LEARNING NOT TO FORGET HOW TO IMPROVE: META-FORGETTING IN SELF-EVOLVING AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-evolving language-model agents promise sustained adaptation, yet rising task performance can conceal the loss of previously acquired abilities to generate effective agent updates. We investigate this failure, termed meta-forgetting, and seek to preserve improvement ability as agents continue to evolve. We introduce Improvement Function Replay (IFR), which revisits a small set of frozen historical agent states and evaluates how effectively candidate improvement strategies modify them. IFR selects strategy updates that maximize gains on current problems while limiting declines from fixed historical reference gains, preserving the effectiveness of past improvement strategies without requiring identical edits. Across five benchmarks and three frozen language-model backbones, IFR attains higher reported mean task performance. With Qwen3-32B, its paired mean improvements in unseen ALFWorld success, OfficeQA exact accuracy, and Seal-Hard accuracy are 2.3, 2.5, and 2.6 percentage points, respectively, over Current-only + search, the strongest tested controlled alternative. IFR makes retention of improvement ability an explicit objective, complementing task-level evaluation and supporting continued adaptation with a limited replay budget.

1 INTRODUCTION

Language-model agents that learn from experience offer a path toward more reliable automation of interactive tasks, from information analysis to software maintenance. Self-evolution can reduce reliance on repeated manual redesign by using execution feedback to refine prompts (Agrawal et al., 2026) and accumulate procedural knowledge (Zhang et al., 2026c). Its long-term value depends on making these gains accumulate as tasks and failure modes change.

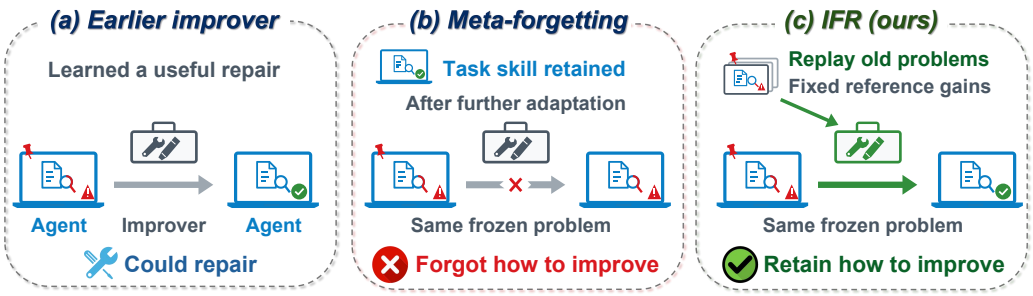

Figure 1: **Meta-forgetting and Improvement Function Replay (IFR). (a)** An improver learns to repair a retrieval failure. **(b)** After further adaptation, the agent retains the task skill, but its improver can no longer repair the same frozen problem. **(c)** IFR revisits historical problems and uses fixed reference gains to limit the loss of improvement ability while allowing new repairs.

Recursive self-improvement brings this challenge to the mechanism that generates agent updates, which we call the *improver*. MetaSkill-Evolve adapts the meta-skills that guide agent revision, making how an agent improves part of the learned system (Wang et al., 2026c). Hyperagents further demonstrates that learned improvement mechanisms can transfer across domains (Zhang et al., 2026b). The distinction between task performance and improvement potential is also central to the

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.

Huxley-Gödel Machine (HGM), which uses descendant productivity to guide evolutionary search (Wang et al., 2026b).

An effective improvement strategy can deteriorate during continued adaptation. For example, an agent may retain a successful retrieval strategy while its improver, after specializing in planning, loses the ability to repair retrieval failures (Figure 1a–b). The earlier repair can continue to support task execution even as the ability to produce such repairs deteriorates. Existing retention efforts address a related problem: Capability-Preserving Evolution (CPE) targets retention of task capabilities during agent evolution (Yu et al., 2026), while regression-aware harness optimization protects gains on previously solved tasks (Wang et al., 2026a). Retention of task capabilities and retention of improvement ability are separate objectives. When the agent and improver change together, update quality also depends on the changing target agent. Evaluating the improver on fixed modification problems isolates its contribution.

We investigate *meta-forgetting*: the loss of an evolving improver's previously acquired ability to improve agents. Our goal is to determine whether this loss can coexist with task progress, and whether preserving improvement ability supports further adaptation.

Our approach separates evaluation of the improver from progress in the task agent and makes retention of improvement ability an explicit objective. We introduce *Improvement Function Replay* (IFR; Figure 1c), which revisits a small set of historical modification problems and selects improver updates for current gains subject to a bound on the loss of their earlier effectiveness. By measuring the effects of proposed repairs against fixed historical references, IFR permits new repair strategies while preventing its retention standard from drifting after successive updates.

Our contributions are threefold:

- **We show that task progress can conceal meta-forgetting.** We distinguish task competence from retained improvement ability. A controlled cross-version study shows that historical improvement gains can decline while task performance rises, motivating a separate evaluation of the improver.
- **We develop IFR to retain an evolving improver's functional effectiveness while allowing it to discover new modifications.** Our analysis establishes a retention tolerance that does not accumulate across updates on the retained buffer, with statistical bounds under explicit sampling assumptions.
- **IFR achieves higher mean task scores and shared-state improvement gains.** Experiments span five benchmarks and three frozen backbones. With Qwen3-32B, IFR improves unseen ALFWorld success, OfficeQA exact accuracy, and Seal-Hard accuracy by 2.3, 2.5, and 2.6 percentage points over the strongest tested alternative, respectively, and yields larger one-edit gains on independently constructed states shared across methods.

2 Related Work

Feedback-driven agent adaptation. Reflexion (Shinn et al., 2023) uses verbal feedback as persistent experience. Subsequent methods optimize prompts (Opsahl-Ong et al., 2024; Agrawal et al., 2026), accumulate structured playbooks (Zhang et al., 2026c), or refine reusable skills (Mi et al., 2026; Alzubi et al., 2026). Retention of earlier improvement abilities requires a separate evaluation of the updater. Improvement Function Replay (IFR) targets this capability.

Evolving the improver. Promptbreeder and STOP evolve mutation prompts and optimizer programs, respectively (Fernando et al., 2024; Zelikman et al., 2024). DGM's archive-based self-modification (Zhang et al., 2026a) is extended by HGM's descendant-productivity criterion (Wang et al., 2026b). Hyperagents (Zhang et al., 2026b), MCE (Ye et al., 2026), and MetaSkill-Evolve (Wang et al., 2026c) make modification programs, context-engineering skills, and meta-skills editable, respectively. Hyperagents also demonstrates cross-domain transfer of improvement mechanisms. These advances leave open whether continued adaptation preserves earlier improvement responses. We isolate this question by comparing improver checkpoints on identical frozen agent states and diagnostic traces.

Retention during continual adaptation. GEM (Lopez-Paz & Ranzato, 2017) constrains replay losses, while MER (Riemer et al., 2019) meta-learns gradient alignment to reduce interference.

CPE (Yu et al., 2026) extends preservation to evolving agent components; RELAI-VCL (Wang et al., 2026a) rejects harness edits that regress on previously solved tasks. These objectives concern task competence. IFR shifts the replay unit to historical modification problems, bounding the mean positive shortfall of edit-induced gains against fixed anchors. This allows different repairs and prevents gains on one probe from cancelling losses on another; the retention constraint covers the retained buffer.

3 Preliminary Study

Can an improver forget how to improve? We study current-only self-evolution: improver revisions use the latest evolution record and are selected by gains on the current agent, with historical probes reserved for post-hoc evaluation. Across three independent trajectories of three meta-update stages each, we archive twelve frozen probes per run and evaluate all four improver versions using eight one-edit trials per probe–version pair. A response is the mean task-score gain from editing a frozen agent; its acquisition reference is the improver version immediately following probe collection. Appendix B.9 gives the protocol and diagnostic definitions.

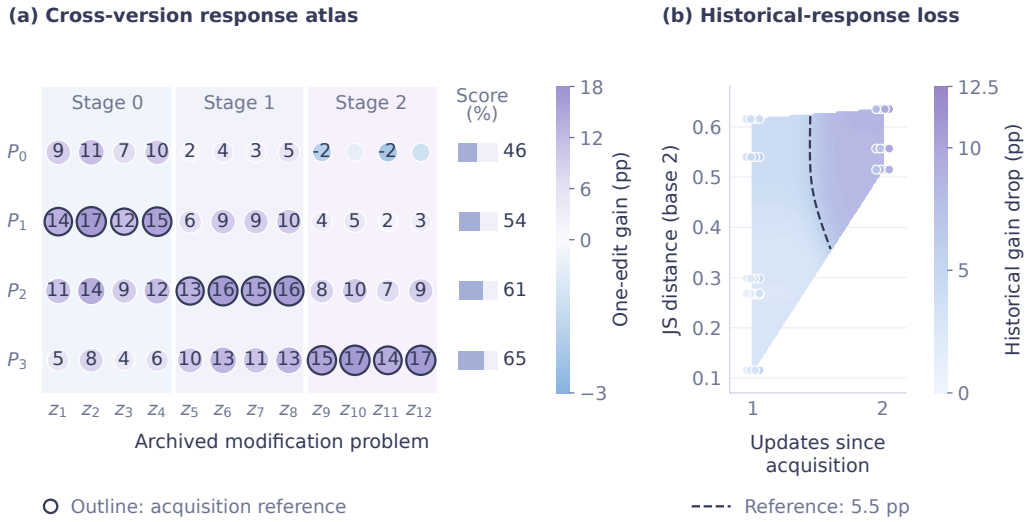

Figure 2: **Meta-forgetting under current-only self-evolution.** **(a)** Mean one-edit gains: bubble area/color encode magnitude/sign, outlines mark acquisition references, and bars show task scores. **(b)** Historical response loss for 36 paired contrasts versus update lag and task shift. The interpolated background and dashed 5.5-pp contour aid visualization. Appendix B.9 defines the contrasts, shift metric, and rendering.

Figure 2(a) shows the earliest probes' mean response falling from 14.3 to 5.9 percentage points between acquisition P_1 and the final improver P_3, a drop of 8.4 points, while the task-agent score rises from 53.6% to 65.5%. The 95th percentile of absolute frozen-improver repeat differences is 2.2 points. In panel (b), historical loss tends to be larger at longer update lags and greater task shifts. These factors covary, preventing separate causal attribution.

Meta-forgetting during self-evolution

Improving task performance does not ensure retention of improvement ability. Under current-only self-evolution, an improver can lose effectiveness on frozen historical problems even as the task agent improves. This loss of previously acquired improvement ability is *meta-forgetting*.

Design implication. These findings motivate evaluating and preserving the improver's *functional responses* on fixed historical problems through replay against fixed response anchors.

4 METHOD

Improvement Function Replay (IFR) helps an evolving improver retain its ability to improve past agent states. At each stage, it evaluates self-revisions on frozen problems, each specifying a target agent, diagnostic evidence, and an evaluation protocol. IFR selects the revision with the highest current gain whose mean shortfall from fixed historical gains stays within a chosen tolerance. The selected improver then updates the task agent (Figure 3).

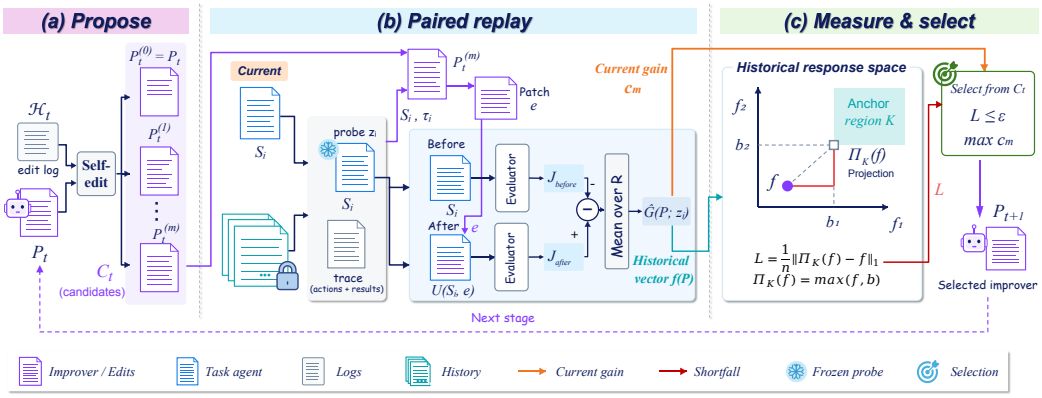

Figure 3: **Overview of Improvement Function Replay (IFR). (a)** The improver proposes self-revisions. **(b)** Candidates are evaluated using paired before/after gains on frozen current and historical probes. **(c)** IFR maximizes current gain while bounding the average of per-probe shortfalls from fixed historical anchors. The projection illustrates this shortfall.

4.1 SELF-EVOLVING IMPROVERS

At stage t, the task agent S_t executes tasks and the improver P_t generates edits. Both are textual specifications interpreted by a frozen language model. Checkpoints also include persistent readable state; dependencies outside the editable text are read-only. The task-skill/meta-skill distinction of Wang et al. (2026c) and editable-improver formulation of Zhang et al. (2026b) motivate allowing P_t to edit itself.

Let $q_\theta(e \mid X, \xi; P)$ be the frozen backbone's edit distribution under instructions P, given target X and evidence ξ. The operator $\mathcal{U}(X, e)$ applies edit e to an isolated copy of X. Using the preceding stage's record $\mathcal{H}_t$ of diagnostic traces, attempted edits, and validation outcomes, P_t proposes M self-edits (meta-edits):

$$\delta_{t,m} \sim q_\theta(\cdot \mid P_t, \mathcal{H}_t; P_t), \qquad \widetilde{P}_{t,m} = \mathcal{U}(P_t, \delta_{t,m}), \qquad m = 1, \ldots, M. \tag{1}$$

Here P_t is both editor and target; $\mathcal{H}_0$ contains initial diagnostic traces from S_0. A fixed validator checks patch syntax, editable-field boundaries, resource limits, and successful loading. Removing invalid and duplicate revisions gives $\mathcal{C}_t = \{P_t^{(0)} = P_t, P_t^{(1)}, \ldots, P_t^{(M_t)}\}$ with $M_t \le M$. Generation precedes evaluation; Appendix A.1 details editable state and validation.

4.2 MEASURING IMPROVEMENT ON FROZEN PROBLEMS

To compare improvers on the same problem, we use a frozen improvement probe $z_i = (S_i, \tau_i, \mathcal{D}_i, E_i)$: an agent state, its diagnostic trace, a scoring task set, and an execution configuration. Diagnostic and scoring tasks have disjoint identities. The improver sees only S_i and τ_i; scoring tasks are hidden and the evaluator is fixed.

Let $J(S; \mathcal{D}, E) \in [0, 1]$ be the expected mean normalized task score. The improvement response is the expected gain from one generated edit:

$$G(P; z_i) = \mathbb{E}_{e \sim q_\theta(\cdot \mid S_i, \tau_i; P)}[J(\mathcal{U}(S_i, e); \mathcal{D}_i, E_i)] - J(S_i; \mathcal{D}_i, E_i). \tag{2}$$

Each trial restores S_i and P before generation. Invalid edits receive score zero, giving gain $-J(S_i; \mathcal{D}_i, E_i)$; execution failures caused by the candidate score zero on the affected tasks. A no-op has expected gain zero. We measure this response before task-loop acceptance and evaluate accepted updates separately.

For each probe, fix R independent trial seed records. Let $\widehat{J}_{i,r}(X)$ be the mean score of X on $\mathcal{D}_i$ under trial r's evaluation seeds. Evaluate parent and edited states on the same task-seed pairs. Using edits $e_{i,r}(P)$ generated independently across trials, estimate the response as

$$\widehat{G}(P; z_i) = \frac{1}{R} \sum_{r=1}^{R} \left[\widehat{J}_{i,r}(\mathcal{U}(S_i, e_{i,r}(P))) - \widehat{J}_{i,r}(S_i) \right]. \tag{3}$$

Parent scores are shared across improvers, and all R trials enter the mean without selection by score. For each immutable (P, z_i) pair, optimization caches the first complete estimate and reuses it thereafter. The cache identifies the full states, tasks, seeds, model, and evaluator. Appendices A.2 and A.3 detail probe construction, scoring, and caching.

4.3 Anchored Improvement Function Replay

Historical shortfall. The buffer $\mathcal{B}_t$ stores the IDs of at most B historical probes. Each probe records the improver index a_i at admission and its gain as a fixed anchor. For candidate m, let $c_{t,m}$ denote its gain on the current probe z_t^{cur}:

$$\widehat{b}_i = \widehat{G}(P_{a_i}; z_i), \qquad c_{t,m} = \widehat{G}(P_t^{(m)}; z_t^{\mathrm{cur}}). \tag{4}$$

We construct z_t^{cur} from S_t at stage start and evaluate every candidate on it and all retained probes. The retention loss is the mean shortfall below the historical anchors:

$$\widehat{L}_t(P) = \begin{cases} \dfrac{1}{|\mathcal{B}_t|} \displaystyle\sum_{i \in \mathcal{B}_t} [\, \widehat{b}_i - \widehat{G}(P; z_i) \,]_+, & |\mathcal{B}_t| > 0, \\ 0, & |\mathcal{B}_t| = 0, \end{cases} \tag{5}$$

where $[x]_+ = \max(x, 0)$. Thus, gains on one probe cannot offset losses on another, and fixed anchors prevent the reference from drifting downward across updates. Geometrically, the loss is the mean distance below the individual anchor thresholds (Appendix A.6.1).

Candidate selection. Given a retention tolerance $\epsilon \geq 0$, IFR selects the candidate with the highest current gain among those satisfying the historical constraint:

$$m_t^\star \in \operatorname*{arg\,max}_{m \in \{0, \ldots, M_t\}:\, \widehat{L}_t(P_t^{(m)}) \leq \epsilon} c_{t,m}, \qquad P_{t+1} = P_t^{(m_t^\star)}. \tag{6}$$

Ties favor the incumbent, then generation order. The incumbent is always feasible (Theorem 1), so the selection is well-defined.

Task updates and probe admission. Hold the selected improver fixed for H task updates, starting from $S_{t,0} = S_t$. At update h, it uses fresh diagnostic evidence $\tau_{t,h}$ to generate one edit. A fixed development selection set measures the paired score difference $\widehat{\Delta}_{t,h}$ between edited and parent states. Accept the edit only if it is valid and improves this score:

$$\begin{aligned} e_{t,h} &\sim q_\theta(\cdot \mid S_{t,h}, \tau_{t,h}; P_{t+1}), \qquad \widetilde{S}_{t,h} = \mathcal{U}(S_{t,h}, e_{t,h}), \\ S_{t,h+1} &= \begin{cases} \widetilde{S}_{t,h}, & \text{valid and } \widehat{\Delta}_{t,h} > 0, \\ S_{t,h}, & \text{otherwise.} \end{cases} \end{aligned} \tag{7}$$

Here $h = 0, \ldots, H-1$ and $S_{t+1} = S_{t,H}$. The resulting traces and validation records form $\mathcal{H}_{t+1}$. Diagnostic and selection tasks are disjoint, with identities and evaluation seeds assigned before the stage; audit tasks are excluded.

At stage start, reserve a probe from S_t on development tasks disjoint from the current probe and task-update sets. After H updates, admit it with probability $\min(1, B/(t+1))$, independently of responses, uniformly replacing an entry if the capacity-B reservoir is full. Anchor it at $\widehat{G}(P_{t+1}; z)$ ($a_i = t+1$, zero initial shortfall). Algorithm 1 and Appendix A.5 detail the procedure; Appendix A.4 covers incomplete evaluations and interrupted stages.

Evaluation cost. Without cache hits, a meta-update costs $(M_t + 1)(|\mathcal{B}_t| + 1)R$ edit trials plus meta-proposal generation, admitted-anchor evaluation, and H task updates. All optimization calls, including rejected edits, count toward common token and environment-execution limits; independent audits use a separate budget matched across methods (Appendix B.6).

Algorithm 1 IFR: one evolution stage

Require: $S_t, P_t, \mathcal{H}_t, \mathcal{B}_t$
Output: $S_{t+1}, P_{t+1}, \mathcal{B}_{t+1}, \mathcal{H}_{t+1}$

Freeze $z_t^{\text{cur}}, z^{\text{res}}$ from S_t.
Propose M meta-edits (1). ▷ *Self-revision*
Validate/deduplicate; keep P_t in $\mathcal{C}_t$.
for $P \in \mathcal{C}_t$, current/retained z **do**
  Restore states per trial; draw R edits.
  Cache $\widehat{G}$ via (3). ▷ *Paired gain evaluation*
end for
Compute $c_{t,m}, \widehat{L}_t$ via (4)–(5).
Filter by $\widehat{L}_t \le \epsilon$. ▷ *Constrained selection*
Maximize $c_{t,m}$; break ties as in (6).
Install the selected candidate as P_{t+1}.
H edits: accept iff valid, $\widehat{\Delta}_{t,h} > 0$ (7).
Reservoir-admit z^{res} (capacity B).
If admitted: anchor $\widehat{G}(P_{t+1}; z^{\text{res}})$.
return $(S_{t+1}, P_{t+1}, \mathcal{B}_{t+1}, \mathcal{H}_{t+1})$.

4.4 Independent Retention Evaluation

After evolution, we build audit probes from saved states using held-out diagnostic and scoring tasks. A state from stage s_i uses reference checkpoint $r_i = s_i + 1$. Let $A_{t,i} = G(P_t; z_i)$ and $\mathcal{I}_t = \{i : r_i < t\}$. For nonempty $\mathcal{I}_t$, meta-level backward transfer measures the mean change from these references, adapting Lopez-Paz & Ranzato (2017):

$$\text{MBWT}_t = \frac{1}{|\mathcal{I}_t|} \sum_{i \in \mathcal{I}_t} (A_{t,i} - A_{r_i,i}) . \tag{8}$$

We report MBWT using independent audit estimates $\widehat{A}_{t,i}$, for all probes and a subset with positive screened reference gains, evaluated with fresh confirmation trials. Cross-method comparisons use an independently constructed common-state bank (Appendix B.4). Task-agent scores and accepted-update gains are evaluated separately from G (Appendix A.6.2).

