CAN SMALLER MODELS HELP LARGER AGENTS? MEASURING THE INCREMENTAL VALUE OF PEER ADVICE

Anonymous authors
Paper under double-blind review

ABSTRACT

Whether smaller models can provide more effective corrections than self-revision for the failed tasks of large language model agents remains a critical question, one that distinguishes a model's independent problem-solving capability from cross-model collaborative value. This paper conducts fair comparisons between pure self-revision and peer-assisted revision under unified model checkpoints and fixed revision budgets, evaluating all outcomes on hidden test sets across 300 competitive-programming tasks to quantify the gains and risks of peer advice. Advice from an 8B model raises a 35B MoE agent's success from 52.0% to 64.3%, and the gain survives assigning each packet to a different task; in the reverse direction, collaboration depends critically on the task–advice assignment, separating 63.7% under matched advice from 54.0% under mismatched advice. On shared tasks,the two matching effects differ by 15.2 percentage points, a difference reproduced across seeds. Yet a partner-free review that keeps the revision instruction but removes all advice leaves matched advice with no detectable advantage on a fixed 150-task subset, so part of the collaborative gain reflects a productive revision workflow rather than advice content; adding verified repair steps to the same diagnosis still improves the 35B agent by 6.4–14.0 points across three seeds.Accordingly, this paper establishes a paired evaluation framework that separates workflow gains, task-matching value, and content value,measures the increment of peer advice over partner-free review, and quantifies both its benefits and risks, offering new insights for lightweight collaborative optimization of large language model agents.

1 INTRODUCTION

A programming agent has a failing solution. Another model offers advice, the agent revises, and the program passes. It is tempting to credit the partner with finding the missing insight. Yet the same sequence can arise when the instruction to reconsider is useful, when generic guidance is sufficient, or when the partner provides a decisive repair. The final success rate does not distinguish these possibilities. *What, exactly, did the partner add?*

This question matters even when collaboration works well. We begin with a substantial gain: an 8B advisor raises a 35B MoE recipient from 52.0% success under self-revision to 64.3% on 300 programming tasks. A second, cross-family 8B advisor reaches 65.0% with matched advice, despite solving only 32.0% of the tasks independently. These results suggest that useful advising need not track standalone solving ability. They also create a sharper test: must the advice actually belong to the problem being revised?

For the MoE recipient, replacing each packet with advice for another task preserves the gain: the mismatched policy reaches 65.0%. For the 8B recipient in the reverse direction, matching instead separates 63.7% success from 54.0%. A direct comparison on 211 shared tasks finds a 15.2-point difference in matching effects; a second continuation seed and advice permutation reproduce it. Collaboration succeeds in both directions, but that shared outcome conceals different sensitivity to the task–advice assignment.

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.

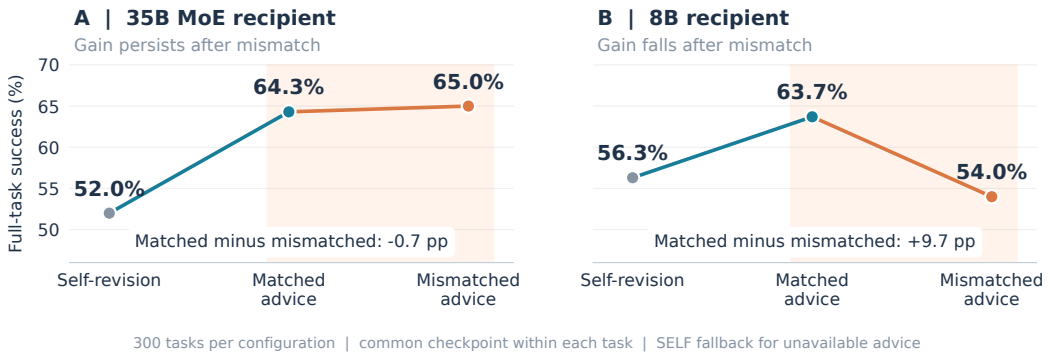

Figure 1: **The same collaboration gain invites different explanations.** Matched advice improves both configurations, but assigning it to other tasks preserves A's gain and reduces B's success. All 300 seed-11 tasks are included, with SELF fallback when advice is unavailable. Points are condition means; lines connect comparisons, not sequential revisions. Paired tests use the populations in Section 4.3.

The next comparison changes the question again. Mismatched advice can be distracting, so outperforming it need not mean that a partner is better than no partner. We retain the review instruction and remove all advice fields. On a fixed 150-task subset, this procedure improves the MoE recipient by 14.0 points. Matched advice has no detectable advantage over no-advisor review for either recipient on that subset. For the 8B recipient, this leaves an important distinction: matching matters on the full comparison set, but an advantage over review alone has not been established on the tested subset.

Does this mean that advice content is dispensable? An exploratory intervention offers a more constructive answer. For tasks with a verified reference repair, we hold the review instruction and diagnosis fixed and add concrete repair steps. This increases the MoE recipient's success by 6.4–14.0 points across three seeds. The experiment uses selected tasks and oracle-assisted advice, but identifies a content intervention worth pursuing: helping translate a diagnosis into a working revision. Small effects from one natural advice source do not imply that better repair guidance has no value.

Our contribution is an empirical account of *where the value of advice enters code revision*. We provide a directly tested, repeated difference in task-matching effects across two collaboration configurations; test advisor value against no-advisor review and alternative real sources; and probe repair guidance within a fixed review interface. Prior work already uses irrelevant-content and self-correction controls in debate (Bertalanič & Fortuna, 2026), and studies reflection versus peer influence (Hao et al., 2026). Building on those controls, we establish a distinction in executable code revision: **a workflow can improve, task matching can matter, and yet the increment over a partner-free review must still be established.**

2 Related Work

Revision and feedback. Self-Refine, Reflexion, self-debugging, and tool-interactive critique revise model outputs using additional reasoning or feedback (Madaan et al., 2023; Shinn et al., 2023; Chen et al., 2023b; Gou et al., 2024). Intrinsic self-correction can nevertheless be unreliable (Huang et al., 2024). DCR separates detection, critique, and refinement (Wadhwa et al., 2024); role-relabeling experiments show that presentation affects explicit error flagging (Chen et al., 2026). These studies motivate separating the revision interface from advice content. Our repair-step intervention tests executed program success, while leaving the internal route from diagnosis to implementation unresolved.

Controls for peer influence. Debate and aggregation combine model outputs (Du et al., 2024; Liang et al., 2023; Wang et al., 2024). The closest control design is Bertalanič & Fortuna (2026), which compares debate with unrelated-problem rationales and self-correction under a shared re-

vision structure. Hao et al. (2026) distinguish spontaneous answer changes from peer-associated changes using counterfactual conditions. Building on these reference conditions, we apply paired interventions to single-exchange code revision, directly compare matching effects across configurations, and contrast natural advice with controlled repair guidance. This links configuration-level differences in matching effects to the separate question of which content improves a review.

Selecting collaborators and evaluating code. Cascades and routers allocate inference across models (Chen et al., 2023a; Ong et al., 2024). COMED selectively escalates to peers while balancing rescues and regressions (Alballa et al., 2026). Weak-to-strong analyses separate a weak supervisor's guidance from a stronger model's capability (Burns et al., 2023); our source comparisons examine the analogous inference-time question of whose advice helps whom, and the NO ADVICE control bounds how far that value extends. Our experiments provide complementary evidence about the revision baselines and advice properties relevant to such decisions. CodeContests supplies executable outcomes (Li et al., 2022); LiveCodeBench and SWE-bench address complementary evaluation settings (Jain et al., 2025; Jimenez et al., 2024). Our findings concern the sampled competitive-programming tasks, rather than repository-level or contamination-free generalization.

