WHAT PROCESS EVALUATION OF CODING AGENTS ACTUALLY MEASURES: ACTION, TASK, AND STEP ARE THREE DIFFERENT LEVELS

Anonymous authors

ABSTRACT

Coding agents are increasingly evaluated not only by whether they solve a task, but also by how they execute it. However, existing process-level evaluations often treat action prediction, task uncertainty, and step attribution as if they were the same problem, which makes it unclear what such evaluations actually measure. In this paper, we introduce a measurement framework for process evaluation in coding agents and instantiate step-level causal attribution with SCAE, a replay-based estimator derived from a structural causal model of agent execution. Our framework combines prefix-conditioned identification, replay/intervention-based estimation, and controlled judge-information manipulation to study process evaluation at the action, task, and step levels. Experiments on 500 file-localization episodes from 12 repositories show that next actions are driven primarily by execution provenance rather than code-graph transitions, and that execution uncertainty is structured at the task rather than step level. They also show that full-trace judges exhibit systematic collider bias, suggesting that current process evaluation often measures semantic relevance rather than certified causal contribution. Our code are available at `https://anonymous.4open.science/r/scae-repro-1B01`.

1 INTRODUCTION

Large language model (LLM) coding agents are increasingly evaluated not only by whether they solve a task, but by how they execute it (Si et al., 2026). Developers want more than a success label: they want to know which steps advanced the task, which caused failure, and whether process signals can be trusted for debugging, benchmarking, or training (Prasai et al., 2026; He et al., 2026).

However, process evaluation remains underspecified because it conflates three distinct levels of analysis: what the agent is likely to do next (*action*) (Wen et al., 2024), whether execution remains uncertain or recoverable (*task*) (Wang et al., 2025b), and whether an action causally changed the outcome (*step*) (Wang et al., 2025a). This conflation obscures what different process-evaluation methods are actually measuring. Methods that condition on different information and target different quantities are therefore often treated as answering the same question. Process reward models (Setlur et al., 2025), full-trace LLM judges (Chen et al., 2026), and restart-based ablations (Kontonis et al., 2026) are often discussed in exactly this way, although they evaluate different objects. Because these targets are not separated, a process score that appears informative may be difficult to interpret, while a score that appears uninformative may reflect a different evaluation target.

We treat action, task, and step as three distinct levels of process evaluation, but distinguishing them requires a reference quantity that is explicit and estimable. To address this, we introduce SCAE, a replay-based estimator derived from a structural causal model of agent execution. SCAE operationalizes step-level causal attribution and serves as a *measurement instrument*: its step-local causal effect provides a concrete reference for interpreting other process signals.

We study this framework in file localization, the simplest realistic setting with replayable environments, fully logged prefixes, and a checkable outcome. We reconstruct execution prefixes and replay continuations to estimate step-local effects. We do so observationally when the replayed action distribution retains overlap, and interventionally by substituting an alternative action and *actually executing* it when the policy is deterministic. The same fitted object also yields next-action distributions and replay variance, enabling action-, task-, and step-level analysis within a single design.

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.

Our contributions are as follows.

- We formulate process evaluation as a three-level measurement problem, distinguishing action prediction, task uncertainty, and step attribution instead of treating them as one notion of process quality, and identify the conditioning set each level requires.
- We introduce SCAE, with a prefix-conditioned identification result, an exactly additive step decomposition, and a hybrid estimator that selects its identification strategy from measured rather than assumed local positivity.
- We show that next actions are governed by execution provenance rather than code-graph transitions, so the dependency graph is a prior over *where* the agent works rather than a local transition model.
- We show that execution uncertainty resides mainly at the task level, and that this—not budget alone—is why step-level attribution remains underpowered at feasible replay cost.
- We isolate collider bias in full-trace LLM judging by manipulating the judge's information set, replicate it across 3 judge models, and show that process attribution is better read as downstream-informed semantic relevance than as certified causal contribution.

2 Related Work

Counterfactual reasoning in discrete sequential systems. Counterfactual reasoning over action sequences is commonly grounded in the Gumbel–max structural causal model (Ravfogel et al., 2025; Khanmohammadi et al., 2025), which has been used to explain sequences under known MDPs and to analyze how counterfactual influence decays as trajectories diverge (Chatzi et al., 2025; Kazemi et al., 2024). Those constructions assume a known transition mechanism and fixed action space; LLM agents over repositories provide neither, so SCAE estimates the mechanism through replay, measures positivity rather than assuming it, and treats compaction as an identification failure.

Step-level credit assignment and process supervision. Process reward models supervise intermediate steps and outperform outcome-only supervision (Zhang et al., 2026b; Setlur et al., 2025; Wang et al., 2024), but their labels are Monte Carlo estimates over completed rollouts, which reintroduces downstream information into the current step's score—as do hindsight and counterfactual credit assignment, deliberately, to reduce gradient variance (Arjona-Medina et al., 2019; Liang et al., 2026; Yang et al., 2026). Shapley-style attributions (Horovicz & Goldshmidt, 2024) give additivity analogous to our telescoping identity at the cost of exponentially many re-evaluations. We do not propose a stronger step scorer: we clarify what a step score must condition on to be causal, and keep additivity through one fitted object instead of 2^T re-executions.

Process-oriented evaluation of coding agents. Coding-agent evaluation remains centered on terminal success (Merrill et al., 2026; Xu et al., 2026a; Xia et al., 2024), while a complementary line calibrates defect detectors against human process annotations (He et al., 2026; Zhang et al., 2026a; Li & Cao, 2026). That estimand—the probability an annotator would mark a step defective—concerns annotation agreement rather than causation, so a step may be consistently judged defective while contributing no identifiable causal effect (Appendix D.13).

Uncertainty quantification and structural priors. Calibration for language models has largely addressed single-turn generation with one scalar per output, and aleatoric/epistemic decompositions and conformal prediction inherit that framing (Xia et al., 2025; Ulmer et al., 2024; Nakkiran et al., 2026; Melo et al., 2024; Ross et al., 2026; Vovk et al., 2005; Angelopoulos & Bates, 2023); in agents, uncertainty compounds across decisions and the agent's own actions determine what becomes observable, so a terminal score cannot say where reliability was lost. Separately, retrieval and localization modules rank files by similarity or dependency structure (Xu et al., 2026b; Edge et al., 2024), building the assumption that code structure guides the next move into current scaffolds; visited files do cluster in the graph, but clustering does not imply graph-governed transitions, which no prior work we know of tests against a region-preserving null—the gap §5.4 fills.

3 Preliminary: Three Levels of Process Evaluation

An episode consists of an issue q, a repository snapshot with graph G, a logged trajectory $\tau_{1:T} = (A_1, O_1, \ldots, A_T, O_T)$, and a verifiable outcome Y computed against the gold file set $F^\star$. Process

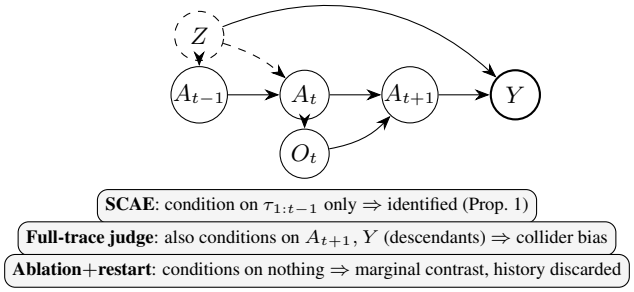

Figure 1: **Three conditioning sets and their causal status.** Different conditioning sets imply different attribution targets. Conditioning on the prefix blocks latent difficulty Z (dashed), because the policy observes nothing unlogged and its decoding noise is exogenous.

evaluation asks three questions of it: predict A_{t+1} from $\tau_{1:t}$ (*action*), quantify how uncertain the outcome remains (*task*), and decide whether A_t changed the outcome (*step*). Only the third is causal and the least precisely stated, so we fix it first: with $\tau_{1:t-1}$ the realized prefix before step t, the step-level effect of replacing a with a' is

$$\theta_t(a, a') \;=\; \mathbb{E}\big[Y_{A_t=a'} - Y_{A_t=a} \mid \tau_{1:t-1}\big], \tag{1}$$

i.e. whether changing a single action, holding the realized history fixed, would change the outcome.

The conditioning set, not the scorer, decides whether attribution is causal. equation 1 conditions on the realized prefix and nothing else, which is what separates it from the quantities common practice computes (Figure 1); process reward labels, by contrast, are back-filled from completed trajectories and therefore condition on descendants of A_t. What makes this framing useful is that it yields a prediction falsifiable without our own estimator being correct: conditioning on descendants of A_t induces collider bias that should displace blame toward *later* steps, which §5.7 tests. Appendix B.2 tabulates each practice and its predicted signature, and Appendix B.7 shows why data-flow reachability is not a proxy for causal irrelevance.

4 The Proposed SCAE

SCAE plays two roles: it provides the formal object that distinguishes step-local causal attribution from other process signals, and it is the instrument with which we test what process evaluation can identify in practice. Figure 2 summarizes the pipeline; Algorithm 1 in Appendix C.1 states the procedure and Appendix B.1 the notation.

4.1 An agent–environment structural causal model and identification

Let $\mathcal{M} = \langle U, V, F, P(U)\rangle$ be a structural causal model with exogenous variables $U = (Z, \{G_\pi^{(t)}\}, \{U_{\text{env}}^{(t)}\})$, where Z is task difficulty, $G_\pi^{(t)}$ decoding noise, and $U_{\text{env}}^{(t)}$ environment noise:

$$A_t := \arg\max_k \big(\log \pi_k(S_t) + G_{\pi,k}^{(t)}\big), \quad G_{\pi,k}^{(t)} \sim \text{Gumbel}(0,1), \tag{2}$$

$$O_t := \mathcal{E}\big(A_t, G;\, U_{\text{env}}^{(t)}\big), \quad S_{t+1} := \mathcal{T}(S_t, A_t, O_t), \quad S_1 := \sigma(q, \text{sys}), \tag{3}$$

$$\hat{F} := \phi(S_T), \qquad Y := \mathbb{1}\big[F^\star \subseteq \hat{F}\big]. \tag{4}$$

The Gumbel–max form makes categorical decisions perturb-and-argmax draws, giving a tractable account of local policy stochasticity, and the outcome is containment rather than exact set equality, so localization is not penalized for naming reasonable auxiliary files such as tests.

Assumption 1 (Context observability)**.** *S_t is a deterministic function of $(q, \tau_{1:t-1})$.*

Assumption 2 (Exogenous decoding noise)**.** *$G_\pi^{(t)} \perp (Z, U^{(<t)}, U_{\text{env}})$.*

Assumption 3 (Replayable environment)**.** *Given $(A_t, G, U_{\text{env}}^{(t)})$, the observation O_t is determined.*

Proposition 1 (Sequential randomization by construction)**.** *Under Assumptions 1–3, $A_t \perp (Z, U^{(\geq t)}) \mid \tau_{1:t-1}$, and consequently for any a with $\pi(a \mid \tau_{1:t-1}) > 0$,*

$$\mathbb{E}\big[Y \mid \tau_{1:t-1}, do(A_t{=}a)\big] = \mathbb{E}\big[Y \mid \tau_{1:t-1}, A_t{=}a\big]. \tag{5}$$

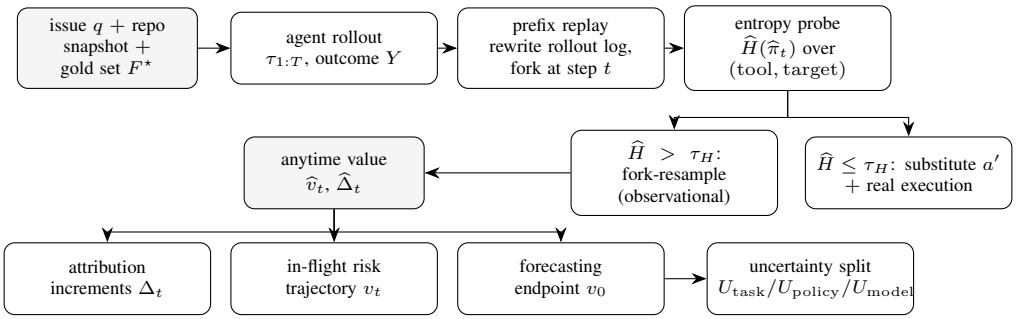

Figure 2: **Overview of SCAE.** An episode is rolled out once to obtain $(\tau_{1:T}, Y)$; prefix replay then reconstructs $\tau_{1:t}$ by rewriting the rollout log at item granularity, which requires no change to the agent. An entropy probe at each estimation point selects the identification strategy: observational fork-resampling where the policy retains randomness (Prop. 1), and substitution of an *actually executed* alternative action where it does not (Assumption 4). Shaded boxes are data and plain boxes pipeline stages; the two branches below the probe are the cases of equation 8.

Step-level effects are therefore identified nonparametrically from observational traces, without a no-unmeasured-confounding assumption: conditioning on the prefix fixes S_t, so the remaining variation in A_t is exogenous policy noise and step t behaves as a locally randomized treatment (Appendix B.6). The assumptions are empirical rather than formal: context observability fails under compaction, since dropped context becomes unobserved state, and replayability fails if the environment cannot be reproduced from the logged state and action—both verified in §5.2—while decoding exogeneity is a property of the serving policy, untestable from traces. Appendix B.5 formalizes the three failure modes, none of which is confounding.

4.2 An anytime value and exactly additive step contributions

Rather than estimating each step effect independently, we define an anytime value $v_t := \mathbb{E}[Y \mid \tau_{1:t}, q, G]$ with increment $\Delta_t := v_t - v_{t-1}$, so each step contributes by changing the expected outcome relative to the preceding prefix.

Theorem 1 (Exactly additive step contributions)**.** *Under Assumptions 1–3,*

$$\sum_{t=1}^{T} \Delta_t = v_T - v_0 = Y - \mathbb{E}[Y \mid q, G], \tag{6}$$

and each increment decomposes into an action effect relative to the policy's own baseline and an observation effect induced by the returned tool output.

The surprise of an episode relative to its pre-execution expectation is distributed over steps without residual, so the increments Δ_t are our attribution object. Appendix B.6 proves the theorem and states the split, whose halves imply different remedies: better planning versus better tools or indexing.

4.3 Hybrid estimation under measured local positivity

To estimate these quantities we reconstruct a prefix, resume from it m times, and set $\widehat{v}_t = m^{-1}\sum_i Y_t^{(i)}$ with $\widehat{\Delta}_t = \widehat{v}_t - \widehat{v}_{t-1}$. Because all arms share the prefix and pinned environment noise, this common-random-numbers design makes a small replay budget usable.

