FROM BELIEF TO ACTION: A BAYESIAN CO-EVOLUTION FRAMEWORK FOR LLM AGENT SYSTEM

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM-centric agent systems have shown remarkable potential, yet their self-evolution from experience remains limited. Existing approaches either optimize context alone while keeping the agent model fixed, or perform joint adaptation for them across heterogeneous spaces. In both cases, nevertheless, context and model are treated as decoupled entities, lacking a common ground for representing and propagating their mutual uncertainties. We propose a unified Bayesian inference framework that treats context and agent model as coupled variables within a single probability measure space, directly updated by online trajectory likelihoods without parameter retraining. This enables few-shot trial-and-error adaptation, where belief updates for the variables are derived directly from empirical returns. Experiments demonstrate consistent outperformance over both existing paradigms, substantiating the viability of formulating self-evolution as principled inference under coherent probabilistic belief dynamics.

1 INTRODUCTION

Large Language Models (LLMs) have transcended the boundaries of mere text generation, emerging as autonomous agents capable of executing complex reasoning (Wang et al. (2026)), task-solving and decision-making (Wang et al. (2025)). However, classical LLM agents are often static, which relies on pre-trained knowledge and fixed prompting paradigms, thus lacking the internal mechanism to constantly perceive environmental changes and learn from execution feedback Gao et al. (2026). Consequently, when faced with uncertain environments, these agents struggle to adapt, thus leading to constant execution errors and increasing computational costs.

To alleviate this limitation, self-evolving agents have been introduced. By continuously interacting with the environment, these agents facilitate adaptation through unilateral context optimization, refining external prompts, skill libraries, and memories. Existing methods rely on heuristic text-based reasoning and probability-guided search to dynamically reconstruct contexts Laskin et al. (2023); Lee et al. (2023); Kirsch et al. (2023); Moeini et al. (2025). To mitigate the blindness of heuristic trial-and-error, subsequent works introduced probability-guided frameworks and Bayesian optimization, formulating context updates as prior-to-posterior refinements based on offline historical trajectories Xie et al. (2022); Zhou et al. (2023); Tonolini et al. (2024); Wu et al. (2026c). Despite providing rigorous theoretical guidance, these probability-driven methods heavily rely on the scale of offline data and lack immediate online adaptability.

Since unilateral context optimization is insufficient for adaptation, researchers have also explored post-training to update the LLM's internal weights Gu et al. (2026); Wu et al. (2026b). While isolating parameter updates from the external context leads to challenges such as context explosion and severe sample inefficiency Liu et al. (2026b); Zong et al. (2026); Tan et al. (2026). To bridge this gap, recent research has focused on the joint optimization of contexts and agents. Nevertheless, the prevailing paradigm typically adopts a discrete alternating iteration strategy Chen et al. (2026c). Specifically, these systems freeze the context after multi-step exploration to apply a post-training paradigm Hebbar et al. (2026), and subsequently revert to context optimization. Although achieving a form of co-evolution, this alternating paradigm still treats the harness and policy as decoupled entities, performing updates independently across the text and parameter spaces.

It is also worth noting that a common gap remains in the above works: they fail to achieve an online and simultaneous co-evolution between the context and the agent model within a unified measure

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.

space. Namely, context-optimization methods optimize the context in text space but leave the agent model in a null space, whereas joint methods update the model in parameter space. To bridge this gap, we propose a unified Bayesian inference framework. Specifically, our framework models the context and the agent policy as coupled variables within the probability space. Agents can directly utilize online trajectory likelihood to **jointly** and **simultaneously** update these two variables, eliminating the need to retrain the underlying parameters. More importantly, it avoids the need (in conventional joint methods) for them to be optimized sequentially and to continually adapt to each other. Overall, the contributions of our work are summarized as follows:

- We carefully design the context variable and the agent model variable in the probability space, which abstract the processes of context modification and model generation, respectively. This formulation eliminates the need for any model parameter updates.
- We propose a unified Bayesian inference framework that jointly updates these variables using only a small number of trial-and-error trajectories from environment interaction, yielding substantial gains in task completion rates.
- We establish rigorous theoretical guarantees that the proposed joint update method provably increases task trajectory returns across iterations, providing a solid foundation for the co-evolution of context and agent model.

2 Related Work

Unilateral Context Optimization Existing methods typically achieve context optimization through heuristic text-based reasoning, employing strategies like contextual information density maximization Liang et al. (2026), Pareto-based evolutionary prompt search Agrawal et al. (2026), or verbalized joint refinement of instructions and tool documentation Wu et al. (2025). Furthermore, some works introduce quantitative probabilistic models and rigorous governance to guide skill evolution. These approaches span a diverse spectrum of mechanisms: posterior-guided search for harness optimization Wu et al. (2026c), bounded text-space optimization via controlling agent skills Yang et al. (2026), contrastive causal extraction to prevent unvalidated merging Chen et al. (2026b), full-lifecycle governance for skill recommendation and attribution Liu et al. (2026a), and ICPO formulates multi-round self-reflection Yu et al. (2026). While these approaches remain unilateral context optimizations, only updating harness does not translate to actual harness benefit for the agent Lin et al. (2026).

Alternating joint optimization The latest research explores joint optimization between the external context and internal reasoning. HarnessForge explicitly separates harness-level execution from policy-level reasoning, alternating between fault-guided harness tailoring and policy alignment Chen et al. (2026c). SIA employs a dual-track loop that dynamically switches between updating the task scaffold and applying RL to model weights when progress stalls Hebbar et al. (2026). Similar alternating lifecycles are observed in Continual Harness, which iterates between online prompt refinement and model relabeling Karten et al. (2026) as well as in EvolveR Wu et al. (2026a), SkillRL Xia et al. (2026), and SLEA-RL Wang & Jiang (2026), which co-evolve external experience or skill libraries alongside policy optimization. Although these methods successfully achieve adaptation, they isolate the context and the model into decoupled optimization spaces Lu et al. (2026). This alternating paradigm limits the simultaneous updating of mutual uncertainty between the harness and the policy, incurs computational overhead from repeated retraining phases, and prevents truly online, unified belief updates.

3 Methodology

As established aforementioned, we characterize self-evolution as the capacity of an agent system to concurrently refine its contextual description (**Context**) and generative behavior (**Agent Model**) from limited environmental interactions, drawing upon the trial-and-error trajectories as empirical evidence to jointly update both variables' [1] belief toward successful task resolution. Formally, we cast the core problem as:

[1]We refer to variables, i.e., ϕ and θ, as the guide to control each component's operation: ϕ modulates contents of the context for further generation, and θ steers generative procedure of the agent model.

Problem Definition: Given an agent system composed of context (which encapsulates textual constituents such as skills and prompts) and the agent model, with their variables initialized at $\phi = \phi_1$ and $\theta = \theta_1$, respectively. For a designated task, the agent interacts with the environment over multiple episodes, yielding a stream of trajectories $\mathcal{T} = \{\tau_1, \cdots, \tau_K\}$. Each trajectory $\tau_k \in \mathcal{T}$ is generated autoregressively under the current belief (ϕ_k, θ_k), and subsequently serves the evidence to update the belief to $(\phi_{k+1}, \theta_{k+1})$, under the prescription of *Bayesian Inference*. We introduce v_τ to denote trajectory return, and consequently, our objective is to devise **a configuration of variables and component controls** such that the return of each successive trajectory improves upon the last. By the end of the episode sequence, the final updated belief over the variables is expected to guide generation of the trajectory with substantial return, thereby increasing the possibility of successful task completion. Namely:

Table 1: Key Notations and Definitions for Our Co-Evolution Framework.

Variable	Description
τ (trajectory)	the full agent–env interaction
s_t	interaction before step t (**State**)
a_t	agent model response (**Action**)
r_t	the instance reward at step t
v_τ	the full cumulative gain
$R_{<t}$	the cumulative gain before step t
ϕ	context variable (**Categorical**)
χ_i, ω_i	value and probability for ϕ
θ	agent model variable (**Gaussian**)
$\mu, \sigma, \delta_i, \sigma_\tau$	parameters for θ related
k, K	number of iterations
m	number of categories for ϕ
t, T	interaction steps per iteration

$$\text{Find } (\phi, \theta, \mathcal{G}) : p(\phi_{k+1}, \theta_{k+1}) \leftarrow p(\phi_k, \theta_k | v_{\tau_k}), \tau_k = \mathcal{G}(\phi_k, \theta_k, d), \text{ s.t. } \mathbb{E}[v_{\tau_k}] < \mathbb{E}[v_{\tau_{k+1}}], \quad (1)$$

where $\mathcal{G}(.)$ means the trajectory generation process, which takes the task description d and variable values as input to govern component controls; the variables are updated via standard Bayesian inference, i.e., revising their beliefs based on the posterior conditioned on the observed trajectory.

3.1 The Variables Design

The Context Variable ϕ: For the context variable, we formalize it as a categorical random variable ϕ taking values in a finite set of m discrete modes, i.e., $\phi = \{\chi_1, \cdots, \chi_m\}$, where each mode χ_i corresponds to a specific constraint type governing the modification of the context. These constraints may specify, for instance, the target location of modification and the operation type (e.g., deletion, substitution). The actual modification process of the context can be effectively realized by leveraging the LLM as a generative engine (where $\mathcal{C}$ means the context content):

$$\mathcal{C} \leftarrow \text{LLM}(\mathcal{C}, \chi_i, \text{Revise the context in accordance with the prescribed constraint}). \quad (2)$$

The distribution over ϕ is characterized by a categorical probability mass function:

$$p(\phi = \chi_i) = \omega_i, \text{ s.t. } \sum_{i=1}^{m} \omega_i = 1, \omega_i \geq 0. \quad (3)$$

In this manner, through the sequential accumulation of experience from each trajectory, the categorical distribution over the constraint types (parameterized by each ω_i) undergoes iterative refinement, eventually converging to a distribution that is increasingly adapted to the task, thereby steering the ultimate update of the context. This discrete categorical representation obviates the intractable dimensionality inherent in the text space, facilitating efficient context editing.

