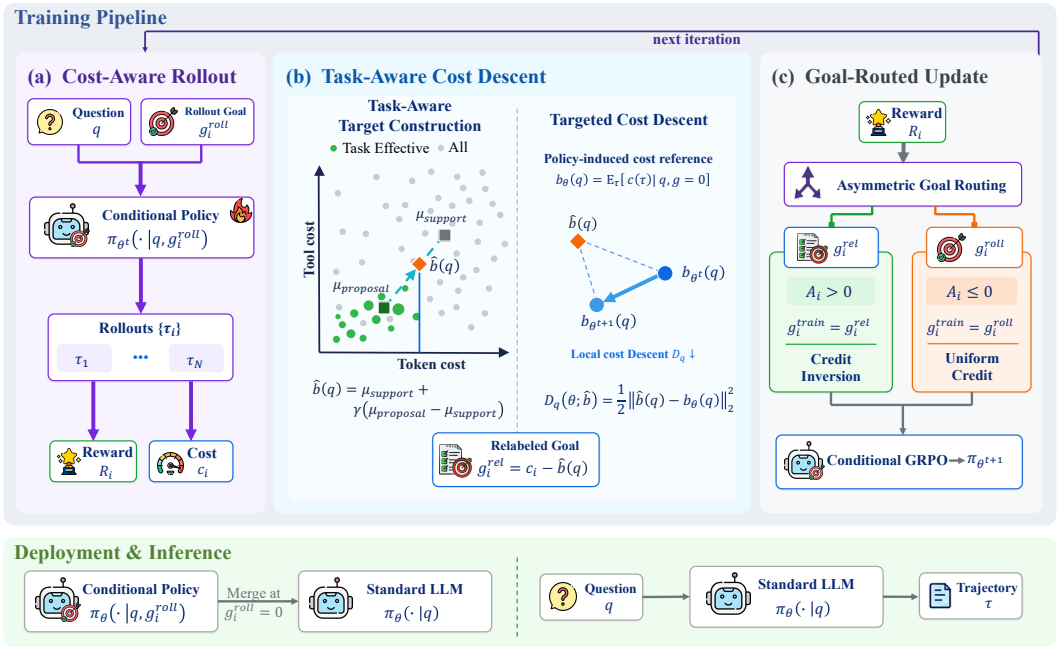


Figure 2: **Overview of CostDescent.** Cost-aware rollout samples trajectories at the default and exploratory cost goals, task-aware cost descent constructs a lower-cost target from on-policy outcomes and relabels achieved costs around it, and goal-routed updates move the default cost reference toward that target. At deployment, the default goal $g = 0$ is merged into the parameters for standard inference.

membership $h_i = (\sum_j S_j \mathbf{1}[i \in \mathcal{B}_j])/(\sum_j S_j)$ favors lower-cost trajectories, including ties. We combine acceptance, positive task advantage, and this cost preference in the proposal:

$$w_i = S_i[A_i]_+ h_i, \qquad \mu_{\text{proposal}} = \frac{\sum_i w_i c_i}{\sum_i w_i}, \tag{4}$$

where $[A_i]_+ = \max(A_i, 0)$. The proposal emphasizes task-effective trajectories with fewer tool calls and, among equal tool calls, fewer tokens.

We construct the target reference by interpolating from the rollout support mean toward the cost proposal according to whether task-performance acceptance remains concentrated among lower-cost trajectories. For groups with a valid proposal, we define

$$\gamma = \min\left(1, \frac{G \sum_i h_i S_i}{(\sum_i h_i)(\sum_i S_i)}\right), \quad \hat{b}(q) = \mu_{\text{support}} + \gamma\big(\mu_{\text{proposal}} - \mu_{\text{support}}\big). \tag{5}$$

The factor γ attenuates movement when task acceptance is less concentrated among lower-cost trajectories. Here, μ_{support} is the mean cost over the estimator support; exclusions and fallbacks are specified in Appendix A.4.

The target reference then relabels each achieved cost by subtracting the reference gap $\delta_b(q) = \hat{b}(q) - b_\theta(q)$ from its achieved relative goal $c_i - b_\theta(q)$:

$$g_i^{\text{rel}} = \underbrace{\big(c_i - b_\theta(q)\big)}_{\text{achieved relative goal}} - \underbrace{\delta_b(q)}_{\text{cost-reference gap}} = c_i - \hat{b}(q). \tag{6}$$

Targeted Cost Descent To characterize whether the positive-advantage update at these relabeled goals moves $b_\theta(q)$ toward $\hat{b}(q)$, consider the cost-reference-gap potential

$$\mathcal{D}_q(\theta; \hat{b}) := \frac{1}{2}\left\|\hat{b}(q) - b_\theta(q)\right\|_2^2. \tag{7}$$

Because the relabeled goals and $g = 0$ share the backbone and goal-independent SVD-LoRA parameters, an update at a relabeled goal also changes default-condition behavior. The affine goal path controls the local variation of this response across goals.

Proposition 2 (Local contraction of the cost-reference gap, informal)**.** *Under the sufficient aggregate response and cross-goal coupling conditions formalized in Appendix B.2, the ideal positive-advantage hindsight update is a strict descent direction for $\mathcal{D}_q(\theta; \hat{b})$. Consequently, every sufficiently small step contracts the cost-reference gap.*

Appendix B.2 gives the formal statement and proof and Appendix B.3 validates the movement empirically. The preference scores typically place the target reference on the lower-cost side of the current rollout support, so contracting this gap yields task-aware cost descent. The updated default policy then supplies the reference and evidence for the next rollout group, closing the training loop.

4.4 Goal-Routed Update

Asymmetric Goal Routing The task-aware target reference enters policy optimization through hindsight-relabeled goals, while the advantage estimator uses task reward alone. Given the GRPO (Shao et al., 2024) advantage A_i, the training goal is

$$g_i^{\text{train}} = \begin{cases} g_i^{\text{rel}}, & A_i > 0, \\ g_i^{\text{roll}}, & A_i \le 0. \end{cases} \tag{8}$$

The positive branch reinforces achieved behavior under its relabeled goal and forms the update analyzed in Proposition 2. The non-positive branch retains the rollout goal; negative advantages therefore suppress lower-reward behavior under the condition that generated it.

Hindsight Credit Inversion Relabeling moves a positive-advantage trajectory to the goal under which it is reinforced. HPG (Rauber et al., 2019) corrects such goal shifts with relabeled-to-original likelihood ratios multiplied over the trajectory prefix, and in practice uses a weighted estimator that lowers variance at the cost of bias. Under outcome supervision, the correction view weakens further: the GRPO advantage A_i is shared by all tokens, so a token-level ratio no longer corrects a token-specific credit estimate but decides how A_i is distributed across tokens (Wang et al., 2026). We therefore treat the hindsight ratio as a credit weight. Let x_{it} denote the token history preceding action a_{it}, and

$$\omega_{it} = \frac{\pi_{\text{old}}(a_{it} \mid x_{it}, g_i^{\text{train}})}{\pi_{\text{old}}(a_{it} \mid x_{it}, g_i^{\text{roll}})} \tag{9}$$

which measures how strongly the relabeled goal already favors a_{it} under the old policy. Weighting credit by ω_{it} would concentrate it on tokens that the relabeled goal already makes likely, the positive-token rich-get-richer pattern that ASPO (Wang et al., 2026) identifies for the PPO ratio. We instead invert it, shifting credit toward tokens that the relabeled goal has not yet made likely, over the valid policy tokens $\mathcal{T}_i$ of each trajectory:

$$u_{it} = \text{clip}\big(\omega_{it}^{-1}, \ell_+, u_+\big), \qquad z_{it} = \frac{u_{it}}{|\mathcal{T}_i|^{-1} \sum_{s \in \mathcal{T}_i} u_{is}}, \qquad \widetilde{A}_{it} = A_i z_{it}. \tag{10}$$

Clipping bounds token-weight disparity, and trajectory-local mean-one normalization preserves $\sum_{t \in \mathcal{T}_i} \widetilde{A}_{it} = |\mathcal{T}_i| A_i$: the relabeled goal changes where credit lands, not how much.

For policy optimization, we use the standard PPO ratio computed at the training goal, $\rho_{it}(\theta) = \pi_\theta(a_{it} \mid x_{it}, g_i^{\text{train}}) / \pi_{\text{old}}(a_{it} \mid x_{it}, g_i^{\text{train}})$, to which we apply the usual PPO clipping.

Final Objective Based on the ARPO (Dong et al., 2026) objective, we combine asymmetric goal routing with hindsight credit inversion, yielding the following objective:

$$\begin{aligned} \mathcal{J}_{\text{CostDescent}}(\theta) = \mathbb{E}_{q \sim \mathcal{D}, \{\tau_i\}_{i=1}^{G} \sim \pi_{\text{old}}(\cdot \mid q, g_i^{\text{roll}})} \Bigg[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|\mathcal{T}_i|} \sum_{t \in \mathcal{T}_i} \\ \min\Big(\rho_{it}(\theta) \widetilde{A}_{it}, \text{clip}(\rho_{it}(\theta), 1-\epsilon, 1+\epsilon)\, \widetilde{A}_{it} \Big) \Bigg] - \lambda_{\text{reg}} \mathcal{R}_{\text{SVD}}(\theta), \end{aligned} \tag{11}$$

where $\widetilde{A}_{it}$ is the reweighted GRPO advantage computed from equation 10, $\mathcal{R}_{\text{SVD}}$ is a small SVD-LoRA orthogonality regularizer with weight λ_{reg}. Full training algorithm in Appendix A.1.

Table 1: **Main results on six benchmarks.** † and ∗ indicate results retrained and re-evaluated in our environment, respectively; the remaining baseline results are taken from ET-Agent (Chen et al., 2026c). Top two results are highlighted in **bold** and underlined. Variance is reported in App. D.3.

