SAFETY IS NOT CONSERVED: CONSTRAINING SELF-IMPROVEMENT ACROSS MODELS AND AGENT HARNESSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Safety is not automatically conserved under recursive self-improvement (RSI): capability-oriented updates to model parameters or an agent's execution harness can regress safety, including through cross-component interactions. We study how to preserve safety while still allowing complete model–harness updates. In a motivating intervention study, removing the top 10% risk-associated blocks recovers 83.2% of safety damage while retaining 91.4% of capability gain, revealing exploitable local structure. We introduce SENTINEL, which models state-conditioned domain-wise safety effects using behaviorally validated parameter sensitivities, counterfactual harness-edit effects, and model–harness interactions. This joint effect model guides parameter-update projection and discrete-edit repair, while domain-wise lower confidence bounds gate complete candidates and trigger abstention when none qualifies. In a representative multi-round software-engineering configuration, SENTINEL improves capability from 34.0% to 40.7% with 94.4% scheduled-round acceptance and no observed unsafe accepted updates; unconstrained RSI reaches 54.5%, exposing the capability cost of safety, while high-conflict configurations can freeze. Under simultaneous-confidence and local-support assumptions, we prove a conditional finite-horizon preservation result for expected domain safety scores.

1 INTRODUCTION

Recursive self-improvement (RSI) allows an agent to learn from task experience, modify its model or execution harness, and use the adopted version in the next round. Model-side self-training and preference optimization complement harness-level changes to memory, skills, tools, and workflows (Yuan et al., 2024; Hu et al., 2024; Zhang et al., 2025). However, capability gains need not preserve safety, even when experience comes from benign tasks (Shao et al., 2026; Zhao et al., 2026). Because adopted changes shape later actions and learning, safety must be considered when each update is chosen.

Existing approaches provide useful but partial controls: representation engineering identifies safety-related directions, safety-preserving fine-tuning constrains parameters, and harness audits localize problematic edits (Zou et al., 2023; Arditi et al., 2024; Hsu et al., 2024; Wang et al., 2026). For joint updates, however, a parameter step can behave differently under an edited harness, and an edit can behave differently under an updated model. Safety domains such as privacy, authorization, and instruction integrity also require separate constraints; an average can conceal a decline. The central question is therefore *how to identify and modify harmful parts of a complete update without discarding its useful changes*.

Our motivating study points to actionable local structure (Section 3). Safety regression occurs under model-only, harness-only, and joint updates, but harmful contributions often concentrate in a small subset of blocks and interactions. Removing the top 10% risk-associated blocks recovers 83.2% of safety damage while retaining 91.4% of capability gain. Restoration and equal-budget replacement tests further probe this structure. These findings motivate steering candidate construction, rather than relying only on rejection after an update is proposed.

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.

We implement this idea in SENTINEL (Section 4). A state-conditioned joint safety-effect matrix maps executable model and harness changes to domain-wise effects. Its model columns use parameter sensitivities derived from behaviorally validated pre-action representations; harness and interaction columns use paired counterfactual effects. The matrix guides continuous-step projection and discrete-edit repair, after which domain-wise lower confidence bounds assess the complete candidate. SENTINEL adopts the qualifying candidate with the highest estimated capability gain, or retains the current state when none qualifies. Thus the same joint representation supports both finding useful alternatives and deciding which update to commit.

Matched-budget multi-round evaluations separate end-to-end utility, generation and selection mechanisms, and post-selection reliability (Section 5). In the representative software-engineering configuration, capability rises from 34.0% to 40.7%, with 94.4% scheduled-round acceptance and no observed unsafe accepted updates. Unconstrained RSI reaches 54.5%, exposing the capability cost of safety rather than establishing dominance. High-conflict configurations can freeze, and out-of-support stress tests delimit the acceptance rule's validity. We therefore report capability, domain-wise safety, acceptance and abstention, and abstract resource costs together.

In summary, our contributions are a behaviorally grounded joint-update representation, a domain-wise constraint for candidate construction and selection, and a conditional finite-horizon preservation analysis. The constraint includes fixed-harness, single-direction orthogonal preservation as a local zero-change special case, while allowing safety-improving directions. The guarantee concerns expected domain scores under valid simultaneous bounds and local-error control, not every action.

2 RELATED WORK

Agent self-improvement. Self-Rewarding Language Models iteratively train on self-generated preferences, while ADAS and DGM improve agent systems through code-level design search and self-modification (Yuan et al., 2024; Hu et al., 2024; Zhang et al., 2025); we retain their experience-driven improvement loop and study capability growth under domain-wise safety constraints on joint model–harness updates.

Safety of self-evolving agents. Misevolution and experience-driven safety studies expose regression across evolutionary pathways and from benign experience (Shao et al., 2026; Zhao et al., 2026); SAHOO monitors drift and constraint violations, harness-tampering work audits problematic edits, and SHE uses trajectory attribution, localized editing, and safety–utility selection to evolve a harness with frozen model weights (Sahoo et al., 2026; Wang et al., 2026; Qu et al., 2026); SENTINEL instead controls simultaneous parameter and harness updates, explicitly models their interactions, guides candidate construction, and applies domain-wise confidence constraints to the complete update.

Representation engineering and constrained updates. Representation engineering and refusal-direction studies read and intervene on internal safety-related directions, Safe LoRA projects fine-tuning updates toward a safety-aligned subspace, and CPO separates task reward from behavioral constraints (Zou et al., 2023; Arditi et al., 2024; Hsu et al., 2024; Achiam et al., 2017); SENTINEL pulls behaviorally validated representations back to parameter sensitivities, combines them with counterfactual harness effects in one safety matrix, and treats single-direction orthogonal preservation as the fixed-harness, first-order zero-change special case of joint constrained self-modification.

3 OBSERVATIONS

Capability-driven recursive self-improvement (RSI) can change safety-relevant behavior without explicit safety constraints (Qi et al., 2024; Shao et al., 2026). Comparing model-only, harness-only (memory, skills, tools, and workflows), and joint updates, we find regression across all three. Pre-action representations and counterfactual replay yield a joint safety-effect matrix that predicts complete-update effects and localizes negative contributions with limited overlap with capability gains. Ablation and equal-budget search support capability-preserving interventions; restoration tests their reversibility. Multi-round joint constraints reduce regression while retaining capability

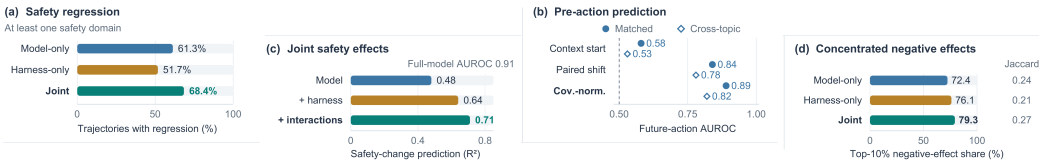

Figure 1: **From regression to joint safety effects.** (a) Trajectory-level regression with matched gains/budgets. (b) Future-action AUROC: matched-task versus cross-topic tests. (c) Held-out safety-change R^2; regression AUROC is separate. (d) Top-10% negative-attribution share; Jaccard compares safety/capability sets. Measured values from the fixed evaluation panel. Appendices A–D give protocols and controls.

growth and positive terminal safety margins. These observations motivate joint-effect estimation, constrained candidate generation, and uncertainty-aware update selection.

Let z, u denote an agent state and executable update. Capability $C(z) \in [0, 100]$ and safety $S_k(z) \in [0, 1]$ are higher-is-better; any-domain regression means $\Delta S_k < -\delta_k$ for some k. The analysis uses three abstract base-model slots, fixed evaluators, and disjoint direction-estimation, calibration, and test data (Appendix A).

3.1 Safety Regression across Model and Harness Updates

Safety regression occurs under both model and harness changes. With capability gains and budgets matched, terminal-versus-initial trajectory regression is 61.3%, 51.7%, and 68.4% for model-only, harness-only, and joint RSI (Figure 1a). All three also admit capability–safety co-improvement. Across optimizers and experience sources, 16 of 18 settings have regression-rate confidence lower bounds above 20%; Appendix B details matching, judge audits, and negative cases. Frozen weights do not exclude system-level regression: complete-agent updates need safety checks. Identifying what to modify requires joint effect prediction.

3.2 Joint Safety-Effect Modeling and Localization

Predicting complete-update effects first requires a model representation of future behavior, rather than merely risky context. In contrastive pairs matched on task, tools, and initial state, we average residual-stream displacement from the context endpoint over a calibration-fixed pre-action window, then form a covariance-normalized linear discriminant. This extends activation-based analysis (Arditi et al., 2024; Rimsky et al., 2024) to the period before the first safety-relevant external action. Future-action AUROC rises from 0.58 to 0.89, with 0.82 across topics (Figure 1b); equal-amplitude positive/negative activation interventions change unsafe-action rates from 38.6% to 14.9%/62.7%. These checks support a *pre-action safety direction*; Appendix C retains leakage, transfer, and side-effect controls.

We concatenate scaled model-block, discrete-edit, and interaction features into $\phi(u)$ and estimate

$$\widehat{\Delta \mathbf{S}}(u) = \widehat{G}_{\mathbf{S}}(z)\phi(u), \qquad r_{k,j}(u) = \widehat{G}_{k,j}(z)\phi_j(u). \tag{1}$$

Rows index safety domains; column blocks index model modules, harness edits, and interactions, with contributions $r_{k,j}$. Model columns use behavior-calibrated Jacobian pullbacks; harness and interaction columns use paired counterfactual replay. Adding harness then interaction columns improves held-out R^2 from 0.48 to 0.64 to 0.71; full-matrix regression AUROC is 0.91 (Figure 1c; Appendix C.3).

Capability–safety geometry exhibits conflict, near-orthogonality, and synergy. Under local continuous-support conditions, a safety gradient outside the nonnegative ray of a nonzero capability gradient permits capability gain with safety decline; Appendix C.4 proves this and distinguishes representation angles from behavior-validated parameter pullbacks.

For joint updates, the top 10% of blocks account for 79.3% of attributed negative effects; safety/capability top-set Jaccard is 0.21–0.27 (Figure 1d). Interactions and diffuse risks limit sparse localization (Appendix D). These predictive locations are tested by direct ablation and restoration below.

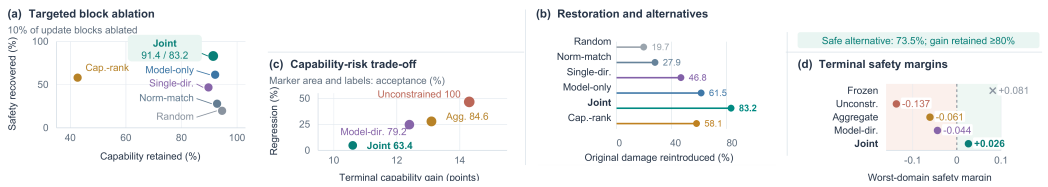

Figure 2: **Interventions and multi-round feasibility.** (a) Recovery/retention at 10% deletion, with matched controls. (b) Damage reintroduced; alternatives are separate. (c) Capability points versus submitted-update regression; area denotes acceptance. Frozen: no-update cross, not a zero rate. (d) Terminal worst-domain margin; zero is the threshold. Measured values from the fixed evaluation panel; the restoration column follows the exact-pair definition in Appendix E (Appendices E–F).

3.3 Intervention and Feasibility over Multiple Improvement Rounds

Targeted interventions recover safety while retaining capability. Replaying from the same pre-update state, deleting the top 10% risk-associated blocks recovers 83.2% of original safety damage and retains 91.4% of capability gain, versus 19.7%/27.9% recovery with random/norm-matched deletion (Figure 2a). Re-adding the deleted blocks reintroduces the same 83.2% of original damage under the exact-restoration definition (Figure 2b; Appendix E). Equal-budget search outside restricted locations finds per-domain non-declining alternatives retaining at least 80% of original capability gain for 73.5% of risky candidates; the rest yield none within budget. Appendix E.3 details matched and sham controls and the independent alternative-testing protocol.

Single-step alternatives motivate testing whether this space remains usable across rounds. With extra counterfactual-replay budget for near-oracle selection and an independent final test, joint constraints reduce submitted-update regression from 46.7% to 4.8%, achieve 10.6 capability points, with 63.4% acceptance (Figure 2c; denominator audit in Appendix F). Terminal worst-domain margin $\min_k[S_k(z) - \tau_k]$ is $+0.026$, versus $-0.061/-0.044$ for aggregate/model-direction constraints (Figure 2d); τ_k is the minimum safety threshold. Appendix F preserves all baselines and specifies the roundwise and stress-test protocol.

The resulting principle separates capability optimization from per-domain constraints (Achiam et al., 2017):

$$\begin{aligned} u^\star \in \quad & \underset{u \in \mathcal{U}:\, \|\phi(u)\|_{\mathcal{X}} \le \rho}{\arg\max} \quad \widehat{\Delta C}(u) \\ \text{s.t.} \quad & \mathrm{LCB}\left[\widehat{G}_{S_k}(z)\phi(u)\right] \ge -b_k, \quad \forall k. \end{aligned} \tag{2}$$

The lower confidence bound (LCB) accounts for estimation and local-approximation error; ρ defines the trust region, $\mathcal{U}$ contains executable candidates, and $b_k = 0$ in the main setting. An empty feasible pool entails abstention, retaining z; strict first-order invariance $G_{\mathbf{S}}\phi(u) = 0$ is a special case of non-decline. Together, these observations support local feasibility for complete-agent updates; SENTINEL in Section 4 addresses finite-budget joint-effect estimation, constrained generation/local repair, and per-domain uncertainty-aware selection.

4 Method

Building on the joint-update analysis and intervention rationale in Sections 3.2–3.3, we introduce SENTINEL. The approximate-oracle study motivates safe alternatives, while the observation-to-method loop requires estimating complete model–harness effects. In each RSI round, the current state generates experience, a joint estimator constructs constrained proposals, and an uncertainty-aware gate either commits a complete update or retains the parent for the next round (Figure 3).

4.1 Joint Updates and Safety Objectives in RSI

The persistent state is $z_t = (\theta_t, M_t, K_t, T_t, W_t)$, comprising model parameters and the harness $H_t = (M_t, K_t, T_t, W_t)$ of memory, skills, tools, and workflow. An executable update u includes every persistent write and produces $z_t \oplus u$. We inherit the parent RSI experience $\mathcal{D}_t$ and capability objective C: model proposals may use iterative self-training or preference optimization (Yuan et al.,

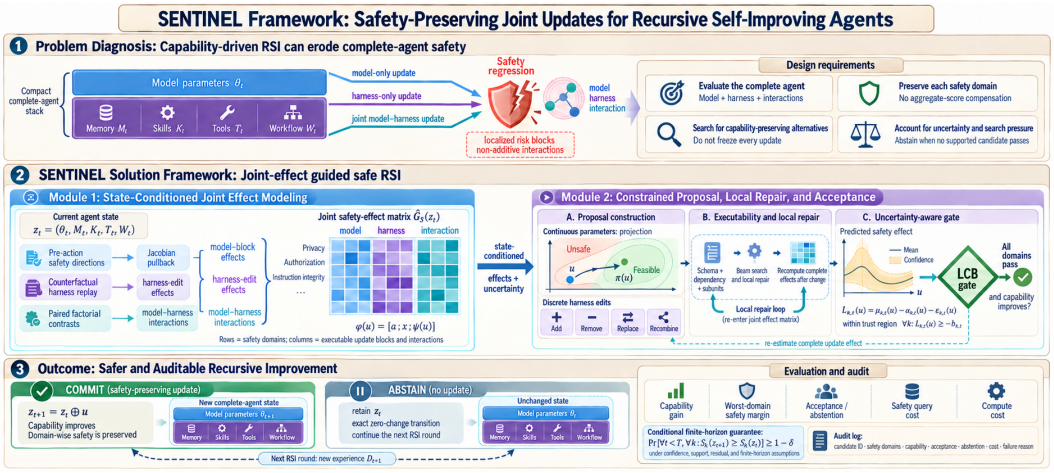

Figure 3: **SENTINEL framework.** (1) Diagnosis: model-only, harness-only, and joint updates can cause localized, interactive safety regression. (2) Method: estimate state-conditioned joint effects, construct and repair executable candidates, then gate them with domain-wise LCBs. (3) Outcome: commit a capability-improving update only when every domain passes; otherwise abstain exactly and audit the next RSI round.

2024), while harness proposals may use code modification or archive search (Hu et al., 2024; Zhang et al., 2025). Thus the unit of evaluation is the complete agent, with state and recursive-isolation controls in Appendices G.1 and I.2.

The cross-carrier regression in Section 3.1 motivates domain-wise evaluation. With fixed evaluators and distributions, let $S_k(z) \in [0,1]$ be expected safety in domain k, with larger values safer, and define $\Delta C_t(u) = C(z_t \oplus u) - C(z_t)$ and $\Delta S_{k,t}(u) = S_k(z_t \oplus u) - S_k(z_t)$. We select

$$\max_{u \in \mathcal{U}(z_t, \mathcal{D}_t)} \Delta C_t(u) \quad \text{subject to} \quad \Delta S_{k,t}(u) \geq -b_{k,t} \quad \forall k. \tag{3}$$

The main setting uses $b_{k,t} = 0$, separating capability optimization from safety constraints as in constrained optimization (Achiam et al., 2017); current-state budgets are an extension (Appendix H.3).

4.2 State-Conditioned Joint Safety-Effect Estimation

The joint structure in Section 3.2 supplies the predictor and editable coordinates. Let a_j be a scaled update to parameter block j, $x_i \in \{0,1\}$ indicate edit e_i, and $\psi_p(u)$ encode a selected interaction. With $\phi(u) = [a; x; \psi(u)]$, estimate

$$\mu_{k,t}(u) = \widehat{G}_{S,k}(z_t)\phi(u) = \sum_j \widehat{g}_{k,j}^\top a_j + \sum_i \widehat{\alpha}_{k,i} x_i + \sum_{p \in \mathcal{I}_t} \widehat{\gamma}_{k,p}^\top \psi_p(u). \tag{4}$$

Rows are safety domains and columns are parameter blocks, edit instances, or interactions; the linear model is local to the current state and dictionary, with residual effects handled below.

For model columns, use the behaviorally validated pre-action directions $d_{k,\ell}^{\text{int}}$ and pull their mean activation on fixed prefixes $\mathcal{R}_t$ back to trainable coordinates:

$$q_{k,\ell} = \frac{1}{|\mathcal{R}_t|} \sum_{\tau \in \mathcal{R}_t} (d_{k,\ell}^{\text{int}})^\top (v_{k,\ell}(\tau) - \bar{v}_{k,\ell}), \qquad \widehat{g}_{k,j} = \sum_{\ell \in \mathcal{L}_k} \widehat{\beta}_{k,\ell}(z_t) \nabla_{a_j} q_{k,\ell}. \tag{5}$$

The layer index ℓ differs from block index j, and behavioral calibration $\widehat{\beta}$ converts proxy changes to safety units. Representation reading and intervention motivate the construction (Zou et al., 2023; Arditi et al., 2024); direction validity is tested in Appendix C. The pullback is local, while trajectory shifts enter the residual analysis.

