How Reliably Does Replay Reproduce a Tool-Using Agent's Harmful Outcome?

Anonymous authors
Paper under double-blind review

Abstract

Counterfactual replay is a common instrument in replay-based attribution for tying a tool-using language agent's harmful action to a span of its context: neutralize a candidate span, re-execute, and read the outcome. Whether that verdict is a value or a draw, and how concentrated the draw is, has not been measured. We report a census. On the AgentDojo workspace suite with a stock prompt-injection attack and gpt-4o-mini, all 120 configurations are executed 20 times in each of two decoding conditions, 4,800 executions with no failed runs. The benchmark-level attack success rate is 0.563 under greedy decoding and 0.566 at temperature 1, against a round-to-round standard deviation of 0.019 and 0.026 – steadiness that is a property of averaging rather than evidence that an individual outcome repeats. In each condition a single replay of an observed harmful execution returns the opposite verdict 0.087 of the time (95% CI [0.053, 0.124]) under greedy decoding and 0.141 ([0.091, 0.189]) at temperature 1, and 0.306 ([0.183, 0.389]) and 0.404 ([0.292, 0.534]) of harmful configurations have an observed harm frequency strictly inside (0.1, 0.9) over 20 identical repeats. An aggregate error bar therefore does not bound the per-incident quantity, and on this census the two move independently. Trajectory comparison localizes the first divergence to a model output in 91 of 91 divergent pairs with byte-identical payloads, the population is far more polarized than a common-rate model allows, and that excess appears again on a second AgentDojo task suite, on a benchmark with no attacker, and on an independently implemented attacked benchmark whose checker, attack and decoding condition are none of ours. Executing the counterfactual side as well, over 6,120 preregistered executions, four interventions on the injected payload leave no harmful outcome on any contested configuration, while deleting count-matched spans that never touch it moves a configuration's harm probability almost as far – a noise-corrected root-mean-square of 0.44 [0.32, 0.54] against 0.57 [0.48, 0.65]. Sensitivity of that order recurs on a second task suite whose nine eligible configurations, from six user tasks, were selected independently: the ratio there is 1.10 [0.79, 1.75], passing a point rule fixed in advance at [0.20, 1.40], with the direction reversed – harm rises on seven of nine, each to 20 of 20, where it split eleven to fourteen here. Unsigned change alone therefore does not identify the injected source. This sensitivity is distinct from failure of signed candidate ranking and persists regardless of the replay budget used to estimate it.

1 Introduction

Untrusted content can steer a tool-using language agent into an action its user never requested (Greshake et al., 2023). Counterfactual replay diagnoses such incidents by neutralizing a candidate span and re-executing the agent (Kang et al., 2026; Bonagiri et al., 2026; Mukhopadhyay et al., 2026). Related operations underlie single-call context attribution (Cohen-Wang et al., 2024; Wang et al., 2025) and delta debugging (Zeller & Hildebrandt, 2002). Interpreting one flipped verdict as identifying the injected source requires checking both replay repeatability and sensitivity to edits outside that source. Our census measures the first; a preregistered follow-up on 26 configurations measures the second. Under stochastic replay, a flip compares Bernoulli draws rather than fixed outcomes. Moreover, count-matched deletion outside the payload moves harm probability nearly as

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.

far as payload deletion in unsigned magnitude. These measurements test the single-flip interpretation, not the accuracy of multi-sample or graph-based attribution methods.

The literature has not settled that question. Replay-based attribution states its identification results for a deterministic outcome function and names the stochastic case as future work (Wang et al., 2025; Kang et al., 2026). Others take the variation as given and absorb it: AttriGuard, at temperature zero with seeds fixed, still observes it, attributes it to providers routing requests to different backend configurations, and characterizes it by repeating each experiment five times (He et al., 2026). That characterizes an aggregate, how far a rate computed over a whole benchmark moves between rounds, while attribution runs on one incident and concludes about that incident. The two are free to move independently, and none of the six works closest to this measurement reports the per-incident one (Wang et al., 2025; Cohen-Wang et al., 2024; He et al., 2026; Kang et al., 2026; Ge et al., 2025; Wang et al., 2026b).

We measure it. On the AgentDojo workspace suite (Debenedetti et al., 2024) with gpt-4o-mini-2024-07-18 and the stock `important_instructions` attack, all 120 configurations of the suite are executed 20 times in each of two decoding conditions, 4,800 executions with no failed runs (Section 3.2). A configuration is one user task paired with one injection task under a fixed attack and model, measured separately at each decoding temperature; it is harmful when at least one of its executions returns the benchmark's security verdict: 85 are harmful under greedy decoding and 94 at temperature 1. A single replay of an observed harmful execution flips the verdict 0.087 of the time under greedy decoding, the condition most favorable to a deterministic-replay assumption, and 0.141 at temperature 1 (95% CI [0.053, 0.124] and [0.091, 0.189]). Classified by configuration rather than replay, 0.306 and 0.404 of harmful configurations have an observed harm frequency strictly inside a band (0.1, 0.9) fixed in advance, and 0.400 and 0.606 return both labels over their 20 repeats.

The same executions give the quantity an evaluation reports, and it does not move: attack success is 0.563 and 0.566 on identical populations, a difference of 0.003 against round-to-round standard deviations of 0.019 and 0.026. Individual flips offset in the mean, so that steadiness is arithmetic rather than evidence about any one incident, and the scale-free version of the same point is a beta-binomial intraclass correlation over configurations already observed harmful of 0.623 and 0.530. We read no effect of decoding temperature into the pair – the intervals overlap and each arm fixes its condition instead of randomizing it – only that the aggregate does not track the per-incident quantity. Nor is the irreproducible group what sampling alone would produce: a raw fraction inside a band depends on the base rate, and at each arm's pooled rate the expectation is 0.808 and 0.945 against the observed 0.306 and 0.404, with the homogeneous null rejected in both, $\chi^2/(n-1) = 12.03$ and 10.42 at $p < 10^{-10}$, and rejected again inside each of six variants of the injected text spanning attack success from 0.917 to 0.383.

The census rests on one model family, one benchmark suite and one attack family, and further arms mark its edges. The same dispersion signature appears on tau2-bench, which has no attacker and judges task completion against a database, and on InjecAgent, a separate codebase whose tools, agent loop, success criterion and attack are none of ours and which fixes temperature 0 itself. A contemporary model, gpt-5-mini, is not measurable at all: the payload-matched attack compromises gpt-4o-mini in 41 of 48 executions and gpt-5-mini in none, and a quantity defined over harmful incidents is not estimable without incidents.

Four things follow. We report a census of replay reproducibility on an executable agent benchmark, keeping the replay-level flip probability and the configuration-level band occupancy distinct. We measure the aggregate error bar an evaluation reports and the per-incident quantity attribution consumes on the same executions, and find the first steady in both conditions and the second large in both. We show what replay variability does to three attribution implementations, which over 26 contested incidents agree with themselves 0.579, 0.240 and 0.182 of the time once each incident's own chance floor is removed. And we execute the neutralization rather than modelling it, over 33 configurations and 6,120 preregistered executions: removing the injected payload leaves no harmful outcome, while a count-matched deletion outside it reaches 0.77 [0.56, 0.98] of that magnitude, rising on eleven configurations and falling on fourteen, so unsigned change alone does not identify the injected source. The protocol is reported with the failure modes it must survive, the sharpest being a three-repeat screen that misses 10 and 14 of the harmful configurations, concentrated on the low-probability ones.

2 RELATED WORK

We organize this section by what each line of work assumes, measures, or exploits about the reproducibility of an agent execution. TracLLM (Wang et al., 2025) attributes an LLM output to spans of its context with a cost-aware group-splitting search, and states its identification guarantee over the deterministic outcome function of Section 1, with the Shapley value function an indicator on the equality $f(I \oplus U) = O$. If f returns a sample rather than a value, that equality becomes an event, not a fixed predicate, and the guarantee does not carry over unchanged. Kang et al. (2026) predicts the verdicts of a counterfactual replay oracle without paying for the replays, calibrates against a *deterministic* oracle, and names the stochastic extension as future work. Our measurement answers that question without challenging those results.

He et al. (2026) verifies a proposed tool call's necessity by re-executing the agent under attenuated observations, and is explicit that run-to-run variation exists: at temperature zero, top-k one and seeds fixed it still observes it, attributes it to provider-side routing across backend configurations, and absorbs it with a fuzzy survival criterion characterized by repeating whole experiments five times. We take no credit for noticing the phenomenon; our claim concerns the statistic, since an aggregate error bar and the reproducibility of an individual incident are different quantities and Section 4.2 measures how far apart they are. Spectrum-based localization (Ge et al., 2025) takes the opposite stance, scoring actions by how their presence covaries with failure across many re-executions, so variation is what makes the spectrum informative. The field holds both positions at once, which is why we do not write that prior work assumes determinism.

ContextCite (Cohen-Wang et al., 2024) attributes a generated response to context sources by ablating them and fitting a sparse surrogate to the change in conditional probability. Its estimand is a probability under one forward pass, not a discrete outcome of a multi-step execution, so the concern here does not apply to it as published; it applies once the method is lifted to an agent, where the available quantity is a checker verdict, and Appendix A.4 describes the lift we implement. A growing line of work localizes the responsible step or agent in a failed trajectory, by prompting, causal scaffolding, hierarchical causal graphs, or a trained tracer (Zhang et al., 2025b; West et al., 2025; Wang et al., 2026b; Zhang et al., 2025a; Bonagiri et al., 2026; Wang et al., 2026a). We make no attribution-accuracy claim against it: our subject is a property of the oracle these methods query, not a competing localizer.

The assumption is inherited rather than invented. Delta debugging (Zeller & Hildebrandt, 2002) and spectrum-based fault localization were built for programs where re-execution on the same input is reproducible by construction, so natural that the assumption is rarely stated. Agent tracing and observability stacks (Dong et al., 2024; Rombaut et al., 2024; Zheng et al., 2025) record what happened with rising fidelity, which is orthogonal: a complete record of one execution says nothing about whether a second would have produced the same outcome. Across the six works closest to this measurement, execution stochasticity is an unexamined assumption, an aggregate error bar, an unquantified resource, an inapplicable concern, or silence – and none reports how often a replay of an observed harmful incident returns the opposite verdict.

3 SETUP

3.1 BENCHMARK, MODEL, AND OUTCOME LABELS

The primary census runs on the AgentDojo workspace suite at a pinned checkout (commit `089ed468`) (Debenedetti et al., 2024). The suite injects an attacker-chosen string through a named vector and returns two verdicts per execution, both used unmodified: a security check, true exactly when the injection's goal was achieved, and a utility check, true when the user's task was completed. The outcome label therefore reads the environment's own state, no annotation step enters, and ground truth for what was injected holds by construction.

The suite pairs each of 40 user tasks with three injection tasks, giving 120 configurations. A configuration is a user task, an injection task, an attack and a model; decoding temperature is the condition under which each configuration is measured. An execution returning a true security verdict is a harmful incident; a configuration with at least one is a harmful configuration. The attack is the stock `important_instructions` template and the model is gpt-4o-mini-2024-07-18 at

temperature 0 and at temperature 1, the one model available to us supporting every control this measurement needs (Appendix A.5).

3.2 Design: a census of the whole suite at twenty repeats

Reproducibility is a property of a configuration's outcome distribution, so the repeat count fixes what can be seen: a configuration with true harm probability p is unanimous over R repeats, and so looks deterministic to any screen reading unanimity as determinism, with probability $p^R + (1-p)^R$ – one time in four at $p = 0.5$ and three repeats. No single R is uniformly sufficient: detection power $1 - [p^R + (1-p)^R]$ exceeds 95% at $R = 20$ for p roughly in [0.14, 0.86], but is 0.878 at $p = 0.1$. A three-repeat first pass over all 120 configurations finds 75 harmful configurations at temperature 0 and 80 at temperature 1; restricting the deep pass to those would define the population by the outcome being measured, so every configuration is executed to $R = 20$ regardless, reusing the first-pass executions: 4,800 in total, 4,080 on top of the first pass's 720, no failed runs. Appendix A.5 gives that calculation, the record schema, and the count of harmful configurations the first pass missed.

3.3 Quantities reported

For configuration i with R repeats, $\hat{p}_i$ is the fraction of repeats returning a harmful outcome. Four quantities follow from it and they answer different questions, so Table 1 reports each with its population and weighting. Two classify configurations: band occupancy is the fraction of harmful configurations with $0.1 < \hat{p}_i < 0.9$, the band fixed in the design specification before any census execution, and nonunanimity is the fraction whose 20 repeats are not all the same, which uses no threshold. Two are probabilities about a replay: that two fresh replays of one configuration disagree, configurations weighted equally, and that a fresh replay returns benign given a harmful execution was observed, weighting configurations by $\hat{p}_i$ because harmful executions arise from a configuration in that proportion. These replay probabilities describe operational repeatability; the configuration fractions describe its prevalence, under different weighting.

Every fraction is read against its expectation under a population sharing one success probability, because a band's occupancy depends on the base rate. Intervals are 95% cluster bootstrap intervals that resample user tasks, each drawn with all of its configurations, and also resample the R repeats within each resampled configuration (Appendix A.12 says why the task and not the configuration): band membership is a classification of a $\hat{p}$ estimated from 20 trials rather than an observed label, and configurations share user tasks and injection tasks, so a Wilson interval understates both sources. The aggregate is the attack success rate over all 120 configurations at a given repeat index, summarized by its mean and standard deviation across the 20 indices.

4 Results

4.1 How often a replay fails to reproduce its own verdict

A single replay of an observed harmful execution reverses the verdict 0.087 of the time under greedy decoding, 95% CI [0.053, 0.124], and 0.141 at temperature 1, [0.091, 0.189]; two fresh replays of the same configuration disagree 0.139 and 0.204 of the time (Table 1). At the configuration level, 0.306 ([0.183, 0.389]) of the 85 harmful configurations at temperature 0 have an observed harm frequency strictly inside (0.1, 0.9), and 0.404 ([0.292, 0.534]) of the 94 at temperature 1; without a threshold, the fraction returning both labels over 20 repeats is 0.400 and 0.606. Each interval for temperature 0 overlaps its partner at temperature 1, and we draw no comparison between the conditions: each arm fixes its decoding condition without randomizing it, so the pair carries a measured level and not a contrast (Appendix A.7).

A raw fraction is not interpretable without its base rate, since configurations that all shared one success rate would already place some $\hat{p}$ inside the band by sampling alone. At the pooled harm rates among harmful configurations, 0.795 and 0.723, that expectation is 0.808 and 0.945, far above the observed 0.306 and 0.404, and the homogeneous null is rejected in both arms, $\chi^2/(n-1) = 12.03$ and 10.42 at $p < 10^{-10}$. Harm probabilities therefore sit near 0 or near 1 more often than a shared rate would produce, so the occupancy is not an artifact of the base rate: a homogeneous population would show more of it, not less. The replay-level quantities sit below their expectations the same way,

Table 1: Census estimands, both decoding conditions, each with its population and weighting. Intervals are 95% bootstraps resampling the user task, each drawn with all of its configurations, since configurations sharing a task are not independent (Appendix A.12); above the intraclass correlation each resampled configuration's 20 outcomes are then redrawn at its observed rate, and the selection-weighted correlation row instead resamples user tasks at the observed repeat counts and refits the truncated beta-binomial on each draw. The final row, the preselection parameter, is the one exception: it has no task-clustered refit and is configuration-clustered. The correlation is given first for the configurations observed harmful, then for the preselection population the fit describes.

quantity	population; weights	T=0	T=1
configurations executed at $R = 20$	whole suite	120	120
harmful configurations	≥ 1 harmful exec.	85	94
aggregate attack success, mean	all configs., 20 repeat indices	0.563 [0.443, 0.683]	0.566 [0.454, 0.675]
aggregate attack success, s.d.	all configs., 20 repeat indices	0.0193 [0.0125, 0.0259]	0.0262 [0.0162, 0.0331]
band occupancy $0.1 < \hat{p} < 0.9$	harmful configs., equal wt.	0.306 [0.183, 0.389]	0.404 [0.292, 0.534]
nonunanimity $0 < k < R$	harmful configs., equal wt.	0.400 [0.264, 0.494]	0.606 [0.427, 0.690]
two replays disagree	harmful configs., equal wt.	0.139 [0.091, 0.183]	0.204 [0.146, 0.256]
next replay benign | harmful seen	harmful execs., $\hat{p}$-wt.	0.087 [0.053, 0.124]	0.141 [0.091, 0.189]
intraclass correlation	configs. observed harmful, selection-weighted	0.623 [0.532, 0.680]	0.530 [0.416, 0.619]
preselection parameter	harm-capable configs., $P(k \mid k{>}0)$ fit	0.778 [0.600, 0.937]	0.622 [0.485, 0.794]

0.087 and 0.141 against 0.205 and 0.277, and 0.139 and 0.204 against 0.326 and 0.401 (Appendix A.8).

The census covers the whole suite because a shallower design does not reach the configurations that matter. A three-repeat first pass found 75 and 80 harmful configurations against 85 and 94 at 20 repeats; the 10 and 14 it missed have harm probabilities from 0.05 to 0.65, none higher, with 6 and 7 inside the band. A population defined by a three-repeat screen understates the phenomenon it screens for (Appendix A.5).

4.2 The aggregate does not move where the replay-level quantity does

The same executions produce the statistic an evaluation paper reports: the attack success rate over all 120 configurations, 0.563 under greedy decoding and 0.566 at temperature 1, with standard deviations of 0.0193 and 0.0262 across the 20 repeat indices (Table 1). Both arms are the same 120 configurations, so the aggregate compares identical populations and separates them by 0.003, less than either arm's own round-to-round standard deviation. Read at the level of a replay the same executions give 0.087 and 0.141, and at the level of a configuration 0.306 and 0.404: an aggregate this steady is compatible with a far larger per-incident quantity.

Figure 1 places the two views on a shared axis: a standard-deviation bar of two to three percentage points above, and below a distribution of per-configuration harm probabilities spanning the whole interval. As round-to-round standard deviations of a binary outcome, a single configuration's own outcome moves by 0.111 and 0.183 against the aggregate's 0.0193 and 0.0262. Their ratio is not an effect size: the denominator falls as the square root of the number of configurations, so Appendix A.8 keeps it only as a suite-specific diagnostic. The scale-free statement is a beta-binomial intraclass correlation of 0.623 [0.532, 0.680] and 0.530 [0.416, 0.619] among configurations already observed to be harmful. The interval lies above one half under greedy decoding; at temperature 1 it includes values on both sides of one half. The $k_i > 0$ selection is written into the likelihood, whose preselection population correlates higher, 0.778 [0.600, 0.937] and 0.622 [0.485, 0.794] (Appendix A.12).