The estimator is valid only where the local action distribution retains overlap, and positivity is not globally available: some prefixes generate diverse next actions under replay, others repeatedly produce the same one. We therefore measure it: with $\mathcal{C}$ denoting action buckets at the level of tool and target object, $\widehat{\pi}_t(c) = m^{-1}\sum_i \mathbb{1}[\text{bucket}(A_t^{(i)}) = c]$ has entropy

$$\widehat{H}(\widehat{\pi}_t) = -\sum_{c \in \mathcal{C}} \widehat{\pi}_t(c) \log \widehat{\pi}_t(c), \tag{7}$$

which measures whether the next action remains stochastic under replay from the same prefix. Appendix C.4 reports it at every probed point and shows that overlap varies *within* an episode, so

neither a globally observational nor a globally interventional estimator is correct. This forces a hybrid strategy,

$$\widehat{\Delta}_t = \begin{cases} \widehat{v}_t - \widehat{v}_{t-1}, & \text{if } \widehat{H}(\widehat{\pi}_t) > \tau_H \quad \text{(observational branch)}, \\ \widehat{\mathbb{E}}[Y \mid \tau_{1:t-1}, do(A_t = a')] - \widehat{\mathbb{E}}[Y \mid \tau_{1:t-1}, do(A_t = a_t)], & \text{if } \widehat{H}(\widehat{\pi}_t) \le \tau_H \quad \text{(intervention branch)}. \end{cases} \tag{8}$$

In the intervention branch we substitute a' into the prefix, execute it for real, splice its genuine output into the continuation, and replay the remainder; we never fabricate outputs, since intervening on an observation rather than an action targets a different quantity.

Assumption 4 (Perturbation transferability)**.** *The substituted action a' lies in the support of a suitably perturbed version of the deployed policy.*

This makes explicit that the low-entropy branch is not observational identification under the original policy: it estimates the effect of moving to a nearby action that remains plausible under a perturbed one. We label which branch produced every estimate, and Appendix B.4 places the resulting quantities on the causal ladder.

4.4 Outcome choice and uncertainty

Theorem 1 requires only bounded Y, and we exploit that freedom: with Y binary, single-target instances are almost always solved, so many substitutions leave the indicator unchanged, while exact set equality is more brittle. We therefore use recall over the gold file set, which preserves the most intervention-sensitive variation (Appendix C.6 screens all six candidates), and because values are estimated at sampled prefixes under a finite budget, contributions are segment-level approximations.

The same fitted object also supports uncertainty analysis. The predictive uncertainty of the left endpoint $v_0 = \mathbb{E}[Y \mid q, G]$ splits conceptually into task ambiguity, execution variability, and epistemic model uncertainty (equation 13, Appendix D.7); we do not claim all three are identified on real traces, and use the split only to estimate U_{policy} from repeated rollouts and to test whether outcome-relevant uncertainty is structured at the step or task level.

5 Experiments

5.1 Research questions

RQ1: Which structural signal best predicts a coding agent's next action—the code dependency graph, or the agent's own execution provenance?

RQ2: If the graph does not predict the next step, what role does it actually play?

RQ3: Do commonly reported reuse patterns reflect genuine temporal behavior, or permutation-invariant artifacts of the visited multiset?

RQ4: At which level does execution uncertainty reside, and is step-level attribution measurable at feasible replay cost?

RQ5: What do observational attribution methods, especially full-trace judges, measure when no certified step-level causal reference is available?

5.2 Experimental setup

Dataset and outcome. We study 500 episodes in the sense of §3, derived from SWE-bench-Verified over 12 repositories. The corpus is dominated by `django` at 46%, so leave-one-repository-out is the protocol wherever a predictor is fitted (Appendix D.1). The outcome is recall over $F^\star$, selected on intervention sensitivity: 98.6% of instances can move under intervention against only 73.2% under binary success.

Preconditions and protocol. Assumptions 1 and 3 are verified rather than asserted: replay is byte-identical at 1.00 over 100 actions and compaction occurs in 0 of rollouts (Appendix C.2). For attribution and uncertainty we replay 8 continuations from each of 190 prefixes across 38 instances—1901 continuations over 12.7 hours of real agent execution.

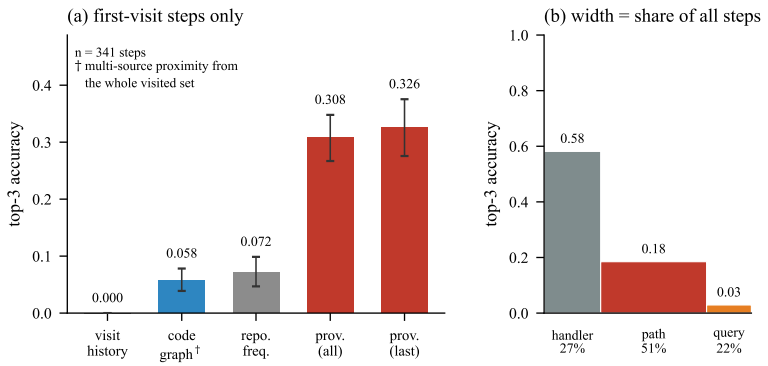

Figure 3: **Execution provenance versus code structure on first-visit steps.** (a) First-visit steps, where visit history is empty by construction. Bars are closed-form expectations over random tie-breaking; whiskers are episode-level bootstrap 95% CIs. (b) Predictability by object type, bar width being the share of all 7616 steps.

Next-object predictor	**cov.**	**top-3**	**95% CI**	**top-10**	$\exp(H)$
Repository structure					
Repository frequency (marginal)	0.349	0.072	[0.047, 0.099]	0.163	97.62
Code dependency graph, strengthened	0.677	0.058	[0.039, 0.078]	0.172	1719.13
Visit history					
Visit-history frequency	0.000	0.000	[0.000, 0.000]	0.000	4.84
Execution provenance					
Uniform over mentioned paths	**0.689**	0.136	[0.116, 0.155]	0.354	25.00
Frequency-weighted history	**0.689**	0.194	[0.162, 0.227]	0.431	22.17
Recency-weighted history	**0.689**	0.308	[0.267, 0.348]	**0.499**	18.75
Last output only	0.501	0.297	[0.256, 0.338]	0.431	4.00
Last output, position-ranked	0.501	**0.326**	[0.276, 0.375]	0.455	**3.46**

Table 1: **The RQ1 predictor suite on the 341 first-visit decision points.** All rows share one denominator—points whose realized object falls outside the candidate set are scored as misses rather than dropped—and top-k are closed-form expectations over random tie-breaking. Higher is better for coverage and top-k, lower for the effective candidate count $\exp(H)$; best per column in bold. Two tool-routed constructions are deferred to Table 7 in Appendix G.

Baselines and nulls. Every reference is a competing predictor on identical decision points or a null with a closed form; Appendix C.3 inventories all of them. Two are load-bearing enough to name here. The within-episode permutation null preserves the visited multiset and destroys only order, which is what separates "the graph describes where the agent works" from "how it moves". And the prefix-only versus full-trace ablation of the judge's own information set is the only attribution comparison that needs no external reference.

Metrics and statistics. Next-action prediction is scored by top-k accuracy, NLL, calibration error, and effective candidate count $\exp(H)$; attribution methods by positional displacement, the mean of $\mathrm{pos}(\hat{t}_i^M) - \mathrm{pos}(\hat{t}_i^{M'})$ over episodes, which we prefer to raw top-1 disagreement for a reason Appendix D.12 works out. Intervals are episode- or instance-level bootstrap, nulls permutation or closed-form, ranking metrics use the closed-form expectation over random tie-breaking, and multiple comparisons use BH-FDR at $q = 0.10$ (Appendix C.5).

5.3 RQ1: What predicts the next action?

Execution provenance beats code structure on first-visit steps. On the 67.0% of steps that move to a previously unvisited object, a provenance-only predictor reading the most recent tool output attains top-3 $= 0.326$ (CI $[0.276, 0.375]$), against 0.058 for a strengthened dependency-graph predictor and 0.072 for repository frequency (Figure 3a); the paired improvement over the graph is $+0.136$ at top-3 (CI $[+0.102, +0.173]$) and $+0.259$ at top-10. Neither design favours provenance: first-visit steps remove visit history by construction, which is why that row of Table 1 sits at zero coverage, and the graph predictor is deliberately *strengthened* rather than strawmanned (Appendix C.3), yet

still loses to every provenance variant including the uniform one. The mechanism is candidate efficiency: 0.689 of first-visit targets were already mentioned in earlier outputs, and provenance reaches its accuracy with 3.46 effective candidates against a median in the thousands for the graph, which ranks the repository rather than what has been seen (Appendix D.2).

Predictability is heterogeneous, and queries—not paths—are the bottleneck. Figure 3(b) shows that a single aggregate would conceal where difficulty lies: handler steps, at 27.3% of all steps, reach top-3 $= 0.582$; path steps, at 51.1%, reach 0.185; and steps whose object is a novel free-text query, at 21.6%, reach only 0.029. That last number survives an attempt to improve it, and the one parser defect we found biases it *upward*, so it is conservative for the claim (Appendix D.5).

The action must be modelled jointly, and trust is stratified. Tool identity and object type are dependent ($I = 0.159$ nats, $p = 5 \times 10^{-4}$), so a factored $P(\text{tool})P(\text{obj})$ model is misspecified rather than suboptimal: routing the *construction* of the candidate set on the predicted next tool raises top-1 from 0.109 to 0.150 and tightens the set from 39.93 to 12.85 at unchanged calibration. The distribution is also trustworthy only in identifiable regimes: right after a search the next action is accurate and calibrated (top-3 $= 0.469$, ECE $= 0.013$), after a read much harder (0.160), and low predicted entropy is *not* sufficient for trust, since the worst-calibrated stratum is the one with the fewest candidates (Appendix G). RQ1 therefore has a clear answer: the dominant predictor of the next action is the agent's own execution provenance, not repository structure.

5.4 RQ2: What is the code dependency graph actually useful for?

The graph does not describe step-to-step movement. Against the within-episode permutation null, which preserves the visited set and destroys only order, consecutive steps are not meaningfully closer in the graph than chance: one-hop lift 0.94, three-hop 1.02, same-community 1.01 (Figure 7, Appendix F). The lift is a ratio, not a rate: adjacency is common, and the point is that a null preserving the visited multiset predicts it just as well (Appendix D.3). The only quantity clearly exceeding chance is immediate self-repetition, at lift 1.47, a temporal reuse effect (§5.5) rather than graph-driven movement.

The same graph does describe the working region. Its second panel shows a different role for the same object: the visited set has internal edge density 0.269 against a null of only 0.0087, a lift of $30.9\times$, so the files an agent touches form a graph-coherent region even though movement inside it is not graph-driven. The graph also ranks unseen gold files highly once leakage is controlled, at rank quantile 0.049 (CI [0.021, 0.084])—control that matters, since 65% of gold files have already been visited by then. RQ2 therefore answers that the graph is a *scoping prior* rather than a next-step transition model, for the 46.2% of agent objects that resolve to a graph node.

5.5 RQ3: Which reuse patterns are real?

The same null retires a widely reported statistic. The reuse *rate* is a permutation invariant—fixed by the visited multiset, with a null of standard deviation exactly zero here—so the observed tool and object rates, 79.1% and 41.6% over 487 episodes, are arithmetic consequences of trajectory length and vocabulary size rather than evidence of a tendency. What survives permutation is temporal organization: objects are repeated in consecutive steps $1.83\times$ more often than chance ($p \sim 10^{-141}$), and among revisits the target's prior visit count and recency rank exceed the uniform expectation by $1.70\times$ and $1.74\times$. The tool half of the folk claim is false as stated: agents do *not* preferentially repeat a tool ($1.02\times$, $p = 0.51$); the channel carries a fixed transition grammar instead ($1.13\times$ chance), so tool usage is predictable but not by blind repetition. RQ3 therefore separates two conflated notions—reuse *rates* are uninformative, reuse *patterns* are not (Appendix D.4).

5.6 RQ4: At which level does execution uncertainty live?

Cheap action entropy does not proxy replay-based execution uncertainty. The first result rules out a convenient but misleading shortcut: zero-cost predictive entropy from the action model does not track replay variance, at $\rho = -0.088$ at the step level (CI $[-0.244, +0.062]$) and $\rho = -0.219$ at the instance level. A control quantity known to respond does respond, at $\rho = -0.425$ (CI $[-0.695, -0.102]$), so the null reflects an absent relationship rather than a dead outcome measure.

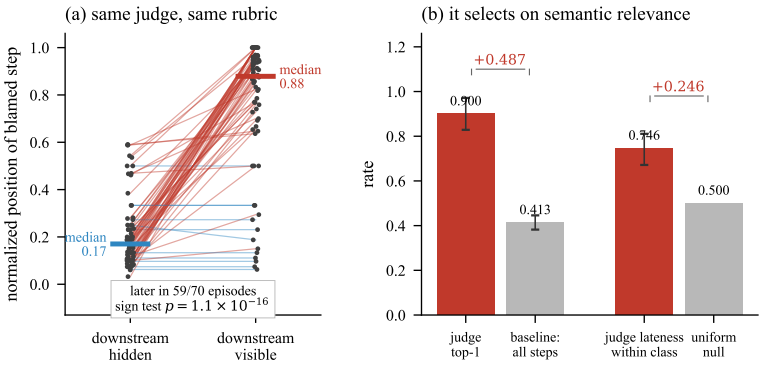

Figure 4: **Collider isolation in full-trace judging.** (a) Each line is one episode, plotted by the normalized position of the blamed step under the prefix-only and full-trace conditions. (b) The class of step the full-trace judge selects, against the share of all steps falling in that class.

	Collider displacement				Gold-prov. enrichment	
Judge	**mean**	**95% CI**	**later**	**sign p**	**rate**	**diff CI**
gpt-5.5 (original)	+0.533	[+.399, +.656]	19/23	4×10^{-6}	0.826*	[+.241, +.500]
qwen3.6-plus (general)	+0.434	[+.299, +.557]	18/23	8×10^{-6}	0.565	[-.068, +.297]
qwen3-coder-plus (code)	+0.394	[+.251, +.551]	15/23	6×10^{-5}	0.826*	[+.222, +.514]

Judge heterogeneity (permuting judge labels within episode): displacement $p = 0.181$, enrichment $p = 0.044$.

Table 2: **The same manipulation across 3 judges** on their 23 common episodes. "later" counts episodes in which the full-trace verdict falls after the prefix-only one; * marks enrichment significantly above the common-episode baseline of 0.443.

