

Wrong but Useful: Trajectory Value Beyond Answer Correctness in Multi-Agent Messages

Chih-Hsuan Yang^{1*}, Anjir Ahmed Chowdhury², Cheng-Hau Yang¹,
Weijian Zheng¹, Fernando Llorente³, Xiaolong Ma¹,
Xinyang Li², Eliu A. Huerta^{1,4}, Ian T. Foster^{4,1}, Rajeev Thakur¹

¹Argonne National Laboratory, Lemont, IL, USA

²University of Houston, Houston, TX, USA

³Brookhaven National Laboratory, Upton, NY, USA

⁴University of Chicago, Chicago, IL, USA

bellayang@anl.gov

Abstract

Multi-agent reasoning systems often use agreement, confidence, or automated scores to decide which messages should shape a final answer. Such filtering assumes that a message likely to be correct is also a message worth keeping. Yet a wrong answer can contain a useful decomposition, constraint, or scientific principle. We test this distinction with **Diverse Hypothesis Deliberation (DHD)**, a controlled measurement protocol that caches five independently generated messages and replays the same downstream solver, called the integrator, with each message available or hidden. The replay comparison measures a message’s *trajectory value*: whether making the message available helps or harms the reasoning that follows. Across five mathematics and science benchmarks and two openly available model families—gpt-oss-120b (OSS) and gemma-4-31B-it (Gemma)—wrong-helpful messages appear in every benchmark-model combination. Among wrong-answer messages that change final correctness, more than four in ten changes are helpful in each model. Controlled repeats show that the number of repeatable message effects is unlikely to arise from replay variation alone ($p = 0.0002$). A focused intervention on repeatable wrong-helpful messages adds a second result: the complete message works best, while retaining its reasoning preserves more success than retaining only its answer. The source of the complete-message advantage remains open. Within the same problem, repeated trajectory-value evidence also identifies a better keep-or-remove choice than answer correctness alone. Together, these results show that answer correctness is informative, but it does not determine trajectory value. DHD measures this missing property and produces reusable labels for learning when agents should listen.

Introduction

Reasoning systems increasingly generate several candidate paths or messages before deciding what influences a final answer. Existing methods select among them using agreement, process scores, learned rankings, or confidence (Wang et al. 2023c; Uesato et al. 2022; Lightman et al. 2023; Jiang, Ren, and Lin 2023; Chen, Saha, and Bansal 2024). Multi-agent systems also prune communication to remove redundant or disruptive traffic (Zhang et al. 2025a). Such methods reflect a

practical constraint: not every generated message should be passed forward. Since ground-truth correctness is unavailable during inference, agreement, confidence, and learned scores act as proxies for message reliability.

Reliability proxies help select a final answer, but they leave a different question unanswered: does the answer attached to a message reveal whether the reasoning inside will help the next agent? We separate the two judgments. *Proposal correctness* asks whether the message’s own answer is right. *Trajectory value* asks whether making the whole message available helps or harms subsequent reasoning. Correctness may predict trajectory value on average, but does not determine trajectory value. The unit we study is one complete agent message: its reasoning plus its proposed answer. We use *message* for this object throughout.

A message can make a final arithmetic or option-selection error while supplying the decomposition, constraint, or scientific principle that another agent needs. Conversely, a correct answer can be paired with misleading reasoning that disrupts integration. We call the counterintuitive first case *wrong but useful*, abbreviated *wrong-helpful* in figures and tables. Viewed through value of information (Howard 1966), a message is valuable because of what a decision-maker can do with the message, not only because every statement inside the message is correct. Trajectory value is therefore context-specific: the value describes a message in a particular pool and integrator, not an intrinsic score attached to the text.

Figure 1 makes the distinction concrete through a DHD trace from a number-theory problem. Looking inside the wrong-answer message reveals a useful even-odd decomposition: the agent analyzes even and odd values of L separately. The agent nevertheless misses the boundary case $L = 1$, so its proposed answer is wrong. With the message available, the downstream integrator recovers the complete answer; when the message is hidden, the integrator misses the same case. Figure 1 therefore shows how a wrong-answer message can still contribute useful reasoning in its integration context.

End-to-end accuracy cannot expose such message-level influence. Unlike systems designed only to maximize final accuracy, DHD holds candidate messages fixed and changes only whether one message is available. Matched replays can therefore separate useful ideas that were never generated from

*Corresponding author.

Wrong-helpful message. Dataset / problem identifier: Omni-MATH-2 / omni2:t4:0291. Message: message 5, Growth-Bound Analyst. Problem: For positive integers L, let $S_L = \sum_{n=1}^L \lfloor n/2 \rfloor$. Determine all L for which S_L is a square. Ground-truth answer: $L = 1$ or L is even. Stored agent-message excerpt Separate even and odd L: $S_{2m} = m^2$ and $S_{2m+1} = m(m+1)$. Every even case is a square, while the product of consecutive integers $m(m+1)$ is not a square for $m \geq 1$. Proposed answer (wrong): All even positive integers L. Observed replay effect: With the target message, the full pool returns “$L = 1$ or L even” (correct); without the target message, the integrator returns only “L even” (wrong). Plausible mechanism: The parity split rules out every odd $L > 1$, leaving $L = 1$ as the only boundary case for the integrator to recover from the rest of the pool, even though the proposed answer omits that case.
--

Correct-harmful message. Dataset / problem identifier: MaScQA / G-XEC-2016-19. Message: message 4, Estimation and Sanity-Bounding Expert. Problem: Natural rubber has repeat-unit mass 68 and sulfur has atomic mass 32. If 10 g sulfur is added to 50 g rubber at a 1:1 repeat-unit-to-sulfur ratio, find the maximum percentage of cross-linked sites. Ground-truth answer: 42–43%. Stored agent-message excerpt Convert both masses to moles and divide moles of sulfur by moles of rubber repeat units. Re-evaluate whether one sulfur atom bridges two sites, which would double the percentage. Proposed answer (correct): Approximately 43%. Observed replay effect: With the target message, the full pool returns 85% (wrong); without the target message, the integrator returns 42.5% (correct). Plausible mechanism: Although its answer is correct, the message also makes the doubling convention salient. Mentioning that caveat reinforces another pool member’s proposed answer of 85% and may help the wrong doubled calculation dominate the final synthesis.

Figure 1: Example trace records for two real OSS five-message in-pool leave-one-out events. Message wording is quoted from stored structured fields, with mathematical typography normalized and nonessential fields omitted. The outlined answer is part of the stored agent message. Each card distinguishes the measured replay effect from a plausible mechanism suggested by the surrounding pool; only the former defines the observed replay-effect label.

ideas that were present but ignored, duplicated, or offset by harmful messages. For system-level context, Appendix reports system-level accuracy and computation on the same benchmarks and models.

Message-level attribution matters whenever a system decides which messages to preserve or prioritize. Judging a message only by its attached answer can discard a useful constraint or preserve distracting reasoning. We therefore ask: *does proposal correctness determine whether making the whole message available helps downstream integration?*

We make the question measurable with **Diverse Hypothesis Deliberation (DHD)**. DHD is a measurement protocol, not an accuracy-maximizing selector. A recruiter assigns five complementary, problem-specific roles, and five agents produce one structured message each. Five messages provide several viewpoints while keeping replay analysis tractable. DHD creates the pool before any answer or downstream effect is observed, and no message is filtered by answer, confidence, or later effect. A separate integrator weighs selected cached messages without peer debate or voting. We compare each message alone with independent solving and rerun the larger pool after hiding the target message. We call the second comparison a *leave-one-out (LOO) replay*.

Across five math and science benchmarks, wrong-helpful messages appear in every benchmark-model combination. Among wrong-answer messages that change final correctness, more than four in ten produce a helpful change. Repeated comparisons confirm that the overall pattern exceeds ordinary replay variation, and a controlled Gemma test that separately hides the reasoning and the answer finds more evidence for benefit from the reasoning than from the attached wrong answer.

Contributions. Conceptually, we distinguish a message’s proposal correctness from its context-specific trajectory

value. Methodologically, DHD measures trajectory value with fixed-pool, available-versus-hidden comparisons and controlled repeated replay. Empirically, we test five scientific reasoning benchmarks and two independently developed model families, then separately hide reasoning and answer fields to probe why established wrong-helpful effects persist. Practically, we package release-safe context-indexed labels and show that repeated value evidence exposes a same-problem message-selection opportunity. The central finding is simple: correctness is useful evidence about a message, but it does not determine that message’s trajectory value.

Related Work

Correctness supervision and refinement. Process reward models score steps or search states by association with correct solutions (Uesato et al. 2022; Lightman et al. 2023; Wang et al. 2024a; Zhang et al. 2024). Self-refinement and critique systems instead revise a solver using feedback (Madaan et al. 2023; Shinn et al. 2023; Lin et al. 2024), although models may resist or misuse even strong feedback (Huang et al. 2023; Valmeekam, Marquez, and Kambhampati 2023; Jiang et al. 2025). We ask instead whether one message changes a separate integrator’s outcome.

Imperfect information can still help. Failed trajectories can provide training signal (Wang et al. 2024b), and incorrect demonstrations can improve in-context learning (Alazraki et al. 2025). Trajectory value complements these aggregate findings by measuring which cross-agent message helps or harms in its current context.

Multiple reasoning paths and selection. Self-consistency, search, hypothesis generation, and candidate ranking maintain multiple paths and select through

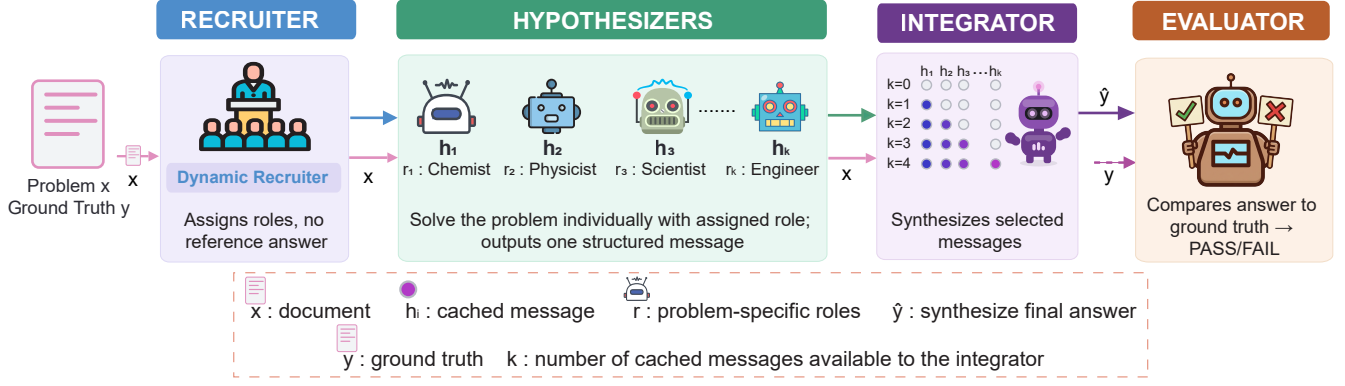


Figure 2: The Diverse Hypothesis Deliberation (DHD) protocol. A recruiter assigns five problem-specific roles without seeing the ground-truth answer. Each hypothesizer independently produces one structured message from its role, with no access to peer messages or the ground-truth answer. An integrator synthesizes a selected subset of the cached messages into one final answer, without being told which proposed answers are correct. An evaluator then compares the integrator’s answer with the ground-truth answer without observing agent reasoning.

agreement, search value, learned rankings, or confidence-weighted consensus (Wang et al. 2023c; Yao et al. 2023; Wang et al. 2023a; Jiang, Ren, and Lin 2023; Chen, Saha, and Bansal 2024). Communication pruning similarly limits which agent messages survive (Zhang et al. 2025a). Selection and pruning methods seek a better or cheaper final solution; DHD instead fixes a generated pool and asks what each complete message contributes.

Multi-agent aggregation and contribution. Debate, role-based collaboration, and reconciliation aggregate reasoning paths (Du et al. 2023; Wang et al. 2023d; Tang et al. 2023; Chen, Saha, and Bansal 2024); related work studies communication topology and agent-level importance (Liu et al. 2023; Tran et al. 2025; Zhang et al. 2025b; Wang et al. 2025). DHD measures contribution per *message*, separating candidate availability from successful integration.

Replay attribution and positioning. Concurrent work uses removal or replay to attribute whole-agent contributions and failures (Chen et al. 2026; Lu et al. 2026; Shah 2026). Removal replay itself is not our novelty. We pair each message’s replay effect with the correctness of its own answer. Unlike Shapley-style attribution, which averages across coalitions (Shapley 1953), our LOO contrast conditions on the observed pool; trajectory value is therefore contextual.

Method

Our method separates a message’s own answer from what happens after the integrator sees the message. We first define these two properties, then describe how DHD holds generated messages fixed and measures their contribution through replay.

Correctness and Trajectory Value

A message can be judged in two different ways: whether its proposed answer is correct and whether the whole message helps the downstream integrator. For a task instance x with

ground-truth answer y , let $E(x, a, y) \in \{0, 1\}$ be the fixed answer-equivalence evaluation procedure used throughout the study. A *message* h_i is the structured output produced for x by one hypothesizer. Each message contains reasoning and a proposed final answer a_i . We use *message* for this complete object throughout and evaluate two of its properties. *Proposal correctness* is local: $c_i = E(x, a_i, y)$ asks whether the message’s proposed answer is judged equivalent to the reference. *Trajectory value* is contextual: it asks whether the integrator is more or less likely to solve the problem when h_i is available, holding the problem and surrounding evidence fixed. Trajectory value belongs to a message in a specified pool and integration context, not to the text alone.

We analyze each whole message. Proposal correctness labels only its answer field; trajectory value measures the downstream consequence of exposing the integrator to its reasoning and answer together.

A *matched replay pair* consists of two otherwise identical integration runs: one makes the target message available and the other hides the target message. We label the observed replay effect helpful when only the run with the message is correct, harmful when only the run without the message is correct, and neutral when final correctness is the same. We write these three outcomes as $\Delta_{i,K} \in \{+1, 0, -1\}$. Thus *wrong-helpful* is the wrong but useful case of the title: the proposed answer is wrong, but making the complete message available changes the result in the helpful direction. Here, harmful describes this matched downstream change; the harmful label does not mean every sentence is false.