3 METHOD

We treat advice value as a comparison between continuations from the same archived state. The protocol separates three questions: whether the collaborative workflow helps, whether its advice must match the task, and whether the partner improves on review alone.

3.1 PAIRED REVISION PROTOCOL

An exchange with a fixed recipient checkpoint. Models first produce independent drafts. An advisor then inspects the recipient's code and public-test feedback and returns a packet containing `target`, `claim`, `proposed_change`, `evidence`, and `scope`, or abstains. These five fields form the recipient-visible advice. The recipient can inspect, partially adopt, or reject the suggestion; the instruction explicitly allows incorrect or inapplicable advice. The archived task, draft, and public feedback are reused across continuations. In the main experiments, hidden tests are used only for final scoring. The exploratory feedback study partitions a separate diagnostic pool from its remaining evaluation tests (Appendix M).

Matched and mismatched advice. SELF continues the recipient's own revision. MATCHED uses correctly extracted advice produced for that checkpoint. MISMATCHED replaces this packet with one produced for a different task: packets are sorted by length, grouped in bins of ten, and deterministically rotated within bins using mapping seed 17. The audit reports zero self-matches and byte-identical non-advice prompt components between the advice conditions. Length binning reduces, but does not eliminate, differences in input length. All three continuations have a 6,144-completion-token cap, including thinking, and at most four public-test executions, each with a 10 s and 1 GB limit. LONG SELF doubles the output cap to 12,288.

NO ADVICE uses the same PEER review instruction but omits all advice fields. This isolates a deployable partner-free review procedure from the complete advice workflow. Corrected delivery is used throughout MATCHED; an earlier extraction defect and its LEGACY diagnostic condition are disclosed in Appendix A. B's mismatch pool includes 24 empty packets out of 233, which limits a purely semantic interpretation of its assignment intervention.

3.2 MEASURING THE PARTNER'S INCREMENT

Let y_{ir}^S, y_{ir}^M, and y_{ir}^X denote success under SELF, MATCHED, and MISMATCHED for task i and recipient r. For a set $\mathcal{I}_r$ with the required paired outcomes, define

$$G_r = \frac{1}{|\mathcal{I}_r|} \sum_{i \in \mathcal{I}_r} (y_{ir}^M - y_{ir}^S), \qquad R_r = \frac{1}{|\mathcal{I}_r|} \sum_{i \in \mathcal{I}_r} (y_{ir}^M - y_{ir}^X). \tag{1}$$

G_r measures the revision gain of the complete matched-advice workflow. R_r measures the advantage of matched over the implemented mismatched condition. We call the latter the *relevance*

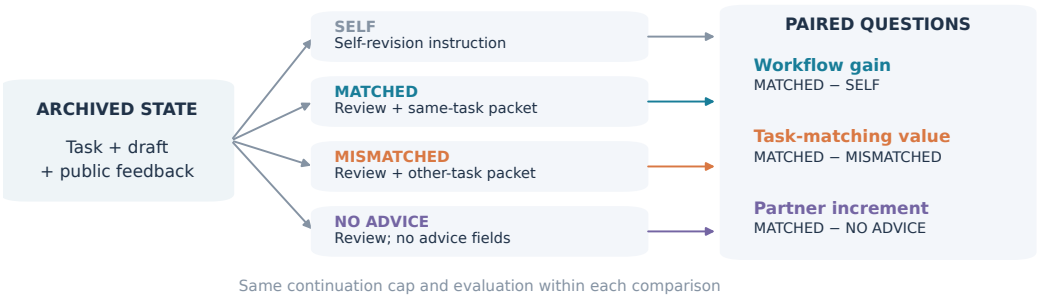

Figure 2: **One checkpoint, three questions about a partner.** All branches start from the archived task, draft, and public feedback. SELF uses its own continuation instruction; the other branches share the PEER review instruction. Pairwise contrasts test the workflow, task assignment, and increment over partner-free review. The main continuation cap is 6,144 tokens; hidden evaluation is common within each comparison. These contrasts are not an additive decomposition.

contrast; it is an outcome contrast, not a direct measure of internal information use. For NO ADVICE outcomes y_{ir}^N, the further contrast $V_r = |\mathcal{I}_r|^{-1} \sum_i (y_{ir}^M - y_{ir}^N)$ measures the increment over partner-free review on its own paired task set. Neither G_r nor R_r establishes that V_r is positive. On a common task set $\mathcal{K}$, the configuration interaction is

$$D = \frac{1}{|\mathcal{K}|} \sum_{i \in \mathcal{K}} \left[(y_{iB}^M - y_{iB}^X) - (y_{iA}^M - y_{iA}^X) \right]. \tag{2}$$

Resampling task IDs jointly preserves dependence across conditions and recipients, providing a direct uncertainty estimate for D.

Full-task performance. For each advice condition, the operational policy uses its continuation when eligible and SELF otherwise. We report this policy on all 300 tasks, including abstentions and parsing failures. The eligible-pair gain multiplied by its coverage gives the full-task gain. Eligibility comes from the runner's gate and does not guarantee a nonempty packet. Policy rates describe overall performance; paired subsets isolate comparisons within explicitly identified task sets.

3.3 Intervening on Repair Guidance

To distinguish diagnoses from guidance for acting on them, we construct paired diagnosis-only (D) and repair-step (R) packets. Both share the same diagnosis and evidence byte-for-byte; only R includes the proposed repair steps. A third, empty-field packet supplies the review baseline. All three use the PEER instruction. Reference repairs are execution-verified, and an oracle-assisted writer converts their differences from the draft into guidance without supplying the reference program itself. The contrast $R - D$ tests this added guidance package rather than a pure formatting change. Its selected population and exploratory status are specified in Section 4.5.

4 Experiments

4.1 Experimental Setup

Models and tasks. A is Qwen3.6-35B-A3B-FP8, a Qwen3-family MoE with 35B total and approximately 3B active parameters (Yang et al., 2025). B is Qwen3-VL-8B-Instruct, used through text only (Bai et al., 2025). C is InternVL3_5-8B (Wang et al., 2025), also text only, and serves as a cross-family advisor. Models are frozen and served with vLLM (Kwon et al., 2023). A and B use 32k contexts; C uses bf16 serving and a 16k context. We identify recipients by model rather than assign a capability ordering from parameter counts. The reported v4 paired-set draft accuracies of A and B are 49.2% and 48.9%; C solves 96/300 tasks independently.

The 300-task pilot is drawn from CodeContests (Li et al., 2022), using Codeforces difficulty categories 1–3. Filtering requires at least one native public test, three hidden tests, and a description no longer than 6,000 characters; splits are grouped by contest. Success requires passing every evaluated hidden test, up to ten per problem. Decoding uses temperature 0.6, top-p 0.95, and fixed run seeds. Drafts have a 6,144-token cap; advice has a separate 4,096-token cap and one schema-repair retry. Advice generation is outside the matched recipient continuation budget.