The variance is nonetheless structured, and the structure is the instance. Replay variance is mechanically constrained by the mean, collapsing to zero in the top quintiles of v_t, so the probes are computed on residuals after binning on v_t (Figure 8, Appendix F). Once that relation is removed, instance identity explains a substantial share of the remainder, at ICC $= 0.640$ against a permutation null of 0.196 ($p = 5 \times 10^{-4}$), whereas step position contributes little ($\rho = -0.131$) and tool type shows no effect ($p = 0.65$). An apparent exception to this null does not replicate (Appendix D.6).

The consequence is a negative result about measurability, not about power alone. Of 190 intervention-estimated step effects, only 5.3% reach uncorrected $p < 0.05$—indistinguishable from chance—and 0 survive BH-FDR at $q = 0.10$. This budget is already substantial, 1901 continuations over 12.7 hours, so the null is informative: formal identification does not imply measurability at the desired resolution, and the limiting factor is where uncertainty lives rather than engineering. The instrument is coarse and we say so: at $m = 8$ the median replay variance is zero and 22.9% of episodes are degenerate, with every estimated contribution identically zero. RQ4 therefore gives the paper's core answer: execution uncertainty resides primarily at the task level, which is why step-level attribution stays underpowered and why scaling replay alone is unlikely to help.

5.7 RQ5: What do observational attribution methods actually measure?

Making downstream steps visible displaces blame later, which isolates the bias. Figure 4(a) gives the central experiment: we hold judge model and rubric fixed and vary only whether steps downstream of the scored step are visible, a manipulation that should not move the location of blame if full-trace judges were approximating a prefix-conditioned step-local quantity. Instead the verdict moves $+0.537$ later in normalized position (CI $[+0.459, +0.610]$), later in 59 of 70 episodes (sign test $p = 1.1\times10^{-16}$), shifting the mean from 0.214 in the first quarter of the trajectory to 0.751 in the last. Since the world being judged is unchanged, this identifies a property of the evaluation mechanism rather than disagreement with a noisy reference—the signature Figure 1 predicts.

What the judge selects is semantic relevance, biased late within a relevant class. Figure 4(b) asks what the judge picks when it sees everything, and the answer is not arbitrary: it places 0.900 of its top-1 verdicts on steps that operate on or reveal a gold file, against a baseline of only 0.413 among

all steps, and within that class it is biased late, at 0.746 against a uniform null of 0.5 (CI [+0.172, +0.311]). Full-trace judging is thus neither random nor causally faithful: it prefers semantically relevant steps, but selects among them using information unavailable to a prefix-conditioned rule—and it is not a lookup table either, the best zero-LLM rule being only $1.7\times$ better than guessing inside that class (Appendix G). On the subset directly comparable with the intervention-based selector the judge lands on gold-provenance steps at 0.826 against 0.261 (CI [+0.304, +0.783]); since RQ4 showed that reference does not survive FDR correction, we read the gap conservatively—the two procedures measure different quantities, and the judge's is downstream-informed semantic relevance (Appendix D.10).

The mechanism replicates across judges; the content does not. Table 2 repeats the visibility manipulation with 3 judges over their 23 common episodes. Displacement is positive in every case, with every interval excluding zero, every sign test below 10^{-4}, no detectable between-judge heterogeneity ($p = 0.181$), and a spread of only 0.14 against per-judge intervals of roughly ± 0.13. Gold-provenance enrichment behaves oppositely: it reaches significance in only 2 of 3 judges, and its heterogeneity *is* significant ($p = 0.044$). A critique of the *mechanism* therefore transfers across scorers, whereas a critique of *what* they score does not.

A pre-registered criterion we retire rather than bank. Finally, the judge/causal top-1 disagreement rate is 97.1%, which looks like strong support for an attribution critique but sits near its ceiling *under the null* as well, so we retire the criterion instead of banking it (Appendices D.12, E.2). RQ5 therefore shows that observational attribution here provides no validated step-level causal explanation, and that what it measures is better described as downstream-informed relevance.

5.8 Discussion

Why the levels come apart. The three findings are consequences of one another rather than independent observations. Because execution provenance rather than repository structure determines the next action (RQ1–RQ2), the candidate set an agent works over is small, self-generated and history-dependent; because outcome-relevant variance is a property of the instance rather than the step (RQ4), individual substitutions move the outcome too little to be certified at feasible budgets; and because no step-level reference can then be certified, an observational scorer is free to condition on whatever correlates with the outcome, as downstream evidence does (RQ5). Cheap signals therefore live at the levels where structure exists, and the signal practitioners want to read them as lives at the level where it does not.

Current limitations. Two limitations bound the central claim hardest, and Appendix E.3 states all five in full. The intervention-based step reference is formally identified but empirically weak—0 of 190 effects survive BH-FDR—so we cannot certify which step mattered in any individual episode, and the strongest attribution result is therefore a mechanism claim about how a judge's verdict moves rather than a correctness claim about the step it selects. And the setting is file localization, chosen because it is the smallest environment in which the identification assumptions can be checked at all, so process structure in code editing or open-ended tool use may differ. Appendix E.4 states, for each central claim, the observation that would undercut rather than merely bound it.

6 Conclusion

We introduced SCAE, a replay-based estimator derived from a structural causal model of agent execution, and used it to recast process evaluation as a three-level measurement problem rather than a single notion of process quality. SCAE identifies a prefix-conditioned step-level effect, distributes it additively over steps, and selects between observational replay and real intervention from measured rather than assumed local positivity. On 500 file-localization episodes the three levels behave asymmetrically: action prediction is strong but driven by execution provenance rather than the code dependency graph, execution uncertainty is structured at the task rather than the step level, and the step-level effect stays uncertifiable at feasible replay cost even as full-trace judges displace blame with downstream visibility. What a process score means is therefore fixed by its conditioning set rather than by its scorer, which we suggest is a design principle worth adopting beyond localization. We hope this encourages process evaluators whose targets are explicit, whose assumptions are testable, and whose residual identification failures are randomized rather than assumed away.

AI USE STATEMENT

In this work, we used generative AI tools to assist in refining hypotheses, in the mathematical argumentation, and in polishing the text. Additionally, we used generative AI tools to edit software code and to suggest experimental parameters. Separately from this authoring use, LLMs are also the object of study and part of the measurement apparatus itself: the agent under analysis is Codex CLI backed by `gpt-5.5-0424-global`, and the process judges are 3 distinct LLMs (Appendix C.2).

We reviewed all AI-assisted work. No AI-drafted number enters the main text unverified, and the authors reviewed and corrected the AI-assisted code. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work analyzes automated agents operating on public open-source repositories drawn from SWE-bench-Verified; no human subjects, personal data, or annotator labor are involved. All agent executions run in a read-only sandbox with the approval policy disabled, and we verify that zero server-side approval requests are received, so no repository is modified. The main risk we see is misreading the negative results: that step-level attribution is not measurable at feasible cost is not license to deploy unvalidated process scores, and that full-trace judges are systematically biased is not a claim that they are useless. Both are arguments for declaring what a process score conditions on before acting on it.

REPRODUCIBILITY STATEMENT

Every reported number and data figure is regenerated by a CPU-only script that fixes its random seed and writes its output to JSON, and every numeric value in this paper is a macro checked against the analysis output that produced it by an automated verifier; Appendix E.1 lists the exact commands and sensitivity analyses. Appendix C.2 gives the agent, backing model, sandbox policy, environment pinning, estimation points, replay counts, entropy threshold, judge conditions, and v_0 covariates, and Appendix C.1 states the estimation algorithm with the two implementation invariants that are load-bearing and easy to violate. Assumptions 1–3 are proved to yield Eq. equation 5 in Appendix B.6, and their empirical verification is reported in §5.2. Appendix E.2 reports the pre-registration record in full, including the four entries that did not hold and the one criterion we retire.

REFERENCES

Anastasios N. Angelopoulos and Stephen Bates. Conformal prediction: A gentle introduction. *Foundations and Trends in Machine Learning*, 16(4):494–591, 2023.

Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1806.07857.

Ivi Chatzi, Nina Corvelo Benz, Eleni Straitouri, Stratis Tsirtsis, and Manuel Gomez-Rodriguez. Counterfactual token generation in large language models. In *Causal Learning and Reasoning (CLeaR)*, 2025. arXiv:2409.17027.

Mengzhuo Chen, Junjie Wang, Fangwen Mu, Yawen Wang, Zhe Liu, Huanxiang Feng, and Qing Wang. Seeing the whole elephant: A benchmark for failure attribution in llm-based multi-agent systems. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19888–19905, 2026.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

Jiawei He, Jie Jia, Chenbo Liu, Chaoyi Xue, Yapeng Song, Xikai Yang, and Dong Sun. Procctrlbench: Evaluating process-level defects and control preservation in llm coding agents. *arXiv preprint arXiv:2605.20251*, 2026.

Miriam Horovicz and Roni Goldshmidt. Tokenshap: Interpreting large language models with monte carlo shapley value estimation. In *Proceedings of the 1st Workshop on NLP for Science (NLP4Science)*, pp. 1–8, 2024.

Milad Kazemi, Jessica Lally, Ekaterina Tishchenko, Hana Chockler, and Nicola Paoletti. Counterfactual influence in markov decision processes. *arXiv preprint arXiv:2402.08514*, 2024.

Reza Khanmohammadi, Erfan Miahi, Mehrsa Mardikoraem, Simerjot Kaur, Ivan Brugere, Charese Smiley, Kundan S Thind, and Mohammad M Ghassemi. Calibrating llm confidence by probing perturbed representation stability. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10459–10525, 2025.

Vasilis Kontonis, Yuchen Zeng, Shivam Garg, Lingjiao Chen, Hao Tang, Ziyan Wang, Ahmed Awadallah, Eric Horvitz, John Langford, and Dimitris Papailiopoulos. Memento: Teaching llms to manage their own context. *arXiv preprint arXiv:2604.09852*, 2026.

Rui Li and Shuang Cao. From trajectories to graphs: Contract-checked editing for verifier-guided llm reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43259–43306, 2026.

Kaiqu Liang, Haimin Hu, Ryan Liu, Thomas L Griffiths, and Jaime Fernández Fisac. Rlhs: Mitigating misalignment in rlhf with hindsight simulation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 11457–11483, 2026.

Luckeciano C Melo, Panagiotis Tigas, Alessandro Abate, and Yarin Gal. Deep bayesian active learning for preference modeling in large language models. *Advances in Neural Information Processing Systems*, 37:118052–118085, 2024.

Mike Merrill, Alexander Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Shin, Thomas Walshe, E Kelly Buchanan, et al. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *International Conference on Learning Representations*, volume 2026, pp. 40903–40986, 2026.

Susan A. Murphy. Optimal dynamic treatment regimes. *Journal of the Royal Statistical Society: Series B*, 65(2):331–355, 2003.

Preetum Nakkiran, Arwen Bradley, Adam Golinski, Eugene Ndiaye, Michael Kirchhof, and Sinead Williamson. Trained on tokens, calibrated on concepts: The emergence of semantic calibration in llms. In *International Conference on Learning Representations*, volume 2026, pp. 34128–34192, 2026.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2nd edition, 2009.

Suraj Prasai, Mengnan Du, Ying Zhang, and Fan Yang. Knowthyself: An agentic assistant for llm interpretability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 41661–41663, 2026.

Shauli Ravfogel, Anej Svete, Vésteinn Snæbjarnarson, and Ryan Cotterell. Gumbel counterfactual generation from language models. In *International Conference on Learning Representations*, volume 2025, pp. 90486–90512, 2025.

James Robins. A new approach to causal inference in mortality studies with a sustained exposure period. *Mathematical Modelling*, 7(9–12):1393–1512, 1986.

Brendan Ross, Noël Vouitsis, Atiyeh Ashari Ghomi, Rasa Hosseinzadeh, Ji Xin, Zhaoyan Liu, Yi Sui, Shiyi Hou, Kin Kwan Leung, Gabriel Loaiza-Ganem, et al. Textual bayes: Quantifying prompt uncertainty in llm-based systems. In *International Conference on Learning Representations*, volume 2026, pp. 133607–133630, 2026.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for llm reasoning. In *International Conference on Learning Representations*, volume 2025, pp. 60808–60838, 2025.

Chenglei Si, Tatsunori Hashimoto, and Diyi Yang. The ideation-execution gap: Execution outcomes of llm-generated versus human research ideas. In *International conference on learning representations*, volume 2026, pp. 74195–74240, 2026.

Dennis Ulmer, Martin Gubri, Hwaran Lee, Sangdoo Yun, and Seong Oh. Calibrating large language models using their generations only. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15440–15459, 2024.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, 2005.

Hanlin Wang, Jian Wang, Chak Tou Leong, and Wenjie Li. Steca: Step-level trajectory calibration for llm agent learning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 11597–11614, 2025a.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024. arXiv:2312.08935.

Qian Wang, Tianyu Wang, Zhenheng Tang, Qinbin Li, Nuo Chen, Jingsheng Liang, and Bingsheng He. Megaagent: A large-scale autonomous llm-based multi-agent system without predefined sops. In *Findings of the association for computational linguistics: ACL 2025*, pp. 4998–5036, 2025b.

Muning Wen, Ziyu Wan, Jun Wang, Weinan Zhang, and Ying Wen. Reinforcing llm agents via policy optimization with action decomposition. *Advances in Neural Information Processing Systems*, 37: 103774–103805, 2024.

Chunqiu Steven Xia, Yinlin Deng, Soren Dunn, and Lingming Zhang. Agentless: Demystifying LLM-based software engineering agents. *arXiv preprint arXiv:2407.01489*, 2024.

Yuxi Xia, Pedro Henrique Luz De Araujo, Klim Zaporojets, and Benjamin Roth. Influences on llm calibration: A study of response agreement, loss functions, and prompt styles. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3740–3761, 2025.

Frank Fangzheng Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, et al. Theagentcompany: benchmarking llm agents on consequential real world tasks. *Advances in Neural Information Processing Systems*, 38, 2026a.

Xiaoliang Xu, Huang Yuan, Junmei Wang, and Can Xu. Gcig: Graphrag-based cross-document instruction generation for boosting llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17593–17609, 2026b.

Matthew Yang, Hao Bai, Ian Wu, Gene Yang, Amrith Setlur, and Aviral Kumar. Int: Self-proposed interventions enable credit assignment in llm reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 85054–85091, 2026.

Guibin Zhang, Junhao Wang, Junjie Chen, Wangchunshu Zhou, Kun Wang, and Shuicheng Yan. Agentracer: Who is inducing failure in the llm agentic systems? In *International Conference on Learning Representations*, volume 2026, pp. 11377–11399, 2026a.

Zheng Zhang, Ziwei Shan, Kaitao Song, Yexin Li, and Kan Ren. Linking process to outcome: Conditional reward modeling for llm reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 87995–88016, 2026b.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023. arXiv:2306.05685.

