SYMBOLIC MUSIC REASONING NEEDS CHECKERS THAT SURVIVE REWORDING: WHAT A HIGH SCORE COVERS

Anonymous authors
Paper under double-blind review

ABSTRACT

On SSMR-Bench, a template-generated benchmark of questions about sheet music in ABC notation, our hand-written `music21` checker, developed on the test items, answers 70% of items at 99.8% precision. *Prove-then-route* systems for symbolic music reasoning build on this: the checker answers the questions it can derive and a language model answers the rest. However, a high score cannot tell whether the checker computes music theory or merely recognizes the benchmark's question templates. Our key idea is that a task-preserving rewording separates the two: a checker that computes should keep its contribution, while one that recognizes wording should lose it as the model's own accuracy stays put. Our audit therefore measures the checker's *matched contribution*, the system's accuracy minus the same model's alone, before and after such rewording; a normalizer that maps reworded questions back onto the known templates tests whether a loss lies in how the checker reads the question. A checker that Claude Opus 4.8 wrote with execution feedback answers 93.4% of questions at 99.67% precision, yet rephrasing only the question line removes most of its contribution (from $+21$ to $+3$–4 points over a Qwen2.5-14B model distilled from DeepSeek-R1 traces) while the model's own accuracy barely moves. Most of that gain sits on outputs the distilled model truncates, yet on the outputs it finishes the smaller gain still falls (both post hoc). Paraphrases by the checker's own writer model also cut its coverage, and mapping them back onto the templates restores every checker answer: under rewording, what breaks is the checker's reading of the question, not its music computation. In logic the failure is worse: on ProofWriter, equivalent rule rewrites make a checker built by the same loop answer wrongly on most items through a silent default "Unknown" (post hoc), pulling the system below the model alone; the published Logic-LM design, run with a Qwen translator, makes the same errors, though its precision holds under rewording. Primary tests were pre-registered with frozen programs, several after the results that motivated them. Where a checker settles part of a task, a prove-then-route result should therefore report its coverage, precision and matched contribution under rewording.

1 INTRODUCTION

Many reasoning benchmarks are generated from a handful of question templates. A short program written against them can answer most items, and a system that runs one before a language model can post a high score. However, it is unclear what such a score measures. Figure 1 shows the concern on SSMR-Bench (Wang et al., 2025a), a generated music-theory benchmark. SYN, a checker program that a frontier model wrote and refined with execution feedback, answers every interval-naming question in its original wording. It answers none after a paraphrase asking for the same interval between the same two notes, and mapping it back onto the template restores every answer.

We study systems in which a program, the *checker*, answers first and a language model, the *policy*, answers whenever the checker abstains (Gao et al., 2023; Pan et al., 2023). Their accuracy mixes what the checker computes through one input interface with what the policy answers when that interface fails. We separate them with the checker's *matched contribution*, the system's accuracy minus the same policy's accuracy on the same items. We freeze the programs, reword the inputs

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.

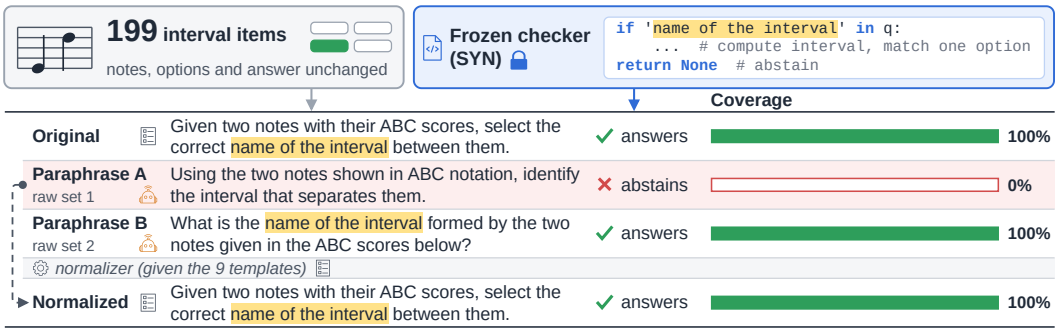


Figure 1: **The same interval question in different words.** All 199 interval-naming items keep their notes, options and answers. Paraphrases A and B are unedited, by SYN's writer model. The frozen checker answers only lines that contain its trigger (found by reading the code†); mapping a line back onto the template restores every answer.

without changing the task, and measure the checker's coverage, precision and matched contribution before and after, with the same policy on both sides (Section 3).

On SSMR-Bench (Sections 4 and 5), SYN answers 93.4% of the 1,600 original items at 99.67% precision and beats our hand-written checker by 9.6 points with the same fallback. Yet two rephrasings of the question line cut its coverage to about 13%. They also remove most of its contribution over a Qwen2.5 policy fine-tuned on DeepSeek-R1 traces, from +21 to +3–4 points at 14B, while the policies' own accuracy moves by under three points. About 70% of that contribution comes from items where the policy runs out of tokens†. On items the policy finishes in both conditions, SYN adds about +7.5 points at 14B†, and rephrasing lowers this by 4–7 points.

Paraphrases by SYN's own writer model also cut its coverage. Shown the nine templates, however, the same model maps each paraphrase back onto its original line. The checkers then give exactly their original answers, and with any of three Qwen policies answering on the paraphrases the system keeps SYN's contribution. Asking the writer for wording-robust code instead keeps most of the coverage on the paraphrases but covers less of the original wording.

The failure is worse on ProofWriter (Section 6). A checker synthesized by the same loop at a smaller budget answers every item, but only 90% correctly. After logically equivalent rule rewrites, it still answers every item at 46% precision, so the system falls 21–26 points below the policy alone. Every observed error is the program's default answer "Unknown"†. Abstaining on it, a rule we proposed after this diagnosis, gave no wrong answer on 900 fresh theories, at a coverage cost. Logic-LM, a published design that translates each item into a logic program, makes the same default-Unknown errors, though its precision holds under rewording with a Qwen translator.

We make three contributions. First, we propose an audit that separates what a program adds from what the policy can do, by measuring matched contribution under task-preserving rewording. Second, we provide evidence from two generated benchmarks that this contribution can be bound to wording, as in music, or turn negative through silent wrong answers, as in logic. We also report two benchmark-level findings. The options alone determine all 352 key-identification items (22% of SSMR-Bench†), and repaired distractors make these items harder for all four Qwen policies. Third, we test two remedies, template normalization and abstention on Unknown, with their costs. Primary tests were pre-registered with programs and predictions fixed in advance (Appendix Z.2), and post-hoc analyses are marked †. The audit concerns these benchmarks and checkers, our own and one published design, and does not show that programs must fail on paraphrases.

2 Related Work

Benchmark audits. Partial-input baselines (Gururangan et al., 2018; Poliak et al., 2018; Kaushik & Lipton, 2018), syntactic heuristics (McCoy et al., 2019), shortcut learning (Geirhos et al., 2020), invariance tests (Ribeiro et al., 2020) and contrast sets (Gardner et al., 2020) motivate checking what a benchmark score measures (Raji et al., 2021; Bean et al., 2025). However, a failed partial-

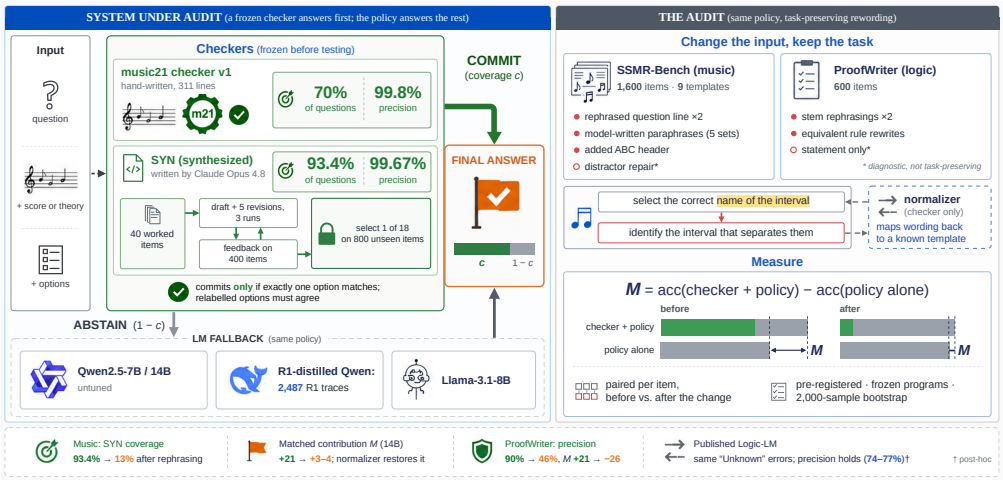


Figure 2: **Overview.** Left: the system under audit. A frozen checker, the hand-written v1 or the synthesized SYN, answers when exactly one option matches, and the same policy answers its abstentions. Right: the audit. On both benchmarks, we change the input but not the task and compare the matched contribution M before and after. An optional normalizer feeds only the checker. Bottom: main results. Bars are schematic, and † marks post-hoc results.

input baseline cannot certify that artifacts are absent (Feng et al., 2019). Policy accuracy can also shift under meaning-preserving prompt changes (Sclar et al., 2024) and template re-instantiation (Mirzadeh et al., 2025). ProofWriter and RuleTaker render formal theories in templated English (Tafjord et al., 2021; Clark et al., 2020), and RuleTaker's crowdsourced paraphrases and RobustLR's logical-equivalence rewrites (Sanyal et al., 2022) test trained reasoners beyond those forms. We instead measure M (Section 3), a checker's contribution over a matched policy, on the same original and rewritten items. Selective prediction trades coverage for accuracy within one model (Geifman & El-Yaniv, 2017), whereas M compares two systems sharing a fallback.

Programs and tools in model systems. PAL and Program-of-Thoughts execute model-written code (Gao et al., 2023; Chen et al., 2023b), and ReAct, Toolformer and modular agents offload computation (Yao et al., 2023; Schick et al., 2023; Karpas et al., 2022; Wu et al., 2023). Logic-LM, LINC and SatLM translate problems for symbolic solvers (Pan et al., 2023; Olausson et al., 2023; Ye et al., 2023). Logic-LM falls back to chain-of-thought when execution fails, and we audit it in Section 6. LATM and CRAFT separate a capable tool writer from a cheaper tool user, and tool-making work refines reusable programs with execution feedback (Cai et al., 2024; Yuan et al., 2024a; Qian et al., 2023; Wang et al., 2024a; Silver et al., 2024). SYN uses this design. We audit what such a program adds to its fallback, including parsing and commit-decision failures. As in work on external verification (Kambhampati et al., 2024; Stechly et al., 2025), precision is not soundness.

3 Measuring What a Checker Adds

A *checker* receives the question, its evidence (a musical score in ABC text notation, or a ProofWriter theory of facts and rules) and the options, and returns an option or abstains. It never sees the gold answer. Its *coverage* c is the fraction of items it answers, and its *precision* p is its accuracy on those items. In a checker-plus-policy system, the checker's answer is final when it gives one, and the policy answers the rest (Figure 2), so

$$\mathrm{acc}(\text{checker+policy}) = c\,p + (1-c)\,a_{\mathrm{abst}},$$

where a_{abst} is the policy's accuracy on the checker's abstentions. The checker's *matched contribution* is

$$M = \mathrm{acc}(\text{checker+policy}) - \mathrm{acc}(\text{same policy alone}),$$

computed item by item on the same policy outputs. The two systems agree wherever the checker abstains, so only its answers move M: a correct answer gains a point where the policy was wrong,

and a wrong answer loses one where the policy was right. M therefore cannot exceed coverage and can be negative. A checker with high coverage can lift a weak policy, add little to a strong one, or sink a system below its policy.

The audit. We freeze each program, change the inputs without changing the task (Figure 2, right), and recompute c, p and M with the same policy on both sides. If the checker computes the task, rewording should leave M roughly unchanged. If it recognizes the generator's wording, M falls while the policy's own accuracy barely moves. Each task-preserving change should keep the score or theory, the options and the gold answer. One paraphrase did not, and we report it. Our rephrasings, the ProofWriter stems and the rule rewrites reach both checker and policy, whereas ABC header changes (index and title lines) reach only the checkers. The model-written paraphrases reach the checkers and, in a later end-to-end test, three Qwen policies. The two diagnostic changes do not preserve the input: repaired key distractors replace the wrong options, and partial-input probes remove the evidence (options alone for music, statements alone for logic). A normalizer maps a reworded question back onto a known template line before the checker sees it. If this restores M, the loss lies in the checker's input interface rather than in its solving code.

Policies and statistics. The policies are Qwen2.5-7B- and 14B-Instruct (Qwen Team, 2024), untuned (not trained by us) or R1-distilled. The latter are LoRA fine-tuned on 2,487 correct DeepSeek-R1 traces for SSMR-Bench training questions (Appendix B.2). Untuned Llama-3.1-8B serves as a second model family. In the corrected tests, policies decode greedily, must state an explicit answer line, and have budgets of 8,192 output tokens for music and 2,048 for ProofWriter (one registered re-run samples with 16,384, Section 5). Each size runs original and changed conditions together on one inference stack. Intervals are 95% paired item-bootstrap intervals (2,000 resamples, seed 0) that keep each item's checker and policy outcomes together. These intervals treat each wording as fixed. Since coverage changes by whole templates, we also bootstrap over templates for the main drops†.

Pre-registration. Each primary test was registered before the first output it constrains, with its programs, predictions and decision rules fixed at registration. Appendix Z.2 lists every registration, its outcome and our code names. Failed predictions are reported like held ones, and post-hoc analyses are marked †. However, some registrations followed related results. The music policy test (Section 5) re-tests earlier runs whose direction was known. The truncation re-runs, the robustness-aware writer, the end-to-end paraphrase test, the fresh-theory test and the Logic-LM test were each registered after the result that motivated them. The music audit as a whole was not registered in advance. The music policy test counts a policy as stable when its accuracy change is within five points by point estimate, while the ProofWriter test requires the whole interval to lie within five points.

4 Short Programs Cover the Original Music Benchmark

SSMR-Bench contains 1,600 multiple-choice questions on scores in ABC notation, generated from nine templates over interval, chord, scale and beat tasks (Wang et al., 2025a). We first ask how much of it a program can answer before any policy is involved. Figure 3 orders the checkers we compare, each of which hands its abstentions to the same untuned Qwen2.5-14B greedy policy.

Hand-written checkers. v1 is a 311-line `music21` (Cuthbert & Ariza, 2010) program with a 211-line helper. It picks a routine from the question text and answers when the result matches exactly one option. On chord naming it only hints to the policy, and on scale selection it always abstains. We developed v1 on the public test items and their labels. On 8,000 training draws it never saw during development, v1 covers 70.25% at 99.86% precision†, though these draws come from the same generator with a different template mix (Appendix B.4). v2† completes v1's key-identification and bar-line matchers for the original distractors (Appendix U). It serves as a diagnostic baseline, whereas the registered synthesis tests compare against the frozen v1.

Synthesized checkers. SYN asks Claude Opus 4.8 to write one reusable `solve(question, abc, options)` program through the loop of Figure 2 (left). The writer sees 40 worked training items and gets execution feedback on a 400-item training split that contains them. We run the loop three times, each with one draft and five revisions. A fixed rule then picks one of the 18 programs on

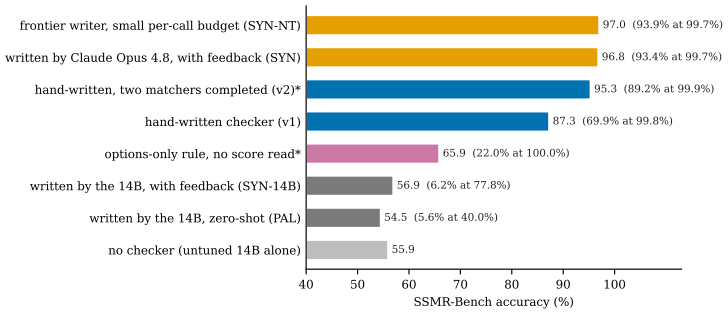


Figure 3: **Short programs cover most original SSMR-Bench items.** Each checker hands its abstentions to the same untuned 14B greedy policy; parentheses give coverage and precision. Against v1, SYN and SYN-NT give the registered outcome C (paired interval entirely above $+3$ points), and PAL and SYN-14B outcome A (entirely below -3). SYN-NT matches SYN-14B's per-call budget, not its total output. *Post-hoc rows: v2 completes two of v1's matchers, and the options-only rule answers key-identification items. Fallback outputs here use the legacy lenient parser (Appendix B.2); the corrected rewording tests use strict extraction (Section 5).

800 items the writer never saw and freezes it before testing. Three controls vary the writer. PAL asks the untuned 14B policy for one zero-shot program per test item. SYN-14B runs the full loop with the untuned 14B as writer, and SYN-NT runs it with Opus limited to SYN-14B's per-call budget (no extended thinking, 8,192 output tokens). A generated program runs in a sandbox and answers only if two runs with cyclically relabelled options agree, which rules out hard-coded letters but not shortcuts in option texts. Each synthesis test compares the system's accuracy with v1's under the same fallback. We registered four outcomes per test (Appendix X): A if the paired interval lies below -3 points, B if inside ± 3, C if above $+3$, and D if it straddles a boundary.

Results. SYN answers 1,494 items (93.4%) at 99.67% precision, with 5 wrong. The policy gets 60 of the remaining 106 right, and the system scores 96.8%, $+9.6$ $[+8.1, +11.2]$ points above v1. This is outcome C: *A frontier model with feedback writes a better checker than ours.* The gain stays above $+3$ points under two registered checks. The first targets answers that ignore the score. It hands to the policy every SYN or v1 answer that stays the same when the item's score is swapped for another item's, and gives $+8.94$ $[+7.50, +10.38]$. The second keeps only templates with at least 50 training draws and gives $+10.36$ $[+8.65, +12.06]$. With three substitute scores whose gold answer differs, 12 of SYN's 1,184 answers outside the note- and chord-completion templates never change†, so SYN reads the score for most of its answers but not all. Much of the gain over v1 is completeness: v2 covers 89.2% at 99.86% precision and scores 95.3%, and SYN beats it by only $+1.5$ $[+0.8, +2.3]$ points, in a registered comparison against a post-hoc checker. Option information also helps: the options alone determine all 352 key-identification items, 22% of the test set† (Appendix U.1).

The writer matters. Three registered controls vary it (wording in Appendix X). PAL, one zero-shot program per item from the untuned 14B, returns outcome A at -32.8 $[-35.1, -30.6]$. This measures zero-shot code writing, since 1,193 of its 1,600 programs raised an execution error, and the comparison favours v1, which was built on these test items. SYN-14B, the full loop with the untuned 14B as writer, also returns A, at -30.3 $[-32.7, -28.1]$. In contrast, SYN-NT, Opus at SYN-14B's per-call output budget (no extended thinking, 8,192 tokens), returns C at $+9.75$ $[+8.25, +11.31]$, so the frontier writer's advantage survives removing extended thinking. Total output is not matched, however (129,096 output tokens against SYN-14B's 0.017M). SYN-NT's first test pass was interrupted and re-run unchanged before any outcome existed, a disclosed deviation.

5 On Music, the Checker's Contribution Is Bound to Wording

5.1 Rephrasing Removes Most of SYN's Coverage

After SYN's test pass, an LLM coding agent (Claude) drafted two rephrasings of each question line under our direction, without a blindness protocol. Before scoring the unmodified v1, v2 and

Table 1: **Rephrasing removes most of SYN's contribution.** Top: checker coverage / precision (%); v2 is a post-hoc completion of v1. Bottom: the corrected policy test, 14B / 7B, with strict answer extraction and 8,192 tokens. M is SYN plus the R1-distilled policy minus the same policy alone. Every drop in M exceeds the registered ten points; on items the policy finishes in both conditions, SYN's original contribution is only $+7.6/+7.4$ at 14B and $+14.3/+13.2$ at 7B†. Details: Appendices Y and Y.5.

Condition	v1	v2	SYN
Original	69.9/99.82	89.2/99.86	93.4/99.67
Rephrasing 1	35.4/99.65	35.4/99.65	12.9/98.07
Rephrasing 2	15.8/100.00	23.9/100.00	12.5/100.00
Condition	**R1-distilled**	**SYN + R1-distilled**	**M (pp)**
Original	75.81/58.88	96.94/95.56	$+21.12/+36.69$
Rephrasing 1	75.56/58.69	79.75/63.69	$+4.19/+5.00$
Rephrasing 2	75.81/59.31	79.00/65.62	$+3.19/+6.31$

SYN, we registered predictions for the rephrasings, for an added ABC header and for repaired key distractors. Rephrasing cuts SYN's coverage from 93.4% to 12.9% and 12.5% (Table 1). We predicted that each rephrasing would at least halve the coverage of v1 and SYN while their precision stayed at 99% or more. This held in 2 of 4 cases: under rephrasing 1, v1 keeps slightly more than half of its coverage and SYN answers 4 of its 207 items wrongly. The added header changes nothing for the three checkers, within the predicted two points, so robustness to a header does not imply robustness to wording. We predicted the coverage loss because both checkers choose a routine from the question's wording. Reading SYN's unmodified code afterwards† finds the trigger: its interval routine fires only on the phrase *name of the interval*, which neither rephrasing contains (Figure 1).

5.2 With the Policy Held Fixed, the Contribution Falls

We next ask whether the checker's contribution falls with the policy held fixed. The test measures M for SYN plus the R1-distilled policy against that same policy alone, with untuned policies as controls. It extracts only an explicit `Answer: X` line, allows 8,192 tokens, runs one inference stack per size, and draws repaired distractors from the generator's own 24 gold spellings. These choices fix four problems we found afterwards in our first runs (Appendix Y.4): a lenient parser, a 2,560-token cap, an unmatched fallback comparison, and a gold-only spelling cue in 27 of 352 repaired key items. The primary prediction, that M drops by at least ten points under each rephrasing, is a confirmatory re-test, registered after the earlier runs had shown the direction. In those runs, with the legacy parser, SYN plus the distilled 14B scored 79.25/78.75% under rephrasings 1 and 2, against the policy's 75.50/75.44% (Appendix Y.5).

The prediction holds for both sizes under both rephrasings (Table 1). At 14B, M falls from $+21.12$ to $+4.19$ and $+3.19$, and at 7B, from $+36.69$ to $+5.00$ and $+6.31$. The drops are $+16.94$ $[+15.00, +19.06]$ and $+17.94$ $[+16.00, +20.00]$ at 14B, and $+31.69$ $[+29.12, +34.19]$ and $+30.38$ $[+27.94, +32.88]$ at 7B. Resampling the nine templates instead of items† widens these intervals to $[+8.6, +26.7]$ and $[+8.6, +28.8]$ at 14B and $[+17.7, +46.6]$ and $[+19.0, +43.4]$ at 7B, so at 14B the lower bounds fall below ten points. The policies' own accuracy, however, barely moves. All eight changes (two sizes, distilled and untuned, two rephrasings) lie within the registered five-point band by point estimate. The largest, $+2.81$ for the untuned 7B under rephrasing 2, has an interval of $[+0.12, +5.50]$ that excludes zero. A second model family shows the same pattern: untuned Llama-3.1-8B, a weak policy at 31.4%, changes by -0.25 and -0.81, while SYN's contribution over it falls by 57.12 and 55.94 points. On ProofWriter, the same Llama policy fails the stricter whole-interval rule in all three conditions, though every point estimate lies inside the band (Appendix Y.5).

Part of the drop sits on truncated outputs. The distilled 14B and 7B policies hit the token cap on 17.1% and 28.6% of original items (a registered report), and these items carry about 70% of SYN's original gains†. On items the policy finishes in both conditions†, one subset per rephrasing, SYN's original contribution is only $+7.58/+7.36$ at 14B, where the policy alone already scores 91.8/92.0%, and $+14.27/+13.18$ at 7B (policy 83.5/84.8%). They fall to $+0.16/+2.93$ and

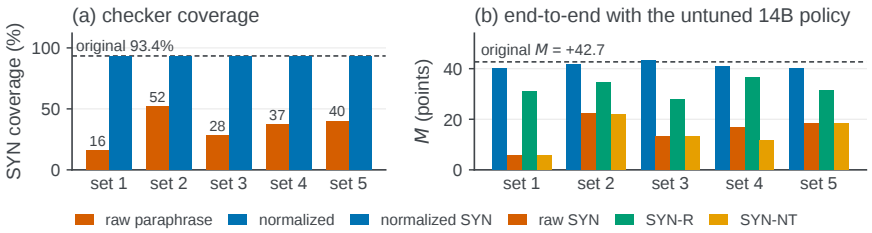


Figure 4: **Paraphrases remove SYN's coverage and contribution; normalization restores both.** (a) SYN's coverage on five paraphrase sets written by its own writer model, raw and after normalization onto the known templates (set 3's note-completion line changes the task†). (b) M with the untuned 14B policy answering on the raw paraphrases: normalized SYN stays within three points of the original M, raw SYN falls by 20.5–37.1 points, and the robustness-aware SYN-R exceeds SYN-NT, the loop at the same budget, in every set (Appendices Y.1 and Y.3).