Diverse Hypothesis Deliberation

DHD isolates message contribution while limiting regeneration confounds. Independent generation and caching hold the message pool fixed; a separate integrator then sees selected subsets without roles or messages being regenerated. Figure 2 summarizes the information flow.

We choose DHD for measurement rather than maximum

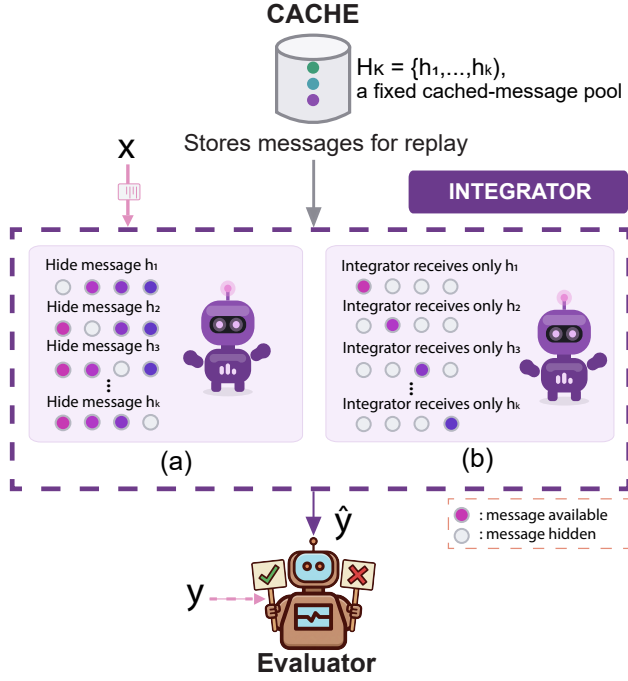


Figure 3: Two replay mechanisms provide complementary observations of trajectory value from the same cached messages. (a) In-pool leave-one-out (LOO) replay hides h_i and compares the result with full-pool integration; (b) single-message replay shows only h_i and compares the result with independent solving. Each matched replay pair yields a helpful, neutral, or harmful observation.

full-pool accuracy. Testing whether a wrong answer can still carry useful reasoning requires independently generated, unfiltered viewpoints, and each message remains unchanged between the compared runs. Debate, revision, or winner selection may improve final accuracy, but they rewrite or discard the message being measured. The five-role design instead creates a small, varied, fixed pool before either label is observed. Here, *diverse* means role-conditioned viewpoints requested before generation, not heterogeneous models or post-hoc filtering.

Protocol Flow DHD is a one-shot message-pool protocol. For each problem x , a dynamic recruiter assigns a problem-specific roster of five complementary roles. The five *hypothesizers* then work independently: each sees the problem and one role, but no peer messages, scores, evaluator feedback, or ground-truth answer. Each returns a structured message containing reasoning and a proposed answer. A separate *integrator* sees the original problem and a selected subset of these cached messages, then verifies and synthesizes one final answer. The integrator is not asked to vote or simply copy the most common proposed answer.

Dynamic recruitment lets the roster follow the problem: an algebra task may elicit symbolic and boundary-case analysts, whereas a biology passage may elicit evidence-retrieval and experimental-design roles. The roster and fixed mes-

sage pool $H_5 = \{h_1, \dots, h_5\}$ are created once. Here, K is the number of messages shown to the integrator, and $H_K = \{h_1, \dots, h_K\}$ is the corresponding nested prefix. A smaller H_K is therefore not a fresh random sample and does not trigger re-recruitment.

Operating outside the reasoning protocol, the evaluator sees only the submitted answer and evaluator-only ground-truth material, then performs constrained answer comparison. Message selection and agent reasoning remain hidden from the evaluator. The full prompt-role interfaces and information boundaries appear in Appendix ; the verbatim templates and protocol configuration are included in `anc/reproducibility_artifact.zip`.

Measuring Trajectory Value by Replay The $K=0$ baseline is an *independent solver*: the same model solves the problem with no messages. Every replay reruns only the integrator on a subset of the fixed H_5 pool; roles and messages are never regenerated.

We estimate the same concept in two contexts, illustrated in Figure 3. *Single-message replay* (Figure 3(b)) compares integration with only h_i against independent solving. *In-pool LOO replay* (Figure 3(a)), used for $K=2, \dots, 5$, compares the full H_K prefix against $H_K \setminus \{h_i\}$. In either matched replay pair, the message is helpful when only the condition containing it is correct, harmful when only the condition without it is correct, and neutral otherwise. Because each message has its own matched replay pair, two messages can both be helpful relative to the same independent baseline.

Single-message and in-pool comparisons answer complementary questions. Single-message replay tests whether one message can create a successful route from an otherwise unsuccessful independent attempt. In-pool LOO asks whether that message remains useful, redundant, or harmful after other messages are already present. We do not expect their labels to agree case by case because trajectory value depends on the surrounding information set.

Within each matched replay pair, the problem, model family, prompt template, cached messages, relative order of the retained messages, and evaluation procedure remain fixed; no fresh pool is sampled. Only target visibility changes, so the contrast measures whole-message availability in context. Removal also changes prompt length and absolute token positions; LOO alone does not isolate semantic content, which a later component-level test probes. Each pair remains one stochastic realization, motivating the expectation below.

For a message set S , define $Y_S = E(x, A_S, y)$ as whether the integrator’s answer A_S is judged correct. If the same matched comparison could be repeated many times, the expected LOO trajectory value would be

$$\tau_{i,K} = \Pr(Y_{H_K} = 1) - \Pr(Y_{H_K \setminus \{h_i\}} = 1). \quad (1)$$

Equation 1 has an intuitive sign: positive $\tau_{i,K}$ means the message raises success on average, zero means the message is neutral or redundant in that pool, and negative $\tau_{i,K}$ means the message lowers success. We keep three empirical objects separate throughout: proposal correctness c_i ; one observed matched-replay effect $\Delta_{i,K}$; and a *repeatable effect*, whose direction is supported across controlled repeats. Proposal

correctness does not determine either replay quantity. A single $\Delta_{i,K}$ is one noisy observation of the comparison; controlled repeats are needed to estimate the expected difference $\tau_{i,K}$.

Crossing proposal correctness with the signed replay effect yields six cells: wrong-helpful, wrong-neutral, wrong-harmful, correct-helpful, correct-neutral, and correct-harmful. Wrong-helpful messages show that correctness is not necessary for positive value; correct-harmful messages show that correctness is not sufficient. We report both the *correctness-flip rate*—the share of eligible replays that are helpful or harmful—and, among those flips, the share that are helpful. Neutral remains separate because redundancy is not harm.

We also distinguish *candidate availability*, whether any proposed answer in a pool is correct, from successful integration of that pool. A complete single-versus-full diagnostic and the descriptive $K=1-5$ trajectories appear in Appendices and . Because H_K is a nested prefix of one recruited five-role roster, changes over K describe pool-size trajectories rather than a randomized causal effect of adding an agent.

Experimental Setup

We study five benchmarks: Omni-MATH-2 (4,181 problems) (Ballon et al. 2026), JEEBench (515) (Arora, Singh, and Mausam 2023), SciBench (580) (Wang et al. 2023b), a strict text-only LAB-Bench slice (741) (Laurent et al. 2024), and MaScQA (649) (Zaki et al. 2023). Together, the benchmarks span open-answer mathematics, mixed-format exams, college science, long-evidence multiple-choice biology, and materials science, testing the distinction under different answer formats and integration demands. We evaluate `gpt-oss-120b` (OSS) (OpenAI 2025) and `gemma-4-31B-it` (Gemma) (Gemma Team 2026); within each run, the same model fills the recruiter, hypothesizer, and integrator roles. We chose two openly available model families so the same prompts, repeated calls, and packaged records can be audited under one setup and replicated across independently developed models. The experiment records use the canonical endpoint identifiers `openai/gpt-oss-120b` and `google/gemma-4-31B-it`; on backends that expose shorter deployment aliases, only the API request name is mapped, while the canonical identifier remains in the run provenance. The hosted endpoints did not expose weight-revision hashes, so we report model identifiers, prompt configurations, decoding settings, and per-run provenance rather than an unavailable checkpoint hash.

Both model families follow the same role-specific decoding settings. Hypothesizers use temperature 0.7 to generate complementary paths, while the recruiter, integrator, and evaluator use temperature 0. Role-specific completion limits do not truncate input messages. A separate `gpt-oss-120b` evaluator applies the same answer-equivalence evaluation procedure to every condition from both families. No reasoning agent sees the ground-truth answer or evaluator judgment, and the evaluator never sees agent reasoning. Keeping the

procedure fixed makes judgments comparable across model families.

Protocol accuracy counts every retained problem, treating missing or non-answer outputs as incorrect. Message-level results require a rendered message and a complete matched comparison; missing replays are never imputed as neutral. The headline LOO analysis contains 91,740 eligible OSS and 83,020 eligible Gemma messages, each with a nonempty extracted answer. Full sample counts and missing-data flow appear in Appendix .

To measure replay variation directly, we additionally sample 200 problems from each benchmark for each model family, for 1,000 problems per family. At $K=5$, each of five matched blocks evaluates the same full-pool input twice and each of the five one-message removals. Each problem therefore yields 35 observations: repeated full–reduced comparisons for estimating message effects, plus an identical-input comparison that measures ordinary replay disagreement. Cached messages remain unchanged throughout; only the integrator is rerun.

Measurement effort. DHD is a measurement protocol rather than a low-cost deployment policy. Recruitment and the five messages are generated once. A $K=5$ LOO scan then reuses that pool for six integrator outcomes: one full-pool result and five one-message removals. Controlled repeated replay uses 35 integrator outcomes per sampled problem; statistical analyses make no model calls. For system-level context, four end-to-end protocols on Omni-MATH-2 average roughly 18K to 616K logged tokens per problem. Appendix reports those costs and four-protocol results across both models and all five benchmarks.

A robustness replay of 16,724 saved submissions with three evaluator models yields 94.2–96.6% pairwise agreement on individual answers. A stricter audit recomputes the proposed-answer, full-pool, and reduced-pool judgments together. When all three judgments needed to form a six-cell label are required to match simultaneously, pairwise agreement is 71.6–78.6% for OSS and 82.4–88.7% for Gemma. Every evaluator still recovers both off-diagonal categories. Complete denominators and confusion counts appear in the Appendix .

Results

Richer Pools Create Potential and Integration Risk

Moving from independent solving ($K=0$) to all five messages changes macro-average accuracy by +1.6 percentage points for OSS and +0.3 for Gemma, with mixed directions across benchmarks (Table 1). Appendix reports the complete $K=0-5$ trajectories. Small net changes are not a measure of collaboration or trajectory value: averaging final accuracy combines helpful, harmful, and neutral effects, so opposing message effects can cancel. DHD is designed to expose that mixture rather than make accuracy grow monotonically with K .

Correct candidates are often available before they are integrated. In the validated OSS matrix, at least one of

Table 1: Final accuracy (%) for independent solving ($K=0$) and integration with all five messages ($K=5$). Δ is $K=5$ minus $K=0$ in points and is computed from unrounded accuracies.

Benchmark	OSS			Gemma		
	$K=0$	$K=5$	Δ	$K=0$	$K=5$	Δ
Omni-MATH-2	74.2	78.3	+4.1	76.8	75.0	-1.8
JEEBench	84.1	86.0	+1.9	88.9	88.5	-0.4
SciBench	75.7	79.3	+3.6	73.1	73.6	+0.5
LAB-Bench	41.6	39.8	-1.8	56.5	61.7	+5.2
MaScQA	90.8	91.2	+0.5	95.1	93.4	-1.7
Macro average	73.3	74.9	+1.6	78.1	78.4	+0.3

five messages contains a correct proposed answer on 76.0–95.4% of problems, depending on the benchmark. Candidate availability exceeds $K=5$ accuracy by 4.2–36.2 points. Candidate availability alone does not establish usable reasoning: a correct answer may be attached to unusable reasoning, and the integrator is not told which proposed answer is correct. Candidate availability nevertheless shows why final accuracy alone cannot tell whether the group failed to produce a good candidate or failed to use an available correct candidate. Matched Gemma results and proposed-answer coverage appear in Appendices and .

Integration can recover or erase available routes. Simple voting does not explain the gap: only 36.9% of complete Gemma pools have a correct majority, yet full-pool accuracy is 78.7%. More directly, the same integrator succeeds with one message but fails with all five on 9.8% of complete OSS problems and 6.9% of complete Gemma problems; the reverse occurs on only 0.5% and 0.2%. Observed paired outcomes suggest integration interference but do not identify which message changed the result. Appendix gives complete counts.

Correctness Does Not Determine Trajectory Value

Complete replay matrices count how often hiding a message changes final correctness; controlled repeated replay tests which directions persist.

One matched replay reveals both mismatches. Figure 4 pools in-pool LOO replay over $K=2$ –5 and all five benchmarks. Both model families contain wrong-helpful and correct-harmful observations. A single-message estimate shows the same qualitative pattern in a different context; see Appendix . The complete matrices describe observed correctness flips from one matched comparison per replay condition; repeated calls test whether an effect persists.

Proposal correctness influences both how often a message flips final correctness and, conditional on a flip, its direction. Figure 4 keeps these denominators separate: colored bars condition on flips, while each row also reports the rate among all eligible messages.