The Agent Model Variable θ: Instead of performing optimization directly over the parameters of the agent model (as is conventional in prevailing post-training paradigms for LLMs), which incurs prohibitive computational costs and is often impractical, we introduce a continuous scalar quantity θ that serves as an estimate of the *expected return* associated with executing the given task. **In other words, θ represents an estimation of the interaction trajectory gain, whereas v_τ (Equation 1) quantifies the realized return of the trajectory**. This variable effectively summarizes the task-level performance potential, providing a compact reference for the generation guidance. To capture the inherent uncertainty in our estimate of the expected return, we adopt a Gaussian formulation (a standard and widely justified choice for modeling continuous-valued estimates), offering both

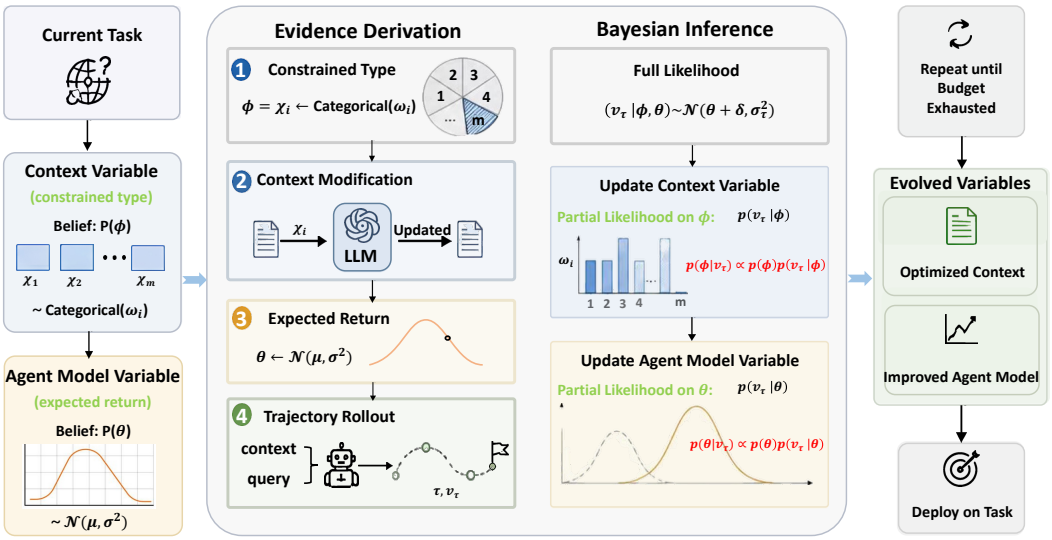


Figure 1: Overview of the co-evolution framework. As described in Algorithm 1, for each new task, the context and agent model variables are first initialized (**first column**). At each iteration, both variables are sampled from their distributions to guide trajectory generation (**second column**), followed by Bayesian inference for distribution updates (**third column**). The final distributions are used to solve the task after several iterations (**fourth column**).

analytical tractability and principled uncertainty quantification. Specifically, conditional on ϕ, which specifies the constraint type governing context modification, θ is assumed to follow a Gaussian distribution. As a consequence, the marginal distribution of θ is a Gaussian mixture model (GMM):

$$\theta \sim \sum_{i=1}^{m} \omega_i \mathcal{N}(\mu + \delta_i, \sigma^2), \;\; \text{s.t. } (\theta \mid \phi = \chi_i) \sim \mathcal{N}(\mu + \delta_i, \sigma^2), \; \sum_{i=1}^{m} \omega_i \delta_i = 0, \tag{4}$$

where μ represents the global mean return, and δ_i captures the deviation induced by the $i-$th constraint type. σ^2 denotes the shared variance. Based on this, **the marginal distribution of θ is well-approximated by a single Gaussian** $\mathcal{N}(\mu, \sigma^2)$ (see Section A.1 for a formal derivation).

3.2 The Component Control

The overall trajectory generation process, i.e., $\mathcal{G}(.)$ in Equation 1, proceeds in two stages: context modification and agent model generation. In the first stage, the context is adapted according to the discrete constraint type specified by ϕ, as detailed in Equation 2, where the LLM generates a modified content based on the selected χ_i, and feeds it into the agent model. In the second stage, the agent model generates actions over a horizon of T steps for current task (task description d and context $\mathcal{C}$ as the beginning input), with the trajectory being $\tau = (d, \mathcal{C}, a_1, u_1, \cdots, a_T, u_T)$. Here a_t denotes the agent model generation, and u_t is the tool execution output.

At each step t, we rectify the generation probability of action a_t by incorporating the agent model variable θ as a bias term. Following prior works Liu et al. (2024); Li et al. (2026), this is achieved by augmenting the original action distribution, i.e., $p(a_t|s_t)$ with an exponential advantage weight (a detailed derivation of our advantage formulation is provided in Section A.2):

$$p^*(a_t|s_t) \propto p(a_t|s_t) \exp\left(A(s_t, a_t)\right),$$
$$\text{s.t. } A(s_t, a_t) = r_t - (1 + \kappa \log p(a_t|s_t)) \frac{\theta - R_{<t}}{T - t + 1}, \; R_{<t} = \sum_{i=1}^{t-1} r_i, \tag{5}$$

where s_t represents the current state (can be expressed as $s_t = (d, \mathcal{C}, a_1, u_1, \cdots, a_{t-1}, u_{t-1})$). r_t is the immediate reward for action a_t, and $R_{<t}$ denotes the cumulative reward prior to step t. The term $\frac{\theta - R_{<t}}{T-t+1}$ quantifies the average per-step adjustment needed to steer the remaining steps toward the

optimal estimate θ, while $1 + \kappa \log p(a_t|s_t)$ modulates this adjustment to balance exploration and exploitation. **In the concrete implementation, at each step, the agent model generates multiple actions, and the final normalization is performed over these actions.**

3.3 Bayesian Inference Procedure

The Full Likelihood: Since both v_τ and θ measure the task return, with v_τ being the realized return of the executed trajectory and θ being the global estimate of the optimal return prior to execution, they share the same semantic scale. It is therefore natural to adopt a similar additive structure for the full likelihood as Equation 4. Specifically, we model the actual trajectory return v_τ conditional on ϕ and θ as:

$$(v_\tau \mid \phi = \chi_i, \theta) \sim \mathcal{N}(\theta + \delta_i, \sigma_\tau^2). \tag{6}$$

Here δ_i appears in both Equations 4 and 6, but its semantic role differs, which does not constitute a double-counting of the same bias: in the first equation it is the belief-level (θ) bias induced by the constraint type, while in the second it is the outcome-level (v_τ) bias induced by the same constraint type. **Notably**, sharing the same scalar encodes the inductive bias that the constraint type affects both levels in a consistent direction and magnitude, meanwhile, the variances σ^2 (Equation 4) and σ_τ^2 (Equation 6) are allowed to differ, absorbing the residual uncertainties specific to each level.

For a clear exposition of the iterative belief update procedure, in the subsequent text, we employ the form given in Equation 1 to differentiate variables and trajectory between iterations, i.e., ϕ_k, θ_k, τ_k, and utilize superscripts to denote the corresponding parameters update, i.e., $\omega_i^{(k)}, \mu^{(k)}, (\sigma^{(k)})^2, \delta_i^{(k)}$. The specific execution process is detailed in Algorithm 1.

Belief Update for Context Variable ϕ: We perform updates for the variables from a partial likelihood perspective. For the context variable at iteration k, we first compute its associated partial likelihood, i.e., the distribution of v_τ conditioned on ϕ, which is given by $(v_{\tau_k}|\phi_k = \chi_i) \sim \mathcal{N}(\mu^{(k)} + 2\delta_i^{(k)}, (\sigma^{(k)})^2 + \sigma_{\tau_k}^2)$ (referring to Lemma 1 for derivation), where $\mu^{(k)}$ and $\sigma^{(k)}$ are parameters mentioned in Equation 4, and σ_{τ_k} is for the full likelihood parameter in Equation 6. $\delta_i^{(k)}$ is the context constrained type bias in both 4 and 6. Then, the posterior update for ϕ based on the trajectory return can be further expressed via Bayes' rule as (see Lemma 3):

$$\begin{aligned}\omega_i^{(k+1)} &= p(\phi_{k+1} = \chi_i) \leftarrow p(\phi_k = \chi_i | v_{\tau_k}) \\ &\propto \begin{cases} \frac{\omega_i^{(k)}}{\sqrt{2\pi((\sigma^{(k)})^2 + \sigma_{\tau_k}^2)}} \exp\left(-\frac{(v_{\tau_k} - (\mu^{(k)} + 2\delta_i^{(k)}))^2}{2((\sigma^{(k)})^2 + \sigma_{\tau_k}^2)}\right) & \text{if } \chi_i \text{ is selected,} \\ \frac{\omega_i^{(k)}}{\sqrt{2\pi((\sigma^{(k)})^2 + \sigma_{\tau_k}^2)}} \exp\left(-\frac{(v_{\tau_k} - \mu^{(k)})^2}{2((\sigma^{(k)})^2 + \sigma_{\tau_k}^2)}\right) & \text{otherwise.} \end{cases}\end{aligned} \tag{7}$$

Note that at each iteration k, we sample only one constrained type i.e., $\phi_k = \chi_i$, and can therefore update only the probability value associated with that type via the Bayesian inference. For the types that are not selected, however, careful design is required: on the one hand, their relative probabilities must be kept unchanged, i.e., $\frac{\omega_{j_1}^{(k+1)}}{\omega_{j_2}^{(k+1)}} = \frac{\omega_{j_1}^{(k)}}{\omega_{j_2}^{(k)}} (\forall j_1, j_2 \neq i)$; on the other hand, the relative probability of the selected type should reflect its actual return—that is, if the return of the selected type falls below expectation, its relative probability should decrease, i.e., $\frac{\omega_i^{(k+1)}}{\omega_j^{(k+1)}} < \frac{\omega_i^{(k)}}{\omega_j^{(k)}}$. The update in the above equation is designed precisely to guarantee these properties.

Theorem 1 *Given $\mathbb{E}[v_{\tau_k} - \theta_k] > 0$ (being proved in Theorem 2), if the trajectory induced by χ_i yields a return expectation below the mean, i.e., $\delta_i^{(k)} < 0$, then the corresponding probability of χ_i decreases, i.e., $\omega_i^{(k+1)} < \omega_i^{(k)}$. This serves to penalize and downweight the constraint type χ_i that leads to inferior performance. (Proof A.4)*

Belief Update for Agent Model Variable θ: Similarly, we update the agent model variable θ based on its partial likelihood. Specifically at iteration k, the likelihood of θ is obtained by marginalizing out the context variable from the full likelihood distribution: $(v_{\tau_k}|\theta_k) \sim \mathcal{N}(\theta_k, \sigma_{\tau_k}^2)$ (see

Table 2: Overall comparison on GLM-5.2 model. The *Score* denotes the gain of an execution trajectory. *Pass* ^{i} represents the ratio of instances that successfully pass all i independent executions (HIGH is better); Δ *P* $^{1\to i}$ means the difference between Pass ^{1} and Pass ^{i} (LOW is better).

	DeepSearch		**τ^2-Bench**							
Method	Acc.	Score	Pass^1	Pass^2	Pass^3	Pass^4	Aver.	Δ P $^{1\to2}$	Δ P $^{1\to3}$	Δ P $^{1\to4}$
Context-Design	50.0	0.402	60.3	51.2	45.8	42.1	49.9	9.1	14.5	18.2
Reflection	73.0	0.586	61.8	52.0	46.3	42.1	50.6	9.8	15.5	19.7
Reasoning-Bank	48.0	0.364	62.7	52.3	45.6	40.4	50.3	10.4	17.1	22.3
Bayesian-Agent	71.0	**0.630**	62.1	51.3	44.1	38.6	49.0	10.8	18.0	23.5
SkillGen	53.0	0.408	62.5	53.4	47.8	43.9	51.9	9.1	14.7	18.6
Ours	**75.0**	0.624	**74.3**	**67.3**	**62.7**	**59.6**	**66.0**	**7.0**	**11.6**	**14.7**

Lemma 2). Following the same rationale, the posterior distribution of θ given the observed return is obtained by applying Bayes' rule to its marginal likelihood (see Lemma 4):

$$\begin{aligned}\mathcal{N}(\mu^{(k+1)}, (\sigma^{(k+1)})^2) = p(\theta_{k+1}) \leftarrow p(\theta_k|v_{\tau_K}) \propto p(v_{\tau_k}|\theta_k)p(\theta_k) \\ = \mathcal{N}(\frac{\mu^{(k)}/(\sigma^{(k)})^2 + v_{\tau_k}/\sigma^2_{\tau_k}}{1/(\sigma^{(k)})^2 + 1/\sigma^2_{\tau_k}}, \frac{1}{1/(\sigma^{(k)})^2 + 1/\sigma^2_{\tau_k}}).\end{aligned} \tag{8}$$

This suggests that we can maintain a Gaussian representation of the posterior, and update the prior of θ using its estimated mean and variance, enabling iterative refinement over successive trajectories.