Method	Mathematical-Reasoning Tasks						Knowledge-Intensive Tasks						Avg.	
	AIME24		AMC23		MATH500		2Wiki		Bamb		MSQ			
	LJ	TP	LJ	TP	LJ	TP	F1	TP	F1	TP	F1	TP	P	TP
Direct Inference														
Qwen2.5-7B-it	10.0	-	30.0	-	57.2	-	18.9	-	24.5	-	8.5	-	24.9	-
Single-TIR Methods														
Search-o1	10.0	10.0	37.5	37.5	65.4	65.2	42.9	26.2	48.5	33.9	13.3	10.9	36.3	30.6
Search-R1	23.3	18.3	35.0	25.6	57.8	39.3	49.2	15.5	40.5	13.7	18.6	4.8	37.4	19.6
Research	10.0	5.8	52.5	21.7	64.0	31.2	67.1	25.5	58.4	24.8	27.8	9.3	46.6	19.7
WebThinker	16.7	16.7	45.0	45.0	66.0	65.7	60.8	24.8	46.3	23.4	18.6	9.1	42.2	30.8
WebSailor	26.7	2.9	65.0	23.5	72.2	13.9	61.5	5.3	58.4	6.1	17.0	1.6	50.1	8.9
DotaMath	16.7	16.7	50.0	50.0	74.6	72.5	16.3	16.3	18.9	18.9	7.6	7.6	30.7	30.3
START	23.3	23.3	67.5	62.1	78.4	74.8	15.6	15.6	19.8	19.8	5.7	5.7	35.1	33.6
Multi-TIR Methods														
Prompting	13.3	6.1	47.5	33.5	67.0	51.7	18.0	10.5	17.9	12.4	8.5	5.6	28.7	20.0
IKEA	10.0	6.1	42.5	32.0	63.6	44.2	34.7	9.7	35.8	11.2	14.8	3.8	33.6	17.8
SMART	3.3	3.3	17.5	15.8	39.4	33.0	21.7	21.1	40.7	40.6	12.3	11.8	22.5	20.9
AutoTIR*	30.0	19.0	65.0	38.0	72.8	57.8	**71.5**	31.0	58.9	29.7	**33.1**	14.6	55.2	31.7
Tool-Star*	13.3	10.7	67.5	59.9	**79.8**	70.6	65.0	28.3	57.3	34.9	25.0	12.7	51.3	36.2
Tool-Light*	23.3	23.3	60.0	57.5	78.0	75.1	59.1	26.6	55.3	32.0	25.4	11.2	50.2	37.6
ET-Agent†	30.0	24.2	65.0	58.8	66.4	61.7	56.9	**42.2**	45.0	37.3	22.1	15.0	47.6	39.9
$\text{CostDescent}_{\text{static}}$	30.0	30.0	**75.0**	**73.8**	76.0	74.5	65.2	33.7	59.4	39.6	26.9	16.3	55.4	44.7
CostDescent	**33.3**	**31.7**	**75.0**	73.3	77.0	**75.3**	66.5	41.7	**60.2**	**50.0**	29.2	**19.4**	**56.9**	**48.6**

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Tasks and Tools Following ET-Agent (Chen et al., 2026c), we consider two families of tool-integrated reasoning tasks. The mathematical-reasoning(MR) suite contains AIME24,[1] AMC23,[2] MATH500 (Lightman et al., 2024), while the knowledge-intensive(KI) suite contains 2WikiMultiHopQA (Ho et al., 2020), Bamboogle (Press et al., 2023), and MuSiQue (Trivedi et al., 2022). All reproduced methods are supplied with python interpreter for mathematical reasoning and local Wikipedia retrieval for knowledge-intensive QA.

Evaluation Metric We report task performance using LLM judge (Li et al., 2024b)(Qwen-2.5-72B) for MR tasks and F1 score for KI tasks. To measure efficiency, we report Tool Productivity(TP) to measure the task score obtained per tool call: $TP = \frac{1}{N}\sum_{i=1}^{N}\frac{P_i}{T_i}$, where P_i is the correctness score and T_i is the tool call count (floored at 1) for sample i, following ET-Agent's $Effi$ metric. The judge prompt and scoring follow ET-Agent's public implementation.

Baselines We compare with direct inference from Qwen2.5-7B, single-tool TIR methods (Search-o1 (Li et al., 2025c), Search-R1 (Jin et al., 2025), Research (Chen et al., 2025), WebThinker (Li et al., 2025d), WebSailor (Li et al., 2025b), DotaMath (Li et al., 2024a), and START (Li et al., 2025a)), and multi-tool or efficiency-oriented methods (Prompting, IKEA (Huang et al., 2025), SMART (Qian et al., 2025), AutoTIR (Wei et al., 2025), Tool-Star (Dong et al., 2025), Tool-Light (Chen et al., 2026d), and ET-Agent (Chen et al., 2026c)).

Implementation Details We collect mathematical-reasoning training data following ET-Agent. For QA, both CostDescent variants and retrained ET-Agent are trained on NQ and HotpotQA following Search-R1 (Jin et al., 2025); all three QA evaluation datasets are out of domain for these

[1] https://huggingface.co/datasets/AI-MO/aimo-validation-aime

[2] https://huggingface.co/datasets/zwhe99/amc23

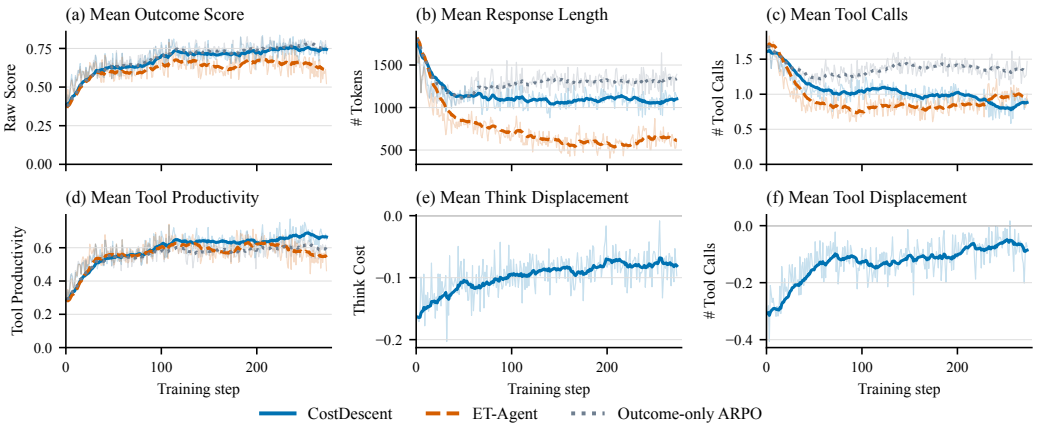


Figure 3: **Training dynamics.** Bold lines denote exponential moving averages. (a)–(d): outcome score (accuracy for MR and F1 for KI), response length, tool calls, and tool productivity of all three methods. (e)–(f): mean reference displacement of CostDescent on both axes.

models. We initialize CostDescent from the RFT checkpoint of our retrained ET-Agent, with orthogonally initialized SVD-LoRA parameters, and jointly optimize the backbone and SVD-LoRA during training. Following the curriculum training of ET-Agent and Tool-Light (Chen et al., 2026c;d), our main configuration, CostDescent, uses DAPO's dynamic sampling as an implicit curriculum. We also report CostDescent$_{\text{static}}$, a weaker curriculum-free setting that selects its training prompts once before RL; being cheaper to train, it is used for the ablations and generalization experiments on other backbones (see Appendix D.4). More implementation detils in Appendix C.

5.2 Main Results

Average Task Performance Table 1 shows that CostDescent achieves the highest average task performance across the six benchmarks (56.9), followed by CostDescent$_{\text{static}}$ (55.4), which exceeds the strongest baseline, AutoTIR (55.2), without a curriculum. CostDescent's 1.7-point lead over AutoTIR comes from mathematical reasoning (61.8 versus 55.9 across three tasks). On QA, CostDescent trails AutoTIR by 2.5 points (52.0 versus 54.5). We note that AutoTIR incorporates the training set of MuSiQue, whereas all three QA benchmarks are out-of-domain for both CostDescent configurations and retrained ET-Agent.

Pushing Further on Productivity Among baselines, AutoTIR leads task performance (55.2; TP 31.7), while retrained ET-Agent leads TP (39.9; performance 47.6). Both CostDescent configurations exceed these separate maxima: CostDescent$_{\text{static}}$ reaches 55.4 performance and 44.7 TP, while CostDescent reaches 56.9 and 48.6. On the three QA tasks, CostDescent also leads mean TP (37.0 versus retrained ET-Agent's 31.5). Appendix D.2 extends the comparison to conciseness, successful execution, and log-token efficiency.

5.3 Training Dynamics

We compare CostDescent, ET-Agent, and outcome-only ARPO from the same initial checkpoint on ET-Agent's Stage 1 curriculum data. ET-Agent uses an annealed scalar efficiency reward, whereas CostDescent derives cost pressure from within-group rollout outcomes.

Performance–Cost Analysis Fig. 3(a)–(d) shows that CostDescent tracks outcome-only ARPO in outcome score (0.744 vs. 0.755) while using fewer tool calls (0.88 vs. 1.34) and tokens (1,101 vs. 1,339), yielding higher tool productivity (0.665 vs. 0.590). ET-Agent produces the shortest responses (615 tokens), but its outcome score falls to 0.604 and tool calls rebound to 0.94, ending at 0.542 productivity. Validation at the deployment goal $g = 0$ shows the same pattern (App. D.1).

Table 2: **Ablation Results.** Top two results are highlighted in **bold** and underlined.