Messages with wrong proposed answers change final correctness more often than messages with correct proposed answers. Direction creates the surprise: when a wrong-answer

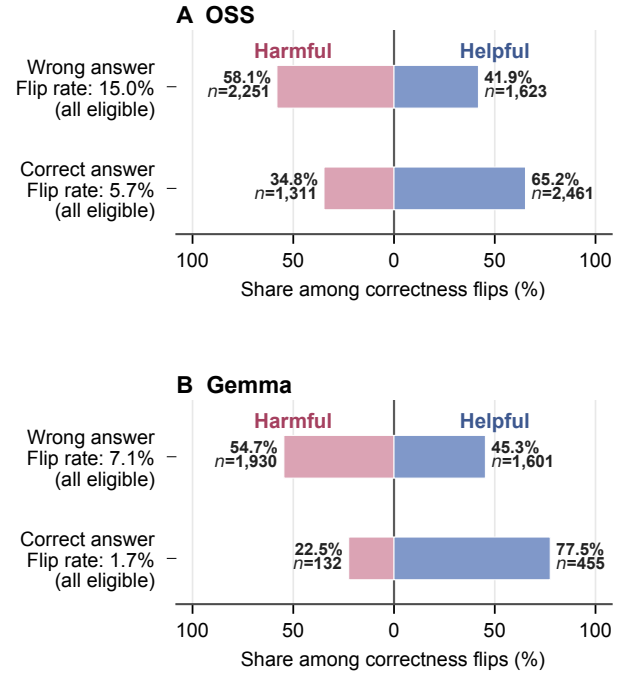


Figure 4: Direction of in-pool LOO effects among replays that flip final correctness. The colored bars answer: among flips, which direction did they take? Helpful and harmful shares therefore sum to 100% within each row, and bar labels give the event counts. The row label separately reports the flip rate among all eligible messages, including neutral replays. Helpful means the full pool is correct and the reduced pool is wrong; harmful means the reverse. More than four in ten flips involving wrong-answer messages are helpful in each model family; across all messages with wrong proposed answers, helpful flips account for 6.3% in OSS and 3.2% in Gemma.

message does flip correctness, 41.9% of OSS events and 45.3% of Gemma events are helpful. A wrong answer therefore makes harm more likely, but does not determine whether the complete message helps or harms.

A problem-cluster bootstrap places the helpful share among flips involving wrong-answer messages at 39.5–44.3% for OSS and 42.6–48.1% for Gemma (95% intervals). Bootstrap intervals quantify problem sampling in the complete replay matrix, not repeated-call uncertainty.

Controlled repeated replay confirms the distinction. We repeat the $K=5$ comparison on 1,000 stratified problems per model. Full-pool calls with exactly the same input disagree on final correctness in 7.3% of OSS pairs and 2.1% of Gemma pairs, so some apparent effects reflect ordinary output variation. We therefore form 5,000 null datasets by exchanging “full pool” and “one message removed” labels only within each matched replay block. Under no systematic availability effect, these labels are interchangeable. No null dataset matches the observed aggregate effect count, giving the add-one permutation result $p = (0 + 1)/(5,000 + 1) = 0.0002$;

randomized call scheduling gives similar pooled shifts.

Because five messages from one problem share full-pool outcomes, we also group them into a problem-level test. A global Benjamini–Hochberg screen, which controls the expected false-discovery share among retained findings, keeps 11 Gemma wrong-helpful problems spanning all five benchmarks and none for OSS. Thus controlled repeats detect message-availability effects beyond ordinary replay variation in both families; multiplicity-controlled wrong-helpful cases are recovered for Gemma, not OSS. Repeated correct-harmful evidence is weaker and is treated as a secondary whole-message diagnostic. Appendix gives the complete counts, overlap analysis, stricter corrections, and sensitivity checks.

Component masking points to reasoning as a source of benefit. Here, *component masking* means hiding either the reasoning or the proposed answer while leaving the other field and the message’s position unchanged. We apply component masking to a fixed sample of 22 Gemma messages selected from categories established before any masking outcome was observed. For the 10 wrong-helpful anchors, integrator success is 82% with the full message, 64% when its wrong answer is hidden, 44% when its reasoning is hidden, 46% after an approximately same-length neutral replacement, and 26% after removal. The small mechanism diagnostic is more consistent with benefit from the reasoning than from the attached wrong answer; the diagnostic is not a prevalence estimate. Appendix gives the complete sample, repetitions, and controls.

Repeated effects expose a same-problem one-removal opportunity. Four repeated blocks choose between keeping all messages and removing one; a fifth, unused for selection, scores that choice. Accuracy improves by 1.68 points for OSS and 2.61 for Gemma, versus 0.94 and 1.00 for a proposal-correctness comparator. Because folds reuse the same problem, the result measures same-problem opportunity rather than a policy for unseen problems. Appendix gives the bootstrap, shuffled-control, and per-benchmark analyses.

Separation Persists Across Benchmarks

Wrong-helpful observations appear in all ten benchmark-model cells (Appendix Figure 7). Both models place Omni-MATH-2 and JEEBench highest in helpful share among wrong-answer flips, SciBench in the middle, and LAB-Bench and MaScQA lower. The cross-benchmark ordering is descriptive because task format, context length, baseline accuracy, and integration demands vary together. Appendix gives $K=2-5$ breakdowns and paired trace examples.

Together, the results expose three distinct stages: generating potentially useful information, preserving useful information during integration, and identifying which messages changed the final trajectory. Proposal correctness predicts influence at the last stage, but does not determine that influence.

Discussion

Standard accuracy collapses candidate availability, successful integration, and message influence, so useful ideas can

be present yet lost or offset. Figure 4 shows that, among wrong-answer messages that change final correctness, more than four in ten produce a helpful change in both families. Controlled repetition detects message-availability effects beyond ordinary replay variation in both families and finds multiplicity-controlled Gemma wrong-helpful cases across all five benchmarks. The component-masking diagnostic localizes more of the observed benefit to reasoning than to the attached wrong answer.

Separating correctness from trajectory value changes the message-selection problem. Correctness can identify a likely answer, but not whether a flawed message contains a reusable decomposition. The fifth-block analysis confirms that repeated value estimates contain decision-relevant information beyond correctness. Deployment would require predicting value before evaluation and transferring to unseen problems; the present study defines the prediction target.

DHD is a measurement protocol rather than an accuracy-maximizing protocol. Its role-diverse pool is fixed before outcomes are known. Editing, filtering, or regenerating messages could improve accuracy but would entangle generation, selection, and integration, the stages DHD is designed to separate.

Limitations

Trajectory-value labels describe a message–pool–integrator context, not an intrinsic property of text; individual signs can change even with fixed problems and messages. Repetition reduces but does not eliminate uncertainty, and the fifth-block analysis measures same-problem opportunity rather than unseen-problem generalization. LOO also hides a whole message in one fixed prompt order, while the smaller masking diagnostic only begins to separate reasoning from its answer field.

Nested K prefixes are not randomized agent additions, and one model family fills all reasoning roles within each run. Results may not transfer to interactive debate, heterogeneous agent models, frontier models, or tasks without a stable ground-truth answer. Gemma rates condition on complete replay records, and compound labels are more evaluator-sensitive than individual answers. The Appendix provides sensitivity tests, sample flow, and evaluator audits.

Conclusion

We asked whether a reasoning message should be discarded solely because its proposed answer is wrong. Across five benchmarks and two model families, the answer is no: more than four in ten observed correctness flips involving wrong-answer messages move the final result in the helpful direction.

DHD makes the distinction measurable by holding candidate messages fixed and comparing integration with and without each message. Controlled repeated replay detects message-availability effects beyond ordinary replay variation in both families; multiplicity-controlled wrong-helpful cases are recovered for Gemma, not OSS. The component-masking diagnostic is more consistent with benefit from the reasoning than from the wrong answer field. Correctness remains useful, but it does not determine trajectory value.

Acknowledgments

This research used resources of the Argonne Leadership Computing Facility, a U.S. Department of Energy (DOE) Office of Science user facility at Argonne National Laboratory (ANL) operated under Contract No. DE-AC02-06CH11357. The work was also supported under the same contract by the DOE Office of Science’s Advanced Scientific Computing Research Program and by Laboratory Directed Research and Development (LDRD) funding from ANL, provided by the Director, DOE Office of Science.

References

- Alazraki, L.; Mozes, M.; Campos, J. A.; Yi-Chern, T.; Rei, M.; and Bartolo, M. 2025. No Need for Explanations: LLMs Can Implicitly Learn from Mistakes In-Context. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 33191–33215.
- Arora, D.; Singh, H.; and Mausam. 2023. Have LLMs Advanced Enough? A Challenging Problem Solving Benchmark For Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.
- Ballon, M.; Algaba, A.; Verbeken, B.; and Ginis, V. 2026. Benchmarks Saturate When the Model Gets Smarter Than the Judge. arXiv:2601.19532.
- Chen, J. C.-Y.; Saha, S.; and Bansal, M. 2024. ReConcile: Round-Table Conference Improves Reasoning via Consensus among Diverse LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 7066–7085. Association for Computational Linguistics.
- Chen, Y.; Sun, Y.; Wang, H.; Wang, J.; Zhang, X.; Shen, X.; Li, W.; and Zhang, W. 2026. Exact Is Easier: Credit Assignment for Cooperative LLM Agents. *arXiv preprint arXiv:2603.06859*.
- Cohen, J. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1): 37–46.
- Du, Y.; Li, S.; Torralba, A.; Tenenbaum, J. B.; and Mordatch, I. 2023. Improving Factuality and Reasoning in Language Models through Multiagent Debate. arXiv:2305.14325.
- Gemma Team. 2026. Gemma 4 Technical Report. arXiv:2607.02770.
- Howard, R. A. 1966. Information Value Theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1): 22–26.
- Huang, J.; Chen, X.; Mishra, S.; Zheng, H. S.; Yu, A. W.; Song, X.; and Zhou, D. 2023. Large Language Models Cannot Self-Correct Reasoning Yet. arXiv:2310.01798.
- Jiang, D.; Ren, X.; and Lin, B. Y. 2023. LLM-Blender: Ensembling Large Language Models with Pairwise Ranking and Generative Fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 14165–14178. Association for Computational Linguistics.
- Jiang, D.; Zhang, A.; Wang, A.; Andrews, N.; and Khashabi, D. 2025. Feedback Friction: LLMs Struggle to Fully Incorporate External Feedback. arXiv:2506.11930.
- Laurent, J. M.; Janizek, J. D.; Ruzo, M.; Hinks, M. M.; Hammerling, M. J.; Narayanan, S.; Ponnampati, M.; White, A. D.; and Rodrigues, S. G. 2024. LAB-Bench: Measuring Capabilities of Language Models for Biology Research. arXiv:2407.10362.
- Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 2023. Let’s Verify Step by Step. arXiv:2305.20050.
- Lin, Z.; Gou, Z.; Liang, T.; Luo, R.; Liu, H.; and Yang, Y. 2024. CriticBench: Benchmarking LLMs for Critique-Correct Reasoning. arXiv:2402.14809.
- Liu, Z.; Zhang, Y.; Li, P.; Liu, Y.; and Yang, D. 2023. A Dynamic LLM-Powered Agent Network for Task-Oriented Agent Collaboration. arXiv:2310.02170.
- Lu, M.; Huang, Y.; Lin, C.; and Lee, S.-I. 2026. Agents that Matter: Optimizing Multi-Agent LLMs via Removal-Based Attribution. *arXiv preprint arXiv:2605.27621*.
- Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegrefe, S.; Alon, U.; Dziri, N.; Prabhunoye, S.; Yang, Y.; et al. 2023. Self-Refine: Iterative Refinement with Self-Feedback. arXiv:2303.17651.
- OpenAI. 2025. gpt-oss-120b and gpt-oss-20b Model Card. arXiv:2508.10925.
- Shah, J. 2026. Causal Agent Replay: Counterfactual Attribution for LLM-Agent Failures. *arXiv preprint arXiv:2606.08275*.
- Shapley, L. S. 1953. A Value for n-Person Games. In Kuhn, H. W.; and Tucker, A. W., eds., *Contributions to the Theory of Games II*, volume 28 of *Annals of Mathematics Studies*, 307–317. Princeton University Press.
- Shinn, N.; Cassano, F.; Berman, E.; Gopinath, A.; Narasimhan, K.; and Yao, S. 2023. Reflexion: Language Agents with Verbal Reinforcement Learning. arXiv:2303.11366.
- Tang, X.; Zou, A.; Zhang, Z.; Li, Z.; Zhao, Y.; Zhang, X.; Cohen, A.; and Gerstein, M. 2023. MedAgents: Large Language Models as Collaborators for Zero-shot Medical Reasoning. arXiv:2311.10537.
- Tran, K.-T.; Dao, D.; Nguyen, M.-D.; Pham, Q.-V.; O’Sullivan, B.; and Nguyen, H. D. 2025. Multi-Agent Collaboration Mechanisms: A Survey of LLMs. arXiv:2501.06322.
- Uesato, J.; Kushman, N.; Kumar, R.; Song, F.; Siegel, N.; Wang, L.; Creswell, A.; Irving, G.; and Higgins, I. 2022. Solving Math Word Problems with Process- and Outcome-Based Feedback. arXiv:2211.14275.
- Valmeekam, K.; Marquez, M.; and Kambhampati, S. 2023. Can Large Language Models Really Improve by Self-critiquing Their Own Plans? arXiv:2310.08118.
- Wang, P.; Li, L.; Shao, Z.; Xu, R.; Dai, D.; Li, Y.; Chen, D.; Wu, Y.; and Sui, Z. 2024a. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. arXiv:2312.08935.
- Wang, R.; Li, H.; Han, X.; Zhang, Y.; and Baldwin, T. 2024b. Learning From Failure: Integrating Negative Examples when Fine-tuning Large Language Models as Agents. arXiv:2402.11651.

Wang, R.; Zelikman, E.; Poesia, G.; Pu, Y.; Haber, N.; and Goodman, N. D. 2023a. Hypothesis Search: Inductive Reasoning with Language Models. arXiv:2309.05660.

Wang, X.; Hu, Z.; Lu, P.; Zhu, Y.; Zhang, J.; Subramaniam, S.; Loomba, A. R.; Zhang, S.; Sun, Y.; and Wang, W. 2023b. SciBench: Evaluating College-Level Scientific Problem-Solving Abilities of Large Language Models. arXiv:2307.10635.

Wang, X.; Wang, J.; Wang, Y.; Dang, P.; Cao, S.; and Zhang, C. 2025. MARS: Toward More Efficient Multi-Agent Collaboration for LLM Reasoning. arXiv:2509.20502.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023c. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations*.