Theorem 2 *The Bayesian update of θ ensures that trajectories of increasing return are generated over successive iterations. Furthermore, the θ itself evolves monotonically toward higher values, accordingly. Namely:*

$$\mathbb{E}[v_{\tau_k} - \theta_k] > 0, \ \mathbb{E}(\theta_{k+1}) > \mathbb{E}(\theta_k). \tag{9}$$

This coupled improvement guarantees that the estimate of the optimal return increases concurrently with the observed trajectory performance, consistent with the expected enhancement in task completion probability through iterative trial-and-error (Proof A.4).

4 Experiment

In this section, we empirically validate our proposed method on the DeepSearch and τ^2-bench datasets. We employ GLM-5.2 and DeepSeek-V4-Flash (both adopting FP8 quantization) as the backbone agent models to conduct a comparative analysis of different context optimization strategies. Furthermore, we benchmark our approach against joint optimization baselines on Qwen-3.8-27B. We also perform comprehensive ablation studies and examine trajectory-level reward gains, which collectively underscore the key advantages of our framework.

Datasets and Baselines: DeepSearch Chen et al. (2025) is designed to assess agents' end-to-end deep search capabilities, requiring the integrated application of planning, search, reasoning, and summarization. τ^2-Bench Barres et al. (2025) is a dual-control environment where the agent and a simulated user can take actions using tools, evaluating not only reasoning but also coordination and communication in shared dynamic settings. (extra tool usage information is in Section B)

For context optimization, we compare against five baselines: *Context-Design*, *Reflection*, *Reasoning-Bank*, *Bayesian-Agent*, and *SkillGen*. Additionally, we adopt two post-training baselines for joint optimization: *Iterative-Alternating training* (where the context and agent model are optimized sequentially per round) and *Full-Adaptation training* (where one component is fully converged before the other is optimized). (refer to Section B for further details)

Context Modification and Reward Design: We decompose the context-relevant content into six distinct regions, denoted as $\mathcal{C}_{\text{base}}$ (the basic rule), $\mathcal{C}_{\text{query}}$ (the retrieval rule), $\mathcal{C}_{\text{action}}$ (the action

Table 3: Overall comparison on DeepSeek-V4-Flash agent model. The metrics are used with the same meanings as specified in the preceding context.

Method	DeepSearch		τ^2-Bench							
	Acc.	Score	Pass^1	Pass^2	Pass^3	Pass^4	Aver.	Δ P $^{1\to2}$	Δ P $^{1\to3}$	Δ P $^{1\to4}$
Context-Design	42.0	0.298	82.0	73.5	67.8	63.2	71.6	8.5	14.2	18.8
Reflection	49.0	0.406	62.9	54.1	49.1	45.6	52.9	8.8	13.8	17.3
Reasoning-Bank	27.0	0.202	82.7	74.9	69.3	64.0	72.7	7.8	13.4	18.7
Bayesian-Agent	55.0	**0.480**	79.6	69.0	61.4	56.1	66.5	10.6	18.2	23.5
SkillGen	35.0	0.284	79.8	70.5	64.7	60.5	68.9	9.3	15.1	19.3
Ours	**58.0**	0.474	**89.7**	**85.8**	**83.8**	**82.5**	**85.5**	**3.9**	**5.9**	**7.2**

generation rule), $\mathcal{C}_{\text{verify}}$ (the result verification rule), $\mathcal{C}_{\text{format}}$ (the output format rule) and $\mathcal{C}_{\text{stop}}$ (the termination-related rule), which correspond to each value of χ in Equation 3 for the context variable ϕ (the number of categories $m = 6$ as the consequence; see Section C for details). During the evolution process, one of these regions is sampled according to the current distribution of ϕ, and the new context content is subsequently generated following Equation 2.

The reward (r_t in Equation 5) is designed as a comprehensive rubric-based metric that encapsulates a holistic evaluation of the current action. Guided by the context rules, the agent model assesses the output across four dimensions, including *Context Rules Compliance*, *Coverage of Evidence*, *Plan Completion*, and *Computational Verification*. The reward value is then normalized to $[0, 1]$ (see Section C, reward part, for details).

Metrics: We adopt the evaluation metrics native to both benchmarks to assess our method. Specifically, for DeepSearch, we primarily report task execution **Accuracy**, defined as the ratio of final results that match the ground truth. Additionally, we report the **Score**, a comprehensive measure derived from multi-perspective comparisons between the executed answer and the gold answer, which is then normalized to assess overall trajectory quality (see Section C, Score part). For τ^2-bench, we adopt **Pass**^{i} ($i \in \{1, 2, 3, 4\}$) as the primary evaluation metric, which denotes the proportion of instances that successfully pass all i independent executions per query. We further report $\Delta\mathbf{P}^{1\to i}$ ($=$ Pass^1 $-$ Pass^i) as an indicator of task execution stability.

4.1 Main Result

Table 4: Overall results on Qwen-3.8-27B agent model. We perform a comparison with two joint training strategies.

Method	DeepSearch		τ^2-Bench							
	Acc.	Score	Pass^1	Pass^2	Pass^3	Pass^4	Aver.	Δ P $^{1\to2}$	Δ P $^{1\to3}$	Δ P $^{1\to4}$
Base-Model	30.0	0.226	67.3	57.9	51.3	46.5	55.8	9.4	16.0	20.8
Iterative-Alternating	39.0	0.327	71.1	64.0	58.8	55.3	62.3	7.1	12.3	15.8
Full-Adaptation	41.0	0.318	73.7	66.7	60.5	55.3	64.1	7.0	13.2	18.4
Ours	**50.0**	**0.376**	**76.8**	**72.8**	**70.6**	**69.3**	**72.4**	**4.0**	**6.2**	**7.5**

As displayed in Tables 2 and 3, on the DeepSearch task, our approach attains the highest accuracy on both GLM-5.2 (75.0 vs. 73.0 from Reflection, the best baseline) and DeepSeek-V4-Flash (58.0 vs. 55.0 from Bayesian-Agent). Notably, while Bayesian-Agent yields slightly higher Score on both backbones (0.630 on GLM and 0.480 on DeepSeek), our method achieves a competitive one (0.624 and 0.474, respectively) while delivering markedly better accuracy. This suggests that our co-evolution framework prioritizes task correctness over trajectory reward optimization, striking a favorable balance between outcome quality and process efficiency.

Table 5: Ablation results on DeepSearch (Acc. and Score) and τ^2-bench (Pass ^{i}) tasks. ϕ Optim.: optimizing context variable only; θ Optim.: optimizing agent model variable only.

Backbone	GLM						DeepSeek					
Configuration	Acc.	Score	Pass^1	Pass^2	Pass^3	Pass^4	Acc.	Score	Pass^1	Pass^2	Pass^3	Pass^4
Ours	**75.0**	**0.624**	**74.3**	**67.3**	**62.7**	**59.6**	**58.0**	**0.474**	**89.7**	**85.8**	**83.8**	**82.5**
ϕ Optim.	71.0	0.590	69.7	61.5	56.6	52.6	54.0	0.412	87.7	82.3	78.7	76.3
θ Optim.	70.0	0.550	71.7	64.5	60.3	57.0	55.0	0.406	87.9	81.3	77.2	74.6

On τ^2-Bench, our method exhibits even more pronounced advantages. On GLM, our method outperforms the best baseline (SkillGen, the best average Pass^i) by 11.8 percentage points on Pass^1 (74.3 vs. 62.5), and the gap widens as i increases—Pass^4 improves by 15.7 (59.6 vs. 43.9). This trend is further amplified on DeepSeek backbone, where our method achieves Pass^1 of 89.7, surpassing the strongest baseline (Reasoning-Bank, 82.7) by 7.0, and Pass^4 of 82.5, exceeding it by 18.5. The average Pass^i score reaches 66.0 on GLM and 85.5 on DeepSeek, far above all baselines (ranging 49.0–51.9 and 52.9–72.7, respectively). Furthermore, our method consistently yields the lowest $\Delta P^{1\to i}$ across all i values, reducing performance degradation across repeated executions. Specifically on DeepSeek, $\Delta P^{1\to 4}$ is reduced to 7.2, compared to 17.3–23.5 for the baselines. This demonstrates that our co-evolution framework not only improves overall success rates but also enhances robustness against multi-run variance.

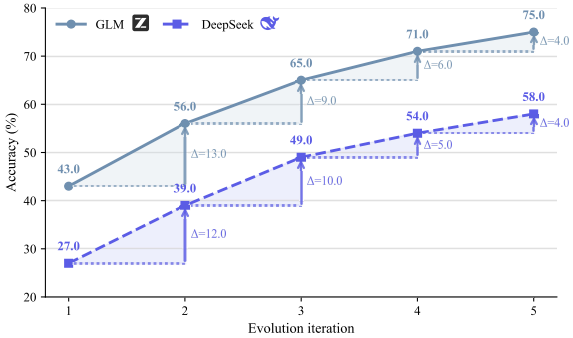


Figure 2: Task completion rate over co-evolution iterations. Both models exhibit sustained performance improvements with increasing iteration count.

We also evaluate joint optimization methods on Qwen-3.8-27B, with results reported in Table 4. Our method consistently outperforms both joint training baselines. On DeepSearch, it attains 50.0 accuracy and 0.376 score, compared with 39.0/0.327 for Iterative-Alternating and 41.0/0.318 for Full-Adaptation. On τ^2-Bench, our method achieves a Pass^1 of 76.8, yielding an average of 72.4 on the Pass^i metrics, whereas Full-Adaptation obtains 73.7/64.1 and Iterative-Alternating obtains 71.1/62.3. Our method also achieves the lowest $\Delta P^{1\to i}$, indicating superior execution stability.

4.2 ABLATION STUDY

The ablation study isolates the contribution of each component in our co-evolution framework. As shown in Table 5, optimizing either the context variable alone (ϕ Optim.) or the agent model variable alone (θ Optim.) consistently underperforms our full co-evolution approach.

On DeepSearch with GLM, it reaches 75.0 accuracy and 0.624 Score, exceeding ϕ Optim. (71.0, 0.590) and θ Optim. (70.0, 0.550). While with DeepSeek, the values are 58.0/0.474, outperforming ϕ Optim. (54.0, 0.412) and θ Optim. (55.0, 0.406). This indicates that joint optimization delivers greater task effectiveness than optimizing either component alone.

On τ^2-Bench, our method improves Pass^1 by 4.6 points over ϕ Optim. (74.3 vs. 69.7) and by 2.6 over θ Optim. (74.3 vs. 71.7). The advantage amplifies as i increases: for Pass^4, our method outperforms ϕ Optim. by 7.0 and θ Optim. by 2.6. On DeepSeek, our method achieves Pass^1 of 89.7, compared to 87.9 (θ Optim.) and 87.7 (ϕ Optim.), and the margin grows to 6.2 over ϕ Optim. at Pass^4 (82.5 vs. 76.3) and 7.9 over θ Optim. (82.5 vs. 74.6). These widening gaps with increasing i indicate that co-evolution not only improves initial success rates but also substantially enhances robustness across repeated executions.