Runs and comparisons. MATCHED B→A and A→B each have seeds 11 and 23. MATCHED C→A and A→A use seed 11; C also generates an independent draft before advising. The initial MISMATCHED run uses seed-11 checkpoints and mapping seed 17. A replication retains those checkpoints and changes the continuation seed to 23 and the mapping seed to 18, rerunning both MATCHED and MISMATCHED. This differs from the seed-23 source runs, which have their own archived checkpoints and advice. Replays reuse archived SELF outcomes and rerun the advice continuation. The initial SELF/PEER collection randomizes branch order within tasks; subsequent replays are separate runs. Paired sign-flip tests compare MATCHED with MISMATCHED on 266 A tasks and 233 B tasks. The direct interaction uses 211 shared tasks and joint task-level bootstrap resampling (Efron & Tibshirani, 1994). Reported tests are nominal.

No-advisor review and source controls. For both A and B, NO ADVICE retains the PEER system instruction but removes all advice fields from the user payload. For A, GENERIC supplies a fixed, task-independent checklist covering input parsing, boundary conditions, and computational complexity under the same instruction. These controls use the first 150 sorted pilot task IDs, fixed before their respective runs, with the original continuation budget. Every comparison on this subset uses all 150 tasks and SELF fallback for unavailable advice. We also evaluate C→B on 300 tasks using corrected delivery and an independent C draft before advice. Source comparisons intersect the eligible tasks of the two corrected conditions. Appendix I records these interventions and their analysis populations.

Alternative uses of inference. LONG SELF raises the effective continuation cap to 12,288 for each recipient. For A, an intermediate 8,192-token run and a second independent SELF continuation provide additional controls. A public-test selector chooses the second output only when it passes more public tests, keeping the first on ties. Appendix E records the effective request limits, including the correction of an initially client-limited budget run. These controls compare ways of revising, not equal-cost deployments.

4.2 Collaboration Improves Revision

Small advisors offer a productive route to revision. Advice from B raises A's full-task success from 52.0% to 64.3%, producing 41 corrections and four regressions on 267 eligible tasks (Table 1). Advice from C reaches 65.0%, with 44 corrections and five regressions on 291 tasks, despite C's 32.0% standalone success. Self-advice is weakest: A→A reaches 60.0% on the full task set, with eligibility on only 126 tasks (Table 1). The reverse direction also benefits: A's advice raises B from 56.3% to 63.7%. Corrected matched advice improves outcomes in seed 23 as well, reaching 63.3% for A (SELF 56.7%) and 64.0% for B (SELF 57.7%). Thus the question is not prompted by a failure to obtain collaboration gains. Both recipients have gains whose dependence on the advice can be tested.

The gains respond differently to task mismatch. For A, assigning packets to other tasks preserves high performance: the mismatched policy reaches 65.0%, with 46 corrections and seven regressions on 266 eligible tasks. For B, the same intervention yields 54.0%, with four corrections and 11 regressions on 233 tasks. Figure 1 makes the contrast visible. A's revision benefit survives disruption of the original assignment; B's matched policy exceeds its mismatched policy by 9.7 points over all 300 tasks. The matched-advice success rates alone would have concealed this difference in how the workflows respond to task matching.

Table 1: **Matched-advice results.** n denotes eligible pairs; help/harm counts compare against SELF. Δ is the eligible-pair gain in pp. SELF_{300} and Policy_{300} are full-task success percentages, with SELF fallback when advice is unavailable. Corrected delivery throughout; seed-23 rows rerun the continuation under a new seed.

Direction	Seed	n	Help/harm	Δ (pp)	SELF_{300}	Policy_{300}
B→A	11	267	41/4	+13.9	52.0	64.3
B→A	23	273	26/6	+7.3	56.7	63.3
C→A	11	291	44/5	+13.4	52.0	65.0
A→A	11	126	29/5	+19.0	52.0	60.0
A→B	11	233	39/17	+9.4	56.3	63.7
A→B	23	260	41/22	+7.3	57.7	64.0

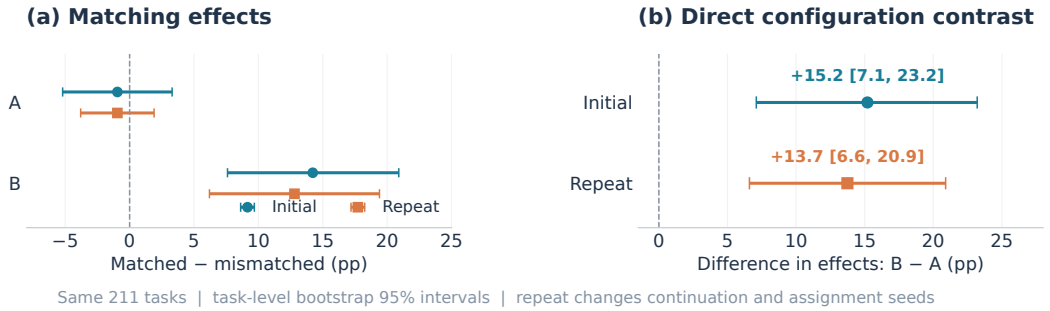

Figure 3: **The matching-effect difference survives a new generation and assignment.** Both panels use the same 211 tasks. Left: matched minus mismatched advice by configuration. Right: the direct difference between those effects. Whiskers show reported 95% task-bootstrap intervals. Blue denotes the initial run and orange the repetition; the latter changes continuation and mapping seeds on the same checkpoints.

4.3 Does the Advice Need to Match the Task?

Task matching has a measurable advantage for B. On B's 233-task comparison set, MATCHED exceeds MISMATCHED by 12.45 pp ($p = 0.0005$). On A's 266-task set, the contrast is -0.75 pp ($p = 0.85$; 95% CI $[-4.5, 3.0]$). The evidence therefore supports a matching advantage for B and no detectable advantage for A at this precision. To determine whether the contrasts themselves differ, we place both directions on the same task set.

The difference remains on the same 211 tasks. For A, MATCHED and MISMATCHED gain 14.7 and 15.6 pp over SELF. For B, they gain 10.4 and -3.8 pp. The relevance contrasts are -0.95 pp for A and $+14.22$ pp for B; their difference is $+15.2$ pp (95% CI $[7.1, 23.2]$) under joint task-level resampling (Figure 3). Restricting the comparison to identical tasks thus preserves the separation between the two advice directions.

The result persists under a new generation and assignment. The initial contrast could depend on a particular permutation or set of sampled continuations. We rerun both advice conditions on the same checkpoints with continuation seed 23 and mapping seed 18 (Table 2). The interaction remains $+13.74$ pp (95% CI $[6.6, 20.9]$), and across the full eligible sets B again benefits from matching ($+10.73$ pp, $p = 0.0009$), while A's contrast is -1.50 pp ($p = 0.449$). The two runs agree on the central pattern despite changing both continuation randomness and advice assignment.

For B, the source of real advice also matters. We next replace A's advice to B with advice from the cross-family model C (Table 3). For B, the two implemented sources differ sharply: on the tasks shared with corrected A→B, A's packets produce 39 corrections against 17 regressions while C's produce ten against 20, a $+13.79$ pp paired difference in favor of A's advice ($p = 0.0001$); C alone is not a significant change from SELF. For recipient A, corrected B and C advice differ by only $+0.77$ pp ($p = 0.80$). These are separate source comparisons, not a cross-recipient interaction: the

Table 2: **Relevance contrasts persist across two seed combinations.** All entries use the same 211 common tasks. Differences are percentage points, with 95% task-bootstrap intervals. The interaction is B minus A; rounding the displayed components can change their difference by 0.01 pp.