Wang, Z.; Mao, S.; Wu, W.; Ge, T.; Wei, F.; and Ji, H. 2023d. Unleashing the Emergent Cognitive Synergy in Large Language Models: A Task-Solving Agent through Multi-Persona Self-Collaboration. arXiv:2307.05300.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. arXiv:2305.10601.

Zaki, M.; Jayadeva; Mausam; and Krishnan, N. M. A. 2023. MaScQA: A Question Answering Dataset for Investigating Materials Science Knowledge of Large Language Models. arXiv:2308.09115.

Zhang, D.; Zhoubian, S.; Hu, Z.; Yue, Y.; Dong, Y.; and Tang, J. 2024. ReST-MCTS*: LLM Self-Training via Process Reward Guided Tree Search. arXiv:2406.03816.

Zhang, G.; Yue, Y.; Li, Z.; Yun, S.; Wan, G.; Wang, K.; Cheng, D.; Yu, J. X.; and Chen, T. 2025a. Cut the Crap: An Economical Communication Pipeline for LLM-Based Multi-Agent Systems. In *International Conference on Learning Representations*.

Zhang, H.; Cui, Z.; Wang, X.; Zhang, Q.; Wang, Z.; Wu, D.; and Hu, S. 2025b. If Multi-Agent Debate is the Answer, What is the Question? arXiv:2502.08788.

Appendix Overview

The appendix provides a compact evidence map for the paper’s central claims. **Diverse Hypothesis Deliberation (DHD)** is the measurement protocol defined in the main paper. *Prompt Interface Details* specifies every role’s information boundary. *System-Level Protocol and Cost Context* reports the complete four-protocol, two-model, five-benchmark matrix and distinguishes end-to-end accuracy from DHD’s measurement effort. *Complete Results and Analysis Scope* and *Offline Selection and Integration Diagnostics* provide the full matrices, pool-size results, and offline comparisons.

Real Trace Examples Across Benchmarks gives paired examples from every benchmark. *Controlled Repeated Replay Robustness* measures replay variation and repeatable effects; *Component Masking: Hiding Reasoning and Answer Fields* and *Integrator Uptake in Matched Replay Pairs* examine message-level mechanisms. *Cross-Fitted One-Removal Opportunity* tests same-problem selection opportunity. Finally, *Evaluation Procedure and Artifact Schema* records evaluator agreement, missing-data treatment, provenance, and the artifact schema.

Prompt Interface Details

Prompt interfaces record DHD’s prompt specification. All benchmarks share the same core templates and information boundaries, with short format-specific guidance; the ancillary reproducibility archive preserves the verbatim templates. The four roles below are separate model calls.

Recruiter prompt. The recruiter receives the problem and proposes five problem-specific solver roles. The roster is dynamic rather than fixed across domains, and the recruiter never receives the ground-truth answer.

Hypothesizer prompt. Each hypothesizer receives the original problem and exactly one assigned role. The hypothesizer reasons from the assigned role’s perspective and outputs a structured message containing: a reasoning direction, a key idea, critical assumptions, checks another solver should perform, confidence/uncertainty, and a proposed final answer when possible. Hypothesizers do not see other messages, any evaluator output, or the ground-truth answer.

Integrator prompt. The integrator receives the original problem and a selected subset of cached messages. The integrator synthesizes, adopts, combines, corrects, or rejects parts of those messages and then provides one final answer in the benchmark’s expected answer format. The integrator is not told which proposed answer is correct and is not asked to simply vote or select the best answer.

Evaluator prompt. The evaluator receives only the submitted final answer and evaluator-only ground-truth material. The evaluator performs constrained answer comparison and emits a correctness label. The evaluator is not a solver, is not part of DHD’s generation protocol, and does not decide which message the integrator should use.

Replay conditions. Single-message replay shows the integrator one cached message and compares its outcome with the independent solver. In-pool leave-one-out (LOO) replay

shows the same fixed message set as the original full-pool run, except that one message is hidden. Here, K is the number of cached messages shown to the integrator, so full- K denotes the condition containing all K messages. Both replay types estimate trajectory value under different comparison contexts. No replay regenerates roles or messages; only the subset shown to the integrator changes.

```
[RECRUITER] problem → five complementary
roles.
[HYPOTHESizer i] problem + role i →
independent structured message + proposed
answer.
[INTEGRATOR] problem + selected messages
→ verified synthesis + one final answer.
[EVALUATOR] submitted answer +
evaluator-only reference → PASS
(equivalent) / FAIL (not equivalent).
```

Figure 5: Compact DHD prompt surfaces. Monospaced text marks information shown to each model call rather than author narration.

Complete Results and Analysis Scope

Complete results expand the pooled findings into their model, benchmark, pool-size, and analysis-scope components. The model-family abbreviations are gpt-oss-120b (OSS) and gemma-4-31B-it (Gemma).

Table 2: Five-benchmark suite. N is the number of unique problems in each evaluated slice.

Benchmark	Domain and format	N
Omni-MATH-2	Open-answer competition mathematics	4,181
JEEBench	Mixed-choice and numeric exam science	515
SciBench	Numeric and short-answer college science	580
LAB-Bench	Long-evidence multiple-choice biology	741
MaScQA	Mixed-format materials science	649

MaScQA contains 650 raw rows; one row is an exact duplicate, so all analyses collapse it to 649 unique problems. SciBench similarly disambiguates three reused source identifiers with problem-text hashes, retaining all 580 distinct questions.

Table 3: Signed LOO outcomes at $K=5$ across benchmarks (OSS). Helpful counts for messages with wrong and correct proposed answers are denoted W+ and C+; the corresponding harmful counts are W- and C-. Labels include only messages rendered to the integrator.

Benchmark	labels	W+	W-	C+	C-	non-neutral
Omni-MATH-2	20,862	452	526	447	234	1,659
JEEBench	2,510	13	29	47	41	130
SciBench	2,886	29	43	54	31	157
LAB-Bench	3,292	71	130	212	151	564
MaScQA	3,245	11	26	35	13	85

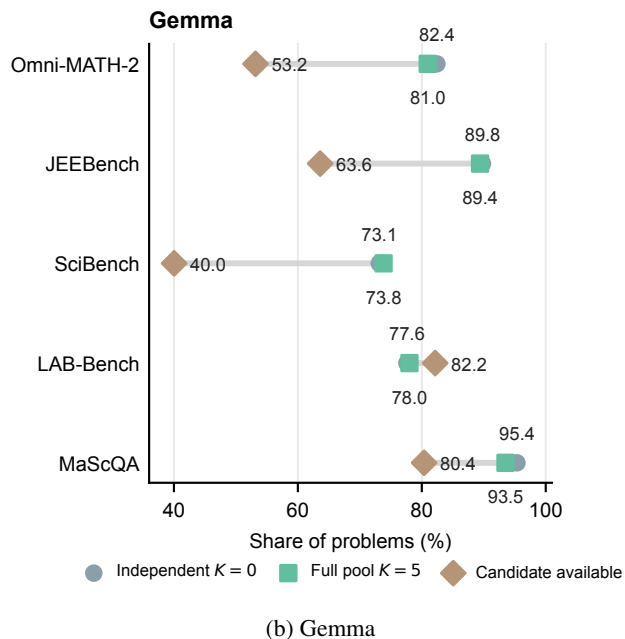
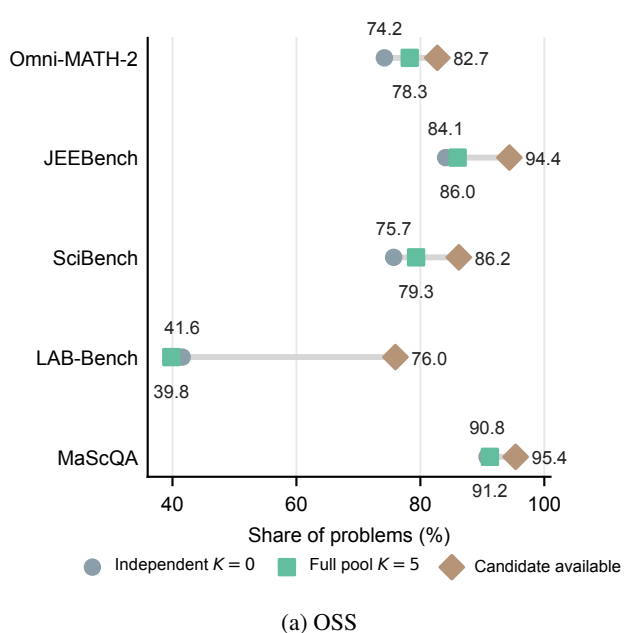


Figure 6: Candidate availability and integration outcomes across model families. Each panel reports independent-solver accuracy, $K=5$ integrator accuracy, and whether at least one message has an evaluator-correct proposed answer. Candidate availability measures answer availability rather than reasoning usability; an integrator can also synthesize an answer absent from the individual messages. The Gemma panel uses its matched complete-LOO sample, whose messages are all answer-bearing.

Table 4: Exact proposal-correctness–observed-replay-effect counts. Columns first condition on whether the message’s proposed answer is wrong or correct, then split its observed replay effect into helpful, neutral, or harmful. The two rows per model are different replay comparisons of the same trajectory-value concept. Gemma rows use the same 5,930-problem complete-LOO sample; the single-message row therefore does not include every retained protocol-accuracy record. Raw counts should not be compared across model families without accounting for these scopes.

Model	Replay comparison	Proposed answer wrong ($c_i=0$)			Proposed answer correct ($c_i=1$)		
		helpful	neutral	harmful	helpful	neutral	harmful
OSS	Single-message vs. independent solver	756	7,108	1,430	1,455	21,424	622
OSS	In-pool leave-one-out, $K=2-5$	1,623	21,963	2,251	2,461	62,131	1,311
Gemma	Single-message vs. independent solver	780	15,455	1,415	297	11,629	74
Gemma	In-pool leave-one-out, $K=2-5$	1,601	45,894	1,930	455	33,008	132

The difference between the protocol and LOO samples is therefore explicit rather than unexplained attrition. Protocol accuracy answers how often the complete workflow succeeds over all assigned problems. Message-level replay instead conditions on a rendered five-message pool and, for the headline matrix, on every required removal outcome being present. The resulting LOO rates are internally paired but should not be treated as full-sample protocol rates, especially for LAB-Bench. OSS has complete LOO coverage for all 6,666 problems. After excluding empty fields that were never shown to the integrator, the headline matrices contain 91,740 OSS and 83,020 Gemma LOO messages.

The intervals use 10,000 problem-cluster bootstrap resamples with a fixed analysis seed of 1203. Each resample draws problem identifiers with replacement and retains all of that problem’s K -by-removal observations, so repeated messages

from one problem are never treated as independent examples.

System-Level Protocol and Cost Context

DHD is a message-level measurement protocol, not a fifth entry in an end-to-end protocol leaderboard. Table 11 provides the broader system-level context from complete matched runs of four existing protocols. *Direct* uses one solver attempt; *Iterative* adds single-agent self-correction; the *planner-executor-reviewer (PER)* protocol uses the corresponding three-stage pipeline; and *Broadcast* uses multi-agent deliberation. All final answers are judged with the same answer-equivalence evaluation procedure. These protocols usually improve final accuracy over direct solving, but not monotonically in every benchmark–model setting. The *Direct* arm belongs to this separately matched four-protocol study; it is not the DHD $K=0$ condition. The two runs use different

	Wrong answer, helpful effect		Correct answer, harmful effect	
	OSS	Gemma	OSS	Gemma
Omni-MATH-2	44.1% flip 16.2% n=2,872	48.6% flip 7.3% n=2,512	33.0% flip 5.2% n=2,103	24.6% flip 2.0% n=345
JEEBench	39.2% flip 16.2% n=158	49.6% flip 5.6% n=230	38.8% flip 4.3% n=258	17.9% flip 1.0% n=28
SciBench	37.0% flip 9.9% n=192	38.5% flip 7.7% n=468	33.6% flip 3.8% n=235	42.3% flip 2.6% n=52
LAB-Bench	35.4% flip 13.5% n=559	29.3% flip 9.7% n=181	39.7% flip 20.9% n=1,045	15.1% flip 2.6% n=126
MaScQA	26.9% flip 9.4% n=93	23.6% flip 5.0% n=140	18.3% flip 1.6% n=131	2.8% flip 0.6% n=36

Figure 7: Off-diagonal in-pool LOO outcomes across benchmarks and model families. Left: helpful share among flips from messages with wrong proposed answers. Right: harmful share among flips from messages with correct proposed answers. Large numbers give these conditional shares; smaller lines give the flip rate among all eligible messages and the number of flips. Neutral replays are excluded only from the conditional share. Cross-benchmark levels are descriptive because tasks and baselines differ.

Table 5: Gemma protocol and leave-one-out (LOO) analysis samples. The protocol column retains every problem and counts missing final answers as incorrect. LOO-eligible problems contain a complete five-message pool; 686 retained problem records do not and therefore cannot define the message-removal comparison. Among eligible problems, 50 have one or more missing removal outcomes, totaling 64 missing pairs; these remain missing rather than being imputed as neutral. The final two columns show why protocol-sample and complete-LOO accuracies should not be mixed.

Benchmark	Protocol	LOO eligible	Complete	Partial	Missing pairs	Protocol $K=0/5$	Complete $K=0/5$
Omni-MATH-2	4,181	3,760	3,724	36	47	76.8 / 75.0	82.4 / 81.0
JEEBench	515	503	500	3	3	88.9 / 88.5	89.8 / 89.4
SciBench	580	578	577	1	1	73.1 / 73.6	73.1 / 73.8
LAB-Bench	741	492	482	10	13	56.5 / 61.7	77.6 / 78.0
MaScQA	649	647	647	0	0	95.1 / 93.4	95.4 / 93.5
Total	6,666	5,980	5,930	50	64	–	–

prompt surfaces, retained samples, and call-accounting contracts, so their numerical accuracies should not be compared as if they were duplicate baselines.