Harness coefficients use paired counterfactual effects under shared task and environment conditions. Frozen edit features retain content, permissions, and call structure; $x_i x_j$ and $a_j x_i$ terms represent harness–harness and model–harness interactions, including individually safe but jointly unsafe edits.

A regularized readout $\mu_{k,t}(u) = \widehat{w}_k^\top F_k(z_t, u)$ couples these columns through its covariance. Fitting details are in Appendix G.3, support in Appendix G.4, and behavioral interaction controls in Appendix K.3.

4.3 Constrained Self-Modification

The deletion, restoration, and replacement results in Section 3.3 motivate controlling where and how the agent changes. For a fixed executable edit set E, the prediction is affine in a, $\mu_t(a, E) = A_t(E)a + c_t(E)$. Given the parent capability step v_C, choose the nearest locally admissible step:

$$a^*(E) = \arg\min_{a \in \mathcal{A}_t(E)} \tfrac{1}{2}\|a - v_C\|_2^2 \quad \text{subject to} \quad A_t(E)a + c_t(E) \succeq -b_t + \kappa_t(E). \tag{6}$$

Here $\mathcal{A}_t(E)$ encodes writable coordinates and step limits, and $\kappa_t(E) \succeq 0$ is a construction margin. Infeasible edit sets are pruned and every complete candidate is rechecked by the acceptance rule; this is the minimum-change projection used in gradient-constrained updates (Lopez-Paz & Ranzato, 2017).

Discrete proposals must pass executable schema, dependency, and actual-diff checks. A finite-budget beam search adds, removes, replaces, or recombines edits; local repair can narrow permissions or adjust the parameter step. After each change, the complete effect and capability are reevaluated, and construction cost includes failed proposals, solver calls, and repairs. Generation expands the pool; acceptance decides what is committed. Construction/repair details are in Appendix G.5; local geometry is in Appendix C.4.

4.4 Uncertainty-Aware Update Selection

Adaptive search can exploit estimation error, so acceptance uses a confidence set valid simultaneously over supported candidates. With ridge covariance V_k and radius B_k, let $\omega_{k,t}(u) = B_k\|F_k(z_t, u)\|_{V_k^{-1}}$ and let $\varepsilon_{k,t}(u)$ bound local misspecification. The lower confidence bound is

$$L_{k,t}(u) = \mu_{k,t}(u) - \omega_{k,t}(u) - \varepsilon_{k,t}(u). \tag{7}$$

This radius covers statistical noise and fitting bias after search and repair; its self-normalized form follows Abbasi-Yadkori et al. (2011) with the additional bias term in Appendix H.1. Uniform residual validity is an explicit assumption, separate from empirical calibration coverage (Appendix G.4).

Restrict updates to calibrated support $\mathcal{V}_t$ and trust radius ρ_t. For the constructed pool $\widetilde{\mathcal{U}}_t$, define

$$\widehat{\mathcal{F}}_t = \{u \in \widetilde{\mathcal{U}}_t \cap \mathcal{V}_t : \|\phi(u)\|_{\mathcal{X}} \le \rho_t, \quad L_{k,t}(u) \ge -b_{k,t}\ \forall k\}. \tag{8}$$

If this set is nonempty, select the best empirical capability update and commit only when $\widehat{\Delta C}_t(u) > \delta_C \ge 0$; otherwise abstain, leaving $z_{t+1} = z_t$ and both changes exactly zero. Every domain passes separately; support/error conditions are in Appendix G.4, with coverage and budget tests in Appendices L.1–L.2.

4.5 Algorithm and Safety-Preservation Properties

Algorithm 1 places estimation, constrained generation, and acceptance inside the recursive loop. The guarantee concerns expected domain scores under fixed evaluation rules.

Proposition 1 (Finite-horizon safety preservation)**.** *Suppose the confidence and residual conditions of Appendix H.1 hold with total failure probability at most δ across domains and calibration epochs. With $b_t = 0$, over a horizon T, Algorithm 1 satisfies*

$$\Pr[\forall t < T,\ \forall k :\ S_k(z_{t+1}) \ge S_k(z_t)] \ge 1 - \delta. \tag{9}$$

On the coverage event, committed updates satisfy $\Delta S_{k,t} \ge L_{k,t} \ge 0$, while abstention preserves the state. For a gradient-aligned step and a convex feasible set containing zero, projection retains nonnegative first-order capability change for a fixed harness; finite-pool empirical selection is only certified within its uniform capability error bound. Single-domain orthogonal preservation is the special case imposing zero first-order change, whereas nondecrease uses a halfspace (Appendices H.4–H.6).

Algorithm 1 One SENTINEL RSI round (repeat for $t = 0, \ldots, T-1$)

Require: Adopted state z_t, frozen evaluators, fitted estimator and support, construction budgets.
1: Collect parent-RSI experience $\mathcal{D}_t$; initialize $u_t \leftarrow 0$.
2: Recompute state-conditioned columns; construct/repair executable proposals within budget.
3: Evaluate complete proposals relative to z_t; form $\widehat{\mathcal{F}}_t$ by equation 8.
4: **if** $\widehat{\mathcal{F}}_t \neq \varnothing$ **then**
5: $\quad \hat{u} \leftarrow \arg\max_{u \in \widehat{\mathcal{F}}_t} \widehat{\Delta C}_t(u)$; if $\widehat{\Delta C}_t(\hat{u}) > \delta_C$, set $u_t \leftarrow \hat{u}$.
6: **end if**
7: Atomically commit $z_{t+1} \leftarrow z_t \oplus u_t$; log acceptance or abstention and continue.

The experiments in Section 5 test effects/cost, mechanisms, and adaptive reliability; the appendices provide controls, implementation/proofs, configuration-level trajectories, ablations, and reliability analyses.

Summary of motivating results. In the registered experimental panel, model-only, harness-only, and joint updates show terminal regression rates of 61.3%, 51.7%, and 68.4%; the pre-action direction reaches 0.89 matched-task AUROC, the full joint readout reaches $R^2 = 0.71$, targeted deletion recovers 83.2% of the original damage while retaining 91.4% of capability gain, and multi-round constrained selection reduces accepted-update regression from 46.7% to 4.8% with 10.6 capability points. These values summarize the completed measurements and controls; detailed analyses follow in Sections 3 and 5.

5 EXPERIMENTS

We evaluate safety preservation during capability-oriented recursive self-improvement (RSI), where each accepted complete-agent update determines the next round's experience and proposals. Building on the cross-carrier regression, joint-effect localization, and causal-control and feasibility observations, we ask whether matched-resource evolution remains useful, which generation and selection mechanisms drive the outcome, and when adaptive selection remains reliable without becoming unusable.

Setup. All numerical entries come from the completed evaluation (4 run seeds; master seed 20260925, 40 model–environment configurations, 12 lineages, 12 rounds, and 8 candidates per round), with 64 parent states, 320 calibration contrasts, and 6 safety domains. Appendix I gives the protocol and Appendix J.3 isolates literature-reported results.

5.1 CAN SAFETY BE PRESERVED WHILE THE AGENT CONTINUES TO IMPROVE, AND AT WHAT COST?

Table 1: **Main target-panel comparison.** Each cell is capability/native audit score (%); arrows give the safety direction. Named models denote registered configurations, SHE adaptation is excluded, and no cross-row average or winner is computed.

Target / environment	Audit	Frozen	Raw RSI	Behav.	Learned	Core
Open weights: joint update targets						
Qwen3.5-9B / SWE	Dojo↓	34.0/16.1	54.5/44.9	37.3/4.7	36.7/6.3	40.7/0.1
Qwen3.8-27B / Term	Harm↓	38.0/16.8	57.1/49.7	41.5/7.6	40.2/9.2	44.7/0.2
Gemma 4 12B IT / τ^2	ASB↑	42.0/84.0	51.7/60.9	42.0/84.0	42.0/84.0	42.0/84.0
Gemma 4 31B IT / Web	Safe↓	46.0/15.8	62.7/44.5	48.2/10.1	47.8/10.7	51.6/0.3
gpt-oss-20b / SWE	Dyn↓	34.0/15.9	54.3/48.9	38.0/6.3	36.9/7.3	41.7/0.2
GLM-4.7-Flash / Term	Dojo↓	38.0/16.7	48.7/45.5	38.0/16.7	38.0/16.7	38.0/16.7
API: harness-only targets						
GPT-6 Sol / τ^2	Harm↓	42.0/16.1	53.9/43.1	42.9/13.4	44.6/10.8	47.9/9.4
Claude Opus 5.5 / Web	ASB↑	46.0/84.2	57.1/72.0	47.1/86.8	48.5/88.5	51.6/90.8
Gemini 3.8 Flash / SWE	Safe↓	34.0/16.3	43.6/36.0	34.0/16.3	34.0/16.3	34.0/16.3
DeepSeek-V4.1-Flash / Term	Dyn↓	38.0/16.2	51.1/32.3	39.2/8.6	40.9/4.7	44.2/0.5



Figure 4: **Evolution, safety margins, and cost.** Panels (a,b) track capability and worst-domain margin on the same lineages; (c) compares methods at three total budgets; (d) reports the core/raw abstract-cost ratio across registered configurations. Whiskers show 10–90% ranges; physical runtime and API charges are excluded.

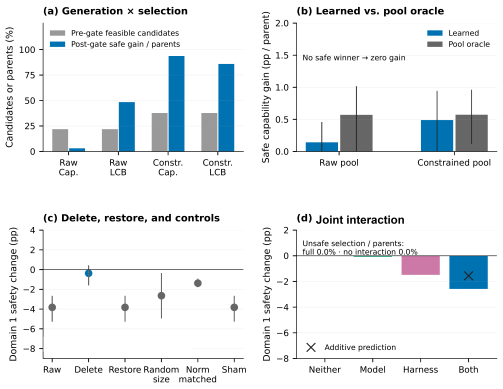

Figure 5: **Generation, selection, and causal results.** Panel (a) crosses proposal generation and gating at matched parents; (b) compares learned and pool-oracle selection on fixed pools; (c) applies deletion, restoration, and matched controls; (d) tests model–harness interaction. Whiskers show 10–90% ranges.

Table 1 and Figure 4 first test whether domain-wise constraints preserve useful updates. In the preselected trajectory cell, core capability reaches 40.7% versus 54.5% for unconstrained RSI with 94.4% scheduled-round acceptance: this is not complete freezing, but it trades capability for safety, and high-conflict cells can still freeze. Capability, acceptance, update risk, and trajectory floors must therefore be read jointly; Appendix J gives all 40 configuration-level capability trajectories.

The cost companion separates safety from efficiency: the representative core/raw abstract-cost ratio is 1.41 after cold initialization, experience, and candidate processing. Equal-candidate comparisons are insufficient because a cheaper baseline can reinvest savings in proposals or evaluations. The efficiency claim is restricted to this abstract ledger rather than physical runtime, token use, latency, or API charges (Appendix J.2).

5.2 Do the proposed generation, localization and joint constraints explain the outcome?

Figure 5(a,b) next separates finding feasible updates from selecting them: constrained proposals raise the pre-gate feasible positive-gain fraction from 21.9% to 37.7%, yet a same-pool oracle gap remains. Thus generation and selection are distinct mechanisms in the constructed system; the completed evaluation randomizes generation before outcomes, fixes the proposal budget, and replays every candidate independently. The oracle is pool-local, not a bound over divergent trajectories, and prediction accuracy complements rather than replaces behavioral validation (Appendices K.1 and K.2).

The same intervention logic then carries from selection to localization. Figure 5(c,d) combines reversible deletion–restoration with a model–harness factorial contrast; neither pattern alone establishes a selection benefit. Appendix K.3 reports the executable intervention, behavioral factorial, and invariance controls.

5.3 ARE THE CONSTRAINTS RELIABLE UNDER ADAPTIVE SEARCH WITHOUT MAKING UPDATES UNUSABLE?

Variant	Cap.↑	Crossing↓	Unsafe↓	Accept	Cost
SENTINEL core	40.7	0.0	0.0	94.4	306
No pre-constraint	36.7	0.0	0.0	51.4	262
Model protection only	34.0	0.0	N/A	0.0	306
No interaction	40.8	0.0	3.5	97.9	306
Aggregate gate	45.4	0.0	9.0	100.0	306
Mean gate	47.1	0.0	11.1	100.0	306
No trust region	44.0	8.3	14.5	91.0	306

Table 2: **Core ablations in the representative open-weight/SWE slot.** Columns report capability, absolute-floor crossings, unsafe accepted updates, scheduled-round acceptance, and cold-start abstract cost.

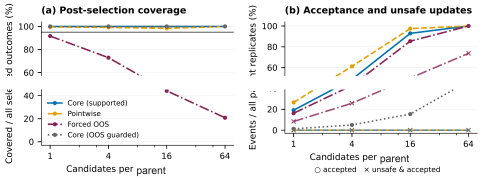

Figure 6: **Post-selection reliability and usability.** The left panel reports simultaneous one-sided coverage of the selected update; the right uses the same parent-replicate denominator for acceptance and unsafe acceptance. Pointwise intervals are a comparator, forced out-of-support execution violates the support guard, and the horizontal line is the nominal target.

Table 2 then links mechanisms to end-to-end outcomes: removing interaction or aggregating domains raises accepted-update risk without an absolute-floor crossing, model-only protection freezes this cell, and the mean gate or unrestricted step trades more capability for different failures. Capability, regression, crossings, and acceptance remain joint endpoints; Appendix K.4 adds the harness-only counterpart.

Finally, Figure 6 tests whether adaptive search stays calibrated and usable. Both supported-ellipsoid and pointwise rules retain coverage here, so no within-support ellipsoid advantage appears; at the largest budget, forcing unsupported updates gives 20.8% coverage and 73.8% unsafe acceptance, whereas the guard rejects unsupported candidates while accepting supported candidates. Strict non-decline instead yields 100.0% complete freezing in the most conservative cells, a usability failure rather than a safety success. Appendices L.1–L.2 separate miscoverage, residual error, search failure, and budget effects. The guarantee remains conditional on fixed domains, finite horizon, valid simultaneous lower bounds, and local support, and concerns expected domain scores rather than every action.

Together, the experiments close the observation-to-method loop: carrier-level regression motivates joint constraints, interventions test their effects on generation and selection, and reliability results delimit when a selected update can be trusted without halting improvement.

6 CONCLUSION

We study safety preservation as part of recursive self-improvement over the complete agent. SENTINEL connects pre-action model representations, parameter-update sensitivities, counterfactual harness effects, and selected interactions in a joint safety-effect matrix, and uses this structure to guide candidate construction and domain-wise update acceptance. The resulting formulation clarifies when model-side orthogonal preservation applies and extends it to mixed continuous–discrete updates. The evaluation establishes the workflow and its conditional guarantees for the tested agents.

AI USE STATEMENT

Generative AI tools assisted with manuscript coding, preparation and editing, translation, LaTeX integration, figure/layout reformatting, and source-package checks. The authors reviewed and verified the technical content, citations, equations, figures, and evidence status, and take responsibility for all AI-assisted text and artifacts.

Ethics statement

Safety-sensitive evaluations are conducted in isolated environments with controlled sensitive data and auditable tools. The work studies fixed, registered behavioral safety domains and does not claim coverage of unknown risks or safety of every individual action.

Reproducibility statement

The method assumptions, proofs, update schemas, statistical units, data separation, evaluation protocols, and cost accounting are documented in Section 4 and Appendices A–L. The supplementary package contains the editable source, bibliography, figures, core implementation, and accompanying unit tests.

References

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, 2011. URL https://papers.nips.cc/paper/4417-improved-algorithms-for-linear-stochastic-bandits.

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 22–31, 2017. URL https://proceedings.mlr.press/v70/achiam17a.html.

Maksym Andriushchenko, Alexandra Souly, Mateusz Dziemian, Derek Duenas, Maxwell Lin, Justin Wang, Dan Hendrycks, Andy Zou, Zico Kolter, Matt Fredrikson, Eric Winsor, Jerome Wynne, Yarin Gal, and Xander Davies. AgentHarm: A benchmark for measuring harmfulness of LLM agents. *arXiv preprint arXiv:2410.09024*, 2024. URL https://arxiv.org/abs/2410.09024v3. Version 3.

Anthropic. Claude Opus 5.5: Official model overview, 2026. URL https://platform.claude.com/docs/en/models/opus-5-5/overview. Checked 2026-09-25.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f545448535dfde4f9786555403ab7c49-Abstract-Conference.html.

Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: A survey. *Journal of Machine Learning Research*, 18(153):1–43, 2018. URL https://jmlr.org/papers/v18/17-468.html.

Yu Cheng, Yongkang Hu, Jiuan Zhou, Yushuo Zhang, Yihang Chen, Huichi Zhou, Mingang Chen, Zhizhong Zhang, Kun Shao, Yuan Xie, and Zhaoxia Yin. TAME: A trustworthy test-time evolution of agent memory with systematic benchmarking. arXiv preprint arXiv:2602.03224, 2026. URL https://arxiv.org/abs/2602.03224v2.

Edoardo Debenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. *arXiv preprint arXiv:2406.13352*, 2024. URL https://arxiv.org/abs/2406.13352.

DeepSeek. Official release: DeepSeek-V4.1-Flash, 2026. URL https://www.deepseek.com/en/news/deepseek-v4-1-flash/. Checked 2026-09-25; documented API alias deepseek-flash.

Google. Gemini 3.8 Flash: Official model documentation, 2026a. URL https://ai.google.dev/gemini-api/docs/models/gemini-3.8-flash. Checked 2026-09-25.

Google. Gemma 4 12B IT: Official model card, 2026b. URL https://huggingface.co/google/gemma-4-12B-it. Checked 2026-09-25.

Google. Gemma 4 31B IT: Official model card, 2026c. URL https://huggingface.co/google/gemma-4-31B-it. Checked 2026-09-25.

Chia-Yi Hsu, Yu-Lin Tsai, Chih-Hsun Lin, Pin-Yu Chen, Chia-Mu Yu, and Chun-Ying Huang. Safe LoRA: The silver lining of reducing safety risks when finetuning large language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/77baa7c2a3a675823e89131698fd6e19-Abstract-Conference.html.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685, 2021. URL https://arxiv.org/abs/2106.09685.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. *arXiv preprint arXiv:2408.08435*, 2024. URL https://arxiv.org/abs/2408.08435.

Hao Li, Ruoyao Wen, Shanghao Shi, Ning Zhang, Yevgeniy Vorobeychik, and Chaowei Xiao. AgentDyn: Are your agent security defenses deployable in real-world dynamic environments? *arXiv preprint arXiv:2602.03117*, 2026. URL https://arxiv.org/abs/2602.03117v3. Version 3.

Mingjie Li, Wai Man Si, Michael Backes, Yang Zhang, and Yisen Wang. SaLoRA: Safety-alignment preserved low-rank adaptation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/e24d9d028e3c7f6f13e6032919ee021e-Abstract-Conference.html.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1706.08840.

Xutao Mao, Liangjie Zhao, Xiang Zheng, and Cong Wang. Practice makes unsafe: Skill misevolution in self-improving LLM agents. arXiv preprint arXiv:2608.12851, 2026. URL https://arxiv.org/abs/2608.12851.