$+1.06/+4.29$, drops whose intervals exclude zero but three of which are below ten points. Two re-runs registered after this finding, with a repetition penalty and with the distilled model's recommended sampling (16,384 tokens), still truncate 9.7–24.8% of items, above their registered 5% bound, so neither isolates truncation. On all items, both again drop by more than ten points, and each registered primary prediction holds in 4 of 4 settings. In the registered reports on items finished in both conditions, the 14B drops stay at 4.1–6.9 points in both re-runs, while the 7B drops reach 10.1–16.3 (Appendix Y.5).

5.3 Paraphrases We Did Not Write, and Restoring the Interface

Since our rephrasings could be unusual, we also use paraphrases from Claude Opus 4.8, the model that wrote SYN. It saw only the nine template lines with parameter placeholders and wrote five paraphrases of each, one per set, which we used unedited and unfiltered. For normalization, the same model sees the nine templates and one reworded line, with no score, options or answer, and maps the line back onto a template. We committed the paraphrases before normalizing them, and the normalized lines before scoring.

As shown in Figure 4a, SYN's coverage falls to 15.6/52.0/28.1/36.9/40.1% on the five raw sets, and the halving prediction holds in 4 of 5 sets, with set 2 as the exception. Per template, coverage is all-or-nothing: a paraphrase either keeps the template's trigger phrase and all of its coverage or loses both. Key identification loses all coverage in every set. Normalization returns the original line on all 1,600 items in every set, so v1, v2 and SYN necessarily give exactly their original answers. The registered restoration predictions hold in 5 of 5 sets, but they follow from this. One paraphrase changed its task†. Set 3's note-completion line adds "above the note", yet 78 of its 201 gold answers lie below. No checker answers these items in raw set 3, and set 3 meets both predictions without this template. By our reading, the other 44 paraphrases add no constraint, and our two rephrasings keep every gold answer correct, with a second correct option on 2 of 48 scale items (Appendix Y.1). The writer model, shown the nine templates, can therefore map its own paraphrases back onto them. This does not show that an unknown interface can be discovered.

In a later registered test, the policies answer on the raw paraphrases (Figure 4b). The untuned 14B's accuracy moves by at most three points from its original 54.0% by point estimate, and on set 1 it changes by $+3.00$ $[+0.50, +5.62]$. With normalization, SYN's contribution over this policy stays within three points of its original $+42.7$ in all five sets, inside the registered five-point bound. Without it, the contribution falls by 20.5–37.1 points. The untuned 7B gives the same verdicts. With the R1-distilled 14B, a registered secondary policy, normalization again keeps SYN's contribution within three points of that run's original $+20.6$. Without it, the contribution falls by 5.4–15.8 points, or 26–77%, and by ten or more in only 2 of 5 sets (Appendix Y.3).

A robustness-aware writer keeps most of its coverage, at a cost on the original wording. SYN-R, a variant registered after the paraphrase results, asks the writer for wording-robust code and adds our two rephrasings to its worked, feedback and selection items. On the five paraphrase sets, which its writer never saw but the same model wrote, SYN-R covers 25.2–59.6 points more than SYN-NT,

Table 2: **ProofWriter: the checker abstains under stem rephrasings but answers wrongly under rule rewrites.** 600 items; untuned Qwen policies, strict extraction. Coverage / precision and accuracy in %; M in points (14B / 7B, not pooled); a dash means the checker answered nothing.

Condition	Checker cov. / prec.	Policy 14B / 7B	M, 14B / 7B
Original	100/90.00	68.83/62.33	$+21.17/+27.67$
Stem 1	0/—	67.83/61.50	0.00/0.00
Stem 2	0/—	68.17/63.67	0.00/0.00
Rule rewrites	100/46.33	72.17/67.50	$-25.83/-21.17$

the loop at the same budget, at 99.49–99.63% precision, largely by dispatching on keywords. On the original wording, however, it covers 85.0%, exactly at the registered floor, against SYN-NT's 93.9%†. We predicted that it would still lose at least ten points of its own coverage on every set. This held in 3 of 5 sets, since it lost 8.5 points on set 2 and none on set 4 (Appendix Y.2). End-to-end, SYN-R's contribution exceeds SYN-NT's in all five sets with both untuned policies. With the untuned 14B, it leads by $+12.8$ to $+25.4$ points and reaches 65–86% of SYN's original $+42.7$. With the distilled 14B, SYN-R leads in 4 of 5 sets, and in set 2 its interval includes zero.

Repaired distractors expose a different dependence. We also replaced the key-identification distractors, which the options alone give away. As predicted, the options-only rule falls below 50% key precision, to 20.50%. However, the checkers do not keep 99% precision within five coverage points: v1 falls to 88.38% and SYN to 93.47%, and v2 loses 15.1 coverage points. The policy test's corrected repair, which removes a spelling cue on 27 items, gives the same picture (v1 88.83%, SYN 93.75%, options-only 25.61%). A rule that counts notes answers all 352 repaired items†, so the checkers' key rules are simply weak (Appendix Y). Under the repair, distillation still improves key identification, and the distilled model's key accuracy falls at both sizes, the registered outcome C: *The distilled model's key-identification accuracy partly depends on how the generator builds distractors.* The untuned policies' key accuracy falls as much or more, by $+12.50$ against $+6.53$ points at 14B and by $+11.08$ for both at 7B. The repair thus makes these items harder for every Qwen policy, while Llama-3.1-8B, near chance on them, barely moves. Appendix Y.5 also reports a registered system ranking, which we do not claim.

6 On ProofWriter, Coverage Can Hide Wrong Answers

We repeat the synthesis, partial-input and rewording tests on the 600-item depth-5, open-world ProofWriter sample of Pan et al. (2023), balanced over True, False and Unknown. The loop runs at SYN-NT's budget, with feedback and selection on training splits disjoint from the test theories, and every condition was registered before the first writer call. Two stem rephrasings and logically equivalent rule rewrites keep the gold labels, which a reference reasoner, used only for validation, confirmed on all 69,810 training items before registration. A statement-only probe removes the theory (Appendix Z).

The checker answers everything, often wrongly. The selected program answers every original item, but only 90.00% correctly (Table 2), so our prediction of 99% precision fails. Under the stem rephrasings, it abstains on everything, as predicted, and its contribution becomes zero. Under the rule rewrites, against our predicted coverage drop, it still answers every item, now at 46.33% precision, so its wrong answers override the policy's. Its contribution of $+21.17/+27.67$ points over the 14B/7B policy becomes $-25.83/-21.17$, and the system is worse than the policy alone. The predicted drop of at least ten points holds in all 6 comparisons, with every lower bound at least 17.00. The stricter whole-interval stability rule holds for 3 of 6 policy comparisons, and the other three fail through gains or intervals above $+5$, never losses. Under rule rewrites, policy accuracy changes by $+3.33$ $[-0.17, +7.00]$ and $+5.17$ $[+1.00, +9.33]$, so part of the negative contribution may come from an easier policy condition. Without the theory, the policies score 30.17/29.50%, near the 33.3% chance level, as predicted.

The errors are default answers. Post-hoc checks locate the failures. Restoring the original stem restores all 600 original answers, so the rephrased stems break statement extraction. All 60 original

and 322 rule-rewrite errors are the program's default "Unknown" on True or False items. Since Unknown is a valid open-world label that survives relabelling, these answers pass the commit rule of Section 4. Our reference reasoner, which also validated the rewrites, answers all 600 items correctly under both rule forms. Selection let these errors through partly because the registered score $J = \text{correct} - 1.27 \times \text{wrong}$ has no precision constraint. The chosen program had 89.48% precision on the selection split. A candidate chosen under a 99% precision constraint covers 19.0% of original and 9.5% of rewritten items with no wrong answer†.

Abstaining on Unknown. A simpler rule, proposed after this diagnosis, keeps the registered program but abstains whenever it answers Unknown. On the test items, it covers 56.67% of original and 13.0% of rewritten items without error, but this holds by construction†. We therefore locked the rule and scored it once, checker only, on 900 fresh theories from ProofWriter's test split that no earlier experiment had used. All four registered predictions hold, namely precision of at least 99% by point estimate and coverage within the registered ranges. Every answer is correct ($507/507$ original and $150/150$ rewritten), at 56.33% and 16.67% coverage. The one-sided 95% lower bounds on precision are 99.41% and 98.02%, so the second falls below 99%. The rule gives no soundness guarantee, since a constructed negated-premise item still gets a wrong True. It also does not undo the failed precision prediction.

A published design shares the silent errors but not the wording collapse. Logic-LM (Pan et al., 2023) translates each item into a logic program and falls back to its model when the program fails. Its released GPT-4 programs, run with its solver, answer 99.0% of original items at 79.6% precision, and all 121 wrong answers are the default Unknown on True or False items. Abstaining on Unknown leaves 45.7% coverage with no wrong answer and $+10.5$ points over the 14B policy, a result seen before registration. In a registered test with the untuned 14B as translator, Logic-LM's parser rejected most programs because Qwen wraps them in Markdown†, so Logic-LM answered only 6–21% of items and added at most 2.2 points in any condition. Coverage stayed within ten points in 2 of 3 changed conditions, and abstaining on Unknown reached 96.3% precision on original items, short of the registered 99%. The untuned 7B does not write the required format. With the Markdown stripped†, Logic-LM answers 57–76% of items at 74–77% precision in every condition. In this setting, its contribution under rule rewrites falls by 4.5 $[+1.0, +8.0]$ points, to $+0.2$, against 47 for the synthesized checker, and 93–96% of its wrong answers are again default Unknowns (Appendix Z.1). Logic-LM thus keeps its precision under rewording but makes the same errors, and its weak point is its translator's output format.

7 Conclusion and Limits

Short programs answer most original SSMR-Bench items, but what they add over a fixed policy depends on the question's wording. On ProofWriter, rule rewrites turn a synthesized checker's silent default answers against its policy. The published Logic-LM makes the same errors yet keeps its precision under rewording. A program-plus-model score should therefore come with the program's coverage, precision and contribution over the exact policy on its abstentions, before and after task-preserving rewording, with paired intervals. Both tested remedies have a cost. Our normalizer was given the templates and is unvalidated on unseen wording. Abstaining on Unknown makes no error on the selection split it was not derived from† (Appendix Z) or on fresh theories in a registered test of the checker only, but gives up most coverage under rule rewrites.

Limits. The audit covers two generated benchmarks and a fixed set of rewordings. It tests our own checkers and one published design, with two sizes of one policy family and a weak second family. The paraphrases come from SYN's own writer, and the distilled policy ran on them only at 14B. We developed v1 on test items, and the frontier writer saw training items and may have met the public generator in pretraining. Distilled policies truncate 10–29% of music outputs in all four corrected runs. The drops shrink on items finished in both conditions, and distilled scores are stack-sensitive† (Appendix B.4). Table 4's historical Qwen results (Appendix B.2) use the legacy lenient parser and keep no raw completions. Other appendices keep earlier studies supporting no claim here.

REPRODUCIBILITY STATEMENT

The supplementary material contains the anonymized audit artifacts: each experiment's pre-registration, committed before the outputs it constrains, the frozen checkers, the evaluation harnesses and analysis scripts, and the per-item outputs, with `bench/REPRODUCE.md` mapping each reported number to its script and inputs. Appendix B.1 lists these artifacts experiment by experiment and states which raw files are omitted for size or remain on the run host; Appendix B.2 gives the training recipe, Appendix B.4 the decoding and inference settings, and Appendix Z.2 every registration with its outcome.

AI USE STATEMENT

In this work, we used generative AI tools to generate synthetic data sets (the model-written paraphrases, normalizer outputs and rephrasings, and the distillation traces used as experimental conditions), to help develop the conceptual framework and formulate its elementary accounting identities (Section 3), to propose and refine hypotheses and draft the pre-registrations, to design and provide feedback on the methodology and experiments, to implement methods (evaluation harnesses, checkers and analysis code), to clean and reformat datasets, to support the qualitative analysis of checker failures, and to interpret results. We have not used generative AI tools for translation, and providing critical ingredients for proving mathematical claims or assisting in the writing of proofs are not applicable to this work, which contains no proofs. Additionally, we used generative AI tools to create and edit software code and scientific figures, to draft and edit large parts of the paper, including complete co-writing rounds, to identify and summarize related literature, to check and format references, to suggest the structure, title and keywords, and for brainstorming. Several generative models are also objects of study: Claude Opus 4.8 wrote the SYN, SYN-NT, SYN-R and ProofWriter checkers and ROBUST-N's paraphrases and normalizer outputs, an LLM coding agent (Claude) drafted the two ROBUST rephrasings under the authors' direction, Qwen2.5 models produced the Logic-LM translations, and the distilled policies are trained on DeepSeek-R1 reasoning traces (Appendix B.2). We have reviewed all AI-assisted work. Each pre-registration was committed before the outputs it constrains, every reported number was re-checked against the committed result files in independent verification passes, all cited references were checked against their original sources, and LLM-based review panels stress-tested the claims before submission; several caveats in the paper originate from that process. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Bret J. Aarden. *Dynamic Melodic Expectancy*. PhD thesis, The Ohio State University, 2003.

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with LLMs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Utkarsh Bahuguna. When self-consistency backfires: Majority vote hurts the majority of hard science problems for small LLMs. In *COLM 2026 Workshop on Efficient Reasoning*, 2026. arXiv:2608.11403.

Andrew M. Bean, Ryan Othniel Kearns, Angelika Romanou, Franziska Sofia Hafner, Harry Mayne, Jan Batzner, Negar Foroutan, Chris Schmitz, Karolina Korgul, Hunar Batra, Oishi Deb, Emma Beharry, Cornelius Emde, Thomas Foster, Anna Gausen, María Grandury, Simeng Han, Valentin Hofmann, Lujain Ibrahim, Hazel Kim, Hannah Rose Kirk, Fangru Lin, Gabrielle Kaili-May Liu, Lennart Luettgau, Jabez Magomere, Jonathan Rystrøm, Anna Sotnikova, Yushi Yang, Yilun Zhao, Adel Bibi, Antoine Bosselut, Ronald Clark, Arman Cohan, Jakob Foerster, Yarin Gal, Scott A. Hale, Inioluwa Deborah Raji, Christopher Summerfield, Philip H. S. Torr, Cozmin Ududec, Luc Rocher, and Adam Mahdi. Measuring what matters: Construct validity in large language model

benchmarks. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2025.

Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. In *International Conference on Learning Representations (ICLR)*, 2024.

Bei Chen, Fengji Zhang, Anh Nguyen, Daoguang Zan, Zeqi Lin, Jian-Guang Lou, and Weizhu Chen. CodeT: Code generation with generated tests. In *International Conference on Learning Representations (ICLR)*, 2023a.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024a.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research (TMLR)*, 2023b.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. In *International Conference on Learning Representations (ICLR)*, 2024b.

Hyeong Kyu Choi, Xiaojin Zhu, and Yixuan Li. Debate or vote: Which yields better decisions in multi-agent large language models? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3882–3890, 2020. doi: 10.24963/ijcai.2020/537.

Michael Scott Cuthbert and Christopher Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. In *ISMIR*, 2010.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Kaihua Ding. When LLMs agree, are they right? auditing self-consistency and cross-model agreement as confidence signals. *arXiv preprint arXiv:2607.08065*, 2026.

Shi Feng, Eric Wallace, and Jordan Boyd-Graber. Misleading failures of partial-input baselines. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 5533–5538, 2019. doi: 10.18653/v1/P19-1554.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In *International Conference on Machine Learning (ICML)*, 2023.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1307–1323, 2020. doi: 10.18653/v1/2020.findings-emnlp.117.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

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

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2018.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

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 (ICLR)*, 2024.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Paul Saldyt, and Anil B. Murthy. Position: LLMs can't plan, but can help planning in LLM-modulo frameworks. In *International Conference on Machine Learning (ICML)*, 2024.

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*, 12:1417–1440, 2024.

Ehud Karpas et al. MRKL systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning. *arXiv preprint arXiv:2205.00445*, 2022.

Divyansh Kaushik and Zachary C. Lipton. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Steven Kolawole, Don Dennis, Ameet Talwalkar, and Virginia Smith. Agreement-based cascading for efficient inference. *Transactions on Machine Learning Research*, 2025. arXiv:2407.02348.

Carol L. Krumhansl. *Cognitive Foundations of Musical Pitch*. Oxford University Press, New York, 1990.

Deepak Kumar, Emmanouil Karystinaios, Gerhard Widmer, and Markus Schedl. How far can pretrained LLMs go in symbolic music? controlled comparisons of supervised and preference-based adaptation. In *NLP4MusA 2026*, 2026. arXiv:2601.22764.

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.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023a.

Mingjie Liu, Nathaniel Pinckney, Brucek Khailany, and Haoxing Ren. VerilogEval: Evaluating large language models for verilog code generation. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2023b.

Shang Liu, Yao Lu, Wenji Fang, Mengming Li, and Zhiyao Xie. OpenLLM-RTL: Open dataset and benchmark for LLM-aided design RTL generation. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–9, 2024. doi: 10.1145/3676536.3697118.

Yao Lu, Shang Liu, Qijun Zhang, and Zhiyao Xie. RTLLM: An open-source benchmark for design RTL generation with large language model. In *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2024.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*, 33:3776–3786, 2025. doi: 10.1109/TASLPRO.2025.3606231.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. GSM-Symbolic: Understanding the limitations of mathematical reasoning in large language models. In *International Conference on Learning Representations (ICLR)*, 2025.

Gagan Mundada, Yash Vishe, Amit Namburi, Xin Xu, Zachary Novack, Julian McAuley, and Junda Wu. WildScore: Benchmarking MLLMs in-the-wild symbolic music reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 16847–16863, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.853.

Theo X. Olausson, Alex Gu, Benjamin Lipkin, Cedegao E. Zhang, Armando Solar-Lezama, Joshua B. Tenenbaum, and Roger Levy. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Theo X. Olausson, Jeevana Priya Inala, Chenglong Wang, Jianfeng Gao, and Armando Solar-Lezama. Is self-repair a silver bullet for code generation? In *International Conference on Learning Representations (ICLR)*, 2024.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

Nathaniel Pinckney, Christopher Batten, Mingjie Liu, Haoxing Ren, and Brucek Khailany. Revisiting VerilogEval: A year of improvements in large-language models for hardware code generation. *ACM Transactions on Design Automation of Electronic Systems*, 30(6):1–20, 2025. doi: 10.1145/3718088.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics (*SEM)*, 2018.

Cheng Qian, Chi Han, Yi Fung, Yujia Qin, Zhiyuan Liu, and Heng Ji. CREATOR: Tool creation for disentangling abstract and concrete reasoning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Inioluwa Deborah Raji, Emily Denton, Emily M. Bender, Alex Hanna, and Amandalynne Paullada. AI and the everything in the whole wide world benchmark. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 4902–4912, 2020. doi: 10.18653/v1/2020.acl-main.442.

Soumya Sanyal, Zeyi Liao, and Xiang Ren. RobustLR: A diagnostic benchmark for evaluating logical robustness of deductive reasoners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9614–9631, 2022. doi: 10.18653/v1/2022.emnlp-main.653.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How I learned to start worrying about prompt formatting. In *International Conference on Learning Representations (ICLR)*, 2024.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. Spurious rewards: Rethinking training signals in RLVR. *arXiv preprint arXiv:2506.10947*, 2025.

Zhihong Shao et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Tom Silver, Soham Dan, Kavitha Srinivas, Joshua B. Tenenbaum, Leslie Kaelbling, and Michael Katz. Generalized planning in PDDL domains with pretrained large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 20256–20264, 2024.

Andries P. Smit, Nathan Grinsztajn, Paul Duckworth, Thomas D. Barrett, and Arnu Pretorius. Should we be going MAD? A look at multi-agent debate strategies for LLMs. In *International Conference on Machine Learning (ICML)*, 2024.

Duncan Soiffer, Steven Kolawole, and Virginia Smith. Semantic agreement enables efficient open-ended LLM cascades. In *Proceedings of EMNLP 2025 (Industry Track)*, 2025. arXiv:2509.21837.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations of large language models on reasoning and planning tasks. In *International Conference on Learning Representations (ICLR)*, 2025.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP*, 2021.

Boyang Wang, Yash Vishe, Xin Xu, Zachary Novack, Xunyi Jiang, Julian McAuley, and Junda Wu. CSyMR: Benchmarking compositional music information retrieval in symbolic music reasoning. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 4244–4249, 2026. doi: 10.1145/3805712.3809923.

Ruocheng Wang, Eric Zelikman, Gabriel Poesia, Yewen Pu, Nick Haber, and Noah D. Goodman. Hypothesis search: Inductive reasoning with language models. In *International Conference on Learning Representations (ICLR)*, 2024a.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2024b.

Zhilin Wang, Zhe Yang, Yun Luo, Yafu Li, Xiaoye Qu, Ziqian Qiao, Haoran Zhang, Runzhe Zhan, Derek F. Wong, Jizhe Zhou, and Yu Cheng. Towards an AI musician: Synthesizing sheet music problems for musical reasoning. *arXiv preprint arXiv:2509.04059*, 2025a.

Zhilin Wang et al. AutoMusicTheoryQA. https://github.com/Linzwcs/AutoMusicTheoryQA, 2025b. GitHub repository. Official code for arXiv:2509.04059; ABC corpus at `data/abc.jsonl`.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *International Conference on Learning Representations (ICLR)*, 2024.

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 (ICLR)*, 2023.

Xi Ye, Qiaochu Chen, Isil Dillig, and Greg Durrett. SatLM: Satisfiability-aided language models using declarative prompting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Zhongkai Yu, Chenyang Zhou, Yichen Lin, Hejia Zhang, Haotian Ye, Junxia Cui, Zaifeng Pan, Jishen Zhao, and Yufei Ding. ChipBench: A next-step benchmark for evaluating LLM performance in ai-aided chip design. *arXiv preprint arXiv:2601.21448*, 2026.

Lifan Yuan, Yangyi Chen, Xingyao Wang, Yi R. Fung, Hao Peng, and Heng Ji. CRAFT: Customizing LLMs by creating and retrieving from specialized toolsets. In *International Conference on Learning Representations (ICLR)*, 2024a.

Ruibin Yuan et al. Chatmusician: Understanding and generating music intrinsically with llm. In *Findings of ACL*, 2024b.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *International Conference on Learning Representations (ICLR)*, 2024.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-1933.

Lizhuo Zhang, Mengmeng Tang, Chenfeng Long, Xiaoyong Tang, and Xiang Luo. Decomposing wrong-consensus agreement in LLM self-consistency. *arXiv preprint arXiv:2608.18795*, 2026.

Roadmap. The audit's appendices are B.2 (the training recipe behind Table 4's historical rows), B.4 (settings), U (v2 and the options-only audit), X (PAL, SYN, SYN-14B, SYN-NT; residual, X.3), Y (ROBUST, ROBUST-N, SYN-R, ROBUST-P/P7), Y.5 (ROBUST-P2–P4, ROBUST-F) and Z (PW-AUDIT, PW-FRESH, LOGICLM, followed by the registration ledger, Z.2). The remaining sections are retained from earlier training and residual-verdict studies and support no claim in the body.

Experiment names. The names "prover" and "oracle" in the appendices imply no formal soundness. The appendices and the ledger use the names in our code; the body describes the same experiments in words. ROBUST is the rephrasing test (two rephrasings, header addition, distractor repair); ROBUST-N the same-model paraphrase and normalization test; ROBUST-NP the end-to-end paraphrase test with three Qwen policies; ROBUST-P/P7 the first policy runs; ROBUST-P2 the corrected policy test (its predictions H-Q1–H-Q4); ROBUST-P3 and P4 the repetition-penalty and sampling re-runs; ROBUST-F the Llama second-family test; PW-AUDIT the ProofWriter audit (H0–H4); PW-FRESH the fresh-theory test of abstaining on Unknown; and LOGICLM the audit of Logic-LM. SYN, SYN-NT, SYN-14B, SYN-R, PAL, v1 and v2 keep their body names.

A Results under the benchmark's own protocol

Additional model evaluations (post-hoc). *Cross-generation transfer.* Stages A and B applied to the Qwen3.5 bases reach 80.4% at 9B and 82.9% at 27B on SSMR-Bench, so the recipe is not specific to the Qwen2.5 generation. *ChatMusician* (Yuan et al., 2024b). The prior music-specialized LLM scores 13.9% through our exact pipeline (Table 4). That number understates it as a model: its 2024 LLaMA2-7B emits free-form chord names rather than answer letters, so most of the loss is format, not musicianship. We report it because it is the only prior music-specialized LLM we can run, not as evidence about its authors' claims.