The matched system-level slice uses the largest problem set complete for every model–protocol combination. Its SciBench and MaScQA counts are therefore slightly smaller than the DHD measurement samples reported in Table 2. The two analyses answer different questions: Table 11 asks how often each complete protocol solves a problem, whereas DHD asks which fixed message changed a downstream inte-

gration outcome.

The protocols also differ substantially in inference effort. Table 12 reports the full 4,181-problem Omni-MATH-2 OSS run, for which token and call accounting is complete. Logged calls include reasoning agents, evaluators, and protocol-managed attempts. The comparison is descriptive rather than equal-budget: richer collaboration improves accuracy while consuming more inference.

These end-to-end costs should not be confused with DHD’s replay accounting. DHD creates one five-message

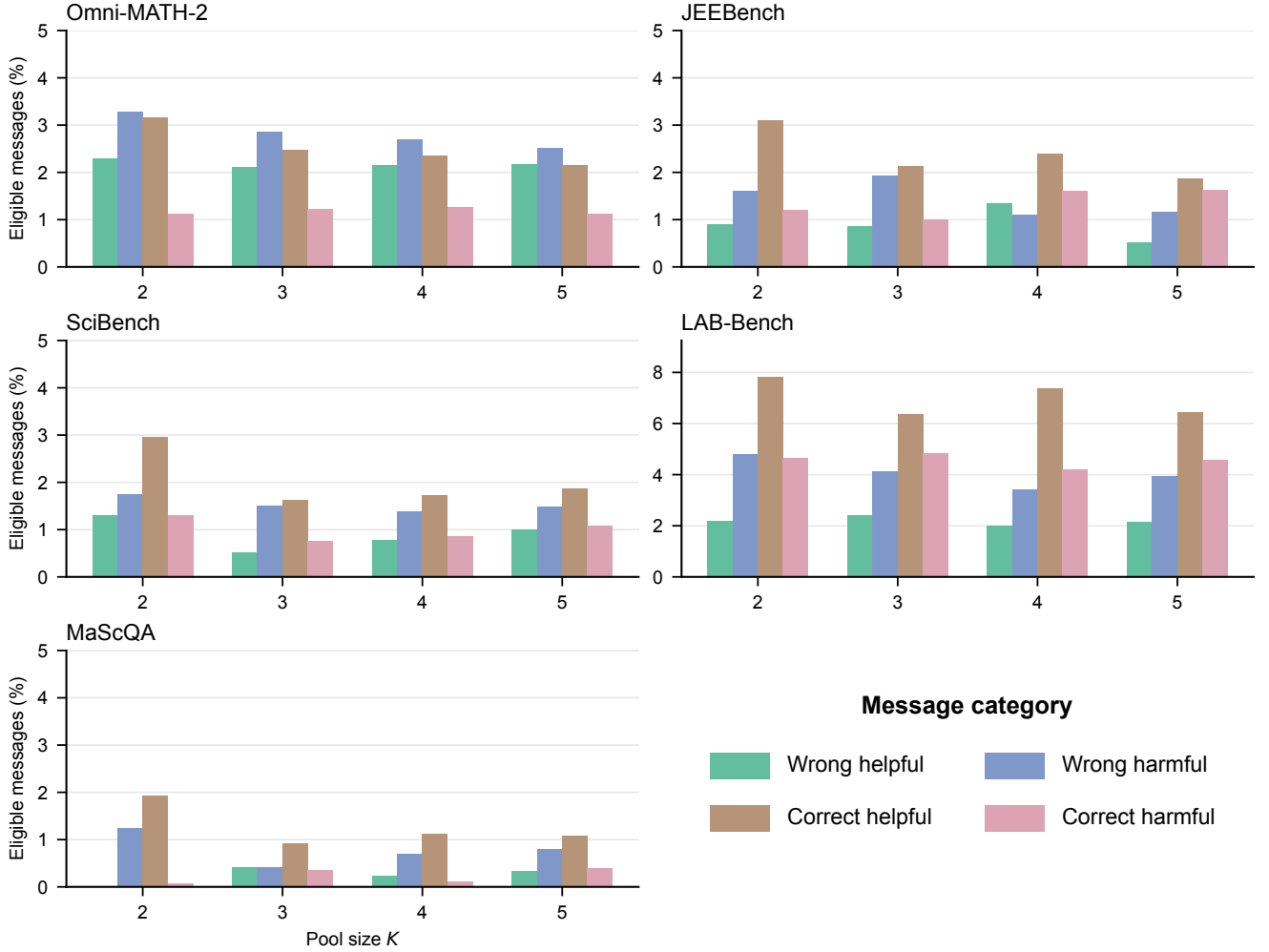


Figure 8: Per-message leave-one-out rates by K and benchmark. Each bar is the share of all eligible LOO messages at that K . Helpful (W+, C+) means removal breaks a correct full-pool result; harmful (W-, C-) means removal fixes a wrong full-pool result. Splitting by proposal correctness prevents the ambiguous reading that “harmful” means only wrong-harmful or only correct-harmful.

pool and reuses it to measure available-versus-hidden effects. A complete $K=5$ LOO scan needs one full-pool integration plus five one-message removals; controlled repeated replay uses 35 integrator outcomes per sampled problem. Detailed DHD call accounting appears in Appendix .

Offline Selection and Integration Diagnostics

The following analyses reuse recorded DHD outcomes and make no additional model calls. They use only problems with complete independent-solver, five single-message, and $K=5$ integration outcomes. The OSS sample contains 6,666 problems; the matched Gemma sample contains 6,017.

Single-message success versus full-pool success. Table 13 expands the pooled comparison into four observable outcome patterns. *Observable interference* means that at least one single-message integration succeeds while $K=5$ inte-

gration fails. *Observable synergy* means that every single-message integration fails while $K=5$ integration succeeds. These labels describe outcomes, not the integrator’s hidden process.

Recorded budget-one opportunity. Each offline policy in Table 14 selects at most one recorded message per problem. Random averages over all five. Answer consensus selects uniformly among messages whose normalized proposed answer is modal; if every answer differs, it reduces to random. Confidence uses the highest self-report, averaging ties. The correctness oracle selects among messages with reference-matching proposed answers and falls back to random if none match. The hindsight best-recorded outcome succeeds whenever any recorded single-message integration succeeds. Because it takes the maximum over one observed outcome per message, it is an opportunity ceiling that can capitalize on

Table 6: Accuracy (%) for nested DHD pools. $K=0$ is independent solving; $K \geq 1$ uses prefixes of one five-message pool. Missing final answers count as incorrect; macro averages are unweighted across benchmarks.

Model	Benchmark	$K=0$	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$
OSS	Omni-MATH-2	74.2	75.6	76.7	77.3	77.6	78.3
OSS	JEEBench	84.1	84.5	85.8	85.2	87.0	86.0
OSS	SciBench	75.7	78.6	79.3	79.1	79.1	79.3
OSS	LAB-Bench	41.6	40.5	40.2	39.5	39.9	39.8
OSS	MaScQA	90.8	90.1	90.8	91.4	91.8	91.2
OSS	Macro average	73.3	73.9	74.6	74.5	75.1	74.9
Gemma	Omni-MATH-2	76.8	74.5	74.8	74.4	75.2	75.0
Gemma	JEEBench	88.9	88.2	88.5	87.0	87.2	88.5
Gemma	SciBench	73.1	72.6	71.4	71.7	71.4	73.6
Gemma	LAB-Bench	56.5	58.7	60.1	61.9	62.2	61.7
Gemma	MaScQA	95.1	94.8	94.3	93.5	93.4	93.4
Gemma	Macro average	78.1	77.7	77.8	77.7	77.9	78.4

Table 7: Pooled offline $K=5$ majority diagnostic (no additional model calls). The majority-correct baseline marks problems with at least three evaluator-correct proposed answers; *integrator only* and *majority only* are correctness disagreements. Strict answer-majority follow rate instead requires at least three matching normalized answer signatures and is descriptive, not causal. Wrong-answer agreement is a conservative lower bound. Rows require five proposed-answer labels and an observed full-pool outcome.

Model	N	Majority-correct	Integrator acc.	Integrator only	Majority only	Strict-majority N	Follow rate
OSS	6,666	71.9	76.0	476	206	5,478	90.1
Gemma	6,456	36.9	78.7	2,736	37	4,423	66.1

Table 8: Dataset drill-down for single-message replay (OSS). W/C denote whether the message’s own proposed answer is wrong/correct; +, 0, and – denote helpful, neutral, and harmful replay effect relative to the $K=0$ independent solver. These rows sum to the OSS single-message row in Table 4.

Benchmark	W+	W0	W-	C+	C0	C-
Omni-MATH-2	584	4,846	974	887	13,303	268
JEEBench	24	243	77	124	1,986	56
SciBench	49	583	67	121	2,027	39
LAB-Bench	80	1,179	233	274	1,281	245
MaScQA	19	257	79	49	2,827	14
Total	756	7,108	1,430	1,455	21,424	622

replay variation; it is not an unbiased estimate of expected best-route accuracy.

Helpful messages discarded and harmful messages admitted are defined using the single-message replay relative to the independent solver. They are reported as mean counts per problem, not as independent causal effects. $K=5$ integration admits all recorded messages, whereas a budget-one policy exposes the integrator to one selected message. These diagnostics explain why policies with similar final accuracy may expose the integrator to different mixtures of messages.

Table 9: Direction of correctness flips from messages with wrong proposed answers in pooled $K=2-5$ LOO replay (OSS). The helpful share conditions on the message having flipped final correctness. Intervals resample whole problems rather than treating repeated K -by-removal observations as independent.

Benchmark	Wrong-answer messages	Flips	Helpful	95% interval
Omni-MATH-2	17,783	2,872	44.1%	[41.3, 46.9]
JEEBench	976	158	39.2%	[28.4, 49.7]
SciBench	1,942	192	37.0%	[28.1, 46.0]
LAB-Bench	4,151	559	35.4%	[29.5, 41.2]
MaScQA	985	93	26.9%	[15.7, 38.3]
Pooled	25,837	3,874	41.9%	[39.5, 44.3]

The proposal-correctness oracle improves over random selection, so correctness is genuinely informative. Its 7.1-point OSS and 5.5-point Gemma gaps to the best recorded outcome show that correctness does not identify every observed successful route. Because the comparison maximizes over single recorded draws, the gap measures available recorded opportunity rather than the expected gain of a deployed selector.

Sampling uncertainty and the source of the recorded gap.

We resample complete problem rows 10,000 times, preserving each problem’s five recorded messages and using the same bootstrap sample for every policy contrast. Table 15 confirms that proposal correctness improves over random selection in both model families, but also that its gap to the best recorded single-message outcome remains substantial. The same ordering holds in all ten benchmark–model cells, with paired intervals excluding zero. Answer consensus and self-reported confidence are modestly above random for OSS but do not improve on random for Gemma. These intervals quantify problem sampling while holding the recorded outcomes fixed; they do not remove replay variation from the hindsight maximum.

Table 16 decomposes the last contrast. A *wrong-only successful route* occurs when at least one message with a wrong

Table 10: Dataset drill-down for leave-one-out replay pooled over $K=2-5$. W/C denote a wrong/correct proposed answer; +, 0, and - denote helpful, neutral, and harmful observed replay effects within the matched comparison. Each model block sums to its LOO row in Table 4. Gemma counts use only complete matched LOO problem records.

Model	Benchmark	W+	W0	W-	C+	C0	C-
OSS	Omni-MATH-2	1,267	14,911	1,605	1,410	38,531	693
OSS	JEEBench	62	818	96	158	5,786	100
OSS	SciBench	71	1,750	121	156	5,897	79
OSS	LAB-Bench	198	3,592	361	630	3,947	415
OSS	MaScQA	25	892	68	107	7,970	24
OSS	Total	1,623	21,963	2,251	2,461	62,131	1,311
Gemma	Omni-MATH-2	1,221	32,056	1,291	260	17,223	85
Gemma	JEEBench	114	3,907	116	23	2,835	5
Gemma	SciBench	180	5,579	288	30	1,979	22
Gemma	LAB-Bench	53	1,688	128	107	4,753	19
Gemma	MaScQA	33	2,664	107	35	6,218	1
Gemma	Total	1,601	45,894	1,930	455	33,008	132

Table 11: System-level final accuracy (%) for four protocols across five benchmarks and two model families. The 40 protocol cells are complete and have no duplicate problem identifiers. N is the number of problems in each matched protocol cell. The comparison establishes the broader collaboration context; the protocols have different computational costs and are not equal-budget DHD baselines.

Benchmark	Model family	N	Direct	Iterative	PER	Broadcast
Omni-MATH-2	Gemma	4,181	69.4	85.9	90.6	93.0
Omni-MATH-2	OSS	4,181	56.8	78.8	85.2	89.2
JEEBench	Gemma	515	70.5	81.4	95.9	98.3
JEEBench	OSS	515	41.6	55.3	91.5	95.0
SciBench	Gemma	574	70.6	79.4	91.3	87.8
SciBench	OSS	574	62.4	72.3	87.5	89.7
LAB-Bench	Gemma	741	44.4	58.2	69.4	89.3
LAB-Bench	OSS	741	19.0	30.2	47.6	74.2
MaScQA	Gemma	642	94.4	97.7	99.4	98.6
MaScQA	OSS	642	80.8	91.4	95.8	97.5
Macro average over 10 benchmark-model cells			61.0	73.1	85.4	91.3

Table 12: Accuracy and inference effort on the complete 4,181-problem Omni-MATH-2 OSS run. Tokens and calls are averages per problem.

Protocol	Accuracy (%)	Tokens (thousands)	Logged calls
Direct	56.8	18.4	9.7
Iterative	78.8	48.1	18.0
PER	85.2	400.1	99.5
Broadcast	89.2	616.3	134.6

proposed answer succeeds in single-message integration but no message with a correct proposed answer does. The category includes problems with no message carrying a correct proposed answer and problems where such messages exist but none succeeds downstream. *Mixed correct outcomes* means that multiple messages with correct proposed answers produce different single-message outcomes, so proposal correctness alone cannot select among them.