Method	Mathematical-Reasoning		Knowledge-Intensive		Average	
	LJ	TP	F1	TP	P	TP
$\text{CostDescent}_{\text{static}}$	**60.3**	**59.4**	50.5	29.9	**55.4**	**44.7**
w/o Cost Conditioning	54.2	49.4	51.0	29.0	52.6	39.2
w/o Asymmetric Routing	56.1	44.5	**52.1**	26.3	54.1	35.4
w/o Task-Aware Target	53.7	52.3	49.3	27.4	51.5	39.9
w/o Credit Inversion	55.1	52.1	50.0	**32.0**	52.6	42.0

Cost-Descent Dynamics Fig. 3(e)–(f) tracks $\hat{b} - \mu_{\text{support}}$, an observable proxy for the cost-reference gap (App. B.2). Its smoothed mean remains negative on both axes, placing the target below achieved support costs, while narrowing from -0.16 to -0.08 on the think axis and from -0.31 to -0.09 tool calls. Together with the overall falling costs in panels (b)–(c), this is consistent with Proposition 2's prediction that the target-reference gap contracts, and because the target lies below the achieved costs, the contraction yields cost descent.

5.4 Ablation Study

Under the $\text{CostDescent}_{\text{static}}$ setting, we remove one component at a time while holding the task reward and all others fixed. Removing Cost Conditioning recovers outcome-only ARPO; removing Asymmetric Routing uses $g_i^{\text{train}} = g_i^{\text{rel}}$ for all trajectories; removing the Task-Aware Target fixes $\hat{b}(q) = \mu_{\text{support}}$ ($\gamma = 0$); and removing Credit Inversion uses ω_{it} in place of ω_{it}^{-1}.

Every removal lowers average task performance and tool productivity (Table 2). Asymmetric Routing is most important for efficiency: removing it retains 54.1 performance but drops tool productivity from 44.7 to 35.4. Removing the Task-Aware Target gives the lowest performance (51.5), and its rollout cost drifts upward toward μ_{support}, faster than under outcome-only ARPO (App. B.3). Removing Cost Conditioning or Credit Inversion also lowers both metrics; the latter further induces repetitive rollouts (App. D.5).

5.5 Attribution Analysis

The Cost Conditioning ablation (Table 2) shows that cost descent operates through the goal input; we now examine how this input shapes resource use. We evaluate CostDescent over a grid of relative cost goals (Appendix C.3) on prompts with at least two default-goal tool calls; as a control without a learned goal input, ET-Agent receives the same goals through a system-prompt instruction.

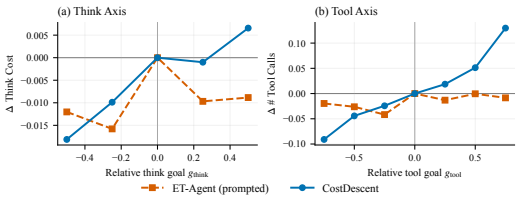


Figure 4: **Response to relative cost goals.** Mean achieved-cost change from $g = 0$, averaged over other goal axis. Think cost uses 512-token units.

Fig. 4 shows ordered responses on both axes, stronger on the tool axis, whereas prompted ET-Agent shows no ordered response. Token-level case studies further show that a negative tool goal lowers the probability of tokens that open a tool call and raises that of the answer opening (Appendix D.6).

6 Conclusion

We decouple cost from reward, where task reward alone evaluates outcome, while token and tool costs enter optimization as a goal input and are reduced through task-aware cost descent. Across six benchmarks, CostDescent achieves the strongest average task performance and tool productivity, suggesting behavioral targets outside the task reward for broader LLM reinforcement learning.

AI USE STATEMENT

In this work, we used generative AI tools for assist in the writing of proofs, implement methods and assist with translation. We have not used generative AI tools for help generate synthetic data sets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, design or provide feedback on research methodology or experiments, clean and reformat dataset, support qualitative and thematic data analysis, interpret results. Propose or refine hypotheses are not applicable to this work. Additionally, we used generative AI tools for create or modify scientific figures or images, edit a research paper to improve readability. We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness by 2 authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work has no explicit ethic concerns.

REPRODUCIBILITY STATEMENT

The main training code is provided in the supplementary material.

REFERENCES

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

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Dingwei Chen, Zefang Zong, Zhipeng Ma, Leo Luo, Yang Li, Chengming Li, Peng Chen, and Jie Jiang. Efficient agentic reinforcement learning with on-policy intrinsic knowledge boundary enhancement. *arXiv preprint arXiv:2605.26952*, 2026a.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Liuji Chen, Dianxing Tang, Xing Shi, Dingshuo Chen, Qiang Liu, Shu Wu, and Liang Wang. Learning when not to act: Mitigating tool abuse in agentic reinforcement learning. *arXiv preprint arXiv:2606.02132*, 2026b.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, sunhaoze, ZhouYijie, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. Research: Learning to reason with search for LLMs via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=OuGAwwAT8G.

Yifei Chen, Guanting Dong, and Zhicheng Dou. Et-agent: Incentivizing effective tool-integrated reasoning agent via behavior calibration. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7337–7359, 2026c.

Yifei Chen, Guanting Dong, and Zhicheng Dou. Toward effective tool-integrated reasoning via self-evolved preference learning. In *International Conference on Learning Representations*, volume 2026, pp. 74102–74122, 2026d.

Muzhi Dai, Shixuan Liu, and Qingyi Si. Stable reinforcement learning for efficient reasoning. *arXiv preprint arXiv:2505.18086*, 2025.

Guanting Dong, Yifei Chen, Xiaoxi Li, Jiajie Jin, Hongjin Qian, Yutao Zhu, Hangyu Mao, Guorui Zhou, Zhicheng Dou, and Ji-Rong Wen. Tool-star: Empowering llm-brained multi-tool reasoner via reinforcement learning. *arXiv preprint arXiv:2505.16410*, 2025.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. In *International Conference on Learning Representations*, volume 2026, pp. 16981–17017, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *Advances in Neural Information Processing Systems*, 38:46375–46408, 2026.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*, pp. 1515–1528. PMLR, 2018.

Loris Gaven, Clement Romac, Thomas Carta, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Sac-glam: Improving online rl for llm agents with soft actor-critic and hindsight relabeling. *arXiv preprint arXiv:2410.12481*, 2024.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24842–24855, 2025.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing A multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pp. 6609–6625, 2020. doi: 10.18653/V1/2020.COLING-MAIN.580. URL https://doi.org/10.18653/v1/2020.coling-main.580.

Ziyang Huang, Xiaowei Yuan, Yiming Ju, Jun Zhao, and Kang Liu. Reinforced internal-external knowledge synergistic reasoning for efficient adaptive search agent. *CoRR*, abs/2505.07596, 2025. doi: 10.48550/ARXIV.2505.07596. URL https://doi.org/10.48550/arXiv.2505.07596.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Jungseob Lee, Seungyoon Lee, Seongtae Hong, Minhyuk Kim, Chanjun Park, and Heuiseok Lim. Beyond penalizing mistakes: Stabilizing efficiency training in large reasoning models via adaptive correct-only rewards. *arXiv preprint arXiv:2606.22716*, 2026.

Chengpeng Li, Guanting Dong, Mingfeng Xue, Ru Peng, Xiang Wang, and Dayiheng Liu. Dotamath: Decomposition of thought with code assistance and self-correction for mathematical reasoning. *CoRR*, abs/2407.04078, 2024a. doi: 10.48550/ARXIV.2407.04078. URL https://doi.org/10.48550/arXiv.2407.04078.

Chengpeng Li, Mingfeng Xue, Zhenru Zhang, Jiaxi Yang, Beichen Zhang, Bowen Yu, Binyuan Hui, Junyang Lin, Xiang Wang, and Dayiheng Liu. START: Self-taught reasoner with tools. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 13512–13553, Suzhou, China, November 2025a. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.683. URL https://aclanthology.org/2025.emnlp-main.683/.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, Kai Shu, Lu Cheng, and Huan Liu. From generation to judgment: Opportunities and challenges of llm-as-a-judge. *CoRR*, abs/2411.16594, 2024b. doi: 10.48550/ARXIV.2411.16594. URL https://doi.org/10.48550/arXiv.2411.16594.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, Weizhou Shen, Junkai Zhang, Dingchu Zhang, Xixi Wu, Yong Jiang, Ming Yan, Pengjun Xie, Fei Huang, and Jingren Zhou. Websailor: Navigating superhuman reasoning for web agent, 2025b. URL https://arxiv.org/abs/2507.02592.

Tingyun li, Zishang Jiang, Jinyi Han, Xinyi Wang, Sihang Jiang, Han Xia, Zhaoqian Dai, Ma Shuguang, Fei Yu, Jiaqing Liang, and Yanghua Xiao. ADaPT: Token-level decoupling for efficient large reasoning models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 3355–3369, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.165. URL https://aclanthology.org/2026.findings-acl.165/.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5420–5438, Suzhou, China, November 2025c. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.276. URL https://aclanthology.org/2025.emnlp-main.276/.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. Webthinker: Empowering large reasoning models with deep research capability. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 120091–120131. Curran Associates, Inc., 2025d. doi: 10.52202/085713-4011. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/ae03bdef276132fae089692445725635-Paper-Conference.pdf.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. Torl: Scaling tool-integrated rl. *arXiv preprint arXiv:2503.23383*, 2025e.

Zichao Li, Gang Wu, Jack Wang, Ruiyi Zhang, Wanrong Zhu, Ryan Rossi, Vlad Morariu, and Jihyung Kil. Spinning straw into gold: Relabeling llm agent trajectories in hindsight for successful demonstrations. In *International Conference on Learning Representations*, volume 2026, pp. 148186–148209, 2026.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pp. 5687–5711, 2023. doi: 10.18653/V1/2023.FINDINGS-EMNLP.378. URL https://doi.org/10.18653/v1/2023.findings-emnlp.378.

Qiuyi Qi, Tian Liang, Jiamu Wang, Jinjian Zhang, Wei Zhou, Pengcheng Zhu, Linjian Mo, Ming Kong, Jie Liu, and Qiang Zhu. Metarag: Belief-action aligned policy optimization for agentic rag. *arXiv preprint arXiv:2608.24214*, 2026.

Cheng Qian, Emre Can Acikgoz, Hongru Wang, Xiusi Chen, Avirup Sil, Dilek Hakkani-Tur, Gokhan Tur, and Heng Ji. Smart: Self-aware agent for tool overuse mitigation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 4604–4621, 2025.