The self-run teacher comparison (post-hoc). Of the 400 stratified SSMR-Bench questions put to DeepSeek-R1 under our protocol, 18 outputs did not parse and are scored wrong, giving the 93.3 in Table 4 against the harness's 93.8 on the same 400 items (94.1 on all 1,600). On the 382 that did parse the teacher scores 97.6% against the harness's 95.5% (paired CI $[-4.5, +0.3]$). We report the parse failures as errors in the headline number and the parseable-only comparison here, because the first is the fair system-level measurement and the second is the one favourable to the teacher; neither is significant.

Table 3 gives the official-protocol numbers (zero-shot, the `<think>/boxed` template) referenced from Appendix B.3. The teacher and GPT-4.1-mini are additionally self-run under the same protocol; the reproduction residual cuts both ways (GPT-4.1-mini +3.5, teacher −3.6), which is why we anchor every protocol claim with a self-run.

Table 3: **Protocol check (post-hoc): the oracle-first harness wins; without it, the mixed single models (78.2/86.4) and 14B R1-SFT (78.5) already exceed the cited GRPO-8B (70.9, not re-run), while the base and 7B R1-SFT rows do not.** Zero-shot, `<think>/boxed` template. The last row, a harness without format-coherence, is a *partial* run (the first 664/448 items: all chord and part of beat), so it is not comparable to the rows above and we draw no conclusion from it.

	Single model		
Reference / Model	7B	14B	cited
GRPO-8B (published)	—	—	70.9
DeepSeek-R1 (published / our run, n=400)	—	—	93.6 / 90.0
GPT-4.1-mini (published / our run)	—	—	71.7 / 75.2
base	45.9	53.1	
+ R1-SFT	66.2	78.5	
+ mixed (single model)	78.2	86.4	
+ mixed + oracle-first harness	**93.9**	**95.7**	
naive harness (partial run, n=664/448)	66.6	82.6	

B Training studies, reproducibility details and background

B.1 Released artifacts

The supplementary material includes training and data-generation configurations, evaluation harnesses, frozen checkers, pre-registrations, and per-item results. `bench/REPRODUCE.md` maps

analyses to committed inputs: the audit's artifacts (PAL, SYN, SYN-14B, SYN-NT, SYN-R, ROBUST, ROBUST-N, ROBUST-P–P4, ROBUST-NP, ROBUST-F, PW-AUDIT, PW-FRESH, the multi-substitution and task-preservation checks) and, as legacy artifacts of the retained earlier studies, the SSMR-Bench single-model and harness runs and caches, frontier anchors, held-out music evaluations, the pre-registered confidence-router comparison, E5's matched-evidence scores, E6 and E7 (with full per-item completions), and scored external-domain cells except Verilog's. ROBUST's registered run saved aggregate counts only; a post-hoc re-scoring stores per-item checker outcomes (`bench/robust/robust_peritem.json`) that reproduce every registered aggregate. ROBUST-P's and ROBUST-P7's registrations, per-item policy outputs, raw text and analyses, including the post-hoc strict re-scoring, are in `bench/robust/` (Appendix Y.4). The corrected policy test's registration, analysis and post-hoc overall-gain script are in the same directory; `results_p2/` retains every raw completion, strict and lenient score, and finish reason (Appendix Y.5); `results_p3/` and `results_p4/` do the same for the two pre-registered reruns. ROBUST-N's paraphrases, normalizer outputs and per-item checker outcomes are in `bench/robustn/`. PW-FRESH's registration, fresh-item outputs and analysis are in `bench/pwfresh/`; ROBUST-F's registration, raw completions and analysis in `bench/family/`; the post-hoc multi-substitution test in `bench/subst/`; SYN-R's registration, frozen programs, development curve, test outputs and analyses in `bench/synr/`; ROBUST-NP's registration, policy outputs and analysis in `bench/robustnp/`. PW-AUDIT's frozen programs, registration, per-item checker outputs, policy completions and post-hoc parsing controls (`pw_posthoc.py`) are in `bench/pwaudit/`; Appendix Z gives the analysis command. `bench/hostcopies/` holds the MATH-500 problem file, the raw completion caches of E1–E4 (both sizes; E7 reads the 7B ones), and the logic per-sample generations behind the oracle-less gate; `bench/reproduce_setup.sh` links these and other committed copies to the run host's absolute paths so that the frozen scripts run unchanged. Table 4's historical Qwen runs retained no raw completions, so their legacy-parser credits cannot be corrected retrospectively. The committed outputs support the re-analyses for which their required inputs exist. Appendix B gives the training and decoding settings, and Appendix A the protocol variants.

Three limits remain. (i) Verilog and the self-built logic re-decision arms retain only aggregate counts; the music arms' per-item generations are committed but their scored outcomes only as aggregates, so these half-widths are conservative unpaired two-proportion intervals (Table 5). (ii) The raw completion caches of E5/E6 and the scored Verilog program dumps remain on the run host (E5/E6's scored per-item results are committed), so raw-cache-dependent audits of those cells still need those files; E5/E6 arm texts that were never retained cannot be recovered (Appendix S). SYN-14B and SYN-NT pin SYN's code by commit, so the SYN-family analyses need a clone at the original absolute path; those that re-execute a frozen checker also need its sandbox interpreter. (iii) Regeneration needs the relevant GPU or paid API access and is not bit-reproducible. The distilled-model scores depend on the pinned vLLM 0.8.3 configuration, and continuous batching can change outputs even at a fixed seed (Appendices B and O). The committed outcomes are reproducible evidence; new runs need not reproduce their last digit.

B.2 Training recipe and historical evaluations (post-hoc)

The historical training studies evaluate the earlier training recipe (Figure 5). **Stage A** distills 2,487 correct DeepSeek-R1 chains-of-thought from SSMR-Bench training questions into Qwen2.5-7B and -14B using LoRA (Hu et al., 2022), rank 16, for three epochs. **Stage B** broadens this supervision with questions about harmony, form, rhythm, texture and expression. R1 generates a question about a sampled ABC excerpt; an independent call answers it. We keep the answering trace when the answers agree. The 5,000 attempts yield 697 traces, added to Stage A's data before re-distillation. The excerpts come from the corpus used by SSMR-Bench's generator, so some rhythm test items overlap; a post-hoc sensitivity analysis (Appendix T) does not identify a causal contamination effect.

Stage C, the v1-based harness, uses the checker first. On abstentions, a planner lists the facts to derive, then three solver samples at temperatures 0.0, 0.5 and 0.8 vote. Both prompts also carry a `music21` analysis of the score, and on chord-name items the solver prompt gets v1's computed chord quality as an advisory hint. Every sub-prompt carries the same final-answer format instruction. The harness includes neither a verifier nor multi-turn retry, an empirical choice from post-hoc sweeps (Table 7), not a consequence of a self-correction law (Appendix V). This planner-and-vote fallback differs from the ladder's greedy one; their gains answer different questions. Besides SSMR-

Table 4: **Historical music evaluations and training-stage comparisons.** Accuracy (%); best per column in **bold**. Historical self-run comparisons are post-hoc except the marked SYN result. Every Qwen row parses answers leniently, falling back to the last bare option letter (single-model runs, harness vote samples, the SYN rows' fallback), and retained no raw completions. The strict replication below supplies a separate training comparison. Published references use their authors' protocols. The v1-based harness adds a planner, vote and `music21` analysis (post-hoc). †SYN with an untuned greedy fallback (pre-registered on SSMR-Bench). Neither checker fires on MusicTheoryBench/WildScore (post-hoc); those SYN cells equal the base. Avg uses unrounded accuracies.

Model	SSMR-Bench	MusicTheoryBench	WildScore	Avg
Published references (cited, not re-run)				
DeepSeek-R1 (671B, teacher)	93.6	–	–	–
AI-Musician GRPO (Qwen3-8B)	70.9	–	–	–
GPT-4.1-mini	71.7	–	–	–
Self-run baselines (our eval pipeline)				
ChatMusician (LLaMA2-7B)	13.9	–	–	–
DeepSeek-R1 (teacher, n=400)	93.3	–	–	–
GPT-4.1-mini (this protocol)	81.4	–	–	–
Claude Opus 4.8 (brief reasoning)	96.2	–	–	–
Qwen2.5-7B				
base	44.2	57.2	47.0	49.5
base + harness	87.5	55.9	50.2	64.5
base + frontier-written checker†	96.1	57.2	47.0	66.7
+ R1-SFT	64.8	51.0	36.8	50.8
+ mixed	74.9	51.8	38.4	55.0
+ mixed + harness	92.9	53.1	43.6	63.2
Qwen2.5-14B				
base	55.9	55.3	55.4	55.5
base + harness	88.9	**58.9**	**56.1**	68.0
base + frontier-written checker†	**96.8**	55.3	55.4	**69.2**
+ R1-SFT	76.2	56.4	41.5	58.0
+ mixed	79.0	54.8	38.3	57.4
+ mixed + harness	94.1	54.2	52.8	67.0

Strict, 8,192-token replication (ROBUST-P2's pre-registered runs, no repetition penalty): on original SSMR-Bench items, untuned / R1-SFT accuracy is $46.88/58.88\%$ at 7B and $54.44/75.81\%$ at 14B, post-hoc gains of $+12.0$ and $+21.4$ points; they do not re-score the historical rows.

Bench, Table 4 evaluates MusicTheoryBench (Yuan et al., 2024b) (367 music-theory MCQs) and WildScore (Mundada et al., 2025) (807 musicological MCQs, text-only track), both held out of training (Appendix Q).

B.3 Training gains depend on the checker (post-hoc)

The new strict, pre-registered runs (Table 4 note) give distillation gains of $+21.38$ $[+18.50, +24.25]$ points at 14B and $+12.00$ $[+8.94, +15.12]$ at 7B (post-hoc contrast, not registered; lenient $+25.69/+19.06$). Historical stage ablations use the legacy parser and are stack-sensitive. In those post-hoc runs, applying the v1-based harness to the untuned base leaves a $+5.2$-point contribution from training at 14B.

Changing the checker changes the value of training. With the greedy fallback fixed, the three trained 14B seeds beat the untuned base by 5.3–6.2 points behind v1, but only 0.5–0.9 behind SYN; two of the latter intervals include zero (post-hoc, Appendix X.2). SYN with the untuned fallback also exceeds all three trained v1-based systems by 2.5–2.8 points (post-hoc). Training helps the residual a particular implementation leaves, so its marginal value cannot be read from the single-model gain.

Post-hoc results under the benchmark's own answer template are in Appendix A. Our post-hoc LoRA-GRPO sweep gains at most $+3.7$ $[+1.1, +6.3]$ (Appendix M); different protocols, backbones and training procedures prevent a matched RL–distillation comparison.

Table 5: **Cross-domain summary; the grid shows which pre-registered re-decision arms were *not* run where** (● run, ○ exploratory sweep arm excluded from Δ, – not run). Prov. S self-built, A externally authored, C third-party checker; Cov. prover coverage (for Verilog, problems with at least one public-test-passing program: a high-precision checker, not a sound prover); Unan./Dis. accuracy when samples or behaviors agree vs. disagree (7B/14B; MBPP+ instead reports the precision of its program-text commit rule); Δ best re-generation arm vs. its registered aggregation control, which pools more samples than the arm (so not matched; Appendix V), ±95% half-width, bold excludes 0; Sign: all re-generation arms ≤ 0 († arms not counted). Arm codes, interval conventions, † and * are defined in Appendix B.4. The DB (debate) arm is *withdrawn* from the registered re-derivation comparison (Appendix O); its cells are shown as run, not as support.

					Arms run								
Domain (7B/14B)	Prov.	Cov. %	Unan. %	Dis. %	SR	SY	DB	TN	RR	CW	SS	Best re-gen Δ (pp)	Sign
SSMR-Bench	A+S	69.9	89.0/89.8	25.6/34.2	○	●†	○	●	○	●†	–	TN −9.0±10.5/−**16.0**±12.0	✓/✓
MusicTheoryBench	A	–	61.7/66.4	15.9/22.5	–	–	–	–	–	–	–	–	–
WildScore	A	–	45.6/58.0	30.8/33.7	–	–	–	–	–	–	–	–	–
Verilog	A+C	64.0	93.6/93.8	53.3/60.0	–	●	–	●	●	–	●	SS −2.0±19.3	✓
Logic (n=1500)	S	34.2	99.8/99.3	85.8/83.7	–	●†	●	–	–	–	●†	*withdrawn* (DB)	–
MATH-500	A	–	99.3/98.6	57.1/66.4	●	●	●	●	–	–	–	SR −0.5±3.9/−**5.2**±4.3	✓/✓
MATH-500 (Llama-8B)	A	–	100.0	36.8	–	–	●	●	–	–	–	TN −**17.4**±3.9	✓
MMLU-Pro	A	–	88.4/83.8	42.2/44.6	●	●	●	●	–	–	–	SR −0.8±3.3/−1.8±4.3	✓/✓‡
MBPP+ (7B)	A+C	86.8	85.1	–	●*	–	–	–	●	–	–	SR* +6.0±10.0	×

B.4 Settings and Supplementary Evaluation Details

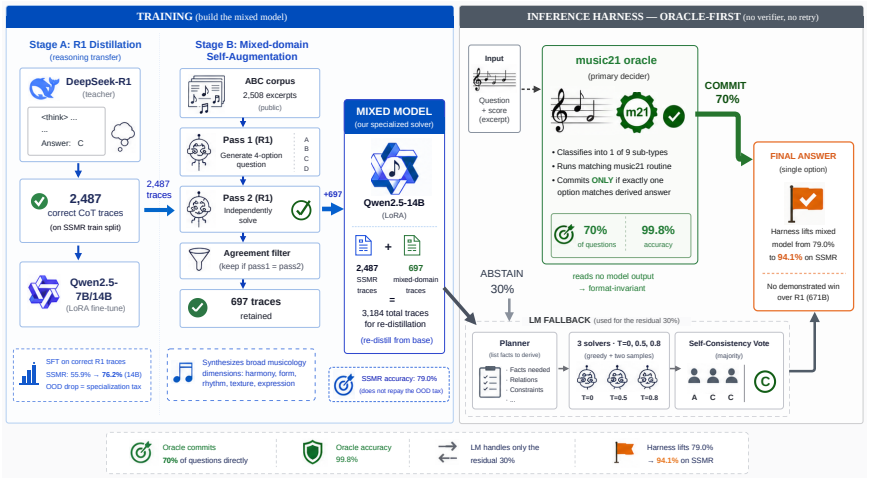


Figure 5: **The v1-based training-and-inference pipeline behind Table 4's historical rows.** *Training* (left): distilling correct DeepSeek-R1 traces (Stage A) and two-pass self-generated mixed-domain questions (Stage B). *Inference* (right) is oracle-first: a `music21` prover commits wherever its derived value uniquely matches an option—70% of SSMR-Bench at 99.8%, independent of the model's answer format—routing its abstentions to a planner and a three-sample vote. The harness ships *no* verifier and no retry (Table 7). This is the v1-based pipeline; the registered SYN comparison uses a greedy fallback. Its question dispatcher is sensitive to rephrasing (ROBUST). Accuracies in the image are Table 4's historical legacy-parser values.

Legend for Table 5. Figure 6 shows the agreement partition per domain; Appendix W.1 discusses it. The external studies use Verilog RTL generation (Appendix F), self-built deductive logic (Appendix G), MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024) (Appendix H), MMLU-Pro (Wang et al., 2024b) (Appendix I), ProofWriter and ARC-Challenge (Clark et al., 2018) (Appendix I.2) and MBPP+ (Liu et al., 2023a; Austin et al., 2021) (Appendix J); each section retains its registration and outcome. Arms: SR self-refine/retry; SY synthesis; DB debate; TN pairwise

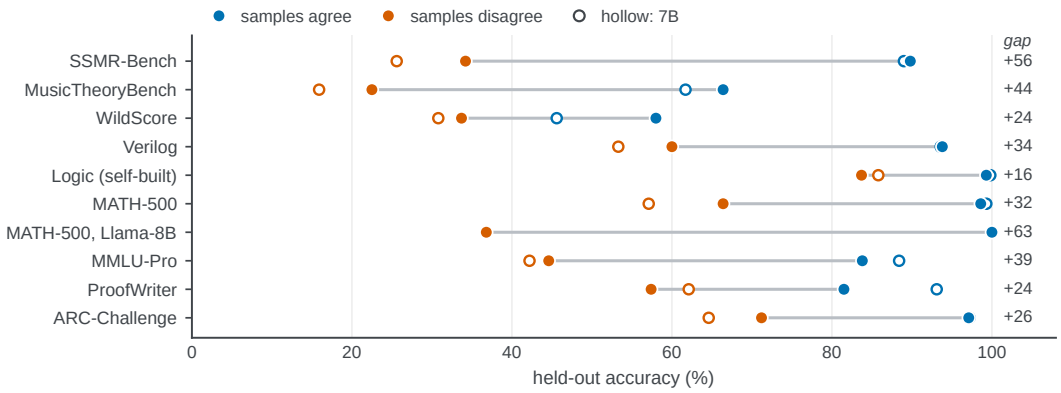


Figure 6: **The agreement partition recurs in every domain shown.** Held-out accuracy where self-consistency samples agree (blue) versus disagree (vermillion); filled markers are the 14B model or the only model run, hollow markers the 7B model, and the gap is the filled pair's. It is a registered hypothesis in MATH-500 and MMLU-Pro and specified in advance, without a decision rule, in the logic setup, at both sizes (all six survive Benjamini–Hochberg control, logic on its original n=600 run, as does the post-hoc Llama test); elsewhere it is post-hoc descriptive. Logic markers show the later, unregistered n=1500 replication (as in Table 5); the n=600 run gives $100.0/75.2$ at 14B and $99.5/85.4$ at 7B (Table 11). The music residual shown is v1's; SYN is reported separately in Appendix X.3. MBPP+ is not shown: grouping its public-test passers by public-test outcome leaves no disagreeing items (328 unanimous, 0 disagreeing), so Table 5 reports its commit precision instead. Which re-decision arms were run where is in Table 5 (Appendix B.4); for ProofWriter and ARC-Challenge, in Table 14.

tournament; RR best-of-N rerank; CW confidence-weighted vote; SS score-argmax (judge-score, a re-generation arm, in Verilog; self-score, an evidence-reading arm, in logic). † marks arms registered as re-decision whose positive Δ violates their domain's registered sign rule: music CW and SY at both sizes ($+8.4/+6.9$ and $+22.9/+21.4$ on the disagreement subset, 7B/14B; the frozen criterion scores $3/5$ at each size, Table 15) and logic SS and SY at 14B (Table 11). Sign counts re-generation arms only and excludes these; classing them as evidence-reading rather than re-decision is post-hoc (Appendix L), so SSMR-Bench's Sign ✓ rests on TN alone. Half-widths are unpaired two-proportion bounds for music, Verilog and logic (only counts are committed) and paired bootstraps for MATH-500, MMLU-Pro and MBPP+. SSMR-Bench arms are scored on the blind-test run (disagreement n=166/131, Appendix L), its partition on the shipped run. *post-hoc arm. ‡MMLU-Pro's registered strong-arm sign rule (H-SA2, scored against the plain vote) is violated at 14B by the tournament arm ($+0.7$); against the registered control all arms are ≤ 0, and against a genuinely compute-matched control $7/8$ are (Appendix I.1). Every cell is read from a committed result file or produced by a committed script (`crossdomain_sources.md`).

Training. R1-SFT and mixed-data SFT use LoRA (rank 16, alpha 32, dropout 0.05) on all attention and MLP projections, three epochs, batch 1 with gradient accumulation 8, learning rate 1×10^{-4}, cosine schedule, sequence length 4096, bf16. Training runs in a containerized LLaMA-Factory environment for reproducibility; configs are released.

Distillation data. Stage A retains 2,487 correct R1 traces from the SSMR-Bench training split. Stage B generates against 2,508 public ABC excerpts across five dimensions; the two-pass agreement filter retains 697. The combined corpus is 3,184 traces (2,984 train, 200 val).

GRPO. Verifier-shaped GRPO uses TRL with a binary correctness reward ($+1$ correct, -1 wrong), 4 generations per prompt, max completion 2560 tokens, warmup 0.1, gradient clip 0.5. We sweep the learning rate over $\{1, 5, 20\}\times10^{-6}$ and the budget over 300–4,500 steps (0.2–3 epochs). We also tried a shaped reward (final correctness + format + `music21` process check + length penalty); it reward-hacked faster. Table 6 gives the sweep and per-category breakdown.

Table 6: **GRPO gains at most $+3.7$ across a $20\times$ learning-rate sweep and a 0.2–3 epoch budget.** The best of four configurations gains $+3.7$ $[+1.1, +6.3]$ paired, which survives a Bonferroni correction for the selection; the others are within noise ($+0.6$, $+2.2$, -1.3). LoRA-GRPO on Qwen3-8B-Base. The training reward never trends up in any configuration (the 3-epoch run's OLS slope is $+0.06$ over $4{,}500$ steps, per-step σ=0.44); the best cell moves $+3.7$ (post-hoc sweep). Our measured 46.1 base and the cited 70.9 endpoint use different protocols; their difference is not the published training gain. Three epochs at the matched rate improve less than one at a higher rate, which does not rule out gains from a different budget or configuration. We do not claim the published endpoint is unreachable at full scale, while the contrast with distillation is *cross-backbone*: we ran no Qwen3-8B distillation, so the $+20$ to $+31$ it buys is measured on the Qwen2.5-7B and -14B bases of Table 4 (three epochs of LoRA SFT), not on the base swept here. That range comes from Table 4's legacy-parser rows; the strict replication of R1-SFT gives $+12.0$ at 7B and $+21.4$ at 14B (post-hoc contrast, Appendix Y.5).

Config	budget	SSMR-Bench	Interval	Chord	Scale	Beat
Qwen3-8B-Base	—	46.1	41.5	49.0	42.8	51.0
GRPO 1×10^{-6}	0.2 ep	44.8	46.5	46.0	39.0	47.5
GRPO 1×10^{-6}	3 ep	46.6	44.8	48.2	41.2	52.2
GRPO 5×10^{-6}	1 ep	49.8	46.0	49.2	46.5	57.3
GRPO 2×10^{-5}	1 ep	48.2	45.0	49.5	43.0	55.5
GRPO (published)	full	70.9	–	–	–	–

Oracle development hygiene, and a post-hoc held-out-draw control. The oracle's matchers, sub-type whitelist, and override/advisory gating were developed and debugged against the public SSMR-Bench test split; the benchmark ships no separate development split, so on its face the $70\%/99.8\%$ figures could be adaptively-selected estimates rather than one-shot measurements. We test this directly. We run the *frozen* oracle—not one line changed—on the $8{,}000$-question `Train_textual` split, which the same public generator produced but which the oracle never saw during development (it is used only for distillation). **The oracle holds out of sample**: it fires on $5{,}620/8{,}000 = 70.25\%$ of held-out draws against 69.9% in sample, and is correct 99.86% of the time when it fires ($5{,}612/5{,}620$) against the in-sample 99.8%. Every firing sub-type is at or above 99.3% out of sample (interval number $1{,}528$ fires, note completion $1{,}252$, time signature $1{,}096$ at 99.3%, key identification 868, chord root 467, bar-line 403, all others 100%), but chord completion fires on only 6 held-out draws against 136 test commits, so its test precision has almost no out-of-sample check. The coverage comparison is not like-for-like either: the training split's category mix differs (interval $2{,}902$, beat $2{,}848$, scale $1{,}660$, chord 590, against 400 each in the test split).

We report this figure only after correcting our own control. The two splits place the ABC key signature in different fields—`Test_textual` in `abc_context`, which `oracle.py` reads, and `Train_textual` inside `question`—so our first run of this control analysed all 467 held-out chord items in default C major and scored the chord-root matcher at 67.6%. We had attributed that to `music21`'s `root()` disagreeing with the generator in minor and flat keys and demoted the matcher on those grounds; the diagnosis was wrong, the demotion was unnecessary, and with the header supplied the same frozen oracle—not one line changed—scores $467/467$. The correction is in the control (`clean_room_oracle.py`), not in the oracle, whose shipped inputs were never affected. This control tests new draws from the same generator. It does not test new task wording: ROBUST finds large coverage losses under rephrasing (Appendix Y).

Harness. Sub-type detection is a keyword classifier over the question text (nine types; 100% agreement with the dataset's type labels). The oracle commits to an option only on a *unique* match and abstains otherwise; six sub-types (chord-root, interval, note-/chord-completion, scale-id, bar-line) hard-override, time-signature overrides only under an equal-bar parse, chord-id yields only an advisory hint, and scale-selection has no matcher and always abstains. When the oracle abstains, a planner call (temperature 0.3) precedes a three-sample ensemble at temperatures $\{0.0, 0.5, 0.8\}$ with a majority vote. A single format instruction—"`Answer: X`" or, under the official protocol, a `\boxed{}` directive—is threaded into every sub-prompt; under the official protocol the per-call budget is raised to 6144 tokens so long `<think>` traces close before truncation. No verifier or multi-turn retry is used: a sweep of four verifier designs found none beats self-consistency voting

Table 7: **No tested verifier demonstrably improves on voting (post-hoc).** Five modes on the 481 oracle-abstain residual questions (127 with MoA disagreement), 14B, planner and MoA samples held *fixed*. Re-verification is neutral (within 0.6 of the vote); only best-of-N reranking, which discards the vote to re-pick, is net-negative (-5.2). The 7B model shows the same pattern (off 74.4, flip/grounded/vote 74.2–75.1, rerank 69.6). All at a 2,560-token budget; an earlier 1,024-token run truncated the distilled model's reasoning and produced spurious net-negatives (both result files are committed, `bench/ssmr/verifier_sweep_*`).