Table 13: Single-message success crossed with $K=5$ integration success by benchmark. The four outcomes are shared success (SS), observable interference (OI), observable synergy (OS), and shared failure (SF). Counts use complete matched problems only.

Model	Benchmark	N	SS	OI	OS	SF
OSS	Omni-MATH-2	4,181	3,254	371	19	537
OSS	JEEBench	515	442	43	1	29
OSS	SciBench	580	459	33	1	87
OSS	LAB-Bench	741	286	186	9	260
OSS	MaScQA	649	592	22	0	35
OSS	Pooled	6,666	5,033	655	30	948
Gemma	Omni-MATH-2	3,787	3,031	272	5	479
Gemma	JEEBench	504	451	22	0	31
Gemma	SciBench	580	425	42	2	111
Gemma	LAB-Bench	499	384	52	3	60
Gemma	MaScQA	647	605	29	0	13
Gemma	Pooled	6,017	4,896	417	10	694

Table 14: Pooled recorded-outcome diagnostic. Accuracy and gaps are percentage points; discarded helpful and admitted harmful are mean message counts per problem. The final column is the gap to the best recorded single-message outcome. Hindsight rows are not deployable policies or expected policy estimates.

Model	Policy	Accuracy	Helpful discarded	Harmful admitted	Best-recorded gap
OSS	Random message	73.6	0.272	0.063	11.7
OSS	Answer consensus	74.3	0.272	0.056	11.1
OSS	Highest self-reported confidence	74.2	0.271	0.057	11.1
OSS	Proposal-correctness oracle	78.2	0.249	0.040	7.1
OSS	Hindsight best recorded	85.3	0.209	0.008	0.0
Gemma	Random message	81.3	0.146	0.052	7.0
Gemma	Answer consensus	81.3	0.147	0.051	7.0
Gemma	Highest self-reported confidence	81.0	0.146	0.055	7.3
Gemma	Proposal-correctness oracle	82.8	0.142	0.042	5.5
Gemma	Hindsight best recorded	88.3	0.118	0.011	0.0

Table 15: Paired problem-bootstrap contrasts for pooled recorded outcomes (percentage points), with 95% percentile confidence intervals (CIs).

Model	Contrast	Difference	95% CI
OSS	Correctness — random	4.63	[4.25, 5.00]
Gemma	Correctness — random	1.42	[1.21, 1.63]
OSS	Consensus — random	0.69	[0.45, 0.92]
Gemma	Consensus — random	−0.05	[−0.21, 0.12]
OSS	Confidence — random	0.63	[0.30, 0.97]
Gemma	Confidence — random	−0.37	[−0.69, −0.05]
OSS	Best recorded — correctness	7.12	[6.63, 7.62]
Gemma	Best recorded — correctness	5.54	[5.11, 5.97]

Table 16: Sources of the proposal-correctness gap relative to the best recorded single-message outcome. Contributions are accuracy points; shares sum to each model’s gap.

Model	Gap source	Problems	Contribution	Share
OSS	Wrong-only successful route	337	3.91	54.9%
OSS	Mixed correct outcomes	566	3.22	45.1%
Gemma	Wrong-only successful route	649	5.00	90.4%
Gemma	Mixed correct outcomes	88	0.53	9.6%

The decomposition sharpens the paper’s central claim. Correctness is useful information, but it misses successful routes carried by messages with wrong proposed answers and cannot distinguish messages with correct proposed answers whose downstream outcomes differ. The result does not identify a deployable routing rule; it identifies the information that a value-aware rule would need beyond local answer correctness.

Real Trace Examples Across Benchmarks

Trace examples below give one wrong-helpful and one correct-harmful OSS example from each benchmark. Every card identifies the dataset, problem identifier, message, problem, ground-truth answer, stored agent message, proposed answer, and matched replay outcome. We reproduce the full problem when it is concise; for long deoxyribonucleic acid (DNA) sequences or option lists, we give a faithful prompt excerpt and identify the omission. Message excerpts come from the stored structured message fields; mathematical typography is normalized and nonessential fields are omitted. The outlined proposed answer was part of the same stored message. These are qualitative illustrations, not prevalence estimates. They show individual observed $K=5$ leave-one-out labels; the aggregate matrices support the population-level claims. Throughout this section, a correct-harmful card records a negative matched replay effect in one fixed pool. The label neither attributes the failure to that message alone nor establishes that any particular part of the message is intrinsically harmful.

Omni-MATH-2

Wrong-helpful message.
Dataset / problem identifier: Omni-MATH-2 / omni2:t4:0291.
Message: h_5 , Growth-Bound Analyst.
Problem: For positive integers L , let $S_L = \sum_{n=1}^L \lfloor n/2 \rfloor$. Determine all L for which S_L is a square number.
Ground-truth answer: $L = 1$ or L is even.
Stored agent-message excerpt
 Compute a closed form by separating even and odd L : $S_{2m} = m^2$ and $S_{2m+1} = m(m+1)$. Thus every even case is a square, while the product of consecutive integers $m(m+1)$ is not a square for $m \geq 1$.
Proposed answer (wrong): All even positive integers L .
Observed replay effect: With this message, the full pool returns “ $L = 1$ or L even” (correct); without it, the integrator returns only “ L even” (wrong). The message supplies the decisive even/odd decomposition despite omitting the boundary case $L = 1$ from its own answer.

Correct-harmful message.
Dataset / problem identifier: Omni-MATH-2 / omni2:t4:1010.
Message: h_1 , Inclusion-Exclusion Analyst.
Problem: A cafeteria reports 80 customers with ham and cheese, 90 with ham and tomato, 100 with tomato and cheese, and 20 with all three ingredients. How many customers were there?
Ground-truth answer: 230.
Stored agent-message excerpt
 Treat the three counts as sets and use inclusion-exclusion. If each pairwise intersection is the 20-person triple intersection, then $80 + 90 + 100 - 20 - 20 - 20 + 20 = 230$.
Proposed answer (correct): 230.
Observed replay effect: With this message, the full pool returns 125 (wrong); without it, the integrator returns 230 (correct).

JEEBench

Wrong-helpful message.
Dataset / problem identifier: JEEBench / JEE Adv 2017 Paper 1:19.
Message: h_3 , Energy-Balance Calculator.
Problem: Select the correct statements about an ideal gas under constant-pressure compression, free expansion, reversible isothermal expansion, and reversible adiabatic expansion. The options are A–D.
Ground-truth answer: ABC.
Stored agent-message excerpt
 For an ideal gas, internal energy depends only on temperature. Free expansion is both adiabatic and isothermal, and reversible adiabatic work is less than reversible isothermal work for the same volume change.
Proposed answer (wrong): BC.
Observed replay effect: With this message, the full pool returns ABC (correct); without it, the integrator returns BC (wrong). The message correctly establishes statements B and C but omits A from its own answer; its reasoning still helps the full pool recover all three correct options.

Correct-harmful message.
Dataset / problem identifier: JEEBench / JEE Adv 2016 Paper 1:16.
Message: h_3 , Energy-Unit Converter.
Problem: A ground-state hydrogen atom absorbs light of wavelength $970 \text{ \AA}$. Given $hc/e = 1.237 \times 10^{-6} \text{ eV m}$ and ground-state energy -13.6 eV , how many lines appear in its emission spectrum?
Ground-truth answer: 6.
Stored agent-message excerpt
 Convert the wavelength to about 12.75 eV , identify excitation to $n = 4$, and count all possible transitions using $n(n-1)/2$.
Proposed answer (correct): 6.
Observed replay effect: With this message, the full pool returns 5 (wrong); without it, the integrator returns 6 (correct).

SciBench

Wrong-helpful message.
Dataset / problem identifier: SciBench / scibench:quan:3.35.
Message: h_2 , First-Principles Deriver.
Problem: An electron in a three-dimensional rectangular box with side lengths 5.00 Å, 3.00 Å, and 6.00 Å transitions from the lowest excited state to the ground state. Find the emitted photon frequency.
Ground-truth answer: $7.58 \times 10^{14} \text{ s}^{-1}$.
Stored agent-message excerpt
Use the three-dimensional particle-in-a-box energies and compare the three single-quantum excitations. The lowest excited state raises the quantum number along the longest side; explicitly verify this against all three side lengths before computing $\nu = \Delta E/h$.

Proposed answer (wrong): $1.1 \times 10^{15} \text{ Hz}$.

Observed replay effect: With this message, the full pool returns $7.6 \times 10^{14} \text{ Hz}$ (correct); without it, the integrator produces no valid final answer (incorrect). Although the message chooses the wrong box direction in its own calculation, it supplies the governing equation and explicitly flags the state-ordering check that the full synthesis resolves correctly.

Correct-harmful message.
Dataset / problem identifier: SciBench / scibench:matter:11.3.
Message: h_1 , Particle-in-a-Box Theorist.
Problem: Model the 60 π electrons of C_{60} as particles in a cubic box of side 0.7 nm and predict the wavelength of the first excitation.
Ground-truth answer: $1.6 \mu\text{m}$.
Stored agent-message excerpt
Fill the cubic-box energy levels with 60 electrons, locate the highest occupied and lowest unoccupied shells, and use $\Delta E = h^2 \Delta N / (8mL^2)$ with $\Delta N = 1$.

Proposed answer (correct): $1.6 \mu\text{m}$.

Observed replay effect: With all five messages, the integrator produces no valid final answer (incorrect); after removing this message, it returns $1.6 \mu\text{m}$ (correct). Other messages in the pool advocate roughly 730–800 nm, so the full synthesis faces conflicting level-filling assumptions.

LAB-Bench

Wrong-helpful message.
Dataset / problem identifier: LAB-Bench / SeqQA:de98a3c3-68b4-4b62-b841-6361de7dc6c8.
Message: h_3 , Pattern Analyzer.
Problem (faithful excerpt): Compute the guanine–cytosine (GC) percentage of the supplied 333-base DNA sequence, rounded to the nearest integer. Options: A=50, B=52, C=51, D=26. The sequence itself is omitted here for space.
Ground-truth answer: C (51%).
Stored agent-message excerpt
Frequent GC-rich motifs suggest a slight GC bias above 50%. A full count should verify whether the percentage rounds to 52% rather than 50% or 51%.

Proposed answer (wrong): B (52%).

Observed replay effect: With this message, the full pool returns C (correct); without it, the integrator returns B (wrong). Although the visual estimate overshoots by one percentage point, it directs the integrator toward the relevant near-50% range.

Correct-harmful message.
Dataset / problem identifier: LAB-Bench / SeqQA:0eafde3b-6246-4ff7-9954-2a55c593a086.
Message: h_1 , Sequence Decomposer.
Problem (faithful excerpt): Choose one of four listed primer pairs to amplify the *E. coli ubiI* gene for Gibson assembly into SmaI-linearized pUC19. The long primer sequences are omitted here for space.
Ground-truth answer: A.
Stored agent-message excerpt
Check vector homology around the SmaI site and the gene-specific start and stop regions. Option A provides the expected overlaps and an adenine–thymine–guanine (ATG) start codon for *ubiI*.

Proposed answer (correct): A.

Observed replay effect: With this message, the full pool returns B (wrong); without it, the integrator returns A (correct).

MaScQA

Wrong-helpful message.
Dataset / problem identifier: MaScQA / G-META-18-51.
Message: h_5 , Analogy-Based Chemist.
Problem: For $4\text{Ag} + \text{O}_2 \rightarrow 2\text{Ag}_2\text{O}$, use $\Delta H^\circ = -61080 \text{ J}$, $\Delta S^\circ = -132.22 \text{ J K}^{-1}$, and $p_{\text{O}_2} = 0.3 \text{ atm}$ to find the temperature above which Ag_2O decomposes.
Ground-truth answer: 427–432 K.
Stored agent-message excerpt
Set the Gibbs free-energy change to zero, include the $RT \ln p_{\text{O}_2}$ term, and solve for T . Check the algebraic sign in the denominator $\Delta S^\circ - R \ln p_{\text{O}_2}$.

Proposed answer (wrong): 499.8 K.

Observed replay effect: With this message, the full pool returns 429.5 K (correct); without it, the integrator returns 499.8 K (wrong). The message supplies the correct equilibrium structure even though its own algebraic result is wrong.

Correct-harmful message.
Dataset / problem identifier: MaScQA / G-XEC-2016-19.
Message: h_4 , Estimation and Sanity-Bounding Expert.
Problem: Natural rubber has repeat-unit mass 68 and sulfur has atomic mass 32. If 10 g sulfur is added to 50 g rubber at a 1:1 repeat-unit-to-sulfur ratio, find the maximum percentage of cross-linked sites.
Ground-truth answer: 42–43%.
Stored agent-message excerpt
Convert both masses to moles and divide moles of sulfur by moles of rubber repeat units. Check whether one sulfur atom bridges two sites or satisfies one site under the stated 1:1 ratio.

Proposed answer (correct): Approximately 43%.

Observed replay effect: With this message, the full pool returns 85% (wrong); without it, the integrator returns 42.5% (correct). Its doubling caveat agrees with another pool member’s proposed answer of 85% and may make the wrong doubled calculation dominate the final synthesis.

Controlled Repeated Replay Robustness

The main matrices use one full-pool outcome and one removal outcome per comparison. To test whether their off-diagonal pattern depends only on single draws, we ran the same controlled repeated-replay study for OSS and Gemma at $K=5$. For each model, we sampled 200 problems per benchmark, stratifying Omni-MATH-2 by difficulty tier. Sampling uses source-eligible five-message pools and does not require that the earlier one-draw replay matrix be complete; the direct original-to-repeated comparison below therefore uses their intersection. Every problem has five local replicate blocks containing two full-pool calls with exactly the same input and one call for each of the five removal conditions. Full, removal, and null calls are interleaved within each block so temporal variation is shared across conditions. OSS yields 35,000 valid events; Gemma yields 34,999, with one documented evaluator-parse failure retained as missing rather than imputed.