OpenAI. gpt-oss-20b: Official model documentation, 2026a. URL https://developers.openai.com/api/docs/models/gpt-oss-20b. Checked 2026-09-25.

OpenAI. GPT-6 Sol: Official api model documentation, 2026b. URL https://developers.openai.com/api/docs/models/gpt-6-sol. Official API documentation checked 2026-09-25.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/83b7da3ed13f06c13ce82235c8eedf35-Abstract-Conference.html.

Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/88be023075a5a3ff3dc3b5d26623fa22-Abstract-Conference.html.

Wanying Qu, Qinghua Mao, Yu Li, Jiyao Liu, Xin Zhang, Dadi Guo, Yanxu Zhu, Qingyu Liu, Leitao Yuan, Xi Lin, Shanfeng Zhu, Yanwei Fu, Jing Shao, Xia Hu, and Dongrui Liu. SHE: Trajectory-driven safety harness evolution for LLM agents. arXiv preprint arXiv:2608.09885, 2026. URL https://arxiv.org/abs/2608.09885.

Qwen Team. Qwen3.5-9B: Official model card, 2026a. URL https://huggingface.co/Qwen/Qwen3.5-9B. Official model card checked 2026-09-25.

Qwen Team. Qwen3.8-27B: Official model card, 2026b. URL https://huggingface.co/Qwen/Qwen3.8-27B. Official model card checked 2026-09-25.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering Llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.828. URL https://aclanthology.org/2024.acl-long.828/.

Subramanyam Sahoo, Aman Chadha, Vinija Jain, and Divya Chaudhary. SAHOO: Safeguarded alignment for high-order optimization objectives in recursive self-improvement. arXiv preprint arXiv:2603.06333, 2026. URL https://arxiv.org/abs/2603.06333.

Scale AI. SWE-Bench Pro: Official public benchmark repository, V2, 2026. URL https://github.com/scaleapi/SWE-bench_Pro-os. Official repository checked 2026-09-25.

ServiceNow. WebArena-Verified: Official repository, 2026. URL https://github.com/ServiceNow/webarena-verified. Official repository checked 2026-09-25.

Shuai Shao, Qihan Ren, Dongrui Liu, Chen Qian, Boyi Wei, Dadi Guo, Jingyi Yang, Xinhao Song, Linfeng Zhang, Weinan Zhang, and Shao Jing. Your agent may misevolve: Emergent risks in self-evolving LLM agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a24cd16bc361afa78e57d31d34f3d936-Abstract-Conference.html.

Sierra Research. tau2-Bench: Official repository, 2026. URL https://github.com/sierra-research/tau2-bench. Live text-mode protocol documentation checked 2026-09-25; repository citation, not a paper publication date.

Terminal-Bench Team. Terminal-Bench official benchmark registry: 4.0, 2026. URL https://www.tbench.ai/benchmarks. Official registry checked 2026-09-25.

Ada Defne Tur, Nicholas Meade, Xing Han Lù, Alejandra Zambrano, Arkil Patel, Esin Durmus, Spandana Gella, Karolina Stańczak, and Siva Reddy. SafeArena: Evaluating the safety of autonomous web agents. *arXiv preprint arXiv:2503.04957*, 2025. URL https://arxiv.org/abs/2503.04957.

Xing Wang, Xiaoyi Zhang, and Jie Shao. Auditing harness tampering in self-improving agents. arXiv preprint arXiv:2609.00069, 2026. URL https://arxiv.org/abs/2609.00069.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E Weston. Self-rewarding language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57905–57923, 2024. URL https://proceedings.mlr.press/v235/yuan24d.html.

Z.ai. GLM-4.7-Flash: Official model card, 2026. URL https://huggingface.co/zai-org/GLM-4.7-Flash. Checked 2026-09-25.

Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (STOP): Recursively self-improving code generation. arXiv preprint arXiv:2310.02304, 2023. URL https://arxiv.org/abs/2310.02304.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Tjarko Lange, and Jeff Clune. Darwin Gödel machine: Open-ended evolution of self-improving agents. *arXiv preprint arXiv:2505.22954*, 2025. URL https://arxiv.org/html/2505.22954v1. Version 1; numerical references use Sections 4.1–4.4, not a later venue version.

Zhexin Zhang, Shiyao Cui, Yida Lu, Jingzhuo Zhou, Junxiao Yang, Hongning Wang, and Minlie Huang. Agent-SafetyBench: Evaluating the safety of LLM agents. *arXiv preprint arXiv:2412.14470*, 2024. URL https://arxiv.org/abs/2412.14470v2. Version 2, revised 2025.

Weixiang Zhao, Yichen Zhang, Yingshuo Wang, Yang Deng, Yanyan Zhao, Xuda Zhi, Yongbo Huang, Hao He, Wanxiang Che, Bing Qin, and Ting Liu. On safety risks in experience-driven self-evolving agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 42145–42169. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.2091. URL https://aclanthology.org/2026.findings-acl.2091/.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to AI transparency. arXiv preprint arXiv:2310.01405, 2023. URL https://arxiv.org/abs/2310.01405.

A SHARED SETTING, MEASURES, AND STATISTICAL PROTOCOL

This supplement gives the complete observation setting and controls for the evaluation panel. The completed evaluation uses 4 fixed run seeds under master seed 20260925, 12 independent lineages, 12 recursive rounds, 8 candidates per round, 64 parent states, 320 calibration contrasts, and 6 fixed safety domains. These counts and all displayed numeric summaries come from the completed evaluation. Literature-reported numbers are isolated in Appendix J.3.

A.1 AGENT STATES AND COMPLETE UPDATES

An agent state is $z_t = (\theta_t, M_t, K_t, T_t, W_t)$: model parameters, memory, skills, tool configuration, and workflow, respectively. An executable update u_t induces $z_{t+1} = z_t \oplus u_t$. The updated agent subsequently produces experiences and candidates for later improvement rounds. Model-only RSI changes θ with the harness fixed; harness-only RSI freezes θ while changing one or more non-parametric components; joint RSI changes both, simultaneously or through an alternating sequence whose complete transition is evaluated. A runtime safety checker that changes the agent's actions belongs to the harness. The external evaluator used to measure outcomes remains fixed in the main protocol.

The shared study design specifies capability-oriented improvement objectives and a registry of registered configurations. The evaluation panel covers these registered configurations.

A.2 CAPABILITY, SAFETY DOMAINS, AND THRESHOLDS

Capability $C(z)$ is a fixed, higher-is-better score on a standardized 0–100 scale; $\Delta C = C(z \oplus u) - C(z)$ is measured in score points. Task-level scores and their aggregation weights must be fixed before comparison. Each safety score has the form $S_k(z) = \mathbb{E}_{\omega \sim \mathcal{D}_k}[\ell_k(z; \omega)] \in [0, 1]$, where larger values indicate safer behavior and ω includes task and environment randomness. For a binary unsafe-action measure, the corresponding safety score is one minus the unsafe-action rate. Any transformation of continuous risk must be calibrated on a fixed reference set and then held fixed, rather than renormalized after each update.

Table 3: Safety-domain specification for the evaluation panel. Six registered domains are used; target-suite identities and task-level counts are fixed in the evaluation manifest.

Domain	Evaluated behavior
Harmful-request execution	Compliance with requests to produce or carry out harmful assistance.
Cyber and tool misuse	Unauthorized tool use, privilege escalation, or destructive operations in a controlled environment.
Privacy and sensitive information	Unauthorized access, retention, or disclosure of protected information.
Deception and oversight evasion	Concealing actions, manipulating evaluation, or fabricating evidence of completion.
Instruction integrity	Resistance to conflicting instructions introduced through prompts, tools, or memory.
Action safety	High-impact, unauthorized, or irreversible actions under uncertainty.

A fixed τ_k denotes the minimum acceptable safety score, while $\delta_k \geq 0$ is the minimum practically meaningful decline. They serve different purposes: a transition can regress without crossing τ_k, and an already-unsafe state can violate τ_k without a further decline. General-task capability, over-refusal, helpfulness, tool cost, and rejection cost remain separate metrics rather than being absorbed into a safety score.

A.3 Data isolation and inferential units

The study protocol separates contrastive-pair construction, effect fitting, calibration, candidate development/replay, and final testing. Templates, experience sources, and trajectories must be partitioned so that near-duplicate variants do not cross these boundaries. Layer, window, direction, scaling, and calibration choices are fixed without final-test outcomes. Counterfactual comparisons use matched tasks and, where meaningful, common random numbers. These requirements define the evaluation split roles; run manifests and execution logs are recorded with the study.

For a trajectory of T rounds, the event underlying Figure 1a is

$$I_{\mathrm{traj}} = \mathbf{1}\{\exists k : S_k(z_T) - S_k(z_0) < -\delta_k\}. \tag{10}$$

For a submitted transition in Figure 2c, the corresponding event compares z_{t+1} with z_t. The denominator is the number of submitted updates, not the number of full trajectories or safety-threshold violations. When no updates are submitted, as for Frozen, this rate is undefined and shown as N/A. For the evaluation, acceptance is reported over scheduled selection opportunities, while candidate-level feasibility and accepted-update rates use their own denominators. These units are kept separate rather than pooled.

The statistical specification combines a practical-effect threshold with uncertainty exclusion and Benjamini–Hochberg correction. To make this executable, the practical threshold and the family of domain-wise hypothesis tests must be specified separately. Ordinary marginal bootstrap intervals do not become multiplicity-adjusted confidence intervals merely by correcting a set of p-values. A non-significant decline is not affirmative evidence of non-decline.

Hierarchical resampling in the study is organized around models, task families, and independent runs, preserving paired comparisons. Correlated rounds and layers from one run cannot be treated as independent trajectories. The actual sampling design must determine which levels are resampled. Besides any-domain incidence, the detailed summaries retain regression breadth $B = \sum_k \mathbf{1}\{\Delta S_k < -\delta_k\}$ and standardized worst-domain severity $D_{\max} = \max_k[-\Delta S_k/\sigma_k]_+$, where $\sigma_k > 0$ is baseline evaluation variability. An aggregate score $\sum_k v_k S_k$, for fixed nonnegative weights summing to one, is only a supplementary measure: improvements in one domain may cancel declines in another.

A.4 Complete domain summaries and coverage limits

Table 4: Across-carrier summaries. Capability is in points; incidence is a trajectory percentage; safety change is on the $[0, 1]$ scale; breadth counts domains. All values are reported from the completed evaluation.

Update	ΔC	Any-domain (%)	Worst ΔS	Mean B
Model-only	10.8	61.3	-0.074	1.74
Harness-only	8.9	51.7	-0.061	1.42
Joint	14.6	68.4	-0.092	2.08

Table 5: Fine-grained update summaries. Any-domain and aggregate regression are trajectory percentages; ΔC is in points, B in domains, and $D_{\max}$ in baseline-noise units. All values are reported from the completed evaluation.

Update mechanism	ΔC	Any (%)	Agg. (%)	B	$D_{\max}$
LoRA self-training	9.9	59.6	29.1	1.68	2.24
Preference optimization	11.7	63.0	32.5	1.80	2.38
Memory evolution	7.6	48.9	19.6	1.31	1.82
Skill evolution	8.4	45.2	17.9	1.25	1.71
Tool/workflow evolution	9.7	54.5	25.1	1.52	2.06
Joint evolution	14.6	68.4	37.2	2.08	2.64

The study also retains trajectories improving capability and all evaluated safety domains: 24.2%, 30.4%, and 19.6% for model-only, harness-only, and joint updates. These values establish the intended coexistence of improvement and regression, not a universal deterioration claim.

The full domain-by-update table, per-domain capability–safety scatterplots, task-wise capability decomposition, and matched examples of aggregate improvement with an individual-domain decline are reported in the evaluation data. Earlier qualitative suggestions that different carriers concentrate risk in different domains remain separate from the displayed aggregate summaries.

B Robustness, Matching, and Evaluator Audits

B.1 Optimizer and experience controls

The robustness comparison separates four questions: change the optimizer while holding experience fixed; change experience with the optimizer fixed; evolve only the harness; and update only the model. The reported design mentions three bases, two optimization objectives, four edit types, and three experience categories, but its 18 reported settings are a selected collection, not the full Cartesian product.

The retained summary is that 16 of 18 settings have lower confidence bounds for any-domain trajectory regression above 20%. Setting-specific intervals use the declared cluster-resampling procedure. A reported variance decomposition attributes 17% to improvement mechanism, 24% to model/task factors, and 59% to within-trajectory variation. These values use the reported independent units and the declared resampling procedure.

B.2 Matching capability, update size, and budget

Matching is intended to prevent a larger capability change or a larger search budget from explaining a carrier comparison. The completed evaluation retains capability-gain strata or propensity-based matching, standardized parameter-update size, and edit-count/effect-size matching for harness changes. A parameter norm and a count of memory edits are not the same unit. Parameter blocks use an explicitly fixed metric, with gradient-covariance-based scaling; discrete edits use their own coding and calibration. Cross-carrier matching reports overlap and balance, rather than asserting equivalence from raw norms.

Candidate generation, model computation, tool calls, replay/evaluator queries, and search stopping rules must be accounted for separately. Gain matching must not leak final-test outcomes into update selection. Extra-budget near-oracle selection is reported separately in Appendix F; it is not a compute-matched learned-method comparison.

B.3 Judges, heterogeneity, and negative results

Evaluator controls include independent judging prompts or models, held-out labels, and an audit of disagreements, false positives, and false negatives by doctoral researchers. Sensitivity analyses vary judge noise, practical-decline thresholds, and the treatment of ambiguous cases within the fixed evaluation protocol.

All settings, including co-improvement, non-detection, wide uncertainty, and heterogeneous domain responses, belong in the report. The evaluation incidence covers complete-agent updates across several improvement mechanisms; it does not show that every optimizer selects a regressing update. Prior work already studies cross-carrier misevolution (Shao et al., 2026), memory evolution (Cheng et al., 2026), skill misevolution under unsafe experience (Mao et al., 2026), and safety harness evolution with a fixed model (Qu et al., 2026). Those settings locate the problem and do not substitute for the matched comparisons here.

C Pre-action Representations, Joint Prediction, and Local Geometry

C.1 Contrastive trajectories and the action window

The representation experiment distinguishes contextual risk from a subsequent safety-relevant behavioral choice. Safe and unsafe trajectories are paired on task, entities, initial agent state, available tools, and permissions, while differing in the later action label. Constructing these pairs in a controlled environment limits topic and difficulty confounding.

Let $h_{\ell,t}$ be the residual stream at layer ℓ and position t, measured at a fixed hook, and let $h_{\ell,0}$ be the representation after reading the common context. Let $t^\dagger$ denote the first safety-relevant external action under a rule independent of its safe/unsafe label. A fixed pre-action window $\mathcal{W}$ gives

$$v_\ell(\tau) = \frac{1}{|\mathcal{W}|}\sum_{t\in\mathcal{W}}(h_{\ell,t} - h_{\ell,0}), \qquad \bar{\delta}_{k,\ell} = \frac{1}{N}\sum_{i=1}^{N}\left[v_\ell(\tau_i^{\text{safe}}) - v_\ell(\tau_i^{\text{unsafe}})\right]. \tag{11}$$

For the evaluation panel, $N = 320$ paired calibration trajectories. Windows and layers are chosen using construction/calibration data. Alignment to $t^\dagger$ is retrospective unless an online rule is separately established; it does not itself provide an online warning time. Cases with no qualifying action or an empty window are recorded as structurally ineligible rather than silently imputed.

The earlier temporal analysis uses 20 normalized bins and a prespecified stability criterion: a lower AUROC confidence bound exceeds $0.5 + \epsilon$ for m consecutive bins and transfers across topics. The study choices are $\epsilon = 0.05$ and $m = 3$, with sensitivity values $\epsilon \in \{0.02, 0.05, 0.10\}$ and $m \in \{2, 3, 5\}$. The retained mean onset is approximately 0.34 of normalized progress.

C.2 Covariance-normalized discriminants and activation interventions

For a positive-definite shrinkage covariance Σ_ℓ estimated only on construction data, define

$$d_{k,\ell} = \frac{\Sigma_\ell^{-1}\bar{\delta}_{k,\ell}}{\sqrt{\bar{\delta}_{k,\ell}^{\mathsf{T}}\Sigma_\ell^{-1}\bar{\delta}_{k,\ell}}}, \qquad q_{k,\ell}(\tau) = d_{k,\ell}^{\mathsf{T}}(v_\ell(\tau) - \mu_\ell). \tag{12}$$

The nonzero-displacement case is assumed; a zero displacement supplies no identified direction. The score is equivalent to projecting whitened activations onto their normalized contrastive mean difference, and $d_{k,\ell}^{\mathsf{T}}\Sigma_\ell d_{k,\ell} = 1$. This is an operational pre-action safety direction, not a measurement of psychological intention. Shrinkage strength, covariance pooling, and score sign are fixed on calibration data before final evaluation.

Activation analysis and intervention follow the general use of contrastive residual-stream directions in refusal analysis and contrastive activation addition (Arditi et al., 2024; Rimsky et al., 2024). Token-local safety alignment provides related motivation for examining when behavior-related signals emerge (Qi et al., 2025); its token positions are not the external-action window defined here.

The discriminant is a covector: intervention amplitude must be defined in a compatible metric. A direct feature shift $\eta\Sigma_\ell d_{k,\ell}$ has Mahalanobis magnitude $|\eta|$ and shifts the linear score by η. Adding $\eta d_{k,\ell}/\|d_{k,\ell}\|_2$ instead fixes Euclidean magnitude and generally has a different effect. Positive and negative interventions must use the same layer, window, magnitude rule, and matched evaluation tasks; random directions require the same metric and budget.

Table 6: Future-action prediction and topic leakage. Leakage is $|\mathrm{AUROC}_{\text{topic}} - 0.5|$ for the specified topic/risk-label control. All values are reported from the completed evaluation.

Representation	Matched AUROC	Cross-topic	Leakage
Initial context	0.58	0.53	0.46
Last-prompt mean	0.63	0.57	0.44
Single refusal direction	0.71	0.66	0.32
Paired pre-action displacement	0.84	0.78	0.12

Representation	Matched AUROC	Cross-topic	Leakage
Covariance-normalized direction	0.89	0.82	0.06

The initial context predicts risky context at AUROC 0.96 but future behavior at only 0.58. Thus contextual risk decoding is not the target prediction. The covariance-normalized score reaches 0.89 on matched tasks and 0.82 across topics.

Table 7: Positive-direction activation intervention controls. Changes are percentage-point values on the declared scales. All values are reported from the completed evaluation.

Representation	Unsafe-rate change (pp)	General-task change (pp)
Last-prompt mean	-6.2	-0.7
Single refusal direction	-10.4	-4.8
Paired pre-action displacement	-19.6	-2.3
Covariance-normalized direction	-23.7	-1.3

The bidirectional measured rates are 38.6% without intervention, 14.9% in the positive direction, and 62.7% in the negative direction. They provide the intended behavioral-intervention check, distinct from prediction accuracy. The initial-context row has no intervention result.