Contrast	Continuation 11 / mapping 17	Continuation 23 / mapping 18
A: matched − mismatched	-0.95 $[-5.2, 3.3]$	-0.95 $[-3.8, 1.9]$
B: matched − mismatched	$+14.22$ $[7.6, 20.9]$	$+12.80$ $[6.2, 19.4]$
Interaction: B − A	$+15.17$ $[7.1, 23.2]$	$+13.74$ $[6.6, 20.9]$

Table 3: **Advice-source comparisons under corrected delivery.** Help/harm counts compare each advisor's matched continuation against SELF on the stated task sets; percentages are success rates under the two advisors. Paired differences are computed within each row, and the B-direction difference favors A-source advice. The 66 C→B eligible tasks outside the shared set have zero net gain. These are separate source comparisons, not a cross-recipient interaction.

Recipient	Comparison	n	Result
B	C advice vs. SELF, all eligible	298	14/24 help/harm; -3.4 pp ($p \simeq 0.15$)
B	A vs. C advice, shared set	232	39/17 vs. 10/20; $+13.79$ pp ($p = 0.0001$)
A	B vs. C advice, shared set	260	73.1% vs. 72.3%; $+0.77$ pp ($p = 0.80$)

advisor pairs differ. For A, the undetected difference is not an equivalence claim, and neither result identifies a cause such as advisor size.

4.4 Does a Partner Improve on Review Alone?

For A, a revision gain does not require an advisor call. Task-mismatched advice could still supply useful generic guidance. We therefore remove the packet entirely while retaining the review instruction. On the predetermined 150-task subset, this NO ADVICE condition raises A from 48.7% to 62.7%. MATCHED reaches 62.0% on the same full subset (Table 4); its paired difference from NO ADVICE is -0.67 pp, with 95% CI $[-6.0, 4.0]$. Thus a substantial revision gain is obtainable without generating or delivering a partner's suggestion. The effective intervention is the implemented review instruction; its individual wording components are not separated by this comparison.

Table 4: **Matching and partner value are different comparisons.** Same 150 task IDs, with SELF fallback for unavailable advice. Success rates are percentages; the last row is a paired difference in pp with 95% CI. These subset rates are separate from the 300-task policies.

Condition	A (35B MoE)	B (8B)
SELF	48.7	60.7
NO ADVICE	62.7	59.3
MATCHED	62.0	58.7
MISMATCHED	62.7	57.3
MATCHED − NO ADVICE	-0.67 $[-6.0, 4.0]$	-0.67 $[-7.3, 6.0]$

For B, task matching does not yet establish partner value. The corresponding NO ADVICE run for B reaches 59.3%, compared with 60.7% under SELF and 58.7% with MATCHED. The prespecified primary contrast, MATCHED minus NO ADVICE, is -0.67 pp ($[-7.3, 6.0]$; two-sided $p = 1.0$). This does not establish equivalence: the interval allows meaningful differences in either direction. It does establish that the full-set matched–mismatched result cannot substitute for a test against partner-free review. An exploratory split analysis finds different matching effects in the first and second halves of the pilot (Appendix L); the second half has no NO ADVICE run.

For A, a fixed generic checklist reaches 66.0%, versus 62.7% for NO ADVICE ($+3.33$ pp, $[-0.7, 7.3]$). The estimates leave room for useful guidance without establishing a generic-content advantage.

A larger revision allowance gives a different kind of help. Doubling the SELF cap reaches 63.0% for A, compared with 59.0% at 8,192 tokens and 57.7% for two SELF outputs with public-test selection. For B, doubling the cap reaches 57.3%, one point above SELF and below matched advice. B makes limited use of the allowance: its median output is 316 tokens and only 22/300 continuations exceed 8,192. These are budget-cap comparisons, with different realized computation. They identify competitive revision alternatives while showing that the recipients use a larger allowance differently. We next hold the review interface fixed and intervene on the guidance itself.

4.5 Can Better Repair Guidance Add Value?

The preceding results locate a tension. A gains from review without a partner, and its natural packets show little detectable task-matching advantage. Yet neither observation implies that useful content cannot improve the review. We therefore intervene on *what the advice contains*, rather than merely which task it comes from.

Hold the diagnosis fixed; add repair steps. We apply the repair-guidance intervention of Section 3.3 to A in an exploratory study (DRX), using the same 6,144-token cap for empty review, D, and R. Across seeds 11, 23, and 47, the study includes 57/73, 57/73, and 63/78 candidate failing/mixed tasks for which such a repair could be obtained. These are selected tasks from the already explored pilot, not a fresh confirmation set.

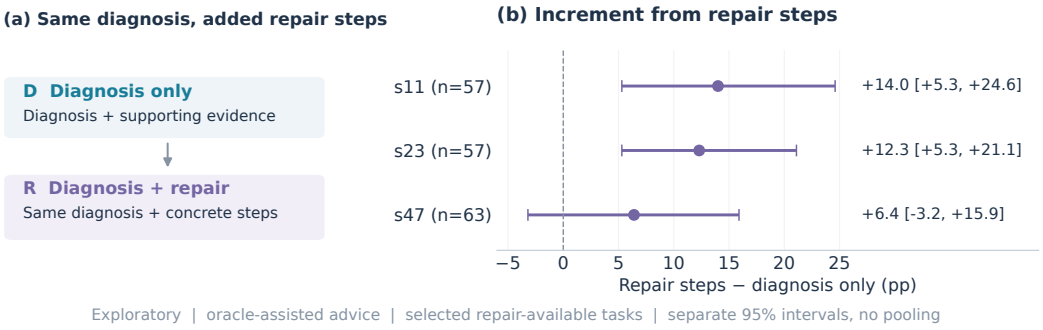

Figure 4: **Content can help even when natural advice has little matching advantage.** Adding verified repair steps to the same diagnosis improves A's point estimate in three exploratory runs. Bars show reported 95% task-bootstrap intervals; seed populations overlap and are not pooled. All conditions share the review instruction. Selected repair-available tasks and oracle-assisted advice limit generalization.

Adding repair steps improves the point estimate in every seed (Figure 4); paired tests give $p = 0.008$ and $p = 0.016$ in the first two runs. In seed 11, empty review, diagnosis, and repair reach 70.2%, 66.7%, and 80.7%, respectively. Thus a content intervention can improve a recipient whose earlier natural-advice matching contrast was small. Diagnosis alone has no detected increment over the empty review in these runs, but the experiment does not establish that diagnosis is unnecessary. R adds both information and implementation guidance, and repair availability selects the population.

Information also has to fit the revision budget. A companion feedback intervention (FBX) adds diagnostic test evidence while retaining the natural advice packet. Under the 6,144-token cap, enhanced-feedback SELF performs worse than base SELF by 3.8, 12.8, and 12.4 points across the three runs; the first interval includes zero. On a 208-task, draft-present subset in seed 23, increasing the effective cap to 8,192 recovers 11.5 points ($[4.3, 18.8]$) relative to the lower-cap enhanced-feedback condition. This ties the observed penalty to the tested resource setting rather than showing that diagnostic information is intrinsically harmful. FBX changes the review framework along with advice, so its interaction is not a pure content effect. Appendix M reports the design and estimates separately from the main evaluation.