A Extended Related Work

§2 states how our treatment differs from the four closest lines of work in compressed form. This appendix gives the full discussion.

Counterfactual Reasoning in Discrete Sequential Systems. Counterfactual reasoning over discrete action sequences is commonly grounded in the Gumbel–max structural causal model (Ravfogel et al., 2025), which also provides the core mechanism underlying SCAE. By representing categorical sampling as perturb-and-argmax (Khanmohammadi et al., 2025), this line of work yields a tractable posterior over latent noise variables conditioned on an observed draw. Building on this formulation, prior studies (Chatzi et al., 2025; Kazemi et al., 2024) have developed counterfactual explanations for action sequences under known MDPs, extended the framework to token-level generation, and analyzed how counterfactual influence decays as trajectories diverge. However, these approaches generally assume access to known transition mechanisms and a fixed action space. In contrast, LLM agents operating over code repositories provide neither. This introduces two challenges that are largely absent in standard simulators: positivity may fail locally at individual decision points, and context compaction may destroy the invertibility of the state mapping. Accordingly, SCAE does not assume a known transition mechanism; instead, it estimates one through replay. It also treats positivity as an empirical property to be measured rather than a standing assumption, and regards context compaction as an identification failure rather than an approximation error.

Step-Level Credit Assignment and Process Supervision. Process reward models supervise intermediate reasoning steps and have been shown to outperform outcome-only supervision on reasoning tasks (Zhang et al., 2026b; Setlur et al., 2025; Wang et al., 2024). Their primary purpose, however, is to improve policy learning rather than to explain the role of a particular step in a single trajectory. In practice, their supervision signals are often derived from Monte Carlo estimates over completed rollouts, for example by judging whether a given prefix can still reach a correct answer. As a result, information from downstream steps is reintroduced into the label, and the contribution of the current step is not cleanly separated from the effect of what follows. A related observation applies to hindsight and counterfactual credit assignment, which deliberately use future information to reduce the variance of policy-gradient estimates (Arjona-Medina et al., 2019; Liang et al., 2026; Yang et al., 2026). Such quantities are well suited for optimization, but they do not correspond to per-episode causal contributions. Another relevant line of work concerns axiomatic attribution methods, such as Shapley values (Horovicz & Goldshmidt, 2024), which satisfy an additive property analogous to our telescoping identity, but at the cost of exponentially many re-evaluations. In contrast, our goal is not to propose a stronger step scorer. Rather, we clarify what information a step score must condition on in order to admit a causal interpretation, show that this distinction systematically changes which step is blamed, and provide an estimator that preserves additivity through a single fitted object instead of requiring 2^T re-executions.

Process-Oriented Evaluation of Coding Agents. Current evaluation of coding agents remains centered primarily on terminal task success (Merrill et al., 2026; Xu et al., 2026a; Xia et al., 2024). In parallel, a complementary line of work studies recurrent execution defects and calibrates detector outputs against human process annotations (He et al., 2026). These approaches target a different estimand: the probability that an annotator would mark a step as defective given trajectory evidence (Zhang et al., 2026a; Li & Cao, 2026). As such, they address a measurement problem about annotation agreement rather than a causal inference problem. By contrast, our focus is the causal effect of substituting one step on the final outcome. This estimand requires that the randomization induced by the logging policy be preserved and is not identifiable under context compaction (§4.1). Because the two estimands are fundamentally different, a step may be consistently judged defective by annotators while contributing no identifiable causal effect to the final outcome. Annotation agreement alone therefore cannot establish whether a process score measures what it is intended to measure. SCAE differs in that it evaluates attribution against a verifiable terminal outcome, thereby turning the question of what a process score measures into an empirically testable one rather than a purely definitional one.

Uncertainty Quantification for Language Models. Research on calibration and confidence estimation for language models has largely focused on single-turn generation, where each output is associated with a single scalar confidence value (Xia et al., 2025; Ulmer et al., 2024; Nakkiran et al., 2026). More broadly, decomposing predictive uncertainty into aleatoric and epistemic com-

ponents is standard in Bayesian deep learning (Melo et al., 2024; Ross et al., 2026), while conformal prediction provides distribution-free coverage guarantees under exchangeability (Vovk et al., 2005; Angelopoulos & Bates, 2023). These tools are well suited to settings in which one prediction corresponds to one confidence estimate. The agent setting introduces two complications that they do not directly address. First, uncertainty compounds across sequential decisions. Second, the agent's own actions determine what information becomes observable later, making uncertainty inherently trajectory-dependent. Consequently, a single terminal confidence score cannot reveal where reliability was lost, nor does it specify the level of the trajectory at which confidence should be attached. File localization itself is a classical subproblem in automated program repair, typically evaluated by retrieval-style accuracy against the files touched by a reference patch (Xia et al., 2024). We adopt localization not as a performance target to improve, but as the minimal controllable setting in which the identification conditions required by our analysis can be satisfied. On this basis, we treat the level at which uncertainty resides as an empirical question rather than a modeling choice, and examine whether uncertainty is primarily a property of the task or of individual steps.

Structural Priors for Agent Navigation. Existing coding-agent systems make substantial use of repository structure. Retrieval and localization modules commonly rank files according to similarity or dependency information, and agent scaffolds expose repository-shaped tools through which the model navigates the codebase (Xu et al., 2026b; Edge et al., 2024). As a result, the assumption that code structure guides the agent's next move is already built into current system designs. Empirically, the files visited within a single episode do tend to cluster in the dependency graph. However, such clustering does not by itself imply that local step-to-step transitions are governed by graph structure. To our knowledge, prior work has not explicitly tested this assumption against a null model that holds the working region fixed, and therefore has not separated descriptive clustering from the actual transition mechanism. Our analysis fills this gap and shows that repository structure is better understood as a prior over the agent's working region than as a local transition model for next-step prediction. Moreover, we identify an alternative structural signal—the agent's own execution provenance—that is more directly predictive of local behavior. This signal provides a more appropriate conditioning basis for step-level prediction and context-selection policies.

B Notation and Formal Material

This appendix supplies the formal material behind §4: the notation used throughout (Table 3), the consequence of each conditioning set for current practice, where on the causal ladder our estimates sit, the three ways identification can fail, and the proofs of Proposition 1 and Theorem 1.

B.1 Notation

Table 3 collects the symbols reused across the SCM, the replay estimator, and the experiments. We list them centrally because the same objects—especially the realized prefix $\tau_{1:t-1}$, the anytime value v_t, the increment Δ_t, and the step-level contrast $\theta_t(a, a')$—appear across multiple sections and levels of analysis.

B.2 Consequences of the conditioning set for current practice

Proposition 1 licenses exactly one conditioning set: the realized prefix. Table 4 states, for each departure from it that appears in current practice, the resulting bias and the signature it should leave in data. The mechanism in the first row is standard — conditioning on a descendant of the treatment opens a non-causal path and biases the estimate (Pearl, 2009) — but it is not usually applied to LLM-as-a-judge attribution (Zheng et al., 2023), which presents the entire trajectory and the realized outcome and then asks which step was at fault. That row is also the only one yielding a falsifiable prediction without requiring that our own estimator be correct, which is why §5.7 tests it and why we foreground it.

B.3 The causal ladder, instantiated for localization

Table 5 places localization-style process evaluation on the causal ladder. The key distinction for this paper is between restarting from intervention alone (Level II), retaining the realized prefix while changing one action (Level II.5), and fixing realized noise at the intervened step (Level III). The

Symbol	Definition
q	natural-language issue text of an episode
$G = (V, E)$	repository graph over files, modules and symbols
$F^\star$	gold target set: files modified by the reference patch
$\hat{F}$	file set predicted by the agent
Y	task-level outcome $\mathbb{1}[F^\star \subseteq \hat{F}]$
Y_f, Y_{rec}	per-file indicator; recall $\lvert F^\star \cap \hat{F}\rvert/\lvert F^\star\rvert$
A_t, O_t	action taken at step t; tool output it returns
S_t	agent state at step t; $S_1 = \sigma(q, \text{sys})$
$\tau_{1:t}$	trace prefix $(A_1, O_1, \ldots, A_t, O_t)$
T	number of steps in an episode; t/T is normalized position
Z	latent task difficulty (exogenous)
$G_\pi^{(t)}$	decoding noise at step t, $\text{Gumbel}(0, 1)$ (exogenous)
$U_{\text{env}}^{(t)}$	environment noise at step t (exogenous, pinned)
$U^{(\geq t)}$	all exogenous noise from step t onward
π, $\tilde{\pi}$	deployed policy; raised-temperature perturbed policy
$\widehat{H}(\widehat{\pi}_t)$	estimated action entropy at step t, equation 7
τ_H	entropy threshold selecting the branch of equation 8
m	replayed continuations per estimation point
v_t	anytime value $\mathbb{E}[Y \mid \tau_{1:t}, q, G]$
v_0	pre-execution prediction $\mathbb{E}[Y \mid q, G]$
Δ_t	step contribution $v_t - v_{t-1}$
U_t^{exec}	replay variance of the outcome at estimation point t
$\theta_t(a, a')$	step-level causal contrast of equation 1
PN_t	probability of necessity at step t
$\text{Infl}(t)$	counterfactual influence at step t
$U_{\text{task}}, U_{\text{policy}}, U_{\text{model}}$	the three components of equation 13
Σ	Laplace posterior covariance of the v_0 model
$\mathcal{C}(q)$	conformal prediction set for the target files

Table 3: Notation used throughout the paper. Symbols are introduced in §4 and reused without redefinition in §5.

Practice	Conditions on	Consequence
Full-trace LLM judge	prefix + *successors* + outcome	Successors are descendants of A_t, and conditioning on a descendant opens a biasing path. Predicted signature: blame displaced toward *later* steps.
Process reward models	nominally the prefix	Labels back-filled from completed trajectories reintroduce future information.
Ablation with restart	nothing	Resamples all noise, so it estimates a marginal interventional contrast and discards the realized history.
Data-flow "dead step" analysis	syntactic dependencies	Conflates data-flow reachability with causal effect on the outcome (§B.7).

Table 4: Each row misestimates equation 1 for a distinct reason, and none of the four reasons is unmeasured confounding. Only the first row yields a signature testable without a causal reference, which is why it is the attribution result we foreground in §5.7.

intermediate Level II.5 is the one most often conflated with standard ablation in agent evaluation, and it is the level our replay design is intended to isolate.

B.4 Position on the causal ladder

With API-served models we obtain per-token log-probabilities only in a reference arm; in the main arm we estimate $\widehat{\pi}_t$ over action equivalence classes by resampling. The intervened step therefore admits Gumbel–max abduction and is genuinely Level III, whereas the continuation marginalizes residual policy noise conditional on the prefix and is Level II.5 in the terms of Table 5. Environment noise is Level III throughout, being pinned by construction. We therefore claim action-level

Level	Expression	Meaning for localization	Typical practice
I	$P(\hat{F} \mid q)$	association between query and result	retrieval recall@k
II	$P(\hat{F} \mid do(a'))$	alter a step and *restart*	tool-ablation studies
II.5	$P(\hat{F} \mid \tau_{1:t-1}, do(a'))$	fix realized history, alter one step	**SCAE (continuation)**
III	$P(\hat{F}_{a'} \mid \tau_{\text{obs}})$	fix realized noise, alter one step	**SCAE (intervened step)**

Table 5: Levels II and II.5 differ in whether the episode's own history is retained. Most agent ablation studies operate at Level II while being described in counterfactual language, which is the confusion the distinction is meant to prevent. §B.4 states where the boundary falls for SCAE.

counterfactuals with a prefix-conditioned continuation, and *not* exact token-level counterfactuals. Quantifying the gap requires the reference arm, whose construction we describe but whose results are not included here.

B.5 Where identification breaks

Three failure modes deserve naming, and none of them is confounding.

Positivity. This is the binding constraint in practice. Necessity-type quantities are the ones it binds hardest on: the probability of necessity at step t,

$$\text{PN}_t = P\big(Y_{A_t=a_t^\star}{=}1 \;\big|\; \tau_{1:t-1}, A_t{=}a_t, Y{=}0\big),$$

asks whether the outcome would have been correct had the agent taken the better action $a_t^\star$, given that it took a_t and failed. It is defined only when $\pi(a_t^\star \mid \tau_{1:t-1}) > 0$. When the policy is effectively deterministic at step t this fails, and no amount of observational data recovers the quantity, because the required comparison arm is never realized. This is why §C.4 measures $\pi(\cdot \mid \tau_{1:t-1})$ rather than assuming it, and why equation 8 switches to a soft intervention exactly where the measurement says overlap is absent.

Context compaction. Let φ denote the harness's compaction map, so that the state actually conditioned on is $\varphi(h_t)$ rather than h_t. When φ discards content, $h_t \mapsto \varphi(h_t)$ is not invertible and there exists $D_t \notin \tau_{1:t-1}$ that influenced A_t. Assumption 1 then fails, Step 1 of the proof in §B.6 does not go through, and D_t acts as an unmeasured confounder between A_t and Y. The estimand is unidentified, not merely harder to estimate: no reweighting of the observed trace recovers it, because the confounder is absent from the log by construction. What makes this tractable in principle is that compaction is triggered by the harness rather than by nature, so it can be *randomized* — assigning compaction at a fixed prefix converts a sensitivity-analysis problem into a controlled experiment. We reserved the hook for this but did not run it, since no episode in our sweep compacted spontaneously.

Counterfactual influence decay. Even with Gumbel–max abduction at the intervened step, the coupling between the factual and counterfactual continuations weakens as the two paths diverge, so a nominal Level III quantity degrades toward a Level II one over long horizons (Kazemi et al., 2024). To keep this measurable rather than rhetorical we define

$$\text{Infl}(t) = \text{TV}\Big(P\big(\hat{F} \mid \tau_{\text{obs}}, do(a')\big),\; P\big(\hat{F} \mid do(a')\big)\Big),$$

the total-variation distance between the counterfactual predicted-set distribution that retains the observed trace and one that discards it. $\text{Infl}(t)$ near zero means the realized history no longer constrains the continuation, so the estimate should be reported as interventional rather than counterfactual.

B.6 Proofs

Throughout, we read Assumption 2 in its joint form: the decoding noises $\{G_\pi^{(t)}\}_{t\geq 1}$ are mutually independent and independent of (Z, U_{env}). This is the standard exogeneity assumption for a sampler whose temperature is a decoder parameter, and it is what "exogenous" asserts; we make it explicit because the proof uses independence across steps, not only at step t.