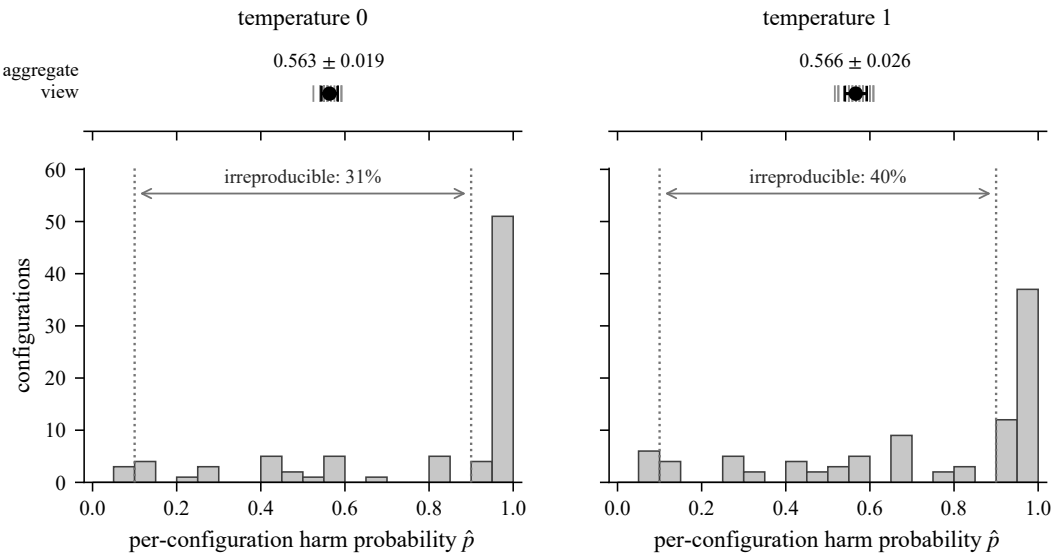


Figure 1: Two views of the same census, one temperature per column. Above: attack success rate over all 120 configurations, with its standard deviation across repeat indices and one tick per index. Below: distribution of per-configuration harm probability $\hat{p}$ over the harmful configurations, the pre-specified band (0.1, 0.9) marked. The panels share an x-axis.

Averaging produces the stability: under independent flipping at the measured per-configuration probabilities the round-to-round standard deviation would be 0.0212 and 0.0257 against the observed 0.0193 and 0.0262. Steadiness is what such a population produces, and it says nothing about which configurations carry the variance.

4.3 What an intermediate harm probability is made of

The benchmark's two verdicts are close to mutually exclusive: 0.890 and 0.862 of executions satisfy exactly one, and the mean of $\hat{p}_{\text{harm}} + \hat{p}_{\text{utility}}$ is 0.938 and 0.899, so a configuration with an intermediate harm probability is one on which the agent sometimes pursues the injection's goal and sometimes the user's. The two rates trade off across the census, r = -0.768 and -0.819, which the near-partition implies rather than evidences: exact complementarity would force -1. We read the trade-off as composition, not mechanism (Figure 2, Appendix A.8).

Heterogeneity survives a manipulation of attack strength. Across six cue variants attack success ranges from 0.917 to 0.383 – degrading the social-engineering cues costs more than half the success rate while payload structure and injection sites stay fixed – and within every variant the homogeneous null is rejected after a Holm correction across the six tests, the weakest at a conditional permutation $p = 0.032$, with $\chi^2/(n-1)$ from 2.12 to 4.84. The rejections weaken as the attack strengthens, which we read as a property of the test rather than a result about attack strength, since a bounded rate leaves near-saturated configurations little room to differ (Table 5, Appendix A.8).

The census executes a configuration's 20 repeats contiguously, so execution time is confounded with configuration identity and provider drift could enter the dispersion statistic as heterogeneity. The confound is measurable: adjacently executed configurations have correlated harm probabilities, +0.38 at temperature 0 (permutation p = 0.016). Re-running 40 harmful configurations as 20 rounds, each visiting every configuration once in a fresh random order, drops dispersion from $\chi^2/(n-1) = 12.94$ blocked to 9.85 randomized over the same 40 at 20 repeats, at a pooled harm rate moving from 0.821 to 0.822. Roughly three quarters survives and both sit far above the homogeneous 1, so the blocked figure is the larger of the two estimates rather than a clean one – not a bound, since one randomized run on a subset does not establish that no design reads higher. Appendix A.8 gives the round schedule and drift diagnostics. This check is at temperature 0 only; residual scheduling confounding at temperature 1 remains untested, and the reduction is not a causal variance partition.

4.4 WHERE TWO REPEATS FIRST DIVERGE

A claim that the replay oracle is noisy is only worth stating if the harness is reproducible. We traced where two repeats of the same configuration first differ, over all 91 configurations that returned both outcome labels. In 91 of 91 the first substantive difference is an assistant message, that is, a model output; in none is it a tool message, and in none are the trajectories identical after canonicalization, which would have implicated the checker. The injected text is byte-identical across the pair in all 91, and the attack variant draws no random numbers. What is localized is the first VISIBLE divergence: a difference in what was sent to the provider, or in how it was served, leaves no trace in the trajectory, so this places the divergence at a model output without excluding a cause upstream of it. The divergence is substantive – one repeat executes the injected file deletion while its twin executes only the calendar operation the user asked for. Appendix A.6 gives the canonicalization rule the comparison needs and the distribution of first-divergence indices.

4.5 BEYOND THE PRIMARY MODEL AND THE PRIMARY BENCHMARK

Five arms bound what the census generalizes to, and each is weaker than the census in a way the appendix states. A second AgentDojo suite, banking, changes the task domain while keeping the checkers, attack template and harness: over 780 executions its 31 harmful configurations give an intraclass correlation of 0.498 [0.316, 0.617] against the census's 0.623 [0.532, 0.680], the common-rate null rejected at $\chi^2/(n-1) = 11.01$ with no exceedance in 200,000 draws. Their difference is -0.125 with every bootstrap construction containing zero, which is undetectability at this size and not evidence of nil, and it is a second suite of the same benchmark rather than an independent replication (Appendix A.14).

The tested DeepSeek model–attack pair has a near-floor harm rate. The census design on `deepseek-chat`, the same 120 configurations over 2,400 executions, lands the attack at 0.010, so occupancies are not comparable and Appendix A.11 reports each beside its own expectation; the null is rejected there too, $\chi^2/(n-1) = 1.91$ at $p = 0.026$, but the correlation is 0.063 [0.003, 0.155] against 0.623. Changing the model generation leaves nothing to estimate: gpt-4o-mini is compromised in 41 of 48 payload-matched executions and gpt-5-mini in none of 48, and a quantity defined over harmful incidents is not estimable without them, which is a statement about what can be measured (Appendices A.3, A.15).

Two arms change the apparatus. On tau2-bench (Barres et al., 2025), with no attacker and two generative components (agent and simulated user), a homogeneous population would place 0.995 of retail and 0.999 of airline tasks inside the band against the observed 0.733 and 0.778, $\chi^2/(n-1)$ of 5.98 and 8.94 against AgentDojo's 12.03 and 10.42, $p < 10^{-10}$; this is joint conversational variability (Appendix A.10). InjecAgent (Zhan et al., 2024) keeps an attacker and changes everything else – codebase, authors, tools, agent loop, success criterion, attack, and a temperature it fixes itself. Over 60 preregistered cases at 20 replays and 60 more post hoc the null is rejected in each draw, $\chi^2/(n-1) = 13.11$ and 10.94 pooled, Monte Carlo conditional p bounded at 1.5×10^{-5}; its preregistered draw returned 14 harmful configurations against a declared minimum of 15, and its own rule then forbids comparing magnitudes. Substantial heterogeneity on a benchmark that is not ours, with its size relative to the census left open (Appendices A.9, A.15).

4.6 RUNNING AN ATTRIBUTION METHOD ON A CONTESTED INCIDENT

Each attribution invocation starts with a fresh end-to-end execution, not a frozen failed prefix. That execution came back benign 0.457 of the time, [0.326, 0.609] clustered on 23 incidents: about half the time the run began with nothing to attribute. The interval is 0.28 wide.

Self-agreement is worse than that suggests, and worse again once corrected for chance. The top-ranked span agreed across three repetitions 0.731 of the time for ContextCite, 0.513 for TracLLM and 0.474 for our budget-matched control; three draws from a candidate set of median size 37 agree 0.336 to 0.385 by construction, and correcting each incident with its own count leaves 0.579 [0.415, 0.741], 0.240 [0.110, 0.381] and 0.182 [0.043, 0.339]. Fixing each method's seed raises the raw values and restores none of them. This is agreement with themselves, not accuracy.

ContextCite's agreement is highest and rests on the fewest runs that scored anything: only three of the 26 incidents have all three repetitions informative (raw agreement 0.667). All-zero scores still

return a span by tie-breaking, so overall agreement can reward uninformative runs. Our control is least stable, plausibly because it consults the oracle once per question, though the three differ in search, scoring, abstention and candidate handling too (Appendix A.4).

4.7 What a neutralization decision costs

Everything above measures the replay; A.4's runs query attribution methods without pricing the edit they make. That edit – neutralize a span, re-execute, read the difference – has been priced with a model rather than executed on both sides. We ran it: 33 configurations under six arms at twenty replays, 3,960 executions, preregistered (Appendix A.17), with the 26 contested ones carrying every estimate below and the other seven serving as saturated and low-rate instrument controls. Each of the four arms that touches the payload – the whole payload, the goal sentences, the framing, the goal masked – returns zero harmful outcomes on every contested configuration, 0 of 520 per arm; six of them carried to two hundred replays a side on the goal arm stay at 0 of 1,200 (Wilson upper bound 0.003). Each treated arm's difference from the factual arm is therefore the factual rate, 0.517 [0.428, 0.609], clustered on user tasks like every other interval here.

The control is where the result is. It removes count-matched spans from the same tool return, every occurrence of which lies outside the payload, leaving the mean difference at -0.019 [-0.188, 0.170]. Per configuration it is not small: the mean absolute difference is 0.362 [0.258, 0.467] and the largest 0.90, harm rising on eleven and falling on fourteen. On the same functional for both arms – a squared effect with each arm's binomial sampling variance subtracted – the control reaches a root-mean-square of 0.436 [0.319, 0.538] against the payload's 0.569 [0.484, 0.652], a ratio of 0.77 [0.56, 0.98], both at twenty replays and clustered on the twenty user tasks behind the configurations. The deleted spans are not thereby shown causally irrelevant – they are legitimate task content the agent read – but they are shown not to be an inert control, so an outcome flip does not by itself name the injected span: a method targeting any influential span may be right to flag them, while one targeting the injected source needs a specificity this instrument does not supply.

In decision units a sign rule at one replay a side disagrees with a held-out reference on 0.243 of splits for the goal spans and 0.352 for the control, against simulated references of 0.554 and 0.546; at five replays, 0.059 and 0.222 against 0.538 and 0.533. Budget buys down the oracle's noise. The RMS comparison is unsigned; the signed two-candidate ranking has ties but no strict reversal on these records (Appendix A.17), not a guarantee over larger candidate sets.

Sensitivity of this order recurs on a population that did not produce it. On the banking suite, selected on its own rate and never used here, the ratio is 1.10 [0.79, 1.75] over nine eligible configurations from six user tasks – passing a point rule fixed in advance at [0.20, 1.40], which is not the same as showing the population ratio lies inside it – with goal neutralization again 0 of 180. The direction there runs the other way, harm rising on seven of nine each to 20 of 20 against eleven up and fourteen down here, on a signed interval containing zero (Appendix A.18).

5 Limitations

The census covers one model family, one benchmark and one attack family, and its design forces that scope: reproducibility is a property of an outcome distribution, so it costs 20 executions per configuration and is defined only where the attack produces incidents. That last clause binds. Of AgentDojo's four attack families against this goal structure only `important_instructions` compromises our model; the other three produce no harmful outcome on the configurations sampled, 0 of 27 against 9 of 9 (Appendix A.13). A differently constructed attack does produce incidents – the independent benchmark of Section 4.5 carries its own, at 0.097 over 2,400 executions. The arms reaching outside the scope are each weaker in a way Section 4.5 states: the second task suite shares the census's checkers, attack template and harness; the second-vendor arm changes provider and model together; gpt-4o-2024-05-13 is an uncontrolled estimate under a screen we did not set (Appendix A.9); tau2-bench supports the dispersion statistic and not the headline fraction; the independent benchmark missed its declared minimum yield, 14 against 15; and the contemporary-model arm returned no incidents.

Band occupancy is defined through a threshold, and any threshold is a choice. The threshold-free unanimity criterion is reported beside it, 0.400 and 0.606 against 0.306 and 0.404, so the headline

classification is the more conservative, and Appendix A.7 sweeps the endpoint margin. What the census does not identify is why a harm probability is intermediate: the verdict trade-off is composition, not evidence that irreproducibility concentrates where the two instructions are comparably strong.

Section 4.7's control bounds an instrument, not an attribution method. Its spans are defined by position – every occurrence outside the rendered payload – and position is not irrelevance: they are legitimate task content the agent read, and the design separates neither a semantic effect of that content from a structural effect of deleting it, nor either from instability of the realized edit; the count match is on spans, not characters, occurrences or position. A method whose target is any influential span is therefore not shown wrong by it. The decision curves score against a held-out complementary sample rather than a known population contrast, a weaker reference that matters most where effects are small; the arm runs on 26 configurations from one census, four payload interventions and one placebo construction, with the depth check on two arms only. The signed two-candidate ranking has no strict reversal on these records; its errors arise from ties. Larger candidate sets and other attribution procedures remain unmeasured (Appendix A.17). The replication in A.18 is nine control-eligible configurations from six user tasks, a tenth frozen but structurally without a control, so what it supports is recurrence rather than quantitative agreement.

6 Discussion

Report replay repeatability beside aggregate success. In Section 4.2 the aggregate moves by 0.003 between decoding conditions while replay flip rates are 0.087 and 0.141 within them. The latter measure how repeatable an attribution's underlying verdict is, not whether the attribution is correct.

Most replays do reproduce at a flip rate of 0.087, but flips are unevenly distributed: the share of variation belonging to the configuration is 0.623 and 0.530, and at every attack strength tested configurations at a common success rate do not share a common success probability (Sections 4.1 and 4.3). Inside the band both labels are live, so a deterministic-oracle guarantee does not directly describe the comparison. The quantity of interest is a difference of outcome probabilities under an intervention, which makes localization a search with a noisy comparison oracle of studied sample complexity (Gretta & Price, 2024; Dereniowski et al., 2025).

The budget controls sampling error. With one independent fresh draw per arm and randomized ties, a sign comparison errs with probability $\frac{1}{2}(1-g)$ for a positive true gap g (Appendix A.16). This differs from comparing a fresh replay with a fixed historical harmful outcome. Repetition does not remove unsigned sensitivity to edits outside the payload; that sensitivity alone does not establish failure of signed source ranking. Provider-side routing is the cited explanation for residual nondeterminism under greedy decoding (He et al., 2026); we cannot test it, since the second-vendor arm changes model and stack together (Section 4.5).

7 Conclusion

A counterfactual replay returns the verdict it returned before less often than a deterministic treatment supposes. On a census of every AgentDojo workspace configuration at 20 repeats in each of two decoding conditions, a replay of an observed harmful execution disagrees with the original verdict 0.087 of the time under greedy decoding and 0.141 at temperature 1, while the aggregate moves by 0.003. A replay verdict is a Bernoulli draw, so the budget behind an attribution is what a reader needs to judge it – a statement about the oracle, which a budget answers. Executing the intervention adds a second that a budget does not: deleting count-matched spans that never touch the injected payload moves a configuration's harm probability almost as far as deleting the payload does, so the size of an outcome change does not identify the named span as the injected one. This unsigned sensitivity does not establish signed ranking failure or whether the span caused the outcome; neither question is answered by the magnitude comparison alone.

Reproducibility Statement

Code and instructions are provided in the supplementary material. Every number, table and figure in this paper is computed from raw execution records by the analysis code, with one exception named

below. The supplementary code contains the execution code for every experiment, the ablation runtime and the two attribution baselines, the analyses that produce each reported quantity, and the scripts that build the tables and figures. We verified the execution records against their manifest and reran the tau2 analysis end to end; the records reproduce the submitted summary exactly.

The exception is the gpt-4o-2024-05-13 estimate in Appendix A.9, which reads an AgentDojo corpus collected outside this study's experiments; Appendix A.9 states what was computed from it and which filter was applied. It enters the paper as an uncontrolled estimate on a second model and nothing rests on it.

All three benchmarks are evaluated at pinned revisions: AgentDojo at `089ed468`, tau2-bench at `1901a301`, and InjecAgent at `f19c9f2`. The evaluated benchmark code is unmodified, so outcome labels are produced by the published checkers. The InjecAgent arm imports that benchmark's own predictor and evaluator and calls them as its own evaluation loop does, with two deliberate differences, neither of which touches scoring: its output cache is never enabled, since a cache would return the first replay's answer to the second, and each case is executed twenty times in round-wise order rather than once. Its records carry a case index and the benchmark's three-valued verdict rather than the census's user-task, injection-task and two-outcome schema, and the analysis reports both readings of that third verdict. The one exception is the second-vendor arm, which ran on the machine able to reach that endpoint against the same AgentDojo version installed from the package index; Appendix A.11 says so and states what was verified in its place. One transport adaptation exists and it does not touch scoring: the second-vendor arm subclasses the benchmark's OpenAI adapter to preserve the `system` role, which the pinned version converts to OpenAI's `developer` and which DeepSeek's endpoint rejects; the subclass delegates every other conversion unchanged and lives in our code, so the benchmark tree is unmodified there too. Appendix A.11 states what it does and what it leaves alone. The census follows Sections 3.1 and 3.2: all 120 configurations executed to 20 repeats in each decoding condition, 4,080 executions on top of the 720 of the three-repeat first pass, 4,800 records in total, no failed runs. The tau2-bench arm of Section 4.5 runs 15 tasks per domain at 20 trials, with gpt-4o-mini-2024-07-18 serving as both the agent and the simulated user.