The constructive lesson is narrower than a universal mechanism and more useful than declaring advice irrelevant: *a review can work without a partner, while carefully specified repair content still*

improves what that review achieves. Natural-advice gains, repair guidance, and the budget available to act on feedback need separate tests.

4.6 Discussion and Limitations

Credit the increment, not just the successful workflow. The results distinguish three claims that a successful collaboration can otherwise collapse. SELF tests whether the workflow improves revision; MISMATCHED tests the implemented task assignment; NO ADVICE tests whether a partner improves on review alone. Source and repair-step interventions ask which content changes that increment. These are complementary references, not additive pieces of an identified causal mechanism. A system designer should compare against a competitive partner-free procedure before paying for an advisor, and should not infer the value of all possible advice from one source's packets.

Availability and cost remain part of the decision. A's self-advice is eligible on only 126/300 tasks, whereas B and C cover 89% and 97% in seed 11. NO ADVICE bypasses that requirement; oracle-assisted repairs introduce a different availability constraint. Advice generation, verification, and continuation are distinct costs. Shorter recipient outputs do not establish lower total compute, and our cap comparisons do not establish matched-cost superiority (Appendix G).

What the evidence does and does not generalize. The matching interaction compares configurations: changing recipient also changes advisor and packet distribution. It does not isolate model size or recipient identity. Mismatch preserves approximate length, not all semantic or delivery properties; B's pool includes empty packets. No detected difference is not equivalence. The 150-task review controls do not cover the remaining tasks, and the seed repetitions do not supply new-task validation.

DRX and FBX are exploratory follow-ups on a repeatedly analyzed pilot. DRX excludes tasks without a verified repair and uses oracle-assisted content. Its seed-specific cohorts overlap, so we do not pool their discordant counts as independent observations. FBX uses a partitioned test pool and changes instruction framework across advice conditions; its aggregate interaction cannot uniquely identify information complementarity. New tasks, additional recipients, and a fully crossed framework–content–budget design are needed for broader mechanism claims.

5 Conclusion

An agent's improvement after consultation is the beginning of an explanation, not its end. In paired code revisions, large gains can survive mismatched advice and even the removal of the advisor, while other configurations respond strongly to matching and source; controlled repair guidance can nevertheless add value within the same review interface. The evidence supports a qualified answer to whether smaller models help larger agents: the headline gains replicate across two continuation seeds, and a cross-family advisor that solves fewer than half the tasks independently still lifts the 35B recipient by over 13 points, but the 35B recipient's entire improvement was reproduced by a partner-free review, the 8B recipient showed no detectable benefit from that review or from matched advice on the tested subset, and two real advisors' packets differed by 13.8 points for the same recipient. A designer should therefore price an advisor call against the partner-free review, verify delivery against the intended assignment, track regressions as carefully as corrections, and stress-test any collaboration gain against task–advice reassignment. These contrasts are also cheap: every comparison replays the same archived checkpoint, so the marginal cost of separating workflow, matching, and content is one additional continuation per condition. Why matching is decisive for one configuration and undetectable for the other, and when specific advice content pays for itself, remain open—as does extending the paired protocol beyond single-exchange revision and beyond the sampled competitive-programming tasks. The habit these results argue for is to measure the increment a partner adds rather than assume it, whether the advice comes from a smaller peer, a cross-family peer, or the model itself.

AI USE STATEMENT

In this work, we used generative AI tools to assist in implementing parts of the code. We have not used generative AI tools for generating synthetic datasets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on the research methodology or experiments, assisting with translation, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, or interpreting results, and the remaining required-disclosure tasks are not applicable to this work. We have reviewed all AI-assisted work: the AI-assisted code was checked by two authors to ensure its correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

The study uses a public programming benchmark and involves no human subjects. Advice can introduce regressions as well as corrections; we report both. Systems using cross-model advice should evaluate this risk and verify content delivery before deployment.

REPRODUCIBILITY STATEMENT

Sections 3 and 4.1 specify models, budgets, scoring, and replay conditions. The appendices distinguish corrected experiments from legacy runs and describe advice eligibility and cost accounting. Aggregate tables and the relevance figures are generated from a versioned summary of the reported results. Reproducing the paired content tests requires the archived task records, raw advice, recipient prompts, and runner configurations described in Appendix N.

REFERENCES

Norah Alballa, Wenxuan Zhang, Salma Kharrat, Fares Fourati, Zafar Ayyub Qazi, Mohamed Elhoseiny, and Marco Canini. COMED: The missing middle between routing and collaboration in multi-LLM inference. *arXiv preprint arXiv:2609.26913*, 2026.

Shuai Bai et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Blaž Bertalanič and Carolina Fortuna. The cost of consensus: Isolated self-correction prevails over unguided homogeneous multi-agent debate. *arXiv preprint arXiv:2605.00914*, 2026.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, et al. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*, 2023.

Kuan-Yen Chen, Fang-Yi Su, Shih-Yen Lin, Bao Li, and Jung-Hsien Chiang. The self-correction illusion: Role relabeling gates explicit error flagging in large language models. *arXiv preprint arXiv:2606.05976*, 2026.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023a.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*, 2023b.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC, 1994.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yangruo Huang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *The Twelfth International Conference on Learning Representations*, 2024.

Xiqi Hao, Zengqing Wu, Yu-Xuan Qiu, Chuan Xiao, Ruiqi Xu, Shuyuan Zheng, and Jianbin Qin. Not all flips are conformity: Decomposing stance convergence in multi-agent LLM debate. *arXiv preprint arXiv:2606.00820*, 2026.

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 *The Twelfth International Conference on Learning Representations*, 2024.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Ofman, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *The Twelfth International Conference on Learning Representations*, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. 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*, 2023.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with AlphaCode. *Science*, 378(6624):1092–1097, 2022.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skylar Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route llms with preference data. *arXiv preprint arXiv:2406.18665*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

Manya Wadhwa, Xinyu Zhao, Junyi Jessy Li, and Greg Durrett. Learning to refine with fine-grained natural language feedback. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. *arXiv preprint arXiv:2406.04692*, 2024.

Weiyun Wang, Zhangwei Gao, Lixin Gu, et al. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

A Delivery Audit and Defect Disclosure

The original JSON extractor could select an object nested in an advice packet's evidence array instead of the packet itself. Rendering that object into the five expected fields yielded an empty advice view. The corrected parser prioritizes schema-bearing objects; the reported regression suite passes all ten tests. Seed-11 legacy delivery is empty in 259/266 B→A eligible cases and 233/233 A→B cases. Legacy A→A and C→A are also identified as defect-affected in the experiment report. Their measured gains must not be described as validated semantic-advice effects.

Reparsing raw archived outputs initially identifies complete content in 212/233 A→B packets. A later, explicit nonempty-claim criterion retains 209/233. The later audit uses 209 as its content-received count and reports 24 unrecoverable packets. These predicates and versions are not interchangeable counts. The latter partitions the reported subset analysis into 209 content-received and 24 claim-empty tasks. Claim emptiness alone does not establish that every other field is empty.

Restored B→A has 267 eligible tasks, compared with 266 in the legacy collection. The final counts are 151 both-success, 71 both-fail, 41 corrections, and 4 regressions. Earlier intermediate reads with 265 pairs are superseded. The source report identifies one truncated record as an abstention task. All full-task policies use 300 tasks; matched content tests require intersecting the actual recorded outcomes.