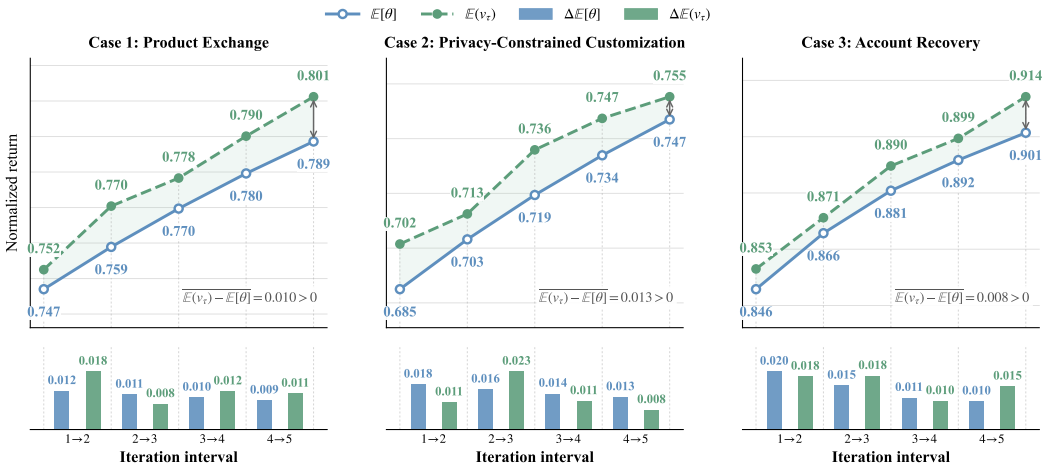


Figure 3: Case evaluation of $\mathbb{E}(\theta)$ and trajectory return $\mathbb{E}(v_\tau)$ across evolution iterations. The values are normalized by step counts (T in Equation 5) for cross-query comparability.

4.3 Empirical Validation of Theoretical Claims

We conduct experiments to empirically validate several key claims of our theory. Figure 2 illustrates the task completion rate as a function of the number of co-evolution iterations (k in Equation 1) for GLM and DeepSeek backbones. As shown, both models exhibit consistent performance improvements with increasing k, yet the gains progressively diminish. For GLM, the completion rate rises from 43.0% at $k = 1$ to 75.0% at $k = 5$ (the growth procedure: iteration $1 \rightarrow 2$, $+13.0$; $2 \rightarrow 3$, $+9.0$; $3 \rightarrow 4$, $+6.0$; $4 \rightarrow 5$, $+4.0$). A similar trend is observed on DeepSeek, completion rates progress from 27.0% at $k = 1$ to 58.0% at $k = 5$ (the growth procedure: iteration $1 \rightarrow 2$, $+12.0$; $2 \rightarrow 3$, $+10.0$; $3 \rightarrow 4$, $+5.0$; $4 \rightarrow 5$, $+4.0$). This diminishing-gains pattern suggests that the most substantial benefits of co-evolution are realized within the first few iterations, where the context and agent model rapidly adapt to align with task requirements. Subsequent iterations yield marginal improvements, indicating that the framework approaches a performance saturation point.

Figure 3 presents the evolution of $\mathbb{E}(\theta)$ and $\mathbb{E}(v_\tau)$ across iterations, where we sample three queries and report both quantities at each iteration (to ensure comparability across different queries, we normalize the cumulative gains by the number of steps, yielding a step-normalized value for each trajectory). As displayed, for all samples, $\mathbb{E}(v_\tau)$ consistently exceeds $\mathbb{E}(\theta)$ at every iteration, and both quantities exhibit a clear upward trend as the co-evolution process proceeds, e.g., for Query 1, $\mathbb{E}(\theta)$ increases from 0.747 to 0.789 over five iterations, while $\mathbb{E}(v_\tau)$ rises from 0.752 to 0.801. The consistent superiority of $\mathbb{E}(v_\tau)$ over $\mathbb{E}(\theta)$ across all iterations and queries empirically corroborates Theorem 2, which ensures that, at each iteration, the distribution of θ can effectively guide the generation of trajectories with higher return. Subsequently, the actual trajectory return serve as evidence to refine θ, which in turn elevates $\mathbb{E}(\theta)$ in the next iteration. This virtuous cycle, where improved trajectory quality drives better parameter estimates, and better estimates in turn yield higher quality trajectory, is precisely reflected in the synchronized upward progression of both $\mathbb{E}(v_\tau)$ and $\mathbb{E}(\theta)$. This iterative enhancement mechanism guarantees that our method consistently increases the gains of generated trajectories over time, leading to steady improvements in task completion rates.

5 Conclusion

In this paper, we propose a joint optimization strategy that co-evolves the context and the agent model for LLM-centric agent systems. Through carefully designed variables, our approach enables adaptation in the probability space, refining both context modification and model generation from a limited number of trial-and-error interaction trajectories with the environment, thereby improving task completion rates. Notably, the proposed variables require no updates to the model parameters, and we provide theoretical guarantees on the monotonic increase of task trajectory returns. Extensive experiments against both context optimization and joint optimization baselines demonstrate the consistent superiority of our method.

AI USE STATEMENT

In this paper, AI tools are used solely for language polishing, such as refining word choices and expressions, as well as checking and adjusting redundant descriptions.

ETHICS STATEMENT

This paper contains no potentially harmful content, including human subjects, harmful insights or applications, conflicts of interest, discrimination or bias concerns, privacy and security issues, legal compliance, or research integrity issues, etc.

REPRODUCIBILITY STATEMENT

Our method is simple to implement. As summarized in Algorithm 1, Section D details the initialization, Section 3.2 describes the full trajectory generation procedure, and Section 3.3 gives the variable update rules; together, these suffice to reproduce the entire pipeline. Notably, our design involves no complicated training process and operates entirely at inference time, making it readily reproducible.

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, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=RQm2KQTM5r.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment, 2025. URL https://arxiv.org/abs/2506.07982.

Guhong Chen, Yingcheng Shi, Yongbin Li, Binhua Li, Xander Xu, Hu Wei, Shiwen Ni, Min Yang, and Jieping Ye. Evotrainer: Co-evolving llm policies and training harnesses for autonomous agentic reinforcement learning, 2026a. URL https://arxiv.org/abs/2606.03108.

Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, Chen Sun, Han Hou, Hui Yang, James Pan, Jianan Lou, Jiayi Mao, Jizheng Liu, Jinpeng Li, Kangyi Liu, Kenkun Liu, Rui Wang, Run Li, Tong Niu, Wenlong Zhang, Wenqi Yan, Xuanzheng Wang, Yuchen Zhang, Yi-Hsin Hung, Yuan Jiang, Zexuan Liu, Zihan Yin, Zijian Ma, and Zhiwen Mo. xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations, 2025. URL https://arxiv.org/abs/2506.13651.

Kunfeng Chen, Qihuang Zhong, Juhua Liu, and Bo Du. Skillcat: Contrastive, assessment-augmented and topology-awareskill self-evolution for llm agents, 2026b. URL https://arxiv.org/abs/2606.13317.

Mingju Chen, Can Lv, Guibin Zhang, Heng Chang, and Shiji Zhou. Harnessforge: Joint harness and policy evolution for adaptive agent systems, 2026c. URL https://arxiv.org/abs/2606.01779.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Qihan Ren, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, Cheng Qian, Zhenhailong Wang, Minda Hu, Huazheng Wang, Qingyun Wu, Heng Ji, and Mengdi Wang. A Survey of Self-Evolving Agents: What, When, How, and Where to Evolve on the Path to Artificial Super Intelligence. *Transactions on Machine Learning Research*, 2026. URL https://mlanthology.org/tmlr/2026/gao2026tmlr-survey/.

Chenyang Gu, Yewen Pu, Bruce Yang, Xiaofan Li, and Huan Gao. Dspo: Stable and efficient policy optimization for agentic search and reasoning, 2026. URL https://arxiv.org/abs/2510.09255.

Prannay Hebbar, Yogendra Manawat, Samuel Verboomen, Alesia Ivanova, Selvam Palanimalai, Kunal Bhatia, and Vignesh Baskaran. Sia: Self improving ai with harness & weight updates, 2026. URL https://arxiv.org/abs/2605.27276.

Seth Karten, Joel Zhang, Tersoo Upaa Jr, Ruirong Feng, Wenzhe Li, Chengshuai Shi, Chi Jin, and Kiran Vodrahalli. Continual harness: Online adaptation for self-improving foundation agents, 2026. URL https://arxiv.org/abs/2605.09998.

Louis Kirsch, James Harrison, C. Freeman, Jascha Sohl-Dickstein, and Jürgen Schmidhuber. Towards general-purpose in-context learning agents. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023. URL https://openreview.net/forum?id=zDTqQVGgzH.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Stenberg Hansen, Angelos Filos, Ethan Brooks, maxime gazeau, Himanshu Sahni, Satinder Singh, and Volodymyr Mnih. In-context reinforcement learning with algorithm distillation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=hy0a5MMPUv.

Jonathan Lee, Annie Xie, Aldo Pacchiano, Yash Chandak, Chelsea Finn, Ofir Nachum, and Emma Brunskill. Supervised pretraining can learn in-context reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=dCYBAGQXLo.

Yibo Li, Zijie Lin, Ailin Deng, Xuan Zhang, Yufei He, Shuo Ji, Tri Cao, and Bryan Hooi. Just-in-time reinforcement learning: Continual learning in LLM agents without gradient updates. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=pLvye0zHUC.

Jiaqing Liang, Jinyi Han, Weijia Li, Xinyi Wang, Zhoujia Zhang, Zishang Jiang, Ying Liao, Tingyun Li, Ying Huang, Hao Shen, Hanyu Wu, Fang Guo, Keyi Wang, Zhonghua Hong, Zhiyu Lu, Lipeng Ma, Sihang Jiang, and Yanghua Xiao. Genericagent: A token-efficient self-evolving llm agent via contextual information density maximization (v1.0), 2026. URL https://arxiv.org/abs/2604.17091.

Minhua Lin, Juncheng Wu, Zijun Wang, Zhan Shi, Yisi Sang, Bing He, Zewen Liu, Tianxin Wei, Zongyu Wu, Zhiwei Zhang, Dakuo Wang, Xiang Zhang, Benoit Dumoulin, Cihang Xie, Yuyin Zhou, Suhang Wang, and Hanqing Lu. Harness updating is not harness benefit: Disentangling evolution capabilities in self-evolving llm agents, 2026. URL https://arxiv.org/abs/2605.30621.

Hongyi Liu, Haoyan Yang, Tao Jiang, Bo Tang, Feiyu Xiong, Yuyu Luo, and Zhiyu Li. Skillsvote: Lifecycle governance of agent skills from collection, recommendation to evolution, 2026a. URL https://arxiv.org/abs/2605.18401.

Xuanzhang Liu, Jianglun Feng, Zhuoran Zhuang, Junzhe Zhao, Maofei Que, Jieting Li, Dianlei Wang, Hao Tong, Ye Chen, and Pan Li. Coda: A context-decoupled hierarchical agent with reinforcement learning. In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining*, pp. 436–446. ACM, February 2026b. doi: 10.1145/3773966.3777986. URL http://dx.doi.org/10.1145/3773966.3777986.

Zhixuan Liu, Zhanhui Zhou, Yuanfu Wang, Chao Yang, and Yu Qiao. Inference-time language model alignment via integrated value guidance. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4181–4195, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.242. URL https://aclanthology.org/2024.findings-emnlp.242/.

Xinyu Lu, Kaiqi Zhang, Jinglin Yang, Boxi Cao, Yaojie Lu, Hongyu Lin, Min He, Xianpei Han, and Le Sun. P$\hat{2}$o: Joint policy and prompt optimization, 2026. URL https://arxiv.org/abs/2603.21877.

Yuchen Ma, Yue Huang, Han Bao, Haomin Zhuang, Swadheen Shukla, Michel Galley, Xiangliang Zhang, and Stefan Feuerriegel. Skillgen: Verified inference-time agent skill synthesis, 2026. URL https://arxiv.org/abs/2605.10999.

Amir Moeini, Jiuqi Wang, Jacob Beck, Ethan Blaser, Shimon Whiteson, Rohan Chandra, and Shangtong Zhang. A survey of in-context reinforcement learning, 2025. URL https://arxiv.org/abs/2502.07978.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. Reasoningbank: Scaling agent self-evolving with reasoning memory. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jL7fwchScm.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon llm agents, 2026. URL https://arxiv.org/abs/2603.08754.