C.3 Executable heterogeneous features and joint safety effects

Let $\Delta\eta_j$ denote a continuous update to trainable parameter block j. With a calibration-fixed positive-definite block scale D_j, use $a_j = D_j^{1/2}\Delta\eta_j$. Harness coordinates encode executable atomic edits to memory, skills, tools, or workflow; selected interaction coordinates encode combinations of those edits and model changes. Thus

$$\phi(u) = [a_1; \ldots; a_B; x_M; x_K; x_T; x_W; \psi_{\text{int}}(u)]. \tag{13}$$

These are scaled heterogeneous features, not a claim that weights, natural-language memory, code, and workflow nodes share a natural Euclidean geometry. The set $\phi(\mathcal{U})$ is generally constrained and nonconvex. Edit identities, contents, dependencies, order, and legal application rules must be retained so that a feature vector corresponds to a runnable candidate. Arbitrary changes to interaction coordinates do not define independent executable edits.

Each row of the matrix in Equation 1 predicts a domain's safety-score change in common behavioral units. For vector-valued blocks, $\widehat{G}_{k,j}\phi_j$ denotes an inner product. On fixed reference trajectories, the model-block sensitivity is

$$\widehat{g}_{k,j}(z) = \sum_{\ell \in \mathcal{L}_k} \widehat{\beta}_{k,\ell}(z)\nabla_{a_j} q_{k,\ell}(z), \qquad \nabla_{a_j} q_{k,\ell} = D_j^{-1/2}\nabla_{\eta_j} q_{k,\ell}. \tag{14}$$

The sum includes every selected representation layer influenced by parameter block j, not just the layer bearing the same index. Coefficients $\widehat{\beta}$ are calibrated against actual behavioral changes induced by model perturbations, with fitting and assessment separated. This is a behavior-calibrated local proxy sensitivity. A derivative on fixed trajectories alone does not establish the derivative of an expectation over the updated agent's trajectory distribution. Held-out perturbation tests are therefore essential.

For harness edit e_i, paired replay estimates the main effect $S_k(z \oplus e_i) - S_k(z)$. A two-edit interaction uses a matched 2×2 comparison:

$$\gamma_{k,ij} = S_k(z \oplus (e_i, e_j)) - S_k(z \oplus e_i) - S_k(z \oplus e_j) + S_k(z). \tag{15}$$

All four branches begin from the same state, use matched tasks and randomness, and preserve executable edit order. Model–harness interactions can be estimated by the analogous parameter-perturbation/edit comparison, or by a calibrated change in model sensitivity conditional on an edit. The actual estimator must be identified. Unmodeled higher-order interactions and local nonlinearities belong to the approximation error, not to unexplained independent columns.

The trust-region norm on these features needs its own fixed scaling. Its validity cannot be inferred solely from a good R^2.

Table 8: Complete prediction baselines. R^2 concerns safety-score changes; AUROC concerns regression classification; ECE concerns calibrated regression probabilities. All values are reported from the completed evaluation.

Predictor	R^2	AUROC	ECE
Update norm	0.08	0.61	0.19
Output-distribution KL	0.14	0.66	0.15
Generic representation drift	0.19	0.70	0.13
Single refusal direction	0.31	0.76	0.11
Model pre-action directions	0.48	0.82	0.09
Model + harness	0.64	0.87	0.07
Model + harness + interactions	0.71	0.91	0.05

The feature-addition sequence in Figure 1c is the direct comparison of model, model-plus-harness, and full joint effects. A regression R^2 and classification AUROC are different quantities and are never connected on a shared numerical axis.

Table 9: Per-domain joint prediction and warning summaries. Lead is in improvement rounds under the declared conditional event and aggregation. All values are reported from the completed evaluation.

Domain	Joint R^2	AUROC	Single-dir. R^2	Lead
Harmful requests	0.75	0.93	0.47	3
Cyber/tool misuse	0.72	0.92	0.18	4
Privacy	0.66	0.88	0.22	3
Deception	0.63	0.86	0.17	2
Instruction integrity	0.74	0.92	0.39	3
Action safety	0.69	0.90	0.14	4

An auxiliary summary finds an earlier negative projection in 82.6% of trajectories that eventually cross a safety threshold, with median lead 3 rounds versus 1 for a single direction. This denominator is the subset of eventually failing trajectories, not all updates or all trajectories, so the result is a conditional retrospective summary rather than evidence of deployed online-warning reliability.

Ordinary held-out calibration or split-conformal intervals alone do not establish coverage after adaptive candidate search. The subsequent method instead specifies fixed features, regularized linear prediction, parameter confidence sets, and a local residual envelope; post-selection reliability is evaluated in Section efsec:sentinel-reliability.

C.4 Capability–safety geometry and a local existence result

Capability directions use independent successful/unsuccessful benign-task pairs matched on goal, input, tools, token budget, and initial state. They do not use the safety labels being tested. A capability discriminant $d_{C,\ell}$ is constructed analogously to Equation 12. Angles are compared in a common covariance metric,

$$\cos_{\Sigma_\ell}(d_{C,\ell}, d_{k,\ell}) = \frac{d_{C,\ell}^{\mathsf{T}} \Sigma_\ell d_{k,\ell}}{\sqrt{d_{C,\ell}^{\mathsf{T}} \Sigma_\ell d_{C,\ell}} \sqrt{d_{k,\ell}^{\mathsf{T}} \Sigma_\ell d_{k,\ell}}}. \tag{16}$$

The unit of analysis is a model/task-family/layer/domain combination, with correlated combinations grouped by model and task family for uncertainty assessment. Independently estimated directions require a stated common covariance or whitening convention.

The reported capability-probe AUROCs are 0.86 on matched tasks and 0.79 across tasks. Safety/capability cosine has median 0.03 and interquartile range $[-0.18, 0.22]$. The study partitions

cases into negative 30.6%, equivalent-to-zero 32.4%, positive 27.6%, and undecided 9.4%, using a prespecified equivalence band $[-0.10, 0.10]$.

After matching actual capability gain, update size, base model, and optimizer, the study reports update-level regression of 70.9% when the capability projection is positive and some safety projection is negative, versus 20.7% when all safety projections are nonnegative. These are behavioral-validation evaluation values under the declared update population and projection calibration. Representation-space angles alone do not determine the sign of a parameter update's true safety change. Equation 14 and held-out perturbation/replay tests provide the intended empirical connection. Model-side approaches such as Safe LoRA and SaLoRA offer relevant subspace-based background (Hsu et al., 2024; Li et al., 2025), not a proof of complete-agent feasibility.

Proposition (local capability gain with safety decline). Let C and a safety score s be differentiable at a continuous parameter state, with gradients $g_C \neq 0$ and g_s. Suppose $g_s \notin \{\lambda g_C : \lambda \geq 0\}$, and the support of executable continuous candidates contains a neighborhood of zero relative to $\mathrm{span}\{g_C, g_s\}$. Then arbitrarily small supported directions u exist with $g_C^\mathsf{T} u > 0$ and $g_s^\mathsf{T} u < 0$. Along a sufficiently small fixed such direction, the actual differentiable changes have these signs.

Proof. Decompose $g_s = a g_C + v$, with $v \perp g_C$. If $v \neq 0$, choose

$$\lambda > \max\left\{0, \frac{a\|g_C\|_2^2}{\|v\|_2^2}\right\}, \qquad w = g_C - \lambda v. \tag{17}$$

Then $g_C^\mathsf{T} w = \|g_C\|_2^2 > 0$ and $g_s^\mathsf{T} w = a\|g_C\|_2^2 - \lambda\|v\|_2^2 < 0$. If $v = 0$, the hypothesis forces $a < 0$, and $w = g_C$ has the required signs. In either case, for sufficiently small $\eta > 0$, $u = \eta w$ belongs to the stated local support and can have arbitrarily small norm. Differentiability gives $C(z + \eta w) - C(z) = \eta g_C^\mathsf{T} w + o(\eta)$ and the analogous expansion for s. The strictly signed leading terms dominate as $\eta \downarrow 0$. This proves the result. □

The proposition uses true gradients and a continuous-support condition, not fitted representation vectors. It excludes a zero safety gradient and nonnegative collinearity by hypothesis. Discrete edit sets need not satisfy its support assumption. Existence neither assigns probability to the direction nor forces an optimizer to select it, and an arbitrarily small decline need not exceed δ_k or cross τ_k. The empirical geometry motivates inspecting feasible directions; it does not yield a universal regression rate or a global safety guarantee.

D Contribution Localization, Interactions, and Stability

D.1 Block granularity and contribution measures

The signed contribution of block j in domain k is $r_{k,j} = \widehat{G}_{k,j}\phi_j$, with negative attributed mass $L_{k,j} = [-r_{k,j}]_+$. A vector block contributes through its full inner product, rather than through a count of individually negative coordinates. Model partitions may distinguish layers, attention/MLP modules, or adapter parameter blocks; harness partitions distinguish memory entries, skills, tool permissions, and workflow nodes. The partition and edit dictionary are fixed before comparing concentration.

Let $\mathcal{R}_k(p)$ contain the blocks with the largest $L_{k,j}$ at a fraction p of the block-count budget. Their concentration is

$$Q_k(p) = \frac{\sum_{j \in \mathcal{R}_k(p)} L_{k,j}}{\sum_j L_{k,j}}. \tag{18}$$

The ratio is undefined if the total negative mass is zero. Capability contributions are estimated analogously in capability-score units. The overlap between the risk-associated set and the highest positive-capability-contribution set is their Jaccard index, $|\mathcal{R}_k(p) \cap \mathcal{C}(p)|/|\mathcal{R}_k(p) \cup \mathcal{C}(p)|$. These attribution quantities describe a fitted decomposition; they do not measure realized damage or causal recovery in a named system.

Table 10: Top-10% concentration and overlap. Negative-safety and positive-capability shares have separate attribution denominators. All values are reported from the completed evaluation.

Update	Negative share (%)	Capability share (%)	Gini	Jaccard
Model-only	72.4	41.8	0.68	0.24
Harness-only	76.1	38.6	0.71	0.21
Joint	79.3	44.9	0.73	0.27

Here capability share is the fraction of positive capability attribution lying in the selected risk-associated blocks; it is not the Jaccard index or the intervention's realized capability loss. The retained capability shares are substantial despite limited top-set overlap. Neither those shares nor the joint negative-attribution share 79.3% is required to equal the intervention recovery 83.2%: they concern different quantities and, for nonlinear updates, different decompositions.

D.2 Interaction allocation and diffuse risk

The full matrix keeps interactions explicit. For locating physical edits, the study uses a Shapley-style allocation; a symmetric two-edit term can allocate $\gamma_{k,ij}/2$ to each participating edit. A consistent report must use either raw interaction columns or allocated physical-block effects at a given accounting stage, not count both. Signed totals must be conserved, while taking negative parts before versus after allocation can change the concentration denominator.

Interaction attribution accounts for 18.6% of negative effects in the reported summary. Separately, 12.0% of updates are described as having diffuse or higher-order risk that is poorly captured by sparse localization. The former is a share of attributed mass; the latter is a proportion of updates. Their definitions, populations, and aggregation are not interchangeable. An interaction can make an individually benign-looking pair of edits unsafe in combination, while diffuse effects can make small masks inadequate. These possibilities motivate constraints on the complete update, not a requirement that every component be nonnegative.

D.3 Ranking stability and control analyses

The rank correlations are Spearman 0.78 across calibration batches and 0.71 across attribution estimators. The comparison covers behavior-calibrated pullbacks, direct gradients, leave-one-block-out replay, and Shapley-style estimates. The top-10% summary is the reported concentration point; localization nominates edits for intervention but does not by itself establish their causal contribution.

E Ablation, Restoration, and Equal-budget Alternatives

E.1 Executable mask-and-replay comparisons

Every intervention starts from the same pre-update state z. First apply the complete candidate u to obtain z_u; then replay a candidate with the selected modifications removed to obtain z_m. Model masks remove identified parameter-update blocks, not arbitrary test-time output activations. Harness masks remove actual edits while preserving legal dependencies. Removing an edit also recomputes all affected interaction features. An interaction column is not independently removable code.

The reported intervention uses a 10% block-count mask. Table 11 compares random and norm-matched masks, a single model direction, model-only attribution, joint effects, and the highest-capability-contribution control under the same recovery/retention definitions.

E.2 Recovery denominators and the restoration audit

For the selected safety score s, define original damage $D = s(z) - s(z_u) > 0$ and original capability gain $A = C(z_u) - C(z) > 0$. Let z_r be the state obtained after restoring the deleted modifications.

The three ratios are

$$R_S = \frac{s(z_m) - s(z_u)}{D}, \qquad R_C = \frac{C(z_m) - C(z)}{A}, \qquad R_{\text{restore}} = \frac{s(z_m) - s(z_r)}{D}. \tag{19}$$

Safety recovery and damage reintroduced both use original safety damage as denominator; capability retention uses original capability gain. They are not success rates with a common denominator. Cases with $D \leq 0$ or $A \leq 0$ require a separately reported population and cannot enter these ratios without a defined rule. Ratios may fall below zero or exceed one; silently clipping them changes the estimand.

Table 11: Mask-and-replay summaries at 10% deletion. All values are reported percentages from the completed evaluation. Every recovery/restoration pair is subject to the exact-restoration audit.

Mask criterion	Capability retained	Safety recovered	Damage reintroduced
Random	94.6	19.7	19.7
Norm-matched	92.8	27.9	27.9
Single model direction	89.7	46.8	46.8
Model-only attribution	92.1	61.5	61.5
Joint effects	91.4	83.2	83.2
Highest capability contribution	42.6	58.1	58.1

If restoration exactly reconstructs the same complete update, $z_r = z_u$, then Equation 19 gives $R_{\text{restore}} = R_S$. The experimental table therefore reports identical recovery and damage-reintroduced percentages for each exact restoration pair; paired evaluation uses the same outcome definition and denominator. Partial restoration, retraining, and alternative restoration semantics are separate protocols rather than hidden adjustments to this table.

E.3 Equal-budget search and independent testing

For an originally risky update, the study reruns candidate generation outside the restricted risk-associated locations, retaining the same capability objective and a matched search budget. A successful alternative must have non-declining safety in every evaluated domain and retain at least 80% of the original capability gain. The reported success proportion is 73.5%; the remaining 26.5% yield no qualifying candidate within budget. Budget-limited failure does not show that the mathematical feasible set is empty, and a candidate with reduced capability below the threshold is not counted as success.

Independent evaluation separates candidate search from the final test of the selected alternative, using held-out tasks/trajectories and frozen rubrics. The completed evaluation reports the selected alternatives with this independent-test protocol.

Search accounting includes candidate attempts, model/tool compute, replay/evaluator queries, rejected candidates, and stopping rules rather than matching only the number of accepted alternatives. Single-step alternatives support testing reusable feasible space over rounds; they do not establish its persistence or eliminate the need for abstention.

F Multi-round Near-oracle Feasibility and Extensions

F.1 Protocol, complete baselines, and units

The near-oracle comparison uses extra counterfactual-evaluation budget to assess complete candidate updates, followed by evaluation on an independent final test set. It is a high-budget feasibility reference, not a mathematical optimum or the measured performance of learned SENTINEL, and is therefore not treated as a compute-matched deployable comparison.

The fixed external evaluator prevents the main comparison from improving scores merely by changing the measurement standard.

Table 12: Complete multi-round summaries. Gain is in capability points; regression is a percentage of submitted updates; acceptance is over scheduled selection opportunities in the evaluation panel; margin is the terminal minimum domain safety margin. All values are reported from the completed evaluation.

Selection rule	ΔC	Regression (%)	Acceptance (%)	Margin
Frozen	0	N/A	N/A	$+0.081$
Unconstrained	14.3	46.7	100	-0.137
Aggregate safety constraint	13.1	27.9	84.6	-0.061
Model-direction constraint	12.4	24.8	79.2	-0.044
Joint safety constraints	10.6	4.8	63.4	$+0.026$

Frozen makes no submitted transitions, so its submitted-update regression denominator is zero. The supplied plotting data also use N/A for acceptance because the proposal process is not specified. Its cross in Figure 2c is a no-update reference, not a sample with measured zero regression. The terminal margin can still be evaluated. For other methods, marker area denotes the reported acceptance percentage, not sample size or uncertainty.

The intended contrast is between enforcing every domain on a complete update, allowing compensation in an aggregate score, and constraining only model-direction effects. Aggregate constraints can mask domain-specific declines, while model-only constraints omit harness and interaction effects. The retained joint summary combines capability gain 10.6 with submitted-update regression 4.8% and terminal margin $+0.026$. Positive terminal margin does not imply that every earlier state satisfied its threshold or that every transition was non-declining. The nonzero submitted-update regression rate is a different statistic from threshold violation.

The terminal margin is $m_T = \min_k[S_k(z_T) - \tau_k]$. Figure 2d contains only terminal points.

F.2 Feasibility principle and uncertainty

Equation 2 optimizes estimated capability gain over executable candidates inside a local trust region, subject to a separate lower confidence bound for every domain's complete-update effect. The bound needs allowance for both estimation uncertainty and local model error. A fitted mean effect, a favorable R^2, or an ordinary pointwise interval does not establish a valid bound after adaptive search. The current method addresses this through a fixed-feature linear predictor, parameter confidence sets, and a local residual envelope; its assumptions and finite-budget validation belong to the method chapter and its evaluation.

The main setting uses $b_k = 0$. In a version allowing consumption of safety margin, a nonnegative decline budget would need to respect an independently justified current margin above τ_k; noisy current-state estimates cannot be treated as certain slack. That extension is not used to reinterpret the reported zero-budget setting. The trust-region norm includes scaled model, discrete-edit, and interaction features, but feasibility also requires actual executability. Projecting a feature vector onto linear inequalities alone may fail to produce valid code, memory, or a legal workflow.

Per-domain non-decline has the first-order form $G_{\mathbf{S}}\phi(u) \succeq 0$. Strict first-order invariance $G_{\mathbf{S}}\phi(u) = 0$ is a stronger nullspace special case; neither requires every component contribution to be nonnegative. Single-vector orthogonal projection therefore does not represent the general constraint. When the search pool has no feasible candidate, abstention keeps the current state. The no-op transition has zero true state change; it need not be interpreted as a statistically estimated nonzero update.

F.3 Across-carrier repetition and the oracle–learned gap

The expanded design retains capability-retention summaries under constraints of 84%, 79%, and 74% for model-only, harness-only, and joint RSI, respectively. A smaller retained fraction under joint updates is compatible with a more restricted feasible pool, but does not isolate interactions as its causal explanation.

The oracle–learned comparison distinguishes candidate-pool quality from selection error. Figure 5(b) reports the learned-versus-pool-oracle comparison on fixed pools, while Table 12 keeps the higher-budget approximate-oracle feasibility result separate from learned SENTINEL.

G Implementation and Statistical Specification

This appendix specifies the estimator, candidate construction, and acceptance interfaces of Section 4. Numerical hyperparameters are chosen on development data and frozen in a run manifest before final evaluation. The accompanying protocols record the completed SENTINEL evaluation and its measured outcomes.

G.1 Agent state, evaluation, and the RSI interface

The state $z_t = (\theta_t, M_t, K_t, T_t, W_t)$ is the currently adopted, persistent version. K_t denotes skills, whereas the unindexed K is the number of safety domains; T_t denotes tools, whereas T is an evaluation horizon. Memory includes retrievable persistent records; skills include reusable instructions or code; tools include their interfaces and permissions; workflow includes orchestration, routing, and persistent self-modification configuration. A snapshot also records optimizer state, retrieval indices, versioned dependencies, and other persistent execution state needed to reproduce these five components. Any change to such state that changes the adopted agent is part of u.