The delivery audit reports increased prompt length after restoration and a byte-level example contrasting an empty claim with its restored text. Task, draft, and public-feedback fields are reconstructed from the same archive. These checks support content-delivery validation; token-count changes alone are not a proof that every non-advice prompt component is unchanged. SELF and PEER use different revision instructions, and this is part of the overall workflow contrast.

B Corrected Arms and Coverage

Corrected runs are `cf_corr_A` (B→A), `cf_corr_B` (A→B), `cf_corr_AA`, `cf_corr_CA`, `cf_corr_B_s23`, and `cf_corr_A_s23`. All use archived raw advice from the corresponding source run and rerun the advice-assisted continuation. Corrected A→B seed 23 has 260 eligible tasks, 30 abstentions, and 10 parsing failures. Corrected B→A seed 23 has 273 eligible tasks, 27 abstentions, and no parsing failures. Its 26 corrections and six regressions give a $+7.3$ pp gain. The seed-11 breakdown is below.

Table 5: Runner eligibility on 300 seed-11 tasks per direction. True abstention is separated from parsing failure.

Direction	Eligible	Abstain	Parse fail	Coverage (%)
B→A	267	33	0	89.0
C→A	291	9	0	97.0
A→A	126	152	22	42.0
A→B	233	37	30	77.7

C Relevance Contrasts and Analysis Populations

All policy bars use 300 tasks with SELF fallback. The matched–mismatched paired tests use 266 A tasks and 233 B tasks. Their point estimates differ by $12.45 - (-0.75) = 13.20$ pp, but these are different task sets. The interaction uses 211 common tasks, on which the contrasts are -0.95 pp for A and $+14.22$ pp for B, giving 15.17 pp before rounding. The reported joint task-bootstrap interval is $[7.1, 23.2]$ pp. The two recipients and all outcomes for a sampled task are retained together during resampling. The rounded gains in Figure 3 are not used to reconstruct the test.

Mismatched advice uses length bins of ten and deterministic rotation with seed 17. The audit finds no self-matches and content in 266/266 A packets and 209/233 B packets. Non-advice prompt components are reported byte-identical between matched and mismatched conditions. The equality concerns those components, not the entire prompt, which necessarily changes with the packet. Equal

Table 6: Matched–mismatched paired comparisons on each recipient's eligible comparison set. These are not the common-set interaction.

Analysis set	n	Matched − mismatched (pp)	p
Recipient A	266	-0.75	0.8500
Recipient B	233	+12.45	0.0005

nonempty counts in B's matched and mismatched pools do not establish that the same individual tasks receive nonempty packets. The reported comparison includes the implemented pool, rather than claiming a content-preserving intervention on every B task.

The replication keeps the same 211 checkpoints and jointly changes continuation seed 11 to 23 and mapping seed 17 to 18. It does not isolate the two sources of randomness or introduce new tasks. On A's full 266-task comparison set, the replicated contrast is -1.50 pp with 95% CI $[-4.5, 1.5]$; on the 211-task common set it is -0.95 pp with CI $[-3.8, 1.9]$. B's common-set contrast is $+12.80$ pp with CI $[6.2, 19.4]$. The difference computed from the unrounded values is 13.74 pp; rounding the displayed components before subtraction instead yields 13.75 pp.

D Matched Advice and Delivery Comparisons

Table 1 in the main text (Section 4.2) gives eligible and full-task matched-advice results. Table 7 separately compares repaired delivery against the legacy workflow.

Table 7: Reported matched-minus-legacy increments (pp), with paired sign-flip tests. These compare delivery conditions, not task matching.

Direction	Seed	Content increment (pp)	p
B→A	11	+1.1	0.635
B→A	23	+0.7	0.834
A→A	11	+1.6	0.826
C→A	11	+1.7	0.405
A→B	11	+7.7	0.026
A→B	23	+9.6	0.003

Removing the seven nonempty legacy B→A packets leaves 259 strictly empty cases; restoring advice adds $+1.5$ pp on this set. The full reported paired increment is $+1.1$ pp. In A→B, the 209 seed-11 cases with a nonempty claim have 39 corrections and 16 regressions under matched delivery, a $+11.0$ pp gain over SELF. Legacy delivery gains $+1.4$ pp on the same subset, giving a content-restoration increment of $+9.6$ pp. The remaining 24 cases have a -4.2 pp gain over SELF. These subset analyses change the evaluated population and are separated from full-task policies.

E Budget and Sampling Controls

The first budget run, `cf_budget12288`, was effectively capped at 8,192 by the client despite its 12,288 stage setting. The corrected `cf_budget12288_r2` sets the client limit to 16,384. Its maximum observed output is 12,288; 197 of 300 continuations exceed 8,192, 143 reach the cap, and the median is 11,588. This verifies the effective doubled budget. The 8,192 run remains a correctly labeled intermediate-budget observation, not a doubled-budget result.

For B, LONG SELF reaches 172/300 successes (57.3%), with seven corrections and four regressions relative to SELF. Its output median is 316 tokens and only 22/300 continuations exceed 8,192 tokens.

The independent second SELF arm uses seed 1011 and produces 33 corrections and 30 regressions relative to the archived SELF output. The public-test selector accepts the new output only for a strictly higher number of public tests passed. Table 8 reports all A-policy rates.

Table 8: Full-task A-policy success on 300 tasks. Restored advice uses SELF fallback; gain is relative to the 52.0% SELF baseline.

Strategy	SR (%)	Gain (pp)
SELF, 6,144	52.0	+0.0
Second SELF alone	53.0	+1.0
Two SELF + selection	57.7	+5.7
SELF, 8,192	59.0	+7.0
SELF, 12,288	63.0	+11.0
A→A, restored	60.0	+8.0
B→A, restored	64.3	+12.3
C→A, restored	65.0	+13.0

F Legacy Results and Their Interpretation

The pilot v4 seeds 11/23/47, dev, and gate-calibration runs predate the delivery correction. Their task-clustered gains were $+9.1$ pp $[6.4, 11.9]$ for B→A and -0.1 pp $[-2.0, 1.9]$ for A→B; the dev B→A gain was $+10.5$ pp. These describe the legacy workflow. We do not use them as replications of restored advice, nor retain the earlier conclusion that only the larger recipient benefits.

Earlier strata, truncation, oracle-selection, and gate experiments also used this delivery implementation. Their figures and tables are retained in the project archive but excluded from the corrected main evidence. The v3 collection additionally had extraction and prompt-compression confounds and is not a clean comparator for either delivery or budget.

G Token Accounting

The updated ledger summaries separate model and condition:

Recipient	Condition	Mean output tokens
A	Legacy SELF	5,480
A	Legacy PEER	3,945
A	Restored B→A	3,933
A	Restored C→A	3,995
A	Restored A→A	4,374
A	SELF, 12,288 cap	9,263
B	Legacy SELF	963
B	Legacy PEER	1,215
B	Restored A→B	899
B	SELF, 12,288 cap	1,376

These are reported run-level means, not matched-task savings estimates. The increase in A's reported mean under LONG SELF is $9{,}263 - 5{,}480 = 3{,}783$ output tokens. For B it is $1{,}376 - 963 = 413$. Earlier mixed-direction means of 3,221 for SELF and 2,670 for PEER are superseded. B advice calls average approximately 1,791 input and 215 output tokens; C advice averages 1,753 and 178. A self-advice calls average 3,271 input and 3,412 output tokens. C's total 894k tokens include its 300 independent drafts and advice generation. Input, output, and model identity should remain separate in efficiency comparisons. No dense/MoE token conversion or matched-compute advantage is asserted.