The run seed fixes problem sampling and call order, but not model decoding. We therefore describe these as controlled repeated calls rather than seeded generations. Within a block, we average the two full-pool correctness outcomes and subtract the matched removal outcome. For descriptive case finding, a problem–message pair is marked interval-separated when a 95% nonparametric bootstrap interval over its five block differences lies wholly above or below zero. Otherwise it is *indeterminate*; this category includes effects that may be absent, small, or unresolved with five blocks and should not be read as neutral. Because these 5,000 per-message intervals are exploratory and not multiplicity adjusted, their counts select candidates for qualitative audit rather than establish a formal number of discoveries.

The five message effects from one problem share the same full-pool outcomes. Table 20 therefore tests the paper’s central wrong-helpful direction after first reducing each problem to its largest effect among messages with wrong proposed answers. The null permutes the seven outcome slots within each matched block, and 1,000,000 permutations estimate each problem-level p -value. Under the global screen, all 11 BH-selected problems come from Gemma and span all five benchmarks. OSS contributes none, and the dependence-conservative BY screen retains none. The problem-level analysis strengthens the within-problem dependence check without changing the aggregate permutation test, which remains the primary cross-model result.

For both models, the total separated count exceeds the within-block permutation control at $p = 0.0002$. The control permutes the seven outcome slots—two full-pool and five removal outcomes—within each problem and matched replay block, then reruns the same interval-selection procedure. Proposal-correctness labels remain attached to their messages. Calls were interleaved, but intervention labels were not originally randomized, so this test assumes within-block slot exchangeability. For each message, a two-sided permutation p -value counts permutations whose absolute trajectory-value estimate is at least as large as the observed estimate, with the standard plus-one correction. The BH and BY corrections use these 5,000 permutation p -values, not the exploratory bootstrap intervals.

The identical-input calls disagree in correctness on 363 of 5,000 pairs (7.3%). The reported rate is an end-to-end replay-stability diagnostic: the rate includes variation in the integrator output and subsequent evaluation and is not a threshold subtracted from trajectory-value estimates. The rate varies from 1.6% on MaScQA to 17.5% on LAB-Bench for OSS. The variation warrants more caution when interpreting individual LAB-Bench replays, but it does not establish that the scientific domain itself is intrinsically less stable.

The OSS intervals separate 170 of 5,000 problem–message pairs, but its $W+$ and $C-$ counts lie inside the permutation control’s 95% count ranges and no individual message survives global multiplicity control. The aggregate relationship is nevertheless clear: Table 18 shows a negative mean effect for messages with wrong proposed answers and a positive one for messages with correct proposed answers. Gemma provides the stronger message-level evidence: Table 19 shows 73 $W+$ effects against a 95% permutation range of 13–38, including 17 BH-selected candidates and 60 under the strict evaluable-output requirement. The dependence-robust BY screen selects none, so the within-block permutation excess is our primary repeated result. By contrast, $C-$ is not above the permutation range in either family; 27 of 40 OSS candidates involve an incomplete evaluable answer. We therefore treat wrong-helpful as the replicated off-diagonal finding and correct-harmful primarily as a whole-message and output-reliability diagnostic. Most messages remain indeterminate at this replication budget rather than becoming neutral.

Matched scheduling and integrator sensitivity. The primary repeated-call study interleaves full-pool and removal calls but does not randomize their order. We therefore select the same 20 frozen problems per benchmark and collect three new blocks with the seven conditions randomly ordered inside each block, comparing them with the five reference blocks. Table 22 shows that the pooled mean shift is small relative to its problem-clustered interval in both families. Fixed call order therefore does not explain the pooled mean result. Exact sign agreement is inflated by the many messages estimated as neutral in both runs, so we separately report direction agreement among messages estimated nonzero in both. We then hold the same problems and frozen Gemma messages fixed while replacing the Gemma integrator with an OSS integrator. The second comparison tests whether message effects remain stable across integrator families; it is not a model-performance comparison.

Changing the integrator family leaves the pooled mean shift statistically unresolved, but the message-level correspondence is weak: trajectory-value estimates have Pearson $r = 0.095$, and only 10 of the 20 messages estimated nonzero in both runs retain the same direction. The OSS-integrator comparison also has higher identical-input disagreement, so it does not isolate a pure model-family effect. The comparison does show that a message’s estimated trajectory value should not be assumed invariant to the integrator. Trajectory value is therefore always defined relative to a fixed message pool and integrator, rather than as an intrinsic property of a message.

Table 17: Controlled repeated replay at $K=5$, with 200 problems per benchmark and model. Disagreement is the share of 1,000 identical-input full-pool pairs with different correctness labels; brackets give Wilson 95% confidence intervals (CIs). Each benchmark also contains 1,000 problem–message estimates. W+/W− and C+/C− count messages with wrong/correct proposed answers whose repeated-call interval lies wholly above/below zero; all other intervals are indeterminate. These intervals are exploratory and unadjusted for multiple comparisons. One Gemma SciBench message has four rather than five removal blocks because one evaluation was missing; no value is imputed.

Model	Benchmark	Disagreement [95% CI]	W+	W−	C+	C−	Indeterminate
OSS	Omni-MATH-2	9.1% [7.5, 11.0]	7	7	23	13	950
OSS	JEEBench	4.1% [3.0, 5.5]	0	7	2	4	987
OSS	SciBench	4.0% [3.0, 5.4]	1	10	7	3	979
OSS	LAB-Bench	17.5% [15.3, 20.0]	14	9	35	16	926
OSS	MaScQA	1.6% [1.0, 2.6]	1	4	3	4	988
OSS	Pooled	7.3% [6.6, 8.0]	23	37	70	40	4,830
Gemma	Omni-MATH-2	1.6% [1.0, 2.6]	42	24	2	0	932
Gemma	JEEBench	1.0% [0.5, 1.8]	5	9	5	0	981
Gemma	SciBench	4.2% [3.1, 5.6]	16	10	2	0	972
Gemma	LAB-Bench	1.9% [1.2, 2.9]	5	5	14	5	971
Gemma	MaScQA	1.6% [1.0, 2.6]	5	9	6	5	975
Gemma	Pooled	2.1% [1.7, 2.5]	73	57	29	10	4,831

Table 18: Mean repeated trajectory-value estimate for OSS by proposal correctness (percentage points). Each message’s estimate averages five matched blocks; 95% intervals hierarchically resample problems and matched replay blocks while keeping each problem’s five messages together. The pooled bootstrap is stratified by benchmark. The final column shows the correct-minus-wrong difference. These are repeated-call estimates, not single-realization rates with the disagreement statistic subtracted.

Benchmark	Wrong-answer messages	Correct-answer messages	Correct − wrong
Omni-MATH-2	0.00 [−2.04, 2.07]	2.62 [0.66, 4.61]	2.62 [0.14, 5.16]
JEEBench	−6.35 [−11.56, −2.00]	0.38 [−0.47, 1.21]	6.73 [2.47, 11.73]
SciBench	−1.69 [−4.06, 0.49]	0.53 [−0.44, 1.56]	2.23 [0.06, 4.56]
LAB-Bench	−1.40 [−3.65, 0.83]	4.19 [1.17, 7.28]	5.59 [2.18, 8.96]
MaScQA	−2.74 [−8.53, 2.07]	0.29 [−0.70, 1.19]	3.02 [−1.64, 8.56]
Pooled	−1.66 [−2.91, −0.45]	1.34 [0.65, 2.02]	3.00 [1.72, 4.28]

Table 19: Cross-model calibration of repeated $K=5$ effects. Brackets give 95% count ranges from 5,000 within-block permutations. Testing 5,000 messages creates many opportunities for chance discoveries, so the Benjamini–Hochberg (BH) column controls the expected proportion of false discoveries among selected candidates. The more conservative Benjamini–Yekutieli (BY) correction allows arbitrary dependence among messages and retains none. Strict counts require evaluable outputs in every arm and block.

Model	Separated [null]	W+ [null]	W− [null]	C+ [null]	C− [null]	BH W+/W−/C+/C−	Strict W+/C−
OSS	170 [83,129]	23 [10,30]	37 [9,26]	70 [20,43]	40 [24,53]	0/0/0/0	7/13
Gemma	169 [45,83]	73 [13,38]	57 [15,42]	29 [0,7]	10 [2,16]	17/20/10/0	60/5

Broader historical sensitivity diagnostic. For completeness, two historical OSS LOO realizations share 34,748 message-removal events over four benchmarks. Their observed replay-effect labels (helpful, neutral, harmful) agree on 88.98% of events, with weighted Cohen’s $\kappa = 0.448$; LAB-Bench has the lowest agreement at 73.36%. All disagreements are between neutral and non-neutral. Direct helpful-to-harmful reversals are impossible in this historical pairing because both realizations reuse the same full-pool outcome. Its incomplete provenance makes it secondary to the controlled study above, but its larger scope independently shows where a single removal replay most often crosses the correctness boundary.

Component Masking: Hiding Reasoning and Answer Fields

Whole-message removal cannot reveal which part of a message carries its trajectory value. Here, *component masking* means hiding either the reasoning or the proposed-answer field while preserving the other field and the message’s position. We apply this test to a *diagnostic sample*: a fixed set of 22 Gemma messages defined from controlled repeated replay before running any masking condition. The effect-stratified sample contains 10 repeatable wrong-helpful anchors, 3 correct-helpful controls, 3 wrong-harmful controls, 2 exploratory correct-harmful messages, and 4 indetermi-

Table 20: Problem-blocked multiplicity check for repeatable wrong-helpful effects. The maximum wrong-helpful statistic first absorbs the five dependent messages in each problem; one-sided permutation p -values are then corrected across eligible problems. “All” applies one global correction across both model families. Coverage counts benchmarks containing at least one BH-selected problem.

Scope	Eligible	Raw $p \leq .05$	BH	BY	Coverage
All models	1,034	38	11	0	5/5
OSS	405	8	0	0	0/5
Gemma	629	30	13	0	5/5

Table 21: Alignment between original wrong-helpful (W+) observations and controlled repeated replay on the overlapping $K=5$ sample. A direction is resolved only when the repeated-call interval excludes zero. Unresolved means the five-block budget does not determine a direction; it is not imputed as neutral. Five sampled Gemma problems absent from the complete headline matrix are excluded from this join.

Model	Original W+	Same direction	Opposite	Unresolved	Same among resolved
OSS	83	2	3	78	40.0%
Gemma	57	17	2	38	89.5%

Table 22: Matched sensitivity over 20 frozen problems per benchmark. Mean shift is the sensitivity run minus its matched reference in percentage points, with a 95% problem-cluster interval. Exact sign includes neutral agreement; nonzero direction conditions on messages estimated nonzero in both runs. Disagreement is the correctness-flip rate between identical-input full-pool calls in the sensitivity run.

Matched change	Messages	Mean shift [95% CI]	Exact sign	Nonzero direction	Disagreement
Order, OSS	500	+0.59 [−1.39, 2.61]	84.2%	55.0%	7.0%
Order, Gemma	500	+0.32 [−1.21, 1.79]	93.0%	68.2%	1.7%
Integrator, Gemma→OSS	500	+1.35 [−1.01, 3.70]	73.8%	50.0%	10.0%

nate controls, spanning all five benchmarks. The design probes mechanisms within established effect categories; it does not estimate their population prevalence. The full message matrices and controlled repeated replay establish the broader scope and repeatability of the phenomenon. The masking sample instead asks the narrower question of which message component carries an already established effect. Each case has five interleaved repetitions of six semantic conditions: the original pool; an identical-input repeat; complete removal of the target message; same-position, approximately same-length neutral replacement; masking only the proposed-answer field; and masking the reasoning fields while retaining the proposed answer. For the two identical-input conditions, we issue two calls per repetition to measure local variation; the other four conditions use one call. The resulting eight outcomes per case and repetition yield $22 \times 5 \times 8 = 880$ valid outcomes.

The wrong-helpful anchor set shows the intended diagnostic contrast. Removing the target message lowers success

by 56 points, while replacing it with same-length neutral text lowers success by 36 points. Hiding only the wrong answer lowers success by 18 points; hiding the reasoning lowers it by 38 points. These comparisons are consistent with the reasoning content carrying more of the observed benefit than the attached wrong answer. They do not provide complete causal decomposition: the neutral replacement is only an approximate footprint control, and the 10-case wrong-helpful anchor subset is small. The control rows support the manipulation but are too small for cell-level inference.

Cross-Fitted One-Removal Opportunity

The matrix from controlled repeated replay lets us ask a practical question without assuming that trajectory value is already predictable on unseen problems: if repeated evidence for the *same* problem were available, could it identify a better integration choice than proposal correctness alone? For each problem, the action set contains the full five-message pool and the five pools obtained by removing exactly one message. In each of five folds, four repeated blocks estimate the accuracy of these six actions and select the best one; the held-out block of that same problem alone scores the choice. Ties conservatively favor the full pool. We rotate the held-out block and cluster bootstrap intervals by problem. No multi-removal bundle is inferred from separate LOO effects.

The cross-fitted choice selects the full pool in 81.3% of OSS folds and 92.5% of Gemma folds, rather than deleting a message by default. Among folds whose two-replicate full-pool average is exactly 0 or 1, the choice produces 99 rescues and 51 harms for OSS, and 121 rescues and 11 harms for Gemma. Folds whose two full-pool replicates disagree have value 0.5 and contribute half-point changes, so the accuracy gain is not simply the rescue-minus-harm count divided by the number of folds. The positive point estimate appears in every benchmark–model cell (Table 25), although the pooled intervals, rather than every individual benchmark, carry the headline inference. Selecting the best held-out action in hindsight gives upper bounds of 79.82% for OSS and 87.37% for Gemma.

Correctness does not recover the same opportunity. For a symmetric comparison, we repeat the same six-action, four-block cross-fitting procedure but allow it to use only proposal-correctness classes, not message identity. The comparator gains 0.94 points for OSS and 1.00 for Gemma, compared with 1.68 and 2.61 from message-specific repeated effects. Two simpler checks give the same ordering: removing the lowest-index message with a wrong proposed answer gains 1.17 and 0.03 points, while uniformly removing such a message gains 1.04 and −0.18. These comparators do not exhaust every possible use of proposal-correctness labels. They test the narrower question needed here: whether proposal correctness alone explains the measured opportunity. The result rejects that explanation.