Paulo Rauber, Avinash Ummadisingu, Filipe Mutz, and Jürgen Schmidhuber. Hindsight policy gradients. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg2viA5FQ.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1312–1320, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/schaul15.html.

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

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

Mingyang Song and Mao Zheng. Walk before you run! concise llm reasoning via reinforcement learning, 2025. URL https://arxiv.org/abs/2505.21178.

Rupesh Kumar Srivastava, Pranav Shyam, Filipe Mutz, Wojciech Jaśkowski, and Jürgen Schmidhuber. Training agents using upside-down reinforcement learning. *arXiv preprint arXiv:1912.02877*, 2019.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multi-hop questions via single-hop question composition. *Trans. Assoc. Comput. Linguistics*, 10:539–554, 2022. doi: 10.1162/TACL_A_00475. URL https://doi.org/10.1162/tacl_a_00475.

Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang, Kam-Fai Wong, and Heng Ji. Acting less is reasoning more! teaching model to act efficiently. *arXiv preprint arXiv:2504.14870*, 2025.

Jiakang Wang, Runze Liu, Qingpeng Cai, Lei Lin, Wenping Hu, Xiu Li, Fuzheng Zhang, Guorui Zhou, Kun Gai, and Ling Pan. When importance sampling misallocates credit: Asymmetric ratios for outcome-supervised rl, 2026. URL https://arxiv.org/abs/2510.06062.

Yifan Wei, Xiaoyan Yu, Yixuan Weng, Tengfei Pan, Angsheng Li, and Li Du. Autotir: Autonomous tools integrated reasoning via reinforcement learning. *CoRR*, abs/2507.21836, July 2025. URL https://doi.org/10.48550/arXiv.2507.21836.

Peilin Wu, Mian Zhang, Xinlu Zhang, Xinya Du, and Zhiyu Chen. Search wisely: Mitigating sub-optimal agentic searches by reducing uncertainty. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 19723–19734, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.998. URL https://aclanthology.org/2025.emnlp-main.998/.

Peilin Wu, Mian Zhang, Kun Wan, Wentian Zhao, Kaiyu He, Xinya Du, and Zhiyu Chen. Hiprag: hierarchical process rewards for efficient agentic retrieval augmented generation. In *International Conference on Learning Representations*, volume 2026, pp. 82317–82347, 2026.

Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 8424–8449, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Tianjun Zhang, Fangchen Liu, Justin Wong, Pieter Abbeel, and Joseph E Gonzalez. The wisdom of hindsight makes language models better instruction followers. In *International Conference on Machine Learning*, pp. 41414–41428. PMLR, 2023.

Yifan Zhong, Chengdong Ma, Xiaoyuan Zhang, Ziran Yang, Haojun Chen, Qingfu Zhang, Siyuan Qi, and Yaodong Yang. Panacea: Pareto alignment via preference adaptation for llms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 75522–75558. Curran Associates, Inc., 2024. doi: 10.52202/079017-2405. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/89f39d0b3d49a47606a165eefba2778c-Paper-Conference.pdf.

A METHOD DETAILS

A.1 TRAINING ALGORITHM

Algorithm 1 summarizes the complete training loop. The target reference is recomputed from each on-policy rollout group and held fixed during its policy update.

Algorithm 1 CostDescent training

Require: Dataset $\mathcal{D}$, policy π_θ, group size G, PPO clip ϵ, token-weight bounds ℓ_+, u_+

for each training iteration **do**
  Sample a task $q \sim \mathcal{D}$
  Sample rollout goals $\{g_i^{\text{roll}}\}_{i=1}^{G}$ by Cost-Aware Rollout, with half of the roots at $g = 0$
  Sample $\tau_i \sim \pi_{\text{old}}(\cdot \mid q, g_i^{\text{roll}})$ for $i = 1, \ldots, G$
  Compute $c_i = c(\tau_i)$, $R_i = R(q, \tau_i)$, GRPO advantage A_i, and task-acceptance indicator S_i
  Update the rollout-scale statistics from the achieved costs
  Select the estimator support $\mathcal{I}_{\text{sup}}$; if empty, set $\mathcal{I}_{\text{sup}} \leftarrow \{1, \ldots, G\}$
  $\mu_{\text{support}} \leftarrow |\mathcal{I}_{\text{sup}}|^{-1} \sum_{i \in \mathcal{I}_{\text{sup}}} c_i$
  if $\sum_i S_i > 0$ **then**
    Compute the soft cost-preference scores h_i and weights $w_i = S_i[A_i]_+ h_i$ by Eq. 4
  else
    $w_i \leftarrow 0$ for all i
  end if
  if $\sum_i w_i > 0$ **then**
    Compute μ_{proposal}, γ, and $\hat{b}(q)$ by Eqs. 4–5
  else
    $\hat{b}(q) \leftarrow \mu_{\text{support}}$
  end if
  for $i = 1, \ldots, G$ **do**
    $g_i^{\text{rel}} \leftarrow c_i - \hat{b}(q)$
    if $A_i > 0$ **then**
      $g_i^{\text{train}} \leftarrow g_i^{\text{rel}}$
      Compute ω_{it}, z_{it}, and $\widetilde{A}_{it} = A_i z_{it}$ by Eqs. 9–10
    else
      $g_i^{\text{train}} \leftarrow g_i^{\text{roll}}$ and $\widetilde{A}_{it} \leftarrow A_i$
    end if
  end for
  Update θ by maximizing $\mathcal{J}_{\text{CostDescent}}$ in Eq. 11; set $\pi_{\text{old}} \leftarrow \pi_\theta$
end for

A.2 SVD-LORA PARAMETERIZATION

We implement the relative cost-goal interface using a variant of SVD-LoRA (Zhong et al., 2024). Each selected projection ℓ has a goal-independent low-rank update $M_{\ell,0}$ and two axis-specific low-rank updates $M_{\ell,1}$ and $M_{\ell,2}$. With factorization rank r, learned factors and a scalar gate define each update as

$$M_{\ell,a} = s_{\ell,a} U_{\ell,a} V_{\ell,a}^\top. \tag{12}$$

With LoRA scaling parameter α_H, the goal-conditioned weight is

$$W_\ell(g) = W_{\ell,\text{backbone}} + \frac{\alpha_H}{r} \left(M_{\ell,0} + \sum_{a=1}^{2} g_a M_{\ell,a} \right). \tag{13}$$

At $g = 0$, the goal-dependent terms vanish and the remaining update can be merged into the backbone without additional inference-time computation. For each selected projection ℓ, the factor matrices satisfy $U_{\ell,a} \in \mathbb{R}^{d_{\text{out}} \times r}$ and $V_{\ell,a} \in \mathbb{R}^{d_{\text{in}} \times r}$ for $a \in \{0, 1, 2\}$. The factors are orthogonally initialized and their scalar gates are initialized to zero, preserving the initial policy for every goal input. The SVD-LoRA regularizer $\mathcal{R}_{\text{SVD}}$ averages

$$\mathcal{R}_\ell = \frac{1}{3r} \left\| (U_\ell^\top U_\ell) \odot (V_\ell^\top V_\ell) - I_{3r} \right\|_F^2 \tag{14}$$

over enabled projections and layers, where $U_\ell = [U_{\ell,0}, U_{\ell,1}, U_{\ell,2}]$ and $V_\ell = [V_{\ell,0}, V_{\ell,1}, V_{\ell,2}]$.

A.3 Cost-Aware Rollout

For each prompt, we allocate half of the rollout roots to the default goal $g = 0$ and sample the remainder independently around this origin. For each cost axis, the adaptive half-width is proportional to an exponential moving average of the achieved-cost standard deviation from preceding batches. Each exploratory root is sampled either from the resulting zero-centered uniform box or, with a fixed mixing probability, from a persistent static box. The mixture choice is shared across the two axes while their coordinates are sampled independently, so each root uses a single exploration scale without coupling token and tool offsets. The static component preserves broad coverage as the adaptive range contracts. We retain each sampled goal as g_i^{roll} for subsequent relabeling and goal-routed optimization.

A.4 Target-Reference Estimator

Task Reward and Advantage Each trajectory receives a task score $s_i \in [0, 1]$, answer correctness for mathematical reasoning and F1 for knowledge-intensive QA. The task reward is $R_i = s_i + f_i$, where the format term f_i is -1 for malformed output and 0 otherwise; no cost term enters R_i. The advantage A_i standardizes R_i by the mean and standard deviation of its group of $G = 16$ trajectories. Groups with zero reward variance carry zero advantage and contribute no policy gradient; under CostDescent, dynamic sampling discards them before the update.

Task-Performance Acceptance The indicator S_i requires valid format. For mathematical reasoning, $S_i = 1$ if and only if $s_i = 1$. For knowledge-intensive QA, whose F1 is graded, acceptance combines an absolute threshold with a group-relative rank: $S_i = 1$ if $s_i \geq 0.8$; if $0.4 \leq s_i < 0.8$ and the trajectory ranks among the $\lceil G/2 \rceil$ highest-scoring trajectories in this band; or if it is the highest-scoring trajectory of its group and $s_i > 0$. Ties in score are broken toward lower cost. The same criterion is used in all experiments.

Estimator Support A trajectory has a repetitive tail when its output is malformed and, for some period p between 16 and 512 characters, the final $\max(p, 96)$ characters of its whitespace-normalized last 2,048 characters equal the span p characters earlier. Such a trajectory keeps its task reward and format penalty but is excluded from the estimator support $\mathcal{I}_{\mathrm{sup}}$; other malformed trajectories remain in the support. This detector is separate from the repetition measure of App. D.5. If the support becomes empty, it falls back to the full group. If a group contains no performance-accepted trajectory, we set $h_i = 0$; if no valid cost proposal remains, the target cost reference stays at μ_{support}.