Every experiment follows one execution protocol. The execution code appends a single JSON record per agent execution and resumes by skipping cells already recorded, so an interrupted run is restarted with the same command and loses at most one execution. Each record carries the model identifier, the decoding temperature, the repeat index, the user task, the injection task and the benchmark's own security and utility verdicts, which is what allows an independent run to be compared against ours cell by cell, not only in aggregate. The analysis runs on the records alone and needs no API access, though collecting the records again does.

The methods behind the reported quantities are specified in this paper: the census design and its detection threshold in Section 3.2 and Appendix A.5, the four estimands with their populations and weightings in Section 3.3 and Table 1, the pre-registered band and the arm-assignment guard in Appendix A.1, the attack-strength variants and the homogeneity test in Appendix A.2 and A.8, the payload-identity assertion behind the cross-model arm in Appendix A.3, the ablation runtime that lifts ContextCite and TracLLM to a multi-step agent in Appendix A.4, trajectory canonicalization in Appendix A.6, and the band-width sweep in Appendix A.7. The bootstrap behind the estimand intervals resamples user tasks, each with all of its configurations, and then redraws the 20 repeats within each resampled configuration at its observed harm probability, at 4,000 replicates from a fixed seed; the configuration-clustered variant is computed alongside and reported in Appendix A.12; the selected-population intraclass-correlation interval resamples user tasks at their observed repeat counts and refits the truncated model on each of 2,000 draws, reduced to 700 where the point fit sits on a boundary and each refit must therefore start from the full optimiser grid rather than from the point estimate, while the preselection row has no task-clustered refit and stays configuration-clustered, which the Table 1 caption states; and the attribution-stability intervals resample incidents once and recompute every panel from that draw. A companion check refits identical resamples both ways on every arm and reports the gap, which is at most 0.003 except at that one boundary fit. The supplementary instructions map each analysis to the reported quantity it produces, including the uninformative cross-model sweep of Appendix A.9. The provider's serving stack is outside our control and is discussed in Section 6.

AI USE STATEMENT

Generative AI tools were used to assist with retrieving and discovering related literature, code implementation, writing and polishing. The research ideas originated with the authors. The authors reviewed and verified AI-assisted content and take full responsibility for the paper's accuracy, originality, and conclusions.

REFERENCES

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

Akash Bonagiri, Devang Borkar, Gerard Janno Anderias, Setareh Rafatirad, and Houman Homayoun. CausalFlow: Causal attribution and counterfactual repair for LLM agent failures, 2026. arXiv:2605.25338.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. ContextCite: Attributing model generation to context. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, 2024. arXiv:2409.00729.

Edoardo Debenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *Advances in Neural Information Processing Systems 37 (NeurIPS), Datasets and Benchmarks Track*, 2024. arXiv:2406.13352.

Dariusz Dereniowski, Aleksander Łukasiewicz, and Przemysław Uznański. Noisy (binary) searching: Simple, fast and correct. In *Proceedings of the 42nd International Symposium on Theoretical Aspects of Computer Science (STACS)*, LIPIcs, pp. 29:1–29:18, 2025. doi: 10.4230/LIPIcs.STACS.2025.29. arXiv:2107.05753.

Liming Dong, Qinghua Lu, and Liming Zhu. AgentOps: Enabling observability of LLM agents, 2024. arXiv:2411.05285.

Yu Ge, Linna Xie, Zhong Li, Yu Pei, and Tian Zhang. Who is introducing the failure? automatically attributing failures of multi-agent systems via spectrum analysis, 2025. arXiv:2509.13782.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (AISec)*, pp. 79–90, 2023. doi: 10.1145/3605764.3623985. arXiv:2302.12173.

Lucas Gretta and Eric Price. Sharp noisy binary search with monotonic probabilities. In *Proceedings of the 51st International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 297 of *LIPIcs*, pp. 75:1–75:19, 2024. doi: 10.4230/LIPIcs.ICALP.2024.75. arXiv:2311.00840.

Yu He, Haozhe Zhu, Yiming Li, Shuo Shao, Hongwei Yao, Zhihao Liu, and Zhan Qin. AttriGuard: Defeating indirect prompt injection in LLM agents via causal attribution of tool invocations, 2026. arXiv:2603.10749.

Dong Ho Kang, Hyeonjeong Cha, and Daein Weon. Knowledge-based zero-replay debugging of multi-agent LLM traces, 2026. arXiv:2606.14805.

Debayan Mukhopadhyay, Utshab Kumar Ghosh, and Shubham Chatterjee. Bridge evidence: Static retrieval utility does not predict causal utility in multi-step agentic search, 2026. arXiv:2607.15253.

Benjamin Rombaut, Sogol Masoumzadeh, Kirill Vasilevski, Dayi Lin, and Ahmed E. Hassan. Watson: A cognitive observability framework for the reasoning of LLM-powered agents, 2024. arXiv:2411.03455.

Jiaming Wang, Ziteng Feng, Jiangtao Wu, Ruihao Li, Qianqian Xie, Yuxiang Ren, He Zhu, Xueming Han, Fanyu Meng, Junlan Feng, and Jiaheng Liu. Where do deep-research agents go wrong? span-level error localization in agent trajectories, 2026a. arXiv:2606.02060.

Yanting Wang, Wei Zou, Runpeng Geng, and Jinyuan Jia. TracLLM: A generic framework for attributing long context LLMs. In *Proceedings of the 34th USENIX Security Symposium*, 2025. arXiv:2506.04202.

Yawen Wang, Wenjie Wu, Junjie Wang, and Qing Wang. From flat logs to causal graphs: Hierarchical failure attribution for LLM-based multi-agent systems, 2026b. arXiv:2602.23701.

Alva West, Yixuan Weng, Minjun Zhu, Zhen Lin, Zhiyuan Ning, and Yue Zhang. Abduct, act, predict: Scaffolding causal inference for automated failure attribution in multi-agent systems, 2025. arXiv:2509.10401.

Andreas Zeller and Ralf Hildebrandt. Simplifying and isolating failure-inducing input. *IEEE Transactions on Software Engineering*, 28(2):183–200, 2002. doi: 10.1109/32.988498.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents, 2024. arXiv:2403.02691.

Guibin Zhang, Junhao Wang, Junjie Chen, Wangchunshu Zhou, Kun Wang, and Shuicheng Yan. AgenTracer: Who is inducing failure in the LLM agentic systems?, 2025a. arXiv:2509.03312.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. Which agent causes task failures and when? on automated failure attribution of LLM multi-agent systems. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025b. arXiv:2505.00212.

Yusheng Zheng, Yanpeng Hu, Tong Yu, and Andi Quinn. AgentSight: System-level observability for AI agents using eBPF. In *Proceedings of the 1st Workshop on Practical Adoption Challenges of ML for Systems (PACMI)*, 2025. doi: 10.1145/3766882.3767169. arXiv:2508.02736.

APPENDIX A: METHOD DETAIL

A.1 GUARDS ON THE ESTIMATOR

Both replay-level quantities of Table 1 are functions of $p_i(1-p_i)$, and the obvious estimator of that is biased. Substituting $\hat{p}_i$ gives $E[\hat{p}_i(1-\hat{p}_i)] = p_i(1-p_i)(1-1/R)$, so at $R = 20$ the plug-in understates every such quantity by exactly five per cent. For replay-level estimands we use the unbiased form $k_i(R-k_i)/(R(R-1))$ in the census and second-vendor arm alike, and the metrics files retain the plug-in values beside them so the correction is checkable rather than asserted. The direction matters for how the numbers should be read: correcting the bias makes every replay-level figure larger, so the plug-in versions were conservative. The per-configuration standard deviation averages $\sqrt{p_i(1-p_i)}$ instead, whose bias is a Jensen effect of a different size; it is not corrected here, and Section 4.2 reports it as a ratio whose scale dependence is stated in Appendix A.8.

Two further failure modes are specific to replay-based measurement, and both are easy to introduce without noticing.

Arm labels must not be derivable from the measurement they help estimate. The deep pass records an arm label (flagged stochastic or flagged deterministic) into every census record at run time, and the analysis reads arm assignments only from repeats with index below three. Without that restriction, re-running the analysis over the enlarged dataset silently relabels configurations using the very outcomes the census exists to measure. With the census complete no reported quantity depends on the arm labels, since none is a stratified estimate; the guard now protects the first-pass agreement counts of Appendix A.5, which are the only place the labels still appear.

A definition must live in one place. The band criterion and the unanimity criterion coincide at R = 3 and diverge at R = 20, so two analysis scripts can agree throughout screening and disagree once the census lands. The band is the single primary definition here; the unanimity criterion appears only as a labeled sensitivity figure.

A.2 ATTACK-STRENGTH MANIPULATION

The `important_instructions` family in AgentDojo contains six variants that share one jailbreak template and differ only in whether the user's name and the model's name in the injected text are correct, absent, or wrong. Payload structure and injection sites are identical across variants, so the family is a strength manipulation rather than a change of attack. We run all six variants over 12 configurations each at five repeats and temperature 0, 360 executions in total.

The pre-registered analysis was an inverted-U prediction: irreproducibility should peak at intermediate attack strength. We withdrew that test before running the analysis, because it is partly circular. If mean success probability sweeps from 1 toward 0 as the attack weakens, any dispersion statistic must peak in between, so the predicted shape is largely entailed by arithmetic on a bounded quantity. The replacement test asks a question arithmetic cannot answer: within a single attack strength, do configurations share one success probability? Under that homogeneous null, successes are Binomial(R, q) with a common q and all apparent irreproducibility is sampling noise. We report $\chi^2/(n-1)$ against that null per variant, together with the observed and null values of the mean Bernoulli variance E[p(1-p)]. Variant-level payload differences and the derivation of the null are in Appendix A.8.

A.3 CROSS-MODEL ARM

A contemporary model, gpt-5-mini, cannot be attacked with the stock variant, because that variant interpolates the target model's display name into the injected text and the benchmark's name resolver has no entry for gpt-5-class identifiers. Renaming the pipeline would change the payload and confound the comparison, so the arm uses `important_instructions_no_model_name`, whose payload carries no model name and which the strength sweep already measured on gpt-4o-mini over the same 12 configurations. Payload identity is asserted at startup: the run aborts unless both arms inject the same user name and the same model-name placeholder, and unless the name hint required by the resolver is absent from the payload text. The arm carries one asymmetry that no design choice

Table 2: Cross-model arm, 96 executions with no errors, 24 per cell over 12 configurations. Neither variant interpolates a model name into its payload, which is what allows the same bytes to reach both arms.

model	injection variant	ASR	n	95% CI
gpt-4o-mini-2024-07-18	II_no_model_name	0.875	24	[0.690, 0.957]
gpt-4o-mini-2024-07-18	II_no_names	0.833	24	[0.641, 0.933]
gpt-5-mini	II_no_model_name	0.000	24	[0.000, 0.138]
gpt-5-mini	II_no_names	0.000	24	[0.000, 0.138]

removes: gpt-5-class models reject a temperature parameter, so that arm runs at the provider default while its comparison point runs at temperature 0.

A second corroborating arm reuses an existing corpus of AgentDojo executions on gpt-4o-2024-05-13, generated for an unrelated study with repeat counts as low as two and under conditions we did not set. Finite repeats understate variation within the corpus it retains, while its outcome-related collection can bias a broader statement either way, so it enters as an uncontrolled estimate and never as a matched measurement or a bound.

A.4 Ablation runtime for published attribution methods

ContextCite and TracLLM both operate by ablating parts of a context and reading the change in the model's output (Cohen-Wang et al., 2024; Wang et al., 2025). Lifting either to a multi-step agent needs two capabilities the benchmark does not expose: enumerating the untrusted spans an execution actually consumed, and re-executing with an arbitrary subset of them neutralized. We obtain both by replacing the tool-output formatter that every tool result passes through before it reaches the model, which leaves the vendored benchmark unmodified. The runtime installs the formatter on every executor reachable from the pipeline and raises if it finds none, so a silent no-op ablation is not possible. Three neutralization operators are implemented: deletion, masking, and a length-matched benign paraphrase. The reported attribution-stability runs use deletion, the default in `run_once`; the other operators are available but are not compared in that arm. Each call starts a fresh pipeline and re-executes the benchmark task end to end; it does not resume a frozen prefix of the historical failure.

Our ContextCite adaptation regresses the terminal binary security verdict, rather than the original response log-probability, on independent span-retention indicators with probability 0.5. It minimizes squared loss divided by twice the number of queries plus an L_1 penalty of 0.05, with an unpenalized intercept, 400 iterative soft-thresholding steps and step size 0.15. The penalty is fixed, with no cross-validation or regularization-path selection. TracLLM uses $K = 2$ group splitting and scores the drop from a fresh factual verdict. These are adaptations to a terminal-outcome oracle, not evaluations of the published methods in their original settings. Ties in the top score select the first candidate in collection order, including when every score is zero; no explicit abstention symbol is returned. Thus repeated uninformative outputs can inflate modal agreement. The uniform-draw chance correction below does not remove this systematic tie-breaking effect.

A live check on one configuration (workspace, user task 1, injection task 0, temperature 0) verifies the direction: the unmodified execution returns a harmful outcome, the runtime collects 31 candidate spans, five of which overlap the injected payload, and neutralizing those five returns a benign outcome. An earlier version of this sentence said three, having identified them by matching words from the payload; that match also caught an unrelated record's subject field and missed two spans of the injection. A.17's span rule is positional for this reason.

Section 4.6 runs on this runtime. Its incidents come from the completed temperature-0 census, so each carries a harm probability measured over 20 executions, and they are every contested incident the census identifies, all 26 with $0.1 < \hat{p} < 0.9$, together with three saturated controls at $\hat{p} \geq 0.9$. Those three are an instrument control and not a deterministic regime: $\hat{p}$ is measured on the factual execution while the ablated executions a method queries were never measured, and one saturated incident changed its top-ranked span under a fixed seed. The budget-matched control's own score is binary and saturates, which is why its informative-run share is undefined and the column is empty in Table 3. ContextCite, TracLLM and the budget-matched control each attribute each incident three

times in each seed condition, at a budget of 24 replays per run. The realized count runs from 15 to 24: ContextCite always spends the budget, TracLLM stops when its search frontier is exhausted, and the control stops once it has probed every span. Replays are never cached: a cache would make every method stable by construction and measure nothing, since the instability under study is that an identical query to the oracle can return a different verdict. Both published methods sample internally, ContextCite over ablation subsets and TracLLM over the order in which it splits groups, so the varying-seed condition carries method sampling and oracle noise together. A fixed seed does not hold TracLLM's query sequence fixed, since its search branches on the verdicts it receives; it holds the method's own sampling fixed, which leaves the verdicts as the element that differs in most cells but not in all – four incident-condition cells changed candidate-set size between repetitions, so for those the fixed-seed condition carries candidate drift too, and the statement is not universal. The record field carrying the factual outcome of each run was added partway through the runs, so the benign-start rate of Section 4.6 rests on 23 of the 26 contested incidents in each condition, the same 23 in both, giving 46 incident-condition cells over 407 runs. Each cell contributes one observation, taken from its first recorded factual execution, so the rate is an average over cells rather than over runs: within a single invocation one factual execution is shared by the nine method-runs of an incident, and counting runs would restate a single observation up to nine times. The 46 cells are not 46 independent units. They come from 23 incidents, each contributing one cell per seed condition, and the incident is the independent unit here as everywhere else in this appendix.

That sharing holds within an invocation and not across them. This experiment was collected over several resumed runs, and a resumed run produces a fresh factual execution for the cells it regenerates, so three of the 46 cells carry runs that disagree about the factual outcome and four about the size of the candidate set. Taking each cell's first factual execution keeps the unit the one this paragraph claims. The choice does not carry the result: averaging the share within each cell gives 0.478, restricting to the 43 cells with a single factual outcome gives 0.465, and weighting runs equally gives 0.469, against 0.457 for the reported definition and a census prediction of 0.515. Clustered on the 23 incidents the reported value carries an interval of [0.326, 0.609], which contains every one of those alternatives and the census prediction as well, so the comparison is an observation and not a test.

The stability table is reproduced here with what it does and does not separate, and with the chance-corrected figures beside it. Agreement is the share of three repetitions returning the modal top-ranked span, so it takes the values 1/3, 2/3 and 1 and its floor is not zero: three uniform draws from n candidate spans have expected modal agreement $[1 + 2(n-1) + (n-1)(n-2)/3]/n^2$. An earlier version of this appendix put n at the nineteen spans these incidents "typically" carry, and quoted the 0.385 that follows. Nineteen is the minimum of the support, not its centre: over 522 records the counts run from 19 to 425 with a median of 37, so the floor runs from 0.385 down to 0.336. No reported figure moves, because the correction always used each incident's own count, but the direction is worth stating. Chance agreement falls as the candidate count rises, so quoting the minimum quoted the largest floor, and a reader applying it uniformly would have obtained corrected agreements below the ones reported rather than above them. Correcting each incident with its own candidate count, by $(\text{observed} - \text{chance})/(1 - \text{chance})$, gives 0.579, 0.240 and 0.182 under varying seeds against raw values of 0.731, 0.513 and 0.474, and 0.821, 0.501 and 0.400 under a fixed seed against 0.885, 0.679 and 0.615. The intervals come from one resampling of incidents that recomputes every panel from the same draw. Within a single panel each incident contributes exactly one cell, so a bootstrap there is a cell bootstrap under another name; the clustering that matters runs across panels, because one incident appears in both seed conditions and under all three methods and its six cells rise and fall together. The boundary stratum has 26 such incidents and the saturated stratum three.