Francesco Tonolini, Nikolaos Aletras, Jordan Massiah, and Gabriella Kazai. Bayesian prompt ensembles: Model uncertainty estimation for black-box large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12229–12272, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.728. URL https://aclanthology.org/2024.findings-acl.728/.

Prince Zizhuang Wang and Shuli Jiang. Slea-rl: Step-level experience augmented reinforcement learning for multi-turn agentic training, 2026. URL https://arxiv.org/abs/2603.18079.

Ruonan Wang, Runxi Wang, Yuwen Shen, Chengfeng Wu, Qinglin Zhou, and Rohitash Chandra. Evaluation of llms for mathematical problem solving. *Next Research*, 9:101705, 2026. ISSN 3050-4759. doi: https://doi.org/10.1016/j.nexres.2026.101705. URL https://www.sciencedirect.com/science/article/pii/S305047592600401X.

Siyuan Wang, Zhuohan Long, Zhihao Fan, Xuanjing Huang, and Zhongyu Wei. Benchmark self-evolving: A multi-agent framework for dynamic LLM evaluation. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert (eds.), *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 3310–3328, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.223/.

Bin Wu, Edgar Meij, and Emine Yilmaz. A joint optimization framework for enhancing efficiency of tool utilization in LLM agents. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 22361–22373, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.1149. URL https://aclanthology.org/2025.findings-acl.1149/.

Rong Wu, Xiaoman Wang, Jianbiao Mei, Pinlong Cai, Daocheng Fu, Cheng Yang, Licheng Wen, Xuemeng Yang, Yufan Shen, Yuxin Wang, and Botian Shi. Evolver: Self-evolving llm agents through an experience-driven lifecycle, 2026a. URL https://arxiv.org/abs/2510.16079.

Wenxun Wu, Yuanyang Li, Guhan Chen, Linyue Wang, and Hongyang Chen. Tool-augmented policy optimization: Synergizing reasoning and adaptive tool use with reinforcement learning, 2026b. URL https://arxiv.org/abs/2510.07038.

Xiaojun Wu, Cehao Yang, Honghao Liu, Xueyuan Lin, Wenjie Zhang, Zhichao Shi, Xuhui Jiang, Chengjin Xu, Jia Li, and Jian Guo. Bayesian-agent: Posterior-guided skill evolution across llm agent harnesses, 2026c. URL https://arxiv.org/abs/2606.08348.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. Skillrl: Evolving agents via recursive skill-augmented reinforcement learning, 2026. URL https://arxiv.org/abs/2602.08234.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RdJVFCHjUMI.

Yifan Yang, Ziyang Gong, Weiquan Huang, Qihao Yang, Ziwei Zhou, Zisu Huang, Yan Li, Xuemei Gao, Qi Dai, Bei Liu, Kai Qiu, Yuqing Yang, Dongdong Chen, Xue Yang, and Chong Luo. Skillopt: Executive strategy for self-evolving agent skills, 2026. URL https://arxiv.org/abs/2605.23904.

Tianrun Yu, yuxiao Yang, Zhaoyang Wang, Kaixiang Zhao, Porter Jenkins, Xuchao Zhang, Chetan Bansal, Huaxiu Yao, and Weitong Zhang. Provable and practical in-context policy optimization for self-improvement. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=TAthdtPe7k.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=92gvk82DE-.

Zefang Zong, Dingwei Chen, Yang Li, Qi Yi, Bo Zhou, Chengming Li, BO Qian, Peng Chen, and Jie Jiang. AT²PO: Agentic turn-based policy optimization via tree search. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24120–24143, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1106. URL https://aclanthology.org/2026.acl-long.1106/.

A THEORETICAL DERIVATION

In this section, we provide a detailed and rigorous proof of the theory underlying the methodology, thereby establishing the theoretical feasibility of our method.

A.1 PROOF FOR MARGINAL DISTRIBUTION OF θ

Following the Gaussian mixture formulation of the marginal distribution of θ in Equation 4, its overall mean and variance are given by (according to the **Law of Total Expectation and Variance**)):

$$\begin{aligned}
\mathbb{E}(\theta) &= \sum_{i=1}^{m} \omega_i(\mu + \delta_i) = \mu + \sum_{i=1}^{m} \omega_i \delta_i = u, \\
\text{Var}(\theta) &= \mathbb{E}[\text{Var}(\theta|\phi)] + \text{Var}[\mathbb{E}(\theta|\phi)] = \sum_{i=1}^{m} \omega_i \sigma^2 + \sum_{i=1}^{m} \omega_i(\mu + \delta_i - \mathbb{E}(\theta))^2 \\
&= \sigma^2 + \sum_{i=1}^{m} \omega_i \delta_i^2 = \sigma^2[1 + \sum_{i=1}^{m} \omega_i (\frac{\delta_i}{\sigma})^2].
\end{aligned} \tag{10}$$

To simplify the subsequent Bayesian update, we would like to approximate this mixture by a single Gaussian, i.e., $p(\theta) \approx \mathcal{N}(\mu, \sigma^2)$. This approximation is justified by considering two extreme regimes:

Case 1: In the maximum-entropy regime, the weights are uniform: $\omega_i = \frac{1}{m}$. There is no prior preference among the context-constrained types χ_i. If the data provide no discriminative information among them, the posterior estimates of the component means shrink toward a common value, so that $\delta_i \to 0$.

Case 2: In the minimum-entropy regime, one context-constrained type χ_i dominates: $\omega_i \to 1$ and $\omega_j \to 0 (\forall j \neq i)$. The mixture then degenerates to a single Gaussian.

Nevertheless, this approximation is not merely a convenience; it is what makes the subsequent Bayesian update tractable. If the mixture structure were retained in full, the posterior over θ would remain a mixture whose number of components grows with each update, and the associated normalization, marginalization, and evidence computations would quickly become intractable. This will preserve the conjugacy of the θ related distributions, allows the update equations to be expressed in closed form.

A.2 PROOF FOR AGENT VARIABLE DESIGN

The Vanilla Advantage: Consider a Markov process $(s_1, a_1, ..., s_t, a_t, ...)$, we define the advantage function at step t as $A(s_t, a_t) = Q(s_t, a_t) - V(s_t)$. Suppose that from this step onward, the average per-step reward is r_{avg}, the $V(s_t)$ can be expressed as:

$$\begin{aligned}
V(s_t) &= r_{avg} + \gamma r_{avg} + ... + \gamma^{T-t} r_{avg} \\
&= r_{avg}(\frac{1 - \gamma^{T-t+1}}{1 - \gamma}),
\end{aligned} \tag{11}$$

where γ denotes the discount factor.

Adopting a remaining-return perspective, let θ denote the global expected total return of the task, and $\sum_{i=1}^{t-1} r_i$ be the cumulative reward already collected in the first $t-1$ steps. At step t, the future remaining value required to fulfill the global objective—equivalently, the state value $V(s_t)$—must then satisfy:

$$V(s_t) = \theta - \sum_{i=1}^{t-1} r_i. \tag{12}$$

In accordance with the definition of the temporal-difference (TD) error, we substitute Equations 11 and 12 into the TD target (we assume herein that r_{avg} remains approximately invariant across

consecutive steps), which leads to:

$$\begin{aligned} A(s_t, a_t) &= r_t + \gamma V(s_{t+1}) - V(s_t) \\ &= r_t + \gamma r_{avg}(\frac{1-\gamma^{T-t}}{1-\gamma}) - r_{avg}(\frac{1-\gamma^{T-t+1}}{1-\gamma}) \\ &= r_t - r_{avg} \\ &= r_t - (\theta - \sum_{i=1}^{t-1} r_i)\frac{1-\gamma}{1-\gamma^{T-t+1}}. \end{aligned} \tag{13}$$

In the limit where the discount factor $\gamma \to 1$, we invoke L'Hôpital's rule to further simplify Equation 13 as:

$$\lim_{\gamma \to 1} A(s_t, a_t) = r_t - \frac{\theta - \sum_{i=1}^{t-1} r_i}{T-t+1}. \tag{14}$$

Effective Advantage: Using the vanilla advantage estimator above yields an immediate consequence, that the dependence on θ cancels out across different action probabilities, because:

$$\begin{aligned} p^*(a_t|s_t) &\propto p(a_t|s_t)\exp\left(A(s_t, a_t)\right) \\ &= \frac{p(a_t|s_t)\exp r_t / \exp \frac{\theta - \sum_{i=1}^{t-1} r_i}{T-t+1}}{\sum_{a_t} p(a_t|s_t)\exp r_t / \exp \frac{\theta - \sum_{i=1}^{t-1} r_i}{T-t+1}} = \frac{p(a_t|s_t)\exp r_t}{\sum_{a_t} p(a_t|s_t)\exp r_t}. \end{aligned} \tag{15}$$

This implies that the intended role of θ in governing the agent model's generation process (Section 3.2) is fundamentally undermined: the entire control procedure is rendered completely independent of θ. Accordingly, we introduce a dynamic temperature coefficient into the θ-related expression, so that the θ-dependent part in the bias term cannot be reduced away and can genuinely guide the generation process. We refer to the advantage modified in this manner as the *effective advantage*:

$$A_{\text{eff}}(s_t, a_t) = r_t - (1 + \kappa \log p(a_t|s_t))\frac{\theta - \sum_{i=1}^{t-1} r_i}{T-t+1} = r_t - (1 + \kappa \log p(a_t|s_t))\Delta_t, \tag{16}$$

where $\Delta_t = \frac{\theta - \sum_{i=1}^{t-1} r_i}{T-t+1}$ for simplification.

The new dynamic temperature coefficient, i.e., $1 + \kappa \log p(a_t|s_t)$, is a function of the agent model's original action probabilities. This design explicitly accounts for the coupling between expected-return (θ) allocation and action exploration:

Case 1: When the model is highly confident about a certain action, i.e., $p(a_t, s_t) \to 1$ (meaning $\kappa \log p \to 0$), no exploration bonus is granted, and the advantage is determined solely by the actual reward r_t. In this way, the agent model's strong prior judgment is not sacrificed.

Case 2: When Δ_t is relatively large, it indicates that the current preference for high-probability actions fails to effectively facilitate the allocation of returns. In this case, the $\kappa \log p$ term imposes a stronger bias penalty on high-probability actions, thereby encouraging exploration over low-probability actions.

Case 3: If the expected return θ is nearly achieved, i.e., a small Δ_t, the advantage value will reduce the exploration incentive, allowing the model to safely exploit the current high-probability actions.

It should be noted that if we further simplify Equation 5, we obtain: $p^*(a_t|s_t) \propto p(a_t|s_t)\exp\left(r_t - (1 + \kappa \log p(a_t|s_t))\Delta_t\right) = p(a_t|s_t)^{1-\kappa\Delta_t}\exp\left(r_t - \Delta_t\right)$. Since the probability bias we introduce is defined at the **action level** rather than the token level, it is implemented by **adjusting the temperature during each agent model generation to sample multiple actions**. The original probability of each action is then corrected according to Equation 5, and the final normalization is likewise performed over these actions. Consequently, probabilities do not become excessively small, and the issue of probability explosion does not arise.

A.3 Proof for Variables Update

In this section, we present the update derivations for the context variable and the agent model variable. The entire derivation starts from the conditional distribution of θ and the full likelihood distribution of v_τ given in Equation 4 and Equation 6.

