MULTIPLE TASK-STATE DESCRIPTIONS INTERFERE WITH IN-CONTEXT COMPUTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Adding a second description of a task state lowers language-model accuracy on a downstream question about it, even when the two agree and even when one is an exact copy of the other. When one description updates the other, the model fails on most items where it can separately resolve the current state and compute from it, once it must do both together. We call this a composition penalty. Neutralizing one token's representation in the obsolete description restores most of the lost accuracy, showing that the obsolete state remains causally active. Duplication and stale-state costs recur in verbatim shell-tool output, and on an independently written benchmark the penalty persists on items whose state the model resolves correctly, including for Claude Opus 5.5, which solves every controlled chain. In nineteen novels, deleting the sentences that support an obsolete state improves accuracy more than deleting as much other text. Reasoning closes some controlled gaps but leaves a gap on long histories. Compiling the current state into a single description before computing nearly eliminates the penalty in the controlled setting. Its end-to-end benefit depends on compilation fidelity: on long histories, inaccurate compilations can outweigh the benefit and reduce overall accuracy.

1 INTRODUCTION

Language-model systems rarely operate on a single, settled description of the world. Retrieval returns passages that disagree with newer ones (Lewis et al., 2020; Chen et al., 2022; Yuan et al., 2026), published facts go stale between training and deployment (Liška et al., 2022; Kasai et al., 2023; Vu et al., 2024), tool outputs supersede earlier observations (Yao et al., 2023; Liu et al., 2024b), users amend requirements mid-task (Laban et al., 2026), and agent memories retain both a fact and its later revision (Packer et al., 2023; Park et al., 2023). The usual assumption is that once newer information is supplied, a competent model will use it. **Yet even repeating an unchanged report can make the model less accurate.** Consider a report listing four sensors with a status and a tier. We ask how many are critical: those whose status is FAIL and whose tier is primary (Appendix A, Figure 5). Showing the same report twice lowers accuracy from 0.992 to 0.886 on Qwen3-8B and from 1.000 to 0.773 on Llama-3.1-8B-Instruct. Both copies are identical and correct. Compiling them into one description restores accuracy; an extra call that restates both copies does not.

Prior work studies feedback acknowledged but not acted on (Jiang et al., 2025; Huang et al., 2024; Kamoi et al., 2024), in-context updates that fail to propagate through multi-step reasoning (Feng et al., 2026; Longpre et al., 2021; Xie et al., 2024; Xu et al., 2024), weight edits that do not reach dependent facts (Meng et al., 2022; Zhong et al., 2023; Cohen et al., 2024; Hua et al., 2024), and distraction by irrelevant material (Shi et al., 2023; Liu et al., 2024a; Levy et al., 2024) (reviewed in Appendix P). Two explanations recur: having to decide which information governs, and the generic cost of extra context. Neither explains our controlled results (Section 3.1). With a state correction instead of an identical copy, Qwen3-8B accuracy falls to 0.28. The model can report the corrected value, yet removing its previous answer worsens the failure. The context still holds two state descriptions. We call this a *composition penalty*: a loss from combining state resolution and downstream computation despite success at each separately. Figure 1 outlines our investigation.

First, we isolate the phenomenon. Each task state is sampled independently per item, so there is no item-specific fact to retrieve from model weights. We keep an item only if the model answers correctly from the original state and also when the corrected state is presented from the start.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

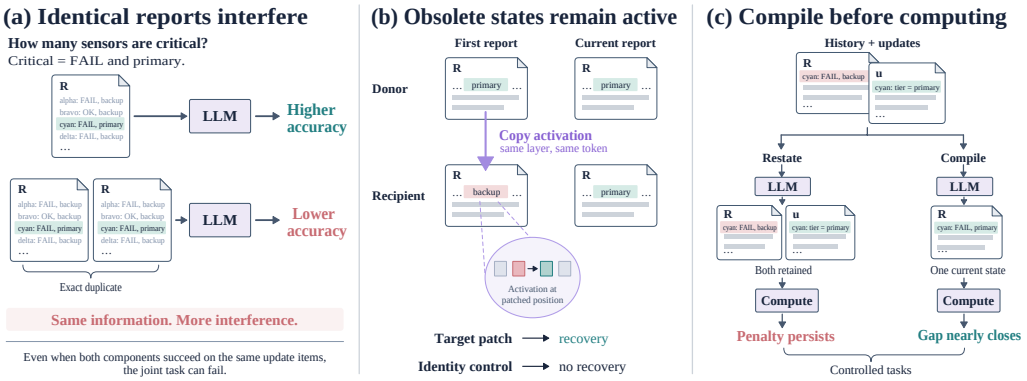

Figure 1: State composition and recovery (schematic). R: report; u: update. **(a)** Duplication hurts accuracy; on update items, state resolution and computation can succeed separately but fail together. **(b)** Copying activations from donor to recipient restores accuracy; identity controls do not. **(c)** Restating retains the penalty; compiling one current state nearly closes the controlled gap.

A later failure to use the update therefore cannot be attributed to missing knowledge or to inability to perform the downstream computation. Under these controls, all ten checkpoints across five model series apply an update that names the answer on every eligible item, and one that names the state variable on at most 0.74 of them. The gap also appears in base checkpoints of Olmo-3 (Team Olmo et al., 2025) and Llama-3.1 (Grattafiori et al., 2024). Narrowing further, on three-update chains, to items where the model both writes the current state correctly and computes correctly from it, joint accuracy is 0.445 for Qwen3-8B and 0.289 for Llama-3.1-8B-Instruct. Duplication and stale-listing costs recur in verbatim shell-tool output, on text no one wrote. Claude Opus 5.5 answers all 300 three-update chains correctly, yet loses accuracy on an independent benchmark's long histories.

Second, we test the obsolete description's causal role. Recognition failure, prior commitment, recency, instruction following and generic context length cannot explain the full pattern. Using activation patching (Vig et al., 2020; Meng et al., 2022; Wang et al., 2023) with controls (Makelov et al., 2024; Zhang & Nanda, 2024), replacing the representation at the obsolete-state token restores most of the lost accuracy. A control copying unchanged prefix activations has no effect. The conclusion does not depend on our items or on activation patching. In nineteen novels, deleting the sentences a benchmark annotates as supporting a superseded state raises accuracy for all three models, and by more than deleting an equal amount of other text. The obsolete state still drives the answer; whether duplication shares its mechanism remains open.

Third, we test a factorized solution. Because joint accuracy falls below both components measured alone, we factorize the task into state resolution and computation, having the model compile the context into a single current-state description, replace the context with it, and only then ask the question. In the controlled setting this restores accuracy to the compute-only level, whereas spending a second model call without resolving the state does not, and neither does supplying the full derivation while the obsolete state remains. On an independently written benchmark, the pipeline performs better on items with faithful compilations and worse on the rest, so its net effect is negative for readers whose compilations mostly fail and positive for one whose compilations mostly succeed. Factorization removes the extra burden of computing amid coexisting descriptions. Its end-to-end benefit depends on compilation fidelity: a wrong compiled state leaves the second call without the original history to recover from.

2 A TESTBED THAT SEPARATES STATE RESOLUTION FROM COMPUTATION

Task. Each item presents four sensors, each with a binary status and a binary tier, and a rule: a sensor is critical if its status is FAIL and its tier is PRIMARY. The question asks how many sensors are critical, and the four options map counts 0–3 to letters through a per-item random bijection (count 4 is excluded), so no fixed old-to-new answer-letter mapping holds across items. States are generated independently per item, preventing advance memorization of item-specific facts, unlike settings built

on real-world entities (Longpre et al., 2021; Feng et al., 2026). Appendix A verifies the generated set; Figure 5 illustrates one current state.

Updates. An update flips exactly one attribute of one sensor and changes the count, so the correct letter differs before and after. On the item of Figure 5 the correction changes cyan's tier from backup to primary, making cyan critical and moving the answer from B to D. No-effect updates flip an attribute without changing the count, giving items on which a correct answer must *not* move: correcting a sensor that already fails the rule on its tier leaves both the count and the answer where they were. Appendix A.1 lists every condition verbatim.

Two capability filters. For single-update behavior, we retain only items answered correctly from the original premise and from the corrected premise. The second filter certifies, per item, that the model can compute the post-update answer, so a later failure to produce that same answer cannot be attributed to the arithmetic, the rule, or the option mapping. The filters do not manufacture the effect: it is present on items where the model demonstrably performed each operation alone (Section 5).

The quantity we measure. Write z for a task state, u for an update, and $z' = u(z)$ for the updated state. Let f map a state to the task's answer, and let $M(\cdot)$ be the model's answer given a context. Three accuracies are available on the same item; the first two are components of the third:

$$\alpha_{\text{res}} = \Pr\left[\,\widehat{z}(z,u) = z'\,\right], \quad \alpha_{\text{cmp}} = \Pr\left[\,M(z') = f(z')\,\right], \quad \alpha_{\text{joint}} = \Pr\left[\,M(z,u) = f(z')\,\right], \tag{1}$$

where $\widehat{z}(z,u)$ is the state the model writes when asked to resolve z and u without answering. The third asks for both at once, from a context that holds z and u side by side. We measure it by the *component-reference gap*, the shortfall of the joint quantity below the weaker of its two components,

$$\Delta \;=\; \min\{\alpha_{\text{res}},\, \alpha_{\text{cmp}}\} \;-\; \alpha_{\text{joint}}, \tag{2}$$

so $\Delta > 0$ says the joint condition falls below both components measured alone, a compositionality gap in the sense of Press et al. (2023). The single-update filters fix $\alpha_{\text{cmp}} = 1$; the unfiltered chains in Section 5 need not.

Measurement and models. Single-update behavioral answers use the logits of four single-token options (Brown et al., 2020), without sampling, generation or answer parsing; divergent decoding cannot confound these comparisons. Compilation and pipeline evaluations instead generate and parse text (Section 5). The controlled experiments test Qwen3-4B and Qwen3-8B (Yang et al., 2025), Qwen3.5-9B, Llama-3.1-8B-Instruct (Grattafiori et al., 2024), the Olmo-3 post-training sequence (Team Olmo et al., 2025) and the Tülu-3 sequence (Lambert et al., 2025), using Wolf et al. (2020) and Kwon et al. (2023). Qwen3.5-9B also runs Sections 6, 7 and 9; Section 7 adds DeepSeek-V4-Pro/Flash and Claude Opus 5.5, and Section 9 two hosted readers.

3 Coexisting states degrade direct computation

Table 1 reports the two channels through which an update can reach the answer. When the update names the answer, every checkpoint applies it. When it names a state variable, none does so reliably, with rates from 0.26 to 0.74 across the ten checkpoints.

Table 1: Answer-naming updates are always applied; state-variable updates are not. TOLD adds the answer to the correction, contrasting delivery, not capability; STATE gives only the correction. SPURIOUS measures changing a correct answer after an irrelevant update; selectivity is STATE minus SPURIOUS. Appendix B includes all ten checkpoints and the Olmo-3 rungs.

Checkpoint	n	TOLD	STATE	SPURIOUS	Selectivity
Qwen3-4B	571	1.000	0.264	0.201	$+0.063$
Qwen3-8B	599	1.000	0.277	0.192	$+0.085$
Llama-3.1-8B-Instruct	353	1.000	0.635	0.739	-0.105
Olmo-3-7B-Instruct	311	1.000	0.714	0.688	$+0.026$
Qwen3.5-9B	783	1.000	0.737	0.474	$+0.263$

In a second task family, the question asks which qualifying item appears first rather than how many qualify. On Qwen3-8B and Llama-3.1-8B-Instruct, answer-naming accuracy is 0.963 and 1.000, versus 0.168 and 0.344 for state-based revision (Table 7).

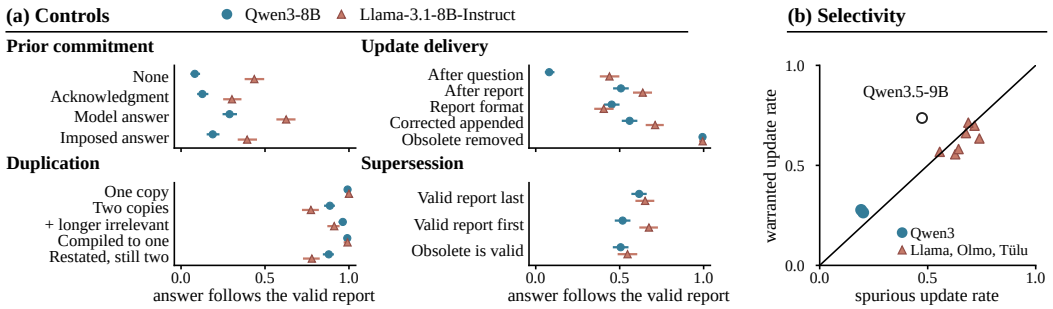

Figure 2: **(a)** Controls for prior commitment, delivery, duplication and supersession, with Wilson intervals on both checkpoints; Table 11 defines each condition. **(b)** Warranted updating versus changing a correct answer after an irrelevant update, one point per checkpoint. Nine lie near the diagonal, low or high on both axes; only Qwen3.5-9B separates the cases.