B.6.1 Proof of Proposition 1

Step 1: conditional on the prefix, the action is a function of decoding noise alone. Fix a prefix value $\tau_{1:t-1} = \tau$ with $P(\tau_{1:t-1} = \tau) > 0$. By Assumption 1 there is a deterministic map f with

$S_t = f(q, \tau)$, so on the conditioning event S_t takes a single fixed value. By equation 2,

$$A_t = \arg\max_k \big(\log \pi_k(f(q,\tau)) + G^{(t)}_{\pi,k} \big) =: g_\tau\big(G^{(t)}_\pi\big),$$

a measurable function of $G^{(t)}_\pi$ only.

Step 2: that noise is independent of the prefix and of the continuation noise. The prefix $\tau_{1:t-1}$ is, by the structural equations, a measurable function of $\big(q, Z, \{G^{(s)}_\pi\}_{s<t}, \{U^{(s)}_{\text{env}}\}_{s<t}\big)$. Write $U^{(\geq t)} = \big(\{G^{(s)}_\pi\}_{s>t}, \{U^{(s)}_{\text{env}}\}_{s\geq t}\big)$. By Assumption 2 in its joint form, $G^{(t)}_\pi$ is independent of the collection $\big(Z, \{G^{(s)}_\pi\}_{s\neq t}, U_{\text{env}}\big)$, hence

$$G^{(t)}_\pi \;\perp\; \big(Z,\; U^{(\geq t)},\; \tau_{1:t-1}\big). \tag{9}$$

Combining equation 9 with Step 1, and noting that conditioning on $\tau_{1:t-1}$ leaves the law of $G^{(t)}_\pi$ unchanged,

$$A_t \;\perp\; \big(Z,\; U^{(\geq t)}\big) \;\big|\; \tau_{1:t-1},$$

which is the first claim.

Step 3: the potential outcome is a function of the continuation noise. Fix an action a. Recursively applying $O_s = \mathcal{E}(A_s, G; U^{(s)}_{\text{env}})$, $S_{s+1} = \mathcal{T}(S_s, A_s, O_s)$ and $\hat{F} = \phi(S_T)$ from step t onward, with A_t set to a, the potential outcome under the intervention is a deterministic function

$$Y_{A_t=a} \;=\; h\big(a,\; \tau_{1:t-1},\; U^{(\geq t)}\big), \tag{10}$$

where determinism of each O_s given $(A_s, G, U^{(s)}_{\text{env}})$ is exactly Assumption 3. By Step 2 and equation 10, $A_t \perp Y_{A_t=a} \mid \tau_{1:t-1}$ for every a, which is conditional ignorability at step t.

Step 4: identification. Consistency holds by construction in a structural causal model: on the event $\{A_t = a\}$ we have $Y = Y_{A_t=a}$. Therefore, for any a with $\pi(a \mid \tau_{1:t-1}) > 0$, so that the conditional expectation is well defined,

$$\begin{aligned}
\mathbb{E}[Y \mid \tau_{1:t-1}, do(A_t{=}a)] &\overset{\text{(i)}}{=} \mathbb{E}[Y_{A_t=a} \mid \tau_{1:t-1}] \\
&\overset{\text{(ii)}}{=} \mathbb{E}[Y_{A_t=a} \mid \tau_{1:t-1}, A_t{=}a] \\
&\overset{\text{(iii)}}{=} \mathbb{E}[Y \mid \tau_{1:t-1}, A_t{=}a].
\end{aligned}$$

where (i) is the definition of the interventional expectation in $\mathcal{M}$, (ii) is Step 3, and (iii) is consistency. This is equation 5, and it is the one-step g-formula (Robins, 1986) with a *known* assignment mechanism. □

Relation to dynamic treatment regimes. The sequential structure here is formally that of a dynamic treatment regime (Robins, 1986; Murphy, 2003), but the difficulty is inverted. In epidemiological applications the central obstacle is time-varying confounding: the analyst must argue that everything the treatment decision depended on was recorded. Here that argument is free, because the policy is a function of a context we log in full and its decoding noise is a decoder property. What is *not* free is positivity, which those applications typically obtain from natural variation and which we must measure and then work around (§B.5, §C.4).

Remark on what the proof does and does not deliver. The positivity requirement in Step 4 is not a technicality but the binding constraint in practice, which is what forces the hybrid estimator of equation 8. Note also that Step 2 is where the argument fails for the conditioning sets of Table 4: conditioning additionally on $A_{t'}$ for $t' > t$ or on Y conditions on functions of $U^{(\geq t)}$, which breaks equation 9 and hence Step 3. Similarly, if the harness compacts context then S_t is not a function of $(q, \tau_{1:t-1})$, Assumption 1 fails, and Step 1 does not go through.

B.6.2 Proof of Theorem 1

Telescoping. By definition $\Delta_t = v_t - v_{t-1}$, so $\sum_{t=1}^{T} \Delta_t = v_T - v_0$ as a finite telescoping sum. The outcome $Y = \mathbb{1}[F^\star \subseteq \phi(S_T)]$ is a deterministic function of S_T, which by Assumption 1 is a function of $(q, \tau_{1:T})$; therefore $v_T = \mathbb{E}[Y \mid \tau_{1:T}, q, G] = Y$ almost surely, and $v_0 = \mathbb{E}[Y \mid q, G]$ by definition. This gives equation 6.

The action/observation decomposition. Insert the intermediate conditioning set $(\tau_{1:t-1}, A_t{=}a_t)$, which lies between $\tau_{1:t-1}$ and $\tau_{1:t} = (\tau_{1:t-1}, a_t, o_t)$:

$$\Delta_t = \underbrace{\Big(\mathbb{E}[Y \mid \tau_{1:t-1}, A_t{=}a_t] - \mathbb{E}[Y \mid \tau_{1:t-1}]\Big)}_{\text{(A)}} + \underbrace{\Big(\mathbb{E}[Y \mid \tau_{1:t}] - \mathbb{E}[Y \mid \tau_{1:t-1}, A_t{=}a_t]\Big)}_{\text{(B)}}. \tag{11}$$

Term (B) of equation 11 is the contribution of the observation: the change in expected outcome from learning $O_t = o_t$, holding the action fixed. For term (A), the law of total expectation over the policy's action distribution gives

$$\mathbb{E}[Y \mid \tau_{1:t-1}] = \sum_a \pi(a \mid \tau_{1:t-1})\, \mathbb{E}[Y \mid \tau_{1:t-1}, A_t{=}a] = \mathbb{E}_{a\sim\pi}\big[\mathbb{E}[Y \mid \tau_{1:t-1}, A_t{=}a]\big],$$

where the sum ranges over the support of $\pi(\cdot \mid \tau_{1:t-1})$, so every conditional expectation appearing in it is well defined. Applying Proposition 1 to each term replaces every observational conditional by its interventional counterpart, giving the action effect relative to the policy's own baseline,

$$\begin{aligned}\Delta_t = &\underbrace{\Big(\mathbb{E}[Y \mid \tau_{1:t-1}, do(A_t{=}a_t)] - \mathbb{E}_{a\sim\pi}[\mathbb{E}[Y \mid \tau_{1:t-1}, do(A_t{=}a)]]\Big)}_{\substack{\text{action effect}\\ \text{vs. policy baseline}}} \\ &+ \underbrace{\Big(\mathbb{E}[Y \mid \tau_{1:t}] - \mathbb{E}[Y \mid \tau_{1:t-1}, A_t{=}a_t]\Big)}_{\substack{\text{observation}\\ \text{contribution}}}.\end{aligned} \tag{12}$$

Remark. Two readings of equation 12 are excluded. The baseline is the policy's *own* action distribution, so Δ_t answers how much better or worse a step was than what this agent would have done there, not than the optimal action — the latter is a necessity-type contrast requiring intervention (§B.5). And equation 6 holds for population quantities, so with finitely many replays per v_t the empirical sum carries error; we use that error as an internal consistency check rather than reporting it as exact. The proof uses boundedness of Y only through the existence of the conditional expectations, so both identities hold verbatim for the recall outcome $Y_{\text{rec}} \in [0, 1]$ and for the per-file indicator Y_f, which is what licenses the outcome choice in §4.4 without restating the theorem.

B.7 Data-flow reachability is not causal effect

Data-flow "inertness" is sometimes offered as a proxy for causal irrelevance: if a step's output is never read downstream, the step is taken to have had no effect. The two notions come apart in both directions, which is why Table 4 lists this as a distinct failure mode.

Inert but causally potent. A `grep` that returns nothing contributes no content to any subsequent context window, so it is inert by any data-flow criterion. It may nonetheless cause the agent to abandon the correct directory, because absence of information is itself information: the empty result licenses the inference that the target is elsewhere. Under our estimand this shows up as a large $|\Delta_t|$ at a step with no outgoing data flow.

Data-carrying but causally inert. Conversely, a long file read may inject thousands of tokens into the context while leaving every subsequent action unchanged. Fixing the prefix and deleting the read verifies this directly: if the continuation is unchanged across replays, the step's contribution is zero despite its data-flow footprint being large. The asymmetry matters because taxonomies that score steps by downstream data flow will systematically misrank exactly the steps that a causal criterion identifies as decisive.

C Implementation and Experimental Details

C.1 Estimation algorithm

Two implementation invariants are load-bearing and easy to violate. First, the continuation prompt must be byte-identical across all arms: it enters the context and is therefore part of the estimand, so any difference between arms becomes an undeclared intervention. Second, the spliced observation o' must be the genuine result of executing a'; substituting a plausible-looking output intervenes on the observation rather than the action, which is a different causal quantity.

Algorithm 1 SCAE: step contributions with per-step identification

Require: episode (q, repo), rollout $\tau_{1:T}$, outcome Y, estimation points $\mathcal{P} \subseteq \{1, \dots, T\}$, replays m, threshold τ_H
Ensure: $\{\widehat{\Delta}_t\}_{t \in \mathcal{P}}$, entropy $\{\widehat{H}_t\}$

reset repository to base commit; pin locale, timezone, terminal width, hash seeds
for $t \in \mathcal{P}$ **do**
  $\tau_{1:t-1} \leftarrow \textsc{TruncateBeforeStep}(\tau, t)$ ▷ rewrite rollout log at item granularity
  $\textsc{AssertNoDanglingCalls}(\tau_{1:t-1})$
  $\mathcal{A} \leftarrow \emptyset$
  for $i = 1$ to m **do** ▷ common random numbers: shared prefix and pinned env
    reset repository; fork agent from $\tau_{1:t-1}$; resume with the byte-identical continuation prompt
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{\textsc{ActionBucket}(A_t^{(i)})\}$ ▷ bucket $= (\text{tool}, \text{target node in } G)$
  end for
  $\widehat{H}_t \leftarrow \textsc{Entropy}(\mathcal{A})$
  if $|\{\mathcal{A}\}| \geq 2$ **and** $\widehat{H}_t > \tau_H$ **then** ▷ overlap available: observational branch
    $\widehat{v}_t \leftarrow \frac{1}{m} \sum_i Y^{(i)}$ from replays at t; $\widehat{v}_{t-1}$ likewise from replays at $t-1$
    $\widehat{\Delta}_t \leftarrow \widehat{v}_t - \widehat{v}_{t-1}$
  else ▷ degenerate policy: soft intervention, Asm. 4
    draw a' adjacent to a_t in G, or an oracle action inspecting a gold file
    $o' \leftarrow$ **execute** a' for real in the reset repository ▷ never a fabricated output
    splice (a', o') into $\tau_{1:t-1}$; resume m times; average to get $\widehat{\mathbb{E}}[Y \mid \tau_{1:t-1}, do(A_t{=}a')]$
    $\widehat{\Delta}_t \leftarrow \widehat{\mathbb{E}}[Y \mid \tau_{1:t-1}, do(A_t{=}a')] - \widehat{\mathbb{E}}[Y \mid \tau_{1:t-1}, do(A_t{=}a_t)]$
  end if
  record which branch was taken, so that the reported estimand is labelled π or $\tilde{\pi}$
end for

C.2 Experimental settings

Agent and model. Codex CLI driven over its JSON-RPC app-server with `capabilities.experimentalApi` enabled, so that thread paths are exposed and forking by path is available. Sandbox `read-only`, approval policy `never`; we verify that zero server-side approval requests are received, since a nonzero count would mean the read-only restriction was not in force. Backing model `gpt-5.5-0424-global`. Per-turn timeout 900 s.

Environment pinning. Locale, timezone, terminal width and hash seeds are fixed; tool outputs are totally ordered and truncation points fixed. §5.2 reports the empirical verification of this pinning, at 1.00 byte-identical replay over 100 actions.

Estimation points and replays. Attribution uses $m = 8$ replays per estimation point and 5 equidistant estimation points per episode, with the recall outcome of §4.4. Entropy probes use 4–6 replays per point, which is why Table 6 reports entropies on a coarse grid: with 4 replays the attainable nonzero values are 0.56, 1.04 and 1.39 nats. The entropy threshold is $\tau_H = 0$, so the observational branch is taken exactly when the replays realize at least two distinct action buckets; this is the most permissive threshold, and it makes the reported 50% an upper bound on how often observational identification is available.

Judge conditions. Both judge conditions use the same model and the same rubric, and differ only in the visible portion of the trace: the full-trace condition sees $\tau_{1:T}$ and the realized outcome Y; the prefix-only condition sees a fixed 60% prefix with the outcome withheld. The truncation fraction is fixed in advance and does not depend on any estimate. This matters, and §E.2 records why: an earlier design truncated just before the step selected by the causal estimate, which would have produced a positive displacement even under the null.

Covariates for v_0. The 7 pre-execution covariates are: issue description length; count of explicitly named files; presence of a traceback; error-term density; repository size; distinct identifier count; and the number of repository files containing those identifiers, which serves as an ambiguity proxy for

U_{task}. None of these may read the trace, since doing so would leak execution-time information into a pre-execution predictor.

C.3 THE BASELINE SUITE IN FULL

Because this paper compares *measurement procedures* rather than agents, each research question carries its own reference set, and every reference is either a competing predictor evaluated on the same points or a closed-form null. We list the full inventory here, since §5.2 only summarizes it.

Next-action prediction (§5.3). Nine predictors, all scored on the same decision points with uncovered points counted as misses rather than dropped (Table 1): a repository-frequency marginal; a visit-history frequency model; a *strengthened* code-dependency-graph predictor that seeds a multi-source proximity ranking from the visited set rather than the last step alone; four execution-provenance variants (uniform over previously mentioned paths, recency-weighted, frequency-weighted, and last-output-only with within-output position ranking); and two tool-routed candidate-set constructions. We additionally contrast a factored $P(\text{tool})P(\text{obj})$ model against a routed one (Table 7). The graph predictor is deliberately strengthened rather than strawmanned, because the claim of §5.3 is that code structure loses even when given every advantage we could construct.