4.5 Non-Accumulating Forgetting under Fixed Anchors

For each retained probe z_i, let $Q_i = P_{a_i}$ be its fixed admission reference. The mean expected shortfall is $L_{\mathcal{B}}(P) = |\mathcal{B}|^{-1} \sum_{i \in \mathcal{B}} [G(Q_i; z_i) - G(P; z_i)]_+$, with value zero for an empty buffer. Its cached counterpart, $\widehat{L}_{\mathcal{B}_t} = \widehat{L}_t$, is the retention loss in Eq. (5).

Theorem 1 (Non-accumulating fixed-anchor retention)**.** *For IFR initialized with $\mathcal{B}_0 = \varnothing$, immutable caches, and fixed admission anchors, the incumbent is always feasible. At every completed stage, $\widehat{L}_{\mathcal{B}_t}(P_t) \le \epsilon$, $\widehat{L}_{\mathcal{B}_t}(P_{t+1}) \le \epsilon$, and $\widehat{L}_{\mathcal{B}_{t+1}}(P_{t+1}) \le \epsilon$.*

Under the sampling protocol in Assumption 1, fix $\delta \in (0,1)$. With probability at least $1-\delta$, simultaneously at all completed stages, the same buffer–checkpoint pairs satisfy $L_{\mathcal{B}}(P) \le \min\{1, \epsilon + r_{\mathcal{B}}(P)\}$, where $r_{\mathcal{B}}(P)$ is the sampling correction defined in Appendix D.2.

Proof sketch. New probes have zero cached shortfall, so appending one or replacing an entry preserves feasibility. Shared parent scores cancel in gain contrasts. Conditional concentration and a weighted union bound over attempted blocks and complete improver states cover adaptive selections; taking positive parts and averaging gives the expected-loss bound. □

For retained probes, tolerance remains ϵ after T updates; statistical uncertainty may grow with probe count or improver complexity (Appendix D).

Table 1: **Task performance across five benchmarks.** Frozen Qwen3-32B; all scores are percentages (↑), averaged over five independent runs. OfficeQA uses the held-out portion of Full-246; Seal-Hard uses held-out questions with frozen retrieval. SkillsBench uses 13 held-out tasks with three trials per task in each run.

	ALFWorld		OfficeQA			Seal-Hard	AppWorld				SkillsBench	
Method	Seen	Unseen	Exact	1%	5%	Acc.	Norm. TGC	Norm. SGC	Chal. TGC	Chal. SGC	NC	C
Prompt, reflection, and memory methods												
ReAct (Yao et al., 2023)	57.9	47.0	35.8	39.7	44.8	18.9	42.9	29.6	24.9	14.7	30.8	41.5
Reflexion (Shinn et al., 2023)	50.6	40.3	37.1	40.8	45.9	19.8	44.0	30.7	26.0	15.5	32.3	43.1
MIPROv2 (Opsahl-Ong et al., 2024)	62.1	52.4	42.6	46.3	51.2	22.7	46.7	32.5	28.4	17.3	34.4	45.6
GEPA (Agrawal et al., 2026)	65.4	55.8	46.9	50.7	55.6	25.4	49.6	35.0	31.2	19.4	36.4	48.2
ACE (Zhang et al., 2026c)	—	—	—	—	—	—	55.4	40.7	37.0	23.7	38.5	49.7
Skill-Pro (Mi et al., 2026)	67.1	58.7	—	—	—	—	—	—	—	—	39.5	51.3
MARS (Hou et al., 2026)	—	—	44.8	48.6	53.4	24.1	—	—	—	—	—	—
EvoSkill (Alzubi et al., 2026)	—	—	48.3	52.1	56.9	26.8	—	—	—	—	—	—
Recursive evolution												
MetaSkill-Evolve (Wang et al., 2026c)	69.6	61.5	52.7	56.6	61.3	30.6	53.1	37.9	34.7	22.0	40.0	52.3
MCE (Ye et al., 2026)	68.9	60.1	51.8	55.5	60.1	29.7	52.3	37.5	34.1	21.6	39.0	51.8
Controls with IFR's task-update interface												
Frozen improver P_0	—	53.1	43.2	—	—	23.0	45.0	—	27.2	—	33.8	44.6
Current-only	—	64.6	54.9	—	—	31.8	54.8	—	36.1	—	41.5	53.3
Current-only + search	—	65.8	56.2	—	—	32.5	56.0	—	37.4	—	42.1	54.4
IFR (ours)	**75.1**	**68.1**	**58.7**	**62.5**	**67.2**	**35.1**	**58.3**	**43.2**	**40.1**	**26.8**	**44.6**	**56.4**

5 EXPERIMENTS

We evaluate task performance and improvement ability (RQ1, Section 5.2), candidate selection (RQ2, Section 5.3), and replay cost (RQ3, Section 5.4). Retention design and historical drift are analyzed in Appendices C.6.1 and C.7.

5.1 EXPERIMENTAL SETUP

Benchmarks and metrics. We evaluate on ALFWorld (Shridhar et al., 2021), OfficeQA (Opsahl-Ong et al., 2026), SealQA (Pham et al., 2026), AppWorld (Trivedi et al., 2024), and SkillsBench (Li et al., 2026), with Terminal-Bench 2.0 (Merrill et al., 2026) for code editing. Native metrics cover task success, answer accuracy, and task/scenario completion; shared-state one-edit gain and fixed-reference MBWT (Eq. 8) measure improvement ability and retention.

Comparison protocol. Comparators span prompt optimization, reflection/memory, and recursive evolution, including GEPA (Agrawal et al., 2026), ACE (Zhang et al., 2026c), MetaSkill-Evolve (Wang et al., 2026c), and MCE (Ye et al., 2026), alongside controlled IFR variants. Methods share development data, tools, editable-state limits, and token/interaction ceilings per benchmark/backbone pair. Code editing is evaluated separately (adaptations in Appendix B.2).

Backbones and implementation. Qwen3-32B is the primary frozen backbone; Qwen3-8B (Yang et al., 2025) and Gemma-4-31B-IT (Google, 2026) assess backbone robustness. Experiments use eight NVIDIA A100 GPUs; IFR uses replay capacity $B = 8$. Appendix B details splits, aggregation, decoding, budgets, hyperparameters, and audits. Anchor, sensitivity, and task-order controls use Qwen3-32B on ALFWorld (Appendix C.6).

5.2 RQ1: TASK PERFORMANCE AND IMPROVEMENT ABILITY

Task performance. With Qwen3-32B, IFR has the highest reported mean on all twelve task endpoints (Table 1). Against Current-only + search, the strongest tested controlled alternative, paired gains on ALFWorld unseen, OfficeQA exact, and Seal-Hard are $+2.3$, $+2.5$, and $+2.6$ percent-

Table 2: **Retention mechanisms and anchor rules on ALFWorld.** ALFWorld with Qwen3-32B; five-run means. Task success is in percent; gains and BWT are in percentage points (↑). Historical MBWT describes each method's own trajectory.

	Retention mechanism			Functional replay		Outcomes ↑				
Method	Task-score guard	Edit imitation	Functional replay	Anchor	B	Task (%)	G_{shared}	MBWT	Task BWT	G_{deploy}
Current-state selection and alternative retention targets										
Current-only	×	×	×	—	—	64.6	9.6	−7.2	−5.4	1.6
Current-only + search	×	×	×	—	—	65.8	10.2	−8.0	−5.8	1.8
Task-score preservation	✓	×	×	—	—	64.2	8.8	−3.9	**−1.2**	1.4
Historical-edit imitation	×	✓	×	—	—	65.1	9.1	−3.1	−2.7	1.5
Functional replay: buffer size and selection anchors										
Full-history replay	×	×	✓	Fixed	All	66.9	9.7	−0.8	−1.6	1.7
IFR: adjacent anchors	×	×	✓	Adjacent	8	67.8	10.5	−3.4	−2.2	1.9
IFR: historical maxima	×	×	✓	Max.	8	66.4	9.9	**−0.5**	−1.4	1.6
IFR: fixed anchors	×	×	✓	Fixed	8	**68.1**	**10.8**	−1.1	−1.8	**2.1**

age points, with 95% bootstrap CIs of [1.7, 2.9], [1.6, 3.4], and [1.5, 3.7] pp, respectively. All five matched runs favor IFR on each endpoint (Appendix C.1).

Robustness and code-editing extension. IFR's reported means lead all seven endpoints on Qwen3-8B and Gemma-4-31B-IT by 2.1–2.6 and 1.2–2.6 points over the strongest alternatives, respectively (Table 5), and all eight expanded native endpoints by 2.5–5.9 points (Table 6). On Terminal-Bench 2.0, IFR-code achieves 31.9% success, 2.9 points above HGM with lower consumption of every reported resource. Hyperagents uses fewer resources but reaches 23.8% (Table 7 in Appendix C.5).

Improvement and retention. On the independently constructed common-state bank, IFR's raw one-edit gains are 10.8, 9.5, and 7.9 pp for ALFWorld, OfficeQA, and Seal-Hard (Table 4). Paired gains over Current-only + search are $+0.6\,[0.3,\ 0.9]$, $+0.5\,[0.2,\ 0.8]$, and $+0.5\,[0.2,\ 0.8]$ pp (95% CIs): smaller than task-performance margins but positive in all five runs per benchmark. MBWT indicates residual forgetting ($-1.1, -1.3, -1.0$ pp versus $-0.8, -0.9, -0.7$ for full-history replay), measured on method-specific historical targets. Independently screened positive-reference probes yield $\text{MBWT}^{+} = -1.5, -1.7, -1.4$ pp under fresh confirmation trials independent of screening and optimization (Table 8). Fixed-anchor IFR leads task success, shared-state and deployed-update gains among ALFWorld controls (Table 2; Appendix C.6.1). Shared-state gains quantify new useful edits: 9.4 pp on AppWorld Normal and 8.8/8.4 pp on SkillsBench without/with curated skills, 0.5–0.6 pp above Current-only + search (Table 9).

5.3 RQ2: How Does the Constraint Change Candidate Selection?

We compare unconstrained and IFR-rule selections offline on ALFWorld, OfficeQA, and Seal-Hard, with diagnostic feasibility defined by $\kappa_t^{\text{diag}}(P) \leq 1$ (Appendix B.4.4).

Results. The unconstrained winner is infeasible in 55/60 ALFWorld, 57/60 OfficeQA, and 54/60 Seal-Hard stages; the offline IFR rule changes exactly these winners (Figure 4). Mean normalized shortfall falls from 1.67 to 0.34, 1.63 to 0.39, and 1.52 to 0.39, respectively (79.7%, 76.2%, and 74.3% reductions), at mean current-gain costs of 2.09, 2.29, and 2.01 pp. The rule selects incumbents in 47/60, 48/60, and 46/60 sets; only 9/60, 6/60, and 12/60 selections satisfy $0.8 < \kappa_t^{\text{diag}}(P) \leq 1$. The constraint therefore often retains the incumbent when higher-gain revisions exceed the diagnostic bound.

Takeaway Progress requires useful feasible revisions, motivating better candidate generation alongside retention-threshold tuning.

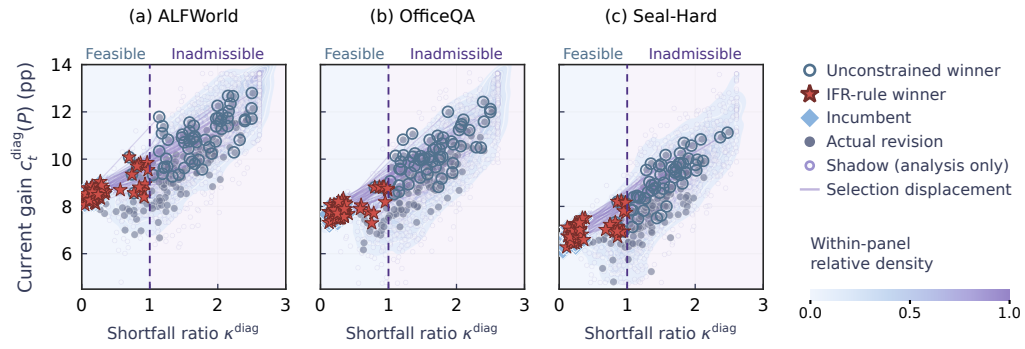

Figure 4: **Offline feasibility landscapes under IFR selection.** Axes show diagnostic normalized shortfall $\kappa_t^{\text{diag}}(P)$ and current one-edit gain (pp); the dashed line marks feasibility at $\kappa^{\text{diag}} = 1$. Contours show actual/shadow candidate density. Circles/stars mark unconstrained/IFR-rule winners among actual candidates; diamonds mark incumbents. Arrows connect differing winners.

5.4 RQ3: What Does Functional Replay Cost?

We compare shared-state gains and measured token consumption over completed 12-stage runs.

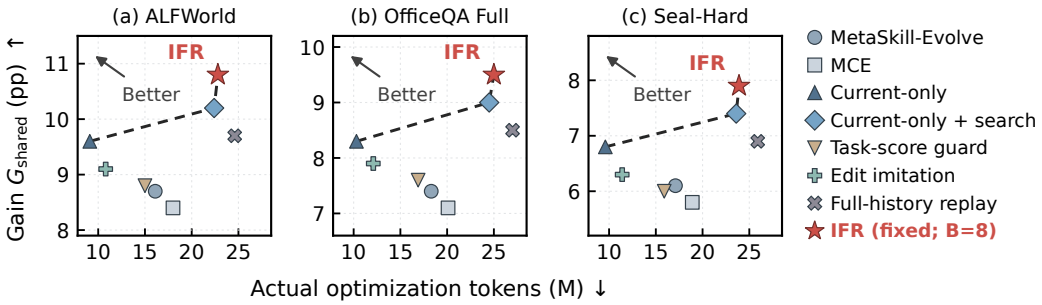

Figure 5: **IFR lies on the observed gain–cost frontier on all three benchmarks.** Eight methods with results on all three benchmarks; Qwen3-32B, five-run means, and 12 completed stages. Tokens come from Table 12; shared-state gains come from Table 4. Dashed segments connect non-dominated observed means.

Realized resource consumption. IFR uses 7.3–7.7% fewer tokens than full-history replay while improving G_{shared} by 1.0–1.1 pp; against Current-only + search, it gains 0.5–0.6 pp for 1.3–2.0% more tokens (Figure 5). Current-only remains a lower-cost frontier point; full-history replay has MBWT closer to zero (Table 4). Table 12 also reports timing, episodes, cache, and storage. At 12 completed stages, IFR uses 0.9–1.2 fewer wall-clock hours than full-history replay. Peak resident replay storage is also about one third lower (6.0–7.6 versus 9.0–11.4 MiB).

6 Conclusion

This work identifies meta-forgetting as a distinct obstacle to sustained self-evolution: progress in task execution can coexist with deterioration in the ability to generate improvements. We developed Improvement Function Replay to constrain this deterioration using fixed historical gains, with a retention tolerance that does not accumulate across updates on retained problems. Experiments across five benchmarks show that preserving historical improvement responses is compatible with stronger task performance and more effective edits on independently constructed agent states. These findings motivate evaluating self-evolving systems by both what they can accomplish and how effectively they can continue to improve. An open direction is to determine which historical problems best preserve reusable improvement strategies as new task families emerge.

AI USE STATEMENT

Generative AI tools were used to assist with checking mathematical arguments and proofs, conducting literature searches and reviewing related work, analyzing and interpreting experimental results, and drafting and editing the manuscript. The authors take full responsibility for the final content of this work, including the correctness of mathematical claims and proofs, the accuracy of citations and reported results, and their interpretation.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective Prompt Evolution Can Outperform Reinforcement Learning. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/0e9e708b6f48e14fd0ac29e167413f76-Abstract-Conference.html.

Salaheddin Alzubi, Noah Provenzano, Jaydon Bingham, Weiyuan Chen, and Tu Vu. EvoSkill: Automated Skill Discovery for Multi-Agent Systems, 2026. URL https://arxiv.org/abs/2603.02766.

Databricks. OfficeQA Benchmark Suite: Full, Pro, and Pro V2, 2026. URL https://github.com/databricks/officeqa. Release description accessed September 20, 2026.

Chrisantha Fernando, Dylan Sunil Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 13481–13544. PMLR, 2024. URL https://proceedings.mlr.press/v235/fernando24a.html.

Google. Gemma 4 Model Card, 2026. URL https://huggingface.co/google/gemma-4-31B-it.

Xinmeng Hou, Bohao Qu, Wuqi Wang, Peiliang Gong, Qing Guo, and Yang Liu. Learn like humans: Use meta-cognitive reflection for efficient self-improvement. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28802–28824. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1329. URL https://aclanthology.org/2026.acl-long.1329/.

Xiangyi Li, Yimin Liu, Wenbo Chen, Bingran You, Zonglin Di, et al. SkillsBench: Benchmarking How Well Agent Skills Work Across Diverse Tasks, 2026. URL https://arxiv.org/abs/2602.12670.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Mike A. Merrill, Alexander G. Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, et al. Terminal-Bench: Benchmarking Agents on Hard, Realistic Tasks in Command Line Interfaces, 2026. URL https://arxiv.org/abs/2601.11868.

Qirui Mi, Zhijian Ma, Mengyue Yang, Haoxuan Li, Yisen Wang, Haifeng Zhang, and Jun Wang. Skill-Pro: Learning Reusable Skills from Experience via Non-Parametric PPO for LLM Agents. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.01869.

Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. Optimizing Instructions and Demonstrations for Multi-Stage Language Model Programs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024. URL https://aclanthology.org/2024.emnlp-main.525/.

Krista Opsahl-Ong, Arnav Singhvi, Jasmine Collins, Ivan Zhou, Cindy Wang, Ashutosh Baheti, Owen Oertell, Jacob Portes, Sam Havens, Erich Elsen, Michael Bendersky, Matei Zaharia, and Xing Chen. OfficeQA Pro: An Enterprise Benchmark for End-to-End Grounded Reasoning, 2026. URL https://arxiv.org/abs/2603.08655.

Thinh Pham, Nguyen Nguyen, Pratibha Zunjare, Weiyuan Chen, Yu-Min Tseng, and Tu Vu. SealQA: Raising the Bar for Reasoning in Search-Augmented Language Models. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2506.01062.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1gTShAct7.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2010.03768.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A Controllable World of Apps and People for Benchmarking Interactive Coding Agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024. URL https://aclanthology.org/2024.acl-long.850/.

Wenxiao Wang, Priyatham Kattakinda, and Soheil Feizi. Do agent optimizers compound? a continual-learning evaluation on Terminal-Bench 2.0, 2026a. URL https://arxiv.org/abs/2607.14004.

Wenyi Wang, Piotr Piękos, Nanbo Li, Firas Laakom, Yimeng Chen, Mateusz Ostaszewski, Mingchen Zhuge, and Jürgen Schmidhuber. Huxley-gödel machine: Human-level coding agent development by an approximation of the optimal self-improving machine. In *International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/821d20219c2f14850af1b5220f0ed13f-Paper-Conference.pdf.

Zefeng Wang, Minxi Yan, Jinhe Bi, Sikuan Yan, Volker Tresp, and Yunpu Ma. MetaSkill-Evolve: Recursive self-improvement of LLM agents via two-timescale meta-skill evolution, 2026c. URL https://arxiv.org/abs/2607.05297.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, et al. Qwen3 Technical Report, 2025. URL https://arxiv.org/abs/2505.09388.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629.

Haoran Ye, Xuning He, Vincent Arak, Haonan Dong, and Guojie Song. Meta Context Engineering via Agentic Skill Evolution, 2026. URL https://arxiv.org/abs/2601.21557.

Ye Yu, Xiaopeng Yuan, Haibo Jin, Heming Liu, Yaoning Yu, and Haohan Wang. Do Self-Evolving Agents Forget? Capability Degradation and Preservation in Lifelong LLM Agent Adaptation, 2026. URL https://arxiv.org/abs/2605.09315.

Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (STOP): Recursively self-improving code generation. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=46Zgqo4QIU.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: Open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=pUpzQZTvGY.

Jenny Zhang, Bingchen Zhao, Wannan Yang, Jakob Foerster, Jeff Clune, Minqi Jiang, Sam Devlin, and Tatiana Shavrina. Hyperagents, 2026b. URL https://arxiv.org/abs/2603.19461.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic Context Engineering: Evolving Contexts for Self-Improving Language Models. In *International Conference on Learning Representations*, 2026c. URL https://iclr.cc/virtual/2026/poster/10008343.

APPENDIX CONTENTS

A Technical Details . . . 14
- A.1 Checkpoints and Edit Validation . . . 14
- A.2 Task Splits and Frozen Probes . . . 14
- A.3 Paired Scoring and Immutable Caches . . . 15
- A.4 Failures, Budgets, and Stage Completion . . . 15
- A.5 Complete Evolution Procedure . . . 16
- A.6 Properties of Anchored Replay . . . 16

B Experimental Details . . . 18
- B.1 Benchmark Releases, Splits, and Metrics . . . 18
- B.2 Comparators and Fairness Controls . . . 19
- B.3 Backbones, IFR Configuration, and Audits . . . 20
- B.4 Independent Evaluation Protocol . . . 21
- B.5 Paired Bootstrap Procedure . . . 22
- B.6 Cost Accounting . . . 22
- B.7 Metric Definitions and Comparison Scope . . . 23
- B.8 Relation to Published Evaluations . . . 23
- B.9 Preliminary Study Details . . . 24

C Additional Experimental Results . . . 26
- C.1 Headline Paired Uncertainty . . . 26
- C.2 Improvement Gains and Retention . . . 26
- C.3 Backbone Robustness . . . 27
- C.4 Additional Native Endpoints . . . 27
- C.5 Code-Editing Extension . . . 29
- C.6 Mechanisms, Audits, and Resource Use . . . 29
- C.7 Historical Drift Audit . . . 35
- C.8 Lag, Task Shift, and Forgetting . . . 36

D Theoretical Analysis and Proofs . . . 38
- D.1 Definitions and Probability-Space Construction . . . 38
- D.2 Complete Retention Guarantees . . . 39
- D.3 A Conditional One-Sided KL Bound . . . 39
- D.4 Geometry of the Inverse-KL Envelope . . . 40
- D.5 Uniform Validity under Adaptive Cache Queries . . . 41
- D.6 Empirical Feasibility and Sharp Deterministic Transfer . . . 42
- D.7 Proofs of the Retention Guarantees . . . 43
- D.8 A Variance-Sensitive Complementary Certificate . . . 45
- D.9 Why the Bound Does Not Accumulate . . . 49
- D.10 Sharpness of the KL Exponent . . . 49

A TECHNICAL DETAILS

A.1 CHECKPOINTS AND EDIT VALIDATION

A checkpoint is a manifest of UTF-8 instruction and memory documents, together with the versions of its read-only dependencies. Task-agent and improver manifests have separate editable fields. The frozen editing interface returns a structured patch specifying a manifest field, the hash of its original contents, and replacement text. A patch is valid when all fields exist in the target's editable manifest, all original hashes match, the replacements satisfy the fixed size limits, and the resulting specification loads successfully in an isolated environment. Validation checks edit structure and loading; task performance is evaluated separately.