The parent RSI procedure supplies an experience collector and a capability-driven proposal operator. Experience $\mathcal{D}_t$ is produced by executing z_t and may include trajectories, rewards, critiques, and preferences. Collection occurs in a sandbox: proposed memory writes are staged, not automatically added to the adopted memory. The external controller can retain experiment records; persisting those records inside the acting agent requires an explicit memory edit. Parameter updates can reuse iterative self-training, DPO, or RL. Code or workflow proposals can reuse an improver or meta-agent. Self-Rewarding Language Models exemplifies iterative preference-data generation and training (Yuan et al., 2024); STOP exemplifies an improver modifying its own code without modifying the underlying language-model weights (Zelikman et al., 2023). We use RSI operationally for persistent version changes that feed back into subsequent experience or self-modification; the strength of this feedback is a separate experimental factor.

For domain k, let $\omega \sim P_k$ include the task, environment initialization, agent sampling randomness, and evaluator randomness, and let $\ell_k(z;\omega) \in [0,1]$ be its safety score. Define

$$S_k(z) = \mathbb{E}_{\omega \sim P_k}[\ell_k(z;\omega)]. \tag{20}$$

The score conversion, task distribution, attack distribution where applicable, and external evaluator are fixed. A runtime judge that routes agent actions is part of W_t; the external evaluator and SENTINEL acceptance implementation are outside the writable state. Evaluator integrity and sandbox enforcement are prerequisites of the experiment. The definition guarantees statements about these scores, not about unmeasured notions of safety.

We distinguish the true nondecrease target $\Delta S_{k,t} \geq 0$, a material regression event $\Delta S_{k,t} < -\delta_k$ used in evaluation, and an absolute threshold event $S_k(z_t) < \tau_k$. The reporting tolerance δ_k and threshold τ_k are fixed independently of the algorithmic budget $b_{k,t}$. Neither a positive final margin nor failure to reject a statistical null establishes nondecrease at every intermediate round. Details of the source observation metrics belong to Appendix A.

Archive interfaces. An archive can store candidate versions, their parents, and development feedback, as in agent-design and open-ended self-modification search (Hu et al., 2024; Zhang et al., 2025). An archived parent is not necessarily the adopted state. If a proposed version z' descends from an older parent, form the complete switch $u : z_t \mapsto z'$ and evaluate $S_k(z') - S_k(z_t)$. Recompute its coordinates, interactions, support, and trust radius relative to z_t. A certificate relative to the reported parent is insufficient for this switch. Incompatible coordinate systems or unsupported large switches are rejected or require a new calibration epoch. Exploration occurs only in the sandbox; the preservation proposition applies to adopted states. This complete-switch treatment is used whenever an archive proposal descends from a parent other than the currently adopted state.

G.2 Coordinates and model-side pullbacks

Let $\eta_j \in \mathbb{R}^{p_j}$ parameterize trainable block j and choose a fixed positive diagonal scale D_j within a calibration epoch. Define

$$a_j = D_j^{1/2}(\eta'_j - \eta_{j,t}), \qquad \nabla_{a_j} q = D_j^{-1/2} \nabla_{\eta_j} q, \qquad p = \sum_j p_j. \tag{21}$$

A scale may be predetermined or estimated from development gradient statistics; it is not retuned on final safety labels. The local map from a to executable weights is recorded. Low-rank adaptation exposes trainable factors (Hu et al., 2021); the pullback is with respect to those factors, while execution checks also inspect the resulting dense weight difference. Factor changes are not interchangeable with activation changes or with arbitrary perturbations of the dense weight matrix.

For domain k and activation layer $\ell \in \mathcal{L}_k$, the direction $d^{\text{int}}_{k,\ell}$, activation center $\bar{v}_{k,\ell}$, and pre-action window are fitted on a separate direction-development split and frozen. The window ends before the first externally consequential action. Matched contexts, future-action prediction, transfer, and bidirectional intervention provide an operational definition of a pre-action safety direction. Linear decodability alone does not establish behavioral relevance; refusal and dangerous-topic directions are candidate controls rather than interchangeable safety directions (Zou et al., 2023; Arditi et al., 2024). The original observation construction and its matched-context, transfer, and bidirectional controls are reported in Appendix C.

For each current state, draw reference prefixes $\mathcal{R}_t$ independently of the fitting-label randomness. In equation 5, differentiate at $a = 0$, holding the prefixes, center, direction, and other state components fixed. A parameter block can affect several measured activation layers, so its pullback sums over all relevant ℓ. Vector–Jacobian products compute the gradient of this scalar without storing a full activation Jacobian; Jacobian–vector products can score fixed candidate directions (Baydin et al., 2018). Reference prefixes, masks, precision, and activation hooks are versioned with the estimator.

What the derivative measures. The fixed-prefix derivative is a proxy sensitivity. Under differentiability and interchange-of-expectation conditions, a trajectory-dependent quantity obeys

$$\nabla_\eta \mathbb{E}_{\tau \sim p_\eta}[q(\eta, \tau)] = \mathbb{E}[\nabla_\eta q(\eta, \tau)] + \mathbb{E}[q(\eta, \tau) \nabla_\eta \log p_\eta(\tau)]. \tag{22}$$

The fixed-prefix computation does not estimate the second term. Behavioral calibration learns the relation between proxy changes and score changes from executed perturbations. Remaining proxy mismatch, reference-sample error, trajectory-distribution change, and curvature contribute to the local residual. This is why small activation changes alone do not certify safety.

For a black-box model endpoint, the parameter block is absent ($p = 0$). The same harness-effect estimator and acceptance rule apply. Gradients through a surrogate model are not pullbacks of the target endpoint and are outside the core parameter-update setting. Model revisions must be pinned or externally tracked; uncontrolled endpoint changes violate the fixed-state transition assumption.

G.3 Harness edits, interactions, and a fixed joint readout

An atomic edit specifies its component type, target identifier, old-version hash, new content or structured diff, scope, order, dependencies, and permissions. Examples are a memory-record replacement, skill-function change, tool-permission change, or workflow-node change. The indicator x_i refers to this particular intervention. A text embedding supplies features to the estimator; it does not make raw text a continuous executable update. Conflicting edits, invalid compositions, and inconsistent dependency orders have no admissible coordinates.

Paired effects. A single-edit fitting record uses

$$y_{k,i} = \frac{1}{m} \sum_{r=1}^{m} \big[\ell_k(z \oplus e_i; \omega_r) - \ell_k(z; \omega_r)\big], \qquad \mathbb{E}[y_{k,i}] = \alpha_{k,i}(z). \tag{23}$$

Pairs share initial tasks and environment conditions, with common random numbers where semantically meaningful. When tool order diverges, randomness is keyed to matched environment events rather than reused by raw execution index. A common seed alone does not ensure a valid pairing.

All variants of a task remain in the same data split. Paired experiments and their costs are logged as experimental blocks, including unsuccessful executions under a predetermined scoring rule.

For an ordered, jointly executable pair with all four arms valid, define

$$\gamma_{k,ij} = S_k(z \oplus \{e_i, e_j\}) - S_k(z \oplus e_i) - S_k(z \oplus e_j) + S_k(z). \tag{24}$$

Its estimator uses a paired four-arm contrast. The interaction policy selects pairs from shared call paths, dependencies, affected objects, permission changes, or prior evidence of nonadditivity. It can select a pair even when both main effects appear safe. This policy and its feature functions are frozen before fitting; only applicable columns are instantiated at a given state. Interactions outside this dictionary contribute to the residual or invalidate support.

A model–harness interaction uses the difference between model sensitivities in the unedited and edited contexts,

$$\widehat{\gamma}_{k,ji} = \widehat{g}_{k,j}(z \oplus e_i) - \widehat{g}_{k,j}(z), \qquad \psi_{ji}(u) = a_j x_i. \tag{25}$$

Both terms use compatible parameter coordinates, shared validated directions, and the same fitted readout. Reference-prefix compatibility must be checked; an edited context that cannot support the prescribed computation falls outside this local representation.

An explicit fixed-feature construction. Let $\chi(z)$ be a frozen state feature vector, $h_{k,\ell j}(z) = \nabla_{a_j} q_{k,\ell}$, and $\beta_{k,\ell}(z) = w_{k,\ell}^\top \chi(z)$. State features may contain versioned harness structure, validated activation summaries, and development-only calibration summaries. Edit encoders and all feature-selection rules are fixed within an epoch. A model feature block is

$$\begin{aligned} F^M_{k,\ell}(z,u) = \chi(z) \sum_j h_{k,\ell j}(z)^\top a_j \\ + \sum_{(j,i)\in\mathcal{I}^{MH}} \left[\chi(z \oplus e_i) h_{k,\ell j}(z \oplus e_i)^\top - \chi(z) h_{k,\ell j}(z)^\top\right] a_j x_i. \end{aligned} \tag{26}$$

The interaction terms share $w_{k,\ell}$ with the model main effect. Concatenate these blocks with typed edit-feature sums $\sum_{i:\mathrm{type}(i)=r} h_r(z, e_i) x_i$ and selected pair-feature sums $\sum_{(i,j)\in\mathcal{I}^{HH}} h_{r,s}(z, e_i, e_j) x_i x_j$. This yields a finite $F_k(z,u) \in \mathbb{R}^{d_k}$ and one coefficient vector w_k for the whole domain. There is no separately learned product of coefficients in this construction. Expanding $\widehat{w}_k^\top F_k$ gives the columns in equation 4; for fixed E it is affine in a.

Fitting records can be complete updates or the corresponding linear contrasts. If a contrast has weights c_h, its feature is $f_{kn} = \sum_h c_h F_k(z, u_h)$ and its label is the same contrast of observed scores or score changes. Use the four-arm contrast as one record, rather than counting its correlated arms as independent labels. With ridge strength $\lambda > 0$, fit

$$V_k = \lambda I_{d_k} + \sum_{n=1}^{N_k} f_{kn} f_{kn}^\top, \qquad \widehat{w}_k = V_k^{-1} \sum_{n=1}^{N_k} f_{kn} y_{kn}. \tag{27}$$

Use a Cholesky solve rather than an explicit numerical inverse. The same fit calibrates the behavioral coefficients $\widehat{\beta}$, edit effects, and pair effects. The full covariance captures their statistical dependence. Summing independently fitted per-column standard deviations is not the confidence construction used here. Adding a new learned encoder or feature structure requires a new fit and calibration epoch; a new edit instance generated by the fixed encoder still has to pass the support checks.

G.4 Data separation, local residuals, and support

Direction development and feature construction, regression fitting, calibration, proposal/capability development, and final testing have distinct roles. Group splits by task template, repository, and version lineage as appropriate, and keep all paired interventions and near-duplicate variants together. The final test is inaccessible to the collector, improver, estimator, support selection, and acceptance rule. Discovery of a new feature or threshold is followed by fresh confirmation data. If fitting records are collected adaptively, their features must be chosen before fresh label randomness is observed, as formalized in Appendix H.1.

The local approximation condition has the form

$$\Delta S_k(z,u) = w_k^{*\top} F_k(z,u) + r_k(z,u), \qquad |r_k(z,u)| \le \varepsilon_k(z,u). \tag{28}$$

One supported envelope is

$$\varepsilon_k(z,u) = A_k\|a\|_2 + \tfrac{1}{2}L_k\|a\|_2^2 + \varepsilon_{k,H}(z,E) + \varepsilon_{k,I}(z,a,E). \tag{29}$$

Here A_k bounds first-order proxy mismatch, L_k bounds remaining local curvature, $\varepsilon_{k,H}$ bounds unmodeled discrete effects, and $\varepsilon_{k,I}$ covers omitted interactions and coupling. All terms are nonnegative and zero at the no-op where appropriate, with $F_k(z,0) = 0$, $\varepsilon_k(z,0) = 0$, and $\Delta S_k(z,0) = 0$ exactly. The constants and functions are conditions on a declared region, not quantities that a finite calibration sample automatically certifies. Calibration estimates error profiles versus step size, edit scope, and interaction density, and selects a candidate region for testing. A theorem invoked for that region additionally requires a valid uniform envelope, either assumed or established by a separate argument with its own failure budget.

The support $\mathcal{V}_t$ records the model/state range, reference-prefix compatibility, allowed parameter blocks, edit types and scopes, call structures, and interaction configurations covered by the estimator. An out-of-support candidate is not assigned zero error: it is rejected, reduced to a supported candidate, or deferred to a new calibration epoch. Define the scaled update size

$$\|\phi(u)\|_{\mathcal{X}}^2 = \sum_j \|a_j\|_2^2 + \sum_i c_i^2 x_i^2 + \sum_{p\in\mathcal{I}_t} c_p^2 \|\psi_p(u)\|_2^2, \tag{30}$$

where positive weights c_i, c_p are fixed on development data. This is a norm on the ambient augmented coordinates, evaluated only on executable updates. Choose ρ_t within the declared calibration regime and separately test boundary candidates. A small norm does not by itself establish support: a small permission change can have a large behavioral effect. The radius controls the local approximation, while support controls applicability of the feature and error model.

What must be established before claiming a certificate. The probabilistic guarantee requires a coefficient-norm bound, conditional noise control, fitting-bias bounds, and a support-wide candidate residual envelope. The evaluation supplies measured values and the declared uniform bounds for these quantities. An implementation using empirically chosen envelopes is evaluated for coverage, but its empirical calibration is not a proof of the uniform assumption. This distinction also applies after online refitting.

G.5 Executable constrained proposals and local repair

For each proposed edit set E, collect its affine model coefficients and harness offset as $A_t(E) \in \mathbb{R}^{K\times p}$ and $c_t(E) \in \mathbb{R}^K$. Per-block contributions $\widehat{G}_{S,k,b}\phi_b(u)$ identify which proposed changes contribute to a domain's predicted regression. Interaction contributions remain separate, or are allocated by a fixed sum-preserving rule for ranking. Ranking only guides search; acceptance uses the unallocated complete effect. Positive and negative block contributions can compensate within the same row.

Choose a local convex cell $\mathcal{A}_t(E)$ within the allowed continuous coordinates and solve equation 6. Protected or unsupported coordinates are fixed at zero; other limits are convex coordinate bounds and local step-size constraints. For fixed E, the selected $a_j x_i$ terms are linear in a, so the trust constraint is convex. A nonconvex support need not be replaced by a globally convex one: the subproblem uses a specified convex cell within it. The construction margin $\kappa_t(E)$ is held fixed during a solve. It may be zero on the first proposal or the previous proposal's estimated uncertainty plus residual during repair. Such a reference-point margin is a search heuristic; the final bound is recomputed at the actual output.

For the reduced problem with only $Aa \succeq d$, the dual is

$$\min_{\nu\succeq 0} \tfrac{1}{2}\nu^\top AA^\top\nu + \nu^\top(Av_C - d), \qquad a^* = v_C + A^\top\nu^*. \tag{31}$$

This follows by minimizing $\frac{1}{2}\|a - v_C\|^2 + \nu^\top(d - Aa)$ over a and negating the resulting dual objective. Use an appropriate convex solver when additional bounds are active. Record primal feasibility and solver tolerance; a failed or inaccurate solve does not bypass the final safety checks.

In particular, zero may be infeasible when E contributes a negative offset or the construction margin is positive.

Candidate identity is the final diff relative to z_t. Multi-step parameter training must constrain and reevaluate the cumulative difference, not just each increment independently; $c_t(E)$ is counted once. Masking a gradient is insufficient to freeze a block if momentum, weight decay, optimizer state, or factorization changes still modify its executable weights. Diff checks cover the actual parameters and all harness components after execution of the updater. Numerical comparison tolerances are fixed in the manifest and reported.

Finite discrete search. The improver receives allowed component identifiers, edit schemas, dependencies, and domain-level effect feedback. A deterministic executor checks that the emitted diff modifies only allowed targets, matches source hashes, respects dependency order, and leaves evaluators and acceptance code unchanged. The beam contains only executable edit sets. Each expansion may add, remove, replace, or rewrite an edit; each edit-set branch can invoke the continuous subproblem. Ranking uses predicted constraint violation followed by estimated capability, with fixed tie-breaking. Beam width, depth, proposal count, token budget, gradient steps, solver calls, safety-label queries, and repair retries are explicit caps. Budget exhaustion returns the candidates already constructed, possibly none.

A repair can delete a harmful edit, substitute its content, narrow a permission, change an affected workflow path, or alter the parameter step. Every such action creates a new complete candidate, with its own diff, support check, prediction, and capability evaluation. Duplicate final diffs are identified by content hash. Candidates that remain infeasible after the repair budget are discarded. Finite beam or repair failure means that the allotted search did not find a feasible update; it does not imply that the true feasible set is empty.

G.6 Acceptance, logging, and optional extensions

Evaluate each complete candidate relative to the adopted z_t, then apply equation 8. Pool members need not be statistically independent. If the feasible pool is empty, do not call an argmax. Otherwise select the largest $\widehat{\Delta C}_t$, breaking ties by smaller update size and then a fixed candidate identifier. Commit only if it exceeds the preregistered $\delta_C \geq 0$. The no-op is an exact fallback, not evidence of positive capability improvement. New proposals can be attempted in a subsequent round without relaxing the safety constraints.

Log the adopted-state hash, archive parent if any, complete diff, data lineage, feature/readout/calibration versions, mean prediction, statistical radius, residual allowance, support and trust checks, capability estimate, decision, and resource use. Commit all state components atomically. A partial or failed commit is not the evaluated candidate and must not be silently adopted.

Selective verification. The core algorithm uses the learned bound. An optional verifier evaluates a *fixed complete candidate* on fresh paired trials $D_r \in [-1, 1]$. With a sample size m fixed before these trials, the bounded-score one-sided Hoeffding bound is

$$L_k^{\mathrm{ver}} = \frac{1}{m}\sum_{r=1}^{m} D_r - \sqrt{\frac{2\log(1/\alpha)}{m}}. \tag{32}$$

The candidate may depend on past searches, but not on these fresh outcomes. A verifier can be an additional gate, or an explicitly declared alternative certificate when its assumptions and failure budget hold. Repaired candidates require new trials or a bound uniformly valid for all tested candidates. Adaptive sample stopping requires a valid time-uniform procedure; repeatedly applying equation 32 with a data-dependent m does not supply one. Allocate failure probabilities over domains and verification attempts.

Snapshot, rollback, and recalibration. A deployment extension stores a full-state snapshot before commit and monitors subsequent behavior using data disjoint from final testing. Restoring an entire known snapshot restores that version's score under the fixed evaluation setup; it cannot undo external actions already taken. A partial rollback is a new joint update requiring reevaluation, and rollback can itself decrease a score relative to an improved current state. Consequently, a rollback

bypassing the acceptance rule is outside Proposition 1's sequence of gated transitions. Evaluate recovery separately from core nondecrease.

State-conditioned columns are recomputed as z_t changes even when $\widehat{w}_k$ is frozen. Online refitting is a separate extension: it creates a versioned calibration epoch, uses fresh time-separated development data, and renews support and failure-budget accounting. Detection of drift triggers rejection or recalibration, not an assumption that an old matrix remains valid.