On this task, selectivity turns negative for Qwen3-8B, from $+0.085$ to -0.030, and stays negative for Llama. The instruction to recompute remains inert for Qwen3-8B at 0.161, but raises Llama-3.1-8B-Instruct from 0.344 to 0.483, the only setting where it helps substantially.

The two failure modes are opposite and equally uninformative. Qwen checkpoints rarely revise; Llama and Olmo checkpoints revise often but change their answer just as often when the update is irrelevant to it. Selectivity spans -0.105 to $+0.085$ on nine of the ten, which apply warranted and unwarranted updates at nearly the same rate. Qwen3.5-9B is the exception at $+0.263$, and it carries the largest composition penalty we measure (Section 5). Revision alone (the STATE rate) would rank Olmo-3-7B-Instruct at $2.6\times$ Qwen3-8B, yet their selectivities are only $+0.026$ and $+0.085$.

The update is read, and asking for a recount does not help. Asked, in the same conversation, what the corrected attribute now is, Qwen3-8B is correct on all 599 items. Having the model state that value itself before the question moves revision from 0.289 to 0.302, so the corrected value's absence from context is not the obstacle. An instruction to recount moves Qwen3-8B from 0.277 to 0.270.

Post-training does not create the gap. On the items that every rung of the Olmo-3 sequence handles correctly, revision rises from 0.613 at the base model to 0.737 after preference tuning (Ouyang et al., 2022; Rafailov et al., 2023), and selectivity from -0.095 to $+0.248$, while TOLD stays at 1.00 throughout (Appendix D). Post-training improves selective revision without closing the gap, so the behavior does not originate in preference optimization (Sharma et al., 2024; Perez et al., 2023).

3.1 WHAT THE PENALTY IS NOT

Not commitment to a previous answer. Removing the model's prior answer and placing the update in the same turn as the question lowers revision to 0.082 and 0.436 on the two checkpoints, below the 0.289 and 0.626 obtained when the model has committed. Forcing the model to assert a letter it never derived, then updating, leaves that letter the rarest of the imposed, original and updated letters for both models, 0.032 and 0.190, while the premise-consistent letter takes 0.743 on Qwen3-8B. The model returns to the premise-consistent answer instead of repeating the imposed one, which distinguishes the effect from the degradation reported for extended dialogue (Laban et al., 2026).

Not the form or the position of the update. Rendering the correction in the report's own line format at the premise site is no better than prose (0.452 versus 0.508, and 0.405 versus 0.637), and no placement or format approaches the ceiling. Interaction-level remedies that edit the prior response rather than the premise (Choi et al., 2025) are therefore addressing a different site.

Not generic context length. A second copy of the corrected report costs more than an irrelevant block that is slightly longer and shares the report's sensor names and line format (Figure 2a, duplication rows). Irrelevant material is comparatively harmless, in contrast to settings where irrelevant context is itself the manipulation (Shi et al., 2023; Levy et al., 2024). The separation is this clean only here: on real tool output the two costs are close (Section 6).

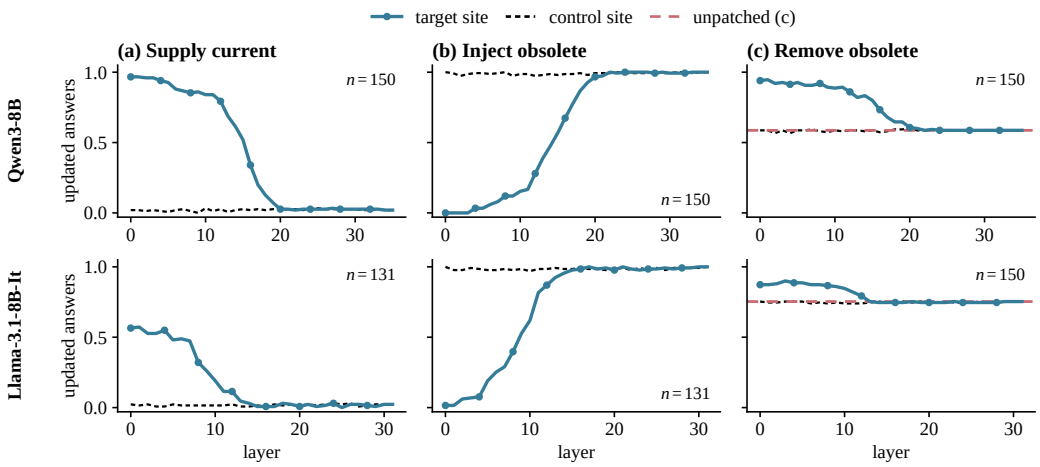

Figure 3: The obsolete state remains causally active in both models and all three directions. **(a)** Patching the current value into a failing run's premise token restores the updated answer. **(b)** Injecting the obsolete value into a current-state-only run destroys that answer. **(c)** With both states present, neutralizing one token of the obsolete block raises accuracy without adding information: the second block already gives the current value. Copies from the shared prefix, 40 tokens earlier, stay at the unpatched rate in every panel. Section 4 gives rates and the (a)/(b) stratum.

Not conflicting information. The strongest of these controls is the duplicated report, in which the two descriptions are identical and both correct, and accuracy still falls. Compiling the copies into one restores accuracy; a second call that preserves both does not (Figure 2a; Appendix C).

Not only a problem of authority. Presenting the two states as timestamped reports and stating that only the most recent is valid yields 0.616 on Qwen3-8B. Reversing which report is authoritative leaves Qwen3-8B at 0.530 among items where it selects one of the two, indistinguishable from chance ($p = 0.17$), while Llama-3.1-8B-Instruct reaches 0.645 ($p < 10^{-4}$; Appendix C). Deciding which source governs is a separate and unsolved problem (Chen et al., 2022; Yuan et al., 2026).

4 The obsolete state remains causally active

We construct two prompts that differ by exactly one token: one holds an obsolete report followed by a corrected report, the other holds the corrected report twice. The token sequences are equal in length and identical at every position except the attribute inside the first block, which Appendix E verifies item by item. At that position, we replace the first run's hidden activation (residual stream) with the second's, neutralizing the obsolete state without adding information.

Figure 3c reports the result. On Qwen3-8B accuracy rises from 0.587 to 0.947 for interventions up to layer 11 and decays to the unpatched baseline by layer 20; on Llama-3.1-8B-Instruct it rises from 0.753 to 0.900 and decays by layer 13. The transition sits at 44% and 30% of depth. Both models' shared-prefix controls stay at baseline (Appendix E).

Because the manipulation removes rather than adds, and because the donor comes from the same item at the same position and layer under a context differing by one token, the effect cannot be read as the transfer of an answer: the current value is already in the recipient's second report. That distinction is not automatic: patching the final answer position instead reproduces a sharp and entirely misleading localization, reported in Appendix F with the control that exposes it (Makelov et al., 2024).

The two single-state directions also replicate on items retaining the pre-update answer (STAY; panels a,b). This outcome is rare for Llama-3.1-8B-Instruct (4.2% of eligible items) but common for Qwen3-8B (62.9%); total error rates are 36.5% and 72.3%, respectively (Table 9). For Llama we patched 131 items from an enlarged pool of 9,000, whose eligibility rate (38.8%) matches the original (39.2%; Appendix E).

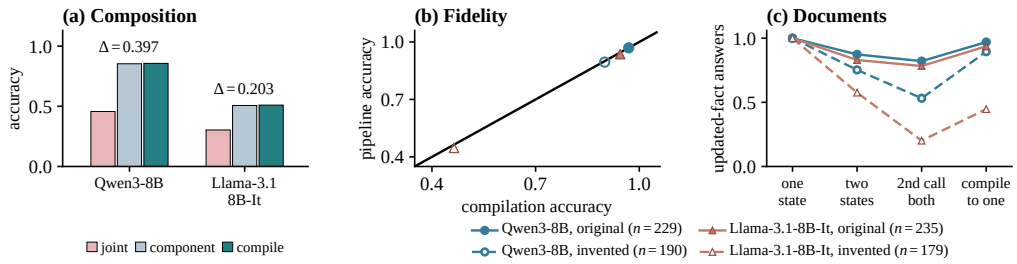

Figure 4: Factorizing state resolution from computation removes the penalty, bounded by compilation. **(a)** On three-update chains, joint accuracy falls Δ below the weaker component; compiling before computing closes that gap. **(b)** In four document settings, pipeline and compilation accuracies differ by at most 0.017: incorrect rewritten sentences undermine the pipeline. **(c)** Coexistence hurts accuracy on HotpotQA's distractor setting: another entity replaces the answer span in its source sentence, and the true sentence is supplied as an update. Dashed lines replace the answer entity everywhere with an invented name of the same type, preventing recovery from weights (Appendix L).

Table 2: The failure is compositional: compiling the current state and computing from it each approach the model's ceiling; joint performance is far below, and factorization recovers the loss. D2/D3 add a model call; D4 copies input by code; all retain the obsolete state. D1 replaces the context with the model's own compiled state, not an oracle. $n = 300$ per cell; Appendix H.

	Qwen3-8B	Llama-3.1-8B-It	Qwen3.5-9B
A compile only, exact state match	1.000	0.830	0.983
B compute only, from the correct state	0.853	0.507	0.893
C jointly, from report and updates	0.457	0.303	0.190
D1 compile, then compute	**0.857**	**0.510**	**0.887**
D2 second call restates the input verbatim	0.477	0.377	0.190
D3 second call produces irrelevant output	0.513	0.287	0.237
D4 the restatement is produced by code	0.387	0.243	0.230

5 COMPILE, THEN COMPUTE

Because joint accuracy falls below both components measured alone, we separate them. We construct chains of three successive updates, some touching the same variable twice, and measure four quantities on all 300 items without capability filtering.

The pipeline reaches the component reference. Using Equation 2, Table 2 gives a component-reference gap of $\Delta = 0.397$ for Qwen3-8B, $\Delta = 0.203$ for Llama-3.1-8B-Instruct (both in Figure 4a) and $\Delta = 0.703$ for Qwen3.5-9B, and after factorizing it falls to within 0.007 of zero in all three. Holding precision fixed, Qwen3-32B widens Δ over Qwen3-8B by 0.213 $[+.120, +.307]$ by improving the component and degrading the joint condition (Appendix K). An answer that cannot be parsed counts as wrong throughout, so every cell has denominator 300.

The gap does not depend on which reference we use. A stricter reference that treats the two component successes as independent, rather than taking the weaker of them, gives the same conclusion (Appendix H). Neither reference is needed to see the effect. Restricting to the items where the model both wrote the correct state and answered correctly from the correct state, it answers the joint question correctly on 0.445 $[.385, .506]$ of them for Qwen3-8B ($n = 256$), 0.289 $[.212, .369]$ for Llama-3.1-8B-Instruct ($n = 128$), and 0.175 $[.130, .223]$ for Qwen3.5-9B ($n = 263$).

A second call is not the active ingredient. D2/D3 add a model call; D4 copies by code. All retain the obsolete state; none approaches D1, and two fall below the single-call baseline. These controls distinguish state consolidation from gains due to intermediate tokens (Nye et al., 2022; Pfau et al., 2024; Goyal et al., 2024). Against a relevance-based System 2 Attention (S2A) adaptation (Weston & Sukhbaatar, 2023), compilation gains 22.7 and 19.7 percentage points on Qwen3-8B and Llama, respectively, without truncation; both paired intervals exclude zero (Appendix Q).

Table 3: Every listing is verbatim command output. C holds stale listing, commands and fresh listing; C_NO_STALE drops only the stale listing; C_IRRELEVANT substitutes a token-length-matched listing of an unrelated directory; C_DUP contains only two copies of the fresh listing. Contrasts use C_NO_STALE, except duplication uses C_FRESH. Opaque directory names identify sources without labeling relevance. $n = 340$ eligible of 600; full-population entries differ by at most 0.05. Intervals: 95% paired bootstrap (Appendix M).

	accuracy	contrast
C_FRESH fresh listing only	0.762	
C_NO_STALE commands, fresh listing	0.932	
C_IRRELEVANT unrelated listing added	0.626	−0.306 [−.365, −.247]
C stale listing added	0.512	−0.421 [−.479, −.362]
C_DUP fresh listing shown twice	0.441	−0.321 [−.376, −.265]

For Qwen3-8B, supplying the reasoning does not close the gap: placing the full per-sensor derivation under the corrected state into the assistant turn reaches 0.653 while removing the obsolete state reaches 0.993. A supplied assistant prefix of the same length, carrying no task content, reaches 0.472, so much of any gain from filling the assistant turn is not semantic (Appendix G).