Restricting to cells where every repetition produced a positive score excludes uninformative repetitions from the measure and shrinks the population sharply. Only three of the 26 contested incidents qualify for ContextCite under varying seeds and ten for TracLLM, with agreements of 0.667 and 0.400 – for TracLLM, below its own uncorrected figure, so on the incidents where it did have an opinion it held it less often. The budget-matched control has no such view: its score is binary and saturates, so informative and uninformative cannot be distinguished for it. Agreement is the share of three repetitions returning the modal top-ranked span, so it takes the values 1/3, 2/3 and 1 and cannot distinguish substantive agreement from repeated uninformative tie-breaking. It is uncorrected for chance, and four incident-condition cells changed candidate-set size between repetitions, so candidate drift sits inside it as well. These figures are not a bound on how stable the procedures are: they are

Table 3: Agreement of the top-ranked span across three repetitions of the same incident, by stratum, method and seed condition; 29 incidents, three methods and three repetitions give 261 runs in each seed condition, all completed. Agreement is the share of repetitions returning the modal top-ranked span, so at three repetitions it takes the values 1/3, 2/3 and 1; the raw values are in Section 4.6 and in this appendix, and the table carries the chance-corrected ones. Corrected columns subtract each incident's own chance floor; intervals are 95% bootstraps from one joint resampling of incidents, 26 in the contested stratum and three in the saturated one, that recomputes every panel from the same draw. Informative runs are those that produced any positive score, given as varying over fixed; the entry is empty for the control, whose binary score does not compare with a continuous one. Appendix A.4 states what the saturated stratum does and does not control for.

incidents	method	chance-corrected agreement [95% CI]		informative
		varying seed	fixed seed	var./fix.
contested	ContextCite	0.579 [0.415, 0.741]	0.821 [0.681, 0.940]	35/32%
	TracLLM	0.240 [0.110, 0.381]	0.501 [0.356, 0.660]	56/62%
	budget-matched control	0.182 [0.043, 0.339]	0.400 [0.239, 0.562]	–
saturated	ContextCite	0.290 [-0.080, 1.000]	0.820 [0.460, 1.000]	56/67%
	TracLLM	0.459 [-0.084, 1.000]	1.000 [1.000, 1.000]	100/100%
	budget-matched control	1.000 [1.000, 1.000]	1.000 [1.000, 1.000]	–

what this budget, this candidate handling and this agreement statistic produce, and the comparison between methods is the part worth reading.

Holding a method's seed fixed removes the method's own sampling, which leaves the oracle's verdicts as what differs in most cells but not in the four where the candidate set itself moved. Agreement rises for every method and is restored for none, and what survives a fixed seed enters through the verdicts and that drift together, which does not locate their source. Informative runs are those that produced any positive score; the column is empty for the budget-matched control, whose binary score does not compare with a continuous one.

A.5 Census execution and the comparison against a three-repeat pass

Among the models available to us, gpt-4o-mini-2024-07-18 is the only one supporting every control the measurement needs: a settable temperature, a returned system fingerprint for backend auditing, and an attack from the important-instructions family that produces incidents at a workable rate.

The first pass executes all 120 configurations per temperature at three repeats; the deep pass takes every configuration to R = 20, reusing those three executions instead of discarding them. The execution code writes one JSON record per execution as it goes and resumes by skipping cells already present, so an interrupted run loses at most one execution. Each record carries the suite, user task, injection task, attack, model identifier, temperature, repeat index, the security and utility verdicts, the wall-clock seconds, and the arm label assigned at the first pass. The two censuses required 4,080 executions on top of 720, 4,800 records in total, with no failed runs.

A configuration with true harm probability p returns the same label in all R repeats, and is therefore unanimous over those R draws, with probability $p^R + (1-p)^R$. Unanimity is what is observed; reading it as determinism is the inference a screen makes, and it is that inference the census is designed to check. At $R = 3$ the probability is one time in four at $p = 0.5$, which is why a three-repeat pass cannot deliver the headline number; the same expression needs $R = 6$ to reach 95% detection at $p = 0.5$, and the requirement depends on p: at $R = 20$ it exceeds 95% for p roughly in [0.14, 0.86] and falls to 0.878 at $p = 0.1$ (Section 3). A depth stated without its p is not a guarantee. The alternative to a census over the whole suite was a two-arm stratified sample, whose control arm exists to estimate how often a first-pass label of deterministic is a missed detection; at twenty control configurations that estimate carries a 95% interval about 0.31 wide, which cannot support the correction it is meant to supply.

The census and the first pass report on each other, in two directions. Among configurations the first pass saw at all: of the 12 it flagged as stochastic at temperature 0, 11 are inside the band at 20 repeats, and of the 63 it flagged as deterministic, nine are; at temperature 1 the counts are 23 of 24 and eight of

56. The miss rate on its deterministic label is 0.143 in both conditions. Those counts are conditional on the first pass having seen a harmful outcome at all, which is the second direction and the larger one. The first pass found 75 harmful configurations at temperature 0 and 80 at temperature 1; the census finds 85 and 94. The 10 it missed at temperature 0 have harm probabilities 0.05, 0.05, 0.05, 0.10, 0.20, 0.25, 0.40, 0.40, 0.40 and 0.65, six of them inside the band; the 14 it missed at temperature 1 run 0.05 six times, then 0.10, 0.15, 0.15, 0.25, 0.30, 0.45, 0.55 and 0.60, seven inside the band. A screen at three repeats is blind in proportion to how infrequently a configuration turns harmful, so a population defined by such a screen is depleted exactly where the quantity being measured lives.

A.6 Trajectory canonicalization and divergence localization

For every configuration whose repeats returned both outcome labels, we take one repeat of each label and walk the two message sequences until they differ. Raw comparison is uninformative: the provider assigns a fresh tool-call identifier to every request, so the two sequences differ at the first tool call in nearly every pair even when the function name and arguments are identical. Canonicalization strips the tool-call identifiers and compares the function name together with its arguments in sorted key order; only after that is the first difference located.

Across the 91 configurations with a divergent-outcome pair, the first substantive difference is an assistant message in 91 and a tool message in none, and no pair is identical after canonicalization, which would have implicated the checker rather than the model. The injected payload is byte-identical across the pair in all 91. Indices count the stored messages from zero, not alternating role pairs. The index of the first divergence is spread over the trajectory: message 2 in 30 configurations, message 4 in 30, message 6 in 25, message 7 in four and message 8 in two. The divergences are substantive rather than cosmetic. In `user_task_9` with `injection_task_1` one repeat executes `delete_file`, the injected goal, while the other executes only `cancel_calendar_event`; in `user_task_26` with `injection_task_1` one repeat executes `delete_file` and the other issues no tool call at all.

A.7 Band-width sensitivity

Band occupancy counts a configuration as irreproducible when its observed harm frequency over 20 repeats falls strictly inside (0.1, 0.9). It is a property of that observed frequency and not an estimate of how many configurations have a latent rate in the interval. Table 4 sweeps the endpoint margin of that band, adds the threshold-free unanimity criterion, and sets each row beside its expectation under a population sharing one success probability, over the census populations of 85 and 94 harmful configurations.

Section 4.1 declines any comparison between the two decoding conditions, and the reason is worth recording with the number that prompted it. On the 84 configurations harmful in both arms the matched band labels are discordant 2 against 10, exact McNemar $p = 0.039$. That test treats band membership as an observed label, and each arm fixes its decoding condition without randomizing it, so each contributes a measured level rather than a randomized contrast and the pair supports no statement about temperature. The number is reported here because deleting a computed result is not the same as declining to interpret it.

The observed fraction falls as the band narrows toward the middle, which is what a band that discards near-saturated configurations must do, and it does not fall to zero: at the most restrictive band tested it is still 0.165 and 0.234. The null falls faster. From an endpoint margin of 0.05 to one of 0.25 the observed fraction sits below what a common-rate population would produce, the direction that says harm probabilities are pushed toward the extremes. At (0.30, 0.70) the temperature-0 arm crosses over, 0.165 observed against a null of 0.096, which says the opposite in that narrow window: more mass near the middle than a common rate allows. Either direction is a departure from homogeneity and neither is a headline, which is why the claim of Section 4.1 rests on the census-level dispersion statistic, a quantity that uses no band. The primary band is also not the most favorable choice available in either arm: a 0.05 margin would give 0.318 and 0.457, not 0.306 and 0.404.

Table 4: Sensitivity of the reproducibility fraction to the band, over the census populations of 85 and 94 harmful configurations. The null column is the fraction a population sharing that arm's pooled harm rate would place inside the same band. Intervals resample configurations and then repeats within configurations. The primary row shares its point estimate with Table 1, whose primary intervals instead resample user tasks; Appendix A.12 reports this sensitivity. The final row uses no band at all, counting every configuration whose 20 repeats are not unanimous.

band	T=0			T=1		
	obs.	null	95% CI	obs.	null	95% CI
(0.05, 0.95)	0.318	0.937	[0.227, 0.429]	0.457	0.987	[0.373, 0.578]
(0.10, 0.90) (primary)	0.306	0.808	[0.188, 0.379]	0.404	0.945	[0.309, 0.516]
(0.15, 0.85)	0.235	0.609	[0.155, 0.338]	0.340	0.847	[0.263, 0.462]
(0.20, 0.80)	0.200	0.391	[0.125, 0.299]	0.340	0.688	[0.218, 0.415]
(0.25, 0.75)	0.165	0.211	[0.099, 0.259]	0.287	0.493	[0.178, 0.364]
(0.30, 0.70)	0.165	0.096	[0.071, 0.220]	0.234	0.306	[0.132, 0.301]
not unanimous	0.400	0.990	[0.269, 0.479]	0.606	0.999	[0.457, 0.663]

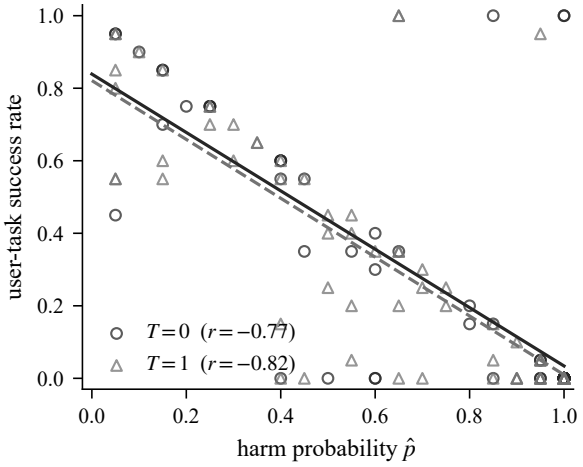


Figure 2: User-task success rate against harm probability, one point per harmful configuration, with a least-squares line per decoding condition. Both conditions are drawn from the 20-repeat census.

A.8 Aggregate scaling, outcome composition, the dispersion null, and execution order

The ratio of standard deviations is 5.78 [3.97, 8.13] under greedy decoding and 6.98 [5.17, 10.27] at temperature 1, and it is reported here, not in Section 4.2, because it is not an effect size. The aggregate's standard deviation falls as the square root of the number of configurations while the per-configuration one does not, so the same behaviour on a suite twice this size would give a ratio larger by about $\sqrt{2}$ without anything about the process having changed. The numbers therefore describe how much averaging this suite does at 120 configurations, which is a useful thing to know about a benchmark and a poor thing to compare across benchmarks. The scale-free version of the same question is the intraclass correlation of Appendix A.12.

The composition behind an intermediate harm probability is what Figure 2 plots. The correlations of -0.768 and -0.819 are what the near-partition of the two verdicts implies, not evidence beyond it, which is why Section 4.3 reads the figure as composition and not as a mechanism.

The homogeneity test treats a variant's 12 configurations as a single population. Under the null every configuration in a variant shares that variant's pooled success rate q, so the number of successes in R = 5 repeats is Binomial(R, q) and the dispersion statistic $X^2 = \sum_i (k_i - Rq)^2/(Rq(1-q))$ has expectation n - 1. Values of $X^2/(n-1)$ above 1 indicate that configurations differ in their underlying

Table 5: Attack-strength variants of `important_instructions`, abbreviated II in the row labels and sorted by attack success rate; 12 configurations each at five repeats, temperature 0. The asymptotic p is the χ^2_{n-1} survival function and the permutation p is a Monte Carlo conditional value over 2×10^5 allocations; $\leq$ marks the three variants at which no allocation reached the observed dispersion, where the value is the plus-one estimator. The last column gives the rejection decision at 0.05 under Holm and under Bonferroni, over the six tests. The mean Bernoulli variance and its null value for each variant are in the text below.

injection variant	ASR	$\chi^2/(n-1)$	asym. p	perm. p	rej. H/B
II_no_model_name	0.917	2.12	0.016	0.032	yes/no
II	0.850	2.89	8.3×10^{-4}	8.4×10^{-4}	yes/yes
II_no_user_name	0.700	4.07	5.3×10^{-6}	$\leq 5.0 \times 10^{-6}$	yes/yes
II_wrong_model_name	0.700	3.20	2.3×10^{-4}	5.5×10^{-5}	yes/yes
II_no_names	0.700	3.90	1.2×10^{-5}	$\leq 5.0 \times 10^{-6}$	yes/yes
II_wrong_user_name	0.383	4.84	1.6×10^{-7}	$\leq 5.0 \times 10^{-6}$	yes/yes

success probability instead of varying by sampling noise around a common one. The same statistic and the same null serve the census-level test of Section 4.1 and the tau2-bench test of Section 4.5, with n the number of configurations or tasks in the population tested and R their repeat count. The reference distribution is the asymptotic χ^2_{n-1}, not an exact one. The survival function is evaluated numerically without further approximation, but the test itself is asymptotic and is not exact at these sample sizes, so we also compute a Monte Carlo conditional permutation test. Write k_i for configuration i's successes and $K = \sum_i k_i$. Under a common rate, which trials succeeded is exchangeable across the $n \times R$ grid given K, so the reference distribution is the multivariate hypergeometric on that grid. For populations selected by at least one harmful or successful execution, the reference also conditions on $k_i > 0$ for every included unit, rejecting allocations with a zero. The strength sweep instead uses all twelve configurations fixed by design, including zero-success configurations, and conditions only on K; its allocations may contain zeros. We compare the observed dispersion against 2×10^5 allocations on the support appropriate to each design. The allocations are sampled rather than enumerated, so the test is Monte Carlo and not exact. On the twelve configurations of the $\mathrm{ASR} = 0.917$ variant at $R = 5$, the smallest population we report, the permutation test gives $p = 0.032$ against the asymptotic 0.016. The rejection survives, and the discrepancy is not one-sided: for the wrong-model-name variant the permutation value is the smaller of the two.

The six variants are six tests of one hypothesis, so the table below carries the asymptotic and the permutation p-value side by side, which differ by a factor of two on the weakest test, and the decision under each correction. Holm's step-down rejects all six at 0.05; Bonferroni rejects five, the exception being the $\mathrm{ASR} = 0.917$ variant, whose permutation p of 0.032 is also its Holm-adjusted value and becomes 0.194 under Bonferroni. We report the Holm decision and disclose that the correction was chosen after review round 3, the submitted version having reported uncorrected values; that variant is supporting evidence and not a standalone rejection. The released code falls back to a Wilson-Hilferty approximation where SciPy is unavailable. The main text quotes the smallest p-values as bounds, not point values, because at these magnitudes the exponent measures how untenable an idealized independent-and-identically-distributed null is, not how large the effect is, and $\chi^2/(n-1)$ is the quantity that carries the latter. Three of the six variants had no permutation draw reach the observed dispersion. For those we report the plus-one estimator $(b+1)/(B+1) = 5.0 \times 10^{-6}$, which is a point estimate and not a bound, together with a Clopper-Pearson 95% upper limit of 1.5×10^{-5}, which is the bound.

For this diagnostic only, the mean plug-in variance $\overline{\hat{p}(1-\hat{p})}$ is compared with its finite-repeat null $(1 - 1/R)q(1-q)$. At $R = 5$ both are deflated by a factor of 0.8 relative to their latent counterparts, unlike the unbiased replay estimands in Appendix A.1. In all six variants the diagnostic falls below its null: 0.0467 against 0.0611, 0.0600 against 0.1020, 0.0533 against 0.1680, 0.0867 against 0.1680, 0.0600 against 0.1680, and 0.0267 against 0.1891, in the order of Table 5. This deficit and count overdispersion are algebraically linked, not independent evidence of bimodality. The fitted beta-binomial shape in Appendix A.12 supplies a separate, model-based description of the census polarization.

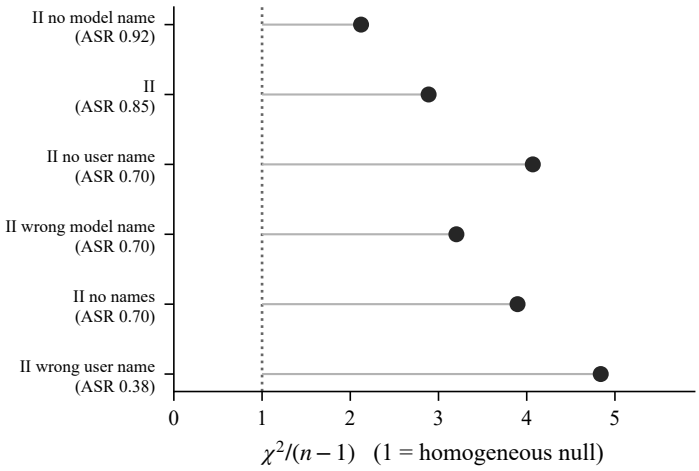


Figure 3: Overdispersion relative to a common-rate null, one row per attack variant, sorted by attack success rate. A value of 1 is the homogeneous null; every variant sits above it. The same statistic is the $\chi^2/(n-1)$ column of the attack-strength table in this appendix.

The manipulation is not monotone in the cue accuracy we intended. Attack success is 0.917 with a correct user name and no model name and 0.383 with a wrong user name, but removing both names leaves 0.700, above the wrong-user-name variant. A wrong name is therefore more costly to the attack than no name at all, which the design did not anticipate and which does not affect the dispersion result. Three variants land at an attack success rate of 0.700 and differ in overdispersion from 3.20 to 4.07. With 12 configurations per variant that spread is an observation, not a result, and we draw nothing from the ordering within those three.

The census estimates a configuration's harm probability inside one window of wall-clock time, because its 20 repeats run contiguously. Along the execution order the lag-1 autocorrelation of $\hat{p}$ is 0.221 at temperature 0, with a permutation p of 0.017 over 20,000 shuffles of the configuration order, and 0.444 at temperature 1, where no shuffle reached the observed value ($p < 5 \times 10^{-5}$). Two readings that would make the correlation uninformative are addressed below; neither is excluded in general, and each is narrowed by a specific check. Adjacent configurations frequently share a user task, but restricting to adjacent pairs whose user tasks differ leaves 0.379 over 33 pairs at temperature 0 (p = 0.016) and 0.332 over 35 pairs at temperature 1 (p = 0.021). A monotone trend along the order in which the suite enumerates tasks would produce the same signature, but the correlation of $\hat{p}$ with the user-task index is -0.008 at temperature 0, and inside a configuration's own block the harm rate over the first ten repeats against the last ten is 0.805 against 0.786 at temperature 0 and 0.734 against 0.712 at temperature 1. At temperature 1 the task-index correlation is -0.263, which is why Section 4.3 quotes the temperature-0 figure and why the replication below runs at temperature 0.