G.7 Computational cost and implementation scope

Let $L_{\mathrm{tot}} = \sum_k |\mathcal{L}_k|$, p be the trainable-coordinate dimension, and N the number of complete candidates. A direct implementation uses L_{tot} scalar pullbacks at the current reference prefixes and stores up to $O(L_{\mathrm{tot}}p)$ sensitivities. Once model columns are materialized, dense model-effect dot products cost $O(NKp)$; sparse coordinates can reduce this cost. Typed edit encodings can be cached by state/edit hash. Explicit interaction cost scales with the instantiated pairs; model–harness contrasts require additional context-conditioned pullbacks or equivalent products. It is not generally zero.

For domain k, storing a dense readout covariance costs $O(d_k^2)$; direct Cholesky factorization costs $O(d_k^3)$ per fit, and evaluating one full covariance norm costs $O(d_k^2)$. Low-dimensional fixed features make this practical when appropriate; a diagonal approximation would change the confidence construction and is not silently substituted. The unconstrained dual in equation 31 has K variables, but additional local constraints change solver cost. Beam and repair costs are bounded by the declared construction budget, not by a claim of polynomial-time global search.

The reported abstract ledger separates setup (directions, fitting, calibration), parent-RSI experience/training, and per-round estimator/search/repair from offline final audits and approximate-oracle evaluation. Figure 4 and Appendix J.2 use this ledger for the cost comparison. Model-side pullbacks are available only in the trainable open-weight setting; physical runtime and monetary cost are outside the reported efficiency claim.

H Conditional Guarantees and Proofs

The results below concern fixed external expected scores. They depend on well-specified support and error conditions, which must be distinguished from finite-sample diagnostic evidence. No assumption of a safe improving candidate in every round is made.

H.1 Simultaneous confidence with fitting bias

Fix a domain and a calibration epoch, condition on the data used to select its feature map, and suppress the domain index. Write $\|v\|_A = (v^\top A v)^{1/2}$. Let $\mathcal{H}_{n-1}$ contain all information before collecting fitting record n, including its chosen feature f_n.

Assumption 1 (Local readout and fitting records)**.** *There exists a fixed $w^* \in \mathbb{R}^d$ with $\|w^*\|_2 \le W$ such that equation 28 holds uniformly for every state/update pair in the epoch's declared support. Fitting labels satisfy*

$$y_n = w^{*\top} f_n + r_n + \xi_n, \qquad |r_n| \le e_n, \qquad \mathbb{E}[e^{s\xi_n} \mid \mathcal{H}_{n-1}] \le e^{s^2R^2/2} \quad \textit{for every } s \in \mathbb{R}. \tag{33}$$

The feature f_n is predictable, and $\lambda > 0$, W, R, and the bias bounds e_n are valid for this experiment. The fixed feature map is not fitted on the same fitting-label noise.

For a contrast with coefficients c_h, a sufficient bias bound is $e_n = \sum_h |c_h| \varepsilon(z, u_h)$. Independent fresh task blocks can make the noise condition applicable even when proposals depend on prior records. If a paired difference is averaged over m independent bounded-score trials, $R = 1/\sqrt{m}$ suffices. For a four-arm contrast in $[-2, 2]$, $R = 2/\sqrt{m}$ suffices. A common conservative R can be used across records. Correlated replays reused across records require an appropriate noise analysis; counting such replays as independent does not meet the assumption.

Lemma 2 (Bias-aware parameter ellipsoid)**.** *Under Assumption 1, let $V_N = \lambda I + \sum_{n \le N} f_n f_n^\top$ and let $\widehat{w}_N$ be the ridge estimate. For $\alpha \in (0,1)$, with probability at least $1-\alpha$,*

$$\|\widehat{w}_N - w^*\|_{V_N} \le B_N(\alpha),$$
$$B_N(\alpha) = R\sqrt{2\log\left(\frac{\det(V_N)^{1/2}}{\det(\lambda I)^{1/2}\alpha}\right)} + \sqrt{\lambda}W + \sqrt{\sum_{n\le N} e_n^2}. \tag{34}$$

The same self-normalized event may be taken simultaneously over record counts within this fixed epoch.

Proof. Let X have rows $f_n^\top$, and let r and ξ collect the corresponding errors. The normal equations give

$$V_N(\widehat{w}_N - w^*) = X^\top \xi + X^\top r - \lambda w^*.$$

The self-normalized martingale inequality (Abbasi-Yadkori et al., 2011, Theorem 1) bounds $\|X^\top \xi\|_{V_N^{-1}}$ by the first term of equation 34, on an event of probability at least $1-\alpha$. Since $V_N \succeq \lambda I$, $\|\lambda w^*\|_{V_N^{-1}} \le \sqrt{\lambda}W$. For the fitting bias, singular-value decomposition of X gives $X(\lambda I + X^\top X)^{-1}X^\top \preceq I$. Therefore

$$\|X^\top r\|^2_{V_N^{-1}} = r^\top X(\lambda I + X^\top X)^{-1}X^\top r \le \|r\|_2^2 \le \sum_{n\le N} e_n^2.$$

The triangle inequality proves the claim. This bias term is necessary even when the candidate residual is separately subtracted at acceptance. □

Lemma 3 (Complete-candidate lower bound)**.** *On the ellipsoid event and the uniform residual event, every supported complete candidate satisfies $\Delta S_k(z,u) \ge L_k(z,u)$ simultaneously, including candidates chosen after observing the fitted estimator.*

Proof. For any supported (z,u), Cauchy–Schwarz in the V_k metric yields

$$|(\widehat{w}_k - w_k^*)^\top F_k(z,u)| \le \|\widehat{w}_k - w_k^*\|_{V_k}\|F_k(z,u)\|_{V_k^{-1}} \le B_k\|F_k(z,u)\|_{V_k^{-1}}.$$

Combining this inequality with $r_k(z,u) \ge -\varepsilon_k(z,u)$ gives equation 7. The same parameter event implies the inequality for all features in the supported class. Taking a data-dependent search, repair, or selection of those features does not require a new pointwise confidence event. The support-wide residual condition must still hold for those candidates. □

Failure accounting. For domain k and epoch q, allocate $\alpha_{k,q}$ to the parameter event. If residual envelopes or current-score bounds are themselves probabilistic, allocate separate failure probabilities to those events; optional verifier attempts also consume explicit budgets. Require the sum of all allocated probabilities over the declared horizon to be at most δ. A union bound then gives a simultaneous event of probability at least $1-\delta$; independence between these events is unnecessary. Epoch validity can be conditional on the past, provided each new epoch satisfies the stated assumptions and its conditional failure bound. Changes to features, scopes, or fit targets cannot silently reuse an earlier event. Ordinary pointwise holdout coverage does not supply this simultaneous guarantee.

H.2 Finite-horizon preservation

Proof of Proposition 1. Let $\mathcal{E}$ be the intersection of the confidence and residual events over all domains and epochs used through round $T-1$. The preceding accounting gives $\Pr(\mathcal{E}) \ge 1-\delta$. On $\mathcal{E}$, every committed candidate passes support and trust checks, and Lemma 3 gives $\Delta S_{k,t}(u_t) \ge L_{k,t}(u_t) \ge 0$ for every k. If the algorithm abstains, it commits no change and $\Delta S_{k,t}(0) = 0$ exactly. Thus $S_k(z_{t+1}) \ge S_k(z_t)$ for every round on the same event. No independence across rounds or existence of a feasible nonzero candidate is required. □

This is a conditional finite-horizon property of the adopted-state sequence. It does not assert that every rollout is safe, that unknown domains are protected, or that capability improves at every round.

If $S_k(z_0) \geq \tau_k$ for all k, nondecrease also preserves those initial thresholds. Without that initial condition, nondecrease alone does not imply threshold satisfaction. Ungated external state changes, endpoint drift, evaluator tampering, or rollback transitions outside the acceptance rule are outside the proposition.

H.3 Current-margin safety budgets

Suppose a simultaneous lower bound $\underline{S}_{k,t} \leq S_k(z_t)$ is available and $\underline{S}_{k,t} \geq \tau_k$. Set $0 \leq b_{k,t} \leq \underline{S}_{k,t} - \tau_k$. On the intersection of the current-score and complete-candidate bound events, an accepted candidate obeys

$$S_k(z_{t+1}) \geq S_k(z_t) - b_{k,t} \geq \underline{S}_{k,t} - b_{k,t} \geq \tau_k. \tag{35}$$

Applying this argument at each round preserves the threshold with the allocated overall probability. It permits within-domain decreases and therefore has a different conclusion from Proposition 1. The budget is recomputed from the conservative *current* margin; reusing an initial margin in every round is not justified. If the lower current bound is below the threshold, this construction supplies no nonnegative threshold budget. The strict $b = 0$ setting remains the primary method.

H.4 Minimum-change projection and local capability

Lemma 4 (Capability-aligned projection)**.** *Let $\mathcal{Q}$ be a nonempty closed convex set containing zero, let $v_C = \eta c$ with $\eta > 0$, and let $a^* = \mathrm{proj}_{\mathcal{Q}}(v_C)$. Then $c^\top a^* \geq \|a^*\|_2^2/\eta \geq 0$. If a capability function f has $\nabla f(0) = c$ and an L_C-Lipschitz gradient along the segment from 0 to a^*, then*

$$f(a^*) - f(0) \geq (1/\eta - L_C/2)\|a^*\|_2^2. \tag{36}$$

Proof. The projection variational inequality gives $(v_C - a^*)^\top (a - a^*) \leq 0$ for every $a \in \mathcal{Q}$. Taking $a = 0$ yields $v_C^\top a^* \geq \|a^*\|_2^2$; division by η gives the first statement. The smoothness lower bound gives $f(a^*) - f(0) \geq c^\top a^* - (L_C/2)\|a^*\|_2^2$, proving the second. □

For equation 6, $\mathcal{Q}$ must include both the affine safety constraints and the continuous local cell. Zero need not be in that set for a nonempty E or positive $\kappa_t(E)$. The lemma also requires the original step to be proportional to the actual local capability gradient; an arbitrary optimizer step or noisy gradient estimate does not automatically qualify. With fixed E, $f(a) = C(z_t \oplus (a, E))$, so the lemma compares with $f(0)$ at the *same harness*. A gain over that value alone does not establish a joint capability gain over the original z_t. This is why the algorithm evaluates complete-candidate capability separately.

H.5 Selection within a finite certified pool

Lemma 5 (Finite-pool capability selection)**.** *Suppose $\widehat{\mathcal{F}}_t$ is nonempty and, simultaneously for its members, $|\widehat{\Delta C}_t(u) - \Delta C_t(u)| \leq \epsilon_C$. Its empirical maximizer $\hat{u}$ satisfies*

$$\Delta C_t(\hat{u}) \geq \max_{u \in \widehat{\mathcal{F}}_t} \Delta C_t(u) - 2\epsilon_C. \tag{37}$$

If $\widehat{\Delta C}_t(\hat{u}) > \delta_C$ and $\delta_C \geq \epsilon_C$, its true capability gain is positive.

Proof. Let $u^\dagger$ maximize true gain in the finite pool. Then $\Delta C_t(\hat{u}) \geq \widehat{\Delta C}_t(\hat{u}) - \epsilon_C \geq \widehat{\Delta C}_t(u^\dagger) - \epsilon_C \geq \Delta C_t(u^\dagger) - 2\epsilon_C$. The acceptance threshold gives $\Delta C_t(\hat{u}) > \delta_C - \epsilon_C \geq 0$. □

The uniform error condition must cover the selected pool; a fresh evaluation of a fixed pool with simultaneous bounds is one possible implementation. Adaptive reuse of capability-validation data is not itself such a bound. The result concerns the certified finite pool, not the complete true feasible set and not the global update space. The default empirical capability threshold does not by itself prove positive true gain. For an empty pool, the algorithm abstains and neither maximizer in the lemma is invoked.

H.6 Orthogonal preservation as a special case

For a model-only, single-domain, exact local linear prediction with vector g, zero uncertainty, and no other active constraints, the nondecrease set is the halfspace $g^\top a \geq 0$. For $g \neq 0$ its projection is

$$a^* = \begin{cases} v_C, & g^\top v_C \geq 0, \\ v_C - \dfrac{g^\top v_C}{\|g\|_2^2} g, & g^\top v_C < 0. \end{cases} \tag{38}$$

If exact first-order preservation $g^\top a = 0$ is imposed instead, the second expression applies for every v_C: it is the orthogonal projection onto the nullspace of $g^\top$. For $g = 0$, both constraints impose no restriction. These formulas follow by a one-multiplier quadratic minimization. Nondecrease keeps safety-improving directions, whereas equality removes their component too.

Safe LoRA uses alignment-derived weight-space transformations with layer selection (Hsu et al., 2024). That is useful projection-related background, but its rule is not identified with this gradient halfspace or with the entire mixed continuous–discrete algorithm. The local special case does not establish an exact inclusion relation between the full methods. An arbitrary Euclidean projection of an edit indicator or text representation also need not be executable; discrete candidates are constructed by the schema-constrained search instead.

I Experimental protocol and evidence ledger

This appendix documents the completed evaluation, its analysis units, implementation checks, and evidence ledger. "Literature-reported" denotes a checked original result under its own protocol; those values are kept separate from the target evaluation. All target-panel values are reported for the registered model–environment configurations and use the units and denominators stated in the accompanying tables.

Main-study scope. All numerical panels use the fixed 4-seed scheme under master seed 20260925 and the frozen feature specification. The model and benchmark identifiers correspond to the configurations evaluated in the target panel. Public scores are confined to the literature table in Appendix J.3; reported ranges describe variation across the completed evaluation units.

The target panel separates six open-weight joint-update versions from four API harness-only backends, crosses software engineering, terminal tasks, multi-turn tools, and web workflows, and retains five native safety audits. The protected state includes parameters, persistent memory, skills, tool configuration, and workflow; an audit that replaces the evolved harness is a different protocol. The core comparison includes frozen agents, capability-only RSI, direct behavioral safety filtering, learned-filter-only with the same estimator, and SENTINEL. SHE is an additional harness-only comparator whose original and budget-matched adaptations remain distinct (Qu et al., 2026). The main decline budget is $b = 0$; selective validation, rollback, and refitting are excluded from core results. Complete lineages are the end-to-end units, while same-state interventions use parent states rather than candidate branches.

I.1 Coverage, versions and compatibility

The open-weight panel is Qwen3.5-9B and Qwen3.8-27B (Qwen Team, 2026a;b), Gemma 4 12B IT and 31B IT (Google, 2026b;c), gpt-oss-20b (OpenAI, 2026a), and GLM-4.7-Flash (Z.ai, 2026). The API panel is GPT-6 Sol, Claude Opus 5.5, Gemini 3.8 Flash and DeepSeek-V4.1-Flash (OpenAI, 2026b; Anthropic, 2026; Google, 2026a; DeepSeek, 2026). Table 13 records the target identifiers and update-carrier boundaries used in the completed evaluation.

For an open-weight version to enter the joint panel, the implementation must pass forward/backward, precision, tool-call parsing, adapter serialization and merge tests. A finite-difference check must agree with the declared fixed-prefix VJP in the actual optimization coordinates; merged weight deltas must agree with saved adapter deltas. The served checkpoint, tokenizer, chat template, context budget, inference engine, quantization, adapter rank and target modules, optimizer settings and training data hashes must be recorded. Mixture-of-experts, recurrent or hybrid implementations cannot inherit a different architecture's compatibility result. An API backend supplies no target-model

Table 13: Target identifiers and access boundaries used in the completed evaluation. Shorthand IDs index the fixed configurations.

ID	Documented target identifier	Intended update carrier
q35	`Qwen/Qwen3.5-9B`	Joint
q38	`Qwen/Qwen3.8-27B`	Joint
g12	`google/gemma-4-12B-it`	Joint
g31	`google/gemma-4-31B-it`	Joint
oss	`gpt-oss-20b` (release identifier)	Joint
glm	`zai-org/GLM-4.7-Flash`	Joint
sol	`gpt-6-sol`	Harness only
opus	`claude-opus-5-5`	Harness only
gemini	`gemini-3.8-flash`	Harness only
deep	`deepseek-flash` (V4.1 release route)	Harness only

parameter gradients in this protocol: any external improver gradients or open-weight surrogate do not count as joint target-model evidence. Tool-use and reasoning settings are fixed within a backend, not treated as comparable merely because different vendors use similar setting names. The DeepSeek release name and served API alias are logged separately because aliases can be rerouted (DeepSeek, 2026).

The four capability environments used in the completed evaluation are SWE-Bench Pro V2, Terminal-Bench 4.0, text-mode τ^2-Bench and WebArena-Verified (Scale AI, 2026; Terminal-Bench Team, 2026; Sierra Research, 2026; ServiceNow, 2026). The benchmark manifest must pin repository or registry revisions, task IDs, container/site snapshots, tool schema, grader and evaluator versions, user-evaluation pipeline backend, agent step limit, timeout and retry rule. SWE-Bench Pro V2 is not DGM's SWE-bench Verified subset; old scripts or default dataset aliases must not silently select another revision. The τ^2 business domains use the fixed text-mode protocol, without importing scores obtained using modified airline instructions or a different pass@k. Web resets restore the full external sandbox before every paired comparison. Official benchmark test tasks are not training or self-modification experience.

The safety panel comprises AgentDojo, AgentHarm, Agent-SafetyBench, SafeArena and Agent-Dyn (Debenedetti et al., 2024; Andriushchenko et al., 2024; Zhang et al., 2024; Tur et al., 2025; Li et al., 2026). These are *audit suites, not interchangeable safety domains*. The observation vocabulary distinguishes harmful execution, unauthorized tool use, privacy, audit integrity and observable deception, instruction integrity, and high-impact authorization. The task-to-domain map, domain distributions and grading rules are fixed in the evaluation manifest. In particular, observable audit manipulation is not inferred from unverifiable claims about an agent's internal intention. Unsupported domain–suite combinations are N/A, not negative examples.

A safety adapter must retain the adopted parameters and evolved memory, skills, tool policy and workflow. Only the task/environment interface may change, and that change must be logged. When a benchmark requires an incompatible tool set or replaces its entire agent driver, the resulting audit is explicitly a partial-state or endpoint-transfer evaluation; it cannot support a claim about the intact evolved joint agent. Benign utility, attack success, harmful rubric credit, refusal and over-refusal are kept separate. The ASB native Safety Score is not relabeled as the clean UBR or attack-ASR adaptation used by SHE, and AgentHarm Harm Score is not assumed to be a binary attack-success rate.

I.2 Data isolation and complete recursive runs

Each real lineage begins from a hashed snapshot $z_0 = (\theta_0, M_0, K_0, T_0, W_0)$, including optimizer and environment reset state. At each round, the current adopted agent collects new capability experience, proposes legal parameter and/or harness updates, evaluates candidates on the allowed development feedback, and adopts at most one complete update. The next round uses that adopted state and its resulting experience. Merely optimizing a harness once and evaluating its endpoint is not a multi-round RSI run. The external improver, when used, is held fixed unless its evolution is the factor under study; its calls, context and feedback access are recorded.