On chains of up to seven updates the gap persists and grows with the number of coexisting descriptions when the answer is one scored token, and unconstrained reasoning removes it; Claude Opus 5.5 likewise answers all 300 three-update chains correctly (Appendix J). Real documents also show the coexistence penalty (Appendix L). On matched single-update items, explicit reasoning raises Qwen3-8B accuracy from 0.290 to 1.000 ($n = 200$; Appendix I).

6 THE SAME COST APPEARS IN REAL TOOL OUTPUT

Everything so far describes state in prose a generator wrote. An agent reads tool output instead, and reads it repeatedly, so the question is whether a stream of verbatim command results carries the same cost. We built a workload in which nothing about the state is authored: we create a directory on disk, run `ls -l` and keep its stdout exactly as printed, apply `chmod`, `truncate` and `mv`, and run `ls -l` again. The question asks how many files are both larger than a stated size and group-writable, answered from the filesystem rather than from either listing. Qwen3.5-9B reads 600 items, balanced across the four options. An agent context accumulates: a listing, the commands it issued, then a fresh listing. Two descriptions of one directory sit in it, one stale and one current, both true when taken.

We find that showing the current listing twice costs 0.321, with the two blocks byte-identical to each other. The effect of Section 3 is not an artifact of authored prose.

Adding the stale listing to a context that already contains the fresh one costs 0.421. A matched listing from an unrelated directory costs 0.306, so most of that 0.421 is also paid for an unrelated listing and the remaining 0.115 is the excess from describing this directory earlier. That excess excludes zero but is the smaller part; in the controlled setting the matched irrelevant block was far milder. The command trace is not the problem. Adding it to the fresh listing *helps*, from 0.762 to 0.932.

Compiling first recovers 0.138 here, less than in the controlled setting. The reason is visible in the components: asked for the current directory contents in a fixed schema the model is exactly correct on 0.438 of items, below even the joint condition's 0.512, against 0.782 for answering from a supplied correct state. The losses above are therefore duplication and stale-listing costs, not a positive gap of Equation 2; Section 8 relates this contrast to compilation fidelity.

7 AN INDEPENDENTLY WRITTEN BENCHMARK

The workloads so far are ours. STALE (Chao et al., 2026) is not: 400 expert-reviewed scenarios, each with a 50-session history, an earlier memory statement, a later one, a description of the change, and queries written by its authors. We use its Implicit Policy Adaptation probe, which is open-ended, so the answer format shares nothing with our four options. We take all of that as given and add only the decomposition. Four models are run: DeepSeek-V4-Pro with its reasoning mode on and off, Claude

Table 4: History versus resolved state. The first four read a deterministically compacted history that preserves the two evidence sessions the benchmark names; the fifth reads the unmodified history. B is at ceiling everywhere; C is below it everywhere. 400 scenarios attempted per condition; unscorable responses are excluded, not counted wrong. Brackets: 95% paired bootstrap intervals (Appendix M).

	B	C	B − C	D1 − C
DeepSeek-V4-Pro, reasoning on	0.997	0.734	+0.264 [+.221, +.307]	−0.106 [−.161, −.050]
DeepSeek-V4-Pro, reasoning off	0.998	0.632	+0.365 [+.318, +.412]	−0.068 [−.122, −.013]
Claude Opus 5.5, reasoning on	1.000	0.900	+0.101 [+.073, +.131]	+0.038 [+.005, +.070]
Qwen3.5-9B	0.990	0.562	+0.427 [+.380, +.475]	−0.125 [−.182, −.070]
DeepSeek-V4-Flash, full history	0.997	0.519	+0.477 [+.430, +.528]	−0.058 [−.111, −.003]

Opus 5.5 with adaptive reasoning at default effort, Qwen3.5-9B, and DeepSeek-V4-Flash on the benchmark's full histories.

We keep the four quantities as defined: A asks for the current state, B supplies the benchmark's resolved state, C gives the history, and D1 puts what A produced into B's frame. A judge (Gemini-3.1-Flash-Lite) blind to condition scores answers against the benchmark's ground truth, one response per call, with protocol and rubrics fixed before any response was read.

Handed the resolved state (Table 4), DeepSeek-V4-Pro with reasoning answers these open-ended queries at 0.997. Given a history carrying the same information, it answers at 0.734. The gap is present in every configuration, every interval excludes zero, and it is largest, 0.477, on the unmodified histories, where the model reads exactly what the benchmark's authors wrote. We therefore do not attribute it to our task family, our queries, our answer format, or a context we compacted.

Reasoning narrows the gap without closing it. Disabled, 0.365; enabled, 0.264. A median of 1,123 reasoning tokens per item recovers about ten points and leaves the rest. Opus 5.5, which answers every controlled chain correctly, still loses 0.101. The gap persists where the model resolves the state correctly when asked separately: +0.175 and +0.283 for DeepSeek-V4-Pro with reasoning on and off and +0.080 for Opus 5.5, all intervals excluding zero.

The factorized pipeline lowers end-to-end accuracy in the other four configurations ($\text{D1} - \text{C} < 0$) and raises it slightly for Opus 5.5; Section 8 relates the sign to compilation fidelity.

A judge from another vendor family (Claude Haiku 4.5) reproduces the gap, if anything larger. The primary and replication judges agree with human labels on 0.836 and 0.812 of answering responses, respectively, and assign fewer positives than the humans (Appendix M).

8 What factorization actually buys

Factorization replaces computation amid coexisting descriptions with externalizing a state the second call can use. Where that state is faithful the exchange is favorable; otherwise, the second call receives a wrong state with no history to fall back on.

In the controlled duplication experiment the compilation is exactly correct on every item: the state is short, in view, and need only be written down. On STALE, DeepSeek-V4-Pro's single-pass compilation is correct on a small minority. Splitting the pipeline result by compilation correctness gives, with reasoning enabled, $(\text{D1} - \text{C}) = +0.087$ on correct compilations and -0.173 on the rest, an interaction of $+0.261$ $[+.155, +.366]$; with reasoning disabled, $+0.101$ against -0.123, interaction $+0.224$. These descriptive effects reverse sign across strata, so the overall sign follows the share of faithful compilations. Under the primary judge, Opus 5.5 compiles correctly on 0.787 of scenarios, against 0.258 for DeepSeek-V4-Pro; it gains $+0.054$ $[+.022, +.086]$ where it compiles correctly, and its net effect is positive. Because compilation success is an outcome of the factorized route on the same item, this stratification identifies a fidelity bottleneck, not a causal rule for choosing a route.

Compilation demands more than noticing that something changed: the benchmark's own state-resolution probe is passed far more often than our requirement to state the current state and its supersession, and rarely the other way round (Appendix M). Short, visible states are cheap to compile and nearly close the gap; reconstructing long histories can instead bottleneck the pipeline and make

Table 5: Deleting the superseded evidence beats deleting as much other text, for every reader. Nineteen novels, $n = 241$ points over 94 questions; CI removes text matched on position and length, differing in size from CS by 1.0% (median) and 5.9% (90th percentile). The pre-specified reader-equal-weight mean is $+0.044$, interval $[+.015, +.074]$ over questions and $[+.010, +.078]$ over books.

	C0	CS	CI	CS − CI
Qwen3.5-9B	0.635	0.689	0.622	+0.066
GPT-5.6-Luna	0.734	0.759	0.730	+0.029
Gemini-3.7-Flash	0.730	0.776	0.739	+0.037
mean over readers				+0.044

it worse than the history route without a fidelity check. In the separate STALE routing experiment, a verifier-gated route gains $+0.010$ $[-.010, +.030]$ over the history route; routing by ground-truth compilation fidelity gains $+0.022$, since faithful compilations occur on easier histories (Appendix N).

9 Deleting the obsolete description recovers accuracy in novels

Patching showed the obsolete state driving the answer from inside the model; the same claim can be tested in the text. If a superseded description is what interferes, deleting the sentences that carry it should recover accuracy, and deleting as much other text should not. OAKS (Kim et al., 2026) allows that test on prose not written for this purpose: nineteen public-domain novels, with the answer that holds after each chunk recorded and the sentences supporting each answer annotated, so the evidence for a superseded state is given rather than located by us.

At a point in a stream we build four contexts from the text so far. C0 leaves it alone, CS deletes the annotated sentences supporting an answer that was correct earlier but not now, CI deletes as much other text from the same chunks, and CN deletes the sentences supporting the current answer. Three readers answer the benchmark's own question in its own format. Population, deletion rule, readers and statistics were fixed before any reader answered; construction and audits are in Appendix O.

Eligibility requires non-overlapping superseded and current evidence. Audits against the benchmark files confirm on all 241 points that CS removes only superseded evidence and CI removes no evidence.

Deleting the evidence for a state the narrative has moved past raises accuracy by 4.4 points beyond deleting as much other text, and the interval excludes zero whether questions or books are resampled. All three readers move the same way across a range of 0.635 to 0.734 on the underlying task, and the effect is positive in ten of the nineteen books and negative in three.

Two conditions on Qwen3.5-9B say what the recovery is not: matched deletion changes accuracy by -0.012 $[-.049, +.021]$, no detectable effect of shortening alone, and deleting the current answer's evidence costs 0.228 $[.149, .312]$, as it must if the annotation locates an answer's evidence.

The effect is smaller here than on the benchmark's synthetic half, where the same readers give $+0.311$, $+0.146$ and $+0.039$, and the attenuation is uneven: two readers keep about a fifth of it and the third keeps all of it. A superseded fact in a synthetic stream is one sentence, and deleting it removes the state from the text; in a novel the state is often still legible in the surrounding narrative, so the deletion is partial by construction.

10 Conclusion

Two descriptions of a task state interfere even when they agree, and when one is obsolete it stays causally active and drives the answer: neutralizing it at a single token restores most of the lost accuracy, and deleting the sentences that carry it recovers accuracy in novels. Compiling before computing nearly eliminates the composition penalty in the controlled setting. On long histories its net effect follows the fidelity of the reconstructed state. Explicit reasoning (Wei et al., 2022; Nye et al., 2022) closes the gap on short controlled chains, including for Claude Opus 5.5, but no reader we test closes it on long histories. Which description is authoritative remains a separate problem, since compilation resolves aggregation rather than adjudication (Appendix P).

AI USE STATEMENT

Generative AI tools appear in this work as the models under study and in two further roles. Two hosted models, `gemini-3.1-flash-lite` and `claude-haiku-4-5-20251001`, serve as automated judges for the open-ended benchmark (Appendix M), calibrated against a condition-blind audit by the authors. Generative AI assistants were used for limited assistance with coding, language editing, and internal consistency checks. The research questions, the analysis rules fixed in advance of each run, and the interpretation of every result are the authors', who verified each reported number, claim and artifact and take responsibility for the final content of this work.

ETHICS STATEMENT

The controlled testbed is generated from random draws over abstract sensor labels and carries no personal or sensitive content. The external evidence comes from three public benchmarks, one of which supplies nineteen public-domain novels, and all are used under the terms on which they are distributed. The human labels that calibrate our judges were applied by the authors to model-generated text; no data were collected from human participants and no personal information was processed. The models we evaluate are publicly released checkpoints and commercial interfaces, used as experimental subjects under their published terms.

What we report is a reliability failure. A system holding an obsolete description alongside the current one computes from the wrong one while reporting the correction correctly, so the failure passes the check a builder is most likely to run. The intervention we study is not a general remedy, and the conditions under which it costs accuracy are reported with it.

REPRODUCIBILITY STATEMENT

The generator that draws the task states and the two capability filters are given in Appendix A, the update message for every condition in Appendix A.1, the second task family in Appendix A.2, and the complete prompt templates in the code submitted as supplementary material. The patching construction, the one-token alignment it requires and the control site are in Appendix E, together with the final-position result we do not report as localization and the control that exposes it (Appendix F). Answers on the open-ended benchmark are judged; the rubric, the two judges, the human calibration and the per-condition rates are in Appendix M. The three external benchmarks are public. For the novel experiment, eligibility rules, the deletion rule and audits performed before model evaluation are in Appendix O. Every rate is reported with its item count, and every interval names its resampling scheme where it appears.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*, 2024.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020.

Hanxiang Chao, Yihan Bai, Rui Sheng, Tianle Li, and Yushi Sun. STALE: Can LLM agents know when their memories are no longer valid? *arXiv preprint arXiv:2605.06527*, 2026.

Hung-Ting Chen, Michael J.Q. Zhang, and Eunsol Choi. Rich knowledge sources bring complex knowledge conflicts: Recalibrating models to reflect conflicting evidence. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 2292–2307, 2022. doi: 10.18653/v1/2022.emnlp-main.146.