B Theoretical Analysis

B.1 Relative Cost Conditioning: Expressive Equivalence

Proposition 3 (Equivalence under aligned reference routing)**.** *Fix a reference router $b(q)$ and a relative-goal domain $\mathcal{G}_q$ containing zero. Let $\mathcal{G}_q^{\mathrm{abs}} = b(q) + \mathcal{G}_q$. Relative policies aligned at $g = 0$ and absolute policies aligned at $g^{\mathrm{abs}} = b(q)$ are in bijection under*

$$\Pi(\cdot \mid q, g^{\mathrm{abs}}) = \pi(\cdot \mid q, g^{\mathrm{abs}} - b(q)).$$

For corresponding goals $g^{\mathrm{abs}} = b(q) + g$, this correspondence preserves trajectory distributions, and hence the achieved-cost distribution and expected task reward.

Proof. Assume that the cost map c is measurable and that expected costs at the reference goal are finite. Alignment means, respectively,

$$\mathbb{E}_{\pi(\cdot|q,0)}[c(\tau)] = b(q), \qquad \mathbb{E}_{\Pi(\cdot|q,b(q))}[c(\tau)] = b(q).$$

Given a relative policy π, construct the absolute policy by defining, for every measurable trajectory event A,

$$\Pi(A \mid q, g^{\mathrm{abs}}) := \pi(A \mid q, g^{\mathrm{abs}} - b(q)).$$

This construction preserves alignment because

$$\mathbb{E}_{\Pi(\cdot|q,b(q))}[c(\tau)] = \mathbb{E}_{\pi(\cdot|q,0)}[c(\tau)] = b(q).$$

For $g^{\mathrm{abs}} = b(q) + g$, it also gives

$$\Pi(\cdot \mid q, g^{\mathrm{abs}}) = \pi(\cdot \mid q, g).$$

Equality of the trajectory measures preserves the probability of every measurable trajectory event, including any event defined through the achieved cost $c(\tau)$, and the expectation of every integrable trajectory statistic, including expected cost and expected task reward.

Conversely, given an aligned absolute policy Π, define

$$\pi(A \mid q, g) := \Pi(A \mid q, b(q) + g).$$

Then

$$\mathbb{E}_{\pi(\cdot|q,0)}[c(\tau)] = \mathbb{E}_{\Pi(\cdot|q,b(q))}[c(\tau)] = b(q),$$

so the router is aligned with the relative policy's expected cost at $g = 0$. Finally, the translations $g \mapsto b(q)+g$ and $g^{\mathrm{abs}} \mapsto g^{\mathrm{abs}} - b(q)$ are mutually inverse, hence so are the two policy constructions. □

B.2 Task-Aware Cost Descent: Local Analysis

We prove Proposition 2 for the positive-advantage policy-gradient component of equation 11 on a fixed task q. The collected trajectories, advantages, target cost reference, relabeled goals, and token weights remain fixed, and all policy gradients are evaluated at the pre-update parameters θ. Let $\mathcal{I}_+ = \{i : A_i > 0\}$ be nonempty. We consider the full-softmax categorical trajectory policy with a bounded rollout horizon and parameter-independent environment and tool-transition kernels. The argument uses local second-order regularity with respect to trainable parameters; it does not differentiate the expected-cost function with respect to q, g, or achieved cost.

Ideal Update and Same-Goal Response Let θ contain all trainable policy parameters. For $i \in \mathcal{I}_+$, the token-mean score direction at the hindsight-relabeled goal is

$$v_i = \frac{1}{|\mathcal{T}_i|} \sum_{t \in \mathcal{T}_i} z_{it} \nabla_\theta \log \pi_\theta(a_{it} \mid x_{it}, g_i^{\mathrm{rel}}), \qquad g_i^{\mathrm{rel}} = c_i - \hat{b}(q), \tag{15}$$

where $\mathcal{T}_i$ contains its policy-generated token positions and z_{it} is the detached token weight from Section 4.4. At the old policy, the PPO ratio is one and clipping is inactive. The positive-advantage part of the objective therefore gives the group-mean direction and ideal update

$$d_+ := \frac{1}{G} \sum_{i \in \mathcal{I}_+} A_i v_i, \qquad \theta^+ := \theta + \varepsilon d_+. \tag{16}$$

Normalized competition in the categorical trajectory simplex illustrates same-goal cost movement. At a fixed goal, let p_j be the probability of trajectory j, let $\mu(q, g) := \sum_j p_j c_j$, and let $\bar{A} = \sum_j p_j A_j$. The ideal categorical natural-policy-gradient flow is $\dot{p}_j = p_j(A_j - \bar{A})$, under which

$$\dot{\mu}(q, g) = \sum_j p_j A_j \big(c_j - \mu(q, g)\big). \tag{17}$$

For the parameterized policy, define

$$J(g) := \mathbb{E}_{\tau \sim \pi_\theta(\cdot|q,g)} \left[\big(c(\tau) - \mu_\theta(q, g)\big) \nabla_\theta \log \pi_\theta(\tau \mid q, g)^\top \right], \tag{18}$$

and write $\mu_i = \mu_\theta(q, g_i^{\mathrm{rel}})$, $J_i = J(g_i^{\mathrm{rel}})$, and $J_0 = J(0)$. The matrix $J(g)$ maps a parameter-space direction to the first-order change in expected cost at goal g. We assess same-goal progress after aggregating the positive-advantage directions in the sampled group.

Condition Neighborhood and Score Regularity For each i, let $\mathcal{G}_i = \{sg_i^{\mathrm{rel}} : s \in [0,1]\}$ be the compact segment connecting $g = 0$ and g_i^{rel}, and write $\mathbb{P}_g = \pi_\theta(\tau \mid q, g)$. The affine SVD-LoRA path in equation 13 makes every effective weight matrix Lipschitz in g. The smooth full-softmax Transformer maps these weights to token logits and parameter scores through a finite composition of smooth operations. A finite cover of $\mathcal{G}_i$, followed by a partition of each connecting subsegment, therefore gives constants $L_{g,i}, B_{s,i}, L_{s,i} < \infty$ such that, uniformly over trajectories within the bounded horizon,

$$\begin{aligned} \|\ell_\theta(x,g) - \ell_\theta(x,g')\|_2 &\le L_{g,i}\|g - g'\|_2, \\ \|s_g(\tau)\|_2 &\le B_{s,i}, \\ \|s_g(\tau) - s_{g'}(\tau)\|_2 &\le L_{s,i}\|g - g'\|_2, \end{aligned} \tag{19}$$

where $s_g(\tau) := \nabla_\theta \log \pi_\theta(\tau \mid q, g)$ and $g, g' \in \mathcal{G}_i$.

Behavioral and Response Lipschitz Bounds For distributions P and Q, let $D_{\mathrm{TV}}(P,Q) := \frac{1}{2}\sum_\tau |P(\tau) - Q(\tau)|$. For a token softmax $p = \mathrm{softmax}(u)$, the Jacobian satisfies $\|(\mathrm{diag}(p) - pp^\top)\delta\|_1 = \mathbb{E}_{k\sim p}|\delta_k - \mathbb{E}_p\delta| \le \|\delta\|_2$, so integrating along the logit segment gives $D_{\mathrm{TV}}(\mathrm{softmax}(u), \mathrm{softmax}(v)) \le \frac{1}{2}\|u - v\|_2$. Let H bound the rollout horizon. Switching the goal one token at a time along the trajectory, trajectory TV is at most the sum of the per-token TVs; tool-response tokens do not depend on g and contribute nothing. With equation 19, for $g, g' \in \mathcal{G}_i$,

$$D_{\mathrm{TV}}(\mathbb{P}_g, \mathbb{P}_{g'}) \le C_{P,i}\|g - g'\|_2, \qquad C_{P,i} := \tfrac{1}{2}HL_{g,i}. \tag{20}$$

We next lift this behavioral bound to the score-function cost Jacobian in equation 18. Assume that the achieved-cost support has finite diameter

$$D_c := \sup_{\tau,\tau'} \|c(\tau) - c(\tau')\|_2 < \infty. \tag{21}$$

Because the trajectory score has zero mean, $J(g) = \mathbb{E}_{\mathbb{P}_g}[(c(\tau) - c_0)s_g(\tau)^\top]$ for any fixed c_0 in the cost support. Using one c_0 for both goals,

$$J(g) - J(g') = \mathbb{E}_{\mathbb{P}_g}\left[(c - c_0)(s_g - s_{g'})^\top\right] + \left(\mathbb{E}_{\mathbb{P}_g} - \mathbb{E}_{\mathbb{P}_{g'}}\right)\left[(c - c_0)s_{g'}^\top\right].$$

By equation 19, the first term is at most $D_c L_{s,i}\|g - g'\|_2$ in operator norm. The integrand of the second term has operator norm at most $D_c B_{s,i}$, so by equation 20 the second term is at most $2D_c B_{s,i} C_{P,i}\|g - g'\|_2$. Hence

$$\|J(g) - J(g')\|_{\mathrm{op}} \le L_{J,i}\|g - g'\|_2, \qquad L_{J,i} := D_c\big(L_{s,i} + 2B_{s,i}C_{P,i}\big). \tag{22}$$

In particular,

$$\|(J_0 - J_i)v_i\|_2 \le L_{J,i}\|g_i^{\mathrm{rel}}\|_2\|v_i\|_2. \tag{23}$$

Local Cost-Reference-Gap Descent Define the expected-cost displacement at the relabeled goal, its response residual, and the cost-reference gap as

$$\Delta\mu_i := \mu_i - b_\theta(q), \qquad e_i^{\mathrm{resp}} := g_i^{\mathrm{rel}} - \Delta\mu_i, \qquad \delta_b(q) := \hat{b}(q) - b_\theta(q). \tag{24}$$

Since $g_i^{\mathrm{rel}} = c_i - \hat{b}(q)$,

$$c_i - \mu_i = \delta_b(q) + e_i^{\mathrm{resp}}. \tag{25}$$

Holding $\hat{b}(q)$ fixed during the update, define

$$\mathcal{D}_q(\theta; \hat{b}) := \frac{1}{2}\|\delta_b(q)\|_2^2 = \frac{1}{2}\left\|\hat{b}(q) - b_\theta(q)\right\|_2^2. \tag{26}$$

Formal Statement of Proposition 2 Under the bounds above and the ideal update in equation 16, define the aggregate same-goal progress and pre-update margin

$$\begin{aligned} Q &:= \frac{1}{G}\sum_{i\in\mathcal{I}_+} A_i(c_i - \mu_i)^\top J_i v_i, \\ m_{\mathrm{agg}} &:= Q - \frac{1}{G}\left|\sum_{i\in\mathcal{I}_+} A_i(e_i^{\mathrm{resp}})^\top J_i v_i\right| \\ &\quad - \frac{\|\delta_b(q)\|_2}{G}\sum_{i\in\mathcal{I}_+} A_i L_{J,i}\|g_i^{\mathrm{rel}}\|_2\|v_i\|_2. \end{aligned} \tag{27}$$

If $m_{\mathrm{agg}} > 0$, d_+ is a strict descent direction for $\mathcal{D}_q(\theta; \hat{b})$ at θ. Consequently, every sufficiently small ideal positive-advantage update decreases $\mathcal{D}_q(\theta; \hat{b})$ and contracts the cost-reference gap.