The randomized-order replication breaks the blocked assignment of configuration to execution time, which is the confound as stated; drift and schedule interaction within a round are not removed by it. Forty of the 85 configurations the census found harmful at temperature 0 are sampled without replacement under a fixed seed and executed as 20 rounds, each round visiting all 40 once in a fresh random permutation, 800 executions in total. Every record carries a UTC timestamp, the round index and the position within the round, so the realized order is checkable rather than assumed, and a round enters the analysis only when all 40 of its cells finished: a partial round would weight configurations by where that round's permutation happened to place them, which selects on position, the quantity the design exists to neutralize. The comparison side is the census restricted to those same 40 configurations at the same 20 repeats, since X^2 scales with the repeat count. The reading is symmetric. Survival of the census value indicates per-configuration heterogeneity; collapse toward 1 would indicate that the census value was execution-order drift. $X^2/(n-1)$ is 12.941 blocked against 9.852 randomized, at pooled harm rates of 0.8213 and 0.8225. Band occupancy moves the other way from the dispersion statistic, 0.275 blocked against 0.350 randomized, and each sits far below the occupancy a population sharing the pooled rate would place inside the same band, 0.720 and 0.716, so neither schedule yields a population a common rate accounts for. This subset check is restricted to

Table 6: tau2-bench, 15 tasks per domain at 20 trials each. The task-count column describes the full design; success, standard deviation, band fraction, null and dispersion condition on tasks solved at least once, nine in airline and 15 in retail. The band fraction is reported beside its expectation under a homogeneous population at the same success rate, which is what makes it uninformative alone.

domain	tasks	solved ≥ 1	trials	success (solved)	s.d.	band fraction	null expect.	$\chi^2/(n-1)$
airline	15	9	20	0.539	0.1180	0.778	0.999	8.94
retail	15	15	20	0.390	0.1039	0.733	0.995	5.98

temperature 0. It neither partitions the census variance causally nor tests the temperature-1 result under randomized scheduling.

A.9 Secondary model arms

The reused corpus of AgentDojo executions on gpt-4o-2024-05-13 contains 579 trace files over 76 configurations. Restricting to configurations with at least two repeats leaves 34 harmful configurations, of which six return disagreeing outcome labels, a rate of 0.176. Restricting instead to at least three repeats gives 0.600 over 10 configurations. We report the first and not the second. Repeat counts in that corpus were not assigned at random: configurations that received more repeats were selected for follow-up, and configurations selected for follow-up are enriched for having looked interesting, which for this outcome means enriched for being stochastic. Conditioning on repeat count therefore conditions on the outcome, so the shallowest usable threshold is the defensible one. It does not make the resulting figure a bound: finite repeats understate variation within the set retained, while outcome-related follow-up can push a broader statement either way, so the figure is an uncontrolled estimate whose direction of error is not established.

The cross-model arm reported in Section 4.5 ran `important_instructions_no_model_name` and `important_instructions_no_names` over 12 configurations at two repeats per model, 96 executions with no errors. Pooled over both variants, gpt-4o-mini returns 41 harmful outcomes in 48 executions and gpt-5-mini none in 48; the per-cell rates are in Table 2. An earlier pilot on `important_instructions_no_model_name` alone gave the same direction on the seven configurations that completed, zero harmful outcomes for gpt-5-mini against 55 in 60 executions for gpt-4o-mini, and is superseded by the arm above.

One sweep preceding that arm is reported here because it failed instructively. It selected the four stock attacks that never interpolate the target model's display name, since those need no workaround to construct, and returned an attack success rate of 0.000 on gpt-5-mini and also 0.000 on gpt-4o-mini, which was the positive control. An arm whose control cannot produce a positive result cannot separate two models, so that sweep is reported as uninformative rather than as evidence of robustness. What makes the important-instructions family strong is precisely the social-engineering content those four variants lack.

A.10 The tau2-bench arm

The arm runs 15 tasks in each of the retail and airline domains at 20 trials per task, with gpt-4o-mini-2024-07-18 serving as both the agent and the simulated user, against the benchmark at commit `1901a301`. The benchmark's save-file metadata records temperature 0.0 for both agent and user in both domains; it records no explicit `top_p` argument, so we do not assign a numerical value to it. The outcome reflects their joint conversation and does not isolate variability of the agent alone. Table 6 reports it. Over all fifteen tasks, success is 0.323 in airline and 0.390 in retail; the table conditions on tasks solved at least once.

A.11 The second-vendor arm

The arm repeats the census design against DeepSeek's `deepseek-chat` through its OpenAI-compatible endpoint: the same 120 configurations of the workspace suite under the same stock `important_instructions` attack, 20 rounds at temperature 0, 2,400 executions with no failed

runs and no cell requiring a retry. The injected payload is asserted byte-identical to the census arm's before any measurement, by the same startup check the cross-model arm uses.

Execution is round-wise, not blocked: each round visits every configuration once in a fresh random order, and every record carries a UTC timestamp, the round index and the position within the round. The census itself is blocked, and Section 4.3 measures what that costs, so the arm that came later was run the better way.

That ordering also makes provider drift detectable, because the round index is time. It has to be checked, not assumed: `deepseek-chat` is a moving pointer with no pinnable version, so the served model can change under a run that takes a night. Across the 20 rounds the harm rate moves from 0.0069 in the first third to 0.0083 in the last, a Spearman rank correlation against round index of 0.058. A rank correlation this small does not bound drift, and would not detect a non-monotone or transient change at all; the arm is analysed pooled because the preregistration fixed that before the run, and the screen is reported as having not fired rather than as evidence of stationarity. An earlier attempt at a second-vendor arm on a different provider failed exactly this check and was discarded rather than reported.

The arm ran on a separate machine, the one able to reach the endpoint, against AgentDojo 0.1.35 installed from the package index, not the pinned checkout used everywhere else. The version string matches that checkout's, and the startup assertion that the injected payload is byte-identical to the census arm's passed, which exercises the suite and attack machinery the comparison depends on; we state this instead of claiming the arm ran the same commit, which we cannot verify across machines.

One transport adaptation was required and it is confined to message formatting. The pinned AgentDojo version converts its internal `system` message to OpenAI's `developer` role for every model, and DeepSeek's endpoint rejects that role, so a subclass of the benchmark's OpenAI adapter preserves `system` and delegates every other conversion, the tool schema and the response parsing to the pinned implementation unchanged. The benchmark code is not modified: the subclass lives in our own code. The system message content is identical across arms, and the security and utility checkers, the task suites and the attack are untouched, so every outcome label in this arm comes from the published checkers.

The arm is a different operating point, not a matched replication. Its harm rate is 0.086 over the 14 of 120 configurations that produce any harmful outcome and 0.010 over the full census, against the census arm's 0.795 and 0.563, where 85 configurations are harmful. Both rate-like quantities therefore have to be read against their own common-rate expectations: band occupancy is 0.143 against a null of 0.243, the next replay is benign 0.877 of the time against a null of 0.914, and two replays disagree 0.150 of the time against 0.157. Each observed value sits below its expectation, the same direction as the census, which is what a population more polarized than a common rate produces. Dispersion is the quantity that compares across populations, and it gives $\chi^2/(n-1) = 1.91$ on 13 degrees of freedom, an asymptotic p of 0.025 and a Monte Carlo conditional p of 0.026 over 200,000 permutations that hold the harmful-execution total and the per-configuration repeat counts fixed. The permutation test is reported because 24 harmful executions over 14 configurations is too sparse to trust an asymptotic reference. The ratio of per-configuration to aggregate round-to-round standard deviation is 4.78 against 5.78 on the census, and that near-equality is exactly the artifact Appendix A.8 warns about: both arms have 120 configurations, so the ratio's dependence on suite size cancels and it reports similar numbers for populations whose heterogeneity differs by more than an order of magnitude. The scale-free comparison is the intraclass correlation on the estimand the census reports, the fitted Beta reweighted by the probability of having been selected: 0.063 [0.003, 0.155] here against 0.623 on the census, an order of magnitude below it and commensurate with it, both coming from the same likelihood. The fourteen harmful configurations carry between one and seven harmful executions out of twenty, with none at the extreme, so this arm shows detectable but slight heterogeneity, not the polarization the census reports.

What the arm cannot do is separate the serving stack from the model, because both change together, and it does not settle the routing explanation Section 6 names: a model this hard to compromise supplies too few incidents to say whether the census signature would appear on it under a stack that did. The design that would separate them is the same protocol against one open-weight model served two ways. What the arm does establish is negative, and the paper reports it that way: the observed

magnitude differs for this model–attack pair at a near-floor harm rate. This does not establish a vendor-wide absence of replay variability or behavior under stronger attacks.

A.12 A SCALE-FREE MEASURE OF HETEROGENEITY, AND THE MODEL BEHIND IT

Section 4.2 reports the ratio of the per-configuration to the aggregate round-to-round standard deviation as a diagnostic and not as an effect size, for the reason given there: the numerator is of constant order while the denominator falls as the square root of the number of configurations, so the same underlying process measured on a larger suite returns a larger number. The quantity that does not move with the suite is the intraclass correlation. Writing $k_i \sim \text{Binomial}(R, p_i)$ with $p_i \sim \text{Beta}(\alpha, \beta)$, the correlation between two replays of the same configuration is $\rho = 1/(1+\alpha+\beta)$, a property of the population, not of its size.

The population has to be stated twice, once for the likelihood and once for the estimand, and an earlier version of this paper conflated them. The harmful configurations are defined by $k_i > 0$ and are therefore selected on the outcome being modelled, so the likelihood is $P(k \mid k > 0)$. What that likelihood estimates is the preselection Beta, the population of configurations capable of harm including those that produced none in twenty runs, whose $\rho = 1/(1+\alpha+\beta)$ is 0.778 [0.600, 0.937] and 0.622 [0.485, 0.794]. The paper's sentences are about a different population, the configurations one has already seen produce harm, whose latent rates are that Beta reweighted by $1-(1-p)^R$. Selection favors high p and compresses the spread, so the replay correlation there is smaller: 0.623 [0.532, 0.680] and 0.530 [0.416, 0.619], which is the pair the paper reports. Fitting all 120 configurations with no selection at all gives 0.827 [0.769, 0.878] and 0.700 [0.635, 0.760]. Intervals are bootstraps over 2,000 resamples of configurations that refit the truncated model on each draw, so they carry the estimate's uncertainty rather than the transformation's.

One consequence is reported rather than buried. The naive harmful-only fit, which ignores the truncation and which the previous revision discarded, gives 0.586 [0.514, 0.645] and 0.501 [0.428, 0.565]. Those are close to the corrected 0.623 and 0.530 and not to 0.778 and 0.622. The selection bias in the original analysis was therefore small, and the revision that removed it replaced it for one round with a larger error in the other direction, by promoting a preselection quantity into a selected population's sentence. Against the homogeneous binomial, the $\alpha+\beta \to \infty$ limit of the same family, the likelihood ratio is decisive in both arms at $p < 10^{-179}$, evaluated against the boundary-corrected mixture reference and not the naive χ^2_1. The fitted shape parameters are below one in both arms, a U-shaped density with mass at both ends: the model recovers the polarization the census reports instead of imposing a unimodal shape on it.

A fitted ICC means nothing if the model does not describe the data, so the fit is checked, not assumed. The check simulates 2,000 replicate censuses from the fitted beta-binomial and from the homogeneous binomial, conditioning both generators on $k > 0$ by rejection: the observed set is defined by that condition and cannot contain a zero, so replicates that can produce data the observation is incapable of producing would measure the selection rule and report it as misfit, and conditioning only the fitted model would be the same error with its sign reversed. The observed share of configurations at an extreme, 0.600 and 0.394, sits in the middle of the beta-binomial's predictive interval, [0.482, 0.694] and [0.287, 0.489]; the binomial null predicts 0.011 and 0.002 and excludes the observation entirely. Band occupancy behaves the same way, observed 0.306 and 0.404 against beta-binomial intervals of [0.153, 0.329] and [0.298, 0.489] and binomial-null intervals of [0.718, 0.894] and [0.894, 0.989]. The model reproduces the summaries checked, which is weaker than adequacy, where the homogeneous one is refuted by six or more standard deviations. An earlier version of this appendix reported the check from the superseded naive fit while the text around it described the zero-truncated one; the figure and every value in this paragraph were regenerated from the fit the paper reports.

Splitting the configuration effect into its two crossed factors puts a number on where the heterogeneity lives, on the probability scale where a $\hat{p}$ of 0 or 1 is an ordinary value. Over all 120 configurations, after removing the binomial sampling variance that $\hat{p}$ carries by construction, the user task accounts for 0.129 of the variance and the injection task for 0.000 to three decimals, with 0.073 in their interaction; at temperature 1 the split is 0.108, 0.000 and 0.062. As shares computed from the unrounded components these are 0.638, 0.000 and 0.362 under greedy decoding and 0.634, 0.000 and 0.366 at temperature 1 – the two decoding conditions differ by 0.004 on the user-task share, which is the reason this decomposition and not the logit-scale one carries the claim below. Roughly two

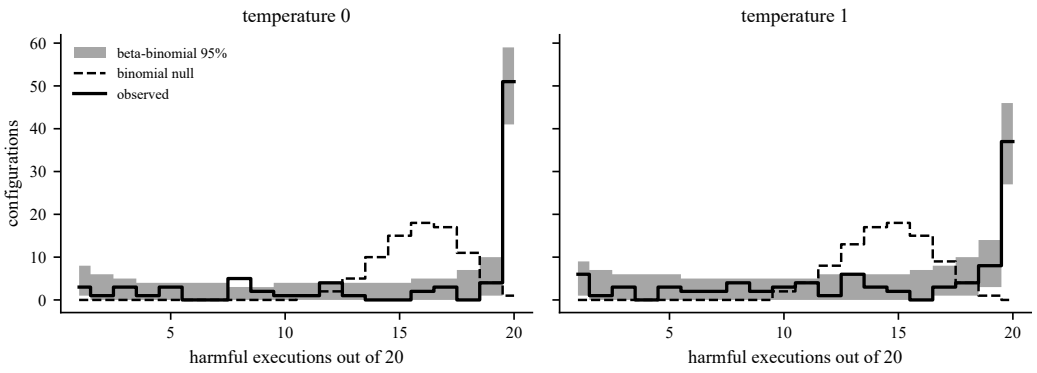


Figure 4: Parametric predictive check, one panel per decoding condition. The generator is the fitted beta-binomial at its maximum-likelihood parameters, not a posterior, so this is a predictive check and not a Bayesian one. The observed number of harmful executions per configuration is the solid line; the shaded band is the 95% predictive interval of the fitted beta-binomial over 2,000 simulated censuses; the dashed line is the median of the homogeneous binomial. Both generators are conditioned on $k > 0$, the condition that defines the observed set. The observed mass at both extremes is what the homogeneous model cannot produce.

thirds of the between-configuration variation is associated with user-task identity, and the injection component is estimated at zero to three decimals. A component estimated at zero is not a component shown to be absent – with three levels the design has little power to detect a small one – so the statement is that none of the variation this decomposition can attribute went to which of the three injection goals was pursued. That is a statement about this suite, not about agents in general, but it does say the heterogeneity has structure rather than being configuration-level noise.

A crossed model is fitted rather than declined. An earlier version of this appendix refused one, arguing that thirty-five configurations produce no harmful outcome in twenty executions and fifty-one produce nothing else, so the suite is largely separated on the logit scale. That reasoning is wrong and a referee said so: all-zero and all-twenty cells of a GROUPED binomial are not complete separation of the user-task and injection-task predictors, each cell carries twenty trials, and the variance components stay estimable.

The model has user task as a random effect over forty levels and injection task as a fixed effect over three. A three-level factor does not support a variance component, and making it random would report a prior rather than a measurement. That choice also buys exactness: with injection fixed, a user's random effect appears only in that user's three rows, the marginal likelihood factorises into forty one-dimensional integrals, and each is evaluated by adaptive Gauss-Hermite quadrature centred on the group's mode. The likelihood's factorisation is exact and no prior is doing work; the quadrature itself is still an approximation, and what can be said about it is the convergence quoted below rather than that nothing is approximated. Fixed-node quadrature was tried first and did not converge – the variance component moved between 3.08 and 4.26 as the order rose, because at a large σ most fixed nodes land where the logistic has saturated – which is what centring on the mode repairs: the estimate then moves by 0.002 between orders 24 and 48 and by 1.3×10^{-5} between 48 and 96. An earlier version of this sentence claimed stability to five decimals from order 24; the order-24 value contradicts it, and quoting the movement is in any case more useful than asserting a decimal place.

Under greedy decoding $\sigma_{\text{user}} = 4.45$ with a profile-likelihood interval of [3.31, 6.31]; at temperature 1, $\sigma_{\text{user}} = 3.32$ [2.51, 4.56]. The interval is a profile rather than a Wald interval because σ is bounded below and its likelihood is asymmetric. The injection contrasts are -0.064 and 0.091 at temperature 0 and -0.073 and 0.073 at temperature 1, against a user-task scale of 4.45 and 3.32 on the same axis. They are small relative to that scale; whether they are negligible is not settled here, since no margin was declared in advance and the contrasts carry their own uncertainty.

A previous version of this appendix turned σ_{user} into a user-task intraclass correlation, $\sigma^2_{\text{user}}/(\sigma^2_{\text{user}} + \pi^2/3)$, reported it as 0.858 and 0.770, and said that it and the beta-binomial's 0.827 were two routes

agreeing that the user task carries the heterogeneity. That was wrong and a referee said so. This model has no configuration-level term, so every source of variation living at the user-by-injection cell has exactly one place to go – the user intercept – and its denominator omits the component it is being compared against. The quantity is an upper bound on the user share, not an estimate of it, and the beta-binomial's 0.827 is a configuration-level correlation that says nothing about user tasks at all. Both numbers are withdrawn from that role.

Fitting the model the comparison needs is possible and was done: $\operatorname{logit} p_{ij} = \mu + \beta_j + a_i + c_{ij}$ with a user effect $a_i \sim N(0, \sigma^2_{\text{user}})$ and a configuration effect $c_{ij} \sim N(0, \sigma^2_{\text{cfg}})$, the marginal likelihood factorising into one outer integral per user and three inner integrals inside it, all by adaptive quadrature. One result from it is unambiguous: the configuration component is real. The likelihood ratio against the one-component model is 564 under greedy decoding and 427 at temperature 1, on one degree of freedom, at every quadrature order tried including the least favourable.