Graph and reuse structure (§5.4–§5.5). A within-episode permutation null, which preserves the multiset of visited objects and destroys only their order, and random same-size file sets for the region comparison. The permutation null is the load-bearing choice: it holds the working region fixed, which is exactly what separates "the graph describes where the agent works" from "the graph describes how it moves". Revisit statistics are compared against closed-form uniform expectations rather than sampled draws.

Execution uncertainty (§5.6). Two controls in opposite directions. The superseded Bernoulli entropy approximation serves as a negative control, and a quantity known to respond serves as a positive control at $\rho = -0.425$; without the latter, a null relating action entropy to replay variance would be uninterpretable, since it could reflect a dead outcome measure rather than an absent relationship. Structural probes with no mechanical coupling to the mean (step position, tool type, instance identity) are used in place of probes that are mechanically coupled.

Attribution (§5.7). Four reference sets. First, 3 judge models including a code-specialised one (Table 2). Second, a prefix-only against a full-trace ablation of the judge's own information set, which is the only comparison here that needs no external reference. Third, five zero-LLM structural rules (Table 12), which test whether the judge is reducible to a positional or provenance heuristic. Fourth, a uniform-random selector reported at its exact per-episode expectation $1/T_i$ rather than as a sampled draw, so that the chance level carries no Monte Carlo noise.

Two families we do not implement. Back-filled process reward models and ablation-with-restart are absent, and both leave a gap we state rather than paper over. A trained process reward model might recover causal structure despite misspecified supervision, which our theoretical account predicts it will not but our experiments do not test. Ablation-with-restart, being a different estimand (Table 4), would need a separate comparison asking how far the marginal contrast departs from the history-conditioned one — a measurable quantity we have not measured.

C.4 POSITIVITY PROBE: ENTROPY AT EACH DECISION POINT

Identification of equation 1 needs overlap at the step being estimated, and §4.3 treats this as something to measure rather than grant. Probing repeated replays from identical prefixes, action entropy is non-zero at 50% of 10 probed decision points across 3 episodes, with a rank correlation with normalized position of +0.072. Table 6 gives every probed point and Figure 5 plots them.

C.5 STATISTICAL CONVENTIONS

Conventions are fixed once and applied throughout, so that no comparison is reported under a rule chosen after seeing it. Confidence intervals are episode- or instance-level bootstrap. Nulls are within-episode permutation or closed-form expectation, and never sampled where a closed form exists. Ranking metrics use the closed-form expectation over random tie-breaking, since count-based

Episode	Entropy (nats) by normalized position	Pattern
`django-10097` (16 steps)	$0.12{:}\,0 \to 0.38{:}\,1.04 \to 0.62{:}\,1.04$	det.–stoch.–stoch.
`astropy-14369` (29 steps)	$0.10{:}\,0 \to 0.31{:}\,0.56 \to 0.52{:}\,0.56 \to 0.76{:}\,0$	det.–stoch.–det.
`astropy-12907` (8 steps)	$0.25{:}\,0.45 \to 0.50{:}\,0 \to 0.75{:}\,0$	stoch.–det.

Table 6: Action entropy at all 10 probed decision points across 3 episodes, which realize three different patterns. Entropies fall on a coarse grid because each point is probed with 4–6 replays: at 4 replays the attainable non-zero values are 0.56, 1.04 and 1.39 nats.

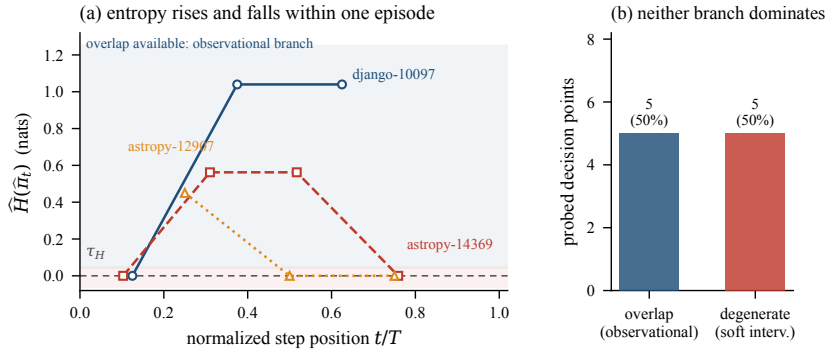

Figure 5: **Overlap is a local property, not a global one**. Each marker is one probed decision point, positioned by its normalized position in the episode and coloured by episode. Points on the zero line admit no observational identification and are routed to the intervention branch of equation 8; points above it are estimated observationally under Proposition 1. We report the sample size plainly: 10 points establish that overlap varies *within* episodes, which is all the estimator requires, but not the distribution of that variation, and the near-zero rank correlation should not be read as evidence of no relationship.

scorers tie heavily and a single arbitrary tie-break would be an undeclared design choice. Predictors with different candidate-set sizes are compared on a *common denominator*, with points whose realized action falls outside the candidate set scored as misses rather than dropped — otherwise a predictor could buy accuracy by abstaining. Multiple comparisons use BH-FDR at $q = 0.10$, and we label exploratory versus confirmatory findings accordingly. Our Wilcoxon implementation declines to compute a p-value below $n = 6$ rather than reporting an unreliable one.

C.6 Choice of outcome variable

§4.4 adopts recall over the gold file set. The choice was made by screening six candidate outcomes on intervention sensitivity, measured as the paired effect divided by its standard deviation: recall attains 2.505, against 2.165 for F_1 and 1.643 for binary success. The second criterion is headroom: 98.6% of instances can move under recall against only 73.2% under the binary outcome, whose failures cannot fall further.

The mechanism behind those numbers is worth stating, because it is the reason a formally adequate outcome can still be empirically useless. With Y binary at the task level, single-target instances are solved almost always, so $\Delta_t \equiv 0$ and attribution says nothing. Refining to the per-file indicator Y_f localizes the question to which step decided whether f was found, but degenerates whenever f is unreachable under every prefix. Recall retains signal from the remaining targets when one is hopeless, which is what reduces the degenerate rate to the 22.9% reported in §5.6. Theorem 1 requires only boundedness, so this substitution changes no formal result (§B.6).

D Additional Results

D.1 Per-repository localization

The high aggregate success rate is itself a constraint on what the corpus can support. It is why the binary outcome has almost no headroom (§C.6), why attribution is restricted to the multi-target

Object model	cov.	top-1	top-3	top-10	NLL	ECE	$\exp(H)$
Routed by predicted tool	0.619	0.150	0.260	0.484	2.76	0.0123	12.8
Factored, shared object model	0.645	0.109	0.241	0.432	3.41	0.0139	39.9
Paired difference	—	+0.041	+0.018	+0.052	—	—	—
95% CI	—	[+.034, +.047]	[+.009, +.026]	[+.047, +.058]	—	—	—

Table 7: Predicting the *complete* next action $(\mathrm{tool}, \mathrm{obj})$ over 7616 decision points from 488 episodes. Both models share the same tool component and the same denominator: every decision point counts, and points whose realized action falls outside the candidate set are scored as misses rather than dropped. top-k are closed-form expectations over random tie-breaking. NLL is over covered points only, so it measures sharpness rather than coverage, and $\exp(H)$ is the median effective candidate count. Routing constructs the candidate set on the predicted next tool—reads take paths, searches take free-text queries, handler tools take no object—which is what the dependence $I = 0.159$ nats ($p = 5 \times 10^{-4}$) between tool and object type requires.

subset, and why a constant predictor is nearly calibrated — which in turn is why §D.8 reports selective-risk quantities alongside ECE rather than ECE alone.

D.2 Additional measurements behind the action-level result

§5.3 reports the headline predictor comparison. Four supporting measurements are recorded here.

The provenance set, not the repository, is the action space. Agents mention 50.5 distinct files per episode and actually visit 6.5, a conversion rate of 18.0%. This is the mechanism behind Table 1: the candidate set an agent draws from is two orders of magnitude smaller than the repository and is constructed by the agent's own earlier outputs. It also explains why the strengthened graph predictor carries a median effective candidate count in the thousands while the provenance predictor carries 3.46 — the graph ranks the repository, whereas provenance ranks what has been seen.

Using the full history rather than the last output is not free. Reading the entire observation history rather than only the most recent output adds $+0.011$ (CI $[+0.005, +0.019]$) at top-3 while inflating effective candidates from 3.46 to 18.8. The last-output variant is therefore the one we report as headline, since it is both sharper and cheaper; the comparison is what rules out the reading that provenance works merely by accumulating a large candidate pool.

Sample-size expansion from dropping the code graph. Provenance requires only string normalization, because agent commands and shell outputs draw paths from the same vocabulary. Dropping the code graph therefore raises the evaluable sample from 509 decision points to 7616 points over 488 episodes, out of 491 episodes usable without a graph. We verified that this expansion does not shift conclusions: on the overlapping graph-built subset the headline metric differs by under 20%, and the residual gap is explained by object-type composition rather than by normalization. The strengthened graph predictor's top-3 on first-visit steps is 0.058 (CI [0.039, 0.078]).

Tool and object type are dependent, and the routed model stays calibrated. The mutual information between tool identity and object type is 0.159 nats against a permutation null of 0.006 ($p = 5 \times 10^{-4}$), which is what makes the factored $P(\mathrm{tool})P(\mathrm{obj})$ model misspecified rather than merely suboptimal. Routing does not buy accuracy at the cost of calibration: the routed model retains ECE $= 0.012$ (Table 7).

D.3 Additional measurements behind the graph and reuse results

Raw rates behind the permutation lifts. The one-hop lift of 0.94 reported in §5.4 is the ratio of an observed adjacent-step one-hop rate of 0.284 to a within-episode permutation expectation of 0.301, with immediate self-repeats removed. We report the raw pair because the lift alone can be read as evidence that adjacency is rare, which it is not: roughly 28% of consecutive steps *are* one hop apart, and the point is that a null which preserves the visited multiset predicts the same rate. The two panels of Figure 7 are therefore consistent rather than contradictory: inside a region of density 0.269 an arbitrary pair of files is already likely to be one hop apart, and the permutation null removes exactly that region effect.

Leakage control in the gold-ranking measurement. 65% of gold files have already been visited by the time the ranking is computed, so leaving them in the candidate set would restate the agent's own history rather than predict anything. The leakage-controlled figures of §5.4 are computed on unvisited gold files only, which is why they rest on $n = 39$ (unseen) and $n = 54$ (prefix-seeded) instances — small samples that we prefer to a larger but contaminated one.

Sample for the reuse analysis. The reuse statistics of §5.5 are computed over 487 episodes. The permutation null for the reuse *rate* has standard deviation zero in our data, which is the formal sense in which that rate carries no information: the number of first occurrences among positions $2..n$ is $\#\text{distinct} - 1$, so the reuse rate is a function of the visited multiset and is invariant to order.

D.4 Reuse analysis in full

§5.5 reports the headline reuse contrasts; the full analysis is recorded here, since only part of it carries information.

Practitioners report that agents reuse what they have already used. We subject that claim to the same within-episode permutation null as §5.4, separating the rate at which reuse occurs from the temporal pattern in which it occurs, because only the second turns out to carry information. The result is also what §5.4's distance-zero excess measures, which is why it is reported rather than dropped.

The reuse *rate* is a permutation invariant and carries no information. The number of first occurrences among positions $2..n$ is exactly $\#\text{distinct} - 1$, so $P(x_{t+1} \in \{x_1..x_t\}) = (n - \#\text{distinct})/(n-1)$ is fixed by the multiset and invariant to order; the permutation null has standard deviation exactly zero in our data. The observed tool and object reuse rates (79.1% and 41.6% over 487 episodes) are therefore arithmetic consequences of trajectory length and vocabulary size, and are not evidence of a tendency. We report them only to retire them.

What is real is the temporal structure of reuse. Under within-episode permutation, objects are repeated in immediately consecutive steps $1.83\times$ more often than chance ($p \sim 10^{-141}$), and successive accesses to the same object are 0.90 as far apart as chance, i.e. temporally clustered. Among revisit events, the target's prior visit count exceeds the closed-form uniform expectation by $1.70\times$ (CI $[1.63, 1.76]$) and its recency rank by $1.74\times$ (CI $[1.67, 1.81]$).

The tool half of the folk claim is false as stated. Agents do *not* preferentially repeat the same tool ($1.02\times$, $p = 0.51$, not significant). What they have is a fixed transition grammar between tools (mutual information $1.13\times$ chance, $p \sim 10^{-22}$). The high predictability of the tool channel is therefore a property of transitions, not of self-repetition.

Taken together, the temporal structure of reuse is real while the reuse rate is not, and the two channels behave differently enough that a single aggregate claim conceals both.

D.5 The query channel and a measured parser defect

§5.3 reports that steps taking a *novel* free-text search query are predicted at top-3 $= 0.029$, and treats that as the binding constraint on next-action prediction. The number deserves scrutiny, because our command parser recovers the search pattern via `shlex`, which fails on 26.6% of command segments — an unbalanced quote such as `grep "a\b"` — and then falls back to splitting on whitespace, retaining the quote character and truncating. 75.3% of query objects are affected, and 249 of them collapse onto the single fragment `q:"def`. We therefore re-extracted the patterns with a parser that does not depend on `shlex` succeeding, and re-ran the channel over 1661 query steps.

The repair moves the number *down*, not up: those frequent fragments were a spurious repetition mode, and removing them costs -0.036 in top-3 (CI $[-0.049, -0.022]$), taking the same model from 0.067 to 0.031. Improving the model then recovers ground. Weighting history by frequency rather than recency helps marginally, and adding identifiers mined from the issue text helps substantially (+0.041, CI $[+0.029, +0.054]$, with coverage rising from 0.060 to 0.181), while adding symbol names harvested from prior tool *outputs* actively hurts (-0.007). Our best configuration reaches top-3 $= 0.076$ at coverage 0.181, and the net change from the original pipeline is not significant ($+0.002$, CI $[-0.016, +0.019]$).

Three consequences follow. The conclusion that queries bound next-action prediction *survives*, and is now the outcome of a deliberate attempt rather than of a weak baseline. The defect biases in the conservative direction for that claim — it made queries look more predictable than they are — so it does not threaten it. And there is a positive mechanism worth recording, which contrasts instructively with the path channel of §5.3: an agent's *paths* come from what it just read in an output, whereas its *query terms* come from the issue text, and importing symbol names from outputs into the query model only dilutes it. We report the defect rather than silently fixing it because its direction is what licenses the claim to stand.