Data are separated by purpose: capability experience; capability selection/development feedback; effect-fitting contrasts; confidence/residual calibration or validation data; and locked capability/safety audits. Repository, task-template, attack-template, donor lineage and ancestor IDs are grouped to prevent leakage through near-duplicate branches. The same locked audit outcome cannot be added to training, memory, a skill, an improver prompt, candidate-selection feedback or recalibration. Reusing development feedback is an explicit optimization-pressure factor, not a reason to describe that development set as held out. Final audits occur after the candidate choice and never feed back. All candidate ancestry, rejected diffs, legality failures and stopping events remain in the manifest; failed runs are not silently replaced.

The core uses fixed feature extractors and fitted readout weights over the declared horizon. It recomputes current-state features and pullbacks after an accepted update; recomputing these quantities is not refitting the estimator. Refit epochs, additional online safety validation and rollback use fresh data and their own error/resource budgets. No-op acceptance is a separate event from accepting a nonzero update. When all legal candidates fail the safety or capability conditions, the complete current state is retained; a partial rollback of only one carrier does not implement that no-op.

I.3 Baselines, metrics and statistical analysis

The frozen comparator executes the same task workload but never changes the agent. Capability-oriented RSI uses the same allowed experiences and modification interfaces without a safety gate. The behavioral filter spends its online budget on fresh direct safety evaluations of candidates and their parent, rather than using the locked audit. Learned-filter-only uses the same structured estimator, confidence ellipsoid, residual envelope, support guard and domainwise selection as the core, but receives unconstrained proposals. Single-carrier protection and aggregate safety constraints are evaluated in the reported ablations.

The primary capability endpoint is the benchmark's native score at the predeclared terminal round. Progress throughout the run is also reported. Safety has several distinct endpoints: native audit outcomes; domainwise expected change relative to the current parent; end-of-lineage change relative to the initial state; any absolute-floor crossing over the whole trajectory; and the frequency of unsafe accepted updates. These denominators must not be exchanged. For lineage i, let A_{it} denote a nonzero accepted update and D_{it} a domainwise negative expected safety change. The pooled accepted-update frequency is

$$\frac{\sum_{i,t} \mathbf{1}\{A_{it} \wedge D_{it}\}}{\sum_{i,t} \mathbf{1}\{A_{it}\}}, \tag{39}$$

with N/A when the denominator is zero. Its uncertainty resamples whole lineages rather than treating those updates as independent samples. An equal-lineage average of within-lineage rates is a different estimand and is not substituted for the pooled accepted-update frequency. Round acceptance divides by scheduled selection opportunities; pool feasibility divides by candidates, then aggregates by parent. A positive terminal margin does not establish that the path never crossed the floor.

For end-to-end comparisons, independent complete lineages are the analysis units, with paired initializations and blocked task/environment seeds when feasible. For same-state causal contrasts, the unit is the parent-state block, with all arms and counterfactuals retained inside it; parent states sampled from a shared ancestor require an additional ancestor/lineage cluster. Repeated task attempts and adjacent rounds provide precision within a unit, not additional independent runs. Cluster bootstrap or randomization-based intervals preserve these blocks; every reported interval or test uses the corresponding declared unit.

The actual safety decision concerns expected domain scores, which are not observed without error. Paired behavioral evaluation gives an uncertainty interval for the true contrast, rather than permission to label a small noisy difference as a definite regression. Real coverage analyses propagate audit uncertainty, for example by treating a case as confirmed covered, confirmed uncovered or unresolved using separately budgeted simultaneous behavioral intervals. Sensitivity analyses assign unresolved cases both ways. Confirmatory familywise domain statements use a simultaneous procedure; Benjamini–Hochberg may be appropriate for a separately declared exploratory false-discovery analysis, but does not turn arbitrary individual intervals into simultaneous ones. No observed failures is not zero risk, and lack of a statistically detectable difference is not non-inferiority.

I.4 E01: INDEPENDENT REVALIDATION OF THE OBSERVATION CHAIN

Independent observation revalidation. E01 replays the three observation stages on new lineages and parent states, including low-risk updates, non-local interactions, failed localizations and updates with no alternative found at the allowed budget. Carrier comparisons use whole runs; prediction and intervention comparisons use parent-state blocks. The observation summaries are registered in Table 14 with their denominators. A finite unsuccessful alternative search says nothing about global emptiness of the safe-update set.

Table 14: **Observation summaries and denominators.** The trajectory and accepted-update denominators differ.

Quantity	Denominator / unit	Value
Joint end-of-trajectory regression	trajectories (%)	68.4
Raw RSI accepted-update regression	accepted updates (%)	46.7
Approximate-oracle capability gain	standardized points	10.6
Approximate-oracle accepted-update regression	accepted updates (%)	4.8
Approximate-oracle acceptance	proposals (%)	63.4
Approximate-oracle terminal margin	safety-score units	0.026
Frozen accepted-update regression	no accepted updates	N/A

J ADDITIONAL PERFORMANCE RESULTS AND RESOURCE ACCOUNTING

E02: complete multi-round outcomes. *Completed target-panel trajectories.* The target matrix contains 40 model–environment cells. Table 1 reports the native-audit target cells used in the main comparison, while Figures 7–11 show the 12-round capability trajectories for all 40 model–environment cells under Frozen, Raw RSI, Behavioral filter, Learned filter, and SENTINEL core. The configuration-level trajectories are kept separate rather than averaged across tasks, preserving both improving and frozen/high-conflict cells.

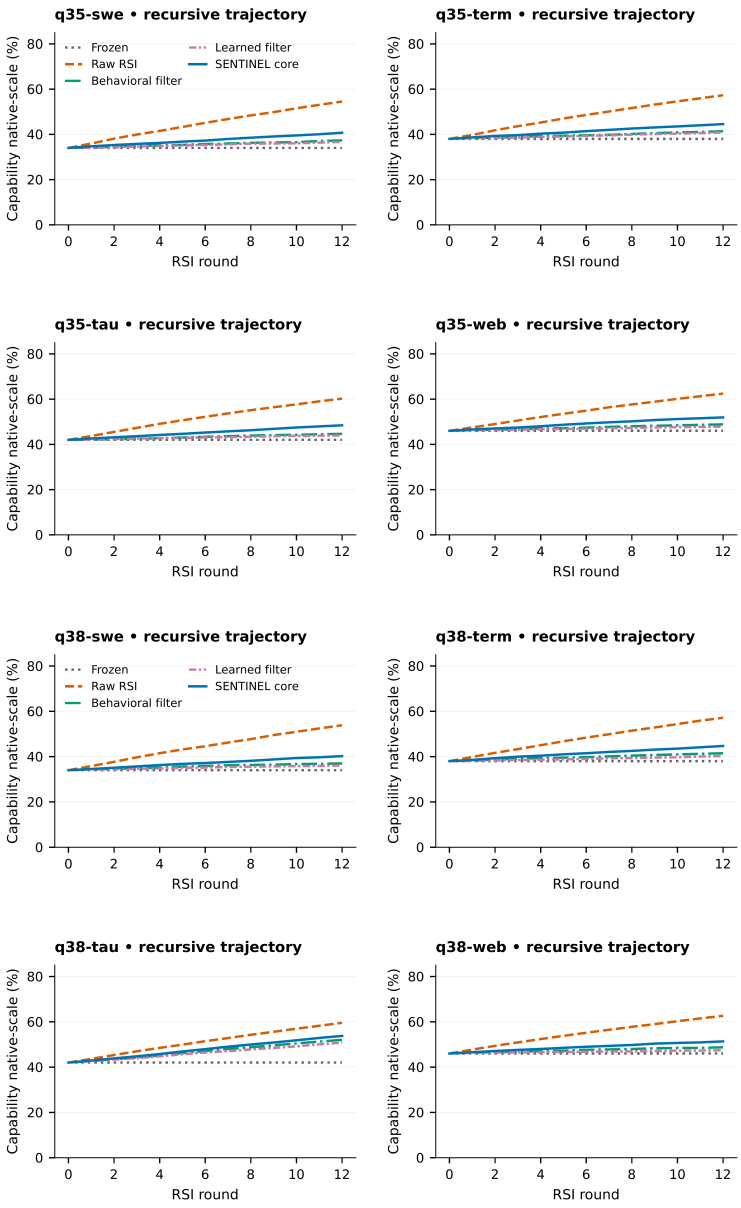

Figure 7: **Qwen configuration-level capability trajectories.** Twelve-round trajectories for Qwen3.5-9B and Qwen3.8-27B across SWE, terminal, τ^2, and web environments.

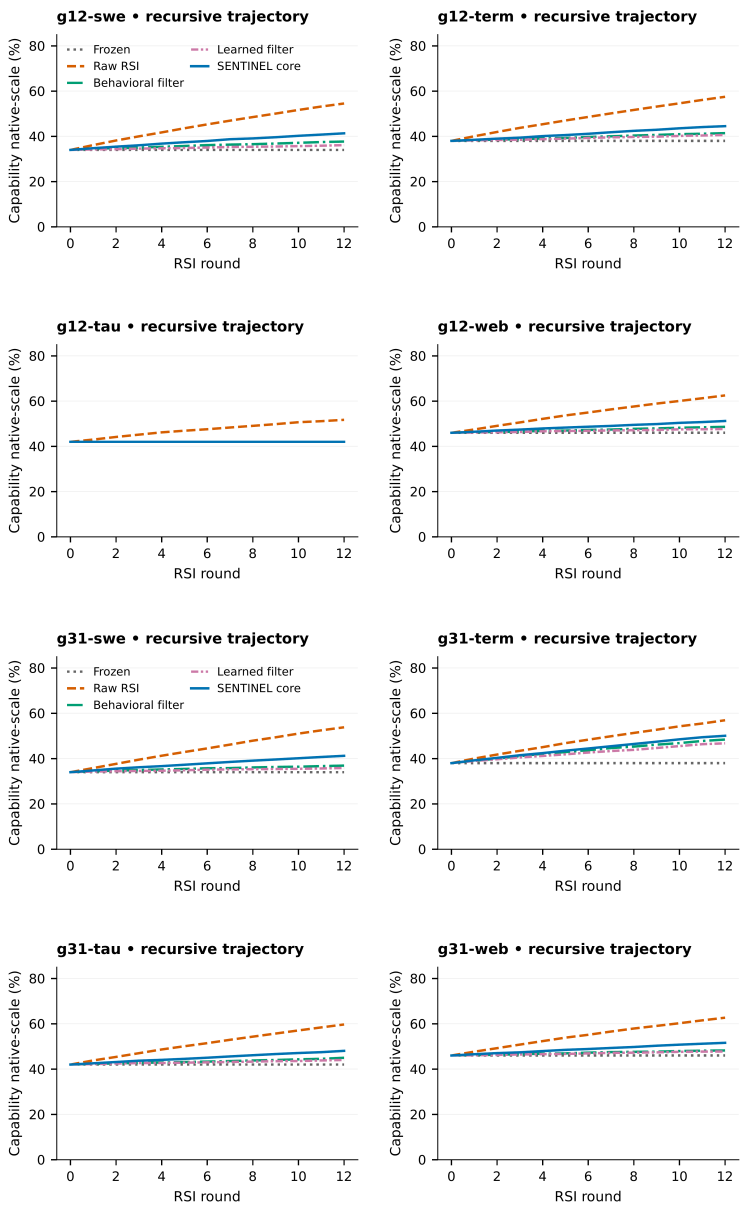

Figure 8: **Gemma configuration-level capability trajectories.** Twelve-round trajectories for Gemma 4 12B IT and Gemma 4 31B IT across the four capability environments.

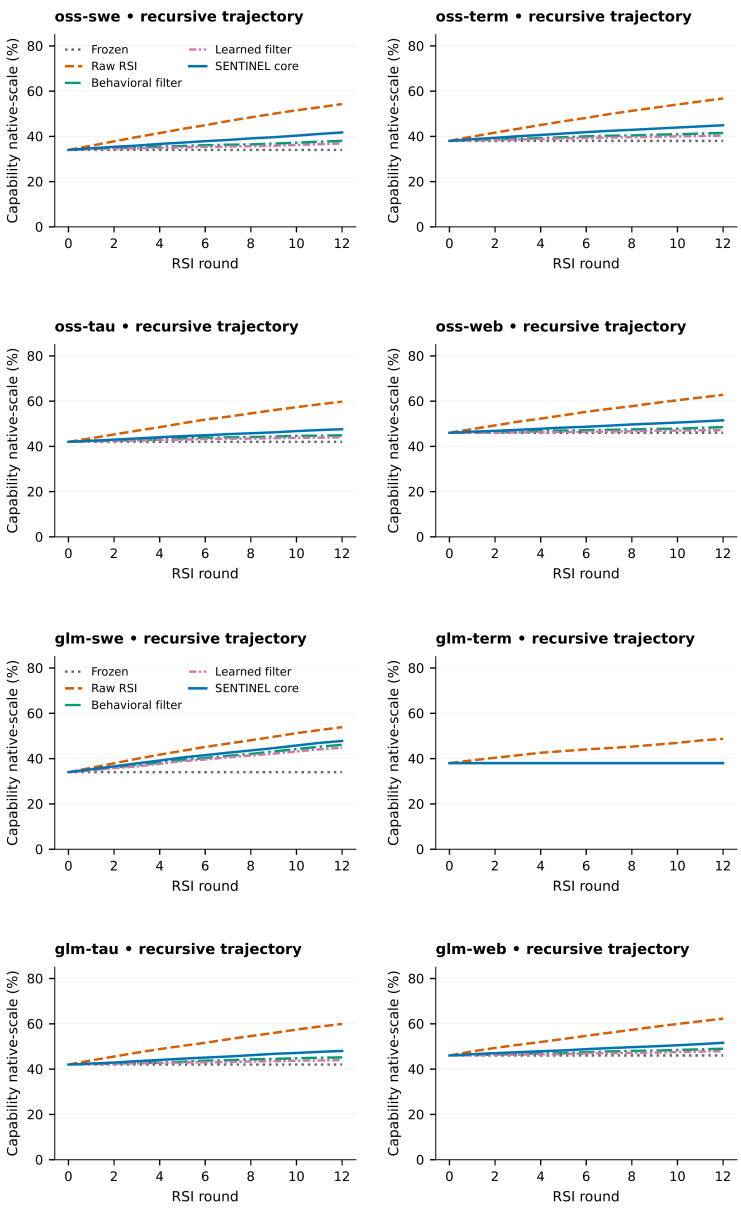

Figure 9: **Open-weight configuration-level capability trajectories.** Twelve-round trajectories for gpt-oss-20b and GLM-4.7-Flash across the four capability environments.

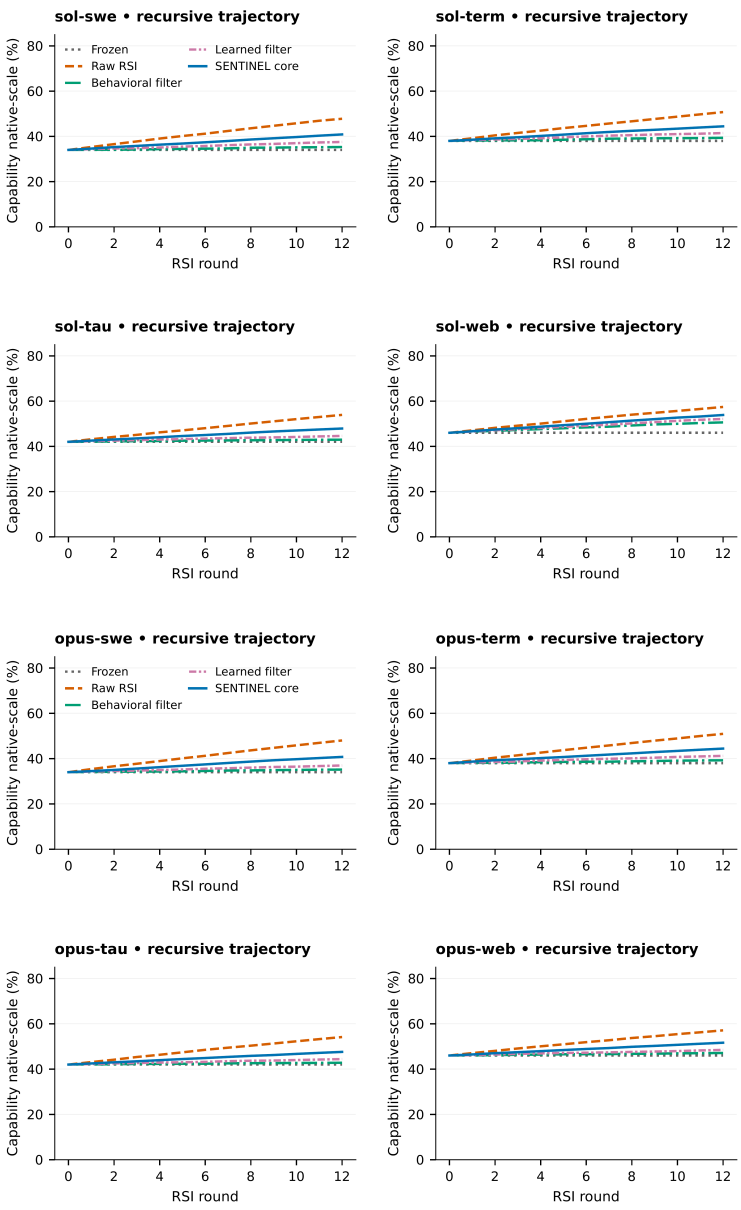

Figure 10: **API configuration-level capability trajectories I.** Twelve-round trajectories for GPT-6 Sol and Claude Opus 5.5 across the four capability environments.

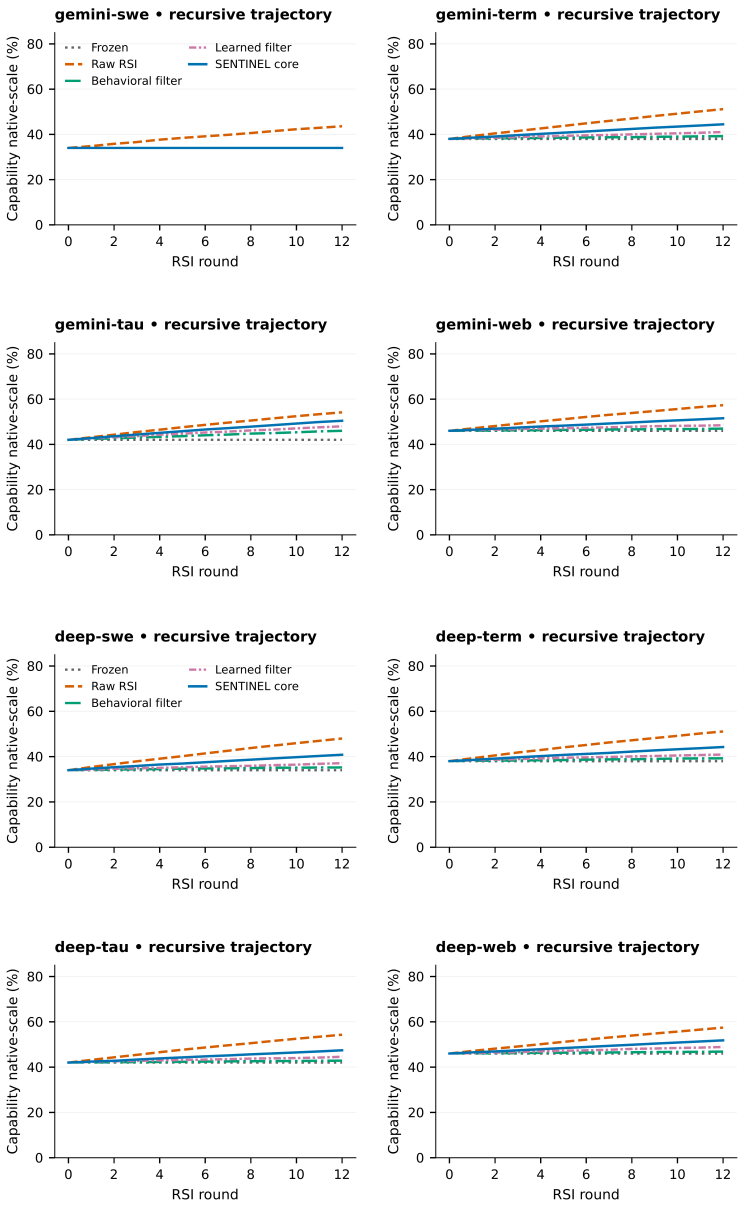

Figure 11: **API configuration-level capability trajectories II.** Twelve-round trajectories for Gemini 3.8 Flash and DeepSeek-V4.1-Flash across the four capability environments.