The Full Likelihood Estimation: Following Algorithm 1, at the $k-$th iteration we obtain a set of trajectories $\{\tau_k\}$ along with their corresponding returns $\{v_{\tau_k}\}$ (N elements). The mean and variance of $p(v_\tau|\phi=\chi_i,\theta)$ are then estimated from the sample mean and variance of these trajectories, which gives:

$$\theta_k+\delta_i^{(k)}=\hat{v_{\tau_k}}=\frac{\sum_{n=1}^{N}R_n}{N},\ \sigma^2_{\tau_k}=\frac{\sum_{n=1}^{N}(R_n-\hat{v_{\tau_k}})^2}{N-1}, \tag{17}$$

where R_n is the cumulative return associated with the $n-$th trajectory in the set. $\hat{v_{\tau_k}}$ represents the average return among the trajectories.

Lemma 1 *The partial likelihood of v_{τ_k} given $\phi_k=\chi_i$ follows a Gaussian distribution $\mathcal{N}(\mu^{(k)}+2\delta_i^{(k)},(\sigma^{(k)})^2+\sigma^2_{\tau_k})$.*

Proof: According to Bayes' theorem, the partial likelihood can be expressed in the following integral form, where we set $A=\frac{1}{(\sigma^{(k)})^2}$ and $B=\frac{1}{\sigma^2_{\tau_k}}$ for simplification:

$$\begin{aligned}
p(v_{\tau_k}|\phi_k)&=\int p(v_{\tau_k},\theta_k|\phi_k)d\theta_k=\int\frac{p(v_{\tau_k}|\theta_k,\phi_k)p(\theta_k,\phi_k)}{p(\phi_k)}d\theta_k=\int p(v_{\tau_k}|\theta_k,\phi_k)p(\theta_k|\phi_k)d\theta_k\\
&=\frac{\sqrt{AB}}{2\pi}\int_{-\infty}^{+\infty}\exp\left(-\frac{(v_{\tau_k}-(\theta_k+\delta_i^{(k)})^2}{2\sigma^2_{\tau_k}}\right)*\exp\left(-\frac{(\theta_k-(\mu^{(k)})+\delta_i^{(k)})^2}{2(\sigma^{(k)})^2}\right)d\theta_k\\
&=\frac{\sqrt{AB}}{2\pi}\int_{-\infty}^{+\infty}\exp(-\frac{1}{2}((A+B)\theta_k^2-2(A(\mu^{(k)}+\delta_i^{(k)})+B(v_{\tau_k}-\delta_i^{(k)}))\theta_k\\
&+(A(\mu^{(k)}+\delta_i^k)^2+B(v_{\tau_k}-\delta_i^{(k)})^2)))d\theta_k\\
&=\frac{\sqrt{AB}}{2\pi}\exp\left(-\frac{A+B}{2}\frac{AB((v_{\tau_k}-\delta_i^{(k)})-(\mu^{(k)}+\delta_i^{(k)}))^2}{(A+B)^2}\right)\\
&*\int_{-\infty}^{+\infty}\exp\left(-\frac{(\theta_k-\frac{A(\mu^k+\delta_i^{(k)})+B(v_{\tau_k}-\delta_i^{(k)})}{A+B})^2}{2\frac{1}{A+B}}\right)d\theta_k\\
&=\frac{\sqrt{AB}}{2\pi}\exp\left(-\frac{AB(v_{\tau_k}-(\mu^{(k)}+2\delta_i^{(k)}))^2}{2(A+B)}\right)\sqrt{2\pi\frac{1}{A+B}}\\
&=\frac{1}{\sqrt{2\pi(\frac{1}{A}+\frac{1}{B})}}\exp\left(-\frac{(v_{\tau_k}-(\mu^{(k)}+2\delta_i^{(k)}))^2}{2(\frac{1}{A}+\frac{1}{B})}\right)\\
&=\frac{1}{\sqrt{2\pi((\sigma^{(k)})^2+\sigma^2_{\tau_k})}}\exp\left(-\frac{(v_{\tau_k}-(\mu^{(k)}+2\delta_i^{(k)}))^2}{2((\sigma^{(k)})^2+\sigma^2_{\tau_k})}\right)\\
&\sim\mathcal{N}(\mu^{(k)}+2\delta_i^{(k)},(\sigma^{(k)})^2+\sigma^2_{\tau_k}).
\end{aligned}\tag{18}$$

Accordingly, the mean and variance of the partial distribution conditioned on ϕ_k are given by $\mu^{(k)}+2\delta_i^{(k)}$ and $(\sigma^{(k)})^2+\sigma^2_{\tau_k}$, respectively.

Lemma 2 *The partial likelihood of v_{τ_k} given θ_k follows a Gaussian distribution $\mathcal{N}(\theta_k,\sigma^2_{\tau_k})$.*

Proof: Consistent with Section A.1, we similarly regard the partial likelihood under θ_k as an averaged state of the full likelihood (which is additionally conditioned on ϕ_k). Accordingly, we remove

the deviation $\delta_i^{(k)}$ induced by $\phi_k = \chi_i$ from the mean of this partial likelihood, while keeping the variance unchanged. The detailed derivation proceeds analogously to Equation 10.

Lemma 3 *The distribution for the context variable, i.e., $p(\phi_k)$, is updated in accordance with the following posterior, where i refers to the index of the selected context constrained type, i.e., χ_i, for the trajectory generated at the current iteration.*

Proof: By Bayes' theorem and the Gaussian distribution mentioned in Lemma 1, the posterior of ϕ conditioned on the current trajectory can be written as:

$$\begin{aligned} p(\phi_k = \chi_i | v_{\tau_k}) &= \frac{p(v_{\tau_k} | \phi_k = \chi_i) p(\phi_k = \chi_i)}{p(v_{\tau_k})} \\ &\propto p(\phi_k = \chi_i) \frac{1}{\sqrt{2\pi((\sigma^{(k)})^2 + \sigma_{\tau_k}^2)}} \exp\left(-\frac{(v_{\tau_k} - (\mu^{(k)} + 2\delta_i^{(k)}))^2}{2((\sigma^{(k)})^2 + \sigma_{\tau_k}^2)}\right) \end{aligned} \tag{19}$$

This implies that, for the currently selected type χ_i, we multiply its current probability by a bias term. This bias term depends on the parameters governing trajectory generation—such as $\mu^{(k)}$ and $\sigma^{(k)}$—as well as on the parameters that treat the trajectory as evidence, like v_{τ_k}, $\delta_i^{(k)}$ and σ_{τ_k}. For the unselected context constraint types $\chi_j (\forall j \neq i)$, we cannot simply assume their probabilities remain unchanged; instead, we must design a reasonable bias term that accounts for trajectory quality. The final design takes the form of Equation 7, with the proof provided in Proof A.4.

Lemma 4 *The posterior of θ_k conditioned on trajectory τ_k follows a Gaussian distribution $\mathcal{N}(\frac{A\mu^{(k)} + B v_{\tau_k}}{A+B}, \frac{1}{A+B})$, with $A = \frac{1}{(\sigma^{(k)})^2}$ and $B = \frac{1}{\sigma_{\tau_k}^2}$.*

Proof: In view of Equation 10 and Lemma 2, both the prior $p(\theta_k)$ and the partial likelihood $p(v_{\tau_k} | \theta_k)$ are Gaussian. We accordingly specify them as:

$$p(\theta_k) \propto \exp\left(-\frac{(\theta_k - \mu^{(k)})^2}{2(\sigma^{(k)})^2}\right), \;\; p(v_{\tau_k} | \theta_k) \propto \exp\left(-\frac{(v_{\tau_k} - \theta_k)^2}{2\sigma_{\tau_k}^2}\right). \tag{20}$$

Then, by virtue of Bayes' theorem, we obtain:

$$\begin{aligned} p(\theta_k | v_{\tau_k}) &= \frac{p(v_{\tau_k} | \theta_k) p(\theta_k)}{p(v_{\tau_k})} \propto p(v_{\tau_k} | \theta_k) p(\theta_k) \propto \exp\left(-\left(\frac{(v_{\tau_k} - \theta_k)^2}{2\sigma_{\tau_k}^2} + \frac{(\theta_k - \mu^{(k)})^2}{2(\sigma^{(k)})^2}\right)\right) \\ &= \exp\left(-\frac{1}{2}(A(\theta_k - \mu^{(k)})^2 + B(v_{\tau_k} - \theta_k)^2)\right) \\ &= \exp\left(-\frac{A+B}{2}(\theta_k^2 - 2\frac{A\mu^{(k)} + B v_{\tau_k}}{A+B}\theta_k + \frac{A(\mu^{(k)})^2 + B v_{\tau_k}^2}{A+B})\right) \\ &= \exp\left(-\frac{A+B}{2}(\theta_k - \frac{A\mu^{(k)} + B v_{\tau_k}}{A+B})^2 - \frac{AB}{A+B}(v_{\tau_k} - \mu^{(k)})^2\right) \\ &\propto \exp\left(-\frac{(\theta_k - \frac{A\mu^{(k)} + B v_{\tau_k}}{A+B})^2}{2\frac{1}{A+B}}\right). \end{aligned} \tag{21}$$

Therefore, the posterior distribution of θ remains a Gaussian distribution, with its mean given by $\frac{A\mu^{(k)} + B v_{\tau_k}}{A+B}$ and variance by $\frac{1}{A+B}$. With the trajectory-based estimate of $\sigma_{\tau_k}^2$ from Equation 17 substituted, $p(\theta_k | v_{\tau_k})$ can be readily solved, and the belief state of θ, i.e., $p(\theta_k)$, is subsequently updated.

A.4 Proof of Theorem

Theorem 1 Proof: By Lemma 2, the expected trajectory return at iteration k, i.e., v_{τ_k}, conditioned on θ_k alone is θ_k, which represents the average return over the context-constrained type, that is, over ϕ_k. When additionally conditioned on $\phi_k = \chi_i$, in Equation 6, the expected trajectory return

instead becomes $\theta_k + \delta_i^{(k)}$. The bias attributable to χ_i at iteration k can therefore be quantified as $\delta_i^{(k)}$, representing the extent of its deviation from this average.