Zixuan Chen, Hao Lin, Zizhe Chen, Yizhou Tian, Garry Yang, Depeng Wang, Ya Guo, Huijia Zhu, and James Cheng. Knowing but not correcting: Routine task requests suppress factual correction in LLMs. *arXiv preprint arXiv:2605.05957*, 2026.

Youngbin Choi, Minjong Lee, Saemi Moon, Seunghyuk Cho, Chaehyeon Chung, MoonJeong Park, and Dongwoo Kim. In-place feedback: Reliable refinement for multi-turn expert-LLM collaboration. *arXiv preprint arXiv:2510.00777*, 2025.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James Glass, and Pengcheng He. DoLa: Decoding by contrasting layers improves factuality in large language models. In *International Conference on Learning Representations*, 2024.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, pp. 283–298, 2024. doi: 10.1162/tacl_a_00644.

Yiyang Feng, Zeming Chen, Haotian Wu, Jiawei Zhou, and Antoine Bosselut. Tracking the limits of knowledge propagation: How LLMs fail at multi-step reasoning with conflicting knowledge. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5813–5847, 2026. doi: 10.18653/v1/2026.eacl-long.273.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12216–12235, 2023. doi: 10.18653/v1/2023.emnlp-main.751.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *International Conference on Learning Representations*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. In *Advances in Neural Information Processing Systems 36*, pp. 17643–17668, 2023. doi: 10.52202/075280-0774.

Wenyue Hua, Jiang Guo, Mingwen Dong, Henghui Zhu, Patrick Ng, and Zhiguo Wang. Propagation and pitfalls: Reasoning-based assessment of knowledge editing through counterfactual tasks. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 12503–12525, 2024. doi: 10.18653/v1/2024.findings-acl.743.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations*, 2024.

Dongwei Jiang, Bowei Zhang, Andrew Wang, Nicholas Andrews, and Daniel Khashabi. Feedback friction: LLMs struggle to fully incorporate external feedback. In *Advances in Neural Information Processing Systems 38*, pp. 60984–61016, 2025. doi: 10.52202/085713-1831.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. When can LLMs actually correct their own mistakes? a critical survey of self-correction of LLMs. *Transactions of the Association for Computational Linguistics*, pp. 1417–1440, 2024. doi: 10.1162/tacl_a_00713.

Jungo Kasai, Keisuke Sakaguchi, Yoichi Takahashi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah Smith, Yejin Choi, and Kentaro Inui. RealTime QA: What's the answer right now? In *Advances in Neural Information Processing Systems 36*, pp. 49025–49043, 2023. doi: 10.52202/075280-2130.

Jiyeon Kim, Hyunji Lee, Dylan Zhou, Sue Hyun Park, Seunghyun Yoon, Trung Bui, Franck Dernoncourt, Sungmin Cha, and Minjoon Seo. Can large language models keep up? benchmarking online adaptation to continual knowledge streams. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 42240–42272, 2026. doi: 10.18653/v1/2026.acl-long.1956.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023. doi: 10.1145/3600006.3613165.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. In *International Conference on Learning Representations*, 2026.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*, 2025.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Prompt repetition improves non-reasoning LLMs. *arXiv preprint arXiv:2512.14982*, 2025.

Mosh Levy, Alon Jacoby, and Yoav Goldberg. Same task, more tokens: the impact of input length on the reasoning performance of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15339–15353, 2024. doi: 10.18653/v1/2024.acl-long.818.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, 2020.

Adam Liška, Tomáš Kočiský, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, et al. StreamingQA: A benchmark for adaptation to new knowledge over time in question answering models. In *International Conference on Machine Learning*, 2022.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, pp. 157–173, 2024a. doi: 10.1162/tacl_a_00638.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, et al. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024b.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. Entity-based knowledge conflicts in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7052–7063, 2021. doi: 10.18653/v1/2021.emnlp-main.565.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems 36*, pp. 46534–46594, 2023. doi: 10.52202/075280-2019.

Aleksandar Makelov, Georg Lange, Atticus Geiger, and Neel Nanda. Is this the subspace you are looking for? an interpretability illusion for subspace activation patching. In *International Conference on Learning Representations*, 2024.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, pp. 153–157, 1947. doi: 10.1007/BF02295996.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems 35*, pp. 17359–17372, 2022. doi: 10.52202/068431-1262.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models. In *Deep Learning for Code Workshop at the International Conference on Learning Representations*, 2022.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35*, pp. 27730–27744, 2022. doi: 10.52202/068431-2011.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–22, 2023. doi: 10.1145/3586183.3606763.

Ethan Perez, Sam Ringer, Kamile Lukosiute, Karina Nguyen, Edwin Chen, Scott Heiner, Craig Pettit, Catherine Olsson, Sandipan Kundu, Saurav Kadavath, Andy Jones, Anna Chen, Benjamin Mann, Brian Israel, Bryan Seethor, Cameron McKinnon, Christopher Olah, Da Yan, Daniela Amodei, Dario Amodei, Dawn Drain, Dustin Li, Eli Tran-Johnson, Guro Khundadze, Jackson Kernion, James Landis, Jamie Kerr, Jared Mueller, Jeeyoon Hyun, Joshua Landau, Kamal Ndousse, Landon Goldberg, Liane Lovitt, Martin Lucas, Michael Sellitto, Miranda Zhang, Neerav Kingsland, Nelson Elhage, Nicholas Joseph, Noemi Mercado, Nova DasSarma, Oliver Rausch, Robin Larson, Sam McCandlish, Scott Johnston, Shauna Kravec, Sheer El Showk, Tamera Lanham, Timothy Telleen-Lawton, Tom Brown, Tom Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Jack Clark, Samuel R. Bowman, Amanda Askell, Roger Grosse, Danny Hernandez, Deep Ganguli, Evan Hubinger, Nicholas Schiefer, and Jared Kaplan. Discovering language model behaviors with model-written evaluations. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13387–13434, 2023. doi: 10.18653/v1/2023.findings-acl.847.

Jacob Pfau, William Merrill, and Samuel R. Bowman. Let's think dot by dot: Hidden computation in transformer language models. In *Conference on Language Modeling*, 2024.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023. doi: 10.18653/v1/2023.findings-emnlp.378.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems 36*, pp. 53728–53741, 2023. doi: 10.52202/075280-2338.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, et al. Towards understanding sycophancy in language models. In *International Conference on Learning Representations*, 2024.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, 2023.

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 783–791, 2024. doi: 10.18653/v1/2024.naacl-short.69.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems*, 2020.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, and Thang Luong. Freshllms: Refreshing large language models with search engine augmentation. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 13697–13720, 2024. doi: 10.18653/v1/2024.findings-acl.813.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: A circuit for indirect object identification in GPT-2 small. In *International Conference on Learning Representations*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35*, pp. 24824–24837, 2022. doi: 10.52202/068431-1800.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. Generating sequences by learning to self-correct. In *International Conference on Learning Representations*, 2023.

Jason Weston and Sainbayar Sukhbaatar. System 2 attention (is something you might need too). *arXiv preprint arXiv:2311.11829*, 2023.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, pp. 209–212, 1927. doi: 10.1080/01621459.1927.10502953.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020. doi: 10.18653/v1/2020.emnlp-demos.6.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *International Conference on Learning Representations*, 2024.

Rongwu Xu, Zehan Qi, Zhijiang Guo, Cunxiang Wang, Hongru Wang, Yue Zhang, and Wei Xu. Knowledge conflicts for LLMs: A survey. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8541–8565, 2024. doi: 10.18653/v1/2024.emnlp-main.486.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, 2018. doi: 10.18653/v1/D18-1259.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Yizhen Yuan, Rui Kong, Dongze Li, Yuanchun Li, and Yunxin Liu. Benchmarking LLM's capability in reasoning over conflicting web references. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 303–322, 2026. doi: 10.18653/v1/2026.acl-long.11.

Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. In *International Conference on Learning Representations*, 2024.

Zexuan Zhong, Zhengxuan Wu, Christopher Manning, Christopher Potts, and Danqi Chen. MQuAKE: Assessing knowledge editing in language models via multi-hop questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15686–15702, 2023. doi: 10.18653/v1/2023.emnlp-main.971.

A TASKS, EXAMPLES, AND PROMPTS

Figures 5 and 6 show complete examples of the two task families.

A.1 CONSTRUCTION AND PROMPT TEMPLATES

The user turn is identical across conditions up to the message that delivers the update, so any difference between conditions is a difference in that message alone. The system turn reads: *You are a careful analyst. Answer with exactly one letter: A, B, C, or D. Do not explain.*

Table 6: The update message for every condition, on the item of Figure 5, whose answer is `B` before the update and `D` after it. The first three are conditions on which a correct answer must not move; `flag_only` withholds the corrected value, which the binary tier still implies.

Condition	Message appended as a further user turn
`irrelevant`	*irrelevant* `Note: the housing for cyan was repainted during last week's maintenance.`
`noop_correction`	*no effect on the answer* `Correction: delta's status was misreported. delta's status is actually OK, not FAIL.`
`assert_only_false`	*unsupported wrong answer* `That is not right. The answer is C.`
`flag_only`	*flagged, value withheld* `The tier you used for cyan is wrong.`
`state_implicit`	*corrected value stated indirectly* `For reference, the asset register lists cyan's tier as primary.`
`state_true`	*explicit correction* `Correction: cyan's tier was misrecorded. cyan's tier is actually primary, not backup.`
`state_true_recompute`	*explicit correction, told to recompute* `Correction: cyan's tier was misrecorded. cyan's tier is actually primary, not backup. Recount the CRITICAL sensors.`
`state_true_assert_true`	*correction plus the correct answer* `Correction: cyan's tier was misrecorded. cyan's tier is actually primary, not backup. So the answer is D.`
`state_true_assert_false`	*correction plus a wrong answer* `Correction: cyan's tier was misrecorded. cyan's tier is actually primary, not backup. So the answer is C.`

Generation. The generator draws four sensors with independent binary status and tier, computes the critical count under the rule, and shuffles the count-to-letter map. It then enumerates every single-attribute flip, keeps those that change the count as candidate updates, and separately keeps one flip that leaves the count unchanged as the no-effect update.

A third letter, correct under neither state, is reserved for the conditions in which a wrong answer is asserted. Items whose count exceeds three under either state are discarded so that all four options remain reachable.

A sensor item: the corrected state

Sensor array report:
alpha: status = FAIL, tier = backup
bravo: status = OK, tier = backup
cyan: status = FAIL, tier = primary
delta: status = FAIL, tier = backup

A sensor is CRITICAL if its status is FAIL and its tier is primary.

How many sensors are CRITICAL?
A) 3 B) 0 C) 2 D) 1

Answer with a single letter.

Gold: D (one critical sensor).

Before the correction, cyan's tier is backup and the answer is B. The correction changes cyan's tier to primary, giving the state shown above.

Figure 5: The sensor example used in Section 2 and Table 6, showing the corrected state. Only cyan satisfies both attributes. Reverting its tier to backup gives the original state with no critical sensors; the correction therefore moves the answer from B to D.

One item of the second family

Shipping manifest:
Kestrel: clearance = clear, route = express
Marlin: clearance = clear, route = express
Onyx: clearance = pending, route = express
Pike: clearance = clear, route = express

A parcel is HELD if its clearance is pending and its route is express.
Report the HELD parcel that appears first in the manifest.

Which parcel is reported?
A) Onyx
B) Marlin
C) Pike
D) Kestrel

Answer with a single letter.

The answer is A under the manifest as stated, and D after the update.

Correction: Kestrel's clearance was misrecorded. Kestrel's clearance is actually pending, not clear.

Figure 6: One item of the second family in full. The predicate is the same conjunction over two binary attributes; the question asks which held parcel appears first, so the answer names an entity and the correction moves it from `A` to `D`.

Verification. All 900 items were checked programmatically before any model was run: the count matches the stated state before and after the update, the count-to-letter map is a bijection, the pre-update, post-update and distractor letters are distinct, exactly one attribute differs between the two states, and the no-effect update leaves the count unchanged. The generated set is balanced across correction types (464 status flips, 436 tier flips) and across answer letters.

Prompt. The user turn lists the four sensors, states the rule, poses the question, lists the four options and instructs a single-letter reply. A correction is delivered as a further user turn, phrased as a record correction rather than as feedback on the answer. The full templates for all nine conditions are reproduced in the supplementary material.

Reading the answer. Each option letter is a single token for every tokenizer used here, so the answer distribution is read from the logits at the final prompt position and renormalized over the four letters. We also record how much probability mass falls on the four letters at all; it is above 0.98 on average for every checkpoint except two rungs of the Tülu-3 sequence, where 7% of rows fall below 0.5 and the corresponding rates should be read as the least precise in the paper.