Verifier on the residual	all (481)	disagree (127)
none (self-consistency vote)	**81.1**	48.0
conservative flip (shipped→off)	81.7	50.4
tool-grounded re-selection	81.7	50.4
self-consistent (3-sample)	81.1	48.0
best-of-N rerank	75.9	28.3

once the oracle is present—re-verification is neutral, and re-deciding the residual by reranking or retry is net-negative (Table 7; retry: `bench/ssmr/multiturn_diag_*`).

Evaluation and stack sensitivity (post-hoc). SSMR-Bench and MusicTheoryBench run with vLLM 0.8.3 (the `EVAL_LEGACY_SP` sampling path); Qwen3 models, unsupported by that build, run with HuggingFace generation. *The distilled model is sensitive to the inference stack*: on test items 0–299 the distilled 14B scores 76.3 in the headline run and 78.3 on a rebuilt twin of the same engine, but 58.7–66.3 under other engine versions and sampling settings (eager mode, fp16, longer outputs, a repetition penalty), while the untuned base, probed on one of these stacks, moves from 54.3 to 57.3 (the distilled model scores 60.0 on that same stack; the base was not run under the other settings) (`bench/ssmr/results_qwen25-14b-*probe*`). The trained model's absolute numbers therefore hold for this pinned engine; the checker numbers do not depend on it. All accuracies are exact-match on the held-out splits. Option order is fixed by a per-question seed for determinism. Figure 7 plots the stage-by-stage SSMR-Bench accuracies of Table 4 at both model sizes.

B.5 Background for the secondary studies

Distillation and reinforcement learning. Reasoning distillation is established in math and code (DeepSeek-AI, 2025), and fine-tuning specialization is a known cost (Luo et al., 2025; Kumar et al., 2026). We measure its marginal value behind different checkers and its held-out music cost under a matched harness. Our LoRA-GRPO sweep complements SSMR-Bench's published GRPO result (Wang et al., 2025a; Shao et al., 2024); it is not a matched RL–distillation comparison, as backbones and training procedures differ.

Agreement, self-correction and routing. Self-consistency (Wang et al., 2023) and sample agreement (Xiong et al., 2024) provide known confidence signals, used in adaptive sampling (Aggarwal et al., 2023) and hallucination detection (Manakul et al., 2023). They can fail through wrong consensus (Ding, 2026; Bahuguna, 2026; Zhang et al., 2026). Our residual partition replicates this signal and uses it diagnostically. The cascade is likewise an application of selective prediction and agreement-based routing (Geifman & El-Yaniv, 2017; Chen et al., 2024a; Yue et al., 2024; Kolawole et al., 2025; Soiffer et al., 2025), not a new routing algorithm. Prior work finds intrinsic self-correction and debate unreliable relative to voting (Huang et al., 2024; Smit et al., 2024; Choi et al., 2025), while external or executable feedback can help (Gou et al., 2024; Chen et al., 2024b; Kamoi et al., 2024), with gains bounded by feedback quality (Olausson et al., 2024). Our registered controls leave open whether re-deriving verdicts hurts once evidence and aggregation width are matched. The Verilog appendix uses behavioral clustering as in AlphaCode and CodeT (Li et al., 2022; Chen et al., 2023a); MBPP+'s commit rule instead selects the most frequent program text among public-test passers.

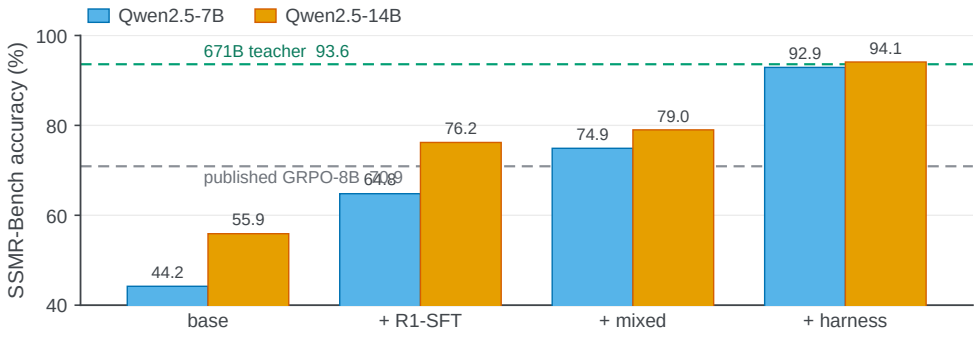


Figure 7: **Stage ablations behind v1 (post-hoc)** (with a frontier-written checker, distillation adds under a point at 14B, post-hoc; Appendix B.3). SSMR-Bench accuracy as training stages stack (deterministic protocol). The 14B full system (94.1) clears the published GRPO-8B reference (dashed, 70.9) and shows no demonstrated difference from the 671B teacher (dashed, cited 93.6; a same-protocol re-run scores 93.3 against 93.8, CI $[-2.3, +3.5]$)—most of the gain being the oracle.

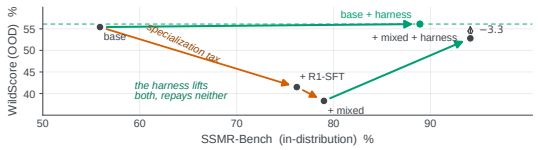


Figure 8: **The specialization tax, and who does not pay it.** The 14B family's path through (in-distribution, out-of-distribution) accuracy: distillation moves right but *down*, and the harness pulls the trained model back up—but it lifts the *untuned base* too (dashed line), and the full recipe ends 3.3 points below it on WildScore while keeping the in-distribution gain (post-hoc matched control; -4.6 $[-6.7, -2.5]$ pooled over the four held-out cells, Appendix Q).

C Mixed-domain augmentation prompt

Stage B's pass 1 prompts R1 to design a question about a target dimension: *"You are an expert music theorist designing a multiple-choice exam. Given the following ABC notation excerpt, design ONE non-trivial multiple-choice question about the {dimension} of this music. Then solve it step-by-step."* followed by a strict output format (question, four options, reasoning, answer). Pass 2 takes the generated question and options and prompts R1 to *answer* it with the standard SSMR-Bench-style instruction. We retain the example only when the pass-2 answer matches the pass-1 key; the supervised target is the pass-2 reasoning. The five target dimensions are harmony & tonality, form, rhythm & meter, texture, and expression & performance. Of 5,000 generation attempts, 697 survived both the format and the cross-agreement filters.

D Per-category breakdown

Table 9 gives the full per-category SSMR-Bench accuracy for the 14B family. The harness lifts every category over the bare mixed model, with the largest gains on interval ($76.2 \to 98.8$, the `music21` oracle) and chord ($75.7 \to 94.5$); beat ($68.5 \to 85.5$) gains mostly from oracle bar-counting (the planner's share is not detectable).

E Harness prompts

When the oracle abstains, the residual is handled by two prompt templates over a shared base question. The base prompt presents the ABC notation, question, and four options, and ends with a single format instruction, threaded from one source so it is identical across calls: *"End with a line exactly: Answer: X"* (deterministic protocol) or *"…put your final choice in* `\boxed{X}`*"* (official protocol). A `music21` analysis block (notes, key, time signature, computed interval/chord) is added to

Table 8: **The oracle is the harness.** `music21` oracle coverage on SSMR-Bench (1,600). Sub-type detected from question text (100% accurate vs. the dataset's type labels); the oracle commits only on a unique option match and otherwise abstains to the model. It fires on 70% of questions at 99.8% accuracy, independently of the model's answer format. Question rephrasing does affect it (ROBUST). Chord-id is advisory-only and scale-selection has no matcher; neither overrides.

Oracle sub-type	n	fires	acc. when fires
chord-root	200	200	100.0
interval-name	199	199	100.0
note-completion	201	189	100.0
chord-completion	156	136	100.0
scale-id	352	175	100.0
bar-line placement	183	52	100.0
time-signature	217	168	98.8
chord-id (advisory) / scale-sel. (none)	92	0	—
Total	1600	**1119 (70%)**	**99.8**

Table 9: Per-category SSMR-Bench accuracy (%), 14B family (post-hoc).

Model	Interval	Chord	Scale	Beat	All
base	56.8	53.0	56.5	57.5	55.9
+ R1-SFT	74.3	73.5	92.5	64.5	76.2
+ mixed	76.2	75.7	95.5	68.5	79.0
+ mixed + harness	98.8	94.5	97.5	85.5	94.1

the planner and solver prompts, and on chord-id items the solver prompt also gets an advisory oracle hint (the computed chord quality). The two role prompts extend the base:

> **Planner.** "Briefly outline (in 2–3 short bullet points) what musical facts you need to derive in order to answer the question. Do NOT solve yet. Do NOT output the answer letter."
>
> **Solver.** The base prompt with the planner's notes appended, asking for step-by-step reasoning ending in the format instruction above. Run at temperatures $\{0.0, 0.5, 0.8\}$; a majority vote selects the answer, with a salvage scan for a final standalone letter if a sample truncates.

An earlier design added a verifier prompt and a multi-turn retry; both were removed after sweeps showed no verifier beats self-consistency voting and every multi-turn re-decision is net-negative once the oracle is present. (A pre-registered blind test raises an exploratory question of whether re-reading the samples' reasoning could help where evidence is cheaply re-verifiable; on the shipped harness's tighter residual core even a perfect selector over the same candidates is worth about a point of system accuracy ($+1.1$ $[+0.6, +1.5]$; its margin over the null, $+1.9$ $[-9.3, +13.1]$ at n=79, is not separated from zero—Appendix K), so we ship the plain vote.)

E.1 The data-construction bug, verbatim (post-hoc)

The single-pass augmentation produced training targets whose `<think>` block reasoned about *how to construct* a question. At test time the model reproduced this behavior. A representative completion on a WildScore question began: "*We need to design a non-trivial multiple-choice question about Harmony & Tonality from the given excerpt…*" and never emitted an answer letter. Across WildScore, $624/807$ (77%) completions had no parseable answer, yielding 11.0% accuracy. The two-pass fix eliminated the failure.

F Agreement and selection in program synthesis

We examine the agreement partition and candidate selection in Verilog RTL generation, where the checker is test execution. The partition is post-hoc; the intervention sign test is pre-registered.

Neither identifies a general mechanism for self-correction, and the intervention residual has a label-selection limitation described below.

Setup. The main experiment evaluates the paper's own untuned students—Qwen2.5-7B/14B-Instruct, byte-identical to the `base` checkpoints of Table 4—on three benchmarks: RTLLM-v2 (Lu et al., 2024; Liu et al., 2024) (44 of 50 problems whose golden reference passes its own oracle), VerilogEval-v2 (Liu et al., 2023b; Pinckney et al., 2025) (152 of 156), and ChipBench (Yu et al., 2026) (29 of 30), K=10–20 samples per problem at temperature 0.8 (8,593 scored programs; a Qwen3-14B pilot with 1,960 programs gave the same picture). A *two-tier* oracle mirrors prove-then-route: the PUBLIC tier is each benchmark's own upstream hand-written testbench—the realistic, gap-prone signal a developer actually has—and the HELD-OUT ground truth is exhaustive random differential testing against the golden reference (4096 vectors $\times$ 3 seeds, all must agree). The asymmetry is real and concentrated where testbenches are directed rather than comparison-based: 32% of public-passing programs fail held-out on RTLLM, 2.9% on VerilogEval, 6.5% on ChipBench.

The partition replicates at scale. Clustering samples by behavioral equivalence on a fixed 256-vector stimulus (as in AlphaCode and CodeT (Li et al., 2022; Chen et al., 2023a)), behavioral self-consensus is a useful internal correctness signal: among 2,722 public-passing programs, the majority behavioral cluster is held-out-correct 86.9% of the time (2170/2497) while minority programs sit at 9.8% (22/225); per bench, 78.0 vs. 7.9% (RTLLM), 97.4 vs. 50% (VerilogEval, n=6 minority), 94.0 vs. 66.7% (ChipBench, n=3). Those are *per-program* precisions pooled over both models, a different statistic from the per-question vote accuracy we report for every other domain, and an earlier version of Table 5 and the partition figure used them as if they were comparable. Recomputed per question and label-free (`verilog_labelfree_partition.py`; a question *agrees* when all its public-passing programs share one behavioral cluster), the vote is right on 93.6%/93.8% of agreeing questions (7B/14B; 94/113) and 53.3%/60.0% of disagreeing ones (15/20), and on the 35 disagreeing questions a correct cluster is available on 74.3% against the vote's 57.1%. This is the SSMR-Bench partition in code, with a gap of about 35 points rather than the 77 the per-program statistic suggests.

A selection flaw in the registered arms, disclosed. The arm test below was registered on residual groups defined as "public-passing programs exist but are not all held-out-correct". That definition uses held-out labels, so the residual it selects is not what a deployed system could see. The arms stored only totals, so they cannot be recomputed on the label-free 35; we report them as registered and read them as sign-only evidence on a label-selected set.

The pre-registered arms. The 50 residual groups (public-passing programs exist but are not all held-out-correct; a held-out-correct candidate is *present* in 28) are the second blind test of Appendix L: four arms, classifications and sign predictions frozen and committed before implementation. Baseline (aggregation, largest-cluster representative) scores 21/50, and the sign verdict below replicates the n=44 construction that preceded the residual-growth run.

Table 10: **Second pre-registered sign test (Verilog residual, n=50 groups).** Δ = held-out-correct picks minus the aggregation baseline (21/50). The original aggregate-vs-re-decide criterion predicts every arm ≤ 0 and goes 4/4 *prospectively* (and 4/4 again on the earlier n=44 construction); the music-derived evidence-vs-verdict refinement predicts the two code-reading arms ≥ 0 and goes 2/4—the exception does not transfer.

Arm	judge consumes	Δ	agg-vs-redecide	evidence-vs-verdict
rerank by public mismatches	verdicts (counts)	-2	✓	✓
self-score 0–10, argmax	verdicts (scores)	-1	✓	✓
pairwise tournament	evidence (code)	-3	✓	×
synthesis over all candidates	evidence (code)	-5	✓	×

Why the music-side exception may not transfer (exploratory). A candidate account, which we flag as a hypothesis rather than a result: in music the judge can *re-verify* the evidence it reads—note

names and intervals are cheap to re-derive, so a synthesis pass recovers two-thirds of the selection headroom—whereas in code, checking whether a program meets the specification is about as hard as writing it; reading more programs adds no verifiable signal, behavioral agreement is informative, while none of the four tested selection arms improves on the aggregation baseline on this label-selected residual (net Δ -1 to -5 of 50). The proposed explanation is not independently confirmed at n=50. We do not attempt the deterministic "oracle answers a large fraction" leg, which has no analog where no tool writes correct RTL outright.

G RESIDUAL EXPERIMENTS IN DEDUCTIVE LOGIC

To test the partition and interventions where neither music21 nor a testbench applies, we build deductive-logic problems and pre-register the same four arms before running (commit in the repository predates the result commit).

Construction. A problem is a small definite-clause (Datalog) world: unary predicates over one entity, a set of conjunctive "if…then" rules, a few base facts, and a yes/no query "does predicate P hold?", rendered in natural language. Gold labels come from forward chaining to fixpoint, which is sound *and* complete for definite clauses; we additionally verify every label against z3 ($400/400$ agreement). Problems are balanced $50/50$ and require multi-step chains; non-entailments demand certifying that *no* derivation exists, a classic LLM failure mode, which gives a genuine residual. The prove leg is a depth-one sound forward chainer: it commits "yes" only on queries it derives within one round (a derivation is a proof, so it is never wrong) and abstains otherwise. It fires on $201/600 = 33.5\%$ of queries at 100% accuracy—a real prove-then-route leg—routing the rest to the model.

Results. On the 399 routed problems, K=10 samples per problem, both untuned Qwen2.5 students:

Table 11: **Third pre-registered domain (deductive logic).** Partition on the model-routed residual, and the four arms on its disagreement subset (Δ vs. the vote; $\pm$ is $1.96\sqrt{0.25/n}$, a conservative single-proportion half-width, not a paired interval). By the registered sign rule (every re-decision $\Delta \leq 0$) the domain confirms on 7B but is *violated* on 14B by self-score and synthesis (the two evidence-reading arms, mirroring music); both flips are small and inside that half-width, which we note as post-hoc. The original debate contrast is negative at both sizes, but is withdrawn as evidence for a re-derivation cost after E4, whose symmetric control was not run on logic (Appendix O).

	Qwen2.5-7B	Qwen2.5-14B
unanimous (acc)	187 (99.5%)	282 (100.0%)
disagreement (acc)	212 (85.4%)	117 (75.2%)
self-score Δ	-1.9 ($\pm$6.7)	$+2.6$ ($\pm$9.1)
debate/refine Δ	$\mathbf{-9.4}$	$\mathbf{-25.6}$
synthesis Δ	$\mathbf{-12.3}$	$+3.4$ ($\pm$9.1)

We are deliberate about what this does and does not confirm. The frozen rule was a sign rule ($\Delta \leq 0$ for every re-decision arm); on 14B, *self-score* ($+2.6$) and *synthesis* ($+3.4$) flip positive, so by the letter of the pre-registration the logic domain confirms on 7B and is a *failure-to-reject*, not a confirmation, on 14B. A higher-power replication (n=1500 problems, disagreement subsets 288/521, tightening the interval to $\pm 5.8/\pm 4.3$) resolves what those flips were: the 14B positives shrink toward zero (synthesis $+3.4 \rightarrow +1.0$, self-score $+2.4$) and stay well inside the tighter interval, while 7B confirms outright (debate -12.7, synthesis -8.6, both significant). So the evidence-reading gain that was *significant* in music ($+5.8/+7.9$) does *not* replicate in logic: at 14B it is within noise, and at 7B synthesis is significantly *negative*. The original debate estimates are $-12.7/-33.0$ here (7B/14B), but this arm uses the same asymmetric prompt that E4 later disqualified as support for a re-derivation cost; no surviving logic re-generation arm remains. The agreement partition is unaffected: unanimous accuracy is $99.8\%/99.3\%$ at n=1500. Evidence-reading effects are heterogeneous, including -3.0 in a post-hoc external pool (Appendix I), rather than a validated boundary between aggregation and re-decision.

Oracle-less gate (post-hoc, not registered). On the 987 problems the prove leg does not settle, the agreement gate routes alone: the 7B vote where its samples agree, the 14B on the 521 (53%) disagreements. It reaches 95.1% against 92.4% for the 7B and 94.7% for the 14B, +2.7 [+0.9, +4.6] over the 7B (`bench/logic/logic_cascade.py`; its per-sample inputs are committed in `bench/hostcopies/logic_domain/`); no random-escalator baseline was computed here.

H PARTITION AND ROUTING ON MATH-500

MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024) contains 500 externally-authored competition problems, scored here by `math_verify`. There is no in-loop checker: this experiment tests the partition and agreement gate. Predictions were pre-registered before any run.

The pre-registered partition replicates (H1). With $K=8$ self-consistency samples, unanimous agreement is held-out-correct 98.6% (14B, 289 problems) and 99.3% (7B, 281) against 66.4%/57.1% on disagreement—a strong conditional accuracy difference on this benchmark, without an in-loop checker.

The gate retains accuracy with fewer 14B calls (H4; H3 on the router only). With no oracle in the loop, the agreement-gated router (7B vote where it agrees, escalate the disagreement residual to the 14B) reaches 85.0%—which is *exactly* the pure-14B accuracy, at 44% of the 14B's calls. So the transferable payoff here is 14B accuracy with 44% of the 14B calls—not compute efficiency, since counting the eight 7B samples per item the router costs 0.96× a pure-14B run (parameter-weighted)—+4.2 over the 7B (95% CI [+1.6, +7.0]), not an accuracy gain over the stronger model. Against a same-budget *random* escalator (82.6%, [81.2, 84.0]) the gate's +2.4 confirms it escalates the right problems—but this follows mechanically from H1 (the agree-region is 99.3%), so we read it as an efficiency check on the partition, not as independent evidence. *Deviation, stated:* H3 as registered also requires the gate to beat a same-budget random escalator on the *frontier* cascade, but R1 was called only on the 14B's 211-item disagreement residual, so that half cannot be computed from committed data. A post-hoc upper bound for the random frontier escalator, crediting R1 with every item it was not run on, is 89.3% against the gate's 94.4% (`bench/math/math_h3_frontier_bound.py`). On external hard math, the partition (H1) is the claim; the gate is its label-free, oracle-free deployment.

The frontier cascade lifts the base (H2, secondary). Mirroring the SSMR-Bench cascade of Figure 9, escalating *only* the 14B's disagreement residual to a frontier reasoner (DeepSeek-R1, the pre-registered alternative to Opus) takes the system from 85.0% to 94.4%, +9.4 (95% CI [+6.6, +12.4]): R1 solves 88.6% of the 211 problems the 14B votes *wrong-or-split* on (66.4% there on its own). The escalation fraction is larger here (42%) precisely because math has no deterministic oracle to absorb the easy majority first—as on SSMR-Bench, where the music21 oracle clears 70% and the frontier sees only the 4.9% residual—but the gate still concentrates the frontier spend where the base errs: 71 of the 14B's 75 errors fall in the 211-item disagreement slice. All numbers reproduce from the committed per-sample files (the R1 residual text is cached: `math_cascade.py`, `r1_math.py`), no oracle, and—for H1/H3/H4—no new generation.

The gate is the parameter-free operating point of a confidence router—no more, and we pre-registered the test that says so. At its budget the gate escalates the *identical* set as a self-consistency-confidence router (ranking the residual by majority vote-fraction traces 83.2 → 84.6 → 85.0 at B=10% → 30% → 44% of calls; the gate's threshold-free point, 85.0 at 44%, sits where that frontier saturates, since unanimous answers are 99.3% correct), so the binary partition loses nothing to the graded confidence. Whether it *beats* off-the-shelf cascades we tested prospectively on MMLU-Pro (pre-registration frozen before the run; `strictdom_analysis.py`): the agreement signal is the best of the cheap signals (frontier AUC 63.7 vs. 62.3 for sequence-logprob and 62.7 for random) and beats token-level confidence (+1.64, 95% CI [+0.30, +3.02]), but it does *not* strictly dominate—it is not pointwise above sequence-logprob, and its full-frontier margin over random is non-significant (+1.28, CI [−0.03, +2.60]). By the frozen rule we therefore do *not* claim the router beats off-the-shelf cascades: the agreement partition is the label-free, *parameter-free* instance of selective prediction (Geifman & El-Yaniv, 2017) and confidence routing (Chen et al., 2024a), and the contribution is the diagnostic, not a dominant method (`math_confrouter.py`).

Against the confidence routers people use (pre-registered). A second registration (`bench/strictdom/PREREGISTRATION_confbaselines.md`, frozen on 2026-09-17 before any pass; run on 2026-09-23 through an engine-compatibility shim that leaves the registered script unchanged) compares the gate with three standard routers on the same regenerated samples: (a) answer-token probability, (b) P(True) (Kadavath et al., 2022) and (c) verbalized 0–100 confidence (Xiong et al., 2024), on MMLU-Pro and MATH-500 at 7B (the 7B→14B cost-accuracy frontier) and 14B (selective prediction), 12 cells, frozen beat/tie/lose rule on a paired bootstrap of the AUC difference. *No cell loses.* On MMLU-Pro at 7B the gate ties all three (gate − (a) $+0.38$ $[-0.62, +1.45]$, (b) $+0.85$, (c) $+0.97$) and also ties a random escalator ($+1.10$ $[-0.22, +2.48]$); in the other nine cells it beats all three with pointwise dominance (MATH 7B: $+1.12$, $+1.27$, $+1.61$, all CIs above zero; 14B: $+3.5$ to $+16.3$). Each of the three baseline predictions holds in three of its four cells: (a) was predicted to tie on MMLU-Pro and be beaten on MATH, but the gate also beats it on MMLU-Pro at 14B ($+7.40$ $[+3.77, +10.88]$, pointwise, which the registration did not expect); (b) and (c), predicted to be beaten everywhere, tie on MMLU-Pro at 7B. Unparsable verbalized replies: 11, 144, 0, 50 (MMLU-Pro 7B, 14B; MATH 7B, 14B); the gate's frontier on the regenerated samples is close to the cached one (63.7 vs 64.4 on MMLU-Pro, a 0.7-point resampling shift that does not enter the paired comparisons, which all use the regenerated samples; 84.5 vs 84.5 on MATH). So the gate is on par with or better than these routers at no extra call, and dominant over none of them on MMLU-Pro at 7B (every registered quantity, including AUCs, frontiers and matched-budget accuracies: `bench/strictdom/conf_baselines_results.json`).