D.6 A BINARY CRITERION THAT MANUFACTURED AN EXCEPTION

An earlier pass over the data of §5.6 thresholded the per-point test at $p < 0.05$ and found that high-entropy steps carried a nominally higher rate of significant effects, which would have contradicted that section's null. It does not replicate. The stratified difference is $+0.056$ with CI [-0.008, +0.135], and replacing the binary indicator with either $|\Delta_t|$ ($\rho = +0.002$) or $-\log_{10} p$ leaves no relationship at all. The diagnostic is the threshold sweep: as α is relaxed and the event count grows from 8 to 24, the ratio decays monotonically from 3.36 to 1.71 — the signature of a small-sample artefact rather than of an effect. We record it because dichotomising a continuous quantity produces clean-looking strata at small n, and the continuous version plus a threshold sweep is what distinguishes the two.

D.7 UNCERTAINTY COMPONENTS AND THEIR ESTIMATORS

The predictive uncertainty of the left endpoint $v_0 = \mathbb{E}[Y \mid q, G]$ introduced in §4.4 decomposes conceptually as

$$\mathbb{H}[Y \mid q, G] = U_{\text{task}} + U_{\text{policy}} + U_{\text{model}}, \tag{13}$$

with U_{task} irreducible task ambiguity, U_{policy} variability induced by stochastic execution, and U_{model} epistemic uncertainty in the forecasting model. Table 8 gives an estimator for each. The three answer different questions, so conflating them discards decision-relevant content.

Component	Type	Estimated by	Implied action
U_{task}	aleatoric, irreducible	residual; graph ambiguity covariates	ask the user to disambiguate
U_{policy}	aleatoric, reducible	between-rollout variance at fixed q	resample and aggregate
U_{model}	epistemic	posterior variance through the link	abstain, escalate, collect data

Table 8: The three components of equation 13. Each answers a different question, so conflating them discards the decision-relevant content: a system that reports only total uncertainty cannot distinguish "the request is ambiguous, ask the user" from "the model is out of its depth, escalate".

Status on our data. We state this plainly, because it is weaker than the derivation suggests. Only U_{policy} is estimated directly, at 0.049 from the 23 instances carrying $K = 4$ repeated rollouts. The three-way separation is validated on a *synthetic* generating process with known truth, not on these traces: parameter recovery reaches correlation 0.994, mean absolute probability error is 0.022, and predictive entropy 0.464 splits as $0.308/0.146/0.012$ across the three components, with U_{model} inflating from 0.012 in support to 0.100 out of support, a factor of 8.1. Figure 6 shows the self-test together with the forecasting result it does not rescue. The identifiability of the three terms on real traces is therefore *not* established here, and the falsifiable prediction that additional sampling cannot help where U_{task} dominates is *not* tested, because it presupposes the calibrated v_0 that §D.8 reports as absent. Table 8 is a specification with a synthetic self-test attached, and should be read as such.

D.8 ESTIMATING THE LEFT ENDPOINT v_0

Our corpus has strong group structure by repository and a high base rate, which is a small-data regime. We therefore fit v_0 as a Bayesian logistic model with a Gaussian prior $w \sim \mathcal{N}(0, I)$ (precision $\lambda = 1$), taking the MAP by IRLS and the posterior covariance Σ by Laplace approximation, then propagating parameter uncertainty by the probit approximation to obtain $U_{\text{model}} \approx p(1-p)\, x^\top \Sigma x$. No hyperparameter search was performed. Evaluation is leave-one-repository-out over 12 repositories. The claims at issue concern calibration and identifiability rather than predictive ceiling, so a small interpretable model with a tractable posterior serves them better than a tuned ensemble whose uncertainty is heuristic.

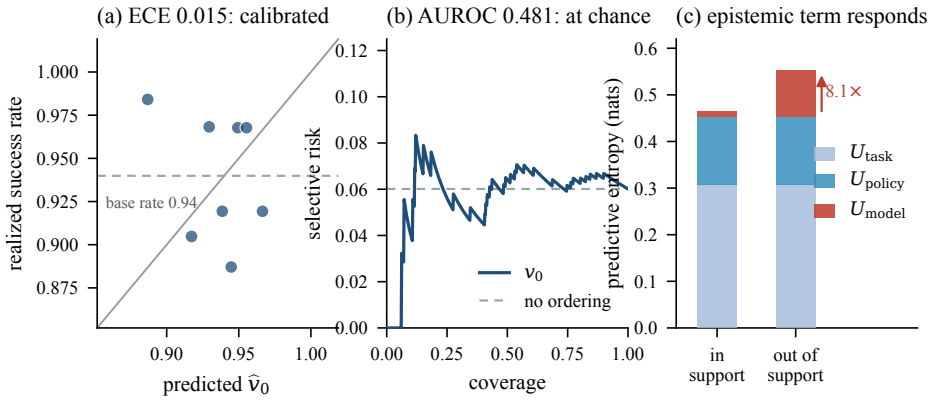

Figure 6: **The uncertainty decomposition, validated synthetically and unvalidated empirically**. The panels report the forecasting behaviour of the left endpoint v_0 on the real corpus alongside the synthetic self-test in which the generating process is known. On the real corpus the risk–coverage curve is essentially flat at the base error rate, the signature of a calibrated but uninformative predictor, which is why we report AURC and AUROC rather than ECE alone. The contrast is the point: the machinery works where the truth is known and does not deliver a usable forecast where it is not.

The measured outcome is negative. Leave-one-repository-out AUROC is 0.481 over 7 covariates, which is at chance against a base rate of 93.99%, with ECE and Brier score as defined in equation 16 reaching 0.015 and 0.058, and area under the risk–coverage curve 0.058. The low ECE is not reassuring: a high base rate makes calibration easy to satisfy by a nearly constant predictor, and the flat risk–coverage curve confirms that the predictor supplies no usable ordering. We therefore report this as a negative result rather than as a calibrated forecaster, and it is the reason the falsifiable prediction attached to equation 13 is untested (§D.7). The predictor may read only q and G, never the trace, since doing so would leak execution-time information.

D.9 Set-valued prediction and exchangeability

Repository	n	Localization success
`django`	231	93.5%
`sympy`	75	94.7%
`sphinx-doc`	44	88.6%
`matplotlib`	34	91.2%
`scikit-learn`	32	100.0%
`astropy`	22	100.0%
`pydata`	22	90.9%
`pytest-dev`	19	100.0%
`pylint-dev`	10	90.0%
`psf`	8	100.0%
`mwaskom`	2	100.0%
`pallets`	1	100.0%
Total	500	93.99%

Table 9: Localization success by repository under the criterion $F^\star \subseteq \hat{F}$. The corpus is dominated by `django`, at 46% of instances, which is why leave-one-repository-out is the protocol for v_0 (§D.8) and why repository-conditional calibration is used for conformal sets (§D.9). Higher is better.

Localization is naturally set-valued, so alongside the scalar v_0 we report conformal sets $\mathcal{C}(q)$ satisfying $P(F^\star \subseteq \mathcal{C}(q)) \geq 1-\alpha$, together with average set size, since a guarantee attained by returning the whole repository is vacuous. Exchangeability requires care rather than assertion in this setting: traces are sequentially dependent, the observations an agent collects are self-selected by its own earlier actions, and repositories are heavily imbalanced (Table 9). We therefore use split conformal with repository-conditional (Mondrian) calibration rather than pooled calibration, report group-conditional coverage gaps rather than marginal coverage alone, and apply covariate-shift weighting

for cross-repository transfer. Marginal coverage that is correct on average while failing badly on the smaller repositories is the failure mode this guards against, and pooled calibration on a corpus that is 46% `django` would conceal it.

D.10 Additional measurements behind the attribution results

§5.7 reports the displacement and enrichment results. The supporting quantities are recorded here, since several of them are what fix the interpretation.

Absolute positions, not just the displacement. The collider contrast moves the verdict from a mean normalized position of 0.214 under the prefix-only condition to 0.751 under the full-trace condition, with a median displacement of +0.667. The absolute positions matter for reading the effect: the prefix-only judge is not merely earlier, it sits in the first quarter of the trajectory, while the full-trace judge sits in the last quarter. For comparison, the intervention-based selector has mean position 0.333. Reporting only the difference would leave open the possibility that both conditions cluster in the same region.

Enrichment interval and the comparable subset. The full-trace judge places 0.900 (CI [0.829, 0.971]) of its top-1 verdicts on gold-provenance steps. The restriction to steps that are directly comparable between the judge and the intervention estimate rests on $n = 23$ episodes, because only 32.9% of judge verdicts fall inside the estimator's window set at all. The two baselines are therefore computed separately over their own selection spaces, and the comparable-subset contrast is reported alongside the unrestricted one rather than instead of it.

Cross-model budget and per-judge enrichment. The replication uses one additional general-purpose model (qwen3.6-plus) and one code-specialised model, at 48 API calls per judge, with the prompt held byte-identical and the trace set fixed. Displacement is positive in all 3 of 3 judges. Per-judge enrichment on gold-provenance steps is 0.826 for the original judge, 0.565 for the second general model and 0.826 for the code-specialised one, against the common-episode baseline of 0.443 (Table 2). We note but do not claim the pattern that the code-specialised model behaves like the original rather than like its own general-purpose sibling: 3 judges cannot separate that explanation from model scale, training data or rubric sensitivity, so we record it as a hypothesis.

Attribution subsample composition. Attribution is computed over 70 episodes drawn from 10 repositories, of which 19 are failing episodes. The mean replay variance across estimation segments is 0.034, and the mean normalized position of the high-entropy steps identified by the positivity probe is 0.416 — close to the middle of the trajectory, which is the sense in which overlap is not concentrated early.

D.11 Reported quantities in full

§4.4 defines positional displacement, which the experiments use. This subsection defines the four statistics specified for completeness and pre-registration but reported nowhere as evidence. Let $\hat{t}_i^M$ be the step selected by method M in episode i and $\mathrm{pos}(t) = t/T_i$ the normalized position; top-1 disagreement and positional displacement are

$$D_1(M, M') = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\big[\hat{t}_i^M \neq \hat{t}_i^{M'}\big], \qquad D_3(M, M') = \frac{1}{N}\sum_{i=1}^{N} \big(\mathrm{pos}(\hat{t}_i^M) - \mathrm{pos}(\hat{t}_i^{M'})\big). \quad (14)$$

Letting $\sigma_i(t)$ be the estimation segment of episode i containing step t and $\mathcal{N} = \{i : \exists\, t \in \mathcal{P}_i,\ \widehat{\Delta}_{i,t} \neq 0\}$ the non-degenerate episodes ($|\mathcal{N}| = 54$ here, with 54.7% of estimation points carrying $\widehat{\Delta}$ exactly zero), the decisive-step recovery rate, the null-blame rate and the mean signed contribution of the blamed step are

$$\begin{aligned} R(M) &= \frac{1}{|\mathcal{N}|}\sum_{i\in\mathcal{N}} \mathbb{1}\Big[\sigma_i(\hat{t}_i^M) = \arg\min_t \widehat{\Delta}_{i,t}\Big], \\ Z(M) &= \frac{1}{|\mathcal{N}|}\sum_{i\in\mathcal{N}} \mathbb{1}\Big[\widehat{\Delta}_{i,\sigma_i(\hat{t}_i^M)} = 0\Big], \\ \bar{\Delta}(M) &= \frac{1}{|\mathcal{N}|}\sum_{i\in\mathcal{N}} \widehat{\Delta}_{i,\sigma_i(\hat{t}_i^M)}. \end{aligned} \quad (15)$$

A method would score well by making R large, Z small and $\bar{\Delta}$ negative, that is by blaming steps that measurably made the outcome worse, and the informative reference would be a uniform-random selector at its exact expectation $\mathbb{E}[R] = |\mathcal{N}|^{-1} \sum_i |\mathcal{P}_i|^{-1}$ rather than a sampled draw.

Pre-execution forecasting is evaluated for both discrimination and calibration. With $\widehat{v}_{0,i}$ the predicted success probability for instance i and $\mathcal{B}_b$ the b-th of B confidence bins,

$$\mathrm{ECE} = \sum_{b=1}^{B} \frac{|\mathcal{B}_b|}{n} \left|\overline{y}_b - \overline{p}_b\right|, \qquad \mathrm{Brier} = \frac{1}{n} \sum_{i=1}^{n} \left(\widehat{v}_{0,i} - Y_i\right)^2. \tag{16}$$

Because the corpus has a high success base rate, §D.8 reports ranking- and selective-risk quantities alongside these, so that a nearly constant calibrated predictor is not mistaken for an informative one.

D.12 Why D_1 is retired

D_1 appears in equation 14 for completeness and because we pre-registered it, but it is not a valid test and we report it nowhere as evidence. A disagreement rate is informative only against a reference whose selection is reliable. Ours is not: §5.6 shows the identified top-1 does not survive FDR correction (0 of 190 effects), so it behaves as a draw over the $|\mathcal{P}_i| \approx 5$ estimation points while a judge selects among $T_i \approx 20$ steps. Two selectors drawing from spaces of such different granularity disagree with probability near one *under the null*, which is what we observe (97.1%). The criterion was ill-posed rather than conservative, and we flag this rather than quietly dropping the metric, since reporting D_1 as a positive finding is a mistake another group could repeat.

The same defect propagates to the three statistics of equation 15. R, Z and $\bar{\Delta}$ are all computed *against* $\widehat{\Delta}$, so each inherits its noise, and none can adjudicate between "the judge applies a different but defensible criterion" and "the judge extracts no causal signal". We retain them in §4.4 as definitions, note that only 54 episodes are non-degenerate and 54.7% of estimation points have $\widehat{\Delta}$ zero, and report them nowhere as evidence. The comparisons that do adjudicate need no causal reference: the provenance-class enrichment and the information-set isolation contrast, both in §5.7.

D.13 Comparison with annotation-based process evaluation

Process-level evaluations of coding agents (He et al., 2026) and SCAE target different estimands, and the difference determines what each can license.

Estimands. That line of work estimates $p_j = P(y_j = 1 \mid x_j, c)$, the probability that a human annotator would mark defect j given trajectory evidence x_j and context c. This is a measurement-theoretic quantity: it is about agreement between a detector and a rater, and it is identified from logged trajectories without any causal assumption, since nothing is intervened upon. SCAE estimates $\mathbb{E}[Y \mid \tau_{1:t-1}, do(A_t{=}a')]$, the effect on the terminal outcome of substituting one step. This requires the logging policy's randomization to be preserved (Proposition 1) and is *not* identified under context compaction (§B.5).