The margin separates three effects of the positive-advantage update. The same-goal progress Q measures whether reinforcing trajectory i at g_i^{rel} moves the expected cost at that goal toward c_i; equation 17 gives its categorical counterpart. Because relabeled goals are centered on $\hat{b}(q)$, equation 25 splits this progress into a part along the cost-reference gap $\delta_b(q)$ and a part along the goal-response residual e_i^{resp}, the part of the requested offset g_i^{rel} not realized by the expected-cost displacement $\Delta\mu_i$. The second term removes the residual part, so the first two terms of m_{agg} lower-bound the gap part and the condition is directional: in aggregate, the update must move expected cost toward the target reference. A calibrated goal response ($e_i^{\mathrm{resp}} = 0$) is the special case in which all of Q lies along the gap. The third term bounds how this movement at g_i^{rel} differs from that at $g = 0$, i.e., how well it carries over to the policy-induced reference. The margin depends on the current policy and its parameterization, so Appendix B.3 tests its consequence directly: whether $b_\theta(q)$ moves toward $\hat{b}(q)$.

Proof of Proposition 2. For each i, substitute $\delta_b(q) = c_i - \mu_i - e_i^{\mathrm{resp}}$ and add and subtract $J_i v_i$:

$$\delta_b(q)^\top J_0 d_+ = \frac{1}{G} \sum_{i \in \mathcal{I}_+} A_i \Big[\big(c_i - \mu_i\big)^\top J_i v_i - \big(e_i^{\mathrm{resp}}\big)^\top J_i v_i + \delta_b(q)^\top (J_0 - J_i) v_i \Big]. \tag{28}$$

The triangle inequality and equation 23 imply

$$\delta_b(q)^\top J_0 d_+ \geq m_{\mathrm{agg}}. \tag{29}$$

Since $J_0 d_+$ is the directional derivative of $b_\theta(q)$ along d_+, local second-order parameter regularity gives, for some $C_2 < \infty$,

$$\mathcal{D}_q(\theta + \varepsilon d_+; \hat{b}) - \mathcal{D}_q(\theta; \hat{b}) = -\varepsilon\, \delta_b(q)^\top J_0 d_+ + r_{\mathcal{D}}, \qquad |r_{\mathcal{D}}| \leq C_2 \varepsilon^2. \tag{30}$$

By equation 29, the right side is at most $-\varepsilon m_{\mathrm{agg}} + C_2 \varepsilon^2$, which is negative for $0 < \varepsilon < m_{\mathrm{agg}}/C_2$. In particular, d_+ is a strict descent direction for the cost-reference-gap potential. □

For on-policy groups sampled at fixed q and θ, the target reference and update direction vary with the group. Taking $d_+ = m_{\mathrm{agg}} = 0$ when $\mathcal{I}_+$ is empty, and assuming an integrable bound on the second-order remainder, equation 30 gives

$$\mathbb{E}\Big[\mathcal{D}_q(\theta + \varepsilon d_+; \hat{b}) - \mathcal{D}_q(\theta; \hat{b})\Big] \leq -\varepsilon \mathbb{E}[m_{\mathrm{agg}}] + O(\varepsilon^2). \tag{31}$$

A positive expected aggregate margin thus gives local descent on average over sampled groups. As with the group-relative baseline of GRPO, each group is evaluated against its own sampled target, which is held fixed during the update and reconstructed at the next iteration.

Rollout Proxy for the Policy-Induced Reference Let $\mathcal{I}_{\mathrm{sup}}$ contain the support-admitted trajectories in a rollout group, using all trajectories under the empty-support fallback, and write $n_{\mathrm{sup}} = |\mathcal{I}_{\mathrm{sup}}|$. Suppose the expected-cost map has a local affine response on the sampled goal range, $\mu_\theta(q, g) = b_\theta(q) + \Gamma_q g + r_q(g)$, where $\Gamma_q \in \mathbb{R}^{2\times 2}$ is a cost-response matrix and $\|r_q(g)\|_2 \leq C_q \|g\|_2^2$ for some finite C_q. The support mean then obeys

$$\mu_{\mathrm{support}} - b_\theta(q) = \Gamma_q \frac{1}{n_{\mathrm{sup}}} \sum_{i \in \mathcal{I}_{\mathrm{sup}}} g_i^{\mathrm{roll}} + \frac{1}{n_{\mathrm{sup}}} \sum_{i \in \mathcal{I}_{\mathrm{sup}}} r_q(g_i^{\mathrm{roll}}) + \frac{1}{n_{\mathrm{sup}}} \sum_{i \in \mathcal{I}_{\mathrm{sup}}} \big(c_i - \mu_\theta(q, g_i^{\mathrm{roll}})\big). \tag{32}$$

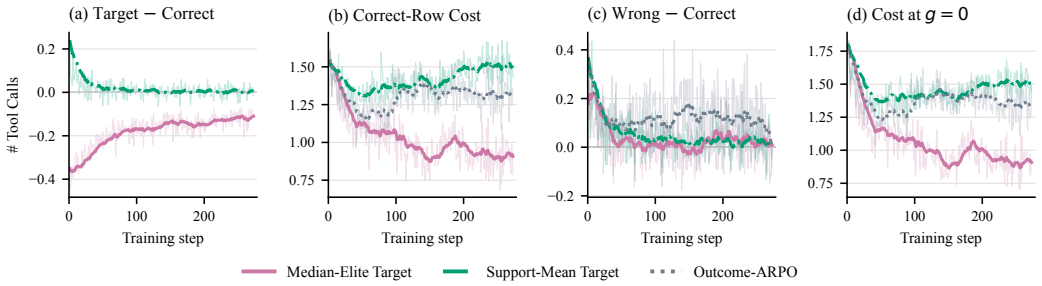


Figure 5: **Direction of policy-induced reference movement.** Tool-axis statistics of the training rollouts at every step; bold lines denote exponential moving averages. A trajectory is *correct* if its outcome score is positive (answer F1 > 0 for KI) and *mixed* groups contain at least one correct and one incorrect support trajectory. (a) Mean over mixed groups of $\hat{b}(q)$ minus the mean cost of correct support trajectories, drawn for the two runs whose target enters the update. (b) Mean cost over all correct support trajectories. (c) Mean over mixed groups of the incorrect-minus-correct mean cost. (d) Mean cost over trajectories sampled at $g = 0$.

Centering of Rollout Goals Cost-Aware Rollout assigns half the roots to $g = 0$ and samples the remainder from a distribution symmetric about zero (Appendix A.3). This centers the rollout-goal distribution around the policy reference; zero-goal roots reduce the curvature term, while averaging across prompt groups reduces sampling fluctuation. These properties motivate the batch mean of $\hat{b} - \mu_{\text{support}}$ as an observable proxy for the corresponding mean ideal gap $\hat{b} - b_\theta$. Fig. 3(e)–(f) averages this proxy over groups with a valid target.

Outcome Generality The argument uses c only through fixed trajectory-level outcomes, their finite support diameter, the expected-outcome map, and the corresponding target reference. Replacing c with any scalar reward or vector-valued behavioral statistic satisfying the corresponding boundedness, local-response, and cross-goal coupling conditions yields the same contraction result. For a scalar outcome, selecting a lower or higher target reference induces local descent or ascent, respectively, whenever the proposition's positive-margin condition holds.

The proposition characterizes the local direction induced by the positive-advantage component at the current on-policy target. Appendix B.3 tests this prediction in the full training loop by placing target references on opposite sides of the reinforced trajectories and measuring the resulting default-policy movement.

B.3 Empirical Validation of Proposition 2

Proposition 2 predicts a direction: the update moves $b_\theta(q)$ toward $\hat{b}(q)$, whichever side of the current costs the target lies on. We test this prediction with two runs whose launch settings differ only in the target-reference estimator and which place the target on opposite sides of the reinforced trajectories. *Support-Mean Target* is the w/o Task-Aware Target variant of Table 2. *Median-Elite Target* takes as its proposal the $[A_i]_+$-weighted mean cost of the accepted trajectories at or below the median accepted cost, and its factor γ is the acceptance rate among all trajectories in that cost range divided by p_G. This hard cost region replaces the soft score h_i and gives a more aggressive lower target than the full method. Fig. 5 traces both runs together with *Outcome-ARPO*, the outcome-only ARPO baseline of Section 5.3, which receives no target. The three runs reach similar outcome scores on their training rollouts (0.756, 0.750, and 0.760 averaged over the last third of training).

Target Placement Equation 5 starts from the support mean μ_{support} and moves toward the cost proposal only by the fraction γ. The support mean is therefore the estimator's fallback: the target stays at μ_{support} when a group has no valid proposal, and $\gamma < 1$ retains part of it whenever performance acceptance is rarer among low-cost trajectories than in the group as a whole. In both cases, the target moves below the current support only as far as the acceptance evidence supports. Support-Mean Target applies this fallback in every group, which is equivalent to $\gamma = 0$. Because

Table 3: **Training-data composition.** AIME contains questions from 2023 and earlier.