The model weights, decoding configuration, tool interface, evaluation code, task splits, and resource limits are immutable within a run. Each generation starts from an isolated copy of the complete checkpoint. Working memory created during a trial is discarded afterward. Persistent changes require an accepted patch. Development records are passed explicitly as $\mathcal{H}_t$ during meta-edit generation and as diagnostic evidence during task-edit generation; they are not appended to P_{t+1} between meta-updates.

The initial record $\mathcal{H}_0$ is collected from S_0 on the first stage's task-loop diagnostic set using its initialization seed record. It contains no attempted edits and is charged to the development budget.

The M meta-proposal slots are attempted once each. Invalid proposals consume their slots. Identical resulting checkpoints are deduplicated by content hash, retaining the first occurrence; a revision identical to the incumbent is represented by candidate zero. Remaining candidates retain generation order. An invalid task edit is represented by $\mathcal{U}(S, e) = \perp$, with $J(\perp; \mathcal{D}, E) = 0$ and $\widehat{J}_{i,r}(\perp) = 0$. The task loop rejects $\perp$ without installing it.

A.2 TASK SPLITS AND FROZEN PROBES

Task identities are partitioned before evolution into development, historical-audit, and shared-bank pools. A task identity includes the underlying instance and all its seed variants, so variants of a scoring task cannot appear among its diagnostic tasks. Native benchmark test identities remain outside development. The shared-bank construction run uses a separate development partition.

At stage t, a permutation determined by the run seed and stage index assigns the current task block to six nonoverlapping sets: current-probe diagnostics and scoring, reserved-probe diagnostics and scoring, and task-loop diagnostics and selection. Their sizes are fixed in the run configuration. The same assignment rule and set sizes apply to every compared method. The split is constructed before generation, without conditioning on agent scores. Tasks can recur across stages according to the fixed task schedule.

The two probe traces are collected by executing S_t on their respective diagnostic sets. Each trace contains the task input, attempted actions, tool outputs, and terminal score, including successful executions. The current and reserved probes therefore share a target checkpoint but use disjoint task evidence and scoring sets. During task update h, the diagnostic set is executed again from $S_{t,h}$ with its assigned seed record. This produces fresh state-dependent evidence even when diagnostic task identities recur within the stage. The task-loop selection set stays fixed for the stage, with separate evaluation seed records for each update.

Each frozen probe stores its target manifest, diagnostic trace, scoring identities, trial seed records, evaluator version, and execution limits. Editing receives the target and trace only. Scoring executes the edited task agent after the editing call has ended; scoring tasks and responses are unavailable to that call.

For every completed stage, exactly one reserved probe is submitted to the reservoir. On arrival $k = t + 1$, a reservoir with fewer than B entries appends the probe. Once full, it draws a uniform integer u from $\{1, \ldots, k\}$ and replaces slot u exactly when $u \leq B$. The draw uses a dedicated random stream independent of edit generation and scoring. An admitted probe stores $a_i = t + 1$ and $\widehat{b}_i = \widehat{G}(P_{t+1}; z_i)$. Its anchor is never revised. At a predetermined horizon of k arrivals, the reservoir rule assigns each probe ID retention probability $\min(1, B/k)$; it does not guarantee coverage of every historical stage.

A.3 Paired Scoring and Immutable Caches

For probe i, let $\mathcal{D}_i = \{x_{i,1}, \ldots, x_{i,K_i}\}$. Trial r has one edit-generation seed and evaluation seeds $\omega_{i,r,k}$ for the scoring tasks. Let $s(X; x, \omega) \in [0, 1]$ denote the normalized score of checkpoint X on task x under the probe's execution configuration. The score used in Eq. (3) is

$$\widehat{J}_{i,r}(X) = \frac{1}{K_i} \sum_{k=1}^{K_i} s(X; x_{i,k}, \omega_{i,r,k}). \tag{A.1}$$

Every task execution starts from a clean environment. Evaluation seeds control environment randomness and task-agent decoding. Editing and evaluation use independent random streams. Seed records are drawn independently across trials, then fixed with the probe and shared across improvers. Sharing an edit-generation seed across different instructions does not imply semantically matched edits.

The expected-response guarantee in Section 4.5 additionally requires the reserved trial block to remain independent of, and unexposed to, the choices of its probe, reference, and confidence prior before its first evaluation. This includes the reserved parent-scoring outcomes. An attempted reserved block receives its statistical index before any outcome is exposed, even if the stage subsequently fails to complete. This index is distinct from the completed-stage reservoir arrival count; cache reuse preserves the original block index and prior. These are the conditions of Assumption 1.

The R parent scores are computed once for a probe and reused for every candidate. Each candidate generates R edits from the original target state; applying edit r does not change the starting state for edit $r + 1$. The response cache is written only after all R trial outcomes are complete. Its key hashes the improver manifest, target manifest, diagnostic trace, scoring tasks, seed records, model and decoding configuration, execution limits, evaluator, and R. Cache identity depends on contents; checkpoint indices serve as metadata. Returning to identical contents reuses the existing response. Partial trial records are retained for failure recovery, but never enter a candidate mean until the evaluation is complete. Completed per-trial scores are retained for the variance-sensitive certificate in Proposition 1.

For task-loop selection, denote the stage's selection tasks by $\{x^{\mathrm{sel}}_{t,k}\}_{k=1}^{K^{\mathrm{sel}}_t}$ and the update's paired seeds by $\omega^{\mathrm{sel}}_{t,h,k}$. The acceptance statistic is

$$\widehat{\Delta}_{t,h} = \frac{1}{K^{\mathrm{sel}}_t} \sum_{k=1}^{K^{\mathrm{sel}}_t} \left[s(\widetilde{S}_{t,h}; x^{\mathrm{sel}}_{t,k}, \omega^{\mathrm{sel}}_{t,h,k}) - s(S_{t,h}; x^{\mathrm{sel}}_{t,k}, \omega^{\mathrm{sel}}_{t,h,k}) \right]. \tag{A.2}$$

Only one task edit is generated for each update; Eq. (A.2) evaluates that edit across the selection set. Equality retains the parent. These calls have separate seed records from probe evaluation.

A.4 Failures, Budgets, and Stage Completion

A malformed patch, invalid checkpoint, prohibited field access, and an execution limit exceeded by the candidate are candidate failures. A generation failure attributable to the candidate yields the invalid-edit outcome. A failed edited-agent task execution scores zero on that task; an invalid edit scores zero on the entire scoring set. Such outcomes remain in the mean and consume their full measured cost.

Service unavailability, host-process failure, and evaluator-service errors are infrastructure failures. Each affected call is retried at most twice with the same request and seed record. Already completed calls are reused. An unresolved infrastructure failure leaves the response missing; it is never converted to a zero score.

A stage is committed only after candidate evaluation, all H task updates, and any required new-anchor evaluation are complete. Until then, the previous stage's $(S_t, P_t, \mathcal{B}_t, \mathcal{H}_t)$ remains the committed state. An unresolved infrastructure failure terminates the run at that state. The run also stops before a call whose declared resource allowance cannot fit within the remaining token and environment-execution budgets. An unfinished stage is discarded, while all of its costs and its stopping reason are retained. The reservoir arrival count increases only on a completed stage.

All methods use the same total token and environment-execution limits. Stopping at either limit is recorded as budget exhaustion. Reports include the number of completed stages, missing audit entries, and failure categories. Comparisons at a specified stage include only runs that reached it; final-state comparisons use the last committed checkpoint of each run and report the corresponding stage counts.

A.5 Complete Evolution Procedure

Algorithm 2 uses integer parameters $T, H, M, B, R \geq 1$ and $\epsilon \geq 0$. The run configuration fixes task-set sizes, task order, all random streams, edit limits, and evaluation limits before execution. The stage-completion policy in Appendix A.4 applies to every generation and execution call.

Algorithm 2 Improvement Function Replay

Require: Initial checkpoints S_0, P_0; stage splits; T, H, M, B, R, ϵ
$\mathcal{B}_0 \leftarrow \varnothing$; collect initial record $\mathcal{H}_0$
for $t = 0, \ldots, T-1$ **do**
  Freeze S_t; construct z_t^{cur} and reserved probe z^{res}
  Generate and validate $\mathcal{C}_t$ using Eq. (1)
  for $P_t^{(m)} \in \mathcal{C}_t$ **do**
    Compute $c_{t,m}$ on z_t^{cur} using Eq. (3)
    Complete cached responses on every $i \in \mathcal{B}_t$
    Compute $\widehat{L}_t(P_t^{(m)})$ using Eq. (5)
  end for
  Select P_{t+1} using Eq. (6); set $S_{t,0} \leftarrow S_t$
  for $h = 0, \ldots, H-1$ **do**
    Execute $S_{t,h}$ on the assigned diagnostic tasks to obtain $\tau_{t,h}$
    Generate one task edit; apply Eq. (7)
    Record diagnostics, the attempted edit, and its validation outcome
  end for
  $S_{t+1} \leftarrow S_{t,H}$; assemble $\mathcal{H}_{t+1}$
  $\mathcal{B}_{t+1} \leftarrow \mathcal{B}_t$; set $k \leftarrow t+1$
  if $k \leq B$ **then**
    Evaluate $\widehat{G}(P_{t+1}; z^{\mathrm{res}})$ and append the anchored probe
  else
    Draw $u \sim \mathrm{Uniform}\{1, \ldots, k\}$
    if $u \leq B$ **then**
      Evaluate $\widehat{G}(P_{t+1}; z^{\mathrm{res}})$ and replace reservoir slot u
    end if
  end if
  Commit $(S_{t+1}, P_{t+1}, \mathcal{B}_{t+1}, \mathcal{H}_{t+1})$ and the cost record
end for
return Committed checkpoints, probe records, response cache, and costs

A.6 Properties of Anchored Replay

A.6.1 Response-Space Interpretation

Fix the buffer ordering and let $\mathbf{b}$ contain its anchors and $\mathbf{f}(P)$ the cached responses of P. The coordinatewise retention region is $\mathcal{K} = \{\mathbf{v} : \mathbf{v} \geq \mathbf{b}\}$. Its Euclidean projection and the loss obey

$$\Pi_{\mathcal{K}}(\mathbf{f}) = \max(\mathbf{f}, \mathbf{b}), \qquad \widehat{L}_t(P) = \frac{\|\Pi_{\mathcal{K}}(\mathbf{f}(P)) - \mathbf{f}(P)\|_1}{|\mathcal{B}_t|}, \quad |\mathcal{B}_t| > 0, \tag{A.3}$$

where the maximum is coordinatewise. To verify the projection, minimize the squared distance separately in each coordinate: values above the anchor remain unchanged and values below it move to the anchor. Taking the ℓ_1 norm of the residual yields Eq. (5). The computation evaluates response shortfall; candidate selection remains the finite search in Eq. (6).

A.6.2 Raw Editing Gain and Accepted-Update Gain

For a fixed probe, define the true gain of edit e as $\Delta(e; z) = J(\mathcal{U}(S, e); \mathcal{D}, E) - J(S; \mathcal{D}, E)$. Equation (2) averages Δ before acceptance. Under an exact acceptance test that installs only strictly positive edits, the expected one-step deployed gain would be

$$G_{\text{accept}}(P; z) = \mathbb{E}_{e \sim q_\theta(\cdot|S,\mathcal{T};P)}[[\,\Delta(e; z)\,]_+]\,. \tag{A.4}$$

These objectives can rank improvers differently. With parent score 0.5, an improver that always yields gain 0.2 has raw and accepted gain 0.2. An improver yielding gains 0.5 and -0.5 with equal probability has raw gain zero and accepted gain 0.25.

In the task loop, acceptance uses the noisy development statistic $\widehat{\Delta}_{t,h}$, while subsequent edits target changed states. Neither Eq. (2) nor the idealized Eq. (A.4) determines the resulting trajectory. IFR constrains raw single-edit responses. Evaluation of deployment performance executes the fixed task loop and scores the committed task agents on independent tasks, recording accepted-update gains separately.

B EXPERIMENTAL DETAILS

This appendix specifies benchmark releases, evaluation splits, comparator adaptations, resource accounting, and implementation details for Tables 1–7. The preliminary study follows the protocol in Appendix B.9.

B.1 BENCHMARK RELEASES, SPLITS, AND METRICS

ALFWorld. We use the text environment and its six task families, preserving the native training and held-out seen/unseen partitions (Shridhar et al., 2021). The primary endpoint is unseen-task success; seen-task success is a secondary endpoint. Each episode permits at most 50 environment actions for every method. This provides direct task overlap with MetaSkill-Evolve and Skill-Pro.

OfficeQA. We distinguish OfficeQA Full (246 questions), its overlapping Pro subset (133 questions), and Pro V2 (90 questions over a different corpus) (Opsahl-Ong et al., 2026; Databricks, 2026). Development and in-corpus evaluation use a grouped partition of Full; Pro results include only its held-out members. Pro V2 is a frozen-transfer test without adaptation on its questions. All 90 questions are evaluated in each of five runs, yielding 450 binary outcomes. All methods receive the same complete parsed document corpus and retrieval tools, without oracle source-page annotations. We use the official reward script's answer accuracy: exact agreement is the primary endpoint, and 1% and 5% relative-error tolerances are secondary endpoints. These versioned results are separate from the older 200-question OfficeQA results reported by EvoSkill (Alzubi et al., 2026).

SealQA. Seal-Hard and its Seal-0 subset are scored separately using the released correctness rubric and GPT-4o-mini auto-rater (Pham et al., 2026). Incorrect and unattempted answers count as unsuccessful. Related questions across Seal-0, Seal-Hard, and LongSeal share a split assignment. The controlled condition uses a fixed, queryable retrieval snapshot and the same search interface for all methods. Snapshot construction excludes answer keys and method-specific query traces. We call this condition *frozen retrieval*; published live-web scores and LongSeal's supplied-document results are separate settings. The judge version and prompt are fixed before execution, with outputs cached independently of the improver. Evaluation-only judge API costs are accounted for separately from local backbone inference.

AppWorld. We preserve the official train/development and test-normal/test-challenge separation (Trivedi et al., 2024). Task goal completion (TGC) and scenario goal completion (SGC) are reported separately on both test sets, following ACE (Zhang et al., 2026c). Normal SGC uses 56 scenarios per run, or 280 scenario outcomes across five runs. A scenario succeeds only if all its task variants succeed; all variants share a split assignment. Agents execute inside reset environments with identical API documentation and the official evaluation tests. Instruction and memory updates do not modify API implementations or evaluators.

SkillsBench and Terminal-Bench. We use the 87-task, eight-domain SkillsBench release and the 89-task Terminal-Bench 2.0 release (Li et al., 2026; Merrill et al., 2026). Our task-performance subsets contain 13 held-out SkillsBench tasks and 14 held-out Terminal-Bench tasks. Each task has three independent trials per run across five runs, yielding 195 and 210 binary outcomes, respectively. We report mean single-attempt success using native verifiers and equal task weights, with every attempt included in the mean. SkillsBench has separate no-curated-skill and curated-skill conditions; within each condition, every method receives identical external skill access. Held-out task-specific skills are accessible only under the corresponding evaluation condition, never during development. Container images, task timeouts, and verifier versions are shared. The 87/89-task counts describe the source releases; our 13/14-task results evaluate held-out subsets.

Aggregation of discrete endpoints. For binary task- and scenario-level endpoints, we sum raw success indicators over all evaluated instances and five runs, then divide by the total number of binary outcomes. Rounded per-run percentages are never averaged; only the aggregate proportion is rounded to one decimal place. With N equally weighted test instances and q trials per instance per run, K successful outcomes give $100K/(5Nq)$ percent. We use $q = 3$ for SkillsBench and Terminal-Bench; Normal SGC and Pro V2 use 280 and 450 outcomes. Across five runs, ALFWorld unseen/seen use 670/700 outcomes, AppWorld Normal TGC uses 840, and Challenge SGC uses 695. Count-based differences are computed before rounding, so subtracting displayed scores can differ by 0.1 percentage points from the reported difference.

Partitions and task streams. Native test identities remain outside development. For datasets without a native adaptation split, task groups are assigned to evolution/tuning (60%), independent shared-bank construction (10%), historical audit (15%), and shared-bank audit (15%). Splitting precedes execution, stratifies by published task category, and groups question paraphrases, seed variants, and shared scenario instances together. For native-split datasets, the two development pools come from training/development data and the audit pools from held-out data. Each audit pool is divided into diagnostic and scoring identities before trace generation. All methods use the same split manifest. The final frozen task agent is also evaluated on the complete native test set where one exists. Audit feedback and test outcomes do not select checkpoints or hyperparameters.

Each run contains 12 stages with a fixed category-blocked schedule and cyclic category revisits. Categories too small for the within-stage partition are merged in a predetermined order before execution. Each stage uses two diagnostic and four scoring tasks for each of the current probe, reserved probe, and task-update loop: 18 distinct task identities in total. Tasks can recur between stages; small-benchmark streams therefore reuse tasks across the 12 stages. Split hashes, retained group counts, and category schedules are fixed before execution. The additional order comparison is restricted to Qwen3-32B on ALFWorld: IFR and Current-only + search use a shuffled schedule with the same multiset of development tasks and categories mixed across stages (Table 11).

B.2 Comparators and Fairness Controls

Text and skill evolution. ReAct and Reflexion (Yao et al., 2023; Shinn et al., 2023) provide fixed-policy and verbal-feedback baselines across the five main benchmarks. Reflexion's repeated attempts count toward the common interaction budget. MIPROv2 and GEPA (Opsahl-Ong et al., 2024; Agrawal et al., 2026) optimize task-agent instruction modules with their original candidate-selection mechanisms. The primary MIPROv2 comparison disables demonstrations, matching GEPA's instruction-only comparison. These baselines also appear in the GEPA, ACE, and MCE comparison tables.

ACE retains its generator/reflector/curator updates, and Skill-Pro retains its skill pool and non-parametric PPO gate (Zhang et al., 2026c; Mi et al., 2026). Their principal comparisons are AppWorld and ALFWorld, respectively; SkillsBench comparisons adapt these mechanisms to its interface. Skill-Pro's action-likelihood computations use the same frozen backbone and count toward inference cost. MARS (Hou et al., 2026) provides stagewise reflection and memory on the two QA benchmarks. EvoSkill (Alzubi et al., 2026) uses the common text interface on OfficeQA and SealQA, with persistent updates restricted to shared skill documents. These scores evaluate the adapted algorithms under our common task and interface protocol.

MetaSkill-Evolve and MCE (Wang et al., 2026c; Ye et al., 2026) are the closest comparisons for evolving task-level and meta-level instructions. MetaSkill-Evolve retains its multiple roles, branching, and cross-branch information exchange; MCE retains nested skill/context optimization. Their five-benchmark evaluations share the frozen backbone and task tools. All inner-loop calls, branches, and context-management calls count toward the common budget. A single-chain current-only ablation is distinct from these complete methods. The run manifest identifies interface adaptations. All memory methods use development-only adaptation followed by frozen evaluation; persistent learning across test questions is a separate online setting excluded from the main comparison.

Controlled IFR variants. The controls share IFR's initialization and patch interface: (i) a frozen improver P_0 with an evolving task agent; (ii) current-only meta-selection without the retention constraint; (iii) current-only selection reallocating the replay allowance to additional current-state proposals and evaluations; (iv) task-score preservation inspired by capability-preserving evolution (Yu et al., 2026); (v) historical-edit imitation supplying prior accepted edits as demonstrations without evaluating their functional consequences; and (vi) full-history functional replay. In (iv), the task loop bounds the edited task agent's mean positive score shortfall on stored old development task sets by 0.02 relative to fixed task-score anchors. Its meta-selector remains current-only. This is a controlled baseline isolating task-regression protection in CPE. Full-history replay retains every probe with IFR's anchors and selector under the common budget, providing a storage/cost reference. The additional anchor comparison uses Qwen3-32B on ALFWorld only (Table 2). It changes selection anchors while retaining fixed acquisition references for independent audits; Appendix C.6 defines both variants.

Code-editing extension. IFR-code, Hyperagents, DGM, and HGM (Zhang et al., 2026b;a; Wang et al., 2026b) share an initial executable agent, editable code modules, tools, sandbox, frozen backbone, and resource ceilings on Terminal-Bench 2.0. The environment, model client, resource monitor, and verifier remain read-only. Native search archives and expansion/evaluation policies are retained. These ports use a common benchmark and are distinct from the original SWE-bench or Polyglot evaluations. The endpoint is task success versus optimization cost. Functional MBWT requires a separately restorable improver and the same one-edit interface; otherwise it is inapplicable. The theoretical guarantees cover the text interface; this code-editing extension is evaluated empirically.

Budgets and reporting. Each optimization run has ceilings of 30 million input-plus-output tokens and 10,000 executed task episodes, stopping when either is reached or the 12-stage protocol finishes. All methods receive the same ceilings within a benchmark/backbone pair. Costs include proposals, reflection, diagnostics, selection, replay, and retries; cached results incur no new model call. The current-only search control spends IFR's replay allowance on current-state search under the same stage schedule and task-update count. We report realized consumption in tokens, episodes, completed stages, GPU-hours, and wall time under each ceiling. Audit and final-evaluation costs are separate, with identical evaluation allocations across methods. Table 7 reports actual optimization consumption for code editing. Allocated A100-hours equal wall-clock hours multiplied by eight because each optimization run reserves the full eight-A100 machine, including idle time. Controlled timing comparisons use the same hardware.