Referring to Equation 7 and the Proof in Lemma 3, under the current trajectory τ_k, the probability corresponding to the selected value of ϕ (i.e., $\omega_i^{(k)} = p(\phi_k = \chi_i)$, the context constrained type responsible for generating τ_k) is updated to $\omega_i^{(k+1)} \propto \frac{\omega_i^{(k)}}{\sqrt{2\pi((\sigma^{(k)})^2+\sigma_{\tau_k}^2)}} \exp\left(-\frac{(v_{\tau_k}-(\mu^{(k)}+2\delta_i^{(k)}))^2}{2((\sigma^{(k)})^2+\sigma_{\tau_k}^2)}\right)$, whereas the probabilities of the remaining $\chi_j (j \neq i)$ are updated to $\omega_j^{(k+1)} \propto \frac{\omega_j^{(k)}}{\sqrt{2\pi((\sigma^{(k)})^2+\sigma_{\tau_k}^2)}} \exp\left(-\frac{(v_{\tau_k}-\mu^{(k)})^2}{2((\sigma^{(k)})^2+\sigma_{\tau_k}^2)}\right)$. Therefore, it can be derived that:

$$\begin{aligned}
\frac{\omega_i^{(k+1)}}{\omega_j^{(k+1)}} &= \frac{\omega_i^{(k)}}{\omega_j^{(k)}} \exp\left(-\frac{(v_{\tau_K} - \mu^{(k)} - 2\delta_i^{(k)})^2 - (v_{\tau_K} - \mu^{(k)})^2}{2((\sigma^{(k)})^2 + \sigma_{\tau_k}^2)}\right) \\
&= \frac{\omega_i^{(k)}}{\omega_j^{(k)}} \exp\left(-\frac{(2\delta_i^{(k)})^2 - 2(v_{\tau_k} - \mu^{(k)})(2\delta_i^{(k)})}{2((\sigma^{(k)})^2 + \sigma_{\tau_k}^2)}\right) \\
&= \frac{\omega_i^{(k)}}{\omega_j^{(k)}} \exp\left(\frac{2\delta_i^{(k)}(v_{\tau_k} - \mu^{(k)} - \delta_i^{(k)})}{(\sigma^{(k)})^2 + \sigma_{\tau_k}^2}\right). \qquad (22)
\end{aligned}$$

Furthermore, by summing over the index $j (j \neq i)$, we obtain:

$$\begin{aligned}
\frac{\omega_i^{(k+1)}}{1 - \omega_i^{(k+1)}} = \frac{\omega_i^{(k+1)}}{\sum_{j \neq i} \omega_j^{(k+1)}} &= \exp\left(\frac{2\delta_i^{(k)}(v_{\tau_k} - \mu^{(k)} - \delta_i^{(k)})}{(\sigma^{(k)})^2 + \sigma_{\tau_k}^2}\right) \frac{\omega_i^{(k)}}{\sum_{j \neq i} \omega_j^{(k)}} \\
&= \exp\left(\frac{2\delta_i^{(k)}(v_{\tau_k} - \mu^{(k)} - \delta_i^{(k)})}{(\sigma^{(k)})^2 + \sigma_{\tau_k}^2}\right) \frac{\omega_i^{(k)}}{1 - \omega_i^{(k)}}. \qquad (23)
\end{aligned}$$

By Theorem 2, we have $v_{\tau_k} - \mu^{(k)} = \mathbb{E}[v_{\tau_k} - \theta_k] > 0$, as a consequence, if the expected return of the trajectory obtained under $\phi_k = \chi_i$ is below the average, i.e., the bias $\delta_i^{(k)} < 0$, it satisfies that:

$$\frac{\omega_i^{(k+1)}}{1 - \omega_i^{(k+1)}} < \frac{\omega_i^{(k)}}{1 - \omega_i^{(k)}} \Rightarrow \omega_i^{(k+1)} < \omega_i^{(k)}, \qquad (24)$$

which penalizes the χ_i whose return is below the mean, thereby reducing its relative probability.

Theorem 2 Proof: As established in Lemma 4, the belief update for θ_k conforms to a Gaussian distribution $\mathcal{N}(\frac{A\mu^{(k)} + Bv_{\tau_k}}{A+B}, \frac{1}{A+B})$, with an initial variance of $(\sigma^{(k)})^2$ and a posterior variance of $\frac{1}{\frac{1}{(\sigma^{(k)})^2} + \frac{1}{\sigma_{\tau_k}^2}}$. Accordingly, the resulting change in variance induced by the update is given by:

$$(\sigma^{(k+1)})^2 - (\sigma^{(k)})^2 = \frac{1}{\frac{1}{(\sigma^{(k)})^2} + \frac{1}{\sigma_{\tau_k}^2}} - (\sigma^{(k)})^2 = \frac{1}{\frac{1}{(\sigma^{(k)})^2} + \frac{1}{\sigma_{\tau_k}^2}} - \frac{1}{\frac{1}{(\sigma^{(k)})^2}} < 0. \qquad (25)$$

This variance reduction indicates that, throughout the iterative updating process, the confidence level is steadily improved. Additionally, the expectation of θ_k after the update satisfies:

$$\mathbb{E}(\theta_{k+1}) - \mathbb{E}(\theta_k) = \frac{A\mu^{(k)} + Bv_{\tau_k}}{A+B} - \mu^{(k)} = \frac{B}{A+B}(v_{\tau_k} - \mu^{(k)}) = \frac{B}{A+B}\mathbb{E}[v_{\tau_k} - \theta_k], \qquad (26)$$

where both A and B are positive. This implies that when the actual trajectory return v_{τ_k} is greater than the expectation of θ_k, the estimate of θ_{k+1} (in the next round) will be improved.

For an arbitrary iteration, we drop the iteration index k, since all iterations follow the same procedure. Then at each step of the sampling process, we select the action with the largest biased log-probability. Accordingly, **for any sampled value of** θ , the actual trajectory return obtained

from the final sampling is given by: $v_\tau = \sum_{i=1}^{T} r_i^*$. Taking the difference between θ and v_τ yields:

$$\mathcal{G}_\theta = v_\tau - \theta = -(\theta - \sum_{i=1}^{T} r_i^*) = -(\theta - \sum_{i=1}^{T-1} r_i^* - r_T^*) = r_T^* - \Delta_T, \tag{27}$$

where $\Delta_T = \frac{\theta - \sum_{i=1}^{T-1} r_i^*}{T-T+1} = \theta - \sum_{i=1}^{T-1} r_i^*$ (as shown in Equation 16, $0 < \Delta_T \leq T$, since $\Delta_T \leq 0$ implies that θ has already been exhausted by step $T-1$, it necessarily follows that $v_\tau > \theta$). To ensure that τ is obtained from the maximum biased log-probability, for any action, the following must hold:

$$\begin{aligned} p^*(a_T^*) &= \frac{p(a_T^*) * \exp(r_T^* - (1 + \kappa \log p(a_T^*))\Delta_T)}{\sum_a p(a_T) * \exp(r_{T,a} - (1 + \kappa \log p(a_T))\Delta_T)} \\ &> \frac{p(a_T) * \exp(r_{T,a} - (1 + \kappa \log p(a_T))\Delta_T)}{\sum_a p(a_T) * \exp(r_{T,a} - (1 + \kappa \log p(a_T))\Delta_T)} = p^*(a_T). \end{aligned} \tag{28}$$

Furthermore, we obtain the following:

$$\log p(a_T^*) + r_T^* - (1 + \kappa \log p(a_T^*))\Delta_T > \log p(a_T) + r_{T,a} - (1 + \kappa \log p(a_T))\Delta_T. \tag{29}$$

Proceeding with further algebraic simplification of the above expression, we arrive at:

$$\begin{aligned} \mathcal{G}_\theta = r_T^* - \Delta_T &> (r_{T,a} - \Delta_T) + (1 - \kappa\Delta_T)(\log p(a_T) - \log p(a_T^*)) \\ &> (\kappa\Delta_T - 1)(\log p(a_T^*) - \log p(a_T)) - 1. \end{aligned} \tag{30}$$

This inequality suggests that for the condition $r_T^* - \Delta_T > 0$ hold, it suffices to choose $(\kappa\Delta_T - 1)(\log p(a_T^*) - \log p(a_T)) > 1$, which then allows us to solve for $\kappa > \frac{\frac{1}{\log p(a_T^*) - \log p(a_T)} + 1}{\Delta_T}$. This implies that when we set κ using the above constraint, the validity of Theorem 2 can be guaranteed.

Although we simplify the problem through the computation at the last step T, the choice of κ can be implemented at each step t. In this iterative manner, we can ensure that each step t receives a substantial immediate reward r_t, thereby preventing the emergence of a large residual margin in θ at later stages—that is, Δ_t is kept within a controllable range at every step. As for the selection of κ, we randomly sample a small subset from the data and compute the probabilities during the agent model's generation process for estimation.

Now, we show the condition at any time step t: At the initial step $t = 1$, it satisfies that: $\Delta_1 = \frac{\theta}{T} < 1$, because, when we design the sampling of θ, we try as much as possible to ensure that it does not exceed T (see Section D). For step $t+1$, we have the following recurrence relation:

$$\Delta_{t+1} = \frac{\Delta_t(T - t + 1) - r_t}{T - t} = \Delta_t + \frac{\Delta_t - r_t}{T - t} = \Delta_t - \frac{r_t - \Delta_t}{T - t}. \tag{31}$$

That is to say, if $r_t - \Delta_t > 0$, then we can ensure that: $\Delta_{t+1} < \Delta_t$. Similarly, with the strategy for which Equation 30 holds, we can solve this expression, and the conclusion is basically the same. This means that, at each step, if we adopt such a strategy, we can ensure that $\Delta_t < \Delta_{t-1} < ... < \Delta_1 < 1$; then at the final step T, we can still obtain $\Delta_T < 1$. This shows that the implementation of the strategy at each step ensures that Δ_t is decreasing throughout the whole process, so that ultimately $v_\tau - \theta > 0$.

Since for any sampled θ, $v_\tau - \theta > 0$ holds, it follows naturally that $\mathbb{E}[v_\tau - \theta] > 0$.

B Datasets and Baselines to Comparison

Dataset Extra Information: For DeepSearch task, the primary tool usage includes web search, document extraction, image OCR, calculator, and safe Python execution, covering various stages such as information retrieval, information processing, and execution. For τ^2-bench, the tools employed include find item ID, find user ID, modify pending information, calculator, get order information, and others, encompassing different steps of interactive transaction processes.

Baselines: We compare our method against several context optimization baselines:

• Context-Design: A static context construction method that leverages an LLM to design feasible context for each query, aiming to facilitate task completion through carefully crafted system prompts.

• Reflection Shinn et al. (2023): A mechanism where an LLM reflects on past trajectories and revises context content accordingly, treating reflection as a background process that refines long-term context based on recent experience.

• Reasoning-Bank Ouyang et al. (2026): A memory mechanism where agents distill generalizable reasoning strategies from past trajectories (both successes and failures), index them, and retrieve relevant strategies for new tasks to inject into the agent's context.

• Bayesian-Agent Wu et al. (2026c): A framework that treats the context modification process as a probabilistic variable, learning its distribution through large-scale offline trajectory data. It maintains a Bayesian registry where posterior beliefs update from verified evidence, enabling context rendering and rewrite decisions based on observed context features, runtime signals, and failure modes.

• SkillGen Ma et al. (2026): An in-context learning framework that extracts domain-oriented, action-centric skills from sampled trajectories via temporal-difference credit assignment. It constructs an action-centric domain knowledge graph and retrieves step-wise skills during inference to guide action generation, achieving significant gains in sequential decision-making tasks.

The joint optimization methods are (for both methods, we likewise use the reward in Equation 32 to guide the RL training process):

• Iterative-Alternating training: Following the same alternating optimization scheme as SIA Hebbar et al. (2026), where the context and the agent model are optimized sequentially in each round, we implement the entire training pipeline using the VeRL library.

• Full-Adaptation training: As in EvoTrainer Chen et al. (2026a), one component (context or agent model) is fully converged before the other is optimized.

C Context Modification and Reward Details

Context Modification: The context content is partitioned into six regions, each dedicated to a specific functional role—such as query construction, action generation, verification, and others. In task execution, these regions are concatenated into a unified prompt that the agent model is mandated to follow strictly. During the evolutionary process, a particular region, e.g., $\mathcal{C}_{\text{query}}$, is first selected according to the current distribution of ϕ; the agent model then revises the selected region, yielding the updated context content for that iteration.

1: Context Regions

$\mathcal{C}_{\text{base}}$
• Treat each question as an independent task, using only the current question and the rule set; do not inherit facts, answers, or trajectories from other questions.
• Clearly identify the target entity, temporal range, geographic scope, and judgment criteria before initiating reasoning or retrieval.

$\mathcal{C}_{\text{query}}$
• Decompose the question into independent constraints, and for each constraint, annotate the source type most likely to provide supporting evidence; then select the most discriminative constraint as the first retrieval anchor.
• Use a concise query to build an initial candidate set, then progressively narrow down around the remaining unsatisfied constraints; switch languages, entity aliases, or source domains as needed, and avoid packing all constraints into a single query.