Dataset	Task	Count
AIME	MR	0.9K
Numina-CoT	MR	5K
Numina-TIR	MR	5K
OpenR1-Math-220K	MR	10K
DAPO-Math-17K	MR	5K
Natural Questions (NQ)	KI	10K
HotpotQA	KI	10K
ASearcherLRM35K	DS	10K
WebShaper	DS	0.5K

μ_{support} averages correct and incorrect support trajectories by count, in a mixed group with correct fraction p and incorrect-minus-correct cost difference s it lies $(1-p)s$ above the mean cost of the correct trajectories. The positive-advantage trajectories, which the update reinforces, are predominantly these correct ones. Their relabeled goals are therefore centered slightly above their own costs under Support-Mean Target and below them under Median-Elite Target. Fig. 5(a) confirms this placement: averaged over the three thirds of training, the target lies 0.24, 0.15, and 0.13 tool calls below the correct-trajectory cost under Median-Elite Target and 0.028, 0.005, and 0.004 above it under Support-Mean Target. The latter offset is small after the first third, so its effect relies on persisting with the same sign over many updates.

Movement Follows the Target Equation 25 states the effect of one update: the relabeled goals are centered on $\hat{b}(q)$, so the same-goal progress Q contains the displacement of the target from the reinforced costs, and its sign sets the direction in which $b_\theta(q)$ moves in expectation. Under stochastic minibatches, this per-step push appears as a drift accumulated over training, which we measure against Outcome-ARPO as the target-free control. All three runs, including the control, reduce tool calls during the first $\sim$50 steps, so we measure the drift from step 51 onward. The least-squares slope of the per-step mean cost over steps 51–273 is -0.088 tool calls per 100 steps at $g = 0$ under Median-Elite Target, $+0.015$ for Outcome-ARPO, and $+0.065$ under Support-Mean Target (Fig. 5(d)); the corresponding slopes for correct trajectories are -0.075, $+0.042$, and $+0.090$ (panel (b)). The drift therefore follows the sign of the target offset relative to the control, and it is larger for the larger offset. By the last third of training, the mean cost at $g = 0$ is 0.94 tool calls under Median-Elite Target, 1.37 for Outcome-ARPO, and 1.51 under Support-Mean Target. Under Support-Mean Target, the incorrect-minus-correct difference also narrows from 0.072 to 0.023 (panel (c)) as correct trajectories approach the support mean, whereas Outcome-ARPO retains a difference of 0.10–0.13 throughout. In the regime the fallback is designed for, this upward drift is the intended response: the policy spends cost where its accepted evidence does not justify spending less.

C Experimental Details

C.1 Datasets

Table 3 summarizes the training mixture. We retain ET-Agent's MR and deep-search (DS) sources and counts (Chen et al., 2026c), and use 10K questions each from Natural Questions (NQ) and HotpotQA for KI, following the QA-domain convention of Search-R1 (Jin et al., 2025). The retrained ET-Agent and both CostDescent variants use this mixture. We use ET-Agent's data flywheel to generate trajectories, retain 10.5K of them for RFT, and exclude their questions from subsequent RL training.

C.2 Training and Evaluation Configuration

Both CostDescent configurations mainly differ in data selection and sampling; all other settings are shared. CostDescent, the main configuration, selects online: it trains on the full RL dataset with DAPO-style dynamic sampling, which retains only prompt groups whose sequence reward has nonzero standard deviation under the current policy and resamples until the batch is filled. $\text{CostDescent}_{\text{static}}$ selects once before RL and uses no dynamic sampling. For each prompt, we sample 16 trajectories from the RFT model and retain the prompt if its outcomes are mixed: an MR prompt must have both correct and incorrect trajectories, and a KI prompt must have both trajectories with answer F1 > 0.75 and trajectories with answer F1 ≤ 0.75. This fixed subset does not follow the policy as prompts become solved or remain unsolved, which is the curriculum that dynamic sampling provides.

Each training batch contains 64 prompts with $G = 16$ trajectories per prompt, generated by ARPO's entropy-guided branching rollout (8 initial rollouts, beam size 2, branch probability 0.5, entropy weight 0.2) at temperature 1.0 with a maximum response length of 4,096 tokens. For Cost-Aware Rollout, the adaptive half-width equals the EMA of the achieved-cost standard deviation (EMA coefficient 0.9); the static box has half-widths of one think unit (512 tokens) and two tool calls, and is selected with probability 0.125. SVD-LoRA uses $r = 32$ and $\alpha_H = 48$ on all linear projections of every layer. We train $\text{CostDescent}_{\text{static}}$ for three epochs and CostDescent for 300 optimization steps on 8 GPUs. For all trained models, including the retrained ET-Agent, we select the checkpoint with the best validation performance. We use a PPO mini-batch of 4 prompts, one PPO epoch per batch, and constant learning rates of 2×10^{-5} for SVD-LoRA and 10^{-6} for the backbone.

We clip the inverse hindsight-ratio weights to $[\ell_+, u_+] = [0.8, 1.2]$. Optimization uses $\varepsilon_A = 10^{-6}$ for GRPO standardization, $\epsilon_{\text{PPO}} = 0.2$, dual-clip constant $\xi_{\text{dual}} = 3$, token-mean loss aggregation, and $\lambda_{\text{reg}} = 10^{-4}$; the entropy and KL coefficients are zero.

Evaluation Protocol Following ET-Agent, we evaluate up to the first 500 samples of each test set. MR tasks use a Python interpreter, while KI tasks use local retrieval consistent with training. We allow at most six tool calls and 4,096 output tokens per example. Following ET-Agent's public implementation,[3] sampling uses temperature 0.6, top-p 0.95, top-k 20, and repetition penalty 1.1.

C.3 Goal-Response Evaluation

For Fig. 4, we evaluate a 5×7 grid of think- and tool-cost goals with four samples per prompt–goal. We exclude 6 of 68,302 generations whose responses exceed 10,000 tokens.

D Additional Results

D.1 Default-Goal Validation

Fig. 6 reports validation behavior under the default goal $g = 0$ during Stage 1 training. The validation score is the equal-weight mean of the MR score and KI answer F1, where the MR score of a response is 1 if correct, 0 if incorrect, and -1 if malformed. The mean tool calls of CostDescent fall from 1.39 at step 10 to 1.01 at the final step, while its validation score rises from 0.381 to 0.469. Outcome-only ARPO raises its validation score to 0.419 but ends at the same number of tool calls as at step 10 (1.47). ET-Agent reduces tool calls to a minimum of 0.78 at step 80 and ends at 1.09, while its validation score ends at 0.356, below its value at step 10 (0.375). Among the three methods, only CostDescent improves both validation score and tool use at the default goal.

Panel (c) reports a raw-score-based efficiency: the raw validation score of each response divided by $\max(T, 1)$, where T is its number of tool calls, averaged over all validation responses. The raw score equals the MR score on MR and the answer F1 on KI, with -1 for a malformed response and a bonus of 0.1 for a KI response with positive F1 that uses both search and Python. Under this measure, CostDescent rises from 0.263 at step 10 to its highest value, 0.393, at the final step. ET-Agent peaks at 0.374 at step 120 and ends at 0.312, and outcome-only ARPO ends at 0.302.

[3] https://github.com/asilverlight/ET-Agent/

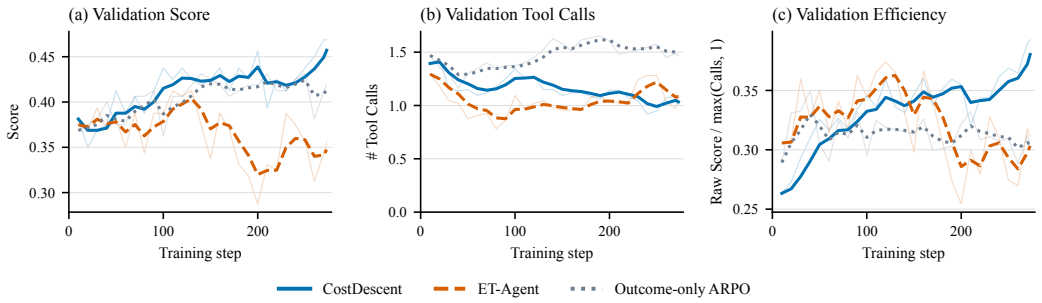


Figure 6: **Default-goal validation.** Validation score (a), mean tool calls (b), and raw-score-based efficiency (c) at $g = 0$ on validation prompts, evaluated every 10 training steps. Bold lines denote exponential moving averages.

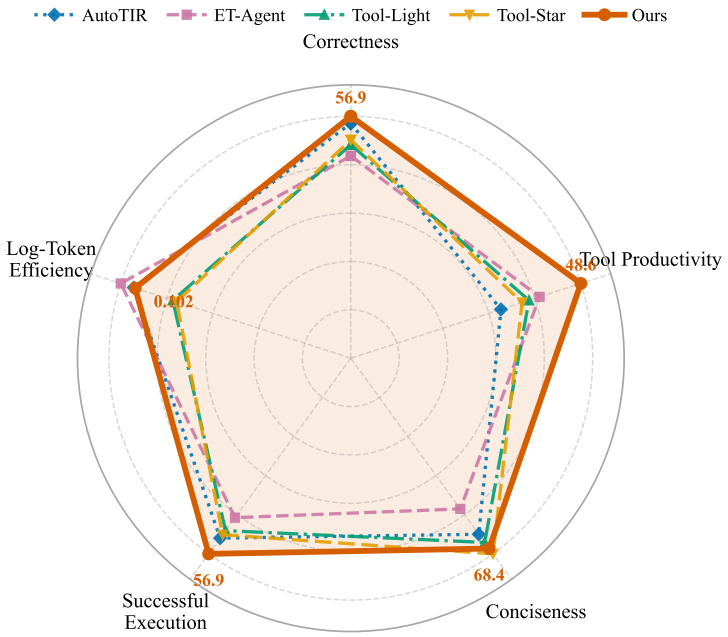


Figure 7: **Overall performance and efficiency profile.** Metrics are macro-averaged over the six benchmarks and normalized by the best displayed value on each axis. Log-token efficiency is $\frac{1}{N}\sum_i P_i/\log(1+L_i)$, where L_i counts model-authored tokens and excludes tool outputs. Higher values are better; orange annotations show the raw values for CostDescent.