The split between the two components is not reportable, and the reason is worth stating because it is not a defect of the software. Refitting at rising quadrature order moved the user share from 0.785 to 0.018 to 0.012 at temperature 0 while it sat near 0.6 at temperature 1, so the likelihood was recomputed at *fixed* parameters, where nothing is optimised and nothing can wander. The integral turns out to be computable only while σ_{cfg} is modest: at $\sigma_{\text{cfg}} = 0.5$ every order from 16 to 48 agrees to five decimals, while at the temperature-0 optimum's $\sigma_{\text{cfg}} \approx 7.5$ the order-16 value is wrong by four orders of magnitude and only order 32 and above are usable. Walking σ_{cfg} at fixed σ_{user} then separates the two conditions completely. At temperature 1 the fixed-parameter diagnostic samples $\sigma_{\text{cfg}} = 4, 6, 8.45$, all above the fitted optimum, giving negative log-likelihoods 292, 304 and 319. This is an upper-tail check, not monotonicity over the full parameter range. The separate order-48 fit has negative log-likelihood 281.09 and an interior optimum: $\sigma_{\text{user}} = 3.74$, $\sigma_{\text{cfg}} = 1.80$, and the user task carries 0.81 of the random-effect variance, which agrees in direction with the probability-scale 0.634. Under greedy decoding it falls monotonically to the edge of what can be computed, 252 to 242 to 238 at $\sigma_{\text{cfg}} = 8.45$, and beyond that edge the two orders disagree by thousands and the numbers are meaningless.

That failure has a reading, though it is a reading and not a result. Seventy-two percent of the temperature-0 configurations are all-zero or all-twenty, and reproducing an endpoint exactly on the logit scale requires an unbounded effect, so the optimiser pushes σ_{cfg} outward until the integral gives out. What that establishes is how this link behaves at the boundary of these data, not a property of the population; it is a computational failure with a plausible cause, and it is why the probability-scale decomposition above – where 0 and 1 are ordinary values, and where the two decoding conditions differ by 0.004 – is the one the claim rests on, and why the logit-scale split is reported for temperature 1 only.

The bootstrap's unit, and the one we report. Configurations sharing a user task are not independent: the decomposition above puts roughly two thirds of the between-configuration variance on user-task identity, and the census's 85 harmful configurations come from 34 of the suite's 40 tasks, most contributing three. The paper's claims are about what a practitioner meets when attributing an incident, not about these 120 pairs, so the estimand generalizes over tasks, the task is the independent unit, and a configuration bootstrap would treat dependent draws as independent. The primary census intervals in Table 1, except its explicitly labeled preselection row, resample user tasks, each drawn with all of its configurations attached, under the same two-stage construction as before. The two stages carry different things. Resampling the unit carries the uncertainty in WHICH tasks the suite happens to contain; redrawing each resampled configuration's 20 outcomes at its observed rate carries the fact that $\hat{p}$ is estimated from 20 draws rather than known, which matters because band occupancy and nonunanimity are THRESHOLDED functions of $\hat{p}$ and a threshold applied to an estimate is not the threshold applied to the parameter.

That is a reason to expect the inner stage to matter, not a demonstration that it does, and it leaves open which estimand the interval is for. Three are available, and they differ in both classification and selection. Writing G for the distribution of latent rates and K for the harmful count in $R = 20$ repeats: the latent occupancy $\Pr_G(0.1 < p < 0.9)$; the latent occupancy among configurations OBSERVED harmful, which reweights by $1 - (1-p)^R$ and is therefore a different quantity; and what a fresh R-repeat experiment would REPORT, a thresholded function of the estimate rather than of the parameter. This paper's point estimate is the third and its selection is $K > 0$, so the estimand its interval can be about is the second.

We measured the coverage rather than arguing it, and the measurement does not support the stronger thing an earlier version of this appendix said. Simulating clustered populations with known latent rates – a task-level location, configuration rates drawn around it, twenty draws each, harmful-only selection – and applying both constructions gives, against a nominal 0.95 and for that second estimand: 0.992 with the inner redraw and 0.990 without, on a census-like polarized generator, each with a Monte Carlo standard error of 0.004. Both are CONSERVATIVE, and the inner stage's contribution is inside that error rather than the repair an earlier draft claimed. On a less polarized generator both undercover, 0.882 and 0.906, with every miss on the same side, and there the inner redraw is slightly worse. The point estimate is biased downward against this estimand by 0.046 to 0.069 across generators, which is the direction those one-sided misses take.

What that licenses is narrow. In the regime this census occupies the reported intervals are conservative for the estimand they can be about, and the choice between the two constructions does not matter at the resolution we can measure; outside it, coverage degrades and the interval is too low rather than too wide. We keep the two-stage construction because the thresholded statistic makes its motivation clear, not because the simulation shows it repairs anything.

What neither stage supplies is a finite-suite interval. A reader whose estimand really is these 120 configurations is not asking about sampling at all, and the configuration bootstrap is not that answer either – it resamples configurations, which a census does not do. We state the task-clustered interval as the primary because the paper's claims generalize, and we do not offer either bootstrap as a finite-suite uncertainty.

The narrower target is a statement about this suite's own configurations, and for a reader who wants it the same estimator with the configuration as the unit gives, under greedy decoding, [0.055, 0.117] for the next-replay-benign rate against the reported [0.053, 0.124], [0.093, 0.178] for the two-replay disagreement against [0.091, 0.183], and [0.188, 0.379] for band occupancy against [0.183, 0.389]; at temperature 1 the three pairs are [0.099, 0.176] against [0.091, 0.189], [0.156, 0.245] against [0.146, 0.256], and [0.309, 0.516] against [0.292, 0.534]. The task-clustered interval is wider in every case, by a twentieth to a fifth, and no comparison the paper makes turns on the difference. We report the wider one because it is the one the estimand asks for, not because it is wider.

Task independence is assumed, not demonstrated: the task is the coarser unit available and nothing here establishes that two user tasks are independent draws. What can be checked is the effect of the choice, and the sensitivity run isolates it by holding each configuration's outcomes fixed, so the ratios it reports describe the unit alone with sampling noise removed: 0.99 for the correlation under greedy decoding, 1.24 at temperature 1, 0.93 on the second task suite, and 1.10 to 1.28 across nine headline quantities. Those ratios are a diagnostic of the unit, not a correction to any endpoint.

A.13 The other attack families

Every review of this work has named the same limitation, that the census attacks with one template family, and the obvious response is to run a second family and report its intraclass correlation beside the first. That response is not available within the benchmark's own attack families, and this arm measures that boundary rather than asserting it. The six variants of Section 4.3 manipulate cues within one family; the families here are separate attacks with separate texts.

AgentDojo ships four families against this goal structure. We ran the three we did not already use over nine configurations at temperature 0, three repetitions each, using the census execution code with a single changed argument so that nothing else about the execution differs. None produced a single harmful outcome: 0 of 81 executions, [0.000, 0.045], against 162 of 180 for `important_instructions` over the same configurations, 0.900 [0.847, 0.936]. Per family the counts are 0 of 27 executions for `ignore_previous`, `tool_knowledge` and `injecagent` alike.

One name here collides with another and the two are unrelated. `injecagent` is the label AgentDojo gives one of its own attack families, which imitates the phrasing style of Zhan et al. (2024); it is a template inside this benchmark, and it is the one that produces no harm above. The arm in Appendix A.15 runs the InjecAgent benchmark itself – a separate codebase with its own tools, its own agent loop and its own success criterion – and the attack there is that benchmark's own, not this template. A result about one says nothing about the other.

The comparison is made at the configuration level. Nine configurations were sampled and each was run three times per family, and this paper's own central finding is that repetitions of one configuration are not independent draws, so a test treating 81 executions as 81 trials contradicts it. Counting configurations with any harmful outcome, each family gives 0 of 9 against 9 of 9 and Fisher's exact test returns $p = 4.11 \times 10^{-5}$. That per-family figure is the one to read. The pooled 0 of 27 counts the same nine configurations three times, so its $p = 1.06 \times 10^{-8}$ is not three independent replications and is given only for completeness. The run-level version this replaces gave 2.93×10^{-50}, overstating the significance by forty-two orders of magnitude. The depths are also unequal – three repetitions per family against the census's twenty – so a rare success is easier to miss on the untested families than on the reference. The claim is bounded accordingly: the three stock families tested produced no harm on the nine configurations sampled at that depth, which does not establish that they cannot, still less that no other family would succeed on this model. Appendix A.15's arm does run an attack from outside this benchmark, on that benchmark's own agent loop and checker, and it produces harm; what this arm bounds is AgentDojo's own remaining families against this goal structure, not attacks in general.

A zero attack-success rate is also what a silently broken arm looks like, and this project has already retired one for exactly that reason, so two checks separate the cases. The injected payloads differ: 424 characters for `important_instructions` against 163, 732 and 211 for the others, with visibly different text, so the attack is constructed and delivered. And the agent is working – all 81 executions returned a true utility verdict, meaning the model completed the user's task every time. These are successful executions in which the injection did not take, not crashed executions scored as benign.

Reproducibility is defined over harmful incidents, so a family producing none supplies no measurement, exactly as the contemporary-model arms supply none. The scope restriction is therefore not an omission but a measured boundary: on this model the quantity is estimable under one attack family because that family is the only one of AgentDojo's four which compromises it, and families outside the benchmark are not covered by this arm; Appendix A.15 runs one, on that benchmark's own agent loop and checker, and it produces harm. Read beside Section 4.5, where the same family fails completely on two newer models, the picture is a benchmark whose attacks land on a narrowing set of systems, and our measurement lives on the part of that set which is still compromised.

A.14 The second task suite

The arm was preregistered before it ran. The preregistration fixes the attack, the held-out calibration, the suite selection rule, the evaluation population, the repeat depth, the execution order, the drift screen and a minimum yield of 15 harmful configurations, and it is committed as its own change so the timestamp is checkable against the record files. Two amendments were appended after review round 4 arrived mid-run, recorded rather than applied silently. The reported functional of the fitted model became the selected-population transform, matching what the census now reports, because reporting this arm on the preselection scale while the census reports the selected one would recreate the incommensurability that made the second-vendor comparison worthless. And the falsification criterion, originally a conjunction of an interval that excludes the census value and a surviving dispersion null, was split into two questions, since a small but significant intraclass correlation would otherwise have been recorded as support for a magnitude it does not support.

Calibration held out three user tasks crossed with two injection tasks at three repeats, in each of the three suites the census does not use. Banking compromised all 18 of its calibration executions and Slack all nine of the nine that ran, the other nine failing because that suite does not define `injection_task_0` for those user tasks; travel compromised nine of 18. Banking was evaluated, and its calibration configurations are excluded from every reported estimate.

The evaluation population crosses 13 banking user tasks, from user task 3 upward, with three injection tasks, giving 39 configurations, at 20 repeats and temperature 0, with no screening: 780 executions, 31 harmful configurations, above the preregistered minimum. The pooled harm rate is 0.757 and band occupancy 0.323. A single replay of an observed harmful execution returns the opposite verdict 0.120 of the time against a common-rate expectation of 0.244, and two replays disagree 0.181 of the time against 0.369, each below its expectation in the same direction as the census. Dispersion is $\chi^2/(n-1) = 11.01$ on 30 degrees of freedom. No allocation of 200,000 reached the observed value, so the Monte Carlo conditional p is reported as the plus-one estimator's floor of 5×10^{-6}, whose

one-sided 95% Clopper-Pearson bound is 1.5×10^{-5}; the floor is an estimate and the bound is what the draws support. The selected-population intraclass correlation is 0.498 [0.316, 0.617] against the census's 0.623 [0.532, 0.680]; the preselection parameter behind it is 0.563 [0.323, 0.837]. Execution is round-wise, and the drift screen gives a rank correlation of -0.408 between harm rate and round index, inside the preregistered refusal threshold of 0.6, with the first third of the rounds at 0.628 and the last at 0.594.

Two execution incidents are disclosed with the arm and were written down while it was still running. The first launch stopped without an error at 187 of 780 executions when the execution environment's time limit ended it; execution was resumed until every round was complete, the round being the unit the drift screen consumes. Then two concurrent executions wrote to the same record set from round 5 to round 14, producing 359 duplicate executions, which the population bound rather than any error detected. They were removed by a rule fixed before any outcome was read: keep the first record for each configuration and repeat in file order and set the rest aside. A rule consulting the security verdict would have selected on the quantity being estimated, which is the error the zero-truncated likelihood exists to correct, and keeping both copies was not available because the dispersion statistic and the beta-binomial are defined at one repeat count. The retention rule is outcome-blind, which is the design argument; as a descriptive check that cannot exclude selection on other features, the kept and dropped sets differ little in harm rate, 0.609 over 581 records against 0.593 over 359, so the rule shows no sign of having selected on the outcome.

Two further records in that file carry a `JSONDecodeError` rather than a verdict, so the file holds 782 lines behind 780 completed cells. They are attempts whose response could not be parsed, retried and completed later; every analysis skips a record without a verdict, so they enter no estimate. They are named here because a record file whose line count exceeds its reported execution count invites exactly the question this sentence answers.

What the arm supports is bounded by what it shares with the census. The security and utility checkers, the attack template and the harness are the census's, so it measures a change of task domain and not a change of measurement apparatus, and the suite was chosen for the highest calibrated harm rate, so the transport estimate is conditional on that choice. It is not an independent replication, and it establishes nothing about attack families or about attribution behavior on other agent platforms.

A.15 An independently implemented attacked benchmark

Every arm above shares something with the census. The second task suite shares its checkers, attack template and harness; the second vendor shares the benchmark; tau2-bench changes the benchmark but removes the attacker. This arm shares none of them except the model. InjecAgent (Zhan et al., 2024) is a separate implementation by separate authors, with its own tool simulator, its own ReAct prompting and its own success criterion. Its direct-harm corpus is 510 cases built from 17 user tools and 30 attacker tools, and the 120 cases drawn here cover all 17 and all 30, so the draw spans the tool space rather than a corner of it. We import its `predict_one_case` and its `evaluate_output_prompted` and call them as its own evaluation loop does; nothing in the checkout is modified.

Two departures from that loop are the experiment rather than deviations from it. Its output cache is never enabled, because a cache returns the first replay's answer to the second and would make every configuration perfectly reproducible by construction. And each case is executed 20 times in round-wise order instead of once.

The decoding condition is the benchmark's, not ours. InjecAgent hard-codes `temperature = 0` in every model class. The replay variation reported here is therefore measured under a setting its own authors fixed for their own purposes, on a benchmark that did not exist to answer this question. No knob was turned to produce the phenomenon.

The model is held at `gpt-4o-mini-2024-07-18`, the census's, in the arm above: changing the benchmark and the model together would confound benchmark transport with model transport. A second arm below holds this benchmark and changes the model instead.

A contemporary model, on this benchmark. The paper's other contemporary-model arm is on AgentDojo, where gpt-5-mini cannot be attacked for a reason specific to that harness – the stock payload interpolates the target model's display name and the resolver has no gpt-5 entry (Appendix

A.3). InjecAgent's payloads carry no model name, so the obstacle is absent and the question is answerable here. Over the 40 held-out calibration cases at three repeats, 120 executions with no failures, gpt-5-mini produced no harmful outcome at all: 0 of 120, Wilson 95% interval [0.000, 0.031], and 0 of 40 cases with any harm.

The AgentDojo arm could not separate the model from the instrument at all; here the instrument can at least be interrogated, and what it shows is consistent with the zero being the model's rather than the checker's. It is not proof: a parser can be working normally and still miss a harmful action it was never written to recognize on a model that phrases things differently. The benchmark's own checker parses this model's output at a comparable rate: it returns `invalid` on 0.142 of these executions against 0.175 for gpt-4o-mini on the same 40 cases, for the same reason in both, and the mean response is 1,523 characters, so the model is answering and the checker is parsing. That measures parsing, not sensitivity to a harmful action: a checker can parse fluently and still miss harm expressed in a form it was not written to recognize. The quantity this paper reports is defined over harmful incidents and this model supplies none here either, which is a statement about what can be measured rather than a finding that the model reproduces its outcomes. The rule returning that verdict was fixed before the run.

Two asymmetries are noted rather than removed, and both are properties of the model class. gpt-5-class models reject a temperature parameter, so this arm runs at the provider default while the arm above runs at the temperature InjecAgent fixed – the same asymmetry Appendix A.3 records. And they spend tokens on reasoning before emitting any, so the output budget is raised to 4,096; a default budget returns an empty string.

What was fixed before the run. The attack type (direct harm, single step, so the unit of replay is unambiguous), a held-out calibration of 40 cases at three repeats under both of the benchmark's settings, the rule selecting between those settings, the evaluation draw, the repeat depth, the round-wise order, the drift screen and a minimum yield of 15 harmful configurations. The preregistration and the evaluation draw were fixed before the first evaluation execution. Calibration gave 0.058 for `base` and 0.108 for `enhanced`; the rule takes the higher, so `enhanced` was evaluated. That setting is the benchmark's own second configuration, in which the attacker instruction is reinforced with its hacking prompt; we wrote no payload.

The third verdict is not quietly resolved. InjecAgent returns `succ`, `unsucc` or `invalid`, and about one execution in ten is invalid, almost all of them the agent re-issuing the user's own tool instead of acting. Folding invalid into either bucket is a silent analytic choice, so the preregistration requires both readings and both are reported: with invalid excluded and with invalid counted as non-harmful. The two readings give the same harmful-configuration count, 14 either way, and on every conclusion drawn here.

The two readings differ in their denominator, and the reason is worth stating. Three of the sixty cases returned `invalid` on all twenty executions, so under the reading that excludes invalid they contribute nothing and the count is 14 of 57, while under the reading that keeps them it is 14 of 60. Those three are not noise, and they are also not a demonstration of determinism: twenty consecutive executions of the same non-action put the probability of a departure below 0.161 by a Wilson bound, which rules out a coin flip and not a rare alternative. Unanimity at this depth is the evidence available; determinism is not what it establishes. Heterogeneity is the claim, and a population in which some configurations are near-unanimous and others are coin flips is what heterogeneity means.