• Once a retrieval result provides a specific URL that potentially contains the target fact, pass that URL to the reading tool to retrieve the full content, rather than reusing the URL as a keyword for repeated searches.
• Within the same task, record all queries used, candidates, and evidence gaps; subsequent queries must introduce new sources, candidates, or constraint evidence, and repeated substantively equivalent queries are prohibited.
• Prioritize retrieval from primary or authoritative sources; when primary sources are unavailable, high-quality secondary sources may be used to form candidates, but their evidentiary level must be explicitly stated.

$\mathcal{C}_{\text{action}}$
• Before each tool call, clearly articulate the evidence gap it is intended to fill, and select the appropriate tool from search, direct webpage access, PDF/document parsing, table extraction, and video subtitle reading.
• Search is used for source discovery; opening is used for source reading; once a specific page or file has been located, prioritize direct reading and do not substitute search snippets for full content.
• When reading long documents or tables, locate relevant paragraphs or rows around the target entity, year, field name, and unit; do not treat page titles, navigation text, or irrelevant excerpts as answers.
• If the current tool type yields no new valid evidence for two consecutive attempts, switch tools or source types rather than merely rewriting keywords.
• Record evidence supporting and contradicting each candidate, along with the corresponding URLs, to avoid confirming initial guesses along a single path.
(+ for DeepSearch Task
• Distinguish between the publication/version year of a report and the reference year of the data; when the question specifies a data year, locate records corresponding to that reference year, and for long-document parsing, input discriminative keywords or non-overlapping page ranges.)

$\mathcal{C}_{\text{verify}}$
• Construct a constraint–evidence matrix for each candidate answer, and verify point by point that the entity, number, unit, date, geographic scope, and causal direction are genuinely covered by the cited content.
• Differentiate among direct evidence, indirect clues, and model inference; when sources conflict, compare authority, publication date, and data specifications, and do not use majority voting as a substitute for evidential judgment.
• When precise numerical values or names are involved, verify indicator definitions, entity ownership, and homonymous entities; for computational questions, recheck the input values, formula, and time span.
(+ for DeepSearch Task
• When displayed values require rounding but computed results require decimal precision, preserve the original decimal precision from the source throughout the calculation, and only take the integer part when displaying the specified fields in the final output.
• If core years, entities, values, or computational inputs remain unverified or mutually conflicting, do not force the candidate to be labeled as a verified result; retain the candidate only in alternatives, and do not write unknown values as zero.)
(+ for τ^2-Bench Task
• When evidence is incomplete, candidates shall still be ranked and the most probable current answer selected, while retaining uncertainty as internal confidence; page titles, navigation text, or random search snippets shall not be used as substitutes for candidate answers.)

$\mathcal{C}_{\text{format}}$
• In the final output, provide only the answer requested by the question; if only a name, title, number, or code is asked, return only that corresponding name, title, number, or code.

• Do not output search processes, self-analyses, rule explanations, candidate comparisons, or refusal responses; for multi-field answers, preserve necessary labels such as year, entity, and indicator in the order specified by the question, so that each value can be uniquely mapped to its corresponding field, and do not degenerate into an unlabeled list of bare values.
(+ for DeepSearch Task
• When the same question requires both rounded display and decimal computation, the displayed fields shall be rounded as specified, while the computed fields shall be calculated using the original high-precision inputs and retain the specified number of decimal places.)

$\mathcal{C}_{\text{stop}}$
• When all core constraints are sufficiently evidenced, or when further retrieval can no longer alter the ranking of candidates, cease research and proceed to answer.
• If two consecutive research actions yield no new candidates, sources, or constraint evidence, a tool or source type switch is mandatory; if no gain is achieved after one substantive switch, the process may enter the answer stage early.
• Each single response may execute at most the configured number of research actions; opening a source also counts against the research budget, so this action should not be used to circumvent the limit.
(+ for DeepSearch Task
• When the research action limit is reached, output only complete candidates that are directly supported by existing evidence; if core constraints remain missing, preserve an explicit failure state for the next iteration of learning, and do not fabricate or use conflicting candidates as fillers.)
(+ for τ^2-Bench Task
• Insufficient tool capability shall only affect confidence levels and shall not constitute grounds for refusing to answer; when the research action limit is reached, the most likely answer must be output using the available evidence and candidates.)

Reward : The reward rubric consists of three major dimensions, each evaluated through a tiered, fine-grained analysis of the current action:

Context Rules Compliance:*This dimension includes two boolean indicators.*

• Format validation (r_f): whether the action produces a valid output in strict accordance with the prescribed format.

• Other context compliance (r_c): whether the action is generated reasonably and adheres to the context rules.

Coverage of Evidence:*This dimension focuses on the adequacy of evidence retrieved or provided by the action.*

• Evidence coverage (r_e): measures the extent to which the evidence goals are covered, primarily targeting the query.

Plan Completion:*This dimension evaluates the action's contribution to task progression and its feasibility.*

• Supported (r_s): assesses whether the action can support the resolution of the query.

• Blocking (r_b): evaluates whether the current plan execution involves unreasonable, infeasible, or blocking constraints. If blocking, set 0; otherwise, 1.

Computational Verification:*This dimension assesses the validity and executability of the action with respect to the query constraints.*

• Calculation required (r_o): a boolean indicator comparing the action against the query to verify whether the constraints are obeyed.

• Executable (r_a): determines whether the current action is executable in practice.

• Action refusal (r_v): assesses whether the action is valid and substantive (i.e., not an invalid response such as a refusal or empty output).

Based on the above fine-grained evaluation dimensions of the action, the final reward is designed as a weighted combination of these dimensions, normalized to the $[0, 1]$ interval:

$$r = (0.10*r_f+0.03*r_c)+(0.15*r_e)+(0.24*r_s+0.10*r_b)+(0.13*r_o+0.10*r_a+0.15*r_v). \quad (32)$$

Score: *The Score is computed by measuring the relationship between the executed answer and the gold answer at three levels. First, semantic similarity (g_s) is obtained by encoding both answers via an embedding model and computing their inner product. Second, the Ratcliff-Obershelp similarity (g_r) is calculated based on the longest common subsequence between them. Third, a token-level F1 score (g_f) is computed by tokenizing both answers and calculating the harmonic mean of precision and recall at the token level.*

$$g = 0.7 * g_s + 0.2 * g_r + 0.1 * g_f. \quad (33)$$

D THE EXECUTION PROCEDURE

Our co-evolution framework consists of two main components in each iteration. The first is trajectory generation: at each action generation step t, we compute a bias term for the action probability based on our proposed advantage estimation, which is then applied to adjust the original agent model's action distribution, enabling exploration toward more optimal actions. The second is variable update: based on the rewards collected from multiple trajectories, we first estimate the trajectory-specific bias parameters, i.e., δ_i and σ_τ, and then update the distributions of the variables according to our theoretical formulation.

To achieve better parameterization of the variables' distributions, we depart from random initialization, e.g., using a discrete uniform distribution for ϕ. Instead, we first execute the task using the original context and the agent model, and use the return of the resulting trajectory as the initial mean of θ, with the variance set to $\sigma = \min\{\frac{\mathbb{E}(\theta)}{3}, \frac{T-\mathbb{E}(\theta)}{3}\}$ (this value is derived from the 3σ-rule of the Gaussian distribution, which prevents the sampled parameter θ from falling outside the feasible range, i.e., $(0, T)$. See the Proof below). We also leverage the agent model's reflections on the trajectory to obtain an estimate of the probability associated with each constrained component χ_i, thereby providing a principled initial distribution for ϕ.

Proof for σ initialization: The variable θ follows a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ (Equation 4). Since θ serves as an estimate of the trajectory return, its value is constrained to the interval $(0, T)$), where each step yields a reward in $[0, 1]$ and the trajectory consists of T steps in total. To ensure that the sampled θ falls within this feasible range with high probability, we leverage the 3σ-rule and constrain the Gaussian distribution such that its 3σ interval lies entirely within $(0, T)$. That is:

$$\mathbb{E}(\theta) - 3\sigma \geq 0 \text{ and } \mathbb{E}(\theta) + 3\sigma \leq T, \quad (34)$$

where we obtain a feasible solution that $\sigma = \min\{\frac{\mathbb{E}(\theta)}{3}, \frac{T-\mathbb{E}(\theta)}{3}\}$. Furthermore, according to Theorem 2, as the iteration proceeds, $\mathbb{E}(\theta)$increases while σ decreases. This further ensures that $\mathbb{E}(\theta) - 3\sigma \geq 0$. The constraint $\mathbb{E}(\theta) + 3\sigma \leq T$ does not require strict enforcement, as surpassing T would imply that the current iteration already yields trajectories of exceptional quality, most likely indicating successful task completion.

E LIMITATIONS

The context and agent model co-evolution framework we propose can be regarded as a form of test-time scaling, since it updates variables by summarizing experience from multiple task-execution iterations. Moreover, during the model generation control stage, θ is required to guide the bias of the original probability distribution, which necessitates recomputing and renormalizing the probabilities over multiple trajectories. Consequently, it incurs additional overhead for the trajectory generation. Nevertheless, it does provide a relatively convenient mechanism for online updating. In future work, we will further investigate the multi-trajectory issue to reduce inference overhead.

Algorithm 1 The Agent System Co-Evolution Framework

Require: Task description d, context $\mathcal{C}$, $\{\chi_1, \cdots, \chi_m\}$, $\{\omega_1^{(1)}, \cdots, \omega_m^{(1)}\}$, $\mu^{(1)}, \sigma^{(1)}$
Ensure: Final task execution result

for $k = 1$ **to** K **do**
  Sample $\phi_k \sim$ Categorical$(\{\chi_1, \cdots, \chi_m\}, \{\omega_1^{(k)}, \cdots, \omega_m^{(k)}\})$ and $\theta_k \sim \mathcal{N}(\mu^{(k)}, (\sigma^{(k)})^2)$
  Neo Context Content Generation: $\mathcal{C}_k \leftarrow$ Modify$(\mathcal{C}, \phi_k)$ ▷ See Equation 2
  Trajectory Generation: $\{$
  $s_1 = (d, \mathcal{C}_k),\ v_{\tau_k} = 0$
  for $t = 1$ **to** T **do**
    $a_t, r_t \leftarrow$ Agent$(s_t, \theta_k, R_{<t} = v_{\tau_k})$ # derive the agent response with the bias term ▷ See Equation 5
    $u_t \leftarrow$ Tool(s_t, a_t) # execute tool calls
    $s_{t+1} \leftarrow$ Concat(s_t, a_t, u_t) # concatenate to obtain the next state
    $v_{\tau_k} \leftarrow v_{\tau_k} + r_t$ # accumulate the reward value
  end for
  $\tau_k = s_{T+1}$ $\}$
  Multiple Trajectories: Execute **Trajectory Generation** N times to obtain $\{\tau_k\}, \{v_{\tau_k}\}$
  Full Likelihood Estimation: $\delta_i^{(k)} =$ Mean$(\{v_{\tau_k}\}) - \theta_k$, $\sigma_{\tau_k} =$ Std$(\{v_{\tau_k}\})$ ▷ See Equation 6
  Update for ϕ: $\omega_i^{(k+1)} = p(\phi_{k+1} = \chi_i) \leftarrow p(\phi_k = \chi_i | v_{\tau_k})$ ▷ See Equation 7
  Update for θ: $\mathcal{N}(\mu^{(k+1)}, (\sigma^{(k+1)})^2) = p(\theta_{k+1}) \leftarrow p(\theta_k | v_{\tau_k})$ ▷ See Equation 8
end for
Sample ϕ and θ from the latest distribution $(\omega_i^{(K+1)}, \mu^{(K+1)}, \sigma^{(K+1)})$, then execute the task.
return final result