A.2 SECOND TASK FAMILY

The second family keeps the conjunctive predicate over two binary attributes and changes what is aggregated. Instead of counting the items that satisfy the predicate, the question asks which of them appears first, so the answer names an entity and an update moves the answer from one entity to another rather than from one count to another. Items are generated, verified and filtered exactly as in the first family: the reported parcel matches the stated manifest before and after the update, the option order is a shuffled bijection onto the four parcels, the pre-update, post-update and distractor letters are distinct, and exactly one attribute differs between the two states. Figure 6 shows one item in full.

Table 7: The same measurements when the aggregation changes. In the second family a parcel is held under a conjunction of two attributes and the question asks which held parcel appears first, so the answer names an entity rather than a count and an update moves the answer to a different entity. The answer-naming channel stays at the ceiling while state-based revision falls in both models, and selectivity is negative in both.

Family	Checkpoint	n	TOLD	STATE	SPURIOUS	Selectivity
count which	Qwen3-8B	599	1.000	0.277	0.192	$+0.085$
count which	Llama-3.1-8B-Instruct	353	1.000	0.635	0.739	-0.105
name which	Qwen3-8B	737	0.963	0.168	0.198	-0.030
name which	Llama-3.1-8B-Instruct	660	1.000	0.344	0.415	-0.071

Eligibility is higher than in the first family, at 81.9% for Qwen3-8B and 73.3% for Llama-3.1-8B-Instruct. The lower revision rates in Table 7 are measured on larger admitted sets.

B FULL BEHAVIORAL RESULTS

Table 8: Eligibility and the two headline rates with Wilson intervals. Eligibility is the share of the 900 generated items on which the checkpoint both answers correctly from the stated premise and answers correctly when the corrected premise is given from the start.

Checkpoint	eligible	% of 900	explicit correction	spurious update
Qwen3-4B	571	63.4	0.264 [0.230, 0.302]	0.201 [0.171, 0.236]
Qwen3-8B	599	66.6	0.277 [0.243, 0.314]	0.192 [0.162, 0.225]
Llama-3.1-8B-Instruct	353	39.2	0.635 [0.583, 0.683]	0.739 [0.691, 0.782]
Olmo-3-7B-Instruct-SFT	153	17.0	0.556 [0.476, 0.632]	0.627 [0.549, 0.700]
Olmo-3-7B-Instruct-DPO	272	30.2	0.699 [0.642, 0.750]	0.717 [0.661, 0.767]
Olmo-3-7B-Instruct	311	34.6	0.714 [0.661, 0.761]	0.688 [0.635, 0.737]
Llama-3.1-Tulu-3-8B-SFT	65	7.2	0.662 [0.540, 0.765]	0.677 [0.556, 0.778]
Llama-3.1-Tulu-3-8B-DPO	98	10.9	0.582 [0.483, 0.674]	0.643 [0.544, 0.731]
Llama-3.1-Tulu-3-8B	81	9.0	0.568 [0.459, 0.670]	0.556 [0.447, 0.659]
Qwen3.5-9B	783	87.0	0.737 [0.705, 0.767]	0.474 [0.439, 0.509]

Table 8 reports eligibility and the two headline rates with Wilson intervals (Wilson, 1927). Table 9 gives the three-way outcome split for every checkpoint on the three conditions that differ only in what the message asserts, and Table 10 gives all nine conditions for the two checkpoints used for the detailed controls.

Two patterns in Table 9 are worth isolating. First, the COPY column under *correction plus a wrong answer* is above 0.96 for every checkpoint: when a message contains both a true state correction and a

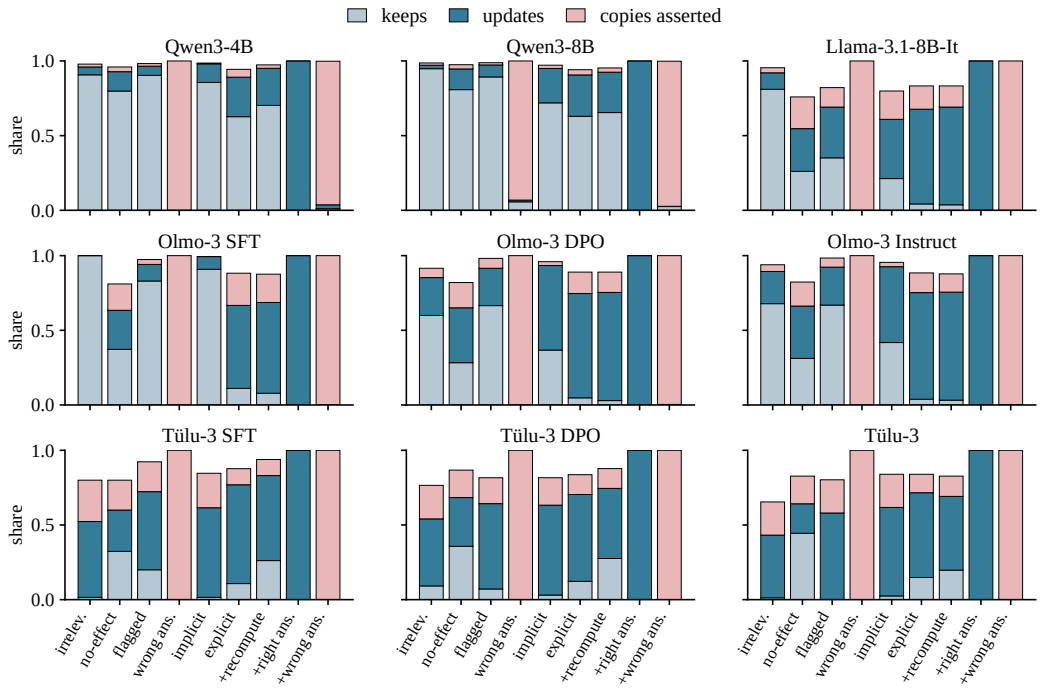

Figure 7: Where the answer lands across nine checkpoints. Bars are stacked: the share that keeps the pre-update answer, the share that gives the post-update answer, and the share that gives the reserved distractor letter. Unfilled mass is the fourth option. The irrelevant, no-effect and wrong-answer conditions are those on which a correct answer must not move; the flagged condition (`flag_only`) withholds a value that the binary tier still implies. The two Qwen panels and the four Llama and Olmo panels show the two failure modes side by side, and the rightmost condition in every panel is almost entirely peach, meaning an asserted wrong answer is taken even when the true correction sits in the same sentence.

false conclusion, the conclusion wins essentially always, even though the information needed to reject it is in the same sentence. Second, the STAY column under *explicit correction* separates the two failure modes cleanly, ranging from 0.63 for the Qwen checkpoints down to 0.04 for Llama-3.1-8B-Instruct.

Table 9: Every checkpoint under three answer-delivery conditions. STAY keeps the pre-update answer, REV gives the post-update answer, COPY gives the reserved distractor letter, correct under neither state, which is the asserted letter in the correction-plus-wrong-answer condition. The three do not exhaust the options, so the remaining mass falls on the fourth letter. The Tülu-3 sequence has lower eligibility, making its rates the least precise.

	explicit correction			correction + correct answer			correction + wrong answer		
Checkpoint	stay	rev	copy	stay	rev	copy	stay	rev	copy
Qwen3-4B	0.627	0.264	0.053	0.000	1.000	0.000	0.011	0.026	0.961
Qwen3-8B	0.629	0.277	0.035	0.000	1.000	0.000	0.025	0.002	0.972
Llama-3.1-8B-Instruct	0.042	0.635	0.156	0.000	1.000	0.000	0.000	0.000	1.000
Olmo-3-7B-Instruct-SFT	0.111	0.556	0.216	0.000	1.000	0.000	0.000	0.000	1.000
Olmo-3-7B-Instruct-DPO	0.048	0.699	0.143	0.000	1.000	0.000	0.000	0.004	0.996
Olmo-3-7B-Instruct	0.039	0.714	0.132	0.000	1.000	0.000	0.000	0.003	0.997
Llama-3.1-Tulu-3-8B-SFT	0.108	0.662	0.108	0.000	1.000	0.000	0.000	0.000	1.000
Llama-3.1-Tulu-3-8B-DPO	0.122	0.582	0.133	0.000	1.000	0.000	0.000	0.000	1.000
Llama-3.1-Tulu-3-8B	0.148	0.568	0.123	0.000	1.000	0.000	0.000	0.000	1.000
Qwen3.5-9B	0.041	0.737	0.114	0.000	1.000	0.000	0.000	0.000	1.000

Table 10: Nine update conditions. Stay/revise denote original/updated answer letters; other options are omitted. Above the rule, correct answers must not change, except under the value-withheld flag, which the binary tier still resolves.

	Qwen3-8B (n=599)		Llama-3.1-8B-It (n=353)	
Condition	stay	revise	stay	revise
irrelevant message	0.948	0.023	0.810	0.110
update with no effect on the answer	0.808	0.139	0.261	0.286
unsupported wrong answer asserted	0.057	0.012	0.000	0.000
told the reading is wrong, value withheld	0.893	0.080	0.351	0.340
corrected value stated indirectly	0.720	0.230	0.212	0.397
explicit correction	0.629	0.277	0.042	0.635
explicit correction, told to recompute	0.654	0.270	0.037	0.654
correction plus the correct answer	0.000	1.000	0.000	1.000
correction plus a wrong answer	0.025	0.002	0.000	0.000

C DUPLICATION AND SUPERSESSION

Table 11 defines the seventeen conditions of Figure 2a.

Table 11: Conditions of Figure 2a. In the first group the correction arrives in a later user turn, after the assistant turn named; in the second, everything is one user turn with no prior answer; in the last, two timestamped reports come with a rule that the later one is valid.

Condition	Context before the question is answered
Prior commitment	
None	question and correction in one user turn, no assistant turn
Acknowledgment	question; assistant turn "Understood."; correction
Model answer	question; the model's own answer; correction
Imposed answer	question; an answer letter the model never derived; correction
Update delivery	
After question	report, rule and question, then the correction in prose
After report	report, then the correction in prose, then rule and question
Report format	the correction written as a report line, appended to the report and marked as corrected
Corrected appended	the full corrected report listed after the original report
Obsolete removed	the corrected report only
Duplication	
One copy	the corrected report once
Two copies	the corrected report twice
+ longer irrelevant	the corrected report and a slightly longer maintenance log with the same sensor names and line format
Compiled to one	two copies; a first call lists the current status, and its output replaces them
Restated, still two	two copies; a first call restates both verbatim, and its output replaces them
Supersession	
Valid report last	obsolete report at 09:00, then corrected report at 14:00
Valid report first	corrected report at 14:00, then obsolete report at 09:00
Obsolete is valid	corrected-looking report at 09:00, then the obsolete-looking one at 14:00, so the correct answer is the pre-update letter

Table 12 reports the conditions that separate duplication from conflict. Duplicating a report costs accuracy whether the duplicated report is the corrected one or the obsolete one, and the two costs are close, which makes the duplication effect interpretable as a cost of multiplicity rather than of error.

The duplication block of Figure 2a carries the remedy. Its conditions are run separately from those of Table 12, so its single-report and duplicated-report rates differ from that table's by a few thousandths. Compiling the two identical reports into one recovers accuracy from 0.886 to 0.990 on Qwen3-8B

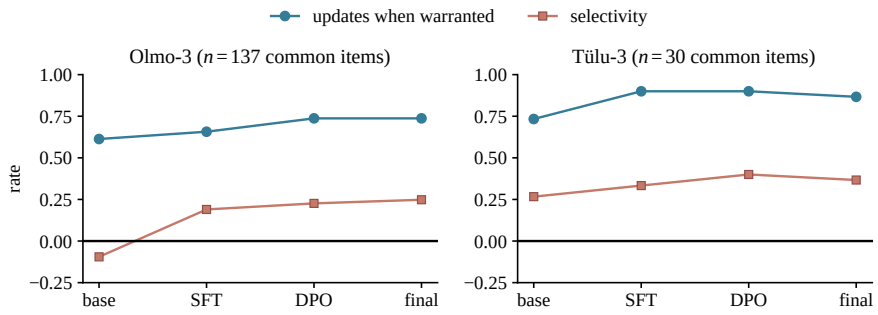

Figure 8: Post-training on common support. Post-training improves both the rate of warranted updates and selectivity. For Olmo-3 selectivity turns positive at the supervised stage and rises through preference tuning, and no rung reaches the level at which an asserted answer is accepted, which is 1.00 throughout. The Tülu-3 common support is small because its base checkpoint is eligible on few items, and we read it only as directionally consistent.

and from 0.773 to 0.992 on Llama-3.1-8B-Instruct, against 0.992 and 1.000 when the report appears once. A control that spends the same second call and restates both reports instead of compiling them leaves accuracy at 0.880 and 0.779. Nothing is contradicted in any of these conditions, so the recovery is not adjudication.