D.2 Overall Efficiency Profile

Fig. 7 complements tool productivity with token efficiency. CostDescent leads in correctness, tool productivity, and successful execution, while remaining close to the strongest conciseness and log-token efficiency results.

D.3 Reliability of Reported Results

Table 1 reports a single evaluation pass. We repeat the evaluation five times with different sampling seeds under the same protocol on most competitive baselines, the results are shown in Table 4. The rankings in average task performance and TP are preserved.

Table 4: **Sampling stability across six benchmarks.** Values report mean ± sample standard deviation over five sampling runs. P denotes the task metric (LJ for mathematical reasoning and F1 for knowledge-intensive tasks), and TP denotes tool productivity. Bold indicates the highest mean in each column.

Method	Mathematical-Reasoning Tasks						Knowledge-Intensive Tasks						Avg.	
	AIME24		AMC23		MATH500		2Wiki		Bamb		MSQ			
	LJ	TP	LJ	TP	LJ	TP	F1	TP	F1	TP	F1	TP	P	TP
AutoTIR	25.3 ± 6.5	16.9 ± 1.7	61.5 ± 3.8	37.5 ± 3.8	73.4 ± 0.6	58.1 ± 0.5	**70.9** ± 0.5	30.8 ± 1.1	60.5 ± 1.5	31.5 ± 1.1	**30.8** ± 1.4	13.8 ± 0.6	53.8 ± 1.3	31.4 ± 0.9
ET-Agent	20.0 ± 7.1	16.8 ± 6.6	60.0 ± 4.7	55.2 ± 5.1	66.4 ± 0.9	62.1 ± 0.8	56.1 ± 0.5	**41.9** ± 0.4	45.4 ± 1.4	38.4 ± 1.5	21.8 ± 0.9	14.0 ± 0.8	44.9 ± 1.8	38.1 ± 1.4
CostDescent	**31.3** ± 1.8	**29.3** ± 1.5	**69.5** ± 4.5	**66.8** ± 5.4	**77.0** ± 0.7	**75.5** ± 0.6	67.0 ± 0.8	40.9 ± 1.1	**60.6** ± 1.3	**48.7** ± 2.1	28.4 ± 0.7	**18.9** ± 0.8	**55.6** ± 0.9	**46.7** ± 1.3

Table 5: **Results on Llama-3.1-8B-Instruct.** Columns follow Table 1.

Method	Mathematical-Reasoning Tasks						Knowledge-Intensive Tasks						Avg.	
	AIME24		AMC23		MATH500		2Wiki		Bamb		MSQ			
	LJ	TP	LJ	TP	LJ	TP	F1	TP	F1	TP	F1	TP	P	TP
Llama-3.1-8B-it	3.3	0.5	17.5	11.0	34.2	16.3	38.7	21.6	50.8	35.8	15.0	8.8	26.6	15.7
SMART	3.3	3.3	17.5	15.8	39.4	33.0	21.7	21.1	40.7	40.6	12.3	11.8	22.5	20.9
$\text{CostDescent}_{\text{Llama}}$	**10.0**	**10.0**	**27.5**	**25.8**	**48.0**	**46.0**	**55.8**	**37.6**	**59.2**	**53.3**	**27.5**	**20.6**	**38.0**	**32.2**

D.4 Generalization to Llama

To test whether CostDescent transfers to another model family, we train $\text{CostDescent}_{\text{static}}$ on Llama-3.1-8B-Instruct. The RFT checkpoint follows the ET-Agent recipe (Chen et al., 2026c), and RL prompts use the same fixed mixed-outcome selection as the Qwen $\text{CostDescent}_{\text{static}}$ configuration; the remaining settings follow App. C. Table 5 reports evaluation results.

On training batches (Fig. 8), the outcome score rises from 0.30 to 0.64 while the response length falls from 2,324 to 940 tokens and the tool calls fall from 1.88 to 1.10, raising tool productivity from 0.23 to 0.61.

D.5 Repetition without Credit Inversion

We measure repetition on the training rollouts of $\text{CostDescent}_{\text{static}}$ and its w/o Credit Inversion variant (Table 2). For each rollout, we remove tool outputs and compute the fraction of its model-authored word 4-grams that repeat an earlier 4-gram of the same rollout. A rollout counts as repetitive when this fraction exceeds 0.3. At each training step we record the share of repetitive rollouts in the batch, and we split this share by correctness, with both parts taken over the whole batch so that they sum to the total. Figure 9 shows these shares over the 273 training steps.

Without inversion, the mean share of repetitive rollouts more than doubles from the first to the last third of training (0.92% to 1.98%), while for the full method it stays below 0.8% in every third. Most of the increase comes from correct rollouts, which typically receive the positive advantage that the reweighting redistributes. This is consistent with the self-reinforcing allocation described in Section 4.4: without inversion, positive credit concentrates on tokens that the relabeled goal already favors, analogous to the positive-token weighting imbalance that ASPO identifies for importance sampling (Wang et al., 2026) and that motivates our inversion.

D.6 Case Study: Token-Level Response to Tool Goals

We take a 2WikiMultihopQA rollout and a MATH500 rollout sampled at $g = 0$ and rescore them with teacher forcing under $g_{\text{tool}} = \pm 2$, keeping $g_{\text{think}} = 0$. For each model-authored token y_t, Fig. 10 colors $\Delta_t(g) = \log \pi_\theta(y_t \mid y_{<t}, g) - \log \pi_\theta(y_t \mid y_{<t}, 0)$.

In the first example, the model finds the director of the film and then searches for the director's birthplace. Under $g_{\text{tool}} = -2$, the tokens leading into the second search ("focus on finding out") and the second `<search>` become less likely, while the opening of `<answer>` becomes more likely; under $g_{\text{tool}} = +2$, these changes reverse. The second example counts battalion selections

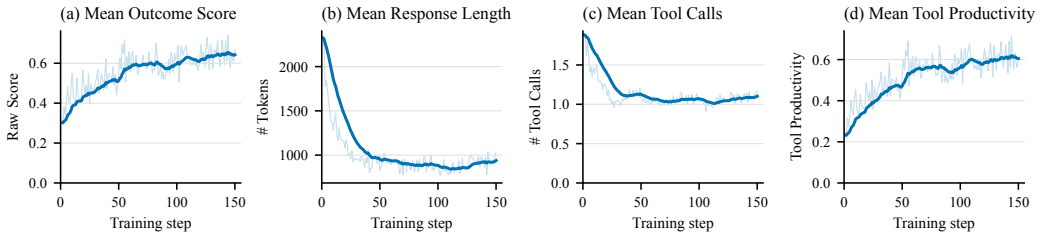


Figure 8: **Training dynamics on Llama.** Training-batch outcome score, response length, tool calls, and tool productivity. Bold lines denote exponential moving averages.

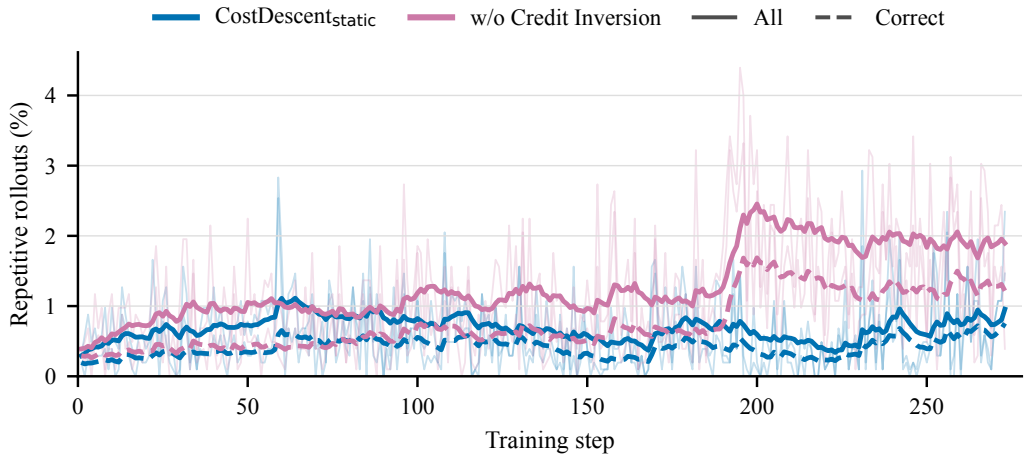


Figure 9: **Share of repetitive training rollouts (%).** Solid lines count all rollouts; dashed lines count correct rollouts, taken over the whole batch. Faint lines are per-step values and bold lines their EMA (smoothing 0.9).

with a single Python call. Under $g_{\text{tool}} = -2$, the `<python>` tag that opens this call becomes less likely. In both examples, $g_{\text{tool}} = -2$ lowers the probability of opening a tool call.

E LIMITATIONS

CostDescent derives its cost target from on-policy rollout groups, so cost descent acts only on groups that contain task-accepted trajectories with positive advantage; otherwise the target stays at the current support mean and exerts no cost-descent pressure. Its efficiency gains therefore depend on how often training groups provide such evidence. Dynamic sampling in our main configuration retains only mixed-outcome groups throughout training, while the curriculum-free setting relies on a subset fixed before RL.



Figure 10: **Token-level response to g_{tool}.** A 2WikiMultihopQA rollout and a MATH500 rollout sampled at $g = 0$ and scored under $g_{\text{tool}} = \pm 2$. Color shows the change in the log-probability of each observed token relative to $g = 0$; tool results are excluded.