H Selected Revision Cases

The following descriptions use three corrected-delivery cases checked against the experiment records by the experiment pipeline. Two successes were selected from 19 C→A candidates where C's independent draft failed and the matched recipient continuation succeeded; one regression was selected from 16 corrected A→B regression candidates. These are illustrations, not a representative sample. The verification summaries report agreement between delivered fields and archived

advice, generated code and continuation records, and recorded hidden-test outcomes, with sampled rescoring. Full bundles are identified as `cc_00018.md`, `cc_00029.md`, and `cc_00001.md`.

A viable algorithm can precede successful implementation. In `cc_00018` (`prime1`), the task is to list primes in an interval ending at up to 10^9, with width at most 10^5. Both C's independent draft and A's SELF continuation fail. A's draft reasoning already mentions a segmented sieve, but its extracted output is a short list of numbers rather than executable code. C's delivered packet recommends a segmented sieve to reduce memory use. A's matched continuation implements base primes up to $\sqrt{n}$ and interval marking, and passes the evaluated hidden tests. The algorithmic suggestion is appropriate, but the packet's memory-error diagnosis is not established by the supplied account of A's non-code output. Nor does the case show that A learned the algorithm from C: it is consistent with renewed execution of an idea already present in A's reasoning.

A formula-directed critique accompanies a successful revision. In `cc_00029` (`j7`), the task maximizes a cuboid's volume subject to wire-length and surface-area constraints. A's SELF output fails; C's delivered packet challenges the volume formula and supplies a derivation in its evidence field. The matched continuation passes the evaluated hidden tests. This establishes that the critique targets a concrete mathematical component and that the paired outcomes differ. The supplied summary does not isolate whether the decisive change was the formula, numerical evaluation, boundary handling, or another edit; we therefore do not label this a verified correction of an algebraic error or a precision error.

Broad advice can accompany a regression. In `cc_00001` (`gcd2`), B must compute the GCD of a small integer and an integer represented by up to 250 digits. SELF succeeds. A's delivered advice instead raises broad concerns about input handling, imports, and possible syntax or logic errors. B's matched continuation uses digit-by-digit modular reduction but fails hidden evaluation. Digit-by-digit reduction is a valid approach to this problem; the summary does not identify the exact faulty edit or demonstrate that the advice caused it. The case is a verified outcome regression under matched advice, not evidence that modular reduction is intrinsically misguided. It also differs from the deliberately mismatched intervention and is not treated as a causal explanation of that aggregate result.

I NO-ADVISOR, GENERIC-ADVICE, AND SOURCE CONTROLS

NO ADVICE (`cf_pa_A`) and GENERIC (`cf_ga_A`) use the first 150 pilot checkpoints fixed before execution. SELF succeeds on 73 tasks; LEGACY, NO ADVICE, MATCHED, MISMATCHED, and GENERIC succeed on 93, 94, 93, 94, and 99, respectively. Each row includes the original SELF outcome for tasks without an eligible advice continuation. Table 9 reports direct paired differences.

Table 9: Paired differences on the full 150-task subset. Discordant counts favor the first or second condition, respectively. Intervals apply to the specified contrasts rather than a universal content effect.

Contrast	Difference (pp)	95% CI	Discordant $+/-$
Matched minus no advice	-0.67	$[-6.0, +4.0]$	7/8
Matched minus mismatched	-0.67	$[-5.3, +4.0]$	6/7
Generic minus no advice	+3.33	$[-0.7, +7.3]$	7/2
Generic minus matched	+4.00	$[-0.7, +8.7]$	10/4

The no-advisor intervention retains the PEER system instruction while removing all advice fields. It establishes an effective procedure without advice delivery, but does not uniquely identify a framing mechanism. Generic guidance, autonomous revision, and instruction-driven changes in generation can coexist. The generic checklist asks the recipient to examine input parsing, boundary conditions, and complexity. It is the same across tasks and requires no advisor generation.

The C→B source run (`cf_crossCB`) has 298 eligible tasks. The source contrast against corrected A→B uses 232 common tasks and 48/16 discordant successes, giving $+13.79$ pp ($p = 0.0001$). C→B alone has a -3.4 pp eligible gain with $p \simeq 0.15$; the significant source difference does not

establish significant harm from C relative to SELF. For A, the corrected B-versus-C comparison uses 260 common tasks and 190 versus 188 successes ($+0.77$ pp, $p = 0.80$). The previously reported $+6.03$ pp B-source contrast used legacy A advice and is superseded by the corrected-content comparison.

J Exploratory Self-Reported Advice Use

The continuation instruction requests a SUMMARY of advice use. Keyword classification gives Table 10; missing summaries are retained so that every row closes to its eligible count. A frequently reports rejection or an unclear response, while B reports partial adoption more often with matched advice and rejection more often with mismatched advice. These are properties of self-reports, not verified code-edit adoption. Missing summaries are associated with failures, so analyses restricted to explicit categories are selective. The analysis is exploratory and does not identify a content-use mechanism.

Table 10: SUMMARY classifications, including missing summaries. "Reject" includes self-reported inapplicability; it does not certify that the suggestion was correctly assessed or had no influence.

Condition	n	Adopt	Partial	Reject	Other	Missing
A matched	267	7	3	74	155	28
A mismatched	266	4	1	97	145	19
B matched	233	2	188	31	1	11
B mismatched	233	3	41	174	1	14

K Extended Delivery Audit

The runtime ledger records median input-token differences relative to SELF of $+257$ for matched and mismatched advice, $+119$ for GENERIC, and $+26$ for NO ADVICE. A separate text audit reconstructs prompts through the production code and tests nonempty claim inclusion: 268/268 matched claims, 266/266 mismatched claims, and 150/150 generic checklists are found. Empty claims are excluded from the denominator. The 268 matched claims comprise 267 eligible tasks and one abstention with a nonempty claim (`cc_00101`); reconstructed inclusion on that abstention is not an additional delivered or evaluated PEER continuation. Runtime token measurements and reconstructed text checks are complementary evidence; the latter are not archived request bodies.

L No-Advisor Review for B and Exploratory Outcome Decompositions

The B no-advice run uses the same first 150 sorted task IDs as A, with B's own archived checkpoints and continuation seed 11. The primary contrast was specified before running this condition; no formal registry entry is asserted. SELF, NO ADVICE, MATCHED, and MISMATCHED succeed on 91, 89, 88, and 86 tasks. NO ADVICE minus SELF is -1.33 pp ($[-4.7, 2.0]$); MISMATCHED minus NO ADVICE is -2.00 pp ($[-6.7, 2.7]$). In the 108-task nonempty-content subset, MATCHED minus NO ADVICE is $+0.93$ pp ($[-8.3, 10.2]$).

The 233 eligible matching-comparison tasks split into 119 first-half and 114 second-half tasks. MATCHED has 11/14 and 28/3 corrections/ regressions; MISMATCHED has 1/6 and 3/5. Thus the respective matching contrasts are $2/119 = 1.68$ pp and $27/114 = 23.68$ pp. Their exploratory difference is 22.0 pp ($[9.9, 34.2]$). On all 150 tasks per half with fallback, the contrasts are instead 1.33 and 18.00 pp. The halves have similar coarse difficulty-label counts (23/104/23 and 25/102/23). This does not establish the absence of difficulty effects within labels; the driver of the subset difference remains unidentified.