The last row is the one that limits what can be claimed about supersession. When the corrected-looking report is timestamped as the obsolete one, so that the correct answer is the pre-update letter, Qwen3-8B chooses the authoritative report on 0.530 of the items where it selects one of the two, which an exact binomial test does not separate from chance ($p = 0.17$, $n = 572$). Llama-3.1-8B-Instruct reaches 0.645 ($p < 10^{-4}$, $n = 299$). Authority tracking is therefore unreliable and model-dependent in this testbed, and we do not claim it is absent.

Table 12: Duplication and supersession. The first block varies how many state descriptions are present, the second states explicitly that only the most recent report is valid. In the last row the corrected-looking report is the obsolete one, so the correct answer is the pre-update letter. Every cell uses the same 599 items for Qwen3-8B and 353 for Llama-3.1-8B-Instruct.

Condition	Qwen3-8B	Llama-3.1-8B-It
corrected report only	0.997	0.992
corrected report twice	0.883	0.768
obsolete report twice	0.878	0.742
obsolete then corrected	0.558	0.708
timestamped, valid report last	0.616	0.652
timestamped, valid report first	0.518	0.674
timestamped, obsolete report is the valid one	0.506	0.547

D POST-TRAINING LADDER

Base checkpoints have no chat template, so running the sequence through chat templates would confound the post-training stage with the prompt format. Every rung is instead scored on the same plain-text few-shot prompt. The two exemplars are balanced by design: in one the update changes the answer and in the other it does not, so neither behavior is demonstrated preferentially.

Each rung admits a different subset; their shared eligible set contains 137 Olmo-3 items and 30 Tülu-3 items. On the Olmo-3 common support, revision rises 0.613, 0.657, 0.737, 0.737 across the base model, supervised fine-tuning (SFT), direct preference optimization (DPO) and the final model, and selectivity rises -0.095, $+0.190$, $+0.226$, $+0.248$. A within-item McNemar test (McNemar, 1947) gives $p = 0.012$ from base to DPO and $p = 0.007$ from SFT to DPO. The Tülu-3 common

support is too small to support a comparison and we report it only as directionally consistent. The TOLD rate is 1.00 at every rung including both base models.

E PATCHING DETAILS

Alignment. For the coexistence experiment the two prompts are built to differ by exactly one token. We verify this rather than assume it: both sequences are tokenized, their lengths are compared, and the set of differing positions is required to be a single contiguous run.

Of 300 candidate items, 260 satisfy this and the divergent run has length one in every case; the remainder are discarded, which happens when the two attribute values tokenize to different lengths. The experiment reads the first 150 of the aligned items in generator order.

Single-state selection. Panels (a) and (b) of Figure 3 select items whose explicit-correction answer equals the pre-update letter, not all incorrect answers. Llama's enlarged screening pool admits 3,492 of 9,000 items (38.8%, against 39.2% in the original pool) and yields 152 STAY candidates. Re-evaluation admits 142 of these, of which 131 retain the pre-update answer; all 131 have a one-token premise alignment and enter both patching directions. Qwen3-8B uses 150 aligned STAY items from the original pool. Selection uses the behavioral screening output; unpatched answers are measured again in the patching run.

Positions. Prompts are left-padded, so an unpadded index is shifted by the padding width of its row before patching. Interventions are applied at the input to a layer through a forward pre-hook and removed immediately afterwards.

Controls. The control copies donor activations from the same layer and number of positions, 40 tokens earlier in the report. These positions precede the first input difference, so both runs see the same prefix. Under causal attention their activations are identical in exact arithmetic. The control therefore checks that copying an unchanged activation leaves the output unchanged; small numerical differences can arise from batching and finite precision. It does not match the activation change at the target and cannot establish that an equally strong intervention elsewhere would have no effect. We interpret the target intervention as evidence that the obsolete-state representation causally affects the answer, not as proof of unique site specificity. Table 13 gives the full layerwise values for all three directions.

Positional alignment. In the coexistence experiment, donor and recipient prompts have identical lengths, and patching is performed at the same token position and layer.

F FINAL-POSITION PATCHING AND ANSWER-IDENTITY CONTROLS

Patching the residual stream at the final answer position produces a sharp transition at layer 25, from 0.00 below to 0.89 above. This transition alone does not distinguish state transfer from answer-identity transfer.

Table 14 distinguishes the two. A donor drawn from a *different* item that happens to share the same answer letter reproduces the effect as well as the item's own computed state, and slightly exceeds it above layer 25; a donor with a different answer letter produces nothing at any layer. What transfers is the identity of the answer, not the state that entails it. Patching a dimension-permuted donor moves the output at a rate near chance among the remaining letters, which shows the output is disruptable but carries no information about where the state lives. The identity patch is flat at 0.02 everywhere, confirming the machinery is inert when it should be.

This is the failure mode that Makelov et al. (2024) describe, and it is the reason the experiment in Section 4 is constructed so that the donor is the same item at the same position and the manipulation removes rather than adds.

Table 13: Layerwise patching, all three directions, with the shared-prefix identity control alongside each. Values are the share of items answering with the updated state. The two models place the transition at different depths but show the same shape and the same flat control.

		supply current		inject obsolete		both present	
Checkpoint	Layer	target	ctrl	target	ctrl	target	ctrl
Qwen3-8B	0	0.967	0.020	0.000	1.000	0.940	0.587
	4	0.940	0.013	0.033	0.993	0.913	0.567
	8	0.853	0.013	0.120	1.000	0.920	0.580
	12	0.793	0.027	0.280	0.973	0.860	0.587
	16	0.340	0.020	0.673	0.987	0.733	0.573
	20	0.027	0.020	0.967	0.993	0.607	0.600
	24	0.027	0.027	1.000	0.993	0.587	0.587
	28	0.027	0.027	0.993	1.000	0.587	0.587
	32	0.027	0.020	0.993	1.000	0.587	0.587
Llama-3.1-8B-Instruct	0	0.565	0.023	0.015	1.000	0.873	0.753
	3	0.527	0.008	0.069	0.992	0.900	0.753
	6	0.489	0.015	0.252	0.977	0.873	0.740
	9	0.260	0.015	0.527	0.985	0.860	0.747
	12	0.115	0.023	0.870	0.985	0.793	0.740
	15	0.008	0.015	0.977	0.985	0.747	0.747
	18	0.031	0.015	0.985	0.985	0.753	0.747
	21	0.023	0.023	1.000	0.992	0.747	0.747
	24	0.031	0.031	0.985	0.977	0.747	0.747
	27	0.023	0.008	0.985	0.992	0.747	0.747
	30	0.023	0.023	1.000	0.992	0.753	0.753

Table 14: Final-position patching, and the control that disqualifies it. A donor taken from a *different* item that happens to share the answer letter reproduces the effect as well as the item's own computed state, so what transfers is the identity of the answer rather than the state. The identity patch confirms the machinery is inert when it should be. Values are for Qwen3-8B on 150 items.

Donor	L20	L24	L25	L27	L30	L35
same item, computed	0.000	0.073	0.887	0.920	0.980	0.993
same item, answer supplied	0.000	0.367	0.853	0.980	1.000	1.000
different item, same answer letter	0.000	0.013	0.920	0.967	1.000	0.993
different item, different letter	0.000	0.020	0.033	0.013	0.000	0.000
identity patch	0.020	0.020	0.020	0.020	0.020	0.020
dimension-permuted donor	0.287	0.220	0.267	0.127	0.247	0.253

G PREFILL DECOMPOSITION

Table 15 isolates the ingredients of an assistant turn placed before the answer. Each variant is supplied as fixed text rather than generated, so nothing depends on the model obeying an instruction such as "restate but do not solve".

The filler row is the reason the other rows cannot be read semantically. On Qwen3-8B a length-matched prefill carrying no task content raises accuracy from 0.339 to 0.472, so most of what restating the correction appears to buy is not about the correction.

Only the full derivation clears the filler by a substantial margin, and even it stops well below the condition in which the obsolete report is removed. On Llama-3.1-8B-Instruct the filler is harmful rather than helpful, so the non-semantic component is not even consistent in sign across families.

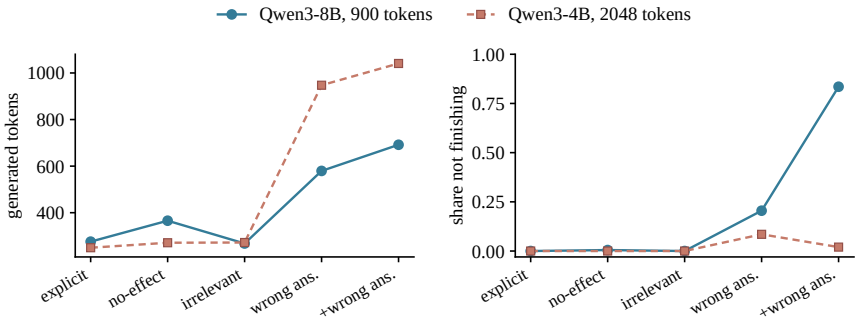

Figure 9: What explicit reasoning costs, by condition. Generation length rises sharply when the message contains both a true correction and a false asserted answer. That condition truncates $167/200$ responses for Qwen3-8B at a 900-token budget and $4/200$ for Qwen3-4B at 2048 tokens. Both the checkpoint and the budget differ; rates must be read with the configuration that produced them.

Table 15: Teacher-forced assistant turns. The filler row carries no task content and is matched in token count to the derivation row, so the difference between it and the baseline is the part of any gain that is not semantic. Every cell uses the same 599 items for Qwen3-8B and 353 for Llama-3.1-8B-Instruct.

Assistant turn contains	Qwen3-8B	Llama-3.1-8B-It
nothing (baseline)	0.339	0.703
length-matched filler	0.472	0.629
the correction restated	0.482	0.646
the corrected report restated	0.541	0.660
the full per-sensor derivation	0.653	0.762
obsolete state removed instead	0.993	0.992

H COMPILE THEN COMPUTE

Against the stricter reference that treats the two component successes as independent, so that the joint condition is compared to their product rather than to the weaker of them, the gap is $+0.397$ $[+.327, +.467]$ for Qwen3-8B and $+0.117$ $[+.042, +.192]$ for Llama-3.1-8B-Instruct, both intervals excluding zero. The two references differ materially only for Llama, whose resolution accuracy is the one that is not near ceiling.

Chains. Each item applies three successive single-attribute updates to a freshly drawn state. Updates may touch the same variable twice, so a later update must supersede an earlier one, and items whose first and last states share a critical count are discarded so that the answer must change.

Compiler output. The compile call is instructed to emit one line per sensor in a fixed format and nothing else. The schema contains only raw state, so the compile call outputs no verdict or count. The output is parsed with a strict pattern; parse failures we report separately from compilation errors. On Qwen3-8B all 300 outputs parse and all 300 match the ground-truth state exactly.

Two-pass controls. D2 replaces the input with the model's own verbatim restatement, which preserves the conflict. Its fidelity is measured rather than assumed: the restatement retains the original report on 0.970 of items and all three updates on 0.997. D4 replaces the input with a copy produced by code rather than by the model, holding the text fixed and removing generation from the comparison. Both remain near the single-call baseline.

Cost. Table 16 reports generated tokens. We report generated tokens only; a serving comparison would also have to account for prompt tokens, cache reuse and two round trips.

Table 16: Generated tokens on the update-chain task. The factorized route issues two calls. Their combined output is shorter than a direct answer for Qwen3-8B (46 versus 55 tokens), but longer for Llama (45 versus 6). Prompt tokens are not included and the second call re-encodes a shorter context than the first. Means are over the 300 chains of Table 2.

	Qwen3-8B	Llama-3.1-8B-It
direct answer	55	6
compile call	41	41
answer call	5	4
compile and compute, total	46	45

I CHAIN OF THOUGHT

Generations are greedy, with token budgets of 2048 for Qwen3-4B, 900 for Qwen3-8B and 1400 for Llama-3.1-8B-Instruct. Of 200 attempted items per model, correct, untruncated clean-state reasoning admits 200, 200 and 181, respectively. For explicit correction, the Qwen models have no exclusions; Llama has 3 truncated responses and 1 further unparseable response, leaving 177. Table 17 recomputes direct accuracy on these same retained item IDs: $62/200$, $58/200$ and $117/177$, respectively.

Truncation depends strongly on condition. For Qwen3-8B, pairing a true correction with a false asserted answer truncates $167/200$ responses; the remaining 33 average 692 tokens, versus 276 for explicit correction. Qwen3-4B's 2048-token run truncates $4/200$ in that condition and answers correctly on $189/196$=0.964 retained items. These are different checkpoints and budgets. Truncated or unparseable responses are excluded from the reported reasoning accuracy and mean token count. Conclusions about the heavily truncated condition under a small budget are drawn from a biased subset, so the rate must be read alongside the budget and exclusions.