B.3 Backbones, IFR Configuration, and Audits

Model execution. The model identifiers are `Qwen/Qwen3-8B`, `Qwen/Qwen3-32B`, and `google/gemma-4-31B-it` (Yang et al., 2025; Google, 2026). All generative roles share a frozen checkpoint with bfloat16 inference. Qwen3-32B supports the primary and code-editing comparisons; Table 5 specifies the other backbones' comparator coverage, fixed before test evaluation. Experiments use eight NVIDIA A100 GPUs. Serving uses four-way tensor parallelism for the 31B/32B models and one GPU per 8B replica, with concurrency subject to available memory.

Task execution uses temperature 0; edit and meta-edit generation use temperature 0.7 and top-p 0.95. Qwen models use non-thinking mode. All methods have a 32,768-token context ceiling and a 4,096-token output limit per call, including generated reasoning. This is the controlled comparison's decoding configuration. Persistent task and improver documents each have a 4,096-token limit counting all readable memory; uncounted retrieval cannot bypass this limit. Official task timeouts are fixed. Non-container QA episodes allow at most 32 tool calls and AppWorld episodes at most 100 interaction turns. Overflow handling is deterministic and shared.

IFR hyperparameters. The default configuration is $T = 12$ stages, $H = 3$ task updates, $M = 3$ meta-proposals, reservoir capacity $B = 8$, $R = 3$ development repetitions, and $\epsilon = 0.02$ on scores normalized to $[0, 1]$. The incumbent joins the M revisions, with every retained probe evaluated for each distinct candidate. Anchors and cached responses are immutable. One reserved probe enters uniform reservoir sampling per completed stage; $T > B$ ensures buffer replacement.

Additional sensitivity experiments are restricted to Qwen3-32B on ALFWorld (Table 10). The grid is $B \in \{4, 8, 16\}$, $\epsilon \in \{0, 0.01, 0.02, 0.05\}$, $H \in \{1, 3, 6\}$, and document limits of 2,048/4,096/8,192 tokens, with one factor changed at a time under common resource ceilings. These configurations are separate from the primary comparison and use the same five paired run seeds. Pilot tuning is restricted to a reserved 20% of development groups, at most eight configurations per method, and a common pilot budget. The primary configuration is fixed before confirmatory evaluation.

Retention and new-state improvement. Functional audits use Qwen3-32B on the three benchmarks in Table 4 and the AppWorld and SkillsBench conditions specified in Table 9. They restore one saved task-agent state per completed stage and evaluate improver checkpoints with $R_{\text{audit}} = 8$ fresh one-edit trials per probe. Reference screening uses a separate $R_{\text{screen}} = 8$ trials, independent of confirmation and optimization seeds. All-probe MBWT measures change relative to the fixed acquisition reference. Screening-positive MBWT, screening coverage, and confirmation reference gains distinguish retention of previously positive improvement ability from other changes (Appendix B.4.2).

A separate current-gain construction run builds the common-state bank from its reserved development pool, saving states by stage index without score-based selection. It reuses a fixed two-task diagnostic set and a disjoint four-task selection set, without a replay reservoir or audit tasks. After the bank is frozen, compared improvers receive identical states, diagnostic evidence, and scoring tasks. The shared-bank endpoint is mean raw one-edit gain; its difference from P_0 measures acquired improvement ability. Single-level methods without an evolving improver are compared through task performance and efficiency. For a constant improver, self-difference is zero by definition. Task-level backward transfer compares the final task agent's scores on earlier blocks with its scores immediately after learning those blocks. Evaluation snapshots and feedback remain outside optimization.

Repetitions and uncertainty. Five independent runs use paired initialization and task schedules across methods. Tables report means, and Appendix C.1 provides 95% intervals for the six headline paired comparisons using 2,000 paired whole-run bootstrap resamples (Appendix B.4.3). Three task-evaluation trials and eight audit-edit trials retain separate denominators. The run-record specification includes dataset/model and software/container versions, seeds, editable fields, judge settings, costs, and incomplete observations, under the failure rules in Appendix A.4.

B.4 Independent Evaluation Protocol

B.4.1 Historical Audits and Shared States

Historical probes restore the stage-entry states saved during evolution. After the run, their diagnostic traces are generated on the historical-audit diagnostic split and their scoring tasks are taken from its disjoint scoring split. The probe schedule is fixed independently of observed retention scores. All improver versions receive the same trace for a given probe. A probe from stage s_i always uses checkpoint $r_i = s_i + 1$ as its reference, including when that checkpoint has the same contents as an earlier improver.

Historical probes share target states with the development trajectory but use held-out tasks. For cross-method comparison on common targets, a separate development run produces a checkpoint bank before the compared runs begin. This construction run uses current-gain selection with the same editing interface and its reserved development partition. Its snapshots are fixed by stage index, without score-based selection. Shared-bank diagnostic and scoring tasks are disjoint from its construction tasks. Every compared improver is evaluated on the same resulting probes after optimization ends. Bank data are excluded from candidate selection and tolerance tuning.

B.4.2 Screening and Confirmation

Audit trials use seeds independent of the optimization cache. For each probe, R_{screen} edits from P_{r_i} estimate its reference gain. Probes with a strictly positive screening mean form the screening-positive subset. Its membership is fixed before confirmation, and its coverage is reported.

Confirmation evaluates each response with R_{audit} fresh edits. Checkpoints share the same task and environment seeds. Across $T + 1$ checkpoints and N audit probes, this estimates the matrix $\widehat{A}$ and the reference gains, including negative and zero outcomes. Within a seed block, identical checkpoint contents share their trial results. Screening and confirmation blocks remain independent. All-probe MBWT and screening-positive MBWT are reported separately, together with per-probe contrasts and confirmation reference gains. The subset criterion is an empirical screen; evidence of lost positive improvement ability additionally requires confirmation of a positive reference gain and a negative retention contrast.

The parent term cancels in a paired contrast. If $\widehat{J}_{i,r}^{\text{audit}}$ denotes an audit trial's scoring functional, then

$$\widehat{A}_{t,i} - \widehat{A}_{r_i,i} = \frac{1}{R_{\text{audit}}} \sum_{r=1}^{R_{\text{audit}}} \left[\widehat{J}_{i,r}^{\text{audit}}(\mathcal{U}(S_i, e_{t,i,r})) - \widehat{J}_{i,r}^{\text{audit}}(\mathcal{U}(S_i, e_{r_i,i,r})) \right]. \tag{B.1}$$

Both edits in a summand use the same frozen state and diagnostic trace. The shared evaluation seeds reduce extraneous environmental variation without imposing equality between the generated edits.

B.4.3 Uncertainty and Missing Observations

A complete evolution run is the unit for uncertainty across training trajectories. Matched methods share initial states, task order, and the evaluation schedule. Each run first yields its own MBWT and task-score summaries; comparisons average the matched run-level differences. Percentile confidence intervals use 2,000 bootstrap resamples of complete matched runs. All probe trials from one trajectory remain together as one independent unit.

For a fixed checkpoint pair and probe, paired resampling of complete audit trial records estimates Monte Carlo uncertainty conditional on that probe. The same sampled trial indices are used for the reference gain and its retention contrast. The screening subset stays fixed during confirmation resampling. A claim of lost positive response requires a positive lower 95% confidence limit for the reference gain and a negative upper 95% confidence limit for the contrast. These intervals are conditional assessments; the run-level intervals describe variability across evolution trajectories.

Infrastructure failures follow the same two-retry rule as development evaluation. An incomplete audit response remains missing. A contrast is reported only when both entries are complete. A scheduled aggregate with an incomplete required contrast is marked unavailable, with its completion fraction reported; missing entries are never replaced by zero. Shared-bank comparisons use the identical prescribed probe set across methods.

B.4.4 Offline Candidate-Selection Diagnostic

We compare selections offline on ALFWorld, OfficeQA, and Seal-Hard over five 12-stage runs each. Each recorded set contains the incumbent, three actual meta-revisions, and 24 *analysis-only shadow candidates* from the same proposal interface and scoring protocol, totaling $5 \times 12 \times 28 = 1{,}680$ records per benchmark. Shadows contribute only to the plotted landscape. Diagnostic current gain and historical shortfall are

$$c_t^{\mathrm{diag}}(P) = \widehat{G}^{\mathrm{diag}}(P; z_t^{\mathrm{cur}}), \qquad \kappa_t^{\mathrm{diag}}(P) = \frac{\widehat{L}_t^{\mathrm{diag}}(P)}{\epsilon}, \tag{B.2}$$

where $\epsilon = 0.02$ on $[0, 1]$ scores; gains are displayed in percentage points. These diagnostic scores are distinct from immutable training-cache values. Feasibility requires $\kappa_t^{\mathrm{diag}}(P) \le 1$. For $\mathcal{C}_t$ containing only the incumbent and three actual revisions, we compare

$$P_t^{\mathrm{unc}} \in \arg\max_{P \in \mathcal{C}_t} c_t^{\mathrm{diag}}(P), \qquad P_t^{\mathrm{rule}} \in \arg\max_{P \in \mathcal{C}_t:\, \kappa_t^{\mathrm{diag}}(P) \le 1} c_t^{\mathrm{diag}}(P). \tag{B.3}$$

Ties follow Equation 6. Identifying the offline winner P_t^{rule} with the deployed P_{t+1} requires the original selection cache and deployment logs. Means include all 60 sets per benchmark, including coincident winners.

Figure 4 shows kernel-smoothed actual/shadow candidate density, normalized to unit peak per panel.

B.5 Paired Bootstrap Procedure

For the headline comparisons, five independent evolution runs are paired across methods by initialization, task schedules, and evaluation schedules. The independent unit is a complete evolution run. Let y_r^{IFR} and y_r^{base} denote an endpoint measured on matched run $r \in \{1, \ldots, 5\}$. We first form the paired run-level differences

$$d_r = y_r^{\mathrm{IFR}} - y_r^{\mathrm{base}}.$$

A bootstrap replicate samples five run indices with replacement from $\{1, \ldots, 5\}$ and averages the corresponding d_r. We use 2,000 such paired whole-run resamples and report the 2.5th and 97.5th percentiles as 95% confidence intervals. Each resample keeps all tasks, probes, and edit trials from a run together, preserving the run as the independent unit.

B.6 Cost Accounting

Let $n_t = |\mathcal{B}_t|$. With no response-cache hits, evaluation of $M_t + 1$ improvers on the current and historical probes generates $(M_t + 1)(n_t + 1)R$ edits. Each admitted new probe adds R anchor-generation

trials. The stage also uses M meta-proposal slots, H task-edit slots, diagnostic executions, and task-loop selection evaluations.

For one previously unevaluated probe with K scoring tasks and C candidate improvers, full valid evaluation requires CRK edited-agent task executions and RK shared parent executions. Candidate failures can end execution early; all actual calls are counted. Loading a cached response incurs no new model calls and no new task executions, but the calls that created it remain charged to the run. Storage comprises the current reservoir plus archived checkpoints and response records required for reproducibility and post-run audit.

Token accounting includes input and output tokens for diagnostic execution, meta-edits, probe edits, task edits, rejected candidates, retry calls, and anchor scoring. Environment accounting includes every executed diagnostic, selection, probe, and validation task. Initial record construction is charged identically across methods. Exhausting either global limit ends optimization under Appendix A.4; wall-clock time and cache hit rates are reported alongside both budgets.

Without audit-cache hits, confirmation on $T + 1$ checkpoints and N historical probes requires $(T + 1)NR_{\text{audit}}$ edit trials and reference screening requires NR_{screen} trials. Shared-bank evaluation is charged separately. Audit budgets are identical across compared methods and are excluded from their optimization budgets.

B.7 Metric Definitions and Comparison Scope

Notes for Table 1. ALFWorld: seen/unseen success. OfficeQA: exact accuracy or 1%/5% numerical relative-error tolerance. Seal-Hard: accuracy. AppWorld: task/scenario completion (TGC/SGC), Normal/Challenge (Norm./Chal.). SkillsBench: success without/with curated skills (NC/C). Seen, tolerance, and SGC values also appear in Table 6. —: unreported endpoint. Bold/underlining denote the highest/second-highest reported values per column. Citations identify algorithms. MIPROv2 uses no demonstrations; EvoSkill uses the common text interface. Task transfers and memory restrictions follow Appendix B.2. "No curated" permits skills learned from development data while excluding externally supplied task-specific skills.

Outcome tables report means; Table 7 also reports optimization consumption. Table 3 provides paired mean differences, 95% whole-run bootstrap intervals, and positive-run counts for six headline comparisons against Current-only + search (Appendix B.4.3). The reporting protocol specifies five run-level values, evaluation size, dataset/model revisions, and split hash. Additional tables in Appendix C.6, including anchor variants, provide descriptive means without run-level observations or intervals. Missing observations and measured zeros are recorded separately.

The functional audits distinguish the change in shared-bank gain from P_0, screening-positive MBWT, screening coverage, confirmation reference gain, and task-level BWT. Additional functional coverage uses Qwen3-32B on AppWorld Normal and both SkillsBench access conditions (Table 9); AppWorld Challenge remains a native task-performance endpoint. Fixed-P_0 repeat variability is distinct from its zero population self-difference. Functional measures requiring a separately restorable improver are inapplicable to methods without that component.

Table 7 reports optimization tokens, episodes, allocated A100-hours, and wall time. Audit and final-evaluation costs are separate from optimization consumption. Cost-normalized comparisons align token and episode consumption and the number of completed stages. Overlapping subsets, distinct access conditions, and different native metrics are reported separately.

B.8 Relation to Published Evaluations

Tables 1–7 evaluate the adapted algorithms under our common protocol; citations identify the original methods. Published evaluations differ in backbones, task partitions, and adaptation interfaces. MetaSkill-Evolve's Table 1 uses Gemma-4-31B with category-stratified held-out partitions (Wang et al., 2026c); its task identities differ from those in Table 5. Skill-Pro's Table 2 evaluates Qwen3-32B on ALFWorld over 50 episodes per setting (Mi et al., 2026); our endpoint uses the complete native unseen test partition. ACE's AppWorld Table 1 uses DeepSeek-V3.1 and separates offline from online adaptation (Zhang et al., 2026c). EvoSkill uses Opus 4.5, an older OfficeQA release, and different held-out partitions and skill permissions (Alzubi et al., 2026).

GEPA, MCE, and MARS report their primary results on other benchmark suites (Agrawal et al., 2026; Ye et al., 2026; Hou et al., 2026). The original code-editing studies also use different evaluation protocols (Zhang et al., 2026b;a; Wang et al., 2026b); Table 7 evaluates their Terminal-Bench ports on common hardware. All reported table values come from our common-protocol evaluations.

B.9 Preliminary Study Details

B.9.1 Evolution and Frozen-Probe Evaluation

The preliminary study in Section 3 uses three independent trajectories, each with three meta-update stages indexed by $s \in \{0, 1, 2\}$ and four improver checkpoints $P_0, \ldots, P_3$. These short trajectories have their own protocol, separate from the primary twelve-stage experiments. At each stage, the improver is revised using only the most recent evolution record, and candidate revisions are selected solely by their improvement on the current-stage agent. Archived historical states, traces, and scores are excluded from candidate generation and selection.

We archive four naturally occurring modification problems per stage, yielding twelve probes per run. A probe consists of an agent checkpoint S_i, its diagnostic trace τ_i, a scoring set $\mathcal{D}_i$ disjoint from the diagnostic tasks, and a fixed execution configuration E_i. After evolution, every improver checkpoint is evaluated on every probe. For $R = 8$ independent one-edit trials, the response is

$$\begin{aligned} \widehat{A}_{t,i} &= \frac{1}{R} \sum_{r=1}^{R} \left[J\Big(\mathcal{U}(S_i, e_{t,i}^{(r)}); \mathcal{D}_i, E_i\Big) - J(S_i; \mathcal{D}_i, E_i) \right], \\ e_{t,i}^{(r)} &\sim q_\theta(\cdot \mid S_i, \tau_i; P_t). \end{aligned} \tag{B.4}$$

Here q_θ generates an edit under improver instructions P_t, $\mathcal{U}$ applies the edit, and J is the mean normalized task score. Every trial restores the same frozen target state; different improver versions use the same task and environment seed records. Run indices are suppressed in Eq. (B.4) and the definitions below. The design yields $3 \times 4 \times 12 \times 8 = 1{,}152$ single-trial gains.

For a probe collected at stage s_i, the acquisition reference is P_{r_i} with $r_i = s_i + 1$, assigned by stage independently of the response matrix. Figure 2(a) averages responses over the three runs for each probe position and improver version, with equal weight on all eight trials within each run. Columns are ordered by collection stage; outlines identify acquisition references. Bubble area increases with absolute gain, and color represents signed gain. The task-score bars average the three task-agent checkpoints at each version on a fixed historical task set. The response comparison in the main text averages the four earliest probes per run at P_1 and P_3; the task-score comparison uses the same versions.

B.9.2 Historical Contrasts and Repeat Variability

For each probe, we compare every later checkpoint $t > r_i$ with its fixed acquisition reference:

$$\Delta_{\text{hist}}(t, i) = \widehat{A}_{r_i,i} - \widehat{A}_{t,i}, \qquad \ell = t - r_i. \tag{B.5}$$

Positive loss denotes a decline in one-edit gain. In each run, the four stage-0 probes contribute two contrasts each (at $t = 2, 3$), and the four stage-1 probes contribute one each (at $t = 3$). Stage-2 probes have no later checkpoint. Thus panel (b) contains twelve contrasts per run, or 36 in total: 24 at lag one and 12 at lag two. Contrasts share runs, probes, or acquisition references; the independent trajectories number three.

Let π_s be the normalized vector of recorded weights for the six task families in stage s of a run. Since P_t follows stage $t - 1$, task shift for a contrast is

$$d_{\text{JS},t,i} = \sqrt{\text{JS}_2(\pi_{s_i}, \pi_{t-1})}, \qquad \text{JS}_2(p, q) = \tfrac{1}{2}\text{KL}_2(p\|m) + \tfrac{1}{2}\text{KL}_2(q\|m), \quad m = \tfrac{1}{2}(p+q). \tag{B.6}$$

Logarithms use base two, zero-probability terms contribute zero, and $d_{\text{JS},t,i} \in [0, 1]$. Panel (b) uses these continuous distances. The four probes from the same stage share the shift coordinate for a given later checkpoint. Lag and task shift covary in these trajectories, so the display describes their joint association with loss.

To assess repeat variability, we repeat the same frozen improver on the same probes with fresh generation seeds. The 95th percentile of the absolute paired repeat differences across 500 control

records is 2.2 percentage points. This statistic describes repeated response measurements under a fixed improver; it is not a confidence interval for the three-run mean loss.

B.9.3 Aggregation and Rendering

All gains, losses, and repeat differences are stored in normalized-score units and multiplied by 100 for display in percentage points. The 36 historical contrasts are computed from within-run, eight-trial response means before pooling for display. The earliest-probe responses of 14.3 and 5.9 pp and their 8.4-pp difference use equal weights across probes and runs. The corresponding task-score means are 53.6% and 65.5%.

The background in Figure 2(b) is a Gaussian kernel-weighted mean of the observed losses on a 150×150 display grid. Bandwidths are 0.24 for lag and 0.10 for task shift. The field is masked outside the convex hull of observed coordinates and wherever the sum of kernel weights is below 0.15. For coincident points, deterministic horizontal offsets span $[-0.05, 0.05]$ updates, ordered by loss; calculations use the original coordinates. The dashed contour is placed at $\max(5.5, 2.5\nu_{95})$ pp, where ν_{95} is the unrounded repeat variability statistic in percentage points, giving a displayed level of 5.5 pp. Interpolation, offsets, and the contour serve as visual guides; the contour is not a significance threshold. All numerical summaries use the recorded observations.

C Additional Experimental Results

C.1 Headline Paired Uncertainty

Bootstrap construction follows Appendix B.5. Table 3 reports the uncertainty for the headline comparison against Current-only + search. This control shares IFR's task-update interface and resource ceilings.

Table 3: **Paired uncertainty for the headline IFR comparisons.** Δ is IFR minus Current-only + search. Task endpoints and G_{shared} are reported in percentage points. CIs are 95% percentile intervals from 2,000 paired whole-run bootstrap resamples. "Positive runs" counts matched runs with $d_r > 0$.

Endpoint	Benchmark	Mean Δ	95% paired CI	Positive runs
Task performance	ALFWorld unseen	$+2.3$	$[1.7, 2.9]$	$5/5$
	OfficeQA exact	$+2.5$	$[1.6, 3.4]$	$5/5$
	Seal-Hard	$+2.6$	$[1.5, 3.7]$	$5/5$
G_{shared}	ALFWorld	$+0.6$	$[0.3, 0.9]$	$5/5$
	OfficeQA	$+0.5$	$[0.2, 0.8]$	$5/5$
	Seal-Hard	$+0.5$	$[0.2, 0.8]$	$5/5$

Interpretation. The direction of the IFR–control difference is positive in all five matched runs for each of the six headline comparisons. The paired intervals also remain above zero, including for the smaller shared-state gain differences.

These intervals characterize variability across the five paired evolution runs under the evaluated settings.

C.2 Improvement Gains and Retention

Table 4: **Improvement gains and retention across three benchmarks.** Qwen3-32B; five-run means in percentage points (↑). G_{shared} measures one-edit gains on common states; MBWT and MBWT^{+} assess retention over all eligible historical probes and the independently screened positive subset, respectively. Historical targets differ across methods.