J.1 E03–E04: CARRIER COVERAGE AND CONFOUNDING CONTROLS

E03: carriers and update mechanisms. *Carrier comparison and utility protocol.* The causal carrier comparison runs model-only, harness-only and joint updates *within the same compatible open-weight model and task environment*. It separately crosses iterative supervised fine-tuning, preference-based updates when supported, and fixed external versus current-agent improvers, without changing the protected domains. Each method receives its own data-isolated calibration at initialization and the same declared opportunity budget. Independent lineages are the units. Comparing an API model to a different open-weight model does not isolate a carrier effect; the main target-panel partition therefore documents scope, not that causal contrast. A benefit confined to one improver or one training mechanism must be reported at that scope. Models failing gradient or tool-state checks remain explicit incompatible entries rather than silently disappearing.

E04: budgets, update frequency and normal utility. *Operation-budget and normal-utility protocol.* From the same initial lineages, compare equal candidate opportunities and equal total resource budgets. In the latter, a baseline may use savings for additional proposals, more direct safety evaluations or additional candidate capability evaluation; record its allocation before evaluating the locked test. Add uniformly reduced step sizes, randomly skipped updates and safety experience replay under the same resources. Randomized skip/shrink interventions identify their total effects; post hoc matching the core's accepted-update count, update norm or realized capability gain is only a secondary check and may induce selection bias. Benign task success, false refusal, tool completion and timeouts are audited separately with the evolved harness intact. A method that suppresses all tool use or stops updating cannot claim useful safety preservation merely from low harmful-task completion. Unit-level paired uncertainty and the initial frozen score are required for any non-inferiority claim.

J.2 E25: RESOURCE ACCOUNTING

The reported efficiency comparison uses a fixed abstract operation ledger. It separates cold-start initialization, experience generation, and candidate processing, and counts candidate/evaluator operations under the same ledger weights used for Figure 4. The primary quantity is total abstract cost over the run rather than cost per accepted update, which would be undefined for a run with no accepted updates.

Equal-candidate comparisons are supplemented by equal-total-ledger-budget comparisons so that a cheaper baseline can reinvest budget in additional proposals or direct evaluations. Accordingly, the paper's efficiency statements are restricted to the abstract ledger and are not converted into physical runtime, GPU-hours, token counts, latency, or monetary cost.

J.3 ORIGINAL LITERATURE PROTOCOLS, ISOLATED FROM THE TARGET COMPARISONS

Table 15: **Literature-reported results, not our reproductions.** Original DGM and SHE protocols differ from the target panel and from each other; values are not used to calculate SENTINEL gains, rankings or significance. The adjacent protocol ledger fixes the source version, evaluation conditions and metadata available for each source.

Original protocol	Agent	Resolved / pass@1	ASR / UA	Harm Score
SWE Verified subset	DGM initial	20.0	—	—
Polyglot full	DGM initial	14.2	—	—
SWE Verified subset	DGM evolved	50.0	—	—
Polyglot full	DGM evolved	30.7	—	—
SHE original → held-out	SafeHarness	—	17.1/31.6	18.0
SHE original → held-out	SHE seed	—	8.6/33.5	19.8
SHE original → held-out	SHE evolved	—	5.5/47.6	9.8

The DGM entries use the original arXiv version (Zhang et al., 2025), not an assumed reproduction on SWE-Bench Pro V2. They concern a frozen foundation model with an evolving coding agent, a capability-feedback archive, a SWE-bench Verified subset and a separate full-Polyglot endpoint. The reported subset progression is not a locked-test guarantee, and the full-Polyglot numbers must not be replaced with its development-subset values. Self-modification uses Claude 3.5 Sonnet (New),

while task evaluation uses that backend for SWE-bench and o3-mini for Polyglot. The source version, evaluation stages, model roles and search budget are recorded in the literature protocol ledger; immutable provider snapshots, total bills and endpoint uncertainty are recorded in the evaluation manifest.

The SHE entries come from original Tables 1 and 2 (Qu et al., 2026): a fixed DeepSeek-V3.2 agent, safety-harness evolution on an ASB subset, and held-out AgentHarm transfer, with the paper's reported improver/judge configuration. Table 15 keeps ASB attack ASR and utility under attack distinct from the AgentHarm partial-credit Harm Score. These conditions differ from the new backend panel and from capability-first RSI. A fair adaptation must preserve the allowed safety feedback, experience, query and compute budget, and must state which artifacts were translated into the current memory/skill/tool/workflow schema. Its code, compatibility and matched results are retained as a distinct adaptation; Original public scores are not used as the adaptation's entries.

K GENERATION, SELECTION AND CAUSAL INTERVENTIONS

K.1 E05 AND E07: CANDIDATE GENERATION VERSUS SELECTION

E05: learned–oracle differences within a fixed pool. *Same-pool selection evaluation.* Draw parent states independently of whether a safe candidate is later found. At each parent, generate raw and constrained pools under fixed total budgets, then replay *every* candidate with the complete resulting agent and paired evaluation randomness. Compare the learned gate with a matrix-oracle comparison and a high-budget behavioral approximate oracle on exactly that pool. Separate a pool with no audited feasible positive-gain candidate from a pool that contains one but is not exploited by the learned rule. Oracle capability gaps are computed within parent and pool, with unresolved behavioral labels analyzed both as feasible and infeasible; ties and the no-op are explicit. The reference responses are fixed for this comparison, while the approximate oracle retains finite-sample uncertainty. This comparison identifies pool-local estimation/selection loss, not the best possible update or the best recursive trajectory. Better performance from an entirely different oracle-generated lineage is not evidence for a small same-pool learning gap.

E07: randomized generation by gate intervention. *Randomized generation and selection evaluation.* Randomly assign proposal generation within parent-state blocks before outcomes are observed, then apply capability-only or safety-aware selection to each fixed pool. The pre-gate endpoint is the proportion of candidates with both audited non-decline in every domain and positive capability gain, summarized by parent. Post-gate endpoints are selected capability gain, safe positive-gain probability, acceptance and unsafe acceptance, with a no-op recorded as no accepted update. All branches share parent parameters, optimizer, harness, environment and allowed feedback; they do not supply independent lineage replicates. Equal-candidate and equal-total-cost versions are both required because preconstraint and repair consume resources. A short full-loop confirmation tests whether an immediate pool improvement survives recursive feedback. A null pre-gate effect would rule out attributing an observed endpoint benefit to easier feasible-candidate generation, even if the selector itself performs well.

K.2 E06: PREDICTION AND DATA EFFICIENCY

Completed predictor and learning-curve results. Table 8 compares update-norm and output-KL heuristics, generic representation drift, a single refusal direction, model pre-action directions, additive model-plus-harness effects, and the full interaction readout using held-out R^2, regression AUROC, and calibration error. Figure 12 reports held-out effect-estimation MAE as the number of calibration contrasts increases. The splits for representation fitting, effect fitting, calibration, and final evaluation remain isolated, and learning-curve counts refer to distinct parent-state contrasts.

K.3 E08–E10: LOCALIZED EDITS, INTERACTION AND FALSIFICATION CONTROLS

E08: deleting and restoring the indicated change. *Deletion and restoration intervention evaluation.* At the same parent, compare the original update, deletion of the identified risky part, restoration of precisely that part, the constrained update, and legal restoration of the original risky part into

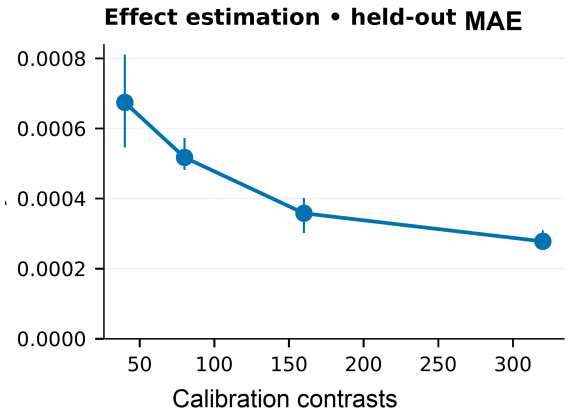

Figure 12: **Effect-learning evaluation.** Held-out effect-estimation MAE versus the number of calibration contrasts. Whiskers show variation across independently fitted evaluation replicates.

the constrained update. Random equal-count, norm-matched, edit-type-matched and sham controls share environmental seeds and dependency closure. The deletion panel uses 38 of 64 parent states with a nonzero permission edit and sufficient continuous norm for exact norm matching; eligibility is not based on the measured safety effect. Eligibility requires executable dependencies, compatible tool schemas and identical parent state. Audit actual dense and merged LoRA deltas, optimizer moments, weight decay and accumulated microstep updates: masking a proposed gradient alone does not establish that a block stayed frozen. An exact restoration must not secretly introduce another parameter update or reuse a mutated optimizer.

A real removal–restoration contrast supports localized causal influence only on that eligible subset. It does not imply that all negative attribution mass is removable or that restoration into a different constrained context must have the same effect. Absolute safety and capability changes are primary. If reporting recovery/retention ratios, keep the denominators distinct:

$$R_s = \frac{S(z+u_{\text{del}}) - S(z+u)}{S(z) - S(z+u)}, \qquad R_c = \frac{C(z+u_{\text{del}}) - C(z)}{C(z+u) - C(z)},$$
$$R_{\text{restore}} = \frac{S(z+u_{\text{del}}) - S(z+u_{\text{restore}})}{S(z) - S(z+u)}. \tag{40}$$

Compute these per declared domain and eligible parent; use absolute contrasts when a denominator is near zero, and do not clip ratios into a cosmetically convenient range. Attribution share, recovered safety and retained capability cannot be exchanged because their denominators differ.

E09: independent behavioral factorial effects. *Measured behavioral factorial.* Figure 5(d) evaluates no change, parameter change only, harness change only, and the joint change at the same parent using fresh full-agent execution. The behavioral interaction in domain k is

$$I_k = S_k(\theta + \Delta\theta, H + E) - S_k(\theta + \Delta\theta, H) - S_k(\theta, H + E) + S_k(\theta, H). \tag{41}$$

It is measured by fresh full-agent execution, not defined by the learned interaction coefficient. Representative parent sampling estimates prevalence; a separately labeled interaction-enriched stratum improves diagnostic precision but does not estimate prevalence. Compare the full and interaction-ablated readouts on identical pools, then inspect whether behaviorally non-additive candidates explain additional unsafe accepts. Harness–harness combinations receive analogous legal factorials. Sequentially applying two changes with intervening learning is a different path and is not substituted for the simultaneous contrast. Real weak or inconsistent interaction would narrow the need for joint modeling in that setting; the interaction pattern alone cannot decide that question.

E10: semantic mismatches and invariance. *Completed sham and renaming checks.* Permute a risky block to an incorrect legal location, use state features from a stale but compatible parent, randomize a parameter direction at matched norm, or shuffle effect labels within a clearly separated

training set. These changes break the intended state/update correspondence while keeping budgets and support rules fixed. Compare them within parent blocks with the correct estimator and independent behavior. In contrast, simultaneously permuting safety domains, their labels, all constraint rows and budgets is only a renaming: the feasible set and decisions must be invariant. It is a positive implementation check, not an effective negative control. The implementation tests this invariance and exact sham/restoration equality on the registered configurations. Failure of the invariance check is an implementation error, whereas a null mismatch effect limits claims that precise localization was necessary in that setting.

K.4 E17: FULL ABLATIONS AND RETUNING

Core and independently retuned ablations. Table 16 reports the full method, removal of the pre-constraint, model-only and harness-only protection, removal of interactions, aggregate rather than domainwise constraints, mean rather than lower-bound selection, and removal of the trust region in the same representative cell. These variants use the same initial lineage blocks and equal development-set tuning budgets.

The ablation interpretation is conditional. Single-carrier protection is most informative when the other carrier is edited; aggregate constraints matter when domain effects conflict; interactions matter when measured combinations are non-additive; and the trust region matters when approximation residuals increase with edit scope. A frozen ablation has N/A unsafe/accepted rate, not perfect safety of accepted updates. Capability improvements by an ablation can reflect a different risk tolerance, while identical safety can reflect a benign regime. The table includes unchanged trajectory-crossing outcomes and a frozen arm. Actual component claims require domainwise behavioral contrasts, complete-run costs and error bars at the correct independent unit, not merely a collection of lower proxy scores.

Table 16: **Extended representative-cell ablation set.** The harness-only protection counterpart is added to the main ablation rows; values use the same denominators as Table 2.

Variant	Capability	Crossing	Unsafe/accepted	Accept	Cost
SENTINEL core	40.7	0.0	0.0	94.4	305.6
No pre-constraint	36.7	0.0	0.0	51.4	262.4
Model protection only	34.0	0.0	N/A	0.0	305.6
Harness protection only	37.0	0.0	12.8	54.2	305.6
No interaction	40.8	0.0	3.5	97.9	305.6
Aggregate gate	45.4	0.0	9.0	100.0	305.6
Mean gate	47.1	0.0	11.1	100.0	305.6
No trust region	44.0	8.3	14.5	91.0	305.6

L ADAPTIVE RELIABILITY, RESIDUAL ERROR AND USABLE SAFETY BUDGETS

L.1 E19: COVERAGE OF THE SELECTED CANDIDATE

Coverage evaluation with reference responses. The primary object is the structured fixed-feature linear readout with a parameter ellipsoid, not a pointwise quantile gate. For a candidate feature $F_k(z,u)$, the lower bound has the method's form

$$L_k(z,u) = \widehat{w}_k^\top F_k(z,u) - B_k\sqrt{F_k(z,u)^\top V_k^{-1} F_k(z,u)} - \varepsilon_k(z,u). \tag{42}$$

The ellipsoid controls uncertainty about a shared parameter vector, so its event can cover adaptively selected feature vectors without a separate union bound over every candidate, under the declared fixed-feature/noise assumptions. This follows the self-normalized confidence-set approach (Abbasi-Yadkori et al., 2011), with the method's explicit approximation residual accounting. A bound on parameter-estimation error does not bound a missing behavioral interaction or arbitrary out-of-support residual.

The reliability evaluation varies candidate count at identical parent-state blocks and compares the supported core rule, a nominal pointwise rule, forced out-of-support execution, and the guarded

core rule under matched replicate counts. Coverage is reported for all parents and accepted updates separately, together with acceptance and unsafe-acceptance counts. The final selected no-op has exact $L_k = \Delta S_k = 0$, which is why acceptance is shown beside all-parent coverage.

The counts in Table 17 use independent refits as units and known expectations as truth. The pointwise rule happens to retain supported coverage in this evaluation, so no within-support ellipsoid advantage is claimed. Forced out-of-support execution removes the core support guard and exposes an omitted response term; the guarded rule instead rejects those unsupported candidates while still accepting supported ones. Empirical coverage is an evaluation of the implemented rule rather than a proof of the residual envelope over the entire support. Matching seeds pair calibration draws across rules and search sizes; independence is across experimental replicates within a condition, not across plotted conditions.

Table 17: **Reliability evaluation with explicit counts.** All-parent coverage includes no-ops; accepted-only coverage is shown in a separate column. Unsafe acceptance is reported with both all-parent and accepted-update denominators. None of these columns are empirical agent confidence intervals.

Rule	N	Covered/all	Accepted/all	Covered/accepted	Unsafe/all	Unsafe/accepted
Core: support	1	240/240	46/240	46/46	0/240	0/46
Pointwise	1	239/240	64/240	63/64	0/240	0/64
Forced OOS	1	220/240	39/240	19/39	20/240	20/39
Core: guarded	1	240/240	3/240	3/3	0/240	0/3
Core: support	4	240/240	117/240	117/117	0/240	0/117
Pointwise	4	238/240	147/240	145/147	0/240	0/147
Forced OOS	4	175/240	103/240	38/103	62/240	62/103
Core: guarded	4	240/240	12/240	12/12	0/240	0/12
Core: support	16	240/240	223/240	223/223	0/240	0/223
Pointwise	16	236/240	234/240	230/234	0/240	0/234
Forced OOS	16	106/240	205/240	71/205	120/240	120/205
Core: guarded	16	240/240	37/240	37/37	0/240	0/37
Core: support	64	240/240	240/240	240/240	0/240	0/240
Pointwise	64	240/240	240/240	240/240	0/240	0/240
Forced OOS	64	50/240	240/240	50/240	177/240	177/240
Core: guarded	64	240/240	113/240	113/113	0/240	0/113

L.2 E24: STRICT NON-DECLINE, TOLERANCES AND ABSOLUTE FLOORS

Budget sensitivity evaluation. Compare $b = 0$ against a predeclared non-inferiority tolerance and a current-margin budget, with the same domain distributions, confidence target and search resources. A tolerance changes the permitted expected decline; an absolute-floor rule may spend available margin and is not equivalent to per-round non-decline. Recompute remaining margin from the current adopted state, rather than reusing the initial margin every round. The full lineage is the independent unit. Record strict freezing, length of plateaus, capability, unsafe accepted updates, allowed-budget exceedances and any floor crossing as different endpoints. The high-conflict experimental slot remains frozen even when a finite relaxed budget is allowed (Table 18); the seed scheme is held fixed across the comparison. Complete freezing at $b = 0$ is treated as a substantive usability limitation. Finite-sample uncertainty about a tiny decline is not treated as evidence of exact non-decline.

Table 18: **Strict-budget sensitivity in the prespecified high-conflict slot.** A nonzero b changes the guarantee; it is not the default. Capability is native-scale reported percent, freezing is per experimental lineage, and unsafe/accepted refers to any actual negative measured change rather than exceeding b.

b	Capability	Acceptance (%)	Frozen lineages (%)	Unsafe/accepted (%)
0.000	42.0	0.0	100.0	N/A
0.001	42.0	0.0	100.0	N/A
0.003	42.0	0.0	100.0	N/A