Table 17: Explicit reasoning repairs single-update accuracy completely for the Qwen checkpoints and partially for Llama-3.1-8B-Instruct. Direct and reasoned accuracy use the same n items, restricted to correct, untruncated clean-state reasoning and parseable, untruncated correction responses. Tokens are mean generated tokens for those reasoned responses. The compile-and-compute cost is from the separate three-update experiment (Appendix H, $n = 300$), not a same-item cost comparison. Appendix I gives the reasoning budgets and exclusions.

		Answer updated		Generated
Checkpoint	n	directly	with reasoning	tokens
Qwen3-4B	200	0.310	1.000	250
Qwen3-8B	200	0.290	1.000	276
Llama-3.1-8B-Instruct	177	0.661	0.780	44
Three-update compile, then compute (Qwen3-8B), both calls				46

J REASONING AS A MODERATOR, ON CONTROLLED ITEMS

Section 7 reports that reasoning narrows the composition gap without closing it. On controlled tasks, Qwen3.5-9B can close the gap by reasoning. We built chains of $K \in \{1, 3, 5, 7\}$ updates over $D \in \{2, \ldots, 5\}$ decision attributes under an AND rule (16 cells), plus an OR rule at $D = 2$ for all four K values (4 cells). Each of these 20 cells has a matched control arm whose preceding updates are genuine corrections to attributes the rule never reads, token-matched to the state arm. Each cell evaluates between 246 and 311 eligible items.

Ordinary decoding keeps thinking disabled, as in Table 2, but allows 14,000 output tokens instead of 256, so the model can write out its reasoning. Under ordinary decoding the model answers a single-update item in about two tokens and a seven-update item in several hundred, and at $K \geq 3$ the joint condition sits at ceiling: it reasons its way past the difficulty. Constraining the answer to a

single scored token removes that compensation and the gap returns, between 0.21 and 0.60 across the grid, rising with K. Compiling first recovers most of it in every cell.

The penalty is not specific to updates the decision rule reads: the control arm, whose extra updates are decision-irrelevant, shows a penalty as large as the state arm's. What the control does separate is resolution: asked for the current state, the model is markedly worse on the state arm than on the token-matched control arm, by up to 0.31 at $K = 7$. Longer decision-relevant history impairs working out the state; it does not uniquely impair computing once several descriptions are present.

A hosted reasoning model. Claude Opus 5.5 (`claude-opus-5-5`), whose reasoning cannot be disabled, runs at its default effort (`medium`) on the 300 three-update chains of Section 5, with the same prompts, parser and scoring. A, B, C and D1–D4 are all 1.000, and none of the $2{,}700$ replies is truncated or refused. At this setting the controlled gap is not observed; on STALE the same model keeps a gap of $+0.101$ (Section 7).

K PRECISION, SCALE, ARCHITECTURE AND ENVIRONMENT

Four runs settle whether the penalty is a property of one checkpoint at one precision in one software stack. Their analysis rules were written down before the first of them was launched. Each uses the item pool, prompts and settings of the 8B run in Section 5, so the first three rows differ from their neighbors in one respect only (precision, then scale); Qwen3.5-9B is a separate checkpoint that replicates the pattern. We compare bfloat16 (BF16) with AWQ-quantized weights.

Table 18: The composition path across precision, scale and architecture, all on the same 300 items. A compiles the current state, B computes from a correct state, C does both at once, D1 compiles and then computes. Δ is Equation 2 and Δ' is the same quantity after factorizing. The first and last rows are the two checkpoints of Table 2.

Checkpoint	precision	A	B	C	D1	Δ	Δ'
Qwen3-8B	bf16	1.000	0.853	0.457	0.857	0.397	-0.003
Qwen3-8B	AWQ	0.997	0.843	0.660	0.840	0.183	$+0.003$
Qwen3-32B	AWQ	0.997	0.940	0.543	0.943	0.397	-0.003
Qwen3.5-9B	bf16	0.983	0.893	0.190	0.887	0.703	$+0.007$

Precision moves the magnitude and not the sign. Quantizing the same checkpoint takes Δ from 0.397 to 0.183. A paired item-level bootstrap over $20{,}000$ resamples puts that reduction at 0.213 with interval $[+.140, +.287]$. The penalty is smaller under AWQ and still there, so we report every comparison at a fixed precision and claim no invariance to it.

The larger checkpoint is the better calculator and the worse composer. Holding precision at AWQ, computing from a correct state rises from 0.843 to 0.940 while doing both at once falls from 0.660 to 0.543, and Δ widens by 0.213 $[+.120, +.307]$. The component that a larger model improves is exactly the one the joint condition already had.

The largest penalty is on the newest architecture. Qwen3.5-9B compiles the state on 0.983 of items and computes from a correct state on 0.893, both near the ceiling, yet answers the joint question on 0.190, giving $\Delta = 0.703$, larger than Qwen3-8B by 0.307 $[+.223, +.390]$. It is also the one checkpoint that updates selectively (Section 3); the two measurements apply one update and three respectively, so we set them beside each other rather than against each other.

Compiling reaches the compute-only level in every configuration. Δ' is within 0.007 of zero in all four rows, and no pipeline exceeds its weaker component by more than sampling noise.

Software environment. Re-running the smaller checkpoint under a second Wolf et al. (2020) release admits 599 against 590 items, with 582 in common and a Jaccard index of 0.959. On the common set no condition moves by more than 0.003 and item-level agreement runs from 0.972 to 1.000.

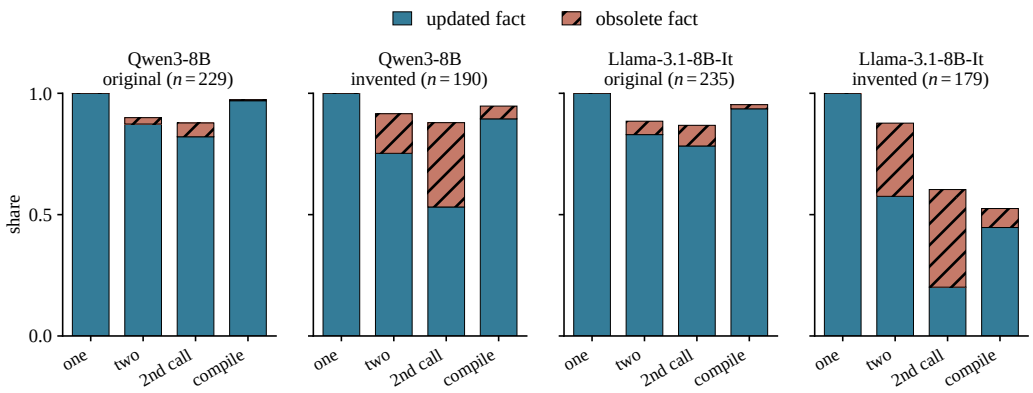

Figure 10: Real documents, showing where the answer lands rather than only whether it is right. Blue is the updated fact and amber the obsolete one. Retaining the obsolete sentence while spending a second call leaves the largest obsolete share in every setting, and the invented-entity versions carry more obsolete answers than the original ones, which is the opposite of what an account based on prior knowledge of the entity would predict.

L REAL DOCUMENTS

We repeat the contrast on HotpotQA (Yang et al., 2018) in the distractor setting, which supplies real multi-document context with sentence-level supporting-fact annotation. In a second version every mention of the answer entity is replaced by an invented name of the same type, so Wikipedia structure, multi-hop dependencies and distractors survive while the possibility of recovering the fact from weights does not. Figure 4c reports the four conditions on both versions.

Table 19: Coexistence lowers accuracy on real documents, with larger losses for invented entities. Compilation does not improve accuracy in every setting. Items are admitted only when the model answers correctly from the uncorrupted context. Compilation accuracy is the rate at which the model's rewritten sentence carries the updated fact.

			Answer accuracy				Compilation
Entities	Checkpoint	n	Single state	Both states	Second call, both states	Compile, compute	accuracy
Original	Qwen3-8B	229	1.000	0.873	0.821	0.969	0.969
	Llama-3.1-8B-It	235	1.000	0.830	0.783	0.936	0.945
Invented	Qwen3-8B	190	1.000	0.753	0.532	0.895	0.900
	Llama-3.1-8B-It	179	1.000	0.575	0.201	0.447	0.464

Invented entities show larger coexistence losses. Adding the second state costs 0.127 and 0.170 on the original entities and 0.247 and 0.425 on the invented ones, whose eligible items differ from those of the original entities. The loss is larger where the answer cannot be recovered from weights, so whatever produces it does not require the model to have known the fact beforehand, which separates the effect from conflicts defined against parametric knowledge (Feng et al., 2026; Longpre et al., 2021; Xie et al., 2024).

The remedy is bounded by compilation, and the bound is visible in advance. Across the four settings, accuracy after compiling tracks the accuracy of the compilation itself to within 0.017. Llama-3.1-8B-Instruct copies invented names correctly on fewer than half of items, and its pipeline inherits exactly that failure.

Compilation fidelity can be scored separately from the downstream answer, and in these controlled settings it tracks pipeline accuracy closely, though Appendix N shows that this alone does not yield a routing policy that pays.

Construction. We use the HotpotQA distractor validation split. An item is kept when the answer is a short span that occurs in exactly one sentence of exactly one paragraph, and that paragraph is annotated as supporting. The obsolete version replaces the answer span in that sentence with another entity, and the true sentence is supplied as an update alongside the ten paragraphs.

An item is discarded if the substituted entity already appears anywhere in the documents, since it would then have independent support.

Anonymization. In the second version the answer entity is classified by a deterministic surface test: a year matches `(1[5-9]\d\d|20\d\d)` in full after trimming whitespace; a person-like name has two or three words, each with an uppercase first character and a lowercase remainder. No NER model or LLM is used. Both the answer and its obsolete replacement are drawn as invented entities of that type. The substitution is applied to every paragraph, every title and the question, so no original mention survives, and the two invented names are required to share no token so that they do not read as related. Items whose answer does not fall into either type are discarded.

Filters. Only items answered correctly from the uncorrupted context are scored, so retrieval and multi-hop reasoning succeed on every scored item before the obsolete fact is added. Answers are scored by normalized span containment against the gold answer and, separately, against the obsolete substitute, so retaining the obsolete fact is measured directly rather than inferred from an error.

M JUDGING PROTOCOL AND CALIBRATION

Confidence intervals. Contrasts in Tables 3 and 4 use 95% percentile bootstrap intervals from 20,000 resamples (the 2.5th and 97.5th percentiles; random seed 0). For tool output, the unit is a filesystem item; all conditions are resampled together within the reported eligible population of 340 items (and separately within all 600 for the population check). For STALE, the unit is a scenario. Each contrast uses the intersection of scenarios judged in both conditions, resampling their paired score differences. An unjudged response is excluded, not scored incorrect. Thus the displayed marginal accuracies can have different denominators from a paired contrast.

Where the bar for a correct compilation is placed changes the level of every rate that depends on it: our judges place it differently enough that we report primary-judge rates and check their sensitivity to the replication judge, while the ordering of the conditions survives every placement we tried.

Answers on the independent benchmark are open-ended and the benchmark supplies no query-specific reference answers, so correctness is judged. The rubric follows the benchmark's own; the harness does not, because the official one shows a single call all three of its probe responses at once, and a judge that sees several responses together can rank them against each other. Each call therefore carries one query, the benchmark's ground truth and one answer, with no condition label, no model identity and no history.

The primary judge is `gemini-3.1-flash-lite` at temperature 0. The replication judge is `claude-haiku-4-5-20251001` at temperature 0, from a different vendor family, and it repeats every judgment rather than a sample. A condition-blind audit of 400 responses by the authors, drawn before any judging by a hash of the item and condition, calibrates both. Verdicts are never combined by majority: the primary judge defines the reported metric, the replication judge is shown beside it, and human labels calibrate rather than replace. A response that cannot be judged is excluded from its denominator rather than counted wrong.

On the DeepSeek and Qwen cells, agreement between the two judges is 0.993 on B, 0.887 on C and 0.850 on D1. Against the human labels the primary judge agrees on 0.836 of answering responses and the replication judge on 0.812. Both assign fewer positive labels than the humans: 0.642 and 0.584 against 0.764. Unscorable responses number 8 of 4800, and at most 2 of 400 in any condition, so the exclusion rule cannot carry the comparison.

Compaction keeps the two evidence sessions the benchmark names and adds the remaining sessions, in an order fixed by a hash of the scenario identifier, until the history would exceed $12{,}000$ tokens (`code/build_stale_manifest.py` in the supplementary material). The judge prompts are in `code/stale_judge.py` and `code/stale_judge_a.py`. For tool output, an item is eligible when, on a separate question about the same directory at another size threshold, the model answers correctly both from a supplied correct state and from the fresh listing alone; 340 of 600 items pass.