Populations for advice-factor analysis. The corrected B eligible set has 233 tasks (39 corrections, 17 regressions). Its 209 nonempty-claim tasks have 39/16, and the remaining 24 have 0/1. The advisor-source comparison instead has 232 tasks, with 39/17 for A advice and 10/20 for C advice.

A separate double-nonempty feature domain must not be substituted for this source-comparison population.

For a subset with SELF failure fraction q, conditional correction rate h, and regression rate r among SELF successes, the net gain is $qh - (1 - q)r$. This is an outcome accounting identity, not an available decision rule: SELF hidden-test outcomes are unavailable at deployment. In the 209-task content subset, the public-all-wrong group has 55 SELF failures (28 corrections) and 17 successes (two regressions); the public-all-pass group has 30 failures (ten corrections) and 103 successes (14 regressions). Four other tasks contribute one correction. These counts do not imply that public-all-pass drafts have no improvement opportunity, or identify defect visibility as the cause of advice value.

A's gains span draft forms. On A's 150-task review subset, 43 no-code drafts contribute ten net additional successes and 107 code-present drafts contribute eleven, closing to $21/150 = 14.0$ pp. Code presence alone does not certify that the draft is incorrect. On this same subset, mean output tokens are 5,517 for SELF and 4,188 for NO ADVICE, with reported cap-hit rates 73% and 31%. These are behavioral observations, not a mediation test.

M Exploratory Feedback and Repair Interventions

These follow-ups reuse the pilot and are exploratory despite design documents fixed before execution. They use A as recipient and run seeds 11, 23, and 47. The feedback study (FBX) partitions diagnostic and control examples from the originally hidden pool, reserving the remainder for evaluation (at least three tests, usually six). All compared archived and new outputs are rescored on that same remainder. These rates are not directly comparable with the main capped-ten rates. The reported construction checks unique 80-character input prefixes; this does not establish the absence of semantic near-duplicates.

FBX estimates. Let S/M denote base-feedback SELF/matched advice, and E/EM the enhanced-feedback versions. S and E use the SELF framework; M and EM use PEER. The contrast $I = (M - S) - (EM - E)$ therefore concerns the combined framework-and-advice intervention, not isolated advice content. The reported eligible cohorts are 266, 273, and 259 tasks.

Table 11: Exploratory FBX paired contrasts in pp (95% task-bootstrap CI). Rows use seed-specific eligible populations and partitioned evaluation.

Seed	$M - S$	$E - S$	I
11	$14.3\ [9.8, 18.8]$	$-3.8\ [-9.8, 1.9]$	$-0.8\ [-7.5, 6.4]$
23	$7.7\ [4.0, 11.7]$	$-12.8\ [-18.7, -7.3]$	$-14.3\ [-21.3, -7.7]$
47	$12.0\ [7.7, 16.6]$	$-12.4\ [-17.8, -7.0]$	$-10.0\ [-15.8, -3.9]$

The protocol targets two failed-draft diagnostic tests and two length-matched passing controls. The full pilot also contains no-code and passing-only drafts; actual diagnostic availability differs by draft state. We therefore do not interpret the all-cohort contrast as the effect of providing two informative failures to every task. For the narrower failing/mixed candidates, reported interactions are $+1.4$, -15.1, and $+2.6$ pp (73, 73, and 78 tasks). This also precludes a claim of a uniformly negative interaction across strata.

Budget follow-up. On 208 draft-present seed-23 tasks, the effective enhanced-feedback SELF cap rises from 6,144 to 8,192, not 12,288: a model configuration limit clips the larger requested value. Success increases by 11.5 pp ($[4.3, 18.8]$, $p = 0.003$); its difference from base SELF is then -3.9 pp ($[-9.1, 1.4]$). This comparison neither proves equivalence nor identifies output truncation as the sole mediator. Passing-example controls also recover with more budget. A complete interaction with framework and budget held fixed remains untested.

DRX construction and scope. An oracle-assisted writer uses verified repair code and its diff to construct diagnosis fields and proposed changes. D blanks the latter; R retains them. Mechanical

Table 12: **Repair guidance can add value within the review interface.** DRX with base feedback: reported paired differences in pp, 95% bootstrap intervals, and discordant counts favoring R/D. Seed populations differ and overlap; rows are reported separately, without a pooled significance test.

Seed (n)	Repair − diagnosis	R/D wins	Diagnosis − frame
11 (57)	$+14.0$ $[5.3, 24.6]$	8/0	-3.5
23 (57)	$+12.3$ $[5.3, 21.1]$	7/0	0.0
47 (63)	$+6.4$ $[-3.2, 15.9]$	7/3	$+6.4$

checks require diagnostic-evidence references and diff-token matches and limit the longest shared substring with the reference program to fewer than 200 characters. These checks reduce direct copying but do not prove semantic validity of every claim. Reference repairs mostly come from archived continuations; the selected population therefore favors tasks with previously observed successful repairs. Six conditions cross base/enhanced feedback with empty review, D, and R. Only seed-specific contrasts are used for inference here. R minus D under enhanced feedback is $+1.8$, $+10.5$, and $+3.2$ pp; the seed-23 interval is $[1.8, 19.3]$. The base-feedback estimates are in Figure 4. Overlapping tasks rule out treating pooled seed observations as independent McNemar pairs.

All of these interventions are evidence about implemented procedures. They do not establish that diagnosis is useless, that relevant information is inaccessible through self-revision, or that advice works through an identified process-stabilization mechanism.

N Evidence Provenance and Reproduction Materials

Corrected outcome counts, eligible-set sizes, and paired content-test summaries come from the final experiment report and subsequent delivery audit. The aggregate table generator checks count closure and that full-task success equals the SELF count plus corrections minus regressions. Such consistency checks complement rather than replace verification of the raw experiments.

Required replay materials are the archived task checkpoint, raw advisor output, rendered recipient prompt, continuation, hidden-test outcome, and effective request budget. The server workflow identifies `run_counterfactual.py`, `analyze_counterfactual.py`, `audit_counterfactual_validity.py`, and `verify_paper_numbers.py` as the corresponding runner and audit scripts. The schema consists of target, claim, proposed change, evidence, and scope. The case descriptions in Appendix H use the corrected-delivery verification summaries; they do not reconstruct missing code or prompt templates from aggregate outcomes.

The earlier $+8.0$ pp matched-minus-legacy recipient contrast, computed with independent resampling, is not used. The current $+15.2$ pp interaction is matched minus mismatched on 211 shared tasks, using joint task-level resampling. It is a different estimand, not merely a revised confidence interval for the earlier contrast. The reported aggregate results correspond to the final relevance audit (server revision `88ac8fb`), with the additional no-advisor and source controls from the mechanism audit (`47a5c54`). The later B review and factor-analysis summaries use the corrected analysis-set manifest (server revision `29c3f47`). The FBX/DRX follow-up uses the September 26 experiment handoff and its per-seed estimates. The local manuscript build does not rerun server experiments or reconstruct unprovided task records. Raw task records are needed to reproduce resampling, packet assignment, and paired tests.