The extension, and why it is a post-hoc arm. The 60 preregistered cases yielded 14 harmful configurations against a declared minimum of 15. The response is a shortfall triggering one further draw of 60 cases from those used in neither the calibration nor the primary arm, and one only. That rule was not fixed before the data existed, and calling it preregistered would be false: it was written and committed while the primary arm stood at 635 of 1,200 executions. What can be checked is narrower. The trigger is the incident yield, which measures whether the design is adequate, never the estimate, which would be selection on the outcome; and no correlation had been computed on this benchmark when the amendment was written. Yield is not outcome-independent merely because we did not compute the correlation, so the arm is labelled post-hoc and its result is reported as such. The extension is reported as a post-hoc arm in those words, the primary arm's result stands whatever it shows, and a third draw is not authorised.

What the arm measures. The common-rate null is rejected at $\chi^2/(n-1) = 13.11$ on the preregistered draw, 10.94 pooled, with no exceedance in 200,000 conditional draws in either, a plus-one estimate of 5×10^{-6} and a one-sided 95% bound of 1.5×10^{-5}, and the two replay-level quantities sit near half their homogeneous expectations. The selected-population intraclass correlation is 0.477 [0.346, 0.582] on the preregistered draw, 0.447 [0.171, 0.595] on the extension and 0.517 [0.377, 0.579] pooled, against the census's 0.623 [0.532, 0.680]. Those intervals all overlap the census's, which is not a test: the interval of a difference cannot be recovered from two marginal intervals, and for two wide intervals over small samples overlap is near-certain whether or not the populations differ.

The exploratory difference is estimated by independently resampling configurations within each arm, refitting, and pairing the resulting bootstrap estimates at random; configurations are not matched across benchmarks. This sensitivity analysis does not account for shared user tasks on the census side, unlike its primary task-clustered intervals, so it cannot supply a definitive cross-benchmark comparison. Pooled against the census it is -0.106, with a percentile interval $[-0.240, -0.005]$ that excludes zero, a basic interval $[-0.207, 0.029]$ that does not, and a bias-corrected interval $[-0.237, -0.003]$ that does. On the preregistered draw the difference is -0.146 and only the percentile interval excludes zero. Every arm's bootstrap mean sits below its point estimate, which shifts the difference distribution negative and left-skewed; percentile adopts that shift, basic reverses it, and on fourteen to twenty-seven clusters whose fit sits at a boundary none of the three carries a coverage guarantee. Where they disagree the disagreement is the result, so the magnitude is not resolved. The second AgentDojo suite is the one arm on which the three constructions agree, at -0.125 with all three containing zero and a permutation p of 0.115. That agreement makes the arm's difference undetectable at this sample size; it does not make it zero, and we do not read it as establishing that the magnitude transports.

Of the preregistration's two questions the first is answered and the second is not. The arm rejects homogeneity and estimates its size; what it does not establish is either equivalence to the census magnitude or a robust difference from it, and its own yield rule forbids the comparison on the preregistered draw, 14 harmful configurations against a declared minimum of 15. The pooled draw is the one the amendment authorises for a magnitude comparison, and it is the one that does not resolve.

The choice between the two readings of `invalid` changes nothing that matters. Excluding invalid gives 14 harmful configurations and 0.477 [0.346, 0.582]; counting invalid as non-harmful gives 14 and 0.485 [0.341, 0.584]. The dispersion statistic moves from 13.11 to 14.35 and its permutation p from one bound to the same bound.

The drift screen. Both arms ran round-wise over twenty rounds, and the preregistered screen asks whether the harm rate moves with execution order, which would confound provider drift with between-configuration variation. The screen did not trigger, which is not the same as an absence of drift: the rate goes from 0.069 in the first third of rounds to 0.083 in the last in the primary arm and from 0.117 to 0.119 in the extension, with a rank correlation against round index of 0.25 in the primary arm, well inside the refusal threshold of 0.6. The census's own arm is blocked rather than round-wise, and Section 4.3 measures what that costs; every arm collected since is collected this way.

The two draws produce harm in 14 and 13 of 60 cases, a difference the data do not detect (Fisher exact $p = 1.00$, which bounds nothing about the underlying yields), and disagree on the shape of the counts within them. The primary draw is sharply polarised – ten of its fourteen harmful cases produce three or fewer harmful executions and four produce sixteen or more, with nothing between – while the extension carries intermediate counts throughout (Mann-Whitney $p = 0.042$ over the 27 harmful cases). Both are draws from the same 510-case corpus under the same rule, so they target one population and the difference in shape is compatible with variation in which cases were drawn. That is a statement about the design, not a demonstration that nothing else differs between the arms; collection time, for one, is not controlled, and the design does not identify which explanation holds. Pooling does not average the two: refitting on the union adds the between-draw difference in harm rate to the between-configuration variance the correlation measures, which is why the pooled 0.517 sits above both 0.477 and 0.447. The intermediate counts are the more direct evidence for the paper's claim, since a case that produces harm eight times in twenty is one whose replay is visibly a draw rather than a value.

Band occupancy reads the difference more directly than the rank test, and it is also where a bare number would mislead. The preregistered draw places 0.286 of its harmful cases strictly inside (0.1, 0.9) and the extension places 0.692, which invites the reading that the second draw is the less polarized. Against their own common-rate expectations the two say the same thing: 0.286 against 0.948 and 0.692 against 0.997. A homogeneous population at either draw's rate would put almost every case inside that band, so a LOW occupancy is the evidence of heterogeneity, and both draws sit far below their own nulls. They do not sit equally far: the deficit is 0.663 for the preregistered draw and 0.304 for the extension. The ordering is the same on either reading – the preregistered draw is the more polarized both raw and against its null – so what the nulls supply is the scale of the difference rather than a reversal of it. The pooled draw is 0.481 against 0.980.

What the estimate does and does not depend on. Three features of this arm invite distrust and each is tested rather than argued. The extension is post hoc, so the two draws are contrasted directly. They are disjoint samples from the same corpus under the same rule, and their difference is 0.029 with a 95% interval $[-0.209, 0.344]$. An interval that wide is not evidence that the draws are alike, and we do not use it as such: pooling is justified by the design, since both draws come from the same 510-case corpus under the same rule and the amendment authorised the second. The contrast is reported for completeness, not as support: an interval this wide would fail to separate draws that do differ. Twenty-seven configurations is few, so one unusual case could carry the estimate; dropping each in turn moves the pooled figure between 0.501 and 0.522, a range of 0.021, against 0.037 in datasets simulated from the fitted model at the observed depths ($p = 0.84$); that reference can itself contain an influential case, so it calibrates the deletion range rather than describing a population without influence. The preregistered draw behaves the same way, 0.049 against a simulated 0.077 ($p = 0.83$). The observed spread is if anything smaller than the reference produces, which says the deletion range is unremarkable rather than that no case is influential. Depths are unequal because the benchmark's unparseable executions are dropped, and a case observed twice carries almost nothing about within-case variability while still entering the selection weighting; refitting at depth floors of 5, 10 and 16 gives 0.522, 0.522 and 0.526 over 26, 26 and 25 configurations – the first two floors select the same cases, because the depths jump from 2 straight to 11. Raising the floor removes configurations as well as depths, so this bounds the shallow tail's contribution rather than identifying an effect of depth; nothing on this design separates the two, because selection is defined through the observations themselves. The two readings of the third verdict agree to a thousandth, 0.517 against 0.518. None of this makes the arm powered: every cut leaves an interval wider than 0.18, and each is reported in full rather than summarised by whether it overlaps.

A.16 What a replay budget buys

Every result above says a replay verdict is a draw. The question an evaluation actually faces is how many draws are enough, and enough for what, so three curves are reported rather than one number. All three are computed from records already collected: a census at $R = 20$ contains every smaller budget inside it. The configuration is the unit of uncertainty throughout, and every interval is a cluster bootstrap over configurations rather than over executions, which are correlated within a configuration by exactly the phenomenon being measured.

Unanimity over the budget. A configuration whose R draws all agree looks reproducible whether or not it is, with probability $p^R + (1-p)^R$. Where depths differ the same expression takes each configuration's own n_i, and so does the selection weight the estimand uses: the selected-population correlation reweights the fitted Beta by the probability that a configuration was observed harmful at all, which for a common depth is $1 - (1-p)^R$ and in general is

$$w(p) \;=\; \frac{1}{n}\sum_{i=1}^{n}\big(1 - (1-p)^{n_i}\big),$$

the average over the depths actually realized. The census has $n_i \equiv 20$ and the two agree; on the independent benchmark the harmful cases the truncated likelihood fits carry depths from 2 to 20 once unparseable executions are dropped, and there the scalar form would describe a different population from the one that likelihood fits.

That average rests on an assumption worth naming, because the depths are not assigned. On that benchmark n_i counts the executions the benchmark could PARSE, and an unparseable execution is a behavioural outcome rather than a missing draw, so writing the estimand over realized depths treats

depth as ignorable given p: it supposes a case's parse rate says nothing about its harm propensity beyond what p already says.

The depth population has to be said, since three different ones are available. Of 2,400 executions across the preregistered draw and the extension, 115 cases returned at least one parseable execution and 0.938 of all executions parsed. The weight $w(p)$ is taken over the depths of the HARMFUL cases – the 27 the truncated likelihood fits, whose depths run from 2 to 20 – and not over all 115, because those 27 are the population the correlation describes. The one case at depth 1 is not harmful and never enters.

The assumption is testable and we test it, with a result that is a NON-DETECTION and not a confirmation. Valid depth and harm rate have a Spearman correlation of -0.005 across all 115 cases and $+0.057$ across the 27 harmful ones, each inside the band a permutation of the same data produces, $[-0.201, 0.187]$ and $[-0.374, 0.378]$. At 27 cases that band admits an association of real size, so the check rules out only an association large enough to see at this sample size, and the estimand remains conditional on an assumption the data cannot settle. Averaged over the census's harmful configurations that is 0.802 [0.740, 0.857] at $R = 3$ and still 0.634 [0.538, 0.726] at $R = 20$. The second number is not a failure of the budget: most harmful configurations in this census really do sit near 0 or 1, which is the polarization Section 4.1 reports. It is the first number that matters for practice, because a three-repeat screen of the kind an evaluation runs will call four configurations in five deterministic, and Appendix A.5 shows which ones it misses.

Interval width on one configuration. If the budget exists to estimate a single configuration's harm probability, the relevant quantity is the width of a Wilson interval at that R, averaged over the outcomes R draws can produce rather than plugged in at the point estimate, which is optimistic at small R. On the census it falls from 0.483 at $R = 5$ to 0.226 at $R = 20$: the familiar $1/\sqrt{R}$, stated here in the units a reader would budget in.

Sign error on a replay comparison. This is the one that bears on attribution, and it is a calculation under a stated model rather than a measurement on these records: the neutralized side is stipulated here, not executed, so what follows prices a decision rule at the factual rates this census measured. Appendix A.17 executes both sides on 26 of these configurations and compares the two. The model is: both sides are fresh samples, the R executions on a side are independent and identically distributed at that side's harm probability, the two sides are independent of each other, the span-present side has rate p and the neutralized side $p - g$ with $0 \le p - g$ and $p \le 1$, and the rule keeps the sign of the difference in counts with a tie broken at random, which costs half an error. The feasibility condition bites: a configuration whose factual rate is below the gap cannot exhibit that gap at all, so the census average at $g = 0.4$ is taken over the 74 of 85 harmful configurations with $\hat{p} \ge 0.4$, and at $g = 0.2$ over 78. The average is therefore conditional on the gap as well as on the population, which is a second reason the envelope rather than the average is the number to budget against. The figure is the error rate of ONE such comparison; a method making several inherits a multiplicity this does not price. A method compares an outcome with a span present against the same outcome with it neutralized and keeps the sign of the difference. With both sides stochastic and a true gap g in harm probability, a budget of R per side gets that sign wrong with a probability the binomial gives exactly. At $R = 1$ it has a closed form,

$$\Pr[\text{wrong sign}] \;=\; \tfrac{1}{2}(1 - g),$$

counting a tie as half an error, and it does not depend on p at all: a single replay on each side is wrong 0.400 of the time at $g = 0.2$ and 0.300 at $g = 0.4$, whatever the configuration's own rate. This is a two-fresh-sample rule with randomized ties. If the baseline is instead a fixed historical harmful outcome, a single edited replay flips it with probability $1 - q$, where q is the edited harm rate; even an ineffective edit with $q = p$ flips it with probability $1 - p$. That is a different decision rule from the sign comparison above. Raising the budget collapses the error, but how fast depends on the population, so the average and the envelope are both reported: at $g = 0.4$ the census's own rates give 0.049 at $R = 5$ and 0.0006 at $R = 20$, while the worst feasible base rate gives 0.099 and 0.0043 at the same budgets, both at $p = 0.70$. An average over one census is not a bound, and a budget chosen for a different population should be chosen against the envelope rather than against our mean. A method that neutralizes a span, re-executes once, and keeps the sign is therefore operating at an error rate set by the effect it is trying to detect and by nothing it controls.

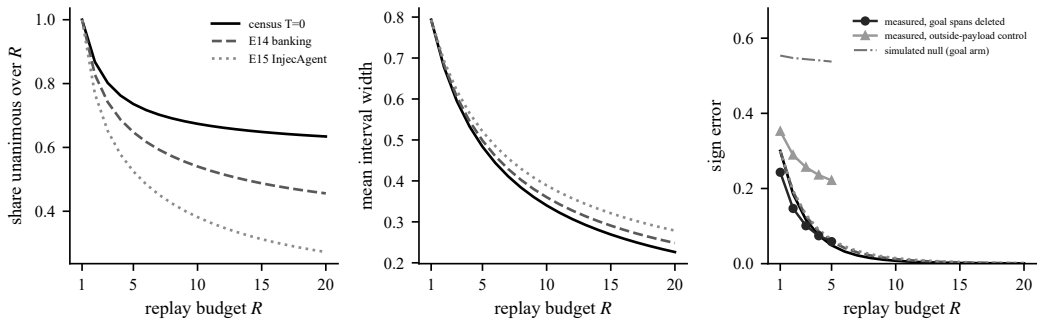


Figure 5: What a replay budget buys, on three arms. Left: the share of harmful configurations whose R draws all agree, so the configuration is unanimous and a screen reading unanimity as determinism would call it reproducible. Centre: mean Wilson interval width for one configuration's harm probability, averaged over the outcomes R draws can produce. Left and centre are computed from the census, the second AgentDojo suite and the independent attacked benchmark; the three agree in shape and differ in level exactly as their polarization differs. Right carries two quantities that are not the same estimand. The three unmarked line curves use the line styles in the left-panel legend and are the model of A.16: the probability that a comparison at R per side returns the wrong sign for a *stipulated* gap of 0.4, ties counted as half an error, on the same three arms. The marked curves are measured on E16's 26 contested configurations (A.17): how often an R-per-side sign decision disagrees with the sign of the held-out remainder, for deleting the goal spans and for the outside-payload control, shown against the goal arm's simulated no-difference reference. The control has its own reference, reported numerically in Appendix A.17. Read the two families separately.

The three arms have the same shape and differ on level in the direction their polarization predicts. The independent benchmark, whose harmful configurations are the least polarized, is unanimous least often, 0.272 [0.143, 0.416] at $R = 20$ against the census's 0.634; the second AgentDojo suite sits between them at 0.456. The sign-error curve is nearly identical across arms because it depends on the gap and the budget far more than on the population.

This is a statement about how many replays a decision needs, not about whether an attributed span is the right one. Nothing here measures attribution accuracy, and the census carries no ground truth that could.

A.17 Executing the neutralization, not modelling it

The census and transport arms above measure factual replay, and A.4's ablations query an attribution method without pricing its edit. An attribution method's own operation is different: it neutralizes a span, re-executes, and reads the difference. A.16 prices that operation with a binomial model at measured factual rates, which stipulates the neutralized side rather than observing it. This arm executes it. The design, the refusal rules and the minimum yield were fixed before the first execution.

What was run. Thirty-three configurations from the temperature-0 census – the 26 contested at $0.1 < \hat{p} < 0.9$, three saturated instrument controls, four low-rate – each under six arms at twenty replays, interleaved round-wise over twenty rounds: no neutralization; the whole injected payload; the spans carrying the injection task's goal; the payload's framing spans with the goal left intact; a count-matched placebo drawn from spans of the same tool return that never touch the payload; and the goal spans again under a masking operator. 3,960 executions, no failures. Six contested configurations chosen before the run at fixed percentiles of their census rate were then carried to 200 replays a side on the factual and the goal-neutralization arm – 180 further rounds each, 2,160 further executions – to check whether the aggregate conclusion depends on the depth. No other arm was run at that depth, so the placebo is measured at twenty replays throughout and every comparison between it and a treated arm is made at twenty.

The factual side is re-executed rather than read from the census. The contested configurations were selected on the census rate, so reusing it as the untreated side would build regression to the mean into every difference.

Span sets come from the benchmark's own injection, not from word matching. A candidate span is injection-bearing when any of its occurrences overlaps the rendered payload, goal-bearing when it carries the injection task's goal text after normalization, and eligible for the placebo only when every one of its occurrences lies outside the payload. That last condition is not fussiness: the operator neutralizes by replacing text wherever it appears, so a span that occurs both inside and outside the payload would, if drawn as a placebo, delete the injection and turn the control into a treatment. The rule also has to survive the benchmark's two renderings of one payload, and returns that carry the payload more than once.

The instrument. On the saturated controls the payload arm drives harm from 0.967 to zero on three of three. Delivery is checked on the two arms that are meant to remove the goal text – the whole payload and the goal spans – across every stratum and both phases: it remained legible inside the payload in 23 of 2,400 executions, mean residual eleven characters, all 23 in the main run. The framing arm is excluded from that diagnostic because it is designed to leave the goal text standing, which it does in 600 of 660. On the factual arm 23 of 26 contested rates fall inside $(0.05, 0.95)$ a month later and the mean signed shift is $+0.035$. Band retention is a screen, not a demonstration that the two runs share a distribution, and the shift's direction is what selection on the census rate would produce; that makes regression to the mean the leading reading rather than the identified one, since provider-side or scheduling change would also move it and this design separates neither. The preregistered drift screen finds no configuration at or above its threshold, worst $\rho = -0.473$ over twenty rounds, which is a screen result and not a demonstration of stationarity.