Calibration targets. The two therefore calibrate against different things. There, calibration means detector-versus-annotator agreement, and a well-calibrated detector is one whose confidence matches the rate at which raters agree. Here, calibration means predicted-versus-realized outcome, and a well-calibrated v_0 is one whose predicted success probability matches the realized success frequency (§D.8). Neither target substitutes for the other: a detector perfectly aligned with annotators tells us nothing about whether the step it flags changed the outcome, and a perfectly calibrated outcome predictor tells us nothing about whether humans would call any particular step defective.

Why this is not a criticism. That work does not claim causal identification, so the comparison is a delineation of scope rather than an objection. It is, however, the reason we do not adopt annotation agreement as a validation target for attribution: a step can be unanimously judged defective and have zero causal contribution, and a step can look unremarkable to every annotator and be decisive, as §B.7 illustrates.

E REPRODUCIBILITY, PRE-REGISTRATION, AND FALSIFIABILITY

E.1 REPRODUCTION COMMANDS AND SENSITIVITY ANALYSES

Commands. Every reported number and every data figure is regenerated by a CPU-only script that fixes its random seed and writes its output to JSON. The uncertainty self-test of §D.7 is reproduced by `python3 -m pilot.analyze.uq_selftest`; the entropy summary of §C.4 by `python3 -m pilot.analyze.entropy_scan`; the localization and forecasting tables by `python3 -m pilot.analyze.full_report`; and the data figures by `python3 paper/figs/make_figures_rq.py` and `python3 paper/figs/make_figures.py`. The figure scripts read only the recorded rollout outputs, so plotted and tabulated values cannot drift apart. A further script, `python3 paper/analysis/verify_macros.py`, checks every numeric macro in the manuscript against the analysis output that produced it, and `python3 paper/analysis/check_tex.py` verifies that every reference, label, macro, citation and figure path resolves.

Outcome definition. §5.2 reports the primary criterion $F^\star \subseteq \hat{F}$ alongside exact set equality. The gap between 93.99% and 15.23% is accounted for entirely by the agent naming test files that gold patches exclude: a filtered variant that discards test paths before comparison coincides with the primary criterion to within rounding. We therefore treat exact equality as a diagnostic of the agent's naming behaviour rather than as a metric of localization ability.

Effect of the leakage reset. Every number is measured after a hard reset of the repository snapshot to its base commit. This is a silent confound rather than a detail: the same pipeline run against the distributed snapshots as shipped would measure an agent that can read the answer out of its own working tree, and would report inflated localization accuracy with no visible symptom.

Measurement error in regression diagnostics. Estimated contributions carry Monte Carlo error, which attenuates regression coefficients toward zero. For the hypothesis that data-flow reachability fails to predict causal effect (§B.7), this bias is favourable to us, so we state its direction explicitly. A correction such as SIMEX, or larger m, is required before that null is asserted rather than merely observed.

Discrete estimation points. Because v_t is estimated at equidistant points rather than at every step, reported contributions are segment effects and their sum does not satisfy equation 6 exactly. We therefore present no additivity check as evidence; a genuine test requires contiguous windows. Equidistant rather than contiguous-tail points was a deliberate choice, because a window concentrated late in the trace would bias the position comparison on which §5.7 depends.

E.2 PRE-REGISTRATION RECORD

The directions, intervals and falsification thresholds in Table 10 were fixed before any attribution experiment was run. We report the record in full, including the four entries that did not hold and the one criterion we retire rather than bank.

What this record does not cover. The decisive-step recovery rate R, the mean blamed contribution $\bar{\Delta}$ and the null-blame rate Z of equation 15 are absent from the table because they were not pre-registered. We introduced them after observing the displacement, to distinguish "the judge applies a different but defensible criterion" from "the judge extracts no causal signal". We now withdraw that use of them, for the reason given in §D.12. Both comparisons that do adjudicate — provenance-class enrichment and the information-set isolation contrast — were reachable from the same recorded rollouts, so the correction cost no additional agent execution.

The defect the record caught. Our initial design truncated the prefix-only judge just before the step selected by the causal estimate. Because a judge cannot select a step it cannot see, this caps the prefix-only condition's reachable positions below the comparison point and would have produced positive displacement even if conditioning on descendants had no effect at all. Writing the falsification criterion forced us to ask what would happen under the null, which surfaced the defect. The corrected design uses a truncation fraction fixed in advance and independent of any estimate (§C.2).

Quantity	Pre-registered	Outcome
D_1 top-1 disagreement	$[40\%, 75\%]$, point 55%; falsified if $< 20\%$	**criterion retired, not passed**: observed 97.1% $(68/70)$, CI $[92.9\%, 100\%]$. We initially recorded this as "direction confirmed"; that reading is withdrawn, because the null itself sits near 97.1% once the reference is known to be noise (§D.12)
D_3 displacement	$[+0.25, +0.40]$; falsified if ≤ 0	**above interval**: +0.418 over $n = 68$, $z = 6.07$, $p < 0.001$
Collider isolation	$[+0.15, +0.30]$; sign is the substantive claim	**above interval**: +0.537 over $n = 60$, $z = 6.70$, $p < 0.001$. Sign confirmed; magnitude implies descendant conditioning accounts for most of the displacement
Wilcoxon p	expected to be uncomputable at small n	superseded: the full attribution sweep makes it computable, and both signed-rank tests reject at $p < 0.001$
Degenerate $\Delta \equiv 0$ rate	30–40%	**better than predicted**: 22.9% $(16/70)$. An early two-episode reading suggested 50%; the full sweep does not support it
U_{policy} estimable	requires $K \geq 2$; not available from the main sweep	met on the $K{=}4$ subset: $U_{\text{policy}} = 0.049$ over 23 instances
Entropy vs. position	monotone decay, $\rho \leq -0.2$ ("early-high, late-low")	**not supported**: $\rho = +0.072$ over 10 points; reported as non-monotone with episode-dependent inflection
v_0 LORO AUROC	≥ 0.60 for the forecasting endpoint to be useful	**not met**: 0.481, at chance; reported as a negative result (§D.8)
U_{model} inflation	$> 2\times$ out of training support	met: $0.012 \rightarrow 0.100$, a factor of 8.1
Replay consistency	≥ 0.99 byte-identical	met: 1.00 over 100 actions, zero mismatches

Table 10: The pre-registration record and its outcomes. Three directional predictions held at magnitudes above the intervals we fixed in advance, which we report as the intervals having been too conservative rather than as successes. Four entries did not hold as stated: the monotone entropy decay is contradicted, the forecasting endpoint failed its usefulness threshold, the degenerate-outcome rate came in below the predicted range, and the D_1 criterion was ill-posed.

E.3 Scope and limitations in full

§5.8 names the two limitations that bind hardest. All five are as follows. *Task scope*: the analysis covers file localization rather than the full space of coding-agent behaviour, because that setting supplies the replayability, prefix reconstruction and verifiable outcomes the method requires; it is the smallest realistic environment in which the identification assumptions can be checked, but process structure in code editing, patch synthesis or open-ended tool use may differ. *A weak step reference*: the intervention-based reference is formally identified but empirically weak, with 0 of 190 effects surviving BH-FDR, so we cannot certify which step mattered in any individual episode, and the strongest attribution result in the paper is a mechanism claim about how a judge's verdict moves rather than a correctness claim about the step it selects. *Budget and granularity*: contributions are segment-level approximations, and denser replay improves resolution only up to the structure of the underlying uncertainty, which §5.6 places at the task level, so scaling replay count alone is unlikely to remove the limitation. *Partially testable assumptions*: replayability and compaction are checked in the harness and reported in §5.2, but exogeneity of decoding noise is a structural property of the serving policy and cannot be verified from traces, and the intervention branch rests on the transferability of substituted actions (Assumption 4). *Sample sizes and coverage*: displacement replicates across 3 judges with no detectable heterogeneity, but that sample is modest and the heterogeneity test necessarily low-powered; only U_{policy} is estimated directly, since the decomposition of equation 13 is validated on a synthetic generating process rather than on these traces; and only 46.2% of agent objects resolve cleanly to dependency-graph nodes, which bounds the graph half of §5.3–§5.4 without affecting the provenance half.

E.4 What would falsify the framework

§E.3 states the limitations that bound how our results should be read. This subsection states the sharper claim: observations that would undercut the central argument rather than merely bound it.

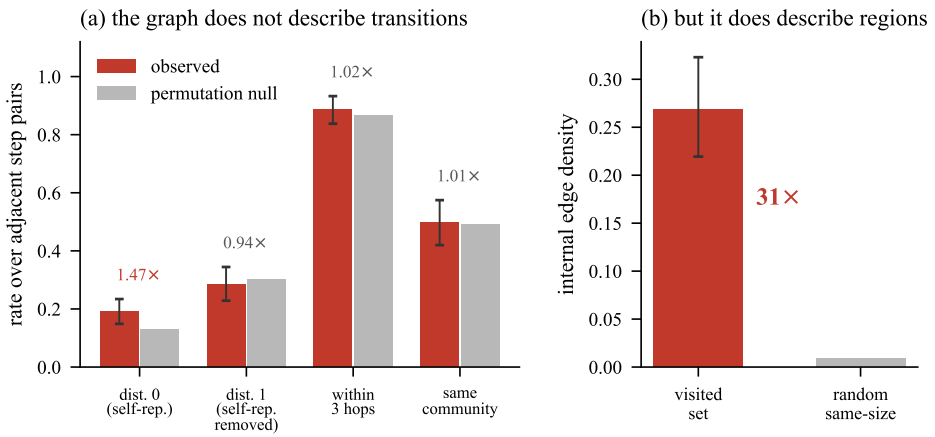

Figure 7: **What the dependency graph does and does not describe.** (a) Graph distance between consecutive steps against a within-episode permutation null, which preserves the visited-file multiset and destroys only order; lifts are observed over null rates, and the distance-zero bar is immediate self-repetition. (b) Internal edge density of the visited set against random same-size file sets.

Three would undercut the framework itself. If prefix replay failed to reproduce agent behaviour — that is, if forking from a reconstructed prefix systematically diverged from the original continuation — then Assumption 1 would be false in practice and the identification result would not apply to real harnesses. If the isolation contrast of §5.7 produced no displacement at scale, the collider-bias explanation of judge behaviour would be wrong even though the structural argument stands; this one has now been run, and the displacement is +0.537 with a sign test at $p = 1.1 \times 10^{-16}$. And if action entropy were zero at essentially all decision points, the observational branch of equation 8 would never apply and the framework would reduce to intervention-based ablation with respect to a perturbed policy.

Three further falsifiers attach to the specific empirical claims. If a code-graph predictor in *any* form matched execution provenance on first-visit steps, §5.3–§5.4 would collapse; we strengthened the graph predictor deliberately, seeding a multi-source proximity ranking from the whole visited set, and it still loses, but a better graph formulation is the obvious line of attack. If the residualised instance-level ICC of §5.6 fell back to its permutation null, then replay variance would carry no detectable non-mechanical structure at $m = 8$, and our null result about structural entropy would revert to a statement about power rather than about the world — which is why we report that ICC before, not after, the null it licenses. And if the judge's placement on gold-provenance steps regressed to its 0.413 baseline under a different judge or rubric, the characterisation of what such judges measure would not generalise beyond the ones we tested.

F Additional Figures

Figure 7 plots the two graph contrasts of §5.4 and Figure 8 the variance-structure probes of §5.6; the numbers in both are reported in the body.

G Additional Tables

Table 7 gives the joint next-action comparison of §5.3, Table 11 the trust stratification, and Table 12 the zero-LLM rule-reproduction suite of §5.7.

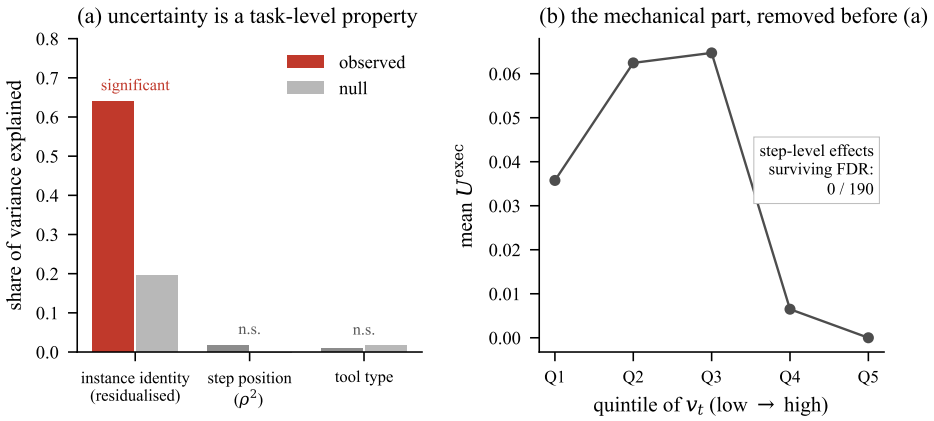

Figure 8: **The level at which execution uncertainty resides.** (a) Probes for structure in the replay variance U^{exec}, each expressed as a share of variance explained, computed on residuals after binning on v_t. (b) The mechanical mean–variance relation that binning removes: U^{exec} is constrained by the mean and collapses to zero in the top quintiles of v_t.

Stratum (observable before predicting)	n	**top-3**	**ECE**
Current step is a search	228	0.469	0.0125
Current step is a read	206	0.160	0.0247
Current step lists a directory	35	0.057	0.0278
Predicted $\exp(H) \leq 5$	154	0.351	0.0762
Predicted $\exp(H) \in (5, 15]$	198	0.409	0.0168
Predicted $\exp(H) > 15$	157	0.159	0.0157
Last output lists 1–3 paths	263	0.357	0.0411
Last output lists 4–10 paths	151	0.298	0.0050
Last output lists ≥ 11 paths	95	0.221	0.0125

Table 11: When the next-action distribution can be trusted. Only strata observable *before* the prediction are listed, because a stratum defined by the realized answer is a conditional accuracy and not a rule: conditioning on coverage gives top-3 $= 0.611$ at ECE $= 0.0103$, which is not actionable and which we therefore exclude.

Zero-LLM structural rule	**agree**	**chance**	**diff**	**sig.**
First step revealing a gold file	0.043	0.053	-0.010	—
First step operating on a gold file	0.100	0.053	+0.047	—
Last gold-provenance step	0.243	0.053	+0.190	✓
Last step of the trace (position only)	0.171	0.053	+0.118	✓
Step revealing the most gold files	0.043	0.053	-0.010	—

Table 12: Can a rule with no language model reproduce the full-trace judge's choice? Agreement is exact same-step agreement over 70 episodes, and chance is the closed-form expectation $1/T_i$ for a rule selecting uniformly among steps. With 8.7 gold-provenance steps per episode the *within-class* chance rate is 0.139, against which the best rule is only $1.7\times$ better; we report that residual rather than presenting the judge as a lookup table.