The Opus 5.5 cell. The replication judge shares a vendor with Opus 5.5, so for this cell it serves only as a check; the human audit covers the other cells. The two judges agree on 1.000 of B, 0.885 of C and 0.955 of D1 verdicts, but on only 0.515 of compilations, of which the replication judge passes 0.313 against the primary judge's 0.787. The conditional result does not depend on this: on the 125 scenarios whose compilation the replication judge passes, B $-$ C is $+0.088$ $[+.040, +.136]$ and D1 $-$ C is $+0.064$ $[+.016, +.120]$ under the primary judge. Two responses, one B and one C, are refusals without text and leave their denominators.

The benchmark's own state-resolution probe and the compilation rubric separate sharply. The probe is passed on 0.748 of items and the rubric on 0.258, they agree on 0.410, and the disagreement runs one way: the probe is passed while the rubric is not on 0.540 of items, the reverse on 0.050. Answering the probe correctly does not carry over to writing the resolved state in the form the second call consumes. The conditional quantities of Section 8 were fixed before the compilation was scored; their difference of differences was not.

Within a fixed human label, judge agreement across answer-length quartiles varies between 0.770 and 0.859 with no monotone trend.

N CAN THE BOUNDARY CONDITION BE USED AS A POLICY?

Section 8 shows that factorizing pays when the compiled state is faithful and costs when it is not. If that condition could be detected before the answer exists, a system could compile only when compilation can be trusted. We tested this under a specification written and hashed before the estimator existed, with a fixed success criterion. The gate did not meet it.

The design. A verifier is worth nothing if it re-solves the compiler's problem. Ours is given less to do. The compiler emits each claim with a citation to the passage it used. A mechanical index, string matching rather than a model, collects the passages that share the claim's rarest content words. The verifier then sees one claim, its cited passage and that short list, and answers whether the citation supports the claim and whether a later passage contradicts it. It never reads the history and never writes a state. The routing policy is fixed rather than learned: accept every claim and the answer comes from the compiled state, otherwise from the history.

Where it was built and where it was run. Every prompt, retrieval parameter and threshold was chosen on a corpus we generate, never on STALE. Development lifted detection of a faithful compilation to a balanced accuracy of 0.693 there. The artifact was then hashed and run once with Qwen3.5-9B on all 400 STALE scenarios. We describe this as a one-shot transfer rather than a held-out test, because the failure reported in Section 7 is what motivated building a verifier at all; what the procedure guarantees is that no STALE item, response or label entered the design.

Result. The gate's estimated gain is small and uncertain. It reaches 0.562 against 0.552 for reading the history directly, a difference of $+0.010$ with interval $[-.010, +.030]$, and it routes 0.147 of items to the compiled state. Both pre-specified conditions, an interval excluding zero and a routing rate of at least 0.20, fail. It clears the unconditional pipeline comfortably, $+0.102$ $[+.050, +.157]$, which is the same statement as D1 being worse than C read from the other side.

Why it fails. Scoring the compiled states against the benchmark's ground truth shows that 0.172 of them are faithful, and that on exactly those items the history route already scores 0.710 against 0.520 elsewhere: faithful compilations are concentrated on items the history route already answers well. A gate that routed perfectly, using the ground truth it is not allowed to see, would therefore reach 0.575 against 0.552, a gain of $+0.022$. This is the gain of oracle routing on fidelity alone, not an upper bound for policies that use more information; our gate already collects about half of it.

The condition is detectable: the verifier separates faithful from unfaithful compilations with an area under the curve of 0.744. Detecting it does not pay, because compilation succeeds on the histories that did not need compiling.

We report one further quantity. An earlier run of this experiment was void: the history labels sessions by date and the compiler was asked to cite a session number, so 43% of citations pointed nowhere and

the verifier refused claims it could not check. The corrected pipeline aborts unless citations resolve. The invalid run's summary is in the supplementary material at `summaries/gate_result_run1_invalid.json`.

Cost. The gated route spends one compilation, one answer and two verifier calls per item against one call for reading the history, with the verifier calls short: the compiled state averages 125 output tokens and each verification is a single word.

O DELETING SUPERSEDED EVIDENCE: CONSTRUCTION, AUDITS, AND WHERE IT DOES NOT CARRY

Eligibility. A point in a stream counts when the answer that holds there is not the benchmark's refusal option, when a different answer held earlier, when annotated evidence for both the earlier and the current answer has already been seen, and when those two sets of sentences do not overlap. A sentence annotated as supporting both is not superseded evidence and removing it would not be the intervention. This leaves 241 points over 94 questions and all 19 books.

Locating a sentence. The benchmark writes its evidence with the line breaks of the printed page, so a sentence is found by matching its words with any whitespace between them and removed at the span that matches; 83% are found this way against 29% by literal comparison. The remainder are not spans but fragments joined by an ellipsis, and a point that needs one of them cannot have its evidence removed exactly and is not eligible. A sentence named as evidence is removed wherever it stands in the stream, since the annotation names where a fact is introduced rather than every place it occurs.

The matched deletion. CI takes text from the same chunks as CS, supporting no answer to the question, until as much text has gone as the evidence carried. One sentence suffices where lengths are close, which is how the synthetic half behaves; a narrative sentence can be several times longer than any single line of prose beside it, and there the control takes more than one. The two removals differ in size by 1.0% at the median and 5.9% at the ninetieth percentile.

Audits. On all 241 points we verified, against the original benchmark files rather than the code that wrote the contexts, that the superseded evidence is absent from CS and the current evidence is not, that the current evidence is absent from CN, that the matched text is absent from CI and overlaps no evidence sentence, and that the text CS and CI are built from is identical. Every check passes on every point.

Readers. Qwen3.5-9B at bfloat16 with greedy decoding, GPT-5.6-Luna and Gemini-3.7-Flash each at their high reasoning setting. Levels are not equalized across vendors, which do not mean the same thing by one; each reader is used at a strong setting so that the comparison is between readers properly used. Contexts run to the thirteenth chunk, about 21k tokens at the median.

The synthetic half. The same intervention on OAKS's BABI-derived streams, where a fact is one formulaic sentence, gives CS − CI of $+0.311$ for Qwen3.5-9B, $+0.146$ for GPT-5.6-Luna and $+0.039$ for Gemini-3.7-Flash over 254 points, each interval excluding zero under both clusterings. Accuracy without the intervention is 0.343, 0.709 and 0.898, so the effect is largest where most of it is available to recover. Deleting the current evidence there takes Qwen3.5-9B to 0.000.

Where it does not carry. We ran the same intervention, unchanged, on the benchmark's comparison questions, which ask which of two entities did something more often: 79 points over 18 questions. The effect is -0.076, interval $[-.291, +.151]$, and positive in three of nine documents. We report this as a boundary. Removing all the evidence for the superseded entity empties one side of the comparison, so the question is asked about something the stream no longer mentions; that is a change to the question rather than the removal of an interference. The narrative population above asks for a current state rather than a comparison between two, and the reading is consistent with the difference, but it was formed after seeing the comparison result, so we did not use it to define eligibility.

Dose. The number of superseded states a point has accumulated does not order the effect, on either half. On the novels it runs $+0.057$ and $+0.065$ for one and two accumulated states and 0.000 for three and four, the last two cells holding 17 and 3 points. On the synthetic half, removing half the superseded evidence rather than all of it leaves accuracy at 0.464 against 0.402 untouched and 0.629 with all of it removed: the order a graded effect would give, on 23 questions, with no contrast among them clearing zero. We report neither as a dose-response.

P Related work

Failures to act on new information have been reported from several directions: models that acknowledge corrective feedback without incorporating it (Jiang et al., 2025; Huang et al., 2024; Madaan et al., 2023; Welleck et al., 2023), updates that fail to propagate through multi-step reasoning when they conflict with parametric knowledge (Feng et al., 2026; Xie et al., 2024; Longpre et al., 2021), retrieved evidence that contradicts other retrieved evidence (Yuan et al., 2026; Chen et al., 2022; Asai et al., 2024), and knowledge edits that do not reach dependent facts (Zhong et al., 2023; Cohen et al., 2024; Hua et al., 2024; Hase et al., 2023). Mechanistic studies of these settings examine how conflicting signals are combined (Geva et al., 2023; Chen et al., 2026) and propose decoding-time corrections (Shi et al., 2024; Chuang et al., 2024). Our setting differs in what is held constant: the task state carries no item-specific counterpart in weights, no source needs to be adjudicated, and the penalty appears even when the two descriptions are identical and correct, so conflict with prior knowledge is not necessary for it. Repeating an entire prompt can instead help non-reasoning models (Leviathan et al., 2025); we duplicate only the task-state report and hold the downstream question fixed. What we add is causal evidence for the obsolete description's continuing contribution and a factorized intervention whose success depends on compilation fidelity.

Context rewriting also precedes answering in System 2 Attention (Weston & Sukhbaatar, 2023), which regenerates a context containing only relevant material. Our compilation is a task-specific form of context rewriting: it resolves updates into one explicit current state. We use this intervention to test whether separating state resolution from computation recovers the measured gap, and compare it with extra calls that retain the original descriptions. These controls isolate the role of state consolidation. Appendix Q additionally compares a relevance-based S2A adaptation on matched update chains, rather than benchmarking context rewriting in general. Our single-pass history compilation also differs from persistent memory management (Packer et al., 2023). We do not test rolling state updates, so its failures do not establish a limit on iterative memory management. STALE (Chao et al., 2026) also documents a gap between recognizing an update and acting on it, and proposes CUPMem, a state-aware memory baseline; we instead isolate the interference under matched component-success controls and intervene on the obsolete description itself.

Q Comparison with relevance-based context rewriting

We compare explicit state compilation with a relevance-based System 2 Attention (S2A) adaptation (Weston & Sukhbaatar, 2023) on the same 300 three-update chains as Table 2. The distributed generator, with seed 0, reproduces every item ID and pre/post answer mapping. Both 8B checkpoints, precision (BF16), answer instruction and scoring rule are fixed. All arms are rerun without capability filtering, using greedy decoding with thinking disabled. Both preprocessing arms receive 1,024 output tokens; the primary answer budget is 256 tokens.

Table 20: Matched context-rewriting comparison, $n = 300$ per model. Budget is the maximum number of answer tokens; both preprocessing budgets are 1,024. The last column is compilation minus S2A, with a 95% paired bootstrap interval. The 1,024-token answer runs have no truncated outputs in any arm.

Model	Budget	Direct	S2A	Compile	Compile − S2A
Qwen3-8B	256	.457	.573	.860	+.287 [.217,.357]
Llama-3.1-8B-Instruct	256	.303	.303	.503	+.200 [.123,.277]
Qwen3-8B	1,024	.513	.630	.857	+.227 [.160,.293]
Llama-3.1-8B-Instruct	1,024	.303	.313	.510	+.197 [.120,.273]

S2A adaptation. We paraphrase the relevance-based prompt in Figure 15 of Weston & Sukhbaatar (2023), using the system message "You are a careful analyst." The user message is the following instruction followed by the original report, updates, rule, question and options:

```
Rewrite the user text to retain the information relevant to
answering its question accurately.  Omit unrelated material
and preserve the question itself.  Return two sections
headed "Relevant context:" and "Detailed question:".  Write
prose rather than a list.
User text:  [original input]
```

The text between these two headings replaces the history. We reattach the original rule, question and options rather than use a rewritten question, keeping the scored task fixed. Heading matching ignores case and optional Markdown bold markers. If a heading is absent, the available rewrite is used. All 300 rewrites per model have both headings. The compiler uses the fixed state schema from Appendix H; unlike S2A, its first call does not receive the downstream question.

Budget sensitivity and scoring. Neither preprocessor truncates. At the 256-token answer cap, Qwen has 22 direct and 27 S2A truncations; Llama has one S2A truncation. Compiler answers do not truncate. After observing this, we fix a sensitivity check that increases every arm's answer budget to 1,024 on all items, reusing the same intermediate texts. Table 20 reports both budgets. Unparseable answers count as wrong; no item is excluded for truncation. Intervals use 20,000 item-paired percentile bootstrap resamples (seed 0). Compiler exact-state accuracy is 1.000 for Qwen and .830 for Llama; one Llama compilation does not parse and counts as wrong. Fresh execution and the changed budgets can shift rates slightly from Table 2, so these runs do not replace its scores.

Explicit compilation outperforms this S2A adaptation in both models and at both budgets. With 1,024 answer tokens, S2A itself improves on direct answering by +.117 [.040,.190] for Qwen and +.010 $[-.057, .077]$ for Llama. This comparison tests one relevance-rewriting prompt on short update chains; it does not establish superiority over all S2A variants or a long-history fidelity limit. Exact prompts, outputs, scored items and analysis code accompany the submission.