Shuffled controls and full-arm sensitivity. The primary analysis averages two identical-input full-pool outcomes in each block. We therefore rerun the complete pipeline with either full replicate alone, making every action outcome binary. We also use two block-preserving shuffled controls: one

Table 23: Integrator success (%) in the component-masking diagnostic. Each cell averages five repetitions; the full and repeat columns each additionally average their two identical-input outcomes. Sample membership was fixed from controlled repeated replay before any masking calls, so these rates describe the diagnostic sample rather than population prevalence. The near-identical full and repeat columns provide a local same-input check. For repeatable wrong-helpful cases, hiding the wrong answer preserves more success than hiding the reasoning.

Diagnostic sample (<i>n</i>)	Full	Repeat	Remove	Neutral replacement	Hide answer	Hide reasoning
Wrong-helpful (10)	82	81	26	46	64	44
Correct-helpful (3)	100	100	7	67	87	100
Wrong-harmful (3)	0	0	100	67	27	47
Correct-harmful (2)	100	100	100	100	100	70
Indeterminate (4)	50	53	50	50	55	50

Table 24: Micro-pooled accuracy (%) on a held-out replay block of the same problem for the cross-fitted one-removal opportunity analysis. Gain intervals are problem-clustered 95% confidence intervals (CIs) from the bootstrap.

Model	Full → cross-fit	Gain [95% CI]
OSS	71.85 → 73.53	+1.68 [0.88, 2.50]
Gemma	83.11 → 85.73	+2.61 [1.76, 3.54]

Table 25: Cross-fitted accuracy gain over the full five-message pool by benchmark (percentage points). Gemma SciBench uses 199 complete problems; every other cell uses 200.

Benchmark	OSS	Gemma
Omni-MATH-2	+2.15	+5.05
JEEBench	+2.75	+2.00
SciBench	+1.00	+2.66
LAB-Bench	+1.95	+1.40
MaScQA	+0.55	+1.95

permutes all six action identities, while the stricter removal-identity control keeps the full action fixed and permutes only which message was removed. Table 26 reports the resulting null intervals.

Table 26: Sensitivity of the cross-fitted gain to the full-pool replicate and two block-preserving shuffled-control tests. Gains and null intervals are in percentage points. *All-action* permutes all six actions; *removal-identity* keeps the full action fixed and permutes the five removal identities.

Model / arm	Observed	All-action	Removal identity
OSS / A	+1.70	[−0.78, 0.82]	[−0.50, 0.64]
OSS / B	+1.18	[−0.82, 0.76]	[−0.74, 0.32]
Gemma / A	+2.60	[−0.60, 0.62]	[0.50, 1.26]
Gemma / B	+2.40	[−0.58, 0.64]	[0.42, 1.18]

The gain persists with either binary full arm. For Gemma, the removal-identity null is centered above zero, showing that part of the advantage comes from deciding whether *some* removal is preferable to the full pool. The observed gains nevertheless exceed the removal-identity maxima of 1.50

and 1.40 points. Across all variants, the largest one-sided *p*-value is 0.002. Removal identity therefore carries additional repeatable information. The corresponding OSS null remains near zero.

Scope and missingness. The complete-fold analysis includes all 1,000 OSS problems and 999 of 1,000 Gemma problems. One Gemma SciBench problem lacks one removal judgment because the evaluator repeatedly returned no parseable decision; we retain it as missing and perform no imputation. Most importantly, all folds reuse repeated calls for the same problem. The analysis therefore quantifies a decision-relevant same-problem opportunity under repeated evidence; it does not demonstrate an online selector or generalization to unseen problems.

Integrator Uptake in Matched Replay Pairs

The aggregate results establish that proposal correctness and trajectory value can disagree. To see how that disagreement appears in the integrator’s own output, we audited ten readable JEEBench cases from controlled repeated replay. For each case, we compared the cached message with full-pool and leave-one-out responses from the same matched replay block. The sample contains three helpful observed replay effects for messages with wrong proposed answers, four wrong-answer messages whose repeated effect estimate had a 95% interval below zero, and three correct-answer messages meeting the same repeatable-harmful criterion. The purposive sample exposes recurring mechanisms; it does not estimate their frequency.

We code *reuse and repair* when the full response repeats a distinctive intermediate result from the target message but corrects its proposed answer; *reuse and filter* when it retains a useful constraint while rejecting an extra option; *error propagation* when it repeats the target’s decisive error; and *format or termination failure* when the full response does not reach an evaluable answer. These labels describe visible text correspondences, not token-level attention or causal attribution.

Wrong-answer messages with helpful observed replay effects. These three cases compare a correct full response with an incorrect removal response in the same matched replay block. They are readable helpful effects, not claims of a repeatable per-message effect.

Polymer classification (2016 P1 Q23, H3). The message derives the saturated repeat structure but incorrectly labels

it polybutylene (D). The full response retains the structure, remaps it to ethylene-propylene, and returns A; the reduced response returns D. *Visible pattern: reuse and repair of a classification.*

Peroxide counting (2020 P2 Q22, H3). The message identifies CrO_5 and two peroxide ligands but proposes 2. The full response retains that structure, recomputes $2 \times 2 = 4$, and returns 4; the reduced response returns 2. *Visible pattern: reuse and repair of a local count.*

Boundary filtering (2016 P2 Q47, H4). The message derives $a(x^2 + y^2) = x$, identifies the $b = 0$ boundary case, and proposes ABCD. The full response returns ACD, whereas the reduced response returns AD. *Visible pattern: reuse of boundary case C while filtering option B.*

Repeatable wrong-harmful messages. For each case below, the 95% interval of the repeated full-minus-removal estimate lies below zero.

Peroxide-bond count (2020 P2 Q22, H1). The message uses an incorrect bond count and proposes 2. The full response repeats the structure and returns 2; the reduced response recognizes two side-on peroxides and returns 4. *Visible pattern: propagation of the local counting error.*

Electrolysis option (2020 P2 Q27, H2). The message wrongly rejects statement D about the Hall-Héroult cathode and proposes ABC. The full response repeats that rejection and returns ABC; the reduced response returns ABCD. *Visible pattern: propagation of a wrong option judgment.*

Screw-gauge conversion (2022 P2 Q35, H1). The message uses a 1.0 mm pitch and 0.01 mm least count, then proposes D. The full response repeats both values and returns D; the reduced response uses 0.5 mm and 0.005 mm and returns C. *Visible pattern: propagation of a wrong unit conversion.*

Reaction mapping (2022 P2 Q52, H5). The message proposes $2\text{ClO}_2 + \text{O}_3 \rightarrow \text{Cl}_2\text{O}_7$ and D. The full response reproduces the reaction and D; the reduced response returns C. *Visible pattern: propagation of a wrong reaction mapping.*

Repeatable correct-harmful messages. These cases show that a correct proposed answer can coincide with a negative downstream effect for different reasons.

Option omission (2019 P1 Q29, H3). The message correctly proposes ACD. The full response returns only CD, whereas the reduced response explicitly restores A and returns ACD. *Visible pattern: the negative effect is observable, but the omission cannot be localized to a specific phrase.*

Incomplete derivation (2021 P1 Q49, H2). The message correctly proposes ABC. The full response ends during its derivation without an evaluable answer; the reduced response completes the derivation and returns ABC. *Visible pattern: output termination rather than semantic reversal.*

Incomplete functional analysis (2020 P2 Q48, H1). The message derives $g(m, n) = 2^{m+n}$ and correctly proposes ABD. The full response ends mid-analysis without an evaluable answer; the reduced response returns ABD. *Visible pattern: integration changes output reliability rather than the truth of the supplied result.*

Across the ten audited cases, all three helpful cases visibly reuse a message-specific intermediate and then repair or filter its local answer. All four repeatable wrong-harmful cases

reproduce the target message’s decisive error. The correct-harmful cases are more heterogeneous: one omits a supported option, while two fail to produce an evaluable final answer. The off-diagonal labels can therefore arise through semantic reuse, semantic interference, or output reliability. Trajectory value remains an outcome-level measure: it records whether the message helped the final trajectory without pretending that proposal correctness or lexical overlap alone explains the mechanism.

We do not treat token-level attention as a causal explanation. The component-masking study in Appendix complements this trace audit by preserving message position and approximate length while separately masking reasoning and proposed-answer fields. Its diagnostic sample, fixed before masking outcomes were observed, provides initial component-sensitive evidence consistent with benefit from the reasoning fields, but the neutral replacement is only an approximate footprint control and does not provide a complete causal decomposition.

Evaluation Procedure and Artifact Schema

Constrained evaluator. Both gpt-oss-120b and gemma-4-31B-it model families use a separate gpt-oss-120b evaluator. The evaluator sees only the submitted answer and an evaluator-only ground truth, never agent reasoning, and checks equivalence rather than solving or selecting messages. The fixed procedure labels independent, proposed, single-message, full-pool, and reduced-pool answers.

Generation and evaluation settings. The two model families use the same role-specific decoding settings. Hypothesizers use nonzero temperature to produce complementary solution paths; the recruiter, integrator, and evaluator use temperature zero. The maximum completion budgets in Table 27 are output-token limits, not context-window truncation rules. Every message in the headline LOO matrices has a nonempty proposed-answer field (32,795 OSS and 29,650 Gemma); 99.8% and 98.7%, respectively, also have complete structured fields. Missing final answers count as incorrect, while unmatched records are excluded rather than relabeled.

Table 27: Role-specific decoding settings shared by both model families.

Role	Temperature	Maximum completion tokens
Recruiter	0.0	2,048
Hypothesizer	0.7	4,096
Integrator	0.0	4,096
Evaluator	0.0	4,096

Measurement call accounting. For a fixed K -message pool, all in-pool LOO effects reuse one full-pool integration and require one additional integration for each of the K message removals. The repeated $K=5$ study therefore uses seven integrator outcomes per block—two identical-input full-pool calls and five removals—and five blocks, for 35 outcomes per problem. Each outcome is judged under

the same evaluation procedure. The component-masking diagnostic separately contributes 880 integrator outcomes (22 cases $\times$ 5 repetitions $\times$ 8 outcomes). Once these outcomes are saved, bootstrap, permutation, masking-summary, and selection analyses make no additional model calls. These counts describe measurement effort, not the cost of a deployment protocol.

Context-indexed artifact records. Each packaged signal-level record retains the anonymized problem key, generating-model and integrator-model families, pool size, message position, proposal-correctness judgment, full- and reduced-pool judgments, observed replay effect, repeated-effect fields when available, evaluator metadata, and missingness status. The pool and integrator fields are part of the label definition: trajectory value is not encoded as a universal annotation of the message text. The artifact therefore supports reproducing the present measurements and training future context-aware selectors without assuming that a label transfers unchanged to a different integrator.

Reproducibility materials. The ancillary archive `anc/reproducibility_artifact.zip` contains the exact prompt and protocol configurations, analysis scripts, sanitized derived records, figure inputs, and checksums used in the paper-facing reproduction tests. It deliberately excludes third-party benchmark question text and raw model messages whose redistribution terms require separate handling.

Cross-evaluator agreement. We test the answer-equivalence evaluation procedure by replaying 16,724 saved submissions with three open evaluator models: `gpt-oss-120b`, `Meta-Llama-3.1-70B-Instruct` (Llama), and `gemma-3-27b-it` (the Gemma-3 evaluator). Every evaluator receives the same submitted answer, ground truth, prompt, and parser; none sees a reasoning trace. Pairwise agreement is 94.2–96.6%, with chance-corrected $\kappa = 0.850$ – 0.915 (Cohen 1960) and 99.45–99.83% valid coverage. The replay tests the shared answer-comparison procedure rather than protocol behavior; a small set of equivalence decisions remains evaluator-sensitive.

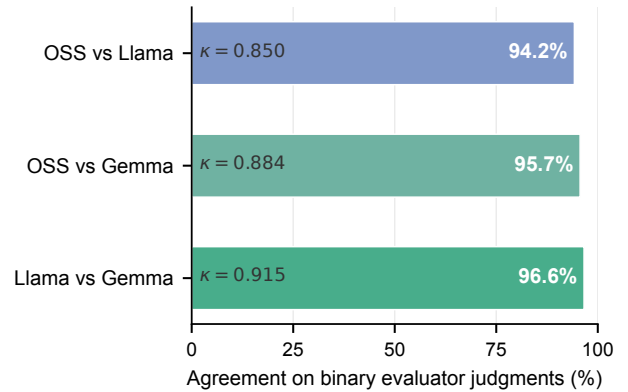


Figure 9: Pairwise agreement among three large language model (LLM) evaluators under the same answer-equivalence evaluation procedure ($N = 16,724$ saved submissions). Percentages are computed on instances with valid judgments from both evaluators; Cohen’s κ is shown inside each bar.

Compound-label agreement. The six-cell label combines three evaluator decisions: the proposed answer, full-pool answer, and reduced-pool answer. To test how this composition changes agreement, we draw 150 triplets per model family, balanced across benchmarks and the original six categories, and ask the same three evaluators to judge all three answers independently. A tuple agrees only when all three judgments and the induced replay direction agree. Table 28 shows high proposed-answer agreement and lower exact-tuple agreement, as disagreement can enter through any decision. Every evaluator reconstructs wrong-helpful and correct-harmful cases in both model-family samples. Because the sample is category-balanced, these rows measure agreement rather than prevalence; persistently unparseable judgments remain missing.

Table 28: Cross-evaluator agreement on balanced proposed-answer/full/reduced triplets. Ranges are the three pairwise evaluator comparisons. Δ is the induced helpful, neutral, or harmful replay direction; tuple agreement requires proposal correctness, both outcome judgments, and Δ to match. N counts triplets with valid judgments from all three evaluators; all agreement columns are percentages.

Model family	N	Proposed answer	Δ	Pairwise tuple	Three-way tuple
OSS	140	93.3–96.4	79.4–85.2	71.6–78.6	67.1
Gemma	148	95.3–96.7	88.5–92.7	82.4–88.7	77.7