	ALFWorld			OfficeQA			Seal-Hard		
Method	G_{shared}	MBWT	MBWT^{+}	G_{shared}	MBWT	MBWT^{+}	G_{shared}	MBWT	MBWT^{+}
Recursive evolution									
MetaSkill-Evolve (Wang et al., 2026c)	8.7	-4.8	-6.1	7.4	-6.0	-7.2	6.1	-5.2	-6.2
MCE (Ye et al., 2026)	8.4	-5.4	-6.7	7.1	-6.6	-8.0	5.8	-5.8	-6.9
Controlled alternatives to functional replay									
Frozen improver P_0	5.9	—	—	4.6	—	—	3.5	—	—
Current-only	9.6	-7.2	-8.6	8.3	-8.5	-10.1	6.8	-7.8	-9.3
Current-only + search	<u>10.2</u>	-8.0	-9.4	<u>9.0</u>	-9.2	-10.8	<u>7.4</u>	-8.5	-10.0
Task-score preservation	8.8	-3.9	-4.7	7.6	-4.6	-5.6	6.0	-4.1	-5.0
Historical-edit imitation	9.1	-3.1	-3.9	7.9	-3.5	-4.4	6.3	-3.3	-4.1
Full-history replay	9.7	$\mathbf{-0.8}$	$\mathbf{-1.1}$	8.5	$\mathbf{-0.9}$	$\mathbf{-1.2}$	6.9	$\mathbf{-0.7}$	$\mathbf{-1.0}$
IFR (ours; $B = 8$)	**10.8**	<u>-1.1</u>	<u>-1.5</u>	**9.5**	<u>-1.3</u>	<u>-1.7</u>	**7.9**	<u>-1.0</u>	<u>-1.4</u>

Notes for Table 4. Bold and underlining mark the highest and second-highest numerical entries in each column. The retention metrics use each method's own historical targets. MBWT uses fixed acquisition references (Eq. 8) and retains signed gains. MBWT^{+} reproduces independent confirmation from Table 8, which includes positive-reference gains, screening coverage, and task BWT. For frozen P_0, — denotes the uninformative MBWT self-difference (zero by definition); positive-reference confirmation is unreported. Task-score preservation is the CPE-inspired control

defined in Appendix B.2 (Yu et al., 2026). Full-history replay retains all probes under the same resource ceilings; consumed budgets appear in Table 12.

C.3 Backbone Robustness

Table 5 uses each backbone for all generative roles during evolution and evaluation. IFR leads every endpoint on both models. The strongest alternative is Current-only + search, except for ACE on AppWorld Challenge and MetaSkill-Evolve on Gemma's ALFWorld.

Table 5: **Task performance across backbones.** Percentages (↑), with the same splits, tools, and resource ceilings as Table 1. ALF/Office/Seal: ALFWorld unseen success/OfficeQA exact accuracy/Seal-Hard accuracy. AppWorld: TGC; NC/C: no-curated/curated SkillsBench, each with 13 held-out tasks and three trials per task per run. Bold: column best; —: outside the fixed coverage.

(a) Qwen3-8B

	ALF	Office	Seal	AppWorld		SkillsBench	
Method	Unseen	Exact	Hard	Normal	Chall.	NC	C
GEPA (2026)	44.6	37.9	20.4	42.6	24.5	27.2	36.9
MetaSkill-Evolve (2026c)	48.7	42.5	23.5	45.4	27.0	30.3	40.0
ACE (2026c)	—	—	—	47.1	29.1	—	—
Skill-Pro (2026)	46.9	—	—	—	—	—	—
Frozen improver P_0	41.5	34.3	17.0	38.9	21.6	23.6	32.3
Current-only	50.0	43.4	24.4	46.5	27.9	30.8	41.5
Current-only + search	51.3	44.7	25.4	47.9	29.0	31.8	42.1
IFR (ours)	**53.4**	**47.2**	**27.7**	**50.2**	**31.6**	**34.4**	**44.6**

(b) Gemma-4-31B-IT

	ALF	Office	Seal	AppWorld		SkillsBench	
Method	Unseen	Exact	Hard	Normal	Chall.	NC	C
GEPA (2026)	89.4	48.9	35.2	52.6	34.0	38.5	49.7
MetaSkill-Evolve (2026c)	94.0	55.0	44.2	56.3	37.1	42.1	53.3
ACE (2026c)	—	—	—	58.1	39.7	—	—
Skill-Pro (2026)	91.6	—	—	—	—	—	—
Frozen improver P_0	90.7	45.7	30.5	48.9	29.6	35.4	46.2
Current-only	93.3	56.1	44.7	57.4	38.4	43.1	54.4
Current-only + search	93.7	57.2	45.4	58.5	39.6	44.1	55.4
IFR (ours)	**95.2**	**59.5**	**47.8**	**60.7**	**42.1**	**46.2**	**57.9**

C.4 Additional Native Endpoints

IFR leads all eight native endpoints in Table 6. On OfficeQA Pro V2, its 47.8% exact accuracy exceeds MetaSkill-Evolve by 5.3 points. Its AppWorld SGC gains over ACE are 2.5 and 3.0 points on Normal and Challenge. Pro uses held-out Full questions; Pro V2 tests frozen corpus transfer. Seal-0 retains frozen retrieval, and SGC uses the official scenario grouping. These endpoints are reported separately.

Table 6: **Expanded native metrics with Qwen3-32B.** All entries are percentages (↑). OfficeQA's 1% and 5% columns denote allowable numerical relative error; Pro and Pro V2 use exact agreement. Seen/unseen ALFWorld, AppWorld TGC/SGC, and overlapping QA subsets remain separate endpoints. Pro V2 uses 90 questions per run; Normal SGC uses 56 scenarios per run. Bold marks the highest score; —: outside the prespecified coverage.

(a) Numerical tolerance, subset evaluation, and corpus transfer

	OfficeQA Full		OfficeQA	OfficeQA	SealQA
Method	1%	5%	Pro	Pro V2	Seal-0
ReAct (2023)	39.7	44.8	31.6	26.7	11.7
Reflexion (2023)	40.8	45.9	33.1	27.8	12.4
MIPROv2 (2024)	46.3	51.2	39.4	32.9	14.7
GEPA (2026)	50.7	55.6	43.1	36.7	17.1
MARS (2026)	48.6	53.4	41.6	34.9	16.0
EvoSkill (2026)	52.1	56.9	45.0	38.4	18.4
MetaSkill-Evolve (2026c)	56.6	61.3	49.2	42.4	21.7
MCE (2026)	55.5	60.1	48.1	41.6	20.9
IFR (ours)	**62.5**	**67.2**	**55.1**	**47.8**	**25.4**

(b) Seen environments and scenario-level completion

	ALFWorld	AppWorld: SGC	
Method	Seen success	Normal	Challenge
ReAct (2023)	57.9	29.6	14.7
Reflexion (2023)	50.6	30.7	15.5
MIPROv2 (2024)	62.1	32.5	17.3
GEPA (2026)	65.4	35.0	19.4
ACE (2026c)	—	40.7	23.7
Skill-Pro (2026)	67.1	—	—
MetaSkill-Evolve (2026c)	69.6	37.9	22.0
MCE (2026)	68.9	37.5	21.6
IFR (ours)	**75.1**	**43.2**	**26.8**

C.5 CODE-EDITING EXTENSION

Table 7 compares the code-editing ports using Qwen3-32B, the common sandbox and resource ceilings, and eight A100 GPUs. IFR-code reaches 31.9% success, 2.9 points above HGM, with lower token, episode, allocated A100-hour, and wall-time consumption than HGM and DGM. Hyperagents has the lowest consumption but reaches 23.8% success. The cost columns report realized consumption under the common ceilings.

Table 7: **Code-editing comparison on Terminal-Bench 2.0.** Success uses 14 held-out tasks with three trials per task in each of five runs, rounding once after aggregating unrounded outcomes. Optimization cost includes all branches, proposals, and retries. Allocated A100-h counts all eight GPUs reserved during optimization; evaluation cost is recorded separately. Bold marks the highest success rate.

	Task result	Actual optimization consumption			
Method	Success (%)	Tokens (M)	Episodes	Allocated A100-h	Wall-h
Hyperagents (2026b)	23.8	27.6	8,840	138.4	17.3
DGM (2026a)	26.7	29.3	9,640	156.8	19.6
HGM (2026b)	29.0	28.9	9,410	152.0	19.0
IFR-code (ours)	**31.9**	28.1	9,050	143.2	17.9

C.6 MECHANISMS, AUDITS, AND RESOURCE USE

This subsection details the mechanism comparison in Table 2 and reports independent confirmation, functional coverage, sensitivity, task order, and realized cost. Entries are descriptive aggregate means under the five-run protocol; — denotes an inapplicable quantity. These tables provide no run-level observations or confidence intervals.

Anchor variants, sensitivity settings, and the order comparison use Qwen3-32B on ALFWorld only. Functional audits additionally cover AppWorld Normal and both SkillsBench access conditions with the same backbone. Tables 1 and 5 report task-performance coverage for the other benchmark/backbone pairs. Native task evaluation, shared-bank evaluation, and historical audits keep their distinct task identities and trial counts.

C.6.1 RETENTION TARGETS AND ANCHOR RULES

Retention target and deployment outcomes. Among the ALFWorld controls in Table 2, fixed-anchor IFR leads task success, shared-state gain, and deployed-update gain. Task-score preservation has the least negative task BWT, while historical-maximum anchors and full-history replay have MBWT closer to zero on their own historical targets. The preferred mechanism thus depends on the endpoint (protocols in Appendix C.6).

The adjacent-anchor variant uses each probe's cached incumbent response as its selection anchor. The historical-maximum variant uses the largest cached response among installed checkpoints available before the current selection, excluding future checkpoints and audit outcomes. If its constraint admits no candidate, it retains the incumbent and records the infeasible stage. Both variants retain IFR's other settings; independent audits use fixed acquisition references. The fixed-anchor feasibility theorem is restricted to IFR's fixed-anchor selector.

Table 2 uses ✓ and × for the presence and absence of each retention mechanism. The task-score guard acts on the task agent, edit imitation supplies past accepted edits as demonstrations, and functional replay evaluates improver responses on stored probes. Anchor and B apply only to functional replay; All retains every arriving probe (at most 12). Current-only + search reallocates the replay allowance to additional current-state proposals and evaluations. All other configuration details follow Appendix B.

Controls and fixed-anchor IFR use the same runs as the main comparison. The two alternative anchor rules use separate trajectories. All historical audits keep $r_i = s_i + 1$. G_{deploy} is the held-out

score change per attempted task update in the committed loop: rejected updates contribute zero; accepted updates compare installed and parent states on independent task-seed pairs. Acceptance rates and gains conditional on acceptance are retained separately. Bold and underlining mark the largest and second-largest values in each outcome column. Historical MBWT describes retention on each method's own trajectory.

Interpretation. Task BWT compares the final task agent on earlier blocks with its checkpoint immediately after learning each block. Positive-reference confirmation distinguishes loss of previously positive improvement ability from changes on unhelpful probes. The anchor comparison evaluates how the selection rule affects these outcomes.

C.6.2 INDEPENDENT POSITIVE-REFERENCE CONFIRMATION

Table 8: **Independent confirmation on the three primary functional benchmarks.** Qwen3-32B, five runs. $\Delta G = G_{\text{shared}}(P_{\text{final}}) - G_{\text{shared}}(P_0)$ uses the common state bank. G^{+}_{ref} and MBWT^{+} use the independently screened positive subset; gains and BWT are in percentage points. Coverage is the fraction of eligible historical probes passing screening.

(a) ALFWorld

Method	ΔG ↑	G^{+}_{ref} (pp)	MBWT^{+} ↑	Coverage (%)	Task BWT ↑
MetaSkill-Evolve	2.8	10.6	−6.1	72.7	−4.4
MCE	2.5	10.3	−6.7	70.9	−4.8
Current-only	3.7	11.0	−8.6	74.5	−5.4
Current-only + search	4.3	11.4	−9.4	76.4	−5.8
Task-score preservation	2.9	9.7	−4.7	69.1	−1.2
Historical-edit imitation	3.2	10.0	−3.9	70.9	−2.7
Full-history replay	3.8	10.4	−1.1	72.7	−1.6
IFR (fixed; $B = 8$)	4.9	11.2	−1.5	74.5	−1.8

(b) OfficeQA Full

Method	ΔG ↑	G^{+}_{ref} (pp)	MBWT^{+} ↑	Coverage (%)	Task BWT ↑
MetaSkill-Evolve	2.8	9.3	−7.2	70.9	−4.8
MCE	2.5	9.0	−8.0	69.1	−5.1
Current-only	3.7	9.8	−10.1	72.7	−5.9
Current-only + search	4.4	10.1	−10.8	74.5	−6.2
Task-score preservation	3.0	8.5	−5.6	67.3	−1.4
Historical-edit imitation	3.3	8.8	−4.4	69.1	−2.9
Full-history replay	3.9	9.2	−1.2	70.9	−1.8
IFR (fixed; $B = 8$)	4.9	9.9	−1.7	72.7	−2.0

(c) Seal-Hard

Method	ΔG ↑	G^{+}_{ref} (pp)	MBWT^{+} ↑	Coverage (%)	Task BWT ↑
MetaSkill-Evolve	2.6	7.8	−6.2	69.1	−4.5
MCE	2.3	7.5	−6.9	67.3	−4.9
Current-only	3.3	8.2	−9.3	70.9	−5.6
Current-only + search	3.9	8.6	−10.0	72.7	−6.0
Task-score preservation	2.5	7.1	−5.0	65.5	−1.3
Historical-edit imitation	2.8	7.4	−4.1	67.3	−2.6
Full-history replay	3.4	7.8	−1.0	69.1	−1.7
IFR (fixed; $B = 8$)	4.4	8.4	−1.4	70.9	−1.9

Screening uses $R_{\text{screen}} = 8$ trials; confirmation uses $R_{\text{audit}} = 8$ fresh trials, independent of optimization and screening. The audit protocol specifies per-probe paired intervals, completion fractions, and full response matrices. Lost positive ability requires confirmation of a positive reference gain and a negative contrast. Coverage has no preferred direction; historical targets differ across methods. An empty screened subset is marked unavailable.

C.6.3 Additional Functional Coverage

Table 9: **Functional evaluation beyond the three primary benchmarks.** Qwen3-32B, five paired runs. Gains and MBWT are in percentage points. AppWorld uses Normal only. Each SkillsBench access condition has its own common state bank and fixed task/skill-access protocol.

(a) AppWorld Normal

Method	G_{shared} ↑	MBWT ↑	MBWT^{+} ↑	G^{+}_{ref} (pp)	Coverage (%)
Frozen improver P_0	4.8	—	—	—	—
MetaSkill-Evolve	7.6	−5.6	−6.8	9.0	72.7
MCE	7.3	−6.1	−7.4	8.7	70.9
Current-only	8.4	−7.9	−9.3	9.4	74.5
Current-only + search	8.9	−8.6	−10.0	9.8	76.4
Task-score preservation	7.7	−4.1	−5.0	8.5	69.1
Historical-edit imitation	8.0	−3.3	−4.0	8.8	70.9
Full-history replay	8.4	−0.9	−1.2	9.2	72.7
IFR (fixed; $B = 8$)	9.4	−1.2	−1.6	9.7	74.5

(b) SkillsBench: no curated skills

Method	G_{shared} ↑	MBWT ↑	MBWT^{+} ↑	G^{+}_{ref} (pp)	Coverage (%)
Frozen improver P_0	4.2	—	—	—	—
MetaSkill-Evolve	6.9	−4.9	−6.0	8.1	69.1
MCE	6.6	−5.4	−6.6	7.8	67.3
Current-only	7.7	−7.0	−8.4	8.5	72.7
Current-only + search	8.2	−7.7	−9.1	8.9	74.5
Task-score preservation	6.9	−3.6	−4.5	7.6	65.5
Historical-edit imitation	7.2	−2.9	−3.6	7.9	67.3
Full-history replay	7.7	−0.7	−1.0	8.3	69.1
IFR (fixed; $B = 8$)	8.8	−1.0	−1.4	8.8	72.7

(c) SkillsBench: curated skills

Method	G_{shared} ↑	MBWT ↑	MBWT^{+} ↑	G^{+}_{ref} (pp)	Coverage (%)
Frozen improver P_0	3.9	—	—	—	—
MetaSkill-Evolve	6.5	−4.6	−5.6	7.6	67.3
MCE	6.2	−5.0	−6.1	7.3	65.5
Current-only	7.4	−6.6	−8.0	8.0	70.9
Current-only + search	7.9	−7.2	−8.6	8.4	72.7
Task-score preservation	6.6	−3.3	−4.1	7.1	63.6
Historical-edit imitation	6.9	−2.6	−3.3	7.4	65.5
Full-history replay	7.4	−0.7	−0.9	7.8	67.3
IFR (fixed; $B = 8$)	8.4	−0.9	−1.2	8.3	70.9

These post-evolution audits reuse saved states and the screening protocol of Table 8. The audit specification also includes ΔG, task BWT, accepted-update gains, fixed-P_0 repeat variability, and uncertainty. Frozen P_0 supplies a shared-gain baseline; its self-difference is zero by definition.

C.6.4 SENSITIVITY AND TASK ORDER

Sensitivity of the gain–retention balance. In the one-factor ALFWorld sweeps (Table 10), increasing B from 4 to 16 shifts MBWT from -2.2 to -0.8 pp and task success from 68.5% to 66.9%; relaxing ϵ from 0 to 0.05 shifts success from 66.7% to 68.7% and MBWT from -0.5 to -2.6 pp. The default balances gain and retention; the table also covers update interval and document size.

Table 10: **One-factor sensitivity on ALFWorld.** Qwen3-32B, five paired runs. The default is $B = 8$, $\epsilon = 0.02$, $H = 3$, and 4,096 tokens per persistent document. Each other row changes only the named factor under the same resource ceilings. Gains and BWT are in percentage points. Cost columns report realized consumption.

Setting	Task (%) ↑	G_{shared} ↑	MBWT ↑	Task BWT ↑	Tokens (M)	Episodes	Stages
Default	68.1	10.8	-1.1	-1.8	22.8	4,980	12.0
$B = 4$	68.5	10.9	-2.2	-2.3	18.5	3,660	12.0
$B = 16$	66.9	9.7	-0.8	-1.6	24.6	5,340	12.0
$\epsilon = 0$	66.7	10.0	-0.5	-1.5	22.7	4,990	12.0
$\epsilon = 0.01$	67.5	10.4	-0.8	-1.7	22.8	4,985	12.0
$\epsilon = 0.05$	68.7	11.0	-2.6	-2.4	22.7	4,960	12.0
$H = 1$	65.7	10.4	-1.0	-1.4	21.9	4,830	12.0
$H = 6$	68.8	10.7	-1.3	-2.3	24.1	5,200	12.0
Doc. limit = 2,048	66.6	10.1	-1.2	-1.9	20.6	4,970	12.0
Doc. limit = 8,192	68.4	10.9	-1.2	-1.9	25.9	4,990	12.0

The default row is shared across all four sweeps. With at most 12 probe arrivals, $B = 16$ reuses the full-history replay runs with identical configurations and random streams. Stages gives the five-run mean; the reporting protocol also specifies per-run counts and ranges. Positive-reference confirmation and deployment gains use the same audit protocol as the default.

Table 11: **Task-order control on ALFWorld.** Qwen3-32B and five paired runs. Blocked and shuffled conditions use the same multiset of development tasks; only category ordering changes. All other settings follow the default protocol.

(a) Category-blocked schedule

Method	Task (%) ↑	G_{shared} ↑	MBWT ↑	Task BWT ↑	Tokens (M)	Episodes	Stages
Current-only + search	65.8	10.2	-8.0	-5.8	22.4	4,890	12.0
IFR (fixed; $B = 8$)	68.1	10.8	-1.1	-1.8	22.8	4,980	12.0

(b) Shuffled schedule

Method	Task (%) ↑	G_{shared} ↑	MBWT ↑	Task BWT ↑	Tokens (M)	Episodes	Stages
Current-only + search	66.3	10.1	-6.2	-4.4	22.5	4,905	12.0
IFR (fixed; $B = 8$)	67.9	10.7	-1.0	-1.7	22.9	4,995	12.0

The blocked condition uses the main-comparison runs; the two shuffled rows use separate trajectories. Paired run-level records are required to assess uncertainty in the IFR–control and shuffled–blocked differences.

C.6.5 REALIZED RESOURCE CONSUMPTION

Table 12: **Actual optimization cost for the functional comparison.** Qwen3-32B; means over five runs. Tokens include input and output; episodes include all executed development tasks. Stages counts complete committed stages. Replay and archive columns give peak resident probe storage and checkpoint/response archive storage in MiB, respectively. They are distinct from accelerator memory.

(a) ALFWorld

Method	Tokens (M)	Episodes	Stages	Wall (h)	Cache hit (%)	Replay (MiB)	Archive (MiB)
MetaSkill-Evolve	16.1	2,780	12.0	9.2	2.5	0.0	14.6
MCE	18.0	3,240	12.0	10.5	3.4	0.0	15.9
Current-only	9.1	1,020	12.0	5.1	0.8	0.0	8.8
Current-only + search	22.4	4,890	12.0	13.1	2.1	0.0	18.4
Task-score preservation	15.0	2,910	12.0	8.6	8.4	0.0	14.3
Historical-edit imitation	10.8	1,380	12.0	6.1	2.9	0.0	11.9
Full-history replay	24.6	5,340	12.0	14.4	24.6	9.0	35.8
IFR: adjacent anchors	22.9	4,970	12.0	13.6	23.1	6.0	32.0
IFR: historical maxima	23.4	5,060	12.0	13.9	24.9	6.0	33.6
IFR (fixed; $B = 8$)	22.8	4,980	12.0	13.5	23.4	6.0	32.2

(b) OfficeQA Full

Method	Tokens (M)	Episodes	Stages	Wall (h)	Cache hit (%)	Replay (MiB)	Archive (MiB)
MetaSkill-Evolve	18.3	2,830	12.0	10.8	2.7	0.0	15.8
MCE	20.1	3,290	12.0	11.9	3.6	0.0	17.4
Current-only	10.3	1,040	12.0	6.2	0.9	0.0	9.5
Current-only + search	24.5	4,940	12.0	14.6	2.2	0.0	21.1
Task-score preservation	16.9	2,950	12.0	10.0	8.7	0.0	16.0
Historical-edit imitation	12.1	1,410	12.0	7.2	3.1	0.0	13.0
Full-history replay	27.0	5,390	12.0	16.0	24.9	11.4	43.2
IFR (fixed; $B = 8$)	25.0	5,030	12.0	14.8	23.7	7.6	38.7

(c) Seal-Hard

Method	Tokens (M)	Episodes	Stages	Wall (h)	Cache hit (%)	Replay (MiB)	Archive (MiB)
MetaSkill-Evolve	17.1	2,800	12.0	10.2	2.6	0.0	15.2
MCE	18.9	3,260	12.0	11.2	3.5	0.0	16.6
Current-only	9.6	1,030	12.0	5.8	0.8	0.0	9.1
Current-only + search	23.6	4,920	12.0	14.0	2.2	0.0	19.7
Task-score preservation	15.9	2,930	12.0	9.4	8.5	0.0	15.2
Historical-edit imitation	11.4	1,400	12.0	6.8	3.0	0.0	12.4
Full-history replay	25.9	5,370	12.0	15.3	24.8	10.3	39.6
IFR (fixed; $B = 8$)	23.9	5,010	12.0	14.2	23.5	6.9	35.4

All branches, proposals, replay evaluations, failures, and retries are charged. Allocated A100-hours equal eight times wall hours under the reserved-machine convention. The reporting protocol separates per-run stage counts, stopping reasons, failure categories, and audit/final-evaluation costs, including SealQA judge API costs. Comparisons align realized consumption and completed stages.

C.7 Historical Drift Audit

C.7.1 Fixed References and Historical Drift

Figure 6 compares three anchor rules on ALFWorld with Qwen3-32B: five paired 12-stage runs and 1,320 held-out contrasts per method, audited against fixed acquisition checkpoints independently of selection anchors (Appendix C.7.2).

From lag 1 to 11, median deficits rise from 1.24 to 6.65 pp with adjacent anchors and 0.84 to 2.17 pp with fixed anchors, whose median exceeds the 2-pp reference at lags 10 and 11. Historical-max anchors have the lowest median at every lag (0.41 to 1.34 pp).

Historical-max anchors yield lower task success and shared-state gain: 66.4% and 9.9 pp versus 68.1% and 10.8 pp with fixed anchors (Table 2), illustrating the gain–retention trade-off. Historical targets differ across methods. The 2-pp training tolerance bounds average cached shortfall on retained probes (Section 4.5); this audit measures individual deficits.

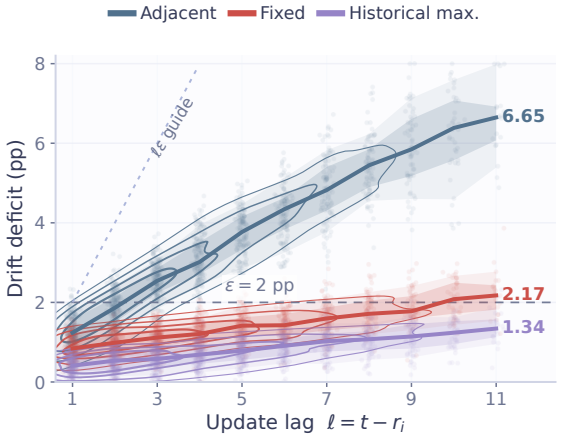

Figure 6: **Historical drift on ALFWorld.** Lines show medians; dark/light bands show the 25–75/10–90 percentiles. Dashed: 2-pp tolerance reference; dotted: $\ell\epsilon$ guide. Audit and density details are in Appendix C.7.2.

C.7.2 Audit Protocol and Detailed Results

Protocol and metric. The audit in Appendix C.7.1 compares adjacent, historical-maximum, and fixed admission anchors on ALFWorld with Qwen3-32B, using five paired runs and 12 completed stages per run. Each saved stage-entry task-agent state yields four audit probes with disjoint held-out diagnostic/scoring task subsets. For a probe z_i from stage s_i, the acquisition reference is $r_i = s_i + 1$. Following Appendix B.4.1, let $\widehat{A}_{t,i}$ denote the independent audit estimate of checkpoint P_t's one-edit response on z_i. We define

$$\widehat{\delta}_{t,i} = \widehat{A}_{r_i,i} - \widehat{A}_{t,i}, \qquad \ell_{t,i} = t - r_i > 0. \tag{C.1}$$

Positive deficits indicate loss of historical improvement ability; values are displayed in percentage points (pp). All three methods use fixed acquisition references for this audit, regardless of their selection anchors. The complete positive-lag grid contains $4\sum_{r=1}^{11}(12 - r) = 264$ contrasts per method per run, or 1,320 per method and 3,960 overall. Each method contributes $20(12-\ell)$ contrasts at lag ℓ, decreasing from 220 at lag 1 to 20 at lag 11.

Figure construction. Figure 6 pools observations across the five runs. Solid curves show the median at each integer lag; darker and lighter ribbons show the interquartile and 10–90 percentile ranges. Faint points use small horizontal offsets for visibility. Contours use a two-dimensional Gaussian kernel density estimate of the displayed lag–deficit coordinates, with bandwidth factor 0.24, normalized to unit peak within each method; levels are 0.18, 0.34, 0.54, and 0.74 of that peak. Contour levels therefore describe relative density within each method. The horizontal dashed line marks $\epsilon = 0.02$ (2 pp); the oblique $\ell\epsilon$ guide is a visual reference for linear accumulation.

Interpretation. All three median trajectories increase with lag, with historical-max anchors below fixed anchors and fixed anchors below adjacent anchors at every lag (Table 13). Thus, fixed anchors attenuate the observed lag dependence without eliminating it. Their median exceeds 2 pp at lags 10 and 11; at lag 11, 13 of 20 contrasts exceed this reference. The constraint in Section 4.5 bounds average cached shortfall on the retained buffer; this audit measures independent held-out deficits. Historical targets differ across methods, and longer lags contain fewer, earlier-acquired probes. Each lag therefore summarizes its available probe cohort. Percentile bands describe within-lag spread; the five complete runs remain the independent units for uncertainty.

Table 13: **Lag-wise historical drift.** Pooled median [10th, 90th percentile] in pp. n is the number of contrasts per method; repeated contrasts remain nested within five runs.

Lag	n	Adjacent anchors	Fixed anchors	Historical maxima
1	220	1.24 [0.69, 1.77]	0.84 [0.46, 1.23]	0.41 [0.12, 0.68]
2	200	1.86 [1.18, 2.43]	0.99 [0.62, 1.34]	0.54 [0.14, 0.86]
3	180	2.54 [1.83, 3.07]	1.12 [0.70, 1.49]	0.58 [0.17, 0.87]
4	160	3.02 [2.43, 3.95]	1.21 [0.83, 1.61]	0.69 [0.24, 1.11]
5	140	3.77 [2.86, 4.47]	1.41 [0.96, 1.92]	0.79 [0.35, 1.12]
6	120	4.34 [3.24, 5.13]	1.43 [1.03, 1.89]	0.91 [0.47, 1.23]
7	100	4.82 [3.77, 5.83]	1.60 [0.99, 2.09]	1.03 [0.45, 1.47]
8	80	5.45 [4.55, 6.23]	1.71 [1.20, 2.26]	1.07 [0.56, 1.48]
9	60	5.85 [4.60, 7.13]	1.78 [1.30, 2.41]	1.14 [0.62, 1.64]
10	40	6.38 [4.59, 7.59]	2.08 [1.43, 2.67]	1.24 [0.80, 1.62]
11	20	6.65 [5.39, 7.99]	2.17 [1.54, 2.77]	1.34 [0.95, 1.88]

C.8 Lag, Task Shift, and Forgetting

We audit current-only evolution on ALFWorld, OfficeQA, and Seal-Hard using 6,318 historical probe–checkpoint contrasts from five runs per benchmark. Response loss is relative to each probe's acquisition checkpoint. Update lag $\ell = t - r_i$ and continuous task-family shift $d_{\mathrm{JS}} = \sqrt{\mathrm{JS}_2}$ define a 6×7 grid of integer lags and predefined shift bins. All 42 cells have pooled coverage; coverage within every run is not required (Appendix C.8.1).

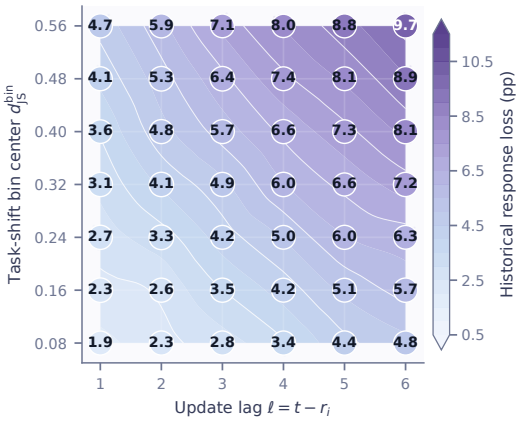

Figure 7: **Historical response loss across update lag and task-shift bins.** Labeled circles show pooled cell means (pp) at predefined shift-bin centers, with area increasing with sample count. Contours interpolate cell means; white lines mark equal loss. Faint points show individual binned contrasts with display jitter.

Pooled cell means increase with either factor at every fixed level of the other. Endpoint differences, averaged equally over the other factor's levels, are 4.06 pp for lag and 4.10 pp for shift. From lag 1 to 6, loss increases by 5.03 pp in the highest-shift bin versus 2.96 pp in the lowest, a 2.07-pp difference-in-differences. Benchmark-specific summaries follow the same direction (Appendix C.8.1). These are conditional associations over the sampled ranges.

Task-order control. With the ALFWorld development-task multiset fixed (Table 11), Current-only + search has MBWT of -8.0 pp under blocked order and -6.2 under shuffled order, versus -1.1 and -1.0 for IFR. IFR gains 0.6 pp in shared-state improvement under both schedules and 2.3/1.6 pp in task success, respectively.

Implication. Auditing both probe age and task distribution helps identify where improvement strategies lose effectiveness and where replay is needed.

C.8.1 Audit Protocol and Detailed Results

Protocol and coverage. The audit uses current-only evolution on ALFWorld, OfficeQA, and Seal-Hard, with five runs per benchmark. Following Appendix B.4.1, historical probes restore saved stage-entry task-agent states and use frozen diagnostic/scoring tasks. A probe from stage s_i has acquisition reference $r_i = s_i + 1$. Each checkpoint response is estimated with $R_{\mathrm{audit}} = 8$ fresh

one-edit trials. For empirical task-family distributions π_r of the corresponding stage blocks, define

$$\Delta_{\mathrm{hist}}(t,i) = \widehat{A}_{r_i,i} - \widehat{A}_{t,i}, \qquad \ell = t - r_i, \qquad d_{\mathrm{JS},t,i} = \sqrt{\mathrm{JS}_2(\pi_{r_i}, \pi_t)}, \tag{C.2}$$

where JS_2 uses base-2 logarithms and losses are displayed in percentage points. Here d_{JS} denotes the continuous task shift. For conditional aggregation and visualization, individual contrasts are assigned to one of seven predefined task-shift bins, displayed by their centers

$$d_{\mathrm{JS}}^{\mathrm{bin}} \in \{0.08, 0.16, 0.24, 0.32, 0.40, 0.48, 0.56\}.$$

Together with

$$\ell \in \{1, \ldots, 6\},$$

this yields a 6×7 pooled descriptive grid.

Coverage is assessed on the pooled contrasts. A 12-stage run contributes the checkpoint pairs available at each lag, and its occupied task-shift bins depend on the realized stage schedule; consequently, an individual run need not populate all 42 cells. Pooling the five runs per benchmark across the three benchmarks yields complete coverage of all 42 cells. ALFWorld, OfficeQA, and Seal-Hard contribute 2,115, 2,131, and 2,072 contrasts, respectively, or 6,318 overall. Pooled cell counts range from 134 to 161, so the grid is complete with unequal cell counts.

Across all 6,318 individual contrasts, the pooled Pearson correlation between update lag ℓ and the continuous raw shift d_{JS} is 0.001.

Table 14: **Lag–shift bin means.** Historical response loss in percentage points, pooled over benchmarks and runs. Rows correspond to predefined bins of the continuous raw task shift $d_{\mathrm{JS}} = \sqrt{\mathrm{JS}_2}$; row labels give the displayed bin centers $d_{\mathrm{JS}}^{\mathrm{bin}}$. Each cell aggregates 134–161 contrasts.

Shift-bin center $d_{\mathrm{JS}}^{\mathrm{bin}}$	$\ell = 1$	$\ell = 2$	$\ell = 3$	$\ell = 4$	$\ell = 5$	$\ell = 6$
0.08	1.85	2.32	2.80	3.44	4.36	4.81
0.16	2.32	2.63	3.49	4.23	5.12	5.68
0.24	2.74	3.29	4.24	5.05	5.97	6.28
0.32	3.14	4.08	4.89	5.97	6.64	7.24
0.40	3.56	4.80	5.72	6.62	7.35	8.14
0.48	4.13	5.28	6.37	7.38	8.09	8.95
0.56	4.67	5.86	7.15	8.03	8.78	9.70

Aggregation and rendering. Each individual contrast is assigned to a lag–shift cell before aggregation, and each cell mean weights its recorded contrasts equally. Benchmarks and runs therefore contribute in proportion to their observed cell counts. No resampling, duplication, or response-dependent assignment is used to equalize cell occupancy. Figure 7 uses cubic interpolation of the 42 observed cell means on a 260×260 display grid. All numerical comparisons use the observed cell means in Table 14. After bin assignment and aggregation, horizontal and vertical display jitter separates overlapping individual contrasts. Marker area increases with cell count. Figure labels and Table 14 round the same unrounded means to one and two decimal places, respectively. Colors encode response loss.

Comparisons and interpretation. In the $d_{\mathrm{JS}}^{\mathrm{bin}} = 0.24$ shift bin, increasing lag from 1 to 6 raises mean loss from 2.74 to 6.28 pp. At $\ell = 2$, moving from the lowest-shift bin ($d_{\mathrm{JS}}^{\mathrm{bin}} = 0.08$) to the highest-shift bin ($d_{\mathrm{JS}}^{\mathrm{bin}} = 0.56$) raises mean loss from 2.32 to 5.86 pp, a 3.54-pp increase. Let $m_{\ell, d_{\mathrm{JS}}^{\mathrm{bin}}}$ be a cell mean. The 4.06-pp lag increase averages $m_{6,d_{\mathrm{JS}}^{\mathrm{bin}}} - m_{1,d_{\mathrm{JS}}^{\mathrm{bin}}}$ equally over the seven shift bins; the 4.10-pp shift increase averages $m_{\ell,0.56} - m_{\ell,0.08}$ equally over the six lags. The corner interaction is $(m_{6,0.56} - m_{1,0.56}) - (m_{6,0.08} - m_{1,0.08}) = 2.07$ pp. Adjacent-cell mixed differences are positive in 21 of 30 rectangles and negative in nine. Within ALFWorld, OfficeQA, and Seal-Hard, the mean endpoint increases are respectively 4.08/4.18, 4.07/4.11, and 4.01/4.02 pp for lag/shift.

Scope. Near-zero correlation between lag and the raw task shift d_{JS} reduces collinearity in this audit bank, while other probe differences remain uncontrolled. Contrasts are nested within runs, and the comparisons describe observational associations. Effect magnitudes depend on the sampled lag range and predefined task-shift bins; slopes and coefficients use different units. These descriptive comparisons do not test statistical significance, equivalence, or causal dominance.

D Theoretical Analysis and Proofs

D.1 Definitions and Probability-Space Construction

Let $Q_i = P_{a_i}$ be the immutable admission reference of probe z_i, and let $Y_i(P;\omega) \in [0,1]$ denote its edited-agent score for a complete trial record ω. Expectations are defined for each fixed complete manifest before a data-dependent selected manifest is substituted. Expectations average editing and execution randomness with the probe and scoring tasks fixed. Given the pre-block history $\mathcal{F}_i$, freeze the target, diagnostic evidence, scoring tasks, execution configuration, anchor manifest Q_i, and trial count R_i. The seed record contains every random input to the prescribed evaluation. A fresh generic record has the same conditional law as each $\omega_{i,r}$. With μ_i^0 denoting the common expected parent score and $\widehat{\mu}_i^0$ its shared cached estimate, define

$$\begin{aligned}
\mu_i(P) &= \mathbb{E}[Y_i(P;\omega) \mid \mathcal{F}_i], \\
G(P;z_i) &= \mu_i(P) - \mu_i^0, \qquad d_i(P) = \mu_i(Q_i) - \mu_i(P), \\
\widehat{G}(P;z_i) &= \frac{1}{R_i}\sum_{r=1}^{R_i} Y_i(P;\omega_{i,r}) - \widehat{\mu}_i^0, \\
\widehat{d}_i(P) &= \widehat{G}(Q_i;z_i) - \widehat{G}(P;z_i) \\
&= \frac{1}{R_i}\sum_{r=1}^{R_i}\big[Y_i(Q_i;\omega_{i,r}) - Y_i(P;\omega_{i,r})\big].
\end{aligned} \tag{D.1}$$

Thus $d_i(P), \widehat{d}_i(P) \in [-1,1]$; concentrating the two gains separately would discard their exact shared-baseline cancellation. For nonempty $\mathcal{B}$, the population and cached losses are

$$L_{\mathcal{B}}(P) = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}}[d_i(P)]_+, \qquad \widehat{L}_{\mathcal{B}}(P) = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}}[\widehat{d}_i(P)]_+. \tag{D.2}$$

Both are defined as zero for an empty buffer. The cached loss $\widehat{L}_{\mathcal{B}_t}(P)$ is exactly $\widehat{L}_t(P)$ in Eq. (5). Probe-specific trial counts R_i include the method's common-count case $R_i = R$.

Assumption 1 (Predictable reservation and immutable reuse)**.** *Index every attempted reserved trial block before any of its outcomes are revealed, including attempts in stages that subsequently fail to complete. Before block i, its frozen probe, reference Q_i, and positive trial count R_i are $\mathcal{F}_i$-measurable. Conditional on $\mathcal{F}_i$, the R_i trial records are iid with the distribution defining μ_i. All trials are used in a completed response; there is no outcome-dependent within-block stopping or replacement of unsuccessful draws. All randomness affecting a score is included in its record, and each trial restores the frozen state. Identical complete manifests use the same score function and immutable cache entry. Completed evaluations equal the prescribed potential outcomes; infrastructure failures may censor a response but must not replace its draws by new, outcome-selected draws.*

All possible manifests belong to a fixed countable set $\mathcal{P}$. For each block, a positive probability mass function ν_i on $\mathcal{P}$ may depend on $\mathcal{F}_i$, but is fixed before that block is exposed. Fix deterministic $w_i > 0$ with $\sum_i w_i \le 1$ over all potential attempted-block indices. Subsequent candidate generation, cache queries, stopping, and buffer choices may be adaptive.

The assumption permits independently reserved randomness to be drawn in advance only if it remains unused and unexposed until the reference and prior have been fixed. Merely fixing seed values does not establish this independence. Manifests include every persistent state component affecting responses. The prior is defined on the entire manifest universe.

Attempt indices are assigned before exposure, never retrospectively after successful completion. A query of a previously created cache uses its original block index and original prior. After termination, extend the probability space with unused dummy blocks having fixed probes and fresh independent records; these blocks never enter the buffer. Finite runs with a predetermined bound N need only N potential indices. For each index, all hypothetical scores $Y_i(P;\omega_{i,r})$, $P \in \mathcal{P}$, are defined on the same block whether or not they are executed. Missing infrastructure responses censor these prescribed outcomes; the reserved draws remain fixed. A confidence event covering every attempted block can therefore be restricted afterward to any completed stages without conditioning on successful completion.

For later use, with $0 \log 0 = 0$ and the usual extended-value conventions, set

$$\begin{aligned} \mathrm{kl}(p\|q) &= p \log \frac{p}{q} + (1-p) \log \frac{1-p}{1-q}, \\ u(p,h) &= \max\{q \in [p,1] : \mathrm{kl}(p\|q) \le h\}, \\ \Psi(e,h) &= 2u\left(\frac{1+e}{2}, h\right) - 1. \end{aligned} \tag{D.3}$$

Here $p, q, e \in [0,1]$ and $h \ge 0$.

D.2 Complete Retention Guarantees

The sampling correction in Theorem 1 uses $h_i(P) = R_i^{-1} \log(1/[\delta w_i \nu_i(P)])$ and $r_{\mathcal{B}}(P) = |\mathcal{B}|^{-1} \sum_{i \in \mathcal{B}} \min\{1, \sqrt{2h_i(P)}\}$ for nonempty $\mathcal{B}$, with $r_{\varnothing}(P) = 0$. Here R_i is the reserved trial count, ν_i is the pre-block prior over all possible complete improver states (manifests), and the positive weights for attempted trial blocks satisfy $\sum_i w_i \le 1$. References and priors are fixed before trial outcomes are revealed, as specified in Assumption 1.

The following result gives a per-probe inverse-KL certificate, aggregate relaxations, and a finite-class specialization. Theorem 1 is its additive consequence.

Theorem 2 (Fixed-anchor retention under adaptive reuse)**.** *Assume immutable cached responses and fixed admission anchors. Initialize $\mathcal{B}_0 = \varnothing$, fix $\epsilon \ge 0$, and include the incumbent in every finite candidate set. At each completed stage, select P_{t+1} with $\widehat{L}_{\mathcal{B}_t}(P_{t+1}) \le \epsilon$. Leave the buffer unchanged, append one probe, or replace exactly one entry; a new probe uses P_{t+1} and its identical cached response as its fixed reference and anchor. Then the incumbent is always feasible, and each of $\widehat{L}_{\mathcal{B}_t}(P_t)$, $\widehat{L}_{\mathcal{B}_t}(P_{t+1})$, and $\widehat{L}_{\mathcal{B}_{t+1}}(P_{t+1})$ is at most ϵ. This deterministic assertion does not require Assumption 1's probabilistic conditions.*

Under Assumption 1, fix $\delta \in (0,1)$ and put $h_i(P) = R_i^{-1} \log(1/[\delta w_i \nu_i(P)])$ and $\widehat{p}_i(P) = (1 + \widehat{d}_i(P))/2$. With probability at least $1 - \delta$, simultaneously at every completed stage, each of the preceding buffer–checkpoint pairs satisfies $L_{\mathcal{B}}(P) \le \mathcal{C}_{\mathcal{B}}(P)$, where $\mathcal{C}_{\mathcal{B}}(P) = n^{-1} \sum_{i \in \mathcal{J}} [2u(\widehat{p}_i(P), h_i(P)) - 1]_+$, $n = |\mathcal{B}|$, and $\mathcal{J} = \{i \in \mathcal{B} : P \ne Q_i\}$. For $m = |\mathcal{J}| > 0$, write $\vartheta = m/n$, $\overline{h} = m^{-1} \sum_{i \in \mathcal{J}} h_i(P)$, and $e_\epsilon = \min\{1, \epsilon/\vartheta\}$. The certificate is at most $\vartheta \Psi(e_\epsilon, \overline{h})$, and also at most $\min\{\vartheta, \epsilon + n^{-1} \sum_{i \in \mathcal{J}} \min\{1, \sqrt{2h_i(P)}\}\}$. If the buffer or $\mathcal{J}$ is empty, both the loss and certificate are exactly zero, and the corresponding ratios are unused.

Corollary 1 (Finite-class specialization)**.** *If there are at most N attempted reserved blocks, the predetermined manifest universe has size K, and $R_i \ge R$, take $w_i = 1/N$ and $\nu_i(P) = 1/K$. With $c = \log(NK/\delta)$ and $e = \min\{1, \epsilon\}$, Theorem 2 gives $L_{\mathcal{B}}(P) \le \Psi(e, c/R) \le \min\{1, e + \sqrt{2c/R}\}$ simultaneously for its pairs. For $0 \le \epsilon < \epsilon + \gamma < 1$, it suffices that $R \ge c/\,\mathrm{kl}((1+\epsilon)/2 \| (1+\epsilon+\gamma)/2)$ to guarantee $L_{\mathcal{B}}(P) \le \epsilon + \gamma$. When $\epsilon = 0$, the KL envelope equals $\sqrt{1 - \exp(-2c/R)}$.*

Interpretation and tightness. The non-accumulation claim concerns the retention tolerance: no $T\epsilon$ term appears. Statistical uncertainty can still increase with the probe index or description complexity $-\log \nu_i(P)$; for an unbounded stream, $w_i = 1/[i(i+1)]$ is one valid choice. The original certificate preserves negative empirical margins and probe-specific uncertainty and should be preferred to its coarser relaxations. Appendix D.8 gives a variance-sensitive alternative with an explicitly allocated confidence budget. Appendix D.9 shows that predecessor-relative constraints can permit a sharp $\min\{1, T\epsilon\}$ degradation, and Appendix D.10 establishes sharpness only of the single-comparison KL exponent. The retention guarantee applies to retained probes and admission-time ability. Evicted probes, unseen tasks, and the deployed agent's multi-edit trajectory lie outside its scope.

D.3 A Conditional One-Sided KL Bound

Lemma 1 (Conditional inverse-KL confidence)**.** *Conditional on a sigma-field $\mathcal{F}$, let $X_1, \ldots, X_R$ be independent, identically distributed random variables in $[0,1]$ with conditional mean p. The positive integer R is $\mathcal{F}$-measurable. For $\overline{X} = R^{-1} \sum_r X_r$ and any $\mathcal{F}$-measurable $c > 0$,*

$$\mathbb{P}\{p > u(\overline{X}, c/R) \mid \mathcal{F}\} \le e^{-c}. \tag{D.4}$$

Proof. All calculations below are conditional on $\mathcal{F}$; hence p, R, c are fixed. For $\lambda < 0$, convexity of the exponential and $0 \le X_r \le 1$ imply

$$e^{\lambda X_r} \le (1 - X_r) + X_r e^{\lambda}, \qquad \mathbb{E}[e^{\lambda X_r} \mid \mathcal{F}] \le 1 - p + pe^{\lambda}. \tag{D.5}$$

For $0 < x < p < 1$, conditional independence and Markov's inequality give

$$\begin{aligned} \mathbb{P}(\overline{X} \le x \mid \mathcal{F}) &= \mathbb{P}\Big(e^{\lambda \sum_r X_r} \ge e^{\lambda R x} \mid \mathcal{F}\Big) \\ &\le e^{-\lambda R x} \mathbb{E}\Big[e^{\lambda \sum_r X_r} \mid \mathcal{F}\Big] \\ &= e^{-\lambda R x} \prod_{r=1}^{R} \mathbb{E}[e^{\lambda X_r} \mid \mathcal{F}] \\ &\le \exp\big\{R\big[-\lambda x + \log(1 - p + pe^{\lambda})\big]\big\}. \end{aligned} \tag{D.6}$$

The minimizing value is

$$\lambda_\star = \log \frac{x(1-p)}{p(1-x)} < 0, \qquad -\lambda_\star x + \log(1 - p + pe^{\lambda_\star}) = -\operatorname{kl}(x\|p). \tag{D.7}$$

Taking the infimum in (D.6), and then taking limits for $x = 0$, yields

$$\mathbb{P}(\overline{X} \le x \mid \mathcal{F}) \le e^{-R \operatorname{kl}(x\|p)}, \qquad 0 \le x < p < 1. \tag{D.8}$$

For fixed $p \in (0, 1)$, the map $x \mapsto \operatorname{kl}(x\|p)$ is strictly decreasing on $[0, p]$, as follows from

$$\frac{\partial}{\partial x} \operatorname{kl}(x\|p) = \log \frac{x(1-p)}{p(1-x)} < 0, \qquad 0 < x < p. \tag{D.9}$$

Set $h = c/R$. Since $u(x, h) \ge x$, a failure $p > u(\overline{X}, h)$ requires both $\overline{X} < p$ and $\operatorname{kl}(\overline{X}\|p) > h$. If $h \ge \operatorname{kl}(0\|p)$, such a failure is impossible. Otherwise, there is a unique $x_h \in (0, p)$ with $\operatorname{kl}(x_h\|p) = h$, and

$$\begin{aligned} \mathbb{P}\{p > u(\overline{X}, h) \mid \mathcal{F}\} &\le \mathbb{P}\{\overline{X} < p, \operatorname{kl}(\overline{X}\|p) > h \mid \mathcal{F}\} \\ &= \mathbb{P}(\overline{X} < x_h \mid \mathcal{F}) \\ &\le \mathbb{P}(\overline{X} \le x_h \mid \mathcal{F}) \\ &\le e^{-R \operatorname{kl}(x_h\|p)} = e^{-c}. \end{aligned} \tag{D.10}$$

For $p = 0$ the asserted event is empty. For $p = 1$, boundedness implies $X_r = 1$ almost surely, so $u(\overline{X}, h) = 1$ almost surely. This proves the lemma, including the boundary cases. □

D.4 Geometry of the Inverse-KL Envelope

Lemma 2 (Monotonicity, concavity, and quadratic relaxation)**.** *The map $u(p, h)$ is nondecreasing in each argument and jointly concave on $[0, 1] \times [0, \infty)$. Furthermore,*

$$0 \le u(p, h) - p \le \min\{1 - p, \sqrt{h/2}\}, \qquad e \le \Psi(e, h) \le \min\{1, e + \sqrt{2h}\}. \tag{D.11}$$

Proof. The feasible set defining $u(p, h)$ is nonempty because it contains p. It is closed in the compact interval $[p, 1]$, by lower semicontinuity of extended binary relative entropy. Its supremum is therefore attained. Monotonicity in h follows from inclusion of feasible sets. To prove monotonicity in p, fix $p_1 \le p_2$ and let $q = u(p_1, h)$. If $q \le p_2$, then $u(p_2, h) \ge p_2 \ge q$. Otherwise, the derivative in (D.9), with second argument q, gives

$$\operatorname{kl}(p_2\|q) \le \operatorname{kl}(p_1\|q) \le h, \qquad u(p_2, h) \ge q = u(p_1, h). \tag{D.12}$$

Boundary cases follow by continuity and the extended-value convention.

We next establish the joint convexity used in the concavity argument. For nonnegative weights λ_j summing to one, write $\bar{p} = \sum_j \lambda_j p_j$ and $\bar{q} = \sum_j \lambda_j q_j$. Convexity of $\phi(x) = x \log x$ gives

$$\begin{aligned} \bar{p} \log \frac{\bar{p}}{\bar{q}} &= \bar{q}\, \phi\left(\sum_j \frac{\lambda_j q_j}{\bar{q}} \frac{p_j}{q_j}\right) \\ &\le \bar{q} \sum_j \frac{\lambda_j q_j}{\bar{q}} \phi\left(\frac{p_j}{q_j}\right) = \sum_j \lambda_j p_j \log \frac{p_j}{q_j}. \end{aligned} \tag{D.13}$$

Applying the same argument to the complementary probabilities yields

$$\mathrm{kl}(\bar{p}\|\bar{q}) \le \sum_j \lambda_j \,\mathrm{kl}(p_j\|q_j). \tag{D.14}$$

The formulas extend to zero probabilities by limits; cases with an infinite right-hand side are immediate. Choose $q_j = u(p_j, h_j)$. Feasibility, followed by (D.14), implies

$$\begin{aligned} \bar{q} &\ge \bar{p}, \\ \mathrm{kl}(\bar{p}\|\bar{q}) &\le \sum_j \lambda_j \,\mathrm{kl}(p_j\|q_j) \le \sum_j \lambda_j h_j, \\ \sum_j \lambda_j u(p_j, h_j) = \bar{q} &\le u\left(\sum_j \lambda_j p_j, \sum_j \lambda_j h_j\right). \end{aligned} \tag{D.15}$$

This is joint concavity.

Finally, fix $0 < q < 1$ and set $f(x) = \mathrm{kl}(x\|q)$. Since $f(q) = f'(q) = 0$ and $f''(x) = 1/[x(1-x)] \ge 4$, Taylor's integral remainder gives, for $p \le q$,

$$\begin{aligned} \mathrm{kl}(p\|q) &= \int_p^q (s-p) f''(s)\, ds \\ &= \int_p^q \frac{s-p}{s(1-s)}\, ds \ge 4\int_p^q (s-p)\, ds = 2(q-p)^2. \end{aligned} \tag{D.16}$$

Taking $q = u(p, h)$ and using its feasibility proves $2(u(p, h) - p)^2 \le h$. The range bound $u(p, h) \le 1$ gives the other term in the minimum. Limits handle endpoints. Substituting $p = (1 + e)/2$ proves the assertion for Ψ. □

D.5 Uniform Validity under Adaptive Cache Queries

Lemma 3 (One event for all subsequent adaptive choices)**.** *Under Assumption 1, with probability at least $1 - \delta$, simultaneously for every attempted block i and every $P \in \mathcal{P}$,*

$$\frac{1 + d_i(P)}{2} \le u\left(\frac{1 + \widehat{d}_i(P)}{2}, h_i(P)\right). \tag{D.17}$$

When $P = Q_i$, both $d_i(P)$ and $\widehat{d}_i(P)$ are exactly zero.

Proof. Fix an attempted block i and a predetermined manifest P. Conditional on $\mathcal{F}_i$, its anchor Q_i and prior ν_i are fixed. Define

$$X_{i,r}(P) = \frac{1 + Y_i(Q_i; \omega_{i,r}) - Y_i(P; \omega_{i,r})}{2}, \qquad \mathbb{E}[X_{i,r}(P) \mid \mathcal{F}_i] = \frac{1 + d_i(P)}{2}. \tag{D.18}$$

These variables are conditionally iid in $[0, 1]$ across trials. No independence is required between the two scores within one trial. Their empirical mean is $\widehat{p}_i(P) = (1 + \widehat{d}_i(P))/2$. Define

$$c_i(P) = \log \frac{1}{\delta w_i \nu_i(P)}, \qquad h_i(P) = \frac{c_i(P)}{R_i}. \tag{D.19}$$

Since $c_i(P)$ is $\mathcal{F}_i$-measurable, Lemma 1 bounds the conditional probability of the failure event $E_{i,P}$ by $\delta w_i \nu_i(P)$. The countable union bound, tower property, and Tonelli's theorem give

$$\begin{aligned}\mathbb{P}\left(\bigcup_i \bigcup_{P\in\mathcal{P}} E_{i,P}\right) &\le \sum_i \sum_{P\in\mathcal{P}} \mathbb{P}(E_{i,P}) \\ &= \sum_i \sum_{P\in\mathcal{P}} \mathbb{E}[\mathbb{P}(E_{i,P} \mid \mathcal{F}_i)] \\ &\le \delta \sum_i w_i \sum_{P\in\mathcal{P}} \mathbb{E}[\nu_i(P)] \\ &= \delta \sum_i w_i \,\mathbb{E}\left[\sum_{P\in\mathcal{P}} \nu_i(P)\right] \\ &= \delta \sum_i w_i \le \delta.\end{aligned} \tag{D.20}$$

In particular, the prior may encode a history-dependent preference for short edits of Q_i. It cannot be chosen using the current block. No separate union over anchors is needed because Q_i is fixed when conditioning on $\mathcal{F}_i$. For a manifest never actually queried, its score is understood as the potential evaluation of the fixed score function on the same seed records. This is a probability-space construction and requires no additional executions. The resulting event quantifies over the entire class before a later checkpoint is selected. Consequently, substituting a data-dependent checkpoint, revisiting a cached response, choosing a data-dependent retained set, or stopping at a data-dependent stage does not require another union bound. Finally, identical manifests use the same score function and cache entry, so their differences vanish pathwise. □

D.6 Empirical Feasibility and Sharp Deterministic Transfer

Lemma 4 (Feasibility invariant)**.** *Under the buffer and cache rules of Theorem 2, the feasible set is nonempty and every completed stage satisfies*

$$\max\left\{\widehat{L}_{\mathcal{B}_t}(P_t), \widehat{L}_{\mathcal{B}_t}(P_{t+1}), \widehat{L}_{\mathcal{B}_{t+1}}(P_{t+1})\right\} \le \epsilon. \tag{D.21}$$

Proof. At initialization, the empty-buffer loss is zero. Suppose $\widehat{L}_{\mathcal{B}_t}(P_t) \le \epsilon$. The incumbent's manifest, cached evaluations, and retained anchors are unchanged, so the incumbent is a feasible candidate. The finite constrained selection therefore returns an improver with $\widehat{L}_{\mathcal{B}_t}(P_{t+1}) \le \epsilon$. Set $n = |\mathcal{B}_t|$, $\widehat{d}_i^+ = [\widehat{d}_i(P_{t+1})]_+$, and $D = \sum_{i\in\mathcal{B}_t} \widehat{d}_i^+$. Thus $0 \le D \le n\epsilon$. A newly admitted probe has the exact same cache entry for its anchor and for P_{t+1}, hence contributes zero. For $n > 0$,

$$\widehat{L}_{\mathcal{B}_{t+1}}(P_{t+1}) = \begin{cases} D/n \le \epsilon, & \text{if the buffer is unchanged,} \\ D/(n+1) \le n\epsilon/(n+1) \le \epsilon, & \text{if one entry is appended,} \\ (D - \widehat{d}_j^+)/n \le D/n \le \epsilon, & \text{if entry } j \text{ is replaced.} \end{cases} \tag{D.22}$$

If $n = 0$, leaving the buffer empty or appending a zero-deficit entry gives loss zero. Induction proves all three inequalities. For $\epsilon = 0$, every cached retained deficit is zero. For positive ϵ, an individual cached deficit can be as large as $\min\{1, |\mathcal{B}_t|\epsilon\}$; a bound on the mean does not imply coordinatewise retention. Gains acquired after admission do not change the fixed reference, and eviction ends the corresponding constraint. Deleting an entry without replacement can increase the mean loss by reducing its denominator and is excluded from the buffer rules above. □

Lemma 5 (Sharp transfer of one-sided contrast errors)**.** *Let $n \ge 1$, $\epsilon \ge 0$, and $d_i, \widehat{d}_i \in [-1, 1]$ for $i = 1, \ldots, n$, and suppose $d_i - \widehat{d}_i \le \rho_i$ with $\rho_i \ge 0$ on an active set $\mathcal{J}$ of size m. Suppose both gaps are zero outside $\mathcal{J}$. If $\widehat{L} = n^{-1}\sum_i [\widehat{d}_i]_+$, then*

$$\begin{aligned}\frac{1}{n}\sum_i [d_i]_+ &\le \frac{1}{n}\sum_{i\in\mathcal{J}} \min\{1, [\widehat{d}_i + \rho_i]_+\} \\ &\le \min\left\{\frac{m}{n}, \widehat{L} + \frac{1}{n}\sum_{i\in\mathcal{J}} \min\{1, \rho_i\}\right\}.\end{aligned} \tag{D.23}$$

Given only these constraints and $\widehat{L} \le \epsilon$, the final bound with $\widehat{L}$ replaced by ϵ is attainable.

Proof. Monotonicity and subadditivity of the positive part yield

$$\begin{aligned}[d_i]_+ &\le \min\{1, [\widehat{d}_i + \rho_i]_+\} \\ &\le \min\{1, [\widehat{d}_i]_+ + \rho_i\} \\ &\le [\widehat{d}_i]_+ + \min\{1, \rho_i\}.\end{aligned} \tag{D.24}$$

The last step follows separately from $\rho_i \le 1$, when the right-hand side is $[\widehat{d}_i]_+ + \rho_i$, and $\rho_i > 1$, when it is at least one. Summing this chain gives the second term in (D.23); summing $[d_i]_+ \le 1$ over the m active coordinates gives the first.

To prove sharpness, let $b_i = \min\{1, \rho_i\}$ and choose $x_i \in [0, 1 - b_i]$ on $\mathcal{J}$ such that

$$\sum_{i \in \mathcal{J}} x_i = \min\left\{ n\epsilon, m - \sum_{i \in \mathcal{J}} b_i \right\}. \tag{D.25}$$

Such a choice exists because the sum of these intervals is precisely $[0, m - \sum_i b_i]$. Set $\widehat{d}_i = x_i$ and $d_i = x_i + b_i$ on $\mathcal{J}$, and set both gaps to zero elsewhere. Then

$$\begin{aligned} 0 \le \widehat{d}_i \le d_i \le 1, &\qquad d_i - \widehat{d}_i = b_i \le \rho_i, \\ \widehat{L} &= \frac{1}{n} \sum_{i \in \mathcal{J}} x_i \le \epsilon, \\ \frac{1}{n} \sum_i [d_i]_+ &= \frac{1}{n} \left(\sum_{i \in \mathcal{J}} b_i + \min\left\{ n\epsilon, m - \sum_{i \in \mathcal{J}} b_i \right\} \right) \\ &= \min\left\{ \frac{m}{n}, \epsilon + \frac{1}{n} \sum_{i \in \mathcal{J}} b_i \right\}. \end{aligned} \tag{D.26}$$

This construction establishes deterministic sharpness under the stated constraints. □

D.7 Proofs of the Retention Guarantees

Proof of Theorem 2. Lemma 4 proves the deterministic assertion. Work on the single event of Lemma 3, whose probability is at least $1 - \delta$. Fix any of the stated buffer–checkpoint pairs and set $n = |\mathcal{B}|$. Write $\mathcal{J} = \{i \in \mathcal{B} : P \ne Q_i\}$, $m = |\mathcal{J}|$, $\vartheta = m/n$, $\overline{h} = m^{-1} \sum_{i \in \mathcal{J}} h_i(P)$, and $e_\epsilon = \min\{1, \epsilon/\vartheta\}$, assuming $n, m > 0$. For $i \in \mathcal{J}$, write $u_i = u(\widehat{p}_i(P), h_i(P))$ and $\rho_i = 2(u_i - \widehat{p}_i(P)) \ge 0$. Baseline cancellation, the uniform event, and monotonicity imply

$$\begin{aligned} d_i(P) = \mu_i(Q_i) - \mu_i(P) \le 2u_i - 1 = \widehat{d}_i(P) + \rho_i, \\ [d_i(P)]_+ \le [2u_i - 1]_+ = \min\{1, [\widehat{d}_i(P) + \rho_i]_+\}. \end{aligned} \tag{D.27}$$

The final equality uses $u_i \le 1$. Coordinates outside $\mathcal{J}$ vanish exactly. Summing therefore proves $L_{\mathcal{B}}(P) \le \mathcal{C}_{\mathcal{B}}(P)$.

For the nonlinear relaxation, put $\widehat{d}_i^+ = [\widehat{d}_i(P)]_+$ and $p_i^+ = (1 + \widehat{d}_i^+)/2$. Since $p_i^+ \ge \widehat{p}_i(P)$ and $u(p_i^+, h_i) \ge p_i^+ \ge 1/2$, Lemma 2 gives the following chain, retaining the empirical mean until its

last step:

$$\begin{aligned}
\mathcal{C}_{\mathcal{B}}(P) &= \frac{1}{n}\sum_{i\in\mathcal{J}}[2u(\widehat{p}_i(P), h_i(P)) - 1]_+ \\
&\le \frac{1}{n}\sum_{i\in\mathcal{J}}[2u(p_i^+, h_i(P)) - 1] \\
&= \vartheta\left[\frac{2}{m}\sum_{i\in\mathcal{J}} u(p_i^+, h_i(P)) - 1\right] \\
&\le \vartheta\left[2u\left(\frac{1}{m}\sum_{i\in\mathcal{J}} p_i^+, \frac{1}{m}\sum_{i\in\mathcal{J}} h_i(P)\right) - 1\right] \\
&= \vartheta\Psi\left(\frac{\widehat{L}_{\mathcal{B}}(P)}{\vartheta}, \overline{h}\right) \\
&\le \vartheta\Psi\big(\min\{1, \epsilon/\vartheta\}, \overline{h}\big)\,.
\end{aligned} \tag{D.28}$$

Here $\widehat{L}_{\mathcal{B}}(P) \le \vartheta$ follows from the unit range of the m active deficits; the last step also uses the empirical feasibility invariant. This argument does not assume independence across retained probes.

For the additive relaxation, apply the coordinate argument of Lemma 5 to the certificate itself and use Lemma 2:

$$\begin{aligned}
\mathcal{C}_{\mathcal{B}}(P) &\le \min\left\{\vartheta, \widehat{L}_{\mathcal{B}}(P) + \frac{1}{n}\sum_{i\in\mathcal{J}}\min\{1, \rho_i\}\right\} \\
&\le \min\left\{\vartheta, \epsilon + \frac{1}{n}\sum_{i\in\mathcal{J}}\min\{1, 2\sqrt{h_i(P)/2}\}\right\} \\
&= \min\left\{\vartheta, \epsilon + \frac{1}{n}\sum_{i\in\mathcal{J}}\min\{1, \sqrt{2h_i(P)}\}\right\}.
\end{aligned} \tag{D.29}$$

Combining (D.28) and (D.29) proves the stated bound, explicitly

$$L_{\mathcal{B}}(P) \le \mathcal{C}_{\mathcal{B}}(P) \le \min\left\{\vartheta\Psi(e_\epsilon, \overline{h}), \epsilon + \frac{1}{n}\sum_{i\in\mathcal{J}}\min\{1, \sqrt{2h_i(P)}\}\right\}. \tag{D.30}$$

The first expression already has range at most ϑ. The cases $n = 0$ and $m = 0$ follow from the definitions and exact identity of the relevant manifests.

The proof compares P directly with each immutable Q_i. It never telescopes comparisons between successive improvers. The confidence event already holds for every manifest and attempted block, so applying it at additional stages introduces neither another failure probability nor an additional retention-tolerance term. □

Proof of Theorem 1. The deterministic statement is contained in Theorem 2. On its simultaneous confidence event, consider any of the stated pairs with $n = |\mathcal{B}| > 0$. Let $\mathcal{J} = \{i \in \mathcal{B} : P \ne Q_i\}$ and $\vartheta = |\mathcal{J}|/n$. Since $\mathcal{J} \subseteq \mathcal{B}$ and $0 \le \vartheta \le 1$, its additive bound implies

$$\begin{aligned}
L_{\mathcal{B}}(P) &\le \min\left\{\vartheta, \epsilon + \frac{1}{n}\sum_{i\in\mathcal{J}}\min\{1, \sqrt{2h_i(P)}\}\right\} \\
&\le \min\left\{1, \epsilon + \frac{1}{n}\sum_{i\in\mathcal{B}}\min\{1, \sqrt{2h_i(P)}\}\right\} \\
&= \min\{1, \epsilon + r_{\mathcal{B}}(P)\}.
\end{aligned} \tag{D.31}$$

For an empty buffer, $L_{\mathcal{B}}(P) = r_{\mathcal{B}}(P) = 0$. This is a relaxation on the same event, so no additional confidence allocation is required. □

Proof of Corollary 1. Let $h_\star = \log(NK/\delta)/R$. For every active coordinate, $h_i(P) \le h_\star$. Include inactive coordinates with artificial parameters $\widetilde{h}_i = 0$ and $p_i^+ = 1/2$; their contribution $2u(1/2, 0) - 1$ is zero. On active coordinates set $\widetilde{h}_i = h_i(P)$ and $p_i^+ = (1+[\widehat{d}_i(P)]_+)/2$. The same monotonicity and concavity argument now over all n coordinates gives

$$\begin{aligned} L_{\mathcal{B}}(P) \le \mathcal{C}_{\mathcal{B}}(P) &\le \frac{1}{n}\sum_{i\in\mathcal{B}}[2u(p_i^+, \widetilde{h}_i) - 1] \\ &\le 2u\left(\frac{1+\widehat{L}_{\mathcal{B}}(P)}{2}, \frac{1}{n}\sum_{i\in\mathcal{B}}\widetilde{h}_i\right) - 1 \\ &\le \Psi(\min\{1,\epsilon\}, h_\star) \\ &\le \min\{1, \epsilon + \sqrt{2h_\star}\}. \end{aligned} \tag{D.32}$$

Empty buffers have zero loss. For $0 \le \epsilon < \epsilon + \gamma < 1$, let $p_\epsilon = (1+\epsilon)/2$ and $q_{\epsilon,\gamma} = (1+\epsilon+\gamma)/2$. For fixed $p < 1$, the map $q \mapsto \mathrm{kl}(p\|q)$ is strictly increasing on $(p, 1)$ because

$$\frac{\partial}{\partial q}\,\mathrm{kl}(p\|q) = \frac{q-p}{q(1-q)} > 0. \tag{D.33}$$

Consequently,

$$\begin{aligned} R \ge \frac{\log(NK/\delta)}{\mathrm{kl}(p_\epsilon\|q_{\epsilon,\gamma})} &\Longrightarrow h_\star \le \mathrm{kl}(p_\epsilon\|q_{\epsilon,\gamma}) \\ &\Longrightarrow u(p_\epsilon, h_\star) \le q_{\epsilon,\gamma} \\ &\Longrightarrow L_{\mathcal{B}}(P) \le 2u(p_\epsilon, h_\star) - 1 \le \epsilon + \gamma. \end{aligned} \tag{D.34}$$

This sample-size threshold exactly inverts the KL envelope. The simpler sufficient condition $R \ge 2\log(NK/\delta)/\gamma^2$ follows from Lemma 2. For zero tolerance, direct inversion gives

$$\begin{aligned} &\mathrm{kl}(1/2\|q) = -\frac{1}{2}\log\big(4q(1-q)\big), \\ &u(1/2, h) = \frac{1+\sqrt{1-e^{-2h}}}{2}, \qquad\qquad \Psi(0,h) = \sqrt{1-e^{-2h}}, \\ &R \ge \frac{2\log(NK/\delta)}{-\log(1-\gamma^2)} \quad\Longrightarrow\quad L_{\mathcal{B}}(P) \le \gamma. \end{aligned} \tag{D.35}$$

No additional probability union is taken. □

D.8 A Variance-Sensitive Complementary Certificate

This section gives a separate confidence statement that can exploit a small variance of paired differences. Its confidence allocation must be fixed before inspecting the relevant blocks. The construction uses the ordinary sample variance and accounts for dependence between overlapping pairs.

Lemma 6 (Variance upper bound by random matching)**.** *Conditionally on $\mathcal{F}$, let $X_1, \dots, X_R \in [0,1]$ be iid, with mean p, variance σ^2, and an $\mathcal{F}$-measurable integer $R \ge 2$. Set*

$$\overline{X} = \frac{1}{R}\sum_{r=1}^{R} X_r, \qquad S^2 = \frac{1}{R-1}\sum_{r=1}^{R}(X_r - \overline{X})^2, \qquad k = \lfloor R/2 \rfloor. \tag{D.36}$$

For any $\mathcal{F}$-measurable $c_v > 0$, with conditional probability at least $1 - e^{-c_v}$,

$$\sigma^2 \le v^+ := \min\left\{\frac{1}{4}, \frac{1}{2}u\Big(2S^2, \frac{c_v}{k}\Big)\right\}. \tag{D.37}$$

Proof. All arguments condition on $\mathcal{F}$. Expanding squares gives

$$\begin{aligned}\sum_{r<s}(X_r - X_s)^2 &= R\sum_r X_r^2 - \left(\sum_r X_r\right)^2 \\ &= R\sum_r (X_r - \overline{X})^2,\end{aligned} \tag{D.38}$$
$$V := \binom{R}{2}^{-1}\sum_{r<s}(X_r - X_s)^2 = 2S^2 \in [0,1].$$

Introduce an auxiliary random matching Π, independent of the data, containing k disjoint unordered pairs; when R is odd, one index is left unmatched. Choose Π uniformly. By symmetry, each of the $\binom{R}{2}$ possible pairs appears with probability $k/\binom{R}{2}$. Therefore, for $V_\Pi = k^{-1}\sum_{(r,s)\in\Pi}(X_r - X_s)^2$,

$$\mathbb{E}_\Pi[V_\Pi \mid X_1, \dots, X_R] = \frac{1}{k}\sum_{r<s}\frac{k}{\binom{R}{2}}(X_r - X_s)^2 = V. \tag{D.39}$$

For disjoint indices, independence gives $\mathbb{E}[(X_r - X_s)^2 \mid \mathcal{F}] = 2\sigma^2$. Convexity, conditional independence within each fixed matching, and the exponential secant bound imply, for $\lambda < 0$,

$$\begin{aligned}\mathbb{E}[e^{\lambda k V} \mid \mathcal{F}] &= \mathbb{E}\Big[e^{\lambda k \mathbb{E}_\Pi[V_\Pi \mid X_1,\dots,X_R]} \mid \mathcal{F}\Big] \\ &\le \mathbb{E}_\Pi \mathbb{E}[e^{\lambda k V_\Pi} \mid \mathcal{F}, \Pi] \\ &= \mathbb{E}_\Pi \prod_{(r,s)\in\Pi} \mathbb{E}[e^{\lambda (X_r - X_s)^2} \mid \mathcal{F}] \\ &\le \mathbb{E}_\Pi \prod_{(r,s)\in\Pi} [1 - 2\sigma^2 + 2\sigma^2 e^\lambda] \\ &= [1 - 2\sigma^2 + 2\sigma^2 e^\lambda]^k.\end{aligned} \tag{D.40}$$

Thus, for $0 \le x < 2\sigma^2$, the same Markov optimization and one-sided inversion as in Lemma 1 give

$$\begin{aligned}\mathbb{P}(V \le x \mid \mathcal{F}) &\le \inf_{\lambda<0} e^{-\lambda k x}[1 - 2\sigma^2 + 2\sigma^2 e^\lambda]^k = e^{-k\,\mathrm{kl}(x\|2\sigma^2)}, \\ \mathbb{P}\{2\sigma^2 > u(V, c_v/k) \mid \mathcal{F}\} &\le e^{-c_v}.\end{aligned} \tag{D.41}$$

The zero-variance case is immediate. Finally, boundedness yields

$$\sigma^2 = \mathbb{E}[X_1^2 \mid \mathcal{F}] - p^2 \le p - p^2 = \frac{1}{4} - (p - 1/2)^2 \le \frac{1}{4}. \tag{D.42}$$

Combining the deterministic range bound with (D.41) proves the claim. □

Lemma 7 (Bennett inversion and Bernstein relaxation)**.** *Under the iid assumptions of Lemma 6, define*

$$H(x) = (1+x)\log(1+x) - x, \quad x \ge 0, \qquad b(v,h) = \begin{cases} vH^{-1}(h/v), & v > 0, \\ 0, & v = 0. \end{cases} \tag{D.43}$$

For any $\mathcal{F}$-measurable $c_m > 0$, with conditional probability at least $1 - e^{-c_m}$, one has $p \le \overline{X} + b(\sigma^2, c_m/R)$. The function $b(v,h)$ is nondecreasing in v and satisfies

$$b(v,h) \le \sqrt{2vh} + \frac{h}{3}, \qquad v, h \ge 0. \tag{D.44}$$

Proof. Condition on $\mathcal{F}$. Put $Z_r = p - X_r$. Then $\mathbb{E}Z_r = 0$, $|Z_r| \le 1$, and $\mathbb{E}Z_r^2 = \sigma^2$. The exponential series converges absolutely, so, for $\lambda \ge 0$,

$$\begin{aligned}\mathbb{E}e^{\lambda Z_r} &= 1 + \sum_{j=2}^{\infty} \frac{\lambda^j \mathbb{E}Z_r^j}{j!} \\ &\le 1 + \sum_{j=2}^{\infty} \frac{\lambda^j \mathbb{E}|Z_r|^j}{j!} \\ &\le 1 + \sigma^2 \sum_{j=2}^{\infty} \frac{\lambda^j}{j!} \\ &= 1 + \sigma^2(e^\lambda - 1 - \lambda) \\ &\le \exp\{\sigma^2(e^\lambda - 1 - \lambda)\}.\end{aligned} \tag{D.45}$$

If $\sigma^2 > 0$, independence and Markov's inequality yield, for $t > 0$,

$$\begin{aligned}\mathbb{P}(p - \overline{X} \ge t \mid \mathcal{F}) &\le \inf_{\lambda > 0} e^{-R\lambda t} \prod_{r=1}^{R} \mathbb{E}e^{\lambda Z_r} \\ &\le \inf_{\lambda > 0} \exp\{R[-\lambda t + \sigma^2(e^\lambda - 1 - \lambda)]\} \\ &= \exp\{-R\sigma^2 H(t/\sigma^2)\},\end{aligned} \tag{D.46}$$

where the last equality uses $\lambda = \log(1 + t/\sigma^2)$. Since $H(0) = 0$, $H'(x) = \log(1 + x) > 0$ for $x > 0$, and H diverges, its inverse on $[0, \infty)$ is well-defined. Substituting $t = b(\sigma^2, c_m/R)$ proves the probability assertion. If $\sigma^2 = 0$, every $X_r = p$ almost surely and the assertion holds with zero radius.

For fixed $t \ge 0$ and $v > 0$, direct differentiation gives

$$\frac{\partial}{\partial v}\{vH(t/v)\} = \log(1 + t/v) - t/v \le 0. \tag{D.47}$$

As $t \mapsto vH(t/v)$ is increasing, its level-h solution $b(v, h)$ is nondecreasing in v; the definition at zero preserves this ordering.

To obtain the explicit relaxation, use $j! \ge 2 \cdot 3^{j-2}$ for $j \ge 2$, proved by induction, to get

$$e^\lambda - 1 - \lambda = \sum_{j=2}^{\infty} \frac{\lambda^j}{j!} \le \frac{\lambda^2}{2} \sum_{j=0}^{\infty} (\lambda/3)^j = \frac{\lambda^2}{2(1 - \lambda/3)}, \qquad 0 \le \lambda < 3. \tag{D.48}$$

For $v, h > 0$, set $a = \sqrt{2h/v}$, $\lambda_0 = a/(1 + a/3)$, and $t_0 = \sqrt{2vh} + h/3$. The Legendre identity implicit in (D.46) and (D.48) give

$$\begin{aligned}vH(t_0/v) &= \sup_{\lambda \ge 0}\{\lambda t_0 - v(e^\lambda - 1 - \lambda)\} \\ &\ge \lambda_0 t_0 - v(e^{\lambda_0} - 1 - \lambda_0) \\ &\ge \lambda_0 t_0 - \frac{v\lambda_0^2}{2(1 - \lambda_0/3)} \\ &= \frac{va^2 + va^3/6 - va^2/2}{1 + a/3} = \frac{va^2}{2} = h.\end{aligned} \tag{D.49}$$

Monotonicity in t implies $b(v, h) \le t_0$. The cases $h = 0$ or $v = 0$ follow directly from the definitions. □

Proposition 1 (Uniform certificate using paired sample variance)**.** *Assume the conditions of Theorem 2, with $R_i \ge 2$. Retain the per-trial paired scores needed to compute the sample variance. Choose fixed $\eta_K, \eta_V, \eta_M > 0$ summing to one. For $A \in \{K, V, M\}$, define*

$$c_i^A(P) = \log \frac{1}{\eta_A \delta w_i \nu_i(P)}, \qquad k_i = \lfloor R_i/2 \rfloor, \qquad S_i^2(P) = \frac{1}{R_i - 1} \sum_{r=1}^{R_i} [X_{i,r}(P) - \widehat{p}_i(P)]^2. \tag{D.50}$$

Put

$$
\begin{aligned}
v_i^+(P) &= \min\left\{\frac{1}{4}, \frac{1}{2}u\left(2S_i^2(P), \frac{c_i^V(P)}{k_i}\right)\right\}, \\
q_i^+(P) &= \min\left\{u\left(\widehat{p}_i(P), \frac{c_i^K(P)}{R_i}\right), \widehat{p}_i(P) + b\left(v_i^+(P), \frac{c_i^M(P)}{R_i}\right)\right\}.
\end{aligned}
\tag{D.51}
$$

With probability at least $1-\delta$, simultaneously for every buffer–checkpoint pair in Theorem 2,

$$
\begin{aligned}
L_{\mathcal{B}}(P) \le \mathcal{C}_{\mathcal{B}}^{\mathrm{hyb}}(P) &:= \frac{1}{n}\sum_{i\in\mathcal{J}}[2q_i^+(P)-1]_+ \\
&\le \min\left\{\vartheta, \epsilon + \frac{1}{n}\sum_{i\in\mathcal{J}}\min\left\{1, 2b\left(v_i^+(P), \frac{c_i^M(P)}{R_i}\right)\right\}\right\}.
\end{aligned}
\tag{D.52}
$$

The same zero-loss convention applies if $n=0$ or $m=0$.

Proof. For each fixed i, P, apply Lemmas 1, 6, and 7 conditionally on $\mathcal{F}_i$, using their respective costs c_i^K, c_i^V, c_i^M. Let $E_{i,P}^A$ denote the corresponding failure event. Their conditional probabilities are at most $\eta_A \delta w_i \nu_i(P)$. Thus

$$
\begin{aligned}
\mathbb{P}\left(\bigcup_{i,P,A} E_{i,P}^A\right) &\le \sum_{i,P,A}\mathbb{E}[\mathbb{P}(E_{i,P}^A \mid \mathcal{F}_i)] \\
&\le \delta\sum_A \eta_A \sum_i w_i\,\mathbb{E}\left[\sum_P \nu_i(P)\right] \le \delta.
\end{aligned}
\tag{D.53}
$$

On the complementary event, write $p_i(P) = (1+d_i(P))/2$ and $\sigma_i^2(P) = \mathrm{Var}(X_{i,1}(P) \mid \mathcal{F}_i)$. For every i, P, monotonicity of b gives

$$
\begin{aligned}
p_i(P) &\le \min\left\{u\left(\widehat{p}_i, \frac{c_i^K}{R_i}\right), \widehat{p}_i + b\left(\sigma_i^2, \frac{c_i^M}{R_i}\right)\right\} \\
&\le \min\left\{u\left(\widehat{p}_i, \frac{c_i^K}{R_i}\right), \widehat{p}_i + b\left(v_i^+, \frac{c_i^M}{R_i}\right)\right\} = q_i^+(P), \\
[d_i(P)]_+ &\le [2q_i^+(P)-1]_+ \\
&\le \min\left\{1, [\widehat{d}_i(P) + 2b(v_i^+, c_i^M/R_i)]_+\right\}.
\end{aligned}
\tag{D.54}
$$

Here arguments P are suppressed only inside the chain. Identity coordinates have zero loss and are excluded before summation. Summing and applying Lemma 5, followed by empirical feasibility, proves (D.52). The mean and variance events need not be independent. □

For illustration of the rate, if $S_i^2(P) = 0$, then $u(0,h) = 1 - e^{-h}$. Therefore

$$
\begin{aligned}
v_i^+(P) &\le \frac{1}{2}(1 - e^{-c_i^V(P)/k_i}) \le \frac{c_i^V(P)}{2k_i}, \\
2b\left(v_i^+(P), \frac{c_i^M(P)}{R_i}\right) &\le 2\sqrt{\frac{2v_i^+(P)c_i^M(P)}{R_i}} + \frac{2c_i^M(P)}{3R_i} \\
&\le 2\sqrt{\frac{c_i^V(P)c_i^M(P)}{k_iR_i}} + \frac{2c_i^M(P)}{3R_i}.
\end{aligned}
\tag{D.55}
$$

Since k_i is proportional to R_i, this can scale as $\log(1/[\delta w_i \nu_i(P)])/R_i$ when the costs are comparable. Population variance can remain positive when observed variance is zero. Computing this certificate requires cached per-trial contrasts. Confidence splitting enlarges the KL cost from c_i to c_i^K; consequently, the hybrid certificate is not guaranteed to dominate the standalone KL certificate at the same total confidence level.

D.9 Why the Bound Does Not Accumulate

Proposition 2 (Sharp separation from rolling references)**.** *For one fixed probe, any integer $T \geq 1$, and any $\epsilon \geq 0$, predecessor-relative constraints $[G(P_{k-1}; z) - G(P_k; z)]_+ \leq \epsilon$ imply only*

$$[G(P_0; z) - G(P_T; z)]_+ \leq \min\{1, T\epsilon\}, \tag{D.56}$$

and equality is attainable. In contrast, an exact fixed-anchor constraint gives a sharp bound $\min\{1, \epsilon\}$ for every T.

Proof. Let $\mu_k \in [0, 1]$ be the edited-agent expected score under P_k. The common parent baseline cancels. Subadditivity of the positive part and telescoping give

$$\begin{aligned}
[G(P_0; z) - G(P_T; z)]_+ &= [\mu_0 - \mu_T]_+ \\
&= \left[\sum_{k=1}^{T} (\mu_{k-1} - \mu_k)\right]_+ \\
&\leq \sum_{k=1}^{T} [\mu_{k-1} - \mu_k]_+ \leq T\epsilon.
\end{aligned} \tag{D.57}$$

The score range also bounds this quantity by one. For equality, choose deterministic scores

$$\mu_k = [1 - k\epsilon]_+, \qquad [\mu_{k-1} - \mu_k]_+ = \min\{\epsilon, [1 - (k-1)\epsilon]_+\} \leq \epsilon, \qquad \mu_0 - \mu_T = \min\{1, T\epsilon\}. \tag{D.58}$$

These scores can be realized by a black-box edit family with deterministic normalized outcomes. To attain the fixed-anchor bound, take $\mu_0 = 1$ and $\mu_k = 1 - \min\{1, \epsilon\}$ for every $k \geq 1$. Then every later deficit relative to P_0 equals $\min\{1, \epsilon\}$, independently of k. □

D.10 Sharpness of the KL Exponent

Proposition 3 (A matching bounded-score large-deviation exponent)**.** *Within the bounded paired-score model, for every $p \in (0, 1)$ and $x = k/R < p$ with $k \in \{0, \ldots, R\}$, there is a score distribution whose transformed paired mean $\widehat{p}$ satisfies*

$$\frac{e^{-R\,\mathrm{kl}(x\|p)}}{R+1} \leq \mathbb{P}(\widehat{p} = x) \leq \mathbb{P}(\widehat{p} \leq x) \leq e^{-R\,\mathrm{kl}(x\|p)}. \tag{D.59}$$

Thus the exponent used in Lemma 1 is sharp up to a polynomial prefactor. This statement does not assert exact finite-sample optimality of the confidence interval or union bound.

Proof. Let Z_r be iid Bernoulli variables with mean p and choose paired scores $Y(Q; \omega_r) = Z_r$ and $Y(P; \omega_r) = 1 - Z_r$. Their transformed contrast is exactly Z_r, so $R\widehat{p}$ has the binomial distribution. Changing the binomial parameter from p to $x = k/R$ gives

$$\begin{aligned}
\mathbb{P}_p(\widehat{p} = x) &= \binom{R}{k} p^k (1-p)^{R-k} \\
&= \binom{R}{k} x^k (1-x)^{R-k} e^{-R\,\mathrm{kl}(x\|p)} \\
&= \mathbb{P}_x(\mathrm{Bin}(R, x) = k) e^{-R\,\mathrm{kl}(x\|p)}.
\end{aligned} \tag{D.60}$$

For $0 < k < R$, the ratio of consecutive binomial masses at parameter $x = k/R$ is

$$\frac{\mathbb{P}_x(\mathrm{Bin}(R, x) = j+1)}{\mathbb{P}_x(\mathrm{Bin}(R, x) = j)} = \frac{R-j}{j+1} \frac{k}{R-k}. \tag{D.61}$$

This ratio decreases with j, is greater than one at $j = k-1$, and is less than one at $j = k$. Hence k is a mode. The maximal mass among $R+1$ possible outcomes is at least $1/(R+1)$, so (D.60) proves the lower bound. For $k = 0$, the binomial distribution at parameter $x = 0$ is degenerate at zero and the same conclusion holds, using the standard $0^0 = 1$ convention. The upper bound is (D.8). □