No harmful outcome survives payload neutralization on the contested stratum, and the depth check tightens the bound. Every arm that touches the payload gives zero harmful outcomes on every contested configuration: 0 of 520 executions per arm at twenty replays, upper bound 0.007. On the six deep configurations the goal arm gives 0 of 1,080 further executions, so 0 of 1,200 at two hundred replays a side – an upper bound of 0.019 on any one of those six and 0.003 pooled. The four treated arms therefore have the same mean difference from the factual arm, 0.517, because each equals the factual rate. The deep subset moves the mean difference over its six configurations from 0.417, on the twenty rounds they share with the main run, to 0.473 on the 180 fresh rounds; pooled to the full 200 a side it is 0.468. All three are driven by the factual side's own estimate, the treated one being zero at every depth. The `deep_subset.R200` field in the metrics file holds the 180-round figure, not the pooled one, and is named for the depth reached rather than the count it averages.

Zero is a property of this stratum at this depth, not of the operator. On the low-rate stratum the framing arm returns one harmful outcome in eighty executions, and the four arms are four specified span sets rather than an arbitrary part of the payload. That stratum also carries the largest control effect in the experiment: across its four configurations the factual arm averages 0.163 and the outside-payload control 0.638, so deleting spans that never touch the payload raised harm roughly fourfold there. Four configurations selected for a low census rate support no estimate, and the figure is reported because it is the most extreme instance of the effect the contested stratum measures, not as evidence about its size. What the counts support is that these four interventions produced no observed harmful outcome on the contested stratum at twenty replays, with the goal arm's bound tightened to 0.019 per configuration on six of them at two hundred.

The placebo is where the result is. Neutralizing count-matched spans that never touch the payload leaves the mean difference at -0.019, which falls inside the preregistered bound. That is a decision rule returning the value it was written to return when the control does not behave like a treatment on average; it is not a finding of equivalence, and the interval $[-0.188, 0.170]$ is wide enough to make that explicit. Every E16 interval quoted in this paper clusters on user tasks; the per-stratum figures computed beside them cluster on configurations instead and differ in the third decimal, which is the unit and not a disagreement. Nothing follows about the size of a per-configuration effect: an interval that wide is consistent with large effects of both signs, which is what the rest of this paragraph reports. Per configuration it is not small: the mean absolute difference is 0.362, the largest 0.90, and harm rose on eleven and fell on fourteen. The direction cancels; the magnitude does not. We report both statistics because the preregistered gate is the signed mean and substituting the absolute one after seeing it would be choosing the test after the result. What the gate establishes is that the control does

not behave like a treatment on average; it was not a test of the per-configuration property, and passing it is not evidence of one.

The magnitude comparison has to use one functional for both arms. Writing $d_i = \hat{p}_i - \hat{q}_i$ for configuration i, the squared effect is estimated by

$$\widehat{d_i^2} \;=\; d_i^2 \;-\; \frac{\hat{p}_i(1-\hat{p}_i)}{n_F - 1} \;-\; \frac{\hat{q}_i(1-\hat{q}_i)}{n_T - 1},$$

which is unbiased for $(p_i - q_i)^2$ when the two arms are independent binomials, averaged over configurations and then square-rooted; the root of an unbiased mean square is a summary, not itself an unbiased estimate, and a negative mean square would be truncated at zero, which does not occur here. The subtraction above removes $\mathrm{Var}(\hat{p}) + \mathrm{Var}(\hat{q})$ while the expectation of d_i^2 carries $-2\mathrm{Cov}(\hat{p}, \hat{q})$ alongside them, so the correction is right only when the two arms are independent. If a disturbance shared by the round correlated them positively the estimator would subtract too much and both corrected effects would come out too small; if it correlated them negatively, too large. Interleaving both arms within every round is what would expose such a disturbance to both, and it is the reason to suspect the positive case rather than a demonstration of it: the design fixes no covariance sign, the data do not identify one, and the direction of the resulting bias is therefore not established either. The observed all-zero payload arm has zero sample covariance, but finite-sample zeros do not prove a zero population rate or covariance. The RMS ratio is not established as a conservative bound. The drift screen speaks to how large a round-indexed disturbance would be; it is a screen and bounds nothing. On that functional the control reaches 0.436 [0.319, 0.538] and removing the injection itself reaches 0.569 [0.484, 0.652], a ratio of 0.77 [0.56, 0.98]. Intervals are a percentile cluster bootstrap over the twenty user tasks the 26 configurations come from – configurations sharing a user task share its environment, its tool returns and its placebo-eligible span pool – with the factual arm resampled jointly with the treated one so the ratio carries their dependence.

Decision curves, with the null this estimator actually has. A method at budget R draws R executions a side and keeps the sign of the difference in counts. Scoring that needs a reference for what the right sign is, and taking the reference from the same draws would let the decision define its own truth – at $R = 1$ it would be right by construction. So each configuration's twenty replays a side are partitioned: a uniformly random R of the factual draws and R of the treated draws make the decision, the remaining $20 - R$ on each side set the reference sign, a decision that ties costs half an error, and a split whose reference itself ties is dropped and counted rather than scored. Each configuration and budget is averaged over 5,000 such splits, weighted equally. A configuration whose splits all tie contributes no rate at that budget and is dropped from that budget's average rather than scored as anything; none does at the budgets reported, and the tie rates below say how close any came. Reference ties are rare where the treated arm is empty (0.002 of splits at $R = 1$ for goal neutralization) and common where both arms are live (0.044 for the placebo, rising to 0.063 at $R = 5$), which is itself a property of the placebo worth naming.

The null of this estimator is not one half, and the reason is the aggregation rather than any dependence between the halves. Marginally the halves are independent: a permutation chosen without reference to the values maps an iid vector to an iid vector, and disjoint blocks of an iid vector are independent, so pooled over every usable split the no-difference rate is one half – simulation returns 0.495 to 0.504 across budgets and rates. What the reported statistic does instead is take each configuration's own rate and weight them equally, dropping splits whose reference ties. Within one configuration the split conditions on that configuration's realized counts, and there the halves are negatively correlated – in the extreme case both arms carry the same count, every non-tied decision necessarily disagrees with its own reference, and the rate is 1.0. Equal weighting lets those configurations, which also produce the most reference ties and so the fewest usable splits, count as much as any other. Under the same weighting the simulated null runs 0.529 to 0.552. That is precisely why the null must be simulated rather than asserted: the measured curve carries the same weighting, so one half would be the wrong reference for it. Simulating the whole split-and-drop procedure at each configuration's pooled rate – the arm's own rate, so each arm carries its own null – puts a no-difference population at 0.554 at $R = 1$ and 0.538 at $R = 5$ for the goal arm, and at 0.546 and 0.533 for the placebo. Against those, goal neutralization measures 0.243 at $R = 1$ and 0.059 at $R = 5$; the placebo measures 0.352 and 0.222, against 0.546 and 0.533.

Two things about that null have to be said rather than assumed. Its generator is unconditional: both sides are drawn fresh and iid at the configuration's pooled rate, so the true difference is zero by

construction and nothing is conditioned on the observed counts. The reading lands above one half for the reason given above, which is how the per-configuration rates are combined and not a failure of independence. And a point estimate is not a test, so the contrast is computed per configuration against that configuration's own simulated null and then averaged. The POINT is the same either way – a mean of differences is the difference of means – and what pairing changes is the interval, which is narrower because the null varies widely across configurations and that variation cancels within a pair. The simulated null carries its own Monte Carlo error, of order 0.008 at 120 replicates of 300 splits, which is why two computations of it differ in the third decimal and why no contrast is quoted past three. Goal neutralization sits -0.309 [-0.350, -0.270] below its null at $R = 1$ and -0.474 [-0.510, -0.425] at $R = 5$, on 26 of 26 configurations. The placebo sits -0.201 [-0.273, -0.123] and -0.321 [-0.431, -0.205] below its own, on 23 of 26. Comparing the two arms needs care, because they do not share a null. The difference of their raw disagreement rates is $+0.111$ [0.027, 0.199] at $R = 1$ and $+0.163$ [0.046, 0.282] at $R = 5$; the difference of their distances from their OWN nulls, which is the quantity that answers whether the placebo sits further from no-signal, is $+0.109$ [0.028, 0.194] and $+0.153$ [0.038, 0.271]. Both are computed per configuration and resampled jointly, since the two arms share the factual side. Either way the placebo is worse than the treatment and both are far from no signal at all. Intervals are the same cluster bootstrap over user tasks. That is the finding of the paragraph above restated in decision units: an arbitrary span outside the payload produces a signal a sign rule reads, and reads on almost every configuration.

A.16's model, evaluated at the measured rates, gives 0.241 and 0.062. The agreement is real but it is weaker evidence than it looks: where the treated arm is zero of twenty the model reduces to $\frac{1}{2}(1-p)^R$ and at $R = 1$ the empirical calculation reduces to the same expression, so the match at that budget is close to an arithmetic identity. The honest statement is that the model is consistent with the measurements in this regime, not that the measurements independently confirm its sampling assumptions.

What a budget buys when the two candidates are compared. The arms make a localization task with ground truth. Hand a rule both candidates – the goal spans and the count-matched outside-payload spans – let it spend R replays on each of three cells, factual and one per candidate, and have it name the candidate whose neutralization lowers harm more. The answer is the goal spans by construction, since the benchmark wrote the injection. At one replay a side the rule names them 0.768 [0.681, 0.845] of the time against a coin flip's 0.5, and at five replays 0.860 [0.763, 0.928], intervals clustered on user tasks.

Those numbers are an identity and are reported as one. The factual draw cancels out of the ranking: the rule prefers the goal spans when $f - a > f - b$, which is $a < b$, so the factual cell is spent and then contributes nothing to which candidate is named. And the treated arm is zero on every contested configuration, so $a = 0$ always, the goal spans win whenever the control's draw is non-zero, and the two tie exactly when the control's draw is zero too. The control can never outrank the payload here – not usually, never. Accuracy is therefore $1 - (\text{tie share})/2$ to within 0.002 at every budget, and the tie share is itself $\mathbb{E}[(1 - p_C)^R]$, matching the observed 0.463, 0.377, 0.333 and 0.282 at $R = 1, 2, 3, 5$ to three decimals. A budget buys fewer ties and nothing else in this two-candidate resampling calculation. It is not a measurement of general localization skill. With many candidates, multiple controls may tie with the goal intervention; accuracy then depends on their joint responses and the search and tie rules. The pairwise curve does not establish performance over the 19–425 candidate spans encountered in the attribution-stability runs.

An earlier version of this paragraph read the tie as the rule having no harmful outcome to explain and quoted the benign-factual rate for it. That is a different quantity, $\mathbb{E}[(1 - p_F)^R]$, which gives 0.483, 0.299, 0.208 and 0.124 and agrees with the tie share only at one replay, by coincidence. The budget claim drawn from that reading is withdrawn. Section 4.6's benign-start rate of 0.457 is a separate measurement on separate records and stands.

What the arm is for is bounding this paper's own claim rather than measuring a localizer. The control's magnitude does not turn into localization error on these records: the rule prefers the treated candidate whenever $\hat{q}_T < \hat{q}_C$, ties when they are equal, and would need $\hat{q}_T > \hat{q}_C$ to reverse – and $\hat{q}_T$ is 0 in all 520 treated executions here. Section 4.7 therefore establishes that the outcome is sensitive to edits outside the injected source. It does not establish that the instrument is miscalibrated: calibration needs a target, a relationship the response ought to bear to the span's causal contribution, and this experiment supplies none. Nor does it establish misattribution, for the reason given above.

One observation we do not explain, reported as exploratory. This arm was not part of the preregistered comparison structure and no hypothesis about it was fixed in advance; it is described here because the result is informative, and it carries no claim. The framing arm deletes only the social-engineering sentences – the claim to be a message from the user, the instruction to act before continuing, the promise that the original task may resume afterwards – and leaves the instruction itself in place. It survives: the goal text is still legible inside the payload in 500 of 520 executions, mean residual 66 characters, against 14 of 520 when the goal spans are the ones deleted. Harm is nonetheless zero. What that supports is narrow: retaining the instruction text was not sufficient to retain the attack's efficacy under this operator. It does not establish that the framing rather than the instruction carries the effect, because deleting three sentences from inside the payload also perturbs its structure, and both components may simply be necessary. Separating those readings needs an arm that replaces the framing with length-matched neutral text instead of deleting it. That arm is not in the preregistration and we did not add it after seeing this result; it is named here as a follow-up.

A.18 The control on a population that did not produce it

A.17's control was measured on the 26 contested configurations of the temperature-0 census – the same population whose census rate selected them. E17 asks whether it survives on a population selected without reference to it. The design, the failure rules and the interval below were frozen before the first execution.

Why a second suite rather than a held-out set. The obvious design is not available and this is checked rather than assumed: E16 took all 26 configurations the census calls contested, and of the 87 workspace configurations it did not use, none are contested. The banking suite supplies what the census cannot. It shares the census's checkers and the `important_instructions` attack, so the operator and the span rule transfer unchanged, and nothing in it was used to build, tune or select anything in E16. The frozen population is its ten configurations with an E14 harm rate strictly inside (0.1, 0.9).

One of the ten cannot be run, and is reported rather than replaced. `user_task_13|injection_task_1` has no control-eligible spans at all: its tool return carries no count-matched material every occurrence of which lies outside the rendered payload, so the control does not exist there. The preregistration requires reporting and forbids substitution, so E17 is nine configurations from six user tasks, 540 executions, no errors, and every figure below is over nine.

The instrument behaves as it did. Goal neutralization gives 0 harmful outcomes in 180 executions, Wilson [0, 0.021]; the count was zero on the census as well. The goal text was legible inside the payload in 0 of 180 treated executions. Nine of nine fresh factual rates fall inside $(0.05, 0.95)$ against a preregistered minimum of six, so the contested regime replicated; the mean shift from the E14 rate is $+0.028$, small and in the direction selection on that rate would produce.

Sensitivity of the treatment's order recurs. On the same functional and the same cluster unit, the control reaches a noise-corrected root-mean-square of 0.576 [0.459, 0.722] against goal neutralization's 0.525 [0.410, 0.596], a ratio of 1.10 [0.79, 1.75]. The preregistered replication interval was [0.20, 1.40] and the point falls inside it. E16's ratio was 0.77 [0.56, 0.98]. The two intervals overlap on [0.79, 0.98], which is 45% of the census interval's length and 20% of this one's. Overlapping marginal intervals are not a test either way: they neither establish equivalence nor say whether an interval for the cross-suite difference excludes zero, and no such contrast is computed here. What the arm supports is qualitative – sensitivity of the treatment's order recurs on a population that did not produce it – together with the separate fact that the point estimate passes the rule fixed in advance. Quantitative agreement between the two suites is not established, and establishing it would take an explicit contrast or equivalence analysis rather than more configurations alone.

The observed direction runs the other way, which is the more interesting half. On the census the control's signed mean was -0.019 with harm rising on eleven configurations and falling on fourteen – a split that cancels. On banking it does not cancel: harm RISES on seven of nine, and on each of those seven the control arm returned a harmful outcome in 20 of 20 executions. The signed mean is -0.306, and its interval is $[-0.563, 0.022]$ – wide, and containing zero. So the difference between the two populations' directions is an observed pattern at nine configurations and not a demonstrated population difference; what is not in doubt is that on these seven, deleting spans that never touch the

payload took an attack succeeding between a tenth and seven tenths of the time to one succeeding in every execution.

Six user tasks is few, and one of them carries three of the nine configurations, so the ratio is recomputed dropping each task in turn. It runs from 0.94 to 1.21 across the six, every value inside the preregistered interval, and dropping the task with three configurations gives 1.21. The result does not rest on one task.

A referee of an earlier round named this mechanism before we had measured it: removing legitimate task content can leave the injected instruction relatively more influential, and would then raise harm rather than lower it. E17 is consistent with that and does not identify it – the design separates neither a semantic effect of the deleted content from a structural effect of deleting it, nor either from what the agent would have done with the content present. What it does settle is that the direction was never the finding. A control whose per-configuration effects cancel on one population and point one way on another is not an inert probe on either, and any reading in which the direction carries no information is contradicted here.

One thing these effects do NOT do is confuse a signed source ranking, and saying so is the difference between the supported result and an overstatement. A rule that ranks candidates by how far each one's neutralization lowers harm compares the treatment's gap against the control's, and on both populations the treated arm is zero in every recorded execution – 0 of 520 on the census and 0 of 180 here, an observed rate rather than a population value. Two things follow that should not be merged: the ORDERING of the two population effects, which these records leave unidentified because q_T is only bounded above, and the FINITE-BUDGET error of a rule reading a few draws, which Section 4.7 measures separately. The treatment gap is therefore the factual rate, which is positive whenever there is anything to attribute, while the control's gap is smaller or negative. The ranking survives however large the control's effect is and whichever way it points. What the arms establish is that the outcome is SENSITIVE to edits outside the payload, at a magnitude of the treatment's order; what they do not establish is that this sensitivity misdirects a method, and on these records it cannot, because a saturating true edit cannot be outranked.

It is worth being exact about the condition, because an earlier version of this work was not. A rule preferring the candidate whose neutralization lowers harm more picks the treated one exactly when $p_T < p_C$; it ties when $p_T = p_C$, and reversal requires $p_T > p_C$. Here p_T is an EMPIRICAL zero – 0 of 520 on the census and 0 of 180 on banking, not an assumed property – so $p_F - p_T \geq p_F - p_C$ holds for any control rate and the ordering survives whatever the factual rate is. We had proposed that an arm using an attack the model resists part of the time would expose misattribution; it would not, because that changes p_F and the contested configurations already have intermediate p_F by construction. An arm that could reverse the ranking needs residual harm AFTER the injected span is neutralized, with source membership still known – a payload delivered redundantly, so that removing the candidate span leaves another route open. Even then a reversal would be possible rather than guaranteed, and we make no prediction that such an arm would find one.

What the direction is consistent with, without claiming it, is this. Section 4.3 reads an intermediate harm probability as composition: the benchmark's two verdicts are close to mutually exclusive, 0.890 of executions satisfying exactly one, so such a configuration is one where the agent sometimes pursues the injection's goal and sometimes the user's. That is a decomposition of an observed rate, read across configurations. A.18's direction is what the same competition would look like under intervention: remove material the user's task was relying on and the injection is left with less to compete against. The design does not establish that the deleted spans carried user-task material – they are count-matched spans of the same tool return, chosen by position and not by role – so this is a reading the result permits and not one it identifies. An arm that deleted spans by role rather than by position would separate them; it is not in the preregistration and is named here as a follow-up.