Re-derivation compared with aggregation (H-RD1, pre-registered). On the 7B's 219-problem disagreement residual, a majority of 16 samples scores 57.5%; self-refine and synthesis-from-candidates each score 57.1%. Both differences are -0.5 points, with 95% intervals $[-4.6, +3.2]$ and $[-5.0, +4.1]$, respectively. Against a fresh $K{=}8$ vote (`aggregate_base`, 53.9%), self-refine gains $+3.2$ $[+0.0, +6.8]$ and synthesis the same $+3.2$ points, while simply aggregating more gains $+3.7$ $[+0.0, +7.3]$; the lower bounds touch zero. H-RD1 is confirmed under its frozen rule: neither same-model arm significantly beats simple aggregation. The registration called this confirmation of a law, but these intervals establish no advantage for either intervention over aggregation, and the comparisons do not match aggregation width. The already-measured stronger-model routing gain is the registered contextual anchor H-RD2, not a new mechanistic test. Sources: `math_rederive.py`, `math_rederive_stats.py`.

I MMLU-PRO RESIDUALS, CROSS-FAMILY ROBUSTNESS, AND FDR CONTROL

A note on wording. This section and its subsections report the registered re-derivation tests in the terms their registrations fixed, including "the law" and its registered outcome wordings. Later controls (E4–E7, Appendices O, N, R, V) narrowed that claim, and Appendix V states what survives: whether re-derivation helps or hurts remains open after five registered attempts (E3–E7). No law is claimed. The registered re-derivation tests are labelled E1–E7: E1, strong re-generation arms (Appendix I.1); E2, the $k{=}5$ extension (Appendix I.2); E3, cross-family (Appendix P); E4, symmetric debate (Appendix O); E5, matched evidence (Appendix N); E6, its disambiguation (Appendix R); E7, matched aggregation width (Appendix V).

MMLU-Pro: a second externally-authored benchmark, non-math (H-MP1, H-MP2). So the cross-domain claim does not rest on a single external benchmark, we pre-registered a fourth domain before any run: MMLU-Pro (Wang et al., 2024b), a hard, externally-authored, deliberately *non-math* ten-option benchmark (a fixed-seed 600-question sample over 13 non-math categories, scored by the answer key—no in-loop prover). *The partition replicates* (H-MP1): unanimous self-consensus is held-out-correct 83.8% (14B, 320) and 88.4% (7B, 233) against 44.6%/42.2% on disagreement. *The oracle-less gate transfers* (H-MP2): the 7B→14B router reaches 65.5%, $+5.3$ over the 7B (95% CI $[+2.2, +8.5]$), matching the 14B at 61% escalation and beating a same-budget random escalator ($+2.1$). As on MATH-500 the transfer is accuracy at a matched 14B-call budget, and here it is not cheaper: at 61% escalation the router costs $1.13\times$ a pure-14B run once the 7B samples are counted; both pre-registered hypotheses hold on a second, non-math, externally-authored benchmark (`mmlupro_run.py`, `mmlupro_analysis.py`).

Re-derivation pool	k	pooled	DL p	KH p	LODO pooled	LODO KH p
Frozen coarse taxonomy (all non-aggregation arms)	18	-1.6	0.579	0.595	$[-5.3, +1.7]$	$[0.04, 0.83]$
Strictly re-generating (tournament/debate/self-refine)	6	-10.8	0.002	0.028	$[-13.2, -7.5]$	$[0.02, 0.16]$
Score arms → re-derivation, all domains	12	-4.6	0.089	0.126	$[-5.8, -2.4]$	$[0.11, 0.45]$
Score arms → evidence, all domains (= row 2)	6	-10.8	0.002	0.028	$[-13.2, -7.5]$	$[0.02, 0.16]$
Paper's class (`redv` in `meta_rederive.py`)	8	-9.6	0.001	0.013	$[-11.6, -6.5]$	$[0.01, 0.08]$
Pre-registered external pool (bootstrap SE)	6	-1.7	0.081	0.074	$[-2.3, -1.2]$	$[0.04, 0.18]$
All re-derivation cells (POST-HOC)	11	-6.9	0.002	0.011	$[-9.0, -4.1]$	$[0.01, 0.05]$
Row 5 by domain, $\bar{s}/\sqrt{m}$ (as committed)	4	-9.2	0.022	0.102	$[-12.4, -6.1]$	$[0.08, 0.28]$
Row 5 by domain, $\bar{s}$ (fully dependent)	4	-9.1	0.041	0.113	$[-13.8, -5.5]$	$[0.05, 0.30]$
Row 7 by domain, $\bar{s}/\sqrt{m}$ (POST-HOC)	5	-7.6	0.024	0.077	$[-9.5, -4.7]$	$[0.08, 0.19]$

Table 12: **Classification sensitivity of the pooled re-derivation effect** (points on the disagreement residual; DerSimonian–Laird random effects, normal-z (DL) and Knapp–Hartung small-k (KH) p; LODO = range over leave-one-domain-out re-pools). Rows 1–5 and 7–10 use the unpaired two-proportion SE of `meta_rederive.py`; row 6 the paired-bootstrap SE of its pre-registration. Score-selection arms = confidence-weighted vote (music, logic), judge-score argmax and public-count rerank (Verilog). Rows marked POST-HOC add the three external self-refine cells generated after the classification was frozen. $\bar{s}$ = mean arm SE within a domain, m = arms per domain. Rows 2, 4, 5 and 7 contain debate arms that E4 later withdrew from the registered re-derivation comparison (Appendix O); logic's re-generation arms are all debate, and dropping logic from row 5 gives -6.5 (KH p=0.054), a sensitivity, not a replacement estimate.

Cross-family partition (post-hoc). Re-running the MATH-500 partition with a *different* family, Llama-3.1-8B-Instruct, reproduces it cleanly: unanimous agreement is 100% correct (24/24) against 36.8% on disagreement (476)—a weak model (39.8% overall) with correct unanimous answers on this small observed subset. This replicates the agreement signal beyond Qwen (`math_run.py`, tag `llama31_8b`).

Multiple-comparison control. Across the partition-direction tests of the externally-checked and externally-authored domains—MATH-500 and MMLU-Pro at both sizes, deductive logic at both sizes, and the cross-family replication (m=7)—every test survives a Benjamini–Hochberg FDR correction at q=0.05 by a wide margin (gaps $+14$ to $+63$ points; one-sided two-proportion p from 7×10^{-33} to 1×10^{-7}; `cross_domain_fdr.py`). Two disclosures. The cross-family (Llama) test was run post hoc, not registered; four of the seven are registered hypotheses (MATH-500 H1, MMLU-Pro H-MP1), and the two logic tests, on the original n=600 run, are specified in advance in that registration's setup without a decision rule. And the family we report is not the one H-MP3 registered, which was Verilog, logic, MATH-500 and MMLU-Pro: we swapped Verilog for the Llama test. The registered family, with Verilog's per-question test (p=3.9×10^{-6} at 7B, 5.3×10^{-6} at 14B; Appendix F), also survives in full. With p this small the correction adds little information either way. The re-derivation arms are *not* uniformly so: only the music-14B tournament (-16.0), the two logic debate arms (-9.4, -25.6 on the n=600 run; -12.7/-33.0 at n=1500, Appendix G) and the MATH-500-14B self-refine cell (-5.2, CI $[-9.5, -0.9]$) are individually significant; the Verilog arms (n=50, ±19 points on the two-proportion scale of Table 5), the music-7B tournament, and the other three self-refine cells are individually within noise; with the strong arms, seven of the eight externally-authored cell×arm effects are individually significant (Appendix I.1).

The original re-derivation family and its sensitivities. The claim this family was declared for was *aggregate, don't re-derive*, so we declare a single family of all 29 tests (the seven partition tests plus every intervention arm across music, Verilog, logic, and MATH-500) and Benjamini–Hochberg-correct it jointly (`meta_rederive.py`; 14/29 survive, including all seven partition tests: four registered, two specified in advance in logic's setup, one post-hoc). The law as *registered* was the *re-derivation axis*—retry, debate, pairwise re-pick, self-revision: the arms that re-*generate* the answer. A DerSimonian–Laird random-effects pool of those eight arms is -9.6 points (p=0.001); the pure debate/tournament core is -13.2 (p<0.001). These are the arm definitions and pooled numbers as originally registered; the debate arm was later withdrawn from the registered re-derivation comparison (Appendix O). Both of logic's arms in that pool are debate, so logic has no surviving re-generation arm; dropping it gives -6.5 (KH p=0.054), which

we report as a sensitivity and not as a replacement estimate. Appendix V makes no claim that re-derivation hurts. We do *not*, however, overclaim a dependence-robust pooled significance, and we are explicit about three limits (all in the same script). (i) *Dependence and small k.* Collapsing the three shared-residual Verilog arms and the 7B/14B size-pairs into four domain-level studies (arm SEs averaged within a domain and divided by $\sqrt{m}$; treating them as fully dependent instead gives -9.1, KH p=0.113) yields -9.2 points under a normal reference (p=0.022), but with only k=4 studies that reference is anti-conservative: under a small-sample Knapp–Hartung correction the pooled effect is *not* significant ($t = -2.3$, p=0.10), and it is fragile to leave-one-domain-out (drop music $\rightarrow$ p=0.28, drop logic $\rightarrow$ p=0.24). Table 12 re-pools the effect under ten classifications of the arms (`meta_rederive_sensitivity.py`): the negative *direction* survives every full-pool classification (one coarse-taxonomy leave-one-domain-out re-pool reaches $+1.7$); the magnitude and its small-k significance do not, ranging from -1.6 (coarse frozen taxonomy, p=0.58) to -10.8 (strictly re-generating arms), so we report the pooled magnitude as suggestive. (ii) *Taxonomy.* The original hypothesis concerns re-*generation*; the coarse-taxonomy wash (Table 12, row 1) arises because the evidence-*reading* arms (synthesis, confidence-weighting) are heterogeneous ($+4.1$, p=0.25, I^2=79%)—the bounded, music-localized exception we have always flagged. (iii) *Provenance, tested at power.* To remove the power excuse we pre-registered and ran the re-generation arm at scale on two externally-authored benchmarks (MATH-500 and MMLU-Pro, both sizes; combined residual n=1,077; `ext_rederive.py`). Self-refine is negative in *every* cell (-0.5/-5.2/-0.8/-1.8; MATH-7B is the pre-existing cell). The registered H-EP1 pool adds the two Verilog score-selection arms (judge-score, public-count rerank) to these four cells (k=6): -1.7, not significant under a small-k correction (KH p=0.074; Table 12); over the four self-refine cells alone it is -1.6 (p=0.19), a secondary view. A second pre-registered test (`ext_strongarms.py`, Appendix I.1) ran the *strong* arms—debate-refine and pairwise tournament—on the same residuals: all eight cell×arm effects are negative (-2.5 to -27.9, $7/8$ individually significant), the registered domain-level pool is -11.7 (KH p=0.064, CI $[-25.2, +1.7]$) and the pre-declared cell-level pool -12.5 (p=0.015). Per the frozen rule this is a directional null: the strong arms are also negative on externally-authored benchmarks but not significantly so under the small-k reference. The negative estimates under these controls do not identify a re-derivation cost independently of aggregation width. We do not upgrade the verdict on the secondary pools. (Tellingly, the music evidence-reading *boost* reverses here—the synthesis arm pools to -3.0, p=0.03, a post-hoc pool no registration defines, so the music gain does not recur in this pool.) So what is robust is the *partition* (four registered tests, two specified in advance in logic's setup and one post-hoc, all FDR-significant) and the consistent negative *direction* of re-generation in every domain without an executable checker (MBPP+ is the disclosed exception, Appendix J); the pooled *magnitude* is significant only at the (dependent) arm level, and we report it as suggestive, not as a dependence-robust law.

I.1 E1: STRONG RE-GENERATION ARMS ON EXTERNALLY-AUTHORED RESIDUALS (PRE-REGISTERED)

The powered test above used the paper's *canonical* arm, self-refine. The arms that carry the large self-built negatives are different and stronger: debate-refine (logic, -9.4/-25.6) and the pairwise tournament (music, -9.0/-16.0). We pre-registered (`PREREGISTRATION_extpower_strongarms.md`, frozen before any GPU call) running both, with their prompts adapted only in the answer-format line, on the same MATH-500 and MMLU-Pro K=8 disagreement residuals (n=219/211/367/280), against the identical registered control `aggregate_more` (majority of 16; the arms spend strictly fewer calls). Primary hypothesis H-SA1: the domain-level DerSimonian–Laird pool with a Knapp–Hartung reference over k=3 studies (MATH-500, MMLU-Pro, and the existing Verilog tournament, -3) is negative; the frozen decision rule counts the result as ceiling-lifting only if the KH interval excludes zero *and* $|\text{pool}| \geq 4$.

Result (Table 13). All eight cell×arm effects are negative against the registered control `aggregate_more` (which spends *more* than the arms, so it is not a "matched" one—see below), seven of the eight individually significant, and H-SA2 (every arm ≤ 0 against the vote) holds in all but one comparison (MMLU-Pro 14B tournament, $+0.7$). The registered primary pool is -11.7 points, KH 95% CI $[-25.2, +1.7]$, p=0.064: the interval includes zero, so by the frozen rule the outcome is a *directional null*: the estimates are negative, but with three domain-level studies the interval remains consistent with zero (the Verilog tournament, -6.0 pp—-3 of 50 groups—is

Cell	n	agg. more	debate Δ (SE)	tournament Δ (SE)	vs. vote
MATH-500 7B	219	62.1	-27.9 (3.7)	-6.4 (2.7)	-23.3 / -1.8
MATH-500 14B	211	68.7	-22.7 (4.0)	-9.5 (2.4)	-17.1 / -3.8
MMLU-Pro 7B	367	42.0	-13.4 (3.7)	-6.0 (2.6)	-11.4 / -4.1
MMLU-Pro 14B	280	48.6	-11.4 (4.1)	-2.5 (2.7)	-8.2 / $+0.7$

Table 13: **Strong re-generation arms on externally-authored residuals.** Accuracy of the registered aggregation control and the two strong arms' differences from it (points; paired-bootstrap SE), with the same differences against the plain K=8 vote. The K base texts are regenerated under the frozen prompt and seed, so this control differs by a point or two from the `ext_rederive.py` cells on the same frozen residual. `bench/extpower/extstrong_*.json`.

the third study; the registration's parenthetical wrote "-3 pp" while the committed script always pooled the accuracy difference, an imprecision in the registration text, not a change of analysis). All eight pre-declared secondary pools, as registered. Against the registered control: cell-level k=4, -12.5 $[-20.3, -4.7]$, p=0.015; debate alone -18.9 $[-31.3, -6.5]$, p=0.017; tournament alone -6.3 $[-10.9, -1.6]$, p=0.023. Against the plain vote: domain-level -8.4 (p=0.052), cell-level -8.6 $[-14.6, -2.7]$ (p=0.019), debate alone -14.9 $[-25.5, -4.4]$ (p=0.020), and tournament alone -2.5 $[-6.0, +0.9]$ (p=0.100)—the one pool that is a null, and the arm it concerns avoids the asymmetric debate display, though not the aggregation-width confound, so we print it here rather than leave it to the script. The registration states that secondary pools cannot upgrade the verdict, and we do not: the claim that the self-built magnitude transfers at the registered significance is *not* made. What the test does establish is that the near-zero external result of the self-refine arm was a property of that arm, not of the domains.

The registered control is a *more*-compute control, and what changes without it. `aggregate_more` spends eight extra sampled calls; the arms spend one (debate) or $d-1$ greedy calls (tournament, mean 1.7–2.8). The label "matched-compute" was inherited from the self-refine test, where the arm did spend eight, and is wrong here, so we drop it and report the genuinely matched comparison from the same cached samples (`compute_matched_control.py`): debate against a majority of nine is -25.1/-19.0/-12.3/-7.5, and the tournament against a majority of $8+(d-1)$ is -5.5/-7.1/-7.1/$+0.4$. Seven of the eight effects remain negative, the eighth is $+0.4$, and the k=3 domain-level pool becomes -10.0 (KH CI $[-21.6, +1.6]$)—the same directional null, from a control that no longer flatters aggregation.

The debate display limits interpretation (post-hoc diagnostic). The prompt shows a bare majority answer but a dissenting sample's full reasoning. The arm leaves the majority on 61–69% of items, picks the displayed dissenter on 15–31%, and a third answer on 31–53% (`debate_anchoring.json`). This describes answer movement; it does not isolate its cause. E4's symmetric-display control later failed to confirm either registered hypothesis, so the debate arm is withdrawn as evidence for a re-derivation cost (Appendix O).

I.2 E2: TWO FURTHER EXTERNALLY-AUTHORED DOMAINS, THE REGISTERED k=5 TEST

E1's primary was a k=3 pool whose Knapp–Hartung reference has two degrees of freedom (t^*=4.303); its interval was wide because there were three studies, not because the effects were small. We therefore pre-registered (`PREREGISTRATION_extk5.md`, frozen before any GPU call on the new domains) a k=5 extension on two further externally-authored benchmarks, chosen to cross the two confounds the evidence still had: **ProofWriter** (Tafjord et al., 2021) (the 600-item depth-5, open-world sample of Pan et al. (2023), 200 per label, whose items match rows of the tasksource depth-5 validation split; deductive logic over a rule base with a 3-way answer space, the externally-authored analogue of our self-built Datalog domain) and **ARC-Challenge** (Clark et al., 2018) (a seed-0 sample of 600 four-option science questions). Because this is a *second look* at the same hypothesis, the registration fixed the primary α at 0.0294, the Pocock boundary for two looks: $0.0294 < p < 0.05$ would *not* count. Because E1 was itself tested at 0.05, the two-look procedure is not an exact overall-0.05 Pocock design, and the E2 intervals below are 97.1% Knapp–Hartung intervals at that α. E1's verdict is not revised.

Cell	residual n	vote	agg. more	debate Δ (SE)	tournament Δ (SE)
ProofWriter 7B	470	61.9	65.5	-30.9 (3.3)	-7.2 (2.2)
ProofWriter 14B	298	58.1	58.4	-14.4 (4.3)	-6.7 (2.7)
ARC-Challenge 7B	99	63.6	65.7	-25.3 (6.6)	-2.0 (3.5)
ARC-Challenge 14B	52	69.2	75.0	-28.8 (9.6)	$+0.0$ (6.7)

Table 14: **The k=5 extension.** Same frozen arms, control and residual rule as Table 13; the ARC residuals are small because its agreement partition is sharp (83.5% and 91.3% of items are unanimous, and the vote is 97% accurate there), and the registration takes whatever residual the frozen rule yields.

Result. Seven of the eight new effects are negative and the eighth is 0.0. The registered primary—domain-level DerSimonian–Laird with a Knapp–Hartung reference over k=5 studies (MATH-500 -16.6, MMLU-Pro -8.3, ProofWriter -14.8, ARC-Challenge -14.0, Verilog -6.0)—is $\mathbf{-13.1}$ points, KH 97.1% CI $[-18.8, -7.4]$ at α=0.0294, $t = -7.55$, τ^2=12.3. The interval excludes zero and $|\text{pool}| \geq 4$, so by the frozen rule the outcome is *confirmed at power*. This confirms the contrast against the registered more-sample control; it does not isolate re-derivation from aggregation width. **This pool has since been narrowed.** A later registered control (E4, Appendix O) failed to confirm that the debate arm's harm survives a symmetric prompt, so as pre-registered the debate arm is withdrawn from the registered re-derivation comparison and the surviving comparison is the tournament-only pool, -6.0 $[-9.3, -2.6]$. The -13.1 above is reported unchanged as the E2 outcome, not as the paper's current claim.

We give this confirmation the same scrutiny the failed k=4 pool received (`extk5_pool.py` prints all of it). Leave-one-domain-out: dropping MATH-500, MMLU-Pro, ProofWriter, ARC-Challenge or Verilog gives -11.9, -15.4, -12.4, -12.9, -13.4, every interval still excluding zero. Per arm, both survive separately: debate -20.0 $[-31.3, -8.7]$ and tournament -6.0 $[-9.3, -2.6]$, so the pool is not carried by the arm with the disclosed template asymmetry. Against the plain vote rather than the registered control the primary is -9.9 $[-14.6, -5.2]$ and debate -16.9 $[-26.4, -7.4]$, but *the tournament arm is then a null*: -2.8, CI $[-6.4, +0.7]$. That is the least flattering quantity the registration obliges us to report, and it bounds the claim—against a plain K=8 vote, the re-pick arm is not shown to hurt; the pooled harm we can demonstrate at power is relative to spending the arm's budget on more aggregation instead. The partition also replicates on both new domains (unanimous vs. disagreeing: 93.1/62.1 and 81.5/57.4 on ProofWriter, 97.2/64.6 and 97.1/71.2 on ARC; majority ties broken by first appearance, as for MATH-500 and MMLU-Pro).

A prospective forecast and its limits. The same registration froze a forecast proportional to the base samples' minority-minus-majority accuracy gap, with a rate of leaving the majority transferred from earlier cells. This is a predictive approximation, not an identity for the intervention effect: accuracy conditional on changed answers need not equal the overall candidate accuracy, and a re-generation arm can produce answers outside the candidate set. Both forecast accuracies are computed from the K=8 base samples. Fitting only the four E1 cells gave frozen coefficients (0.5642 and 0.1839 times that gap, bands ± 6 and ± 4), and the eight resulting point forecasts were emitted and committed before each arm ran. *Five of the eight landed inside their band*, which the registration classifies as a first-order account rather than a law, and we report it as such without refitting. Two disclosures about the artifact itself. The predictor's majority tie-break was initially hash-order dependent, so the first committed forecast file was not bit-reproducible; we made the tie-break deterministic (first appearance) and re-emitted, which moves individual point forecasts by under a point and leaves the $5/8$ verdict unchanged—as it is under every random seed we tried. And the arms regenerate the K base texts, so on 13–29% of residual items the regenerated samples are unanimous and both arms return the vote exactly; the forecast's features come from the sampling-time residual while the effects come from the regeneration-time one. This mismatch complicates interpretation; two of the three forecast misses occur in cells with it. The misses are informative and one-sided: the tournament arm's harm is markedly smaller than predicted on ARC-Challenge (-2.0 and $+0.0$ against -6.7 and -8.4), so the capitulation rate is arm- and domain-specific rather than universal. The registered reading remains partial: a first-order predictive account with errors, not an identified mechanism.

J PUBLIC-TEST SELECTION ON MBPP+

MBPP+ (Liu et al., 2023a; Austin et al., 2021) tests the decomposition with a third-party benchmark and checker. Its public example tests select candidates; the additional hidden tests measure correctness. These public tests are not a sound prover, despite that terminology in the original registration. The experiment uses Qwen2.5-7B, $K{=}10$ candidates, and EvalPlus's execution sandbox.

H-CP1 held: the public checker covers a substantial fraction at the registered precision. The implementation commits the most frequent program *text* among candidates passing all public tests, breaking ties by the first occurrence. It does not cluster program behaviors for this commitment. This rule covers $328/378$ items (86.8%) at 85.1% hidden-test precision, against a 69.4% per-candidate correctness rate, a $+15.6$-point difference. Those are different units of evaluation; the latter is not a matched system baseline. The coverage and precision meet H-CP1's pre-registered thresholds. This establishes a useful empirical filter on these tasks, not soundness or general checker coverage.

H-CP2 held vacuously. On the 50 problems with no public-test passer, no existing candidate is hidden-test-correct. Both the registered behavioral-consensus selection and public-test-count reranking therefore score $0/50$, satisfying the frozen non-inferiority sign rule without testing a meaningful selection advantage. A *post-hoc* refine-and-check arm produces $5/50$ hidden-test-correct solutions, against $2/50$ from four further plain samples: $+6.0$ points with a 10.0-point interval half-width (Table 5). This positive, imprecise result concerns new generation with executable checking; it does not identify which component causes the recovery or support a universal claim against re-derivation.

A distinct system-level comparison (post-hoc). The script also groups all candidates by their public-test pass/fail vectors, selecting a representative from the most frequent vector. This is separate from H-CP1's program-text rule. Across all 378 problems, that selector scores 71.4%, below selection by public-test pass-count at 74.3%: a difference of -2.9 $[-4.8, -1.3]$. Thus how the same public-test outcomes are used matters: ranking by pass count beats a majority over pass/fail vectors. Reproduce with `bench/codeproof/codeproof_analysis.py`; the post-hoc generation arm is described in its accompanying result files.

K POWER ON THE DISAGREEMENT CORE: THE HEADROOM DIAGNOSTIC

The post-hoc v1 headroom diagnostic is computed per item by `bench/ssmr/recon_analysis.py` from the vote-recording caches `h2cache_qwen25-14b-recon-det_Test_textual.jsonl` and its 7B twin, so it admits a paired interval; we print it here rather than leave a bare point estimate in a paper that prints intervals everywhere else. For a disagreeing residual item i with k_i distinct candidates, $b_i = \mathbf{1}[\text{gold} \in \text{candidates}_i]$ is oracle-best coverage and $u_i = k_i/4$ is the zero-knowledge null—the chance a uniform guess over the four options lands on one of the k_i candidates. The headroom is the paired mean $\bar{b} - \bar{u}$ and its standard error is that of the per-item differences $b_i - u_i$. ($k_i{=}2$ for 67 of the 79 items at 14B and for 80 of 90 at 7B; $k_i{=}3$ otherwise.)

	n	oracle-best	null	$\bar{b} - \bar{u}$	SE	95% CI (threshold)
14B	79	55.7%	53.8%	$+1.9$	5.62	$[-9.3, +13.1]$ (11.2)
7B	90	61.1%	52.8%	$+8.3$	5.12	$[-1.8, +18.5]$ (10.2)

What the diagnostic does not license. Under this paper's own convention (detection threshold $= t^* \times \text{SE}$ at $\alpha{=}0.05$, i.e. 50% power; 80% power needs 16.0 here) the smallest detectable headroom on these cores is 11.2 points at 14B and 10.2 at 7B. Neither margin clears it: $t = 0.34$, $p = 0.74$ at 14B and $t = 1.63$, $p = 0.11$ at 7B, and a 200,000-resample paired bootstrap agrees ($[-9.2, +12.7]$ and $[-1.7, +18.3]$). So we do *not* claim that these candidate sets are uninformative or that their headroom is exhausted: the 14B interval does not exclude a 13-point selection headroom, and oracle-best coverage on its own is 55.7% with a Wilson interval of $[44.7, 66.1]$. The 7B margin is likewise not a gap, and we report it with its interval rather than as 8.3 points. Driving the threshold down to the observed 1.9-point margin at this per-item dispersion ($\text{sd} = 0.50$) would take about 2,700 disagreement items—$34\times$ this core, or roughly $54{,}000$ benchmark questions at this run's 4.9% disagreement rate—so no run at SSMR-Bench scale can settle it.

The scope of the selection ceiling. The fixed-candidate result does not depend on the null. Oracle-best beats the shipped vote on the same items by $+21.5$ $[+12.3, +30.8]$ at 14B and $+35.6$ $[+25.5, +45.6]$ at 7B; since the core is only 79 of 1,600 questions (90 at 7B), a perfect selector over the model's own candidates is worth $+1.06$ $[+0.61, +1.52]$ points of system accuracy at 14B and $+2.00$ $[+1.43, +2.57]$ at 7B. This is a ceiling only for selecting among these fixed candidates in this v1-based run. It does not constrain new generation, a stronger model, or arms run on different residuals. In particular, a stronger model can correct unanimous wrong answers. Failure to separate candidate coverage from its null also does not establish equivalence to chance.

L THE PRE-REGISTERED TAXONOMY BLIND TEST

The criterion, pre-registered and not confirmed. A taxonomy drawn after the fact predicts nothing, so we froze the criterion and sign predictions for five never-run interventions and ran ten blind sign tests at both sizes. The frozen criterion scored $6/10$, a score a fair coin matches and a constant rule beats ("What $6/10$ is worth", below); its two misses are the arms whose judge re-reads the samples' full *reasoning*: confidence-weighting ($+1.9/+2.9$) and synthesis ($+5.8/+7.9$) helped where it predicted harm. Is that evidence-reading gain robust or music-specific? The further pre-registered domains answer: it is bounded—significantly positive only in music, negative in logic at 7B, within noise in Verilog (synthesis -5 of 50 groups, so the music gain does not recur in code), and reversed on the externally-authored benchmarks (post-hoc synthesis pool -3.0, p=0.03; Appendix I)—while re-generation is negative in both self-built domains.

What $6/10$ is worth. We report the score without inflating it. The ten tests are five interventions run at two checkpoints, and every arm returned the *same* verdict at both sizes (Table 15), so the size axis contributes no independent evidence: the criterion is $3/5$ at 14B and $3/5$ at 7B, exactly the median of a fair coin ($P[X \geq 3]$=0.50 at n=5). Even crediting all ten cells as independent, $6/10$ carries a one-sided binomial p=0.38. And a trivial competitor does better: the constant rule "no intervention falls below -1.0"—predict the permissive aggregation branch for everything—is right in 8 of the 10 cells ($4/5$ at each size), because `tournament` is the only arm that crosses the threshold downward at either size. So the blind test does *not* establish the aggregate-vs-re-decide criterion as a predictor, and we do not use it as one. What we use it for is the one reading it does support: two arms contradict the criterion's strong form, with the same sign and comparable magnitude at both sizes, and that is what motivated the exploratory evidence-vs-verdict refinement and the prospective tests of it in Appendices F and G. The later matched controls, especially E7 (not detectable), likewise do not establish the proposed mechanism (Appendix V).

The criterion, the five interventions, their classifications, and the sign predictions were frozen in a pre-registration committed to the project repository *before* any intervention was implemented beyond its frozen prompt wording or any GPU run started; no intervention was dropped, re-classified, or re-run for this test after results were seen, and the table scores every arm under its frozen class; the later evidence-versus-verdict reading reclassifies confidence-weighting and synthesis post-hoc (Appendix B.4), and this test does not score it. All interventions share one planner pass and the same three solver samples (temperatures $0.0/0.5/0.8$) per run, and are scored paired against that run's own majority-vote baseline on the oracle-abstain residual (n=481; disagreement subsets n=131 at 14B, 166 at 7B).

Three notes. (1) The blind runs use the sweep configuration rather than the shipped harness; their disagreement subsets are broader and easier (selection ceiling $81\%/80\%$ vs. the null 55%), which is what gives synthesis headroom to harvest. On the shipped harness's tighter disagreement core the ceiling is worth about a point of system accuracy ($+1.1$ $[+0.6, +1.5]$; its margin over the zero-knowledge null, $+1.9$ $[-9.3, +13.1]$ at n=79, is not separated from zero—Appendix K), so the shipped system still ends at the plain vote. (2) The paired within-run design makes every Δ immune to that configuration difference. (3) Confidence-weighting is a borderline case for the exploratory refinement: its judge re-reads one sample's reasoning, but the cross-candidate decision is taken over scalar confidence *scores*—a verdict signal under the very rule that classes the Verilog self-score arm (Table 10) as verdict-consuming. Read that way it is a *second* miss for the refinement ($8/10$, not a clean fit), which is part of why we report evidence-vs-verdict as an exploratory hypothesis rather than an established law, and why the Verilog test's label-selected residual limits its interpretation.

Table 15: **Ten pre-registered sign tests.** Δ = accuracy minus the shared vote baseline (points), on the full residual and on the disagreement subset. Prediction rule (frozen): aggregation $\Rightarrow \Delta_{\text{resid}} > -1.0$; re-decision $\Rightarrow \Delta_{\text{resid}} < -1.0$. The frozen criterion scores $6/10$—$3/5$ at each size, with every arm's verdict identical across sizes, so five effective tests at the median of a fair coin (p=0.38 even treating all ten as independent; the constant rule "no arm falls below -1.0" scores $8/10$). The misses are confidence-weighting and synthesis (both re-decisions, at both sizes), which *motivate* but do not confirm the exploratory evidence-vs-verdict reading discussed below.

		14B Δ		7B Δ		prediction	
Intervention	frozen class	resid	disag	resid	disag	14B	7B
vote-of-5	aggregation	-0.4	-1.5	$+4.0$	$+11.4$	✓	✓
planner ensemble	aggregation	$+1.7$	$+6.1$	$+4.6$	$+13.3$	✓	✓
confidence-weighted	re-decision	$+1.9$	$+6.9$	$+2.9$	$+8.4$	×	×
pairwise tournament	re-decision	-4.4	-16.0	-3.1	-9.0	✓	✓
synthesis (MoA pass)	re-decision	$+5.8$	$+21.4$	$+7.9$	$+22.9$	×	×

The genuinely compute-matched control, extended to the full pool. The registered control `aggregate_more` is a majority over $K+8 = 16$ samples, while the strong arms spend one extra call (`debate_refine`) or $d-1$ greedy calls (`tournament`, mean 1.7–2.8). It therefore spends *more* than the arms and flatters aggregation; the label "matched-compute" was inherited from the earlier self-refine test, where the arm did spend eight extra calls. `compute_matched_control.py` recomputes, from the same 16 cached texts on the run host and with no new generation, the control that matches each arm's actual extra budget (maj_T: a majority over the first $8 + (d-1)$ samples). We have extended it from the four E1 cells to all eight cells of the k=5 pool. Per cell, `tournament` against maj_T is $-5.5/-7.1$ (MATH-500 7B/14B), $-7.1/+0.4$ (MMLU-Pro), $-2.8/-5.7$ (ProofWriter) and $-1.0/+3.8$ (ARC-Challenge): six of eight negative, two positive. The domain-level k=5 pool is $\mathbf{-4.2}$, KH CI $[-7.4, -1.1]$ at α=0.0294 and $[-6.9, -1.5]$ at α=0.05, with $\tau^2 = 0.00$. The effect is smaller than against `aggregate_more` and more homogeneous. We report this as a sensitivity analysis: the registered primary remains the pool against `aggregate_more`, which we report unchanged.

What this control does *not* settle, and a ladder that probes it. maj_T matches the arm's extra *budget*, not the number of samples that determine its final answer. We therefore ran, **post hoc and without pre-registration**, a ladder comparing the tournament arm against majority-over-k for a range of k, recomputed from the same run-host cached samples with no new generation (`bench/extpower/ess_ladder.py`). Over the four recomputable domains the pooled deficit is monotone in the control's sample count: $+1.9$ at k=1, -0.8 at 3, -2.7 $[-6.8, +1.5]$ at the plain vote, -4.1 $[-7.9, -0.3]$ at the arm's matched budget, and -5.9 $[-9.9, -1.9]$ at k=16. So the registered deficit is a null at the plain vote and significant only against controls that aggregate more evidence than the arm's verdict does.

Three limits of the ladder, stated because we found them by audit rather than by design. Verilog is excluded: its frozen -6.0 was measured against a different control and cannot be recomputed per rung, so including it would hold a fifth of the pool constant along the swept axis. A "majority of two" is bit-identical to a majority of one under our tie-break, so that rung is omitted. And a control matched to the $d-1$ judge calls alone would *not* be matched: the arm forms its bracket from all eight base samples, so such a control sees strictly less than the arm does—an earlier draft of this analysis made that error and reported a sign reversal that does not exist. We therefore do not claim that re-deriving a verdict is harmless, nor that it is harmful; we claim the registered comparisons are confounded with the evidence the control aggregates. We registered an attempt to separate them (Appendix N); it does not succeed, because its arm is still one verdict against a majority over many.

M THE ON-POLICY RL CONTROL IN FULL

This appendix asks whether on-policy RL buys what distillation does, since the best published method trained for this benchmark trains Qwen3-8B-Base with verifier-shaped GRPO. We sweep the learning rate over a 20× range and the budget from 0.2 to 3 epochs (4,500 steps; same base,

binary reward, LoRA-GRPO; grid, curves and recipe in Appendix B). *No configuration produces an improving signal*: the training reward never trends up (3-epoch OLS slope $+0.06$ against a per-step SD of 0.44, clean terminations, small KL—flat, not collapsed), the matched recipe lands at 46.6% after three epochs ($\approx$ the 46.1% base), and the best of the grid reaches 49.8%, a $+3.7$ gain (post-hoc). The cited 70.9 endpoint was measured under the prior work's protocol: subtracting our base from it does not recover that work's reported gain. Distillation's $+20$ to $+31$ (legacy parser; strict R1-SFT replication: $+12.0/+21.4$ at 7B/14B) is on different Qwen2.5 backbones. At 14B both shaped and binary rewards instead reward-hacked (batch reward up, SSMR-Bench down 1–3 points). Three epochs at the matched rate move the model less than one epoch at a higher rate; this does not isolate budget as a cause. We conjecture, but cannot diagnose, a signal problem: on a residual the model cannot self-verify, the sparse outcome reward offers little to climb—and, consistent with evidence that RLVR gains can stem from clipping bias rather than the reward (Shao et al., 2025) and need not expand the base model's reasoning (Yue et al., 2025), we do not claim GRPO cannot reach 70.9 at full-parameter, multi-GPU scale—only that across a deliberately wide sweep it gains at most $+3.7$ $[+1.1, +6.3]$ on Qwen3-8B-Base (Table 6), while three epochs of LoRA distillation lift the Qwen2.5-7B and -14B bases $+20$ to $+31$ on Table 4's legacy-parser rows; the strict, same-stack replication of R1-SFT gives $+12.00$ $[+8.94, +15.12]$ at 7B and $+21.38$ $[+18.50, +24.25]$ at 14B (a post-hoc contrast of the pre-registered runs, Appendix Y.5). That comparison is *cross-backbone* and not a matched RL-versus-SFT experiment: we ran no Qwen3-8B distillation, so the two legs differ in base model as well as in training objective.

The cause of the modest gain is unresolved. A sparse outcome reward and dense teacher traces provide different training signals, but the experiments do not isolate that difference. The RL base is wrong even on many items the checker can answer; the residual argument alone cannot explain the flat sweep. The evidence is a result for the tested configurations, not a general RL-versus-distillation comparison.

N E5: THE MATCHED-EVIDENCE CELL

Registered in `bench/matchedev/PREREGISTRATION_matchedev.md` (frozen commit `f5b4547`) and amended once, before any result was inspected, at `5c78050`; the framing for every possible outcome was fixed in advance at `e6665c4`.

Why it was amended. The design was first frozen at $k{=}2$. We had not computed its power. Using the four published E4 cells' paired standard errors as pre-data information, a $k{=}2$ Knapp–Hartung pool ($t^* = 12.71$) has a detection threshold near 23 points (25 at 80% power) against effects of two to seven: it was uninterpretable by construction. We amended to $k{=}2 \to 4$ and, this time, printed the power first. This is the third registered test in this project filed without a power calculation—E1 returned a directional null at $k{=}3$, and E4's detection threshold of 18.1 exceeded its own 11.0 criterion—and we state the pattern rather than let a reader find it. The superseded design's one completed cell was moved aside unread and every analysed number comes from a run launched after the amendment.

Design. For each residual item the arm sees the question and all sixteen cached traces (cache order, each truncated to 800 tokens), then answers in one greedy call: `max_model_len` 16384, `max_tokens` 1536, temperature 0, seed 0. Its control is the majority over those same sixteen parsed answers. The arm therefore sees *strictly more* evidence than the control, which makes the test *a fortiori*. A secondary arm sees only the sixteen answers—exactly the control's evidence—and is reported either way. Cells: MATH-500, MMLU-Pro, ProofWriter, ARC-Challenge at 7B, on the frozen residuals ($n = 219/367/470/99$); every cell cleared the registered 90% parseability gate.

	MATH-500	MMLU-Pro	ProofWriter	ARC
`rederive_all16` (all traces)	-5.5	$+1.4$	-3.6	-6.1
`rederive_answers16` (answers only)	-8.2	-7.1	-9.4	-3.0

Verdict. The primary arm pools to $\mathbf{-3.0}$, KH CI $[-8.5, +2.6]$, $\tau^2 = 7.79$. The realised detection threshold is 5.5 points (7.2 at 80% power), so by the registered rule we report the result as **not**

detectable at this k, never as a null, and the identifying question is open after three attempts. We note that our amendment predicted a threshold near 4.1 by assuming $\tau^2 \approx 0$; the realised heterogeneity made the test less powerful than we had projected, which is a fourth instance of the same optimism and is why we now print the realised threshold beside every estimate.

The secondary arm—shown exactly the aggregator's evidence and nothing more—pools to $-\mathbf{7.4}$, KH CI $[-11.3, -3.5]$, with $\tau^2 = 0.00$ across all four domains and its realised detection threshold at 3.9. We report it as registered and do *not* promote it to primary, which our pre-committed framing forbids precisely because it is the more favourable of the two.

Read together the arms say that at matched evidence re-derivation costs something, and that supplying the reasoning behind the answers roughly halves that cost. That is consistent with the evidence account of the registered deficits, and the registered primary does not decide between it and the alternative that re-derivation carries an independent cost the traces partly repair.

O E4: SYMMETRIC-TEMPLATE DEBATE CONTROL

The objection, and what was frozen. The `debate_refine` prompt shows the model its majority answer as a bare string but the dissenting sample's *full reasoning trace*. The measured harm might therefore reflect one-sided evidence *display* rather than re-generation. Before any symmetric-template generation existed we froze a control identical in every respect—same base prompt, same dissenter selection, same single greedy call, same `max_tokens`, same seed, same scoring, same residual, same control—except that the majority side also shows its full trace, taken from the first base sample whose parsed answer equals the majority. Cells: MATH-500 and ProofWriter at 7B and 14B, frozen residuals $n = 219/211/470/298$. Decision rule: $D_{\text{sym}} < 0$ with its Knapp–Hartung interval excluding zero (H-SY1), and $|D_{\text{sym}}| \geq 0.5|D_{\text{asym}}|$ (H-SY2), k=4 cells at $\alpha = 0.05$.

cell	n	`debate_sym`	`debate_refine`	paired diff.	McNemar χ^2
MATH-500 7B	219	-13.7	-32.9	$+19.2$	28.0
MATH-500 14B	211	-4.3	-18.0	$+13.7$	13.3
ProofWriter 7B	470	-10.2	-28.7	$+18.5$	41.8
ProofWriter 14B	298	$+12.1$	-8.1	$+20.1$	32.8
pool (k=4)		-4.1	-22.1	$\mathbf{+18.1}$	
KH CI		$[-22.2, +14.0]$	$[-39.7, -4.5]$	$[\mathbf{+14.0}, \mathbf{+22.2}]$	
τ^2		107.1	96.9	**0.00**	

Verdict, in the registration's own words. Neither hypothesis holds: D_{sym}'s interval includes zero, and $|D_{\text{sym}}| = 4.1$ against a threshold of $0.5|D_{\text{asym}}| = 11.0$. As pre-written, "we report that the debate arm's harm *does* depend on the one-sided display, drop 'debate' from the arms that support the law in the abstract and body, and re-pool the k=5 primary without the debate arm." We have done so; the tournament-only pool, -6.0 $[-9.3, -2.6]$, is reported in Appendix I.2; it is a null (-2.8 $[-6.4, +0.7]$) against `aggregate_base`. It pools Verilog's tournament effect as a magnitude although Appendix F treats Verilog's arms as sign-only; without Verilog it is -5.9 $[-10.8, -0.9]$ at the registered α=0.0294, and -2.7 $[-7.7, +2.4]$ against `aggregate_base`, so nothing turns on it.

This is a failure to confirm, not a refutation. At $\tau^2 = 107$ the Knapp–Hartung standard error is 5.69, so at k=4 and $\alpha = 0.05$ the detection threshold is $3.182 \times 5.69 = 18.1$ points, and 80% power would need 23.7—both *above* the 11.0 that H-SY2 itself demanded. The registered test could not have confirmed persistence unless symmetric harm exceeded roughly 18 points. We report the registered verdict and act on it, and we do not claim the control established that symmetric debate is harmless.

What the control did establish (secondary, not pre-specified). The registration asked whether the harm *depends* on the display, which is a difference rather than a level, and that difference is the best-behaved quantity we measure: within run, on the same base samples and matched item sets ($186/175/412/239$ keys), symmetric display recovers $+18.1$ points over the one-sided template, KH CI $[+14.0, +22.2]$ (`symdisplay_effect.py`), with τ^2 exactly 0.00 and all four cells inside

$[+13.7, +20.1]$. Per-item, McNemar is individually significant in $4/4$ cells (all $p < 0.001$) on 310 versus 92 discordant pairs. We report this as a secondary estimator that was not pre-specified; it does not rescue H-SY1 and does not re-enter debate into the k=5 pool. We also do not report a percentage of harm "explained": that would divide two point estimates whose uncertainty is not propagated, and D_{sym}'s interval is consistent with a wide range.

The one reversing cell is a control-drift artifact. ProofWriter 14B is the only cell whose *level* reverses ($+12.1$). On identical items, `aggregate_more` itself moved between the E1/E2 and E4 runs: -6.2 points at MATH-500 14B, -2.7 at ProofWriter 14B, $+0.9$ at MATH-500 7B and exactly 0.0 at ProofWriter 7B. The reversing cell is one whose control dropped, and its paired contrast ($+20.1$) is in line with the other three—which is why the paired estimator is homogeneous where the levels are not.

Provenance and two disclosures. (i) The E4 cells ran at `max_num_seqs=` 16 and `max_model_len=` 4096, against 32 and 12288 for E1/E2. vLLM's continuous batching is not reproducible across batch sizes even at a fixed seed, so the same cell's arms differ between runs (`debate_refine` at MATH-500 7B is -27.9 there and -32.9 here). Every quantity above is therefore computed *within* the E4 run; the original-run pool (-24.3) is reported here only for comparison and is not used in any test. (ii) The same run independently re-ran the surviving arm: tournament is $-9.6/-4.7/-10.0/-4.4$, negative in $4/4$, pooling to -7.3 $[-12.2, -2.5]$ against E2's -7.5 on these cells from different samples. The tournament judge is symmetric by construction—it is shown the two candidate *answers* and no solution text for either side—so the display objection never applied to it.

Scope of the debate withdrawal. E4 tested the symmetric template on MATH-500 and ProofWriter. The self-built logic domain's debate arm (`bench/logic/logic_run.py`) uses the *same* one-sided template—the majority as a bare answer, the dissenter's full trace—so the confound E4 identifies applies to it as well and we withdraw its debate arm on the same grounds, without having run the symmetric control there. The logic partition and its sound-oracle prove leg are unaffected.

P E3: CROSS-FAMILY TEST

Every re-generation cell elsewhere in this paper uses Qwen2.5-Instruct, so "this is a Qwen2.5 post-training artifact" was an open alternative explanation. We pre-registered a test that changes *only* the model: Llama-3.1-8B-Instruct on the MATH-500 disagreement residual already fixed by the committed June sample file ($n = 476$), with arms, control, prompts and decoding byte-identical to E1/E2.

H-CF1 (primary) holds. Both strong arms are negative against `aggregate_more`: `debate_refine` -26.1 and `tournament` -17.4. In the registration's words, "the law is not a Qwen artifact: on a different model family, on the same externally-authored residual and with the same frozen arms, both strong re-generation arms are negative."

H-CF2 (secondary) holds at the tightened α. Recomputing the k=5 pool with the MATH-500 domain effect replaced by the mean of its Qwen and Llama cells gives -13.5, KH CI $[-21.5, -5.6]$ against `aggregate_more` and -10.5 $[-17.0, -4.0]$ against `aggregate_base`, both excluding zero at $\alpha = 0.0167$ (a Bonferroni correction, $0.05/3$, for a third look at the same pooled hypothesis; a three-look Pocock boundary would be about 0.022).

H-CF3 (exploratory) fails, $0/2$. The frozen mechanism forecast, whose coefficients were fitted on Qwen cells only and are *not* refitted here, predicted -15.1 ± 6.0 for debate and -4.9 ± 4.0 for tournament; the actuals are -26.1 and -17.4. Both fall outside their bands and both under-predict harm. Within Qwen the same forecast placed five of eight effects inside its bands; across families it places neither. We therefore report the account as first-order and *family-specific*: it gets the sign right and the magnitude wrong off-family. No decision hangs on H-CF3, and we do not refit it.

Q THE MATCHED OUT-OF-DISTRIBUTION CONTROL

This post-hoc study evaluates MusicTheoryBench (Yuan et al., 2024b) (367 music-theory MCQs) and WildScore (Mundada et al., 2025) (807 musicological MCQs, text-only track), both held out of training. Neither v1 nor SYN fires on any of these items. Other music systems include CSyMR (Wang et al., 2026), which combines a ReAct controller with `music21` operators on human-authored questions.

What was wrong with the comparison we had. An earlier analysis read the harness as repaying the specialization tax, on the strength of WildScore rising from 38.3 (trained model alone) to 52.8 (trained model plus harness) against a 55.4 base. That base was evaluated *without* the harness. Since the SSMR-Bench oracle never fires out of distribution, the harness's remaining components—planner, mixture-of-agents vote and `music21` analysis block—are generic and help any model, so the comparison charges the training with a deficit measured against a weaker configuration than it runs in. The control is to give the base the same harness.

Result. We ran the untuned Qwen2.5-7B and -14B *Instruct* bases through the identical harness-v2 script that produced the trained rows (`eval_wildscore_harness2.py`, `eval_mtbench_harness2.py`; `use_verifier` off, as in Table 4). The harness helps the base in three of four cells, and the trained recipe is behind the harnessed base in all four.

Benchmark	Size	base	+ harness	trained + harness	unmatched Δ	matched Δ
WildScore (n=807)	7B	47.0	50.2	43.6	−3.3	**−6.6**
WildScore (n=807)	14B	55.4	56.1	52.8	−2.6	−3.3
MusicTheoryBench (n=367)	7B	57.2	55.9	53.1	−4.1	−2.7
MusicTheoryBench (n=367)	14B	55.3	58.9	54.2	−1.1	−4.6
pooled					−2.9	**−4.6**

Harness version, disclosed. All four rows use harness v2 (`harness2.py`, the second code version of the v1-based Stage C harness; it calls checker v1, not checker v2), which has no retry or re-ask: its result files record a `use_multiturn` flag inherited from harness v1, but v2 never reads it (`harness2.py`), and the headline run's items are resolved only by the oracle or the vote. The trained rows were first measured in May under harness v1 (WildScore 51.4 at 14B and 46.7 at 7B; MusicTheoryBench 59.7 and 53.7) and rebuilt under v2 in September, before the base rows were run; v1 put the 14B recipe higher on MusicTheoryBench than v2 does (59.7 against 54.2), and against the harnessed base's 58.9 that one cell would flip sign. Comparing a v1 recipe against a v2 base would mix harness versions, so we report v2 on both sides; the result is harness-version sensitive in that cell, and the matched comparison is post-hoc in any case.

Inference. The four paired differences are not independent: each benchmark item contributes once at 7B and once at 14B. We therefore pool with a 2,000-resample bootstrap that resamples *item identities* within a benchmark and carries both sizes of a drawn item together, giving -4.6 $[-6.7, -2.5]$, $p<0.001$ (the resolution floor of 2,000 resamples). Per cell, McNemar is significant only for WildScore 7B (χ^2=12.2, p=0.0005; 137 versus 84 discordant); the other three are directional (p=0.08, 0.32, 0.07). We therefore claim the pooled effect and the $4/4$ sign, not per-cell significance.

What this does and does not change. It does not touch the in-distribution result: training still adds +5.2 over the harnessed base on SSMR-Bench (88.9 → 94.1 at 14B). It changes the attribution out of distribution. The matched deficit is *larger* than the unmatched one (-4.6 against -2.9), so the unmatched comparison flatters the recipe, and it is larger in three of four cells. Averaged over the three benchmarks the untuned base under the harness (68.0 at 14B) beats the full recipe (67.0), and the one configuration above both is also untuned: the base with the frontier-written checker (69.2, Table 4). That margin comes entirely from SSMR-Bench, where the checker fires: it fires on none of the 367 MusicTheoryBench and 807 WildScore items (Appendix X.2), so there this row equals the unharnessed base (55.3, 55.4) and trails the harnessed base (58.9, 56.1). Reproduce with `bench/ood_matched.py`, which reads only committed result files.

R E6: DISAMBIGUATING THE MATCHED-EVIDENCE DEFICIT

Registered before it ran, with its power computed first. Three reviewers converged on the same objection to E5: an arm that loses to a majority may be losing because it *re-derives*, or merely because it decides from one sample while the control aggregates sixteen. Every re-derivation comparison in this paper is confounded that way. E6 was frozen at commit `ca6e845`, with the analysis script committed before the first cell finished generating and written to refuse a subset of cells, so no effect could be computed early. Unlike three earlier registrations in this project, its power was computed *before* the design was fixed: pooling over items with a 2,000-resample domain-cluster bootstrap rather than a $k{=}4$ Knapp–Hartung reference gives a detection threshold of 3.3 points (4.7 at 80% power) against 3.9–5.5, and the effects at issue are 3–8. The realised threshold is 4.0 (5.7 at 80%). All four cells returned exactly the registered residuals (219/367/470/99, $n{=}1{,}155$).

The primary arm was meant to hold aggregation width equal, and does not. `rederive_answers4_voted` samples the answers-only re-derivation $K{=}4$ times at the arms' temperature and majority-votes it, against `maj4`, the majority over the first four base samples. The registration justified this as "both sides aggregate four things; only the origin differs." That is false about what we implemented, and we found it only after reporting the result. The arm's prompt displays *all sixteen* parsed answers (`disambig_run.py:87--89`, "Sixteen independent solvers gave these final answers"), so the arm votes four re-derivations each conditioned on sixteen answers, while the control aggregates four samples. The arm therefore carries an evidence *advantage*, and H-D1 is not an identified test of re-derivation. Everything else is the frozen protocol: same residuals, same caches, no new base sampling, seed 0.

	arm − control	estimate	95% CI	verdict (4.0 threshold)
H-D1 (primary)	four re-derivations − `maj4`	+0.0	[−4.1, +3.7]	not detectable
H-D2 (secondary)	`tally16` − `maj16`	−1.7	[−3.5, +0.5]	not detectable
H-D3 (exploratory)	`trace1` − `maj16`	−8.1	[−14.1, −0.4]	significant
	`trace2` − `maj16`	−8.1	[−13.4, −1.4]	significant
	`trace4` − `maj16`	−6.4	[−10.7, −1.0]	significant
	`trace8` − `maj16`	−6.1	[−9.5, −2.6]	significant

Verdict, and why we do not read it as the registration intended. The frozen analysis script labels the primary NOT DETECTABLE: $|+0.0|$ is below its realised 4.0-point threshold. The registration's pre-written wording for that branch is "we report it as not detectable; no claim either way, and we say the question is open after four attempts"; after E7 the count is five registered attempts (Appendix V). H-D1's literal wording ("not significantly negative") is also satisfied, but we do not treat that as support for anything, and we leave the statistic exactly as registered. We do *not* apply the framing the registration attached to it, because that framing ("at equal aggregation width, re-deriving is no worse than aggregating") presupposes a matching the design does not achieve. Applying it would state a conclusion we now know the test cannot support. The +0.0 is an exact cancellation of two effects that the same frozen estimator separates:

contrast	estimate	95% CI	what it varies
`maj16` − `maj4`	+4.42	[+0.51, +7.79]	evidence only, no re-derivation
`arm` − `maj16`	−4.42	[−6.89, −1.79]	re-derivation, evidence matched
`arm` − `maj4` (H-D1)	+0.00	[−4.07, +3.70]	both at once

The arm is not degenerate—it differs from `maj4` on 33.3% of items, with 266 discordant correctness pairs split 133/133—so the zero is a real cancellation rather than an arm copying its control. Per cell the H-D1 differences are −0.9, −3.0, +2.8, 0.0. We withdraw the claim, made in an earlier version of this appendix, that the interval "excludes every deficit we registered": [−4.07, +3.70] contains −3.0, −1.6 and −2.8, and only excludes −7.4 and −13.1.

Post-hoc contrasts, reported here only (not pre-specified). The design failed to identify re-derivation. The committed rows contain three contrasts that hold evidence fixed, and they point the same way, but each carries confounds that keep it out of the body and the abstract, as the registra-

tion's not-detectable branch requires. `tally16` is the sharpest: it sees exactly the sixteen answers the arm sees and is asked only to *count* them.

contrast	estimate	95% CI	reading
`tally16` − `maj16`	-1.65	$[-3.48, +0.52]$	the model can aggregate on demand
`arm` − `tally16`	-2.77	$[-5.37, -0.29]$	same evidence, arm re-decides
`arm` − `maj16`	-4.42	$[-6.89, -1.79]$	same evidence, arm re-decides

Levels: `maj16` 57.4, `tally16` 55.8, `maj4` 53.0, arm 53.0. Three confounds. (i) *Temperature*: the arm is sampled at T=0.8 and voted, while `tally16` is a single greedy call, so the `arm`–`tally16` contrast varies decoding as well as the act of re-deciding. (ii) *Width*: `arm`–`maj16` sets four voted re-derivations against a majority of sixteen. (iii) *Parsing*: the shared MCQ parser mis-reads text that discusses other answers (a letter followed by a word, or the last parenthesized option mentioned; Appendix S). Tally and arm outputs are exactly that kind of text, and E6 stored neither, so these contrasts cannot be audited. A signature of mis-parsing is visible in the committed rows: `tally16` picks an option with at most two votes over one with at least ten on 18 of 1,155 items. We therefore read the table as *consistent with* a re-derivation cost of a few points and no more.

The trace ladder ($-8.1/-8.1/-6.4/-6.1$ at $1/2/4/8$ traces against the sixteen-sample vote) closes as evidence grows, with the caveat the registration states: context length co-varies with evidence quantity here, so it is consistent with an evidence account rather than independent evidence for one.

What E6 does and does not revise. It does not touch E1–E5's registered pools, which stand as reported, and it changes no frozen statistic. It does not restore the registered law, and it does not identify a re-derivation cost: the registered test is not detectable, and the contrasts above are confounded post-hoc decompositions of a design that failed to identify. E7 (Appendix V) then matched evidence and width, found no detectable cost, and put the arm at $+1.3$ against the same tally. Reproduce with `bench/disambig/disambig_analysis.py` and `bench/disambig/disambig_contrasts.py`.

S PARSER AUDIT OF THE RE-GENERATION ARMS

The defect. The shared multiple-choice parser (`ext_strongarms.py:parse_letter`) takes the *last* match of `answer is\s*\(?\s*([A-J])` case-insensitively with no word boundary, so "…this answer is consistent with the majority" parses as C and "the answer is also B" as A; failing that, it takes the last parenthesized option mentioned. Base samples end in the fixed "The answer is (X)" format and are immune. Outputs that *discuss* other solvers' answers are exposed, and because a one-sided mis-parse is almost always scored wrong, it would bias every arm-minus-aggregate contrast *negative*—in the direction of the re-derivation claim this paper registered. An independent pre-registration review found it; we had not.

The registered arms are not artifacts of it. We re-parsed every cached output with a strict parser (same patterns and last-match rule, but the letter must be upper case and end at a word boundary) and recomputed each registered delta (`bench/extpower/parse_audit.py`):

outputs	parse-change rate	largest delta shift
base samples (all cells)	0–0.6%	—
E1/E2 debate (MMLU-Pro, ProofWriter, ARC; both sizes)	0–0.3%	0.27 points
E4 symmetric debate (ProofWriter; both sizes)	0%	0.00
tournament judges (`Winner:` parser)	0/2,049	0.00

The debate templates end in the same fixed answer format as the base samples, which is why they are unaffected. MATH-500 uses `math_verify` and is not affected.

What cannot be audited. E5 and E6 stored parsed answers but not the texts behind them, and their arm and tally outputs are exactly the exposed kind. Their contrasts are therefore unaudited for this defect; Appendix R reports the one signature visible without texts. This is a further reason the body makes no claim from them.

T OVERLAP BETWEEN STAGE B'S CORPUS AND THE TEST SET

What we found. Stage B samples its excerpts from the ABC corpus released with SSMR-Bench's own generator (Wang et al., 2025b). We had treated that corpus as independent of the test set. It is not, for one category: the generator builds its bar-line and time-signature items from corpus excerpts with the bar lines removed. Normalising both sides to pitch sequences (headers, bar lines, durations, chord symbols and decorations stripped) and matching 16-note runs, the synthetic categories (scale identification, chord completion) share no run with the corpus, which serves as a chance-match control, while 325 of the 400 rhythm items do (`bench/ssmr/stageB_overlap.json`).

How much of it reached training. The benchmark's *own* training split already shares music with 172 of the 400 rhythm test items, a property of SSMR-Bench that affects every method trained on it, including the published GRPO baseline. Our Stage B prompts share music with 66 of the 400 (4.1% of the benchmark); 28 of those are not already shared with SSMR-Bench's training split.

An assumption-dependent sensitivity, post-hoc. Comparing the 14B model before and after Stage B:

items	untuned	Stage A	Stage A+B	Stage B adds
rhythm, exposed to Stage B (66)	60.6	65.2	74.2	+9.1
rhythm, not exposed (334)	56.9	64.4	67.4	+3.0
other categories (1,200)	55.4	80.1	82.5	+2.4

If exposed and unexposed rhythm items would otherwise have gained equally, attributing their entire observed gain difference to overlap singles out roughly four net items among the 66 exposed. This is a sensitivity calculation under an exchangeability assumption, not a causal estimate or an upper bound. The groups were not randomized, and training can affect unexposed items too. In the 94.1 harness run, v1 answers 32 of the 66 exposed items and the model 34; those routing counts alone do not bound the effect on the full system. The checker and untuned-base systems use no Stage B training. No overlap-free retraining was run.

U COMPLETING TWO MATCHERS (ORACLE-V2, POST-HOC)

Status. Post-hoc, and a diagnostic comparison rather than a replacement. A review panel located two sub-types on which the frozen prover abstains for reasons of implementation rather than difficulty. We wrote complete matchers for exactly those two, developed them on `Train_textual` only, and froze them with a short registration (`PREREGISTRATION_oracle_v2.md`) and the one-shot test script before running on the test split. The registered promotion rule—use v2 for no system score unless test precision is at least 99.5%—passed. v2 is a comparison in the registered checker-writing tests (Appendix X.2) and a tested checker in ROBUST (Appendix Y). The earlier harness and residual analyses use frozen v1, and the registered checker-writing controls (Appendix X) are scored as systems of their own.

What changes. *Scale identification*: v1 asks `music21`'s default key-finder (`analyze('key')`, Krumhansl–Schmuckler (Krumhansl, 1990) with Aarden–Essen weights (Aarden, 2003)) for a key, which can return the relative key; v2 requires the pitch-class set of the excerpt's notes to equal the natural major or minor scale of exactly one option. *Bar-line placement*: v1 trusts `music21`'s measure parsing, which silently re-bars; v2 accepts an option only if its bar lines fall where greedy measure-filling under the question's own meter puts them. Both abstain unless the match is unique; every other sub-type is v1.

	training draws (development)		test (once)	
sub-type	v1 cov.	v2 cov. / prec.	v1 cov.	v2 cov. / prec.
scale identification	52.3%	100.0% / 100.00%	49.7%	100.0% / 100.00%
bar-line placement	29.2%	97.0% / 100.00%	28.4%	100.0% / 100.00%
all sub-types	70.25%	91.9% / 99.89%	69.9%	89.2% / 99.86%

System scores, with no new generation. v2's answer where it fires and the committed harness vote elsewhere: the untuned 14B goes from 88.9 to 96.2, level with Claude Opus 4.8 (96.2, brief reasoning, 1,600-token cap, no extended thinking) under the same protocol; the untuned 7B from 87.5 to 94.7; the distilled 14B from 94.1 to 96.5. Distillation's contribution falls from $+5.2$ to $+0.3$, showing that training's marginal contribution depends on the frozen checker's implementation.

What it does to the rest of the paper. v2 leaves 173 items. The agreement partition on them is weaker for the untuned 14B (165 unanimous at 67.3%, only 8 disagreeing) and similar for the others (7B: 82 at 68.3% vs. 91 at 37.4%; distilled 14B: 133 at 79.7% vs. 40 at 32.5%). The original SSMR-Bench analyses were run on v1's residual and should be read as properties of that residual; Appendix X.3 separately reports the post-hoc SYN analyses. Reproduce with `bench/ssmr/oracle_v2_test.py`.

U.1 A PARTIAL-INPUT AUDIT: ANSWERS THE OPTIONS GIVE AWAY (POST-HOC)

Status. Post-hoc and descriptive. A reviewer checking the SYN registration noticed that key-identification distractors are the gold key with a sharp, flat or minor added. We test the simplest rule that exploits this and reads neither the score nor the question: pick the option whose text is most similar to the other three (sum of `difflib` ratios), abstaining on a tie. It is a partial-input baseline in the sense of Gururangan et al. (2018); Poliak et al. (2018).

test class	n	options-only rule
key identification (`ScaleIdentificationFromAbc`)	352	**100.0%**
bar-line placement	183	55.2%
scale selection / chord naming	48 / 44	75.0% / 75.0%
the other five classes	973	2.0–28.0%

What it shows. All 352 key-identification items (22% of the test set), and all 1,660 key-identification training draws, are answered from the option texts alone. The frozen prover does not use this: it fires on 175 of the 352 by deriving the key from the score; the rule is ROBUST's and ROBUST-P2's registered reference checker, and Figure 3's post-hoc options-only row applies it to key-identification items selected by the class label. Joined with it (the prover where it fires, the rule on key identification where it abstains), a program with no model settles 81.0% of the test set at 99.85%. Thus a model can score on these items without reading the notation; this audit does not establish that any particular trained model uses the shortcut. ROBUST-P and its 7B replication first test its effect on the answer policies (pre-registered, legacy parser, Appendix Y.4). The distillation gain survives the tested repair under both legacy scoring and post-hoc strict re-scoring, but the latter does not establish a change in the gain. The repair leaves a gold-spelling leak on $27/352$ items (post-hoc, `bench/robust/robust_spelling_leak.txt`), so it does not remove all options shortcuts. The corrected, pre-registered test draws every option from the same gold-spelling vocabulary and preserves the distillation gain at both sizes (Appendix Y.5). Reproduce the options-only audit with `bench/ssmr/options_only_audit.py`.

V E7: RE-DERIVATION MATCHED ON EVIDENCE AND AGGREGATION WIDTH

Re-derivation remains open. The registered taxonomy predicts only $3/5$ interventions per size (Appendix L). A registered symmetric-debate control confirms neither hypothesis, withdrawing debate as evidence for a re-derivation cost (Appendix O). The final control, matched on evidence and aggregation width (E7), tests four non-music residuals at 7B and returns registered branch B, *not detectable*: -0.35 $[-3.46, +2.35]$. Its interval is no equivalence bound and $1.96 \times$ SE is not a minimum detectable effect. The question remains open after five registered attempts (experiment map: Appendix I).

Registered before it ran. E6 failed to identify because its arm saw sixteen answers while its control aggregated four. E7 (`bench/e7/PREREGISTRATION_e7.md`, frozen at `49de27b`) matches both: the arm's prompt is byte-identical to E6's and shows the sixteen parsed base answers; it is sampled $K{=}16$ times and majority-voted, against the majority of those same sixteen

answers. Before freezing, two rounds of adversarial review found and fixed a seed overlap with E6, the parser defect of Appendix S, a parse gate that could never fail, and a resume path that would have failed permanently. The power was computed first (threshold 2.6, 80% power at 3.7), with the stated expectation that the effect might fall below it. The analysis code, parser and estimator were unchanged from the freeze commit when the analysis ran.

Run quality. All four cells returned their registered residuals (219/367/470/99). Explicit, untruncated answers: 96.3%, 99.7%, 100.0%, 100.0% of verdicts (gate 90%). Effective voting width: 16.0 on both sides (minimum 14). The arm states "answer is (X)" as often as the base samples do, so no cell is format-divergent.

contrast	estimate	95% CI	status
H-E7: arm − `maj16`, strict parser	−0.35	[−3.46, +2.35]	primary; branch B
same, frozen `parse_letter`	−0.43	[−3.54, +2.28]	sensitivity; same branch
arm − `tally16` (E6)	+1.30	[−2.00, +4.13]	secondary
first 1 re-derivation − `maj16`	−8.23	[−11.23, −3.52]	exploratory
first 4, voted	−3.12	[−6.21, −0.52]	exploratory
first 8, voted	−2.16	[−5.02, +0.39]	exploratory
all 16, voted	−0.35	[−3.46, +2.35]	exploratory

Verdict, in the registration's pre-written words. Branch B: the CI contains zero and $|D|$ is below the realised 2.9-point threshold. "Matched on evidence and width, this design detects about 2.9 points and the effect is smaller; we report it as not detectable." The question remains open after five registered attempts. The realised threshold is 1.96×SE, not a power-based minimum detectable effect, and the interval $[-3.46, +2.35]$ allows a loss of several points, so "the effect is smaller" is the registration's wording, not an equivalence bound. E6's post-hoc contrasts stay exploratory; its −2.8 against `tally16`, which was temperature-confounded and unaudited, does not reappear here (+1.3 against the same tally). The ladder shares samples across rungs and is exploratory, but its shape is the clearest thing E7 shows: one re-derivation trails the aggregator by eight points, and the gap closes as more re-derivations are voted. Reproduce with `bench/e7/e7_analysis.py` after `bench/reproduce_setup.sh` links the committed E6 rows, 7B E1/E2 caches and MATH-500 problems to their run-host paths; the output is committed as `bench/e7/e7_results.txt`.

W THE AGREEMENT-GATED CASCADE

W.1 RESIDUAL AGREEMENT AND CONFIDENCE ROUTERS

Agreement is useful conditionally (post-hoc on SSMR-Bench). On v1's residual, the distilled-14B harness is 89.8% accurate on agreements versus 34.2% on disagreements. On the 87 SYN abstentions with cached samples, the corresponding rates are 73.3/33.3%, but the untuned 14B has only three disagreements. Figure 6 and Appendix B.4 report the external replications and their registration status. The signal depends on the checker–policy pair.

Routing is a secondary application. SSMR-Bench routing is post-hoc (Appendix X.3; Figure 9). Behind SYN, escalating the distilled 14B's 27 disagreements gains +0.69 [+0.25, +1.19]; its cached hybrid fallback includes 19 v1 answers where model samples are absent. With the untuned cached harness the gain is +0.12 [+0.00, +0.31]. The pre-registered confidence-router comparisons on MATH-500 and MMLU-Pro give a qualified result. Only MMLU-Pro at 7B shows no strict dominance; agreement pointwise dominates all three routers in the other three cells. Each router prediction holds in 3/4 cells. Counting the 7B samples, the pre-registered cascades cost 0.96 × /1.13× a pure-14B run on MATH-500/MMLU-Pro, so fewer 14B calls need not mean less compute (Appendices H, I). This is *a useful diagnostic and operating point, not a superior routing method*.

Figure 9 shows an earlier cascade behind v1; Appendix X.3 gives the routing behind SYN summarized above. On SSMR-Bench it is post-hoc: `bench/ssmr/agreement_cascade.py` was written after the instrumented run and the Opus anchor existed, and no registration covers it; the random-escalator comparison (94.9%, +2.0) has no interval. It uses v1's residual and the distilled

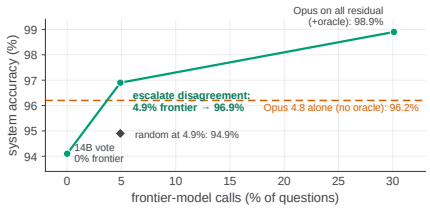


Figure 9: **Agreement-gated escalation behind v1 (post-hoc).** The distilled 14B calls Opus on the 4.9% of questions with residual disagreement, reaching 96.9% (+2.8 over its v1-based harness). Calling Opus on all of v1's residual instead reaches 98.9%; this is a measured comparator, not an upper bound. The same-budget random escalator scores 94.9% (diamond; no interval computed). Dashed: Opus answering all questions with brief reasoning.

14B, not the registered SYN system. The registered versions of the gate are MATH-500 and MMLU-Pro (Appendices H, I).

X A MODEL THAT WRITES THE CHECKER: ZERO-SHOT (PAL) AND WITH FEEDBACK (SYN), PRE-REGISTERED

X.1 ZERO-SHOT, THE UNTUNED MODEL (PAL)

Registered before it ran. The rival to a hand-written checker is a model given the same library (PAL / Program-of-Thought). The registration (`bench/pal/PREREGISTRATION_pal.md`, frozen at `0100037`) fixed the prompt, model, decoding, sandbox, commit rule, fallback, estimator and a ±3-point margin before any generation, with wording for all four outcomes. A pre-freeze review found the first draft broken in ways that would have decided the result—a hard-coded letter counted as a checker, a list return crashed scoring, and a per-user process limit made every `numpy` import fail—and all were fixed before freezing.

Design. For each of the 1,600 test items the untuned Qwen2.5-14B-Instruct is shown the question, the score and the options, told that its code receives them as `ABC` and `OPTIONS`, and asked for a no-argument `solve()` that uses `music21` and returns the option letter or `None`, without guessing (zero-shot, greedy). Each program runs twice in a `bubblewrap` sandbox (read-only root, no network, private PID namespace, CPU, memory and file limits): once with the real options and once with the option texts relabelled by a cyclic shift. It *commits* only if both runs return letters naming the same option text, so a hard-coded letter cannot commit. PAL and the frozen prover both fall back to the model's own greedy answer wherever they do not commit.

	coverage	precision	system (same greedy fallback)
PAL, untuned 14B	5.6%	40.0%	54.5
hand-written prover (frozen v1)	69.9%	99.82%	87.25
greedy answers alone	—	—	55.9
D = PAL − v1, paired item bootstrap			$\mathbf{-32.75}$ $[-35.06, -30.56]$

Verdict, in the registration's pre-written words. Branch A: "Given the same library, the untuned policy does not write a checker as good as a hand-written one: -32.8 $[-35.1, -30.6]$. The checker is doing work the policy does not." The hand-written checker was developed against these test items and labels; PAL is zero-shot. Against the post-hoc complete prover the gap is -40.8 $[-43.2, -38.5]$.

Why it fails, and what that does and does not show. Of the 1,600 programs, 1,193 crash, 308 return `None` (the abstention the prompt allows), 9 are unstable under relabelling, and 90 commit. 1,192 of the 1,193 crashes call `converter.parse(ABC)` without `format='abc'`, which `music21` cannot resolve for a header-less excerpt. So the headline gap is largely one missing API detail—a real failure to use the library, but a narrow one. It is not the whole story: of the 90 programs that ran and survived the relabelling, only 36 are right. Two asymmetries favour our prover

and we state them: it was developed and debugged against these very items, and PAL is zero-shot with no retry. The result says the untuned model does not, unaided, write a working checker; it does not say that no policy could. Reproduce with `bench/pal/pal_analysis.py`.

Does the one API detail explain the gap? No (post-hoc). After the primary we froze a variant that adds the missing sentence—"Parse any ABC string `s` with `music21.converter.parse(s, format='abc')`"—and nothing else (`bench/pal/ADDENDUM_hint.md`). Every crashing program then passes the format argument, yet the gap barely moves: coverage 10.8%, precision 59.5%, system 55.4, $D = -31.88$ $[-34.25, -29.62]$. The crashes become genuine `music21` misuse and logic errors (for example `'Chord' object has no attribute 'octave'`, `'Key' object has no attribute 'tonicName'`, index and parsing errors in hand-rolled ABC handling). So the untuned model does not write a working checker even when told how to parse the score. This is post-hoc and does not replace the registered branch.

The registered 7B secondary. It ran after the primary (a first attempt failed at start-up when another user claimed the GPU). None of its $1{,}600$ programs commits: $1{,}583$ crash, 15 return no letter and 2 are unstable, so its system equals its own greedy answers, 44.2, against 84.6 for the frozen prover with the same fallback: $D = -40.44$ $[-42.81, -37.94]$. Its crashes are not one detail: only 413 omit `format='abc'`, and the rest fail in varied ways. A secondary, it cannot change the registered branch.

X.2 With feedback: a frontier model (SYN) and the untuned 14B (SYN-14B)

Registered before any writer call. PAL (Appendix X.1) withholds everything a checker author normally has. SYN grants it, to a frontier writer: the registration (`bench/syn/PREREGISTRATION_syn.md`, frozen at `d648c13`) fixed the writer, prompts, data splits, feedback, sandbox, commit rule, selection rule, estimator, a ± 3-point margin, two qualifiers and the wording of all four outcomes before the first call. Two adversarial pre-freeze checks by five reviewers found and we fixed, among others: a split rule that left the main interval templates out of the feedback set; a one-process commit rule a program could beat by coordinating its two calls; result-line checks (a line can still be forged in-process, which confers nothing a return value could not); SysV shared memory and a controlling terminal reachable from the sandbox; unbounded child processes; and an options-only artifact (Appendix U.1), which motivated the score-blind qualifier.

Design. Claude Opus 4.8 (adaptive thinking, effort `high`) is asked for one general `solve(question, abc, options)` that uses `music21`, returns `None` rather than guessing, and does not rely on letter positions. It sees only `Train_textual`: 40 worked items with answers, then, for five revision rounds, the best program so far and its execution feedback on a 400-item feedback split (aggregate counts, up to 25 wrong commits and 15 uncommitted items with their answers, up to 15 tracebacks). Three independent runs of six rounds give 18 programs; the registered rule picks the one maximizing $J = \text{correct} - 1.27 \times \text{wrong}$ on an 800-item selection split the writer never sees (the weight is the primary metric's break-even given the 55.9% greedy fallback). Every call runs in its own sandbox with a minimal filesystem (no repository, data, prover or results mounted), its own user, network, PID and IPC namespaces, no controlling terminal and a four-task cap; an item commits only if the program's answer survives cyclic relabelling of the options, run in a second sandbox that sees only the relabelled options. The chosen programs were committed before the single test pass (`bench/syn/frozen/`). The primary is PAL's: D = system with the synthesized checker minus system with the hand-written prover, both falling back to the untuned 14B's greedy answer, paired item bootstrap.

checker (same greedy fallback, untuned 14B)	coverage	precision	system
hand-written prover v1 (frozen)	69.9%	99.82%	87.25
PAL, untuned 14B, zero-shot	5.6%	40.0%	54.5
completed matchers v2 (post-hoc, Appendix U)	89.2%	99.86%	95.3
SYN, Opus 4.8-written (primary)	**93.4%**	99.67%	**96.8**
D = SYN − v1, paired item bootstrap			$+\mathbf{9.56}$ $[+8.06, +11.19]$

Verdict, in the registration's pre-written words. Branch C: "A frontier model with feedback writes a better checker than ours." Neither qualifier fires. On the 1,352 items whose question family has at least 50 training draws $D = +10.36$ $[+8.65, +12.06]$, and on the 248 rare or unseen ones $+5.24$ $[+1.61, +9.27]$ ($+2.82$ $[-0.81, +6.85]$ with the 7B's fallback). A post-hoc breakdown finds $+18$ net items from chord naming (44 items, 4 training draws), -5 from chord completion (156, 9 draws; 75.6% coverage against the prover's 87.2%), and on scale writing (48, no training draw) the checker commits on none (`bench/syn/syn_posthoc.py`). Replacing by the fallback each commit a system repeats when the score is swapped for one other item's (option-only commits, but also answers that happen to be equal) gives $D = +8.94$ $[+7.50, +10.38]$: the synthesized checker reads the score. Each run's own best program lands at $+9.38$ to $+9.69$, and with the untuned 7B's fallback $D = +11.44$. Against the post-hoc complete matchers the synthesized checker is $+1.50$ $[+0.75, +2.31]$; against PAL, $+42.3$ (both registered secondary comparisons). A post-hoc multi-substitution test (`bench/subst/syn_multisub.py`; after an external review) replaces each commit's score three times by the scores of other same-class test items with a different gold answer. On the 310 commits on chord- and note-completion items, whose options begin with the full score, every answer stays unchanged: for these question types SYN computes its answer from the notes in the options and never reads the score field (`bench/syn/frozen/primary.py`), so this test cannot separate using those notes from ignoring the score. On the other 1,184 commits, 205 are unchanged under the registered single substitution but only 12 (1.01%) under all three; 10 of these are bar-line items whose three substitutes share the original's bar length (equal answers), and 2 are time-signature items. Sending those 12 to the fallback (v1 unchanged) gives $D = +9.06$ $[+7.56, +10.69]$, against the registered $+9.56$. Sending SYN's 310 completion commits to the fallback as well would give $+0.69$ $[-1.50, +2.88]$, but that comparison is not like-for-like: v1 also answers 189/201 note- and 136/156 chord-completion items from the same option-borne information and would keep them.

What it commits and what it leaves. It commits on 1,494 items, 5 of them wrong (4 time signatures, 1 chord name); it covers 394 items the prover leaves and misses 19 the prover takes. Its 106-item residual is scale writing (48, a template with no training draw), chord completion (38), time signature (10), note completion (9) and one chord name. The winning program is 304 lines and came from round two of its run; all three runs reach 96–99% selection coverage by round three. The development stage used 18 calls, 0.25M input and 0.71M output tokens (thinking included). Like the prover, it fires on none of the 367 MusicTheoryBench and 807 WildScore items (`bench/syn/syn_ood_fire.py`, post-hoc).

Post-hoc, beside the registered result. With the synthesized checker the untuned 14B's system (96.81) is $+2.75$ $[+1.50, +3.94]$, $+2.50$ $[+1.25, +3.69]$ and $+2.62$ $[+1.44, +3.94]$ above the trained 14B system of Table 4 over its three seeds (94.06, 94.31, 94.19; `bench/syn/syn_posthoc.py`), and level with Claude Opus 4.8 (brief reasoning, 1,600-token cap, no extended thinking) answering every question itself (96.19; $+0.62$ $[-0.50, +1.75]$). Replacing the untuned fallback by the three distilled 14B seeds gives 97.3–97.7, so distillation adds $+0.50$ $[-0.31, +1.31]$, $+0.50$ $[-0.25, +1.31]$ and $+0.88$ $[+0.19, +1.62]$ points (two intervals include zero) where the checker is strong, against 5.3–6.2 over the hand-written prover with the same greedy fallback (87.25 → 92.6–93.4).

Asymmetries, stated. The hand-written prover was developed against the test items; the synthesized checker saw training draws only, whose template mix differs (chord completion 9 training draws against 156 test items, scale writing 0 against 48). The writer may have seen the public generator in pretraining; it is also the model that anchors Table 4, but under different settings: the anchor answers with brief reasoning, a 1,600-token cap and no extended thinking, while the writer used adaptive thinking at high effort and a 64K-token budget (0.71M output tokens over 18 calls, against 0.017M for the untuned 14B writer below). Reproduce with `bench/syn/syn_analysis.py`.

The same loop with the untuned 14B as writer (SYN-14B, pre-registered). A registered twin (`bench/syn14b/PREREGISTRATION_syn14b.md`, frozen at `45e9ed9` before any 14B writer call, written while SYN's development stage was running and after seeing its training-split rounds) reuses SYN's prompt templates, splits, feedback rules, sandbox, commit rule, selection and analysis with untuned Qwen2.5-14B-Instruct, Qwen's chat sampling and three seeded runs of

six rounds. The writer-specific context cap can reduce the number of feedback examples shown; decoding and output budgets also differ. It tests the same development protocol with a different writer configuration, not an isolated effect of model size; PAL and SYN additionally differ in per-item versus reusable-program development. Branch A, in its pre-written words: "Given the same library, labelled training items and five rounds of execution feedback, the untuned 14B does not write a checker as good as the hand-written one": $D = -30.31$ $[-32.69, -28.12]$, and neither qualifier fires ($D_{\text{common}} = -31.95$, $D_{\text{rare}} = -21.37$). Two of its three runs never commit on a single training item; the third's best program commits on 6.2% of the test at 77.8%, every commit unchanged when the score is swapped out (option-only), and its system (56.9) barely clears its own greedy answers (55.9). Its first programs crash on `music21` misuse (undefined names, `stream.Note`, `Key.tonicName`), and feedback does not remove it: one run trades its crashes for abstentions and non-letter returns, another alternates between abstaining on everything and a new misuse (`Interval.quality`), and the selected program keeps its `stream.Note` crash in all six rounds and still crashes on 983 of the 1,600 test items. Against PAL it is $+2.4$ $[+1.3, +3.5]$; against the Opus-written checker, -39.9 $[-42.4, -37.6]$. So SYN's protocol barely helps the small writer ($+2.4$ over PAL, from option-only commits): the large performance difference remains associated with the writer configuration; SYN-NT below removes thinking and the per-call budget difference, and the gap stays. Reproduce with `bench/syn14b/syn14b_analysis.py`.

The frontier writer at the small writer's budget (SYN-NT, pre-registered). A second registered twin (`bench/synnt/PREREGISTRATION_synnt.md`, frozen at `4ee7b9b` before any writer call, after SYN and SYN-14B were scored) imports SYN's loop unchanged (pinned to its registration commit) and changes only the writer's decoding to SYN-14B's: Claude Opus 4.8 with no extended thinking, no effort setting and `max_tokens` 8,192 per call, three runs of six rounds. SYN-14B's prompt cap, a limit of the small model's context window, is not imposed. Branch C, in its pre-written words: "At the small writer's decoding budget, the frontier writer still writes a better checker than ours": $D = +9.75$ $[+8.25, +11.31]$ with the untuned 14B fallback (system 97.00%; primary: run 1, round 6, 93.88% coverage at 99.67% precision); no qualifier fires ($D_{\text{common}} = +10.21$ $[+8.58, +11.91]$, $D_{\text{rare}} = +7.26$ $[+4.03, +10.89]$, $D_{\text{blind}} = +9.19$ $[+7.69, +10.62]$); with the untuned 7B fallback $+11.62$ $[+10.06, +13.19]$. Registered secondaries: against SYN (Opus with adaptive thinking) $+0.19$ $[-0.19, +0.56]$; against SYN-14B $+40.06$ $[+37.75, +42.62]$. The writer produced 129,096 output tokens over 18 calls ($0.18\times$ SYN's, $7.8\times$ SYN-14B's); 8 calls stopped at the cap, 6 of them without extractable code (run 1 rounds 4–5, run 2 rounds 3–6). Per-run picks (registered secondaries, no branch reading): run 0's is its round-1 program, one 5,007-token call (72.0% coverage at 99.57%; $D = +0.00$ $[-1.00, +1.06]$); run 2's reaches 81.25% at 99.62% ($D = +3.62$ $[+2.19, +5.12]$). Coverage and precision per run and round are in `bench/synnt/synnt_results.txt`. So at a matched per-call budget and without thinking, the frontier writer keeps the whole advantage over the 14B writer; total output still differs $7.8\times$ (the 14B's programs were short), so budget is matched per call, not in total. Deviation from the registered test-once procedure: the first test pass was stopped by an operator time limit before any result was written; its start marker was renamed (`bench/synnt/test_started_attempt1.json`, committed with this explanation) and the pass was re-run unchanged. Scoring is deterministic, and no test outcome existed before the re-run. Reproduce with `bench/synnt/synnt_analysis.py`.

X.3 The SYN residual: partition and cached routing (post-hoc)

Partition. The frozen SYN checker abstains on 106 test items. Three-sample harness outputs exist for 87 of them, those also left by v1; no new samples are generated. The distilled 14B agrees on 60 and answers 44 correctly (73.3%), while it disagrees on 27 and answers 9 correctly (33.3%). The untuned 14B agrees on 84, with 46 correct (54.8%), and disagrees on only 3, with one correct. These are descriptive partitions of the available overlap, not estimates for all of SYN's residual. Source: `bench/syn/syn_residual_partition.txt` and its analysis script.

Routing and selection headroom. A further post-hoc re-analysis combines the frozen SYN decisions with the committed harness and Opus anchor outputs (`bench/syn/syn_residual_routing.txt`). It does not rerun inference. On the 19 SYN-residual items without samples, the cached harness returns v1's checker answer. The

resulting fallback therefore contains v1 as well as the model; it is distinct from SYN's registered untuned-greedy fallback.

Cached fallback on SYN abstentions	No escalation	Agreement gate	Random (expected)
Distilled 14B harness	97.56	98.25	97.94
Untuned 14B harness	97.19	97.31	97.24

All entries are system accuracy over the 1,600 questions. For the distilled fallback, the gate calls Opus on 27 items (1.7%); the paired system gain is $+0.69$ $[+0.25, +1.19]$. A correct candidate exists on $13/27$, compared with 9 correct harness answers and 20 correct Opus answers. For the untuned fallback, only 3 items (0.2%) are routed, giving $+0.12$ $[+0.00, +0.31]$; a correct candidate exists on $1/3$, the harness answers one correctly, and Opus answers all three. Random escalation samples the same number uniformly from all 106 abstentions; these are expected accuracies without intervals. Calling Opus on all 106 gives 99.06% system accuracy, with 96 correct residual answers, compared with 60 for the registered greedy fallback (96.81% system accuracy). The stronger checker leaves less routing gain, and the untuned model's near-unanimity limits this particular gate. These results do not replace the registered SYN comparison or establish a deployment cost advantage.

Y ROBUST: FROZEN-CHECKER ROBUSTNESS, PRE-REGISTERED

Design and status. The registration `bench/robust/PREREGISTRATION_robust.md` was committed before any perturbed item was scored. Its date header is a typo; the results file records the preceding commit time. The frozen v1, v2 and SYN checkers are evaluated on the original test items and under four perturbations. Two replace the first line of each question with a template rephrasing drafted by an LLM coding agent (Claude) under the authors' direction after SYN's test pass, without a blindness protocol (ROBUST-N's same-model, code-blind paraphrases replicate the coverage collapse), to preserve meaning and the embedded chord, interval, key/scale and direction parameters (on 2 of 48 scale-writing items the rephrasings' "scale" wording also admits a harmonic-minor option; Appendix Y.1). The header condition prepends standard ABC `X:1` and `T:Excerpt` lines. The key-repair condition replaces the distractors on 352 key-identification items by three randomly drawn major/minor keys, excluding the gold tonic/mode, and retains the gold answer and option positions. No checker or perturbation was changed after scoring. The options-only rule is a reference.

H-R1: template dependence. The frozen conjunction predicts at least a halving of coverage for v1 and SYN under each rephrasing, with retained precision at least 99%. It held in $2/4$ cells. Table 1 gives all coverage and precision values, including v2, for which H-R1 makes no prediction.

Cell	Registered result	Reason
v1, rephrasing 1	not held	$69.9 \to 35.4$ coverage: not halved
v1, rephrasing 2	held	$69.9 \to 15.8$; precision 100.00
SYN, rephrasing 1	not held	precision $98.07 < 99$
SYN, rephrasing 2	held	$93.4 \to 12.5$; precision 100.00

H-R2: header robustness. Coverage and precision were predicted to change by at most two points. H-R2 held for v1, v2 and SYN $(3/3)$; all changes were exactly zero. This is robustness to the tested header addition, not to arbitrary notation changes.

H-R3: repaired key distractors. The options-only rule was predicted to fall below 50% precision; each executable checker was predicted to retain at least 99% precision and change coverage by at most five points on key identification. Only the options-only prediction held.

Checker	Key coverage before	Key coverage after	Key precision after	H-R3
Options-only rule	100.0	45.7	20.50	held
v1	49.7	56.2	88.38	not held
v2	100.0	84.9	100.00	not held
SYN	100.0	100.0	93.47	not held

v1 misses both the precision and coverage-change criteria; v2 misses the coverage-change criterion (-15.1 points); SYN misses the precision criterion. These are the registered outcomes, unchanged by the diagnostic below. Full results: `bench/robust/robust_results.txt`.

Relative-key diagnosis (post-hoc). The key-identification scores consist of 16 shuffled notes of the gold scale: the tonic four times, every other degree twice (all 352 test items). A key and its relative major/minor have the same pitch-class set. The original generator never offers that relative key; the repair does on 53 items. Each of v1 and SYN commits on all 53 and errs on 23, always choosing the relative key. Among the other 299, v1 commits on 145 and SYN on all 299, each with 100.00% precision. v2 abstains on exactly the 53 relative-key items and correctly answers all 299 others. The score is not ambiguous: a count-aware rule (unique most frequent pitch class as tonic, scale equal to the pitch-class set) answers $352/352$ original and $352/352$ repaired items, $53/53$ of those offering the relative key. So v1 (music21's Krumhansl–Schmuckler key-finder with Aarden–Essen weights) and SYN (a pitch-class-set match that falls back on that key-finder when two options fit) use key rules whose weakness the generator's distractors never expose, and v2's abstention on the 53 is conservative, not required. This diagnosis does not rescue H-R3. Sources: `bench/robust/robust_posthoc.txt`, `robust_tonic.txt`.

Y.1 ROBUST-N: SAME-MODEL-AUTHORED PARAPHRASES AND A WORDING NORMALIZER (PRE-REGISTERED)

`bench/robustn/PREREGISTRATION_robustn.md` (frozen before any paraphrase existed; registered under the label "independently authored", which we now call same-model because the registered author is SYN's writer model) answers a reviewer's question about unseen paraphrases and wording normalization. Claude Opus 4.8 (no extended thinking), shown only each of the nine canonical template lines with parameters as placeholders, wrote five rephrasings per template, requested as meaning-preserving "as a different benchmark author might"; it saw no checker code, item or answer, and no rephrasing was edited or filtered (`paraphrases.json`, committed before normalization). Set k uses the k-th rephrasing of every template. A normalizer (the same model, shown the nine templates and one rephrased line; no score, options or answer) rewrote each of the 645 unique rephrased lines into a template line with its parameters, or NONE (`normalized.json`, committed before scoring). Amendment 1, committed before any normalizer output: the API rejects `temperature` 0 for this model, so the normalizer runs at the default. Frozen v1, v2 and SYN (sandbox, commit rule) were scored on each raw and normalized set. No policy ran in this test; ROBUST-NP (Appendix Y.3) later ran three Qwen policies on these paraphrases.

Checker	Original	Raw sets 1–5 (coverage %)	Raw precision (%)	Normalized sets 1–5
v1	69.9	23.0/38.7/23.0/38.7/25.6	99.46–99.68	69.9 at 99.82, all five
v2	89.2	23.0/46.9/23.0/46.9/33.8	99.46–99.73	89.2 at 99.86, all five
SYN	93.4	15.6/52.0/28.1/36.9/40.1	98.00–99.40	93.4 at 99.67, all five

Registered outcomes: SYN's coverage is at most half its original on raw sets 1, 3, 4, 5 and not on set 2 (52.0%; $4/5$ held); on every normalized set SYN is back to 93.4% at 99.67% ($5/5$ held); the normalizer returns each item's original first line (up to case, whitespace, trailing punctuation and a/an) on $1{,}600/1{,}600$ items with no NONE in every set ($5/5$ held). The normalizer was given the benchmark's template family, which an auditor of a template-generated benchmark has; it shows that, on the 44 task-preserving paraphrases, the checkers' rewording loss is entirely a front-end failure (set 3's note-completion line is the exception below). Per template (a registered descriptive report; its printing code was added to `robustn_analysis.py` after the results, commit `08c985d`, with the hypothesis code unchanged), coverage is all-or-nothing: each raw set either keeps a template's trigger phrase (full coverage) or loses it (0%); key identification is at 0% in all five raw sets. The normalizer returned no NONE and no call failed. Sources: `bench/robustn/robustn_results.txt`, `robustn_peritem.json`.

Task preservation (post-hoc). After an external review, we read every paraphrase against its template and checked the two flagged templates with `music21`. Set 3's note-completion line, "Identify the correct note to build the $\{0\}$ interval *above* the note given in the score", adds a direction that the

canonical template ("… form the {0} interval") lacks; the benchmark's gold completions lie above the given note on 87 of the 201 items, below on 78 and at the unison on 36. Read literally, 72 of the 78 descending items have no correct option and 6 (items 1230, 1355, 1520, 1531, 1537, 1554) have a different one; the 36 unison items are vacuous but keep their answer. No checker commits on any note-completion item in raw set 3, so no raw-set number scores a wrong-direction gold; the normalizer maps every one of these lines back to the direction-free template, so on these 201 items "restoration" discards the added constraint rather than repairing the same task. Without the template (1,399 items), set 3 still halves SYN's coverage (93.07 → 32.17%; H-N1) and normalization returns exactly the original outcomes of all three checkers (93.07% at 99.62%; H-N2). Under the canonical wording itself, 8 items offer two options that form the named interval (one up, one down), a pre-existing ambiguity. By reading, the remaining 44 paraphrases add or drop no constraint. ROBUST's own rephrasings say "scale" where the template says "key"; a `music21` spelling check finds that on 2 of the 48 scale-writing items (829, 834) a harmonic-minor distractor is then also a correctly spelled scale in the named direction, while the gold (natural minor) stays correct; no other scale item has one. Source: `bench/robustn/robustn_direction.py` and its `.txt`.

Y.2 SYN-R: ROBUSTNESS-AWARE SYNTHESIS (PRE-REGISTERED)

Registered at `bench/synr/PREREGISTRATION_synr.md` before any writer call, after ROBUST-N's and SYN-NT's results were seen and in response to a review asking for a robustness-aware synthesis baseline, SYN-R reruns SYN-NT's loop (Claude Opus 4.8 without extended thinking, 8,192 output tokens per call, $K = 3$ runs of $R = 6$ rounds, SYN's splits, sandbox, commit rule and J selection, SYN's code pinned) with two changes. The task prompt adds a paragraph asking for code that identifies the question and its parameters from meaning rather than exact phrases, states that test wordings will differ, and lists ROBUST's 18 rephrased forms. The 40 worked items cycle through original wording and ROBUST's two rephrasings, and every feedback and selection item also appears with both rephrasings (1,158 and 2,324 items; 59 items of a training-only template have no rephrasing). The frozen primary (run 0, round 3; 90.7% selection coverage at 100% precision; 6 of 18 calls hit the output cap, 5 of them without a program; 140,304 output tokens in all, against SYN-NT's 129,096) was scored once on the original test items and ROBUST-N's five raw sets, which the writer never saw, with SYN-NT's frozen primary as the matched baseline. **H-SR1 held**: original coverage 85.00% at 99.63% precision (thresholds 85% and 99%; 1,360/1,600 commits, so one fewer would have failed it). **H-SR2 (primary) held in 5/5 sets**: SYN-R's coverage exceeds SYN-NT's by +57.00 [+54.56, +59.50], +25.19 [+22.31, +27.94], +33.00 [+30.56, +35.25], +59.62 [+57.19, +61.94] and +32.44 [+30.12, +34.88] (SYN-R 72.6/76.5/61.1/85.0/72.6%; SYN-NT 15.6/51.3/28.1/25.4/40.1% at 97.99/99.39/98.89/99.01/99.22% precision; on set 3 without its non-task-preserving note-completion line, Appendix Y.1, the gain is +37.74 [+34.95, +40.31], neither program committing on those items; post-hoc, `bench/robustn/robustn_direction.py`). **H-SR3 held in** 5/5: precision 99.49–99.63% (5 wrong commits per set). **H-SR4 held in** 3/5: coverage stays at least ten points below SYN-R's original coverage on sets 1, 3 and 5 (−12.44, −23.94, −12.44) but not on set 2 (−8.50) or set 4 (0.00). Per template the dependence is still all-or-nothing: interval naming has zero coverage in sets 1 and 5, chord completion in sets 2 