Scope Before You Persist: Preventing Cross-Family Interference in Agent Memory

Anonymous authors
Paper under double-blind review

Abstract

Persistent memory lets language-model agents improve prompts and skills without updating model weights. We show that matching retrieval scope to certification scope enables these edits to support reliable repeated adaptation across recurring task families. We study frozen-model agents on ProcStream-RSI, a 12-round code-repair stream, using Orthogonal Regression Control (ORC), an execution-grounded gate for persistent skill edits. In an intervention that holds proposals and gate decisions fixed, retrieving each accepted skill only for its originating family raises mean hidden trajectory utility from 0.713 under global memory to 0.816 and changes harmful deployments from six of eight to none. In 27 paired randomized-order streams, Scoped-ORC improves mean trajectory utility by 0.063 [0.037, 0.094] over Global-ORC, accepts 63 rather than 12 updates, and produces multiple accepted updates in 19/27 streams, with 0/63 harmful acceptances. The global control reaches 0.713, below the static agent's 0.775, because locally valid edits can interfere with unrelated families. These results establish scope matching as a complementary control for persistent agent memory: certification determines whether an edit is supported, while retrieval scope determines where that evidence authorizes its use.

1 Introduction

Large language models (LLMs) can critique outputs, retain verbal experience, and edit the agent programs that call them (Shinn et al., 2023; Madaan et al., 2023; Hu et al., 2025). Recent self-referential systems make these edits persistent (Yin et al., 2024; Zhang et al., 2025). Such systems are often called self-improving because a selected descendant outperforms its ancestor. Persistent deployment creates a complementary opportunity: convert local feedback into a live sequence of improvements that remains useful across recurring task families. Grounded feedback, deployed trajectories, and matched-compute controls are important because intrinsic self-correction can be unreliable (Huang et al., 2024), actor–judge loops can share shortcuts (Pan et al., 2024), and initial gains may not transfer or recur (Wang et al., 2026a).

We ask: when does persistent skill editing improve a deployed agent across recurring task families, rather than only the family that produced the latest feedback? At each round, a frozen actor solves discovery tasks, the same frozen model edits a 360-character skill policy, and a gate either deploys the proposal or retains the incumbent. The deployed policy shapes both future task solutions and the evidence used to produce the next edit. No model weights are trained.

Our central claim is that persistent memory requires two decisions: whether an update is supported and where it should apply. Evidence collected within one task family may justify the first decision without justifying global deployment. We make three contributions:

1. A deployment view of self-improvement. We introduce ProcStream-RSI, a 12-round procedural stream with recurring task families and fixed hidden checkpoints, and

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.

evaluate the complete sequence of deployed policies rather than selecting an archive-best checkpoint.

2. A mechanism for deployment-scope mismatch. Across permissive baselines and the execution-grounded ORC gate, we show how a locally valid edit can affect unrelated families when deployed through one compact global skill. We identify this interaction as cross-family interference.
3. A scoped-memory remedy. A fixed-output intervention changes only retrieval while holding proposals, decisions, programs, and scores fixed; paired end-to-end streams then allow later behavior and gate outcomes to diverge. Together, they show that scoped retrieval prevents the observed cross-family harm and supports repeated updates across a 12-round stream.

Family-scoped retrieval preserves locally certified gains and, in a 27-stream randomized-order extension, increases both trajectory utility and the number of useful accepted updates. The global controls explain this advantage: across eight 12-round streams, each global-update method falls below Static because accepted rules affect families outside the gate's evidence. Persistent adaptation therefore depends on matching evidence scope and deployment scope in addition to proposal quality and gate accuracy.

We call this outcome continual adaptation: sequentially accepted edits remain useful as the deployed family-scoped memory evolves.

2 Related Work

Self-correction and self-generated learning. Inference-time refinement and verbal memory can improve outputs without weight updates (Madaan et al., 2023; Shinn et al., 2023), whereas self-training changes weights using known answers, filters, comparisons, or rewards (Zelikman et al., 2022; Singh et al., 2023; Yuan et al., 2024). Both rely on a quality asymmetry between generation and selection. When the selector is ungrounded or exploitable, iterative improvement can saturate, regress, or collapse (Huang et al., 2024; Herel & Mikolov, 2024; Song et al., 2024; Shafayat et al., 2025). Our focus is persistent prompt-level state, and execution supplies the asymmetry.

Persistent agent change. Automated design searches over code, prompts, tools, and workflows (Hu et al., 2025). Gödel Agent makes the task agent self-referential (Yin et al., 2024); Darwin Gödel Machine retains an open-ended archive of self-edited coding agents (Zhang et al., 2025); and later systems use two-timescale skills or comparative lineages (Wang et al., 2026b; Liu et al., 2026). These works establish that frozen foundation models can improve external scaffolds. We study the deployed lineage under partial verification rather than the best member of an archive. The closest continual evaluation reports that only a regression-aware optimizer transfers and improves in a second phase (Wang et al., 2026a). Our experiments isolate a complementary design problem: the scope mismatch between local certification and global memory.

Evaluator error dependence. Self-rewarding and meta-rewarding improve actors and judges together (Yuan et al., 2024; Wu et al., 2024), and evaluator co-evolution is made explicit by the Red Queen Gödel Machine (Iacob et al., 2026). Yet actor and judge can converge on the same shortcut even without gradient updates (Pan et al., 2024). Memory Reward Inflation argues that corrective evidence must both track truth and have sufficiently distinct errors (Asadolahi et al., 2026). We operationalize that idea at deployment: deterministic program execution and metamorphic relations supplement the actor-visible tests, while sealed execution separates same-probe certification error from prospective cross-family interference.

Code evaluation. EvalPlus shows that sparse base tests overestimate generated-code correctness (Liu et al., 2023); BigCodeBench broadens instructions and library use (Zhuo et al., 2025). Public benchmarks may appear in pretraining. Our procedural stream supplies generated instances and held-out cases while keeping task semantics familiar and fixed; it

measures adaptation within recurring contracts. A frozen HumanEval+ sample provides an external transfer diagnostic.

3 Continual Self-Improvement as Deployment

Let π_θ be a frozen actor and K_t the persistent natural-language skill deployed at round t. On discovery batch D_t, the actor receives the task specification, buggy program, visible examples, and K_t, then emits repaired programs. A frozen editor maps the resulting programs and visible failures to a proposal K'_t. A gate G_t evaluates K_t and K'_t on paired probe tasks and chooses

$$K_{t+1} = \begin{cases} K'_t, & G_t(K'_t, K_t) = 1, \\ K_t, & \text{otherwise.} \end{cases} \tag{1}$$

The editor instruction is fixed, and proposed meta-skill text is ignored in confirmatory runs; the only inherited state is K_t. Generated task programs are ephemeral outputs rather than mutable system code.

Let U_t be hidden utility of K_t on a fixed balanced checkpoint bank. We report terminal utility, mean trajectory utility MTU $= \frac{1}{T+1}\sum_{t=0}^{T} U_t$, mean backward transfer, worst-family retention, and a separate final-bank score. Archive-best utility is not used for primary claims.

3.1 Why false acceptance controls drift

Suppose a proposal is beneficial with probability π, has mean gain $G > 0$ if beneficial, and mean loss $L > 0$ otherwise. Let a gate accept beneficial and harmful proposals with probabilities α and β.

Proposition 1. The expected deployed change is positive exactly when

$$\pi\alpha G > (1-\pi)\beta L. \tag{2}$$

For an AND gate over conditionally independent evidence channels with false-acceptance rates β_j, harmful acceptance is $\prod_j \beta_j$. If their false-acceptance events are identical, adding channels leaves β unchanged.

Thus marginal verifier accuracy is insufficient: the errors of additional evidence matter. Our experiments measure channel AUROC, surrogate-positive but hidden-non-improving decisions, harmful decisions, and residual error dependence against sealed execution outcomes. The proposition only covers outcomes represented in the gate evidence; it cannot protect an unscoped global edit from interactions with task families that have not yet appeared.

3.2 Family-scoped retrieval

We test a direct scope intervention. Instead of replacing one global policy, Scoped-ORC stores an accepted candidate in the current family's slot. For a task from family f, the actor receives

$$K_t(f) = \begin{cases} K_t^{(f)}, & \text{if an accepted slot for } f \text{ exists,} \\ K_0, & \text{otherwise.} \end{cases} \tag{3}$$

The active policy remains exactly 360 characters: retrieval changes which policy is supplied, not the prompt budget. Proposal generation and the gate are unchanged. In the round-0 comparison, Global- and Scoped-ORC therefore use the same discovery outputs, candidate, probe evidence, and acceptance decision; only the scope of deployment differs. This design tests a mechanism—cross-family exposure—rather than a stronger editor or verifier.

4 Orthogonal Regression Control

ORC evaluates incumbent and candidate policies on the same probe instances. Each task has three online channels: four public examples, four private exact-output tests absent from

the actor/editor prompt, and at least six answer-free metamorphic checks. For task i and channel $c \in \{\text{pub}, \text{priv}, \text{meta}\}$, let $d_{i,c}$ be candidate-minus-incumbent pass rate.

Probe instances are partitioned into the current family and one group for each historical family. A returning current family remains distinct from its historical instances. For each group/channel pair, ORC computes a deterministic one-sided 95% paired-bootstrap lower bound $L_{g,c}$. We exactly enumerate bootstrap samples when $n^n \leq 50{,}000$ and otherwise use 2,000 seeded draws; every group contains at least four tasks.

With $\delta_{\text{new}} = 0$ and $\delta_{\text{reg}} = 0.02$, a candidate is deployed only if

$$L_{\text{current,pub}} \geq -\delta_{\text{reg}}, \qquad L_{\text{current,priv}} > \delta_{\text{new}}, \tag{4}$$

$$L_{\text{current,meta}} \geq -\delta_{\text{reg}}, \qquad L_{g,c} \geq -\delta_{\text{reg}} \quad \forall g < t, \forall c, \tag{5}$$

and no private or metamorphic evaluation reports an unsafe program. The conjunction prevents a gain in one channel from averaging away a regression in another. Rejected proposals remain in the trace but never affect later actor or editor prompts.

5 Experimental Design

5.1 ProcStream-RSI

ProcStream-RSI v3.1 is a 12-round code-repair stream with the schedule

boundary, rotation, ordering, normalization, boundary, rotation, nested, missing, prefix, chunk, tie-breaking, ordering.

The repeats make retention observable. Each family receives a seed-derived opaque protocol token. Discovery public examples distinguish the family-specific repair rule, whereas probe, checkpoint, and final public examples are neutral: the supplied buggy program already passes them. Their private, metamorphic, and hidden cases exercise that rule. Importantly, the oracle semantics and family order are fixed across all streams; seeds vary tokens, names, program variants, and cases, not the underlying convention. The benchmark therefore measures behavior on procedurally instantiated familiar contracts, not adaptation to randomized new semantics. Static's 0.775 mean trajectory utility is consistent with substantial prior competence.

Each round has four discovery and four probe tasks. Separately generated fixed banks contain one checkpoint and two final tasks per family, for 123 tasks per stream. Exact-output public, private, and hidden inputs are pairwise disjoint, and hidden cases are unavailable to the online agent. Role-normalized public views are unique; full manifest-integrity checks are described in the reproducibility statement and supplement.

5.2 Methods and controls

We compare seven paired methods:

1. Static, which never proposes an update;
2. Frozen-compute, which makes the same editor, replay, gate, and evaluation calls as ORC but never deploys;
3. Latest-only, which deploys every nonempty, well-formed proposal;
4. Self-judge, whose actor estimates correctness of incumbent and candidate programs and accepts the higher mean estimate;
5. Replay, which accepts on current public mean gain and historical public mean non-regression;
6. ORC, the full channel-wise gate; and
7. Batch-ORC, which exposes the editor to cumulative discoveries but keeps the initial policy until only the final proposal may deploy.

Batch-ORC tests sequential advantage over a cumulative one-shot update; Frozen-compute tests the effect of additional calls. A hidden-gated oracle exists only as code-path validation and is excluded from the main comparison.

5.3 Scope and randomized-entry extension

We first isolate retrieval scope on the eight main streams. Every Global-ORC lineage accepts the same kind of round-0 boundary rule, so family-scoped retrieval uses its archived completion on boundary tasks and the archived Static completion elsewhere. The proposals, gate decisions, task programs, and scores remain unchanged; only the task-to-skill assignment differs.

We then test the mechanism prospectively under balanced randomized entry. The v4 generator applies a seed-deterministic permutation to the v3.1 family multiset. Eighteen seeds give two streams for each of the nine possible first families. Each stream contains one update round, four discovery and four probe tasks, and checkpoint and final banks with two tasks per family. Global- and Scoped-ORC are paired within stream and must share their proposal and gate decision. The primary estimand is Scoped minus Global hidden checkpoint utility after that decision; secondary outcomes are final-bank utility, harmful accepted updates, and changes on the current versus eight non-current families. We report stream-bootstrap 95% intervals and a two-sided paired sign-flip test.

5.4 Model, runs, and outcomes

The actor/editor is Qwen3-Coder-Next served by the pinned Parasail BF16 endpoint through OpenRouter, with provider fallback disabled, temperature 0, a fixed stream seed, 700 actor tokens, and 500 editor tokens. The model was selected on public smoke tasks for response and format reliability before the v3 pilot. Actor skill text is padded to 360 characters and editor input to 48,000 characters, making prompt budget independent of policy length. Batch-ORC's cumulative-evidence editor is instead padded to 64,000 characters on every round, and its larger token cost is reported separately. Model weights are never trained.

After a two-seed, six-round difficulty pilot, all thresholds and prompt content were fixed. The main study uses eight unseen seeds and all 12 rounds. The independent unit is a complete stream. Primary contrasts are ORC–Replay mean trajectory utility, ORC–Latest-only mean trajectory utility, and ORC–Batch-ORC terminal checkpoint utility. We report paired raw seeds, bootstrap 95% intervals, two-sided paired sign-flip sensitivity tests, and Holm correction across these three tests. The eight seed-instantiated streams are the analysis population; task-level results are descriptive. Provider-reported token counts determine counterfactual endpoint cost.

For external validity, terminal Static, Latest-only, Replay, ORC, and Batch-ORC policies are evaluated without feedback on 32 HumanEval+ tasks selected before adaptive main results by a fixed SHA-256 ranking. The upstream augmented tests remain offline and execute in a pinned NumPy container. This measures transfer of the learned skills beyond ProcStream-RSI.

A second, actor-only diagnostic applies the same terminal skills to each stream's 18-task final bank using GPT-OSS-120B on the fixed AkashML BF16 endpoint. It receives no update feedback and does not rerun the editor/search process. Because GPT-OSS reasoning shares the completion budget, all cells use a 2,000-token cap. This test asks whether policy effects transfer across actors.

The canonical randomized-entry extension reruns both editing and selection with the same Qwen3-Coder-Next Parasail BF16 endpoint and 700/500 actor/editor completion limits as the main study. A dated protocol fixes its seed list, generator version, estimands, endpoint, and analysis. Endpoint matching isolates family order without changing the editor/search backbone.

Finally, a prespecified full-length extension uses 27 seeds, exactly three for each possible first family, and pairs Global- with Scoped-ORC within every stream. Policies evolve for all

Table 1: Main ProcStream-RSI results over eight paired streams. Brackets are seed-bootstrap 95% intervals. Harmful accepts count accepted rounds whose fixed hidden checkpoint score decreased.

Method	Mean traj. ↑	Final checkpoint ↑	BWT ↑	Accepted	Harmful
Static	0.775 [0.734, 0.814]	0.775 [0.734, 0.814]	0.000 [0.000, 0.000]	0.00	0.00
Frozen-compute	0.775 [0.734, 0.814]	0.775 [0.734, 0.814]	0.000 [0.000, 0.000]	0.00	0.00
Latest-only	0.703 [0.678, 0.725]	0.720 [0.681, 0.755]	-0.253 [-0.300, -0.204]	12.00	6.25
Self-judge	0.736 [0.696, 0.774]	0.722 [0.641, 0.802]	-0.140 [-0.220, -0.059]	3.38	1.75
Replay	0.707 [0.676, 0.738]	0.681 [0.621, 0.745]	-0.240 [-0.297, -0.191]	7.88	3.88
ORC	0.713 [0.653, 0.761]	0.707 [0.645, 0.760]	0.000 [0.000, 0.000]	1.00	0.75
Batch-ORC	0.775 [0.734, 0.814]	0.775 [0.734, 0.814]	0.000 [0.000, 0.000]	0.00	0.00

Table 2: Prespecified paired contrasts. Intervals are descriptive percentile seed-bootstrap 95% intervals. Paired sign-flip sensitivity p-values require symmetry of seed-level differences; Holm adjustment covers the three rows.

Contrast	Mean difference [bootstrap 95%]	p	Holm p
ORC - Replay mean trajectory	0.006 [-0.045, 0.048]	0.8438	1.0000
ORC - Latest-only mean trajectory	0.010 [-0.036, 0.051]	0.7188	1.0000
ORC - Batch-ORC final	-0.068 [-0.123, -0.004]	0.0938	0.2812

12 rounds and may accumulate multiple accepted slots. Primary outcomes are hidden trajectory and final-checkpoint utility; secondary outcomes include acceptance counts, streams with at least two acceptances, harmful acceptances, and router robustness. Router analysis uses oracle labels, a stochastic fallback-on-miss curve, and a leave-one-seed-out character 3–5-gram multinomial Naive Bayes router trained only on public discovery/probe prompts. Thus, the archive isolates the direct effect of scope under fixed candidates and decisions; the full extension estimates its end-to-end effect, including downstream changes in proposals and gates.

6 Results

Global controls reveal the value of scope matching. Static and Frozen-compute are identical at 0.775 mean trajectory utility and final checkpoint utility, despite Frozen-compute's counterfactual endpoint cost being roughly five times as high (table 1). Latest-only accepts every proposal and reaches 0.703 mean trajectory utility with mean BWT -0.253; Self-judge and Replay reach 0.736 and 0.707 with BWT -0.140 and -0.240. Their average accepted-update counts are 12.0, 3.38, and 7.88, of which 6.25, 1.75, and 3.88 respectively reduce the next hidden checkpoint. Their corresponding harmful-acceptance fractions are 52.1%, 51.9%, and 49.2%.

ORC accepts one proposal at round 0 in every stream and then rejects all later proposals. At round 0 no historical family exists, so no historical-family constraint can yet be evaluated. Its post-introduction BWT and worst-family retention are mechanically zero because no later update is deployed. Mean trajectory utility is 0.713, and 6/8 accepted updates reduce the next fixed hidden checkpoint. Its mean-trajectory differences from Replay (0.006) and Latest-only (0.010) are negligible. Batch-ORC accepts no proposal and reproduces Static, providing the cumulative one-shot reference for the sequential trajectory.

The family-level scores expose the mechanism: the accepted rule raises boundary-family hidden utility by 0.398 on average, but simultaneously changes missing, rotation, tie-breaking, and chunk families by -0.461, -0.234, -0.148, and -0.117. The shared global skill therefore couples a useful local update to measurable changes in other families.

Final-bank hidden means are 0.786 for Static, Frozen-compute, and Batch-ORC, 0.705 for Latest-only, 0.719 for Self-judge, 0.682 for Replay, and 0.682 for ORC. These terminal-policy results are consistent with the trajectory analysis.

Table 3: Terminal-policy transfer. Values are means with seed-bootstrap 95% intervals. HumanEval+ uses the fixed 32-task sample; GPT-OSS uses each stream's 18-task sealed final bank. †Family-scoped retrieval over the same archived completions.

Method	HumanEval+ pass rate	GPT-OSS hidden pass rate
Static	0.914 [0.898, 0.930]	0.941 [0.907, 0.974]
Latest-only	0.918 [0.891, 0.941]	0.911 [0.886, 0.937]
Replay	0.922 [0.895, 0.945]	0.916 [0.878, 0.951]
ORC	0.895 [0.867, 0.922]	0.951 [0.926, 0.968]
Batch-ORC	0.914 [0.898, 0.930]	0.941 [0.907, 0.974]
Scoped retrieval†	—	0.960 [0.930, 0.987]

Table 4: Same accepted skills, different retrieval scope on the eight main streams. Scoped retrieval applies each ORC rule only to its originating family and uses the initial skill elsewhere.

Deployment	Mean traj. ↑	Final checkpoint ↑	Accepted	Harmful
Static	0.775 [0.734, 0.814]	0.775 [0.734, 0.814]	0	0
Global ORC	0.713 [0.653, 0.762]	0.707 [0.645, 0.760]	8	6
Scoped retrieval	0.816 [0.782, 0.852]	0.819 [0.786, 0.855]	8	0

Family-scoped retrieval preserves local gains across families. Changing only retrieval scope yields the central improvement (table 4). The fixed-completion Scoped-ORC intervention attains mean trajectory utility 0.816, compared with 0.713 for Global-ORC and 0.775 for Static. Its paired advantage is 0.103 over Global-ORC (bootstrap 95% interval $[0.052, 0.151]$, sign-flip sensitivity $p = 0.03125$) and 0.041 over Static ($[0.022, 0.056]$, $p = 0.03125$). The same eight locally accepted updates raise the immediate global checkpoint by 0.044 on average when retrieved only for boundary tasks, and none is harmful; global deployment made six harmful. Scoped retrieval prevents 0.112 mean round-0 interference without changing a completion used for the active family. The randomized-entry experiment below tests the same mechanism across every possible first family; the full extension then tests whether scoped retrieval supports repeated updating.

Balanced randomized-entry replication. Across 18 streams—two per possible first family—the shared gate accepts 5 candidates. Global deployment produces 3 harmful accepted updates, compared with 0 under family-scoped deployment. Mean post-decision hidden checkpoint utility is 0.761 for Global-ORC and 0.802 for Scoped-ORC. The primary paired Scoped-minus-Global effect is 0.041 [-0.004, 0.107] (sign-flip sensitivity $p = 0.1875$); the final-bank effect is 0.031 [0.001, 0.070] ($p = 0.1250$). The two deployments have the same current-family change by construction when their shared candidate is accepted; their mean non-current-family changes are -0.047 and 0.000, respectively. The paired deployments therefore isolate cross-family exposure: they share the current-family gain, while scoped retrieval prevents changes to every non-current family.

Full randomized-order extension. Across 27 complete 12-round streams, Scoped-minus-Global mean hidden trajectory utility is 0.063 [0.037, 0.094] (sign-flip sensitivity $p < 10^{-4}$; Holm $p < 10^{-4}$). Global-ORC accepts 12 updates in total and has 2 streams with at least two acceptances; Scoped-ORC accepts 63 updates and has 19 such streams. The paired increase in the multiple-acceptance indicator is 0.630 (sign-flip sensitivity $p < 10^{-4}$). The harmful/accepted counts are 6/12 for Global-ORC and 0/63 for Scoped-ORC. On this templated benchmark, the learned router classifies held-out checkpoint prompts with 100.0% accuracy (243/243) and matches oracle routing at 0.848 trajectory utility; the fallback-on-miss curve shows how utility degrades as routing errors increase.

Orthogonal evidence improves selection within scope. Across 479 within-stream proposal–round comparisons pooled descriptively over five methods, the public and private/metamorphic components reach AUROC 0.752 and 0.998, respectively. Yet all eight

Table 5: Balanced randomized-entry replication (18 streams; two per first family). Brackets show stream-bootstrap 95% intervals.

Deployment	Accepted	Harmful	Checkpoint t=1	Non-current Δ
Global ORC	5	3	0.761 [0.709, 0.802]	-0.047 [-0.120, 0.005]
Scoped ORC	5	0	0.802 [0.779, 0.826]	0.000 [0.000, 0.000]

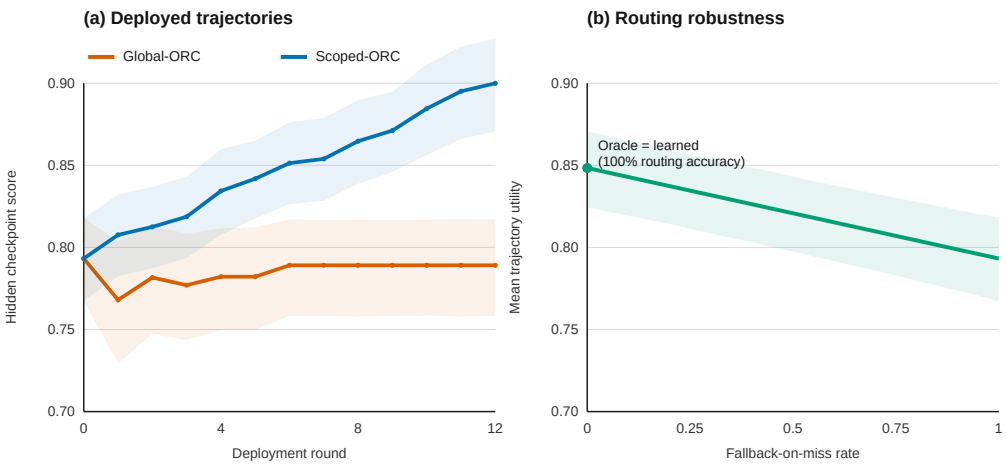

Figure 1: Full randomized-order extension. (a) Mean hidden checkpoint trajectories across 27 paired 12-round streams; shading is the stream-bootstrap 95% interval. (b) Scoped-ORC utility under oracle routing, stochastic fallback-to-initial-policy misses, and the learned leave-one-seed-out router. The curve models conservative fallback to the initial policy as routing misses increase.

ORC acceptances are safe locally while only two improve the next global checkpoint. Thus, strong within-scope selection still requires scope-matched deployment; component-level diagnostics appear in appendix A.

Scope matching supports repeated adaptation. The scoped extension accepts 63 updates, produces multiple acceptances in 19/27 streams, and records 0/63 harmful acceptances. Global ORC instead stops changing after its first accepted update in most streams. The scoped trajectories therefore establish repeated adaptation; testing compounding additionally requires a cumulative one-shot comparator.

7 Conclusion

Persistent agent memory requires alignment between certification and deployment scope. Across the complete extension, scoped deployment raises mean trajectory utility by 0.063, accepts 63 updates versus 12 globally, and yields harmful counts of 0/63 versus 6/12. The resulting principle is actionable: evidence that justifies an update should also determine where it is retrieved. Scope-matched memory turns local certification into repeated continual adaptation.

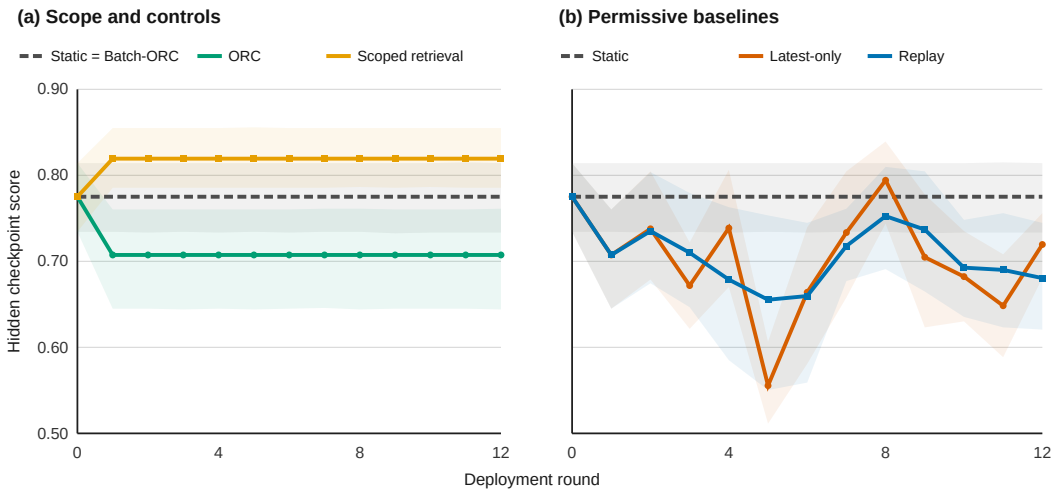

Figure 2: Mean fixed-bank hidden performance of deployed policies. Panel (a) isolates the scope comparison and controls; Batch-ORC exactly overlaps Static. Panel (b) shows the permissive baselines against Static. Shading denotes seed-bootstrap 95% intervals. The trajectory includes $t = 0$ and all 12 deployment decisions. Scoped retrieval applies each accepted rule only to its originating family.

Table 6: Full 12-round randomized-order extension (27 paired streams). Brackets are stream-bootstrap 95% intervals; harmful updates reduce the next hidden checkpoint.

Method	Trajectory	Final checkpoint	Accepts/stream	Streams ≥ 2	Harmful/accepted
Global-ORC	0.785 [0.757, 0.812]	0.789 [0.758, 0.817]	0.44	2/27	6/12
Scoped-ORC	0.848 [0.825, 0.871]	0.900 [0.871, 0.927]	2.33	19/27	0/63

8 Limitations and Broader Impact

The study covers inspectable scaffold edits with one frozen model and nine code-contract families. A cumulative one-shot control for compounding, open-world and wrong-slot routing, additional editor/search backbones, and automatically learned invariants remain future work.

Generated programs run in a pinned, network-free Docker sandbox with least privilege, resource limits, and AST validation. The study has no personal data or human subjects. Full trajectories, bounded terminology, and cost accounting discourage overgeneralizing this scaffold-level result.

References

Mohammad Asadolahi, Amir Amini, Samira Talebi, Amirfarhad Farhadi, and Azadeh Zamanifar. Memory reward inflation in self-improving llm agents, 2026. arXiv preprint arXiv:2608.00017.

David Herel and Tomas Mikolov. Collapse of self-trained language models, 2024. arXiv preprint arXiv:2404.02305.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In International Conference on Learning Representations, 2025. URL https://arxiv.org/abs/2408.08435.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In International Conference on Learning Representations, 2024. URL https://arxiv.org/abs/2310.01798.

Alex Iacob, Andrej Jovanović, William F. Shen, Daniel Burkhardt, Meghdad Kurmanji, Nurbek Tastan, Lorenzo Sani, Niccolò Alberto Elia Venanzi, Ambroise Odonnat, Zeyu Cao, et al. The red queen gödel machine: Co-evolving agents and their evaluators, 2026. arXiv preprint arXiv:2606.26294.

Changzhi Liu, Yilun Liu, Sikuan Yan, Volker Tresp, and Yunpu Ma. Mendel gödel machine: Recursive self-improving coding agents via comparative evolution, 2026. arXiv preprint arXiv:2608.07645.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In Advances in Neural Information Processing Systems, 2023. URL https://openreview.net/forum?id=1qvx610Cu7.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. In Advances in Neural Information Processing Systems, 2023. URL https://arxiv.org/abs/2303.17651.

Jane Pan, He He, Samuel R. Bowman, and Shi Feng. Spontaneous reward hacking in iterative self-refinement, 2024. arXiv preprint arXiv:2407.04549.

Sheikh Shafayat, Fahim Tajwar, Ruslan Salakhutdinov, Jeff Schneider, and Andrea Zanette. Can large reasoning models self-train?, 2025. arXiv preprint arXiv:2505.21444.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In Advances in Neural Information Processing Systems, 2023. URL https://arxiv.org/abs/2303.11366.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, et al. Beyond human data: Scaling self-training for problem-solving with language models, 2023. arXiv preprint arXiv:2312.06585.

Yuda Song, Hanlin Zhang, Carson Eisenach, Sham Kakade, Dean Foster, and Yudong Ghai. Mind the gap: Examining the self-improvement capabilities of large language models, 2024. arXiv preprint arXiv:2412.02674.

Wenxiao Wang, Priyatham Kattakinda, and Soheil Feizi. Do agent optimizers compound? a continual-learning evaluation on terminal-bench 2.0, 2026a. arXiv preprint arXiv:2607.14004.

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, 2026b. arXiv preprint arXiv:2607.05297.

Tianhao Wu, Weizhe Yuan, Olga Golovneva, Jing Xu, Yuandong Tian, Jiantao Jiao, Jason Weston, and Sainbayar Sukhbaatar. Meta-rewarding language models: Self-improving alignment with llm-as-a-meta-judge, 2024. arXiv preprint arXiv:2407.19594.

Xunjian Yin, Xinyi Wang, Liangming Pan, Li Lin, Xiaojun Wan, and William Yang Wang. Gödel agent: A self-referential agent framework for recursive self-improvement, 2024. arXiv preprint arXiv:2410.04444.

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

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. In Advances in Neural Information Processing Systems, 2022. URL https://arxiv.org/abs/2203.14465.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: Open-ended evolution of self-improving agents, 2025. arXiv preprint arXiv:2505.22954.

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, et al. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. In International Conference on Learning Representations, 2025. URL https://arxiv.org/abs/2406.15877.

Reproducibility

The supplement contains generator and evaluator code, public and sealed manifests, companion hashes, exact prompts, seeds, endpoint routing, accepted and rejected policies, immutable code artifacts, raw response IDs, provider usage, the transactional budget ledger, unit and sandbox tests, frozen analysis scripts, external task IDs, and a dated deviation log. Online runners reject sealed manifests; the auditor rejects missing hidden cases, manifest mismatches, duplicate artifacts, split mismatches, and code-hash changes. Ambiguous timeout-after-dispatch requests are conservatively accounted for and never automatically retried. Every incomplete cell is archived and replayed from round zero under a unique logical identifier; no partial result enters the analysis. Method-level resource comparisons reconstruct uncached endpoint use from provider metadata, while a transactional study-wide ledger prevents overspend and records shared-cache reuse. Both randomized extensions follow the same rules and fixed endpoint as the main study. Immutable responses, paired run traces, excluded attempts, and separate dated protocols are included with the supplement.

AI Use Statement

Generative AI tools were used to identify and summarize literature, formulate hypotheses, design the benchmark and experiments, implement and test code, draft and edit the manuscript, and interpret experimental outputs. They were also the frozen systems under experimental study. The authors reviewed the cited sources, protocol, code, analyses, and final artifacts and take responsibility for their correctness.

A Secondary selection and transfer diagnostics

Across 479 within-stream proposal–round comparisons, the public component has AUROC 0.752 and accuracy 0.564 for "current hidden gain with no historical-family regression." It is positive for 209 non-safe comparisons: 197 are harmful and 12 fail to improve. The private/metamorphic margin raises AUROC to 0.998, leaving 12 non-safe positives (three harmful and nine non-improving); its errors correlate only $\phi = 0.164$ with public-component errors. At the frozen threshold, the private component and full conjunction make identical predictions. All eight ORC acceptances are safe on same-family gate probes, but only two

improve the next global checkpoint and six reduce it, changing descriptive positive predictive value from 1.00 locally to 0.25 globally. At round 0, no learned historical family is available to expose that cross-family regression to the gate.

On HumanEval+, Global-ORC scores 0.895 versus Static's 0.914. On the sealed GPT-OSS banks, Global-ORC scores 0.951 and scoped retrieval reaches 0.960, or 0.020 above Static (bootstrap 95% interval $[0.004, 0.035]$) and 0.010 above Global-ORC ($[-0.033, 0.055]$).

B Proof of the gate-drift proposition

Conditioning on whether a proposal is beneficial, the expected deployed change is $\pi\alpha G - (1-\pi)\beta L$, positive exactly under the stated inequality. Under conditional independence, all m channels falsely accept a harmful proposal with probability $\prod_j \beta_j$. If false-acceptance events are identical, their conjunction is the same event and retains probability β.

C Evidence roles and leakage boundaries

Discovery code, visible errors, specifications, and public cases are summarized for the editor. Probe tasks are visible to the actor but never summarized for the editor. Private exact-output tests and metamorphic outcomes are available only to the online gate. Checkpoint scores are recorded after each decision but never enter proposal or acceptance prompts. Sealed hidden cases are absent from online manifests and loaded only by the model-free auditor. The final bank is distinct from the checkpoint bank and evaluated once by the terminal state.

D Qualitative lineage analysis

Every ORC lineage accepts only its round-0 boundary-family proposal. The resulting policies therefore foreground one convention rather than an improvement procedure. For example, seed 101 ends with "count values where low <= x <= high" and seed 337 adds a rule to infer open bounds when endpoints are excluded by examples. These rules are reasonable on their discovery evidence and pass the same-family sealed gate probes, yet the fixed checkpoint often falls. The examples make the observed failure concrete: a locally valid compression is deployed as global persistent state, then the conservative gate freezes it. They are mechanism illustrations, not post-hoc evidence for a separate statistical claim.

E Pilot and protocol changes

The original v2 generator saturated on the first paid pilot stream: Latest-only reached a perfect hidden checkpoint after one update. These results were archived and excluded. The v3 diagnostic/neutral split restored initial hidden performance to the prespecified difficulty range. Across two six-round v3 pilot streams, ORC accepted one of 12 proposals, so the default threshold was conservative but non-degenerate and was retained. During deterministic main- manifest generation, three declared seeds lacked eight neutral boundary inputs; before any main call, v3.1 added explicit neutral and diagnostic constructors, bumped the generator ID, and added a test that generates all eight complete streams. No task semantic, seed, gate, or threshold changed.

The sequential editor input was fixed at 48,000 characters. Batch-ORC was initially given the same limit, but a model-free preflight during main execution found that cumulative evidence for seed 137 required 52,487 characters. The completed seed-101 48k batch run and a partial seed-137 run were archived and excluded; the batch limit was raised uniformly to 64,000 and all eight canonical batch cells were run from scratch with that budget. This correction prevents content truncation and changes cost, not evidence or acceptance logic. It was recorded before the remaining batch matrix was executed. The complete dated log, including network-timeout reruns and logical run identifiers, is distributed with the code.

F Full randomized-order extension details

The extension seed schedule contains 27 outcome-blind seeds with exactly three streams beginning with each of the nine families. Within a seed, Global- and Scoped-ORC use the same manifest, endpoint, limits, and 12-round family order. An acceptance updates the single global policy for Global-ORC and only the current-family slot for Scoped-ORC. The initial policy is the fallback for a family with no accepted slot. We bootstrap complete streams with 50,000 seeded resamples and use two-sided paired sign-flip Monte Carlo tests with 100,000 draws; the three utility contrasts receive Holm correction. The multiple-acceptance comparison uses the paired stream indicator and is secondary.

For router sensitivity, oracle routing selects the family slot. At miss probability q, each checkpoint score is the expectation $(1-q)s_{\text{slot}} + qs_0$, where s_0 is that task's archived initial-policy score; q ranges from 0 to 1 in increments of 0.05. This intervention models conservative fallback, not misrouting to another learned slot. The learned router is a dependency-free multinomial Naive Bayes classifier over character 3–5-grams. For each held-out seed it is trained on discovery and probe prompts from the other 26 seeds and evaluated on the nine checkpoint prompts of the held-out seed. It obtains 243/243 correct routes, so its point overlaps the oracle result. Prompt templates make this an in-distribution diagnostic.

All 54 online cells complete before hidden auditing. Network-incomplete attempts are archived, charged according to the transactional ledger, and replayed from round zero under new logical identifiers. No partial attempt enters the analysis. The extension's ledger-charged cost is \$2.7527; total main, external, one-update, and full-extension spend is \$6.9965.

G Sandbox and budget checks

The test suite exercises direct and indirect file access, dunder traversal, imports, network and process creation, workspace visibility, timeout accounting, cross-process budget races, manifest checksums, seed diversity, role uniqueness, partition disjointness, reference solutions, ORC conjunction logic, and negative backward transfer. All paid cells reserve against one SQLite ledger using BEGIN IMMEDIATE; returned provider cost is authoritative. Raw successful responses are archived before parsing, while a timeout after dispatch is marked ambiguous and charged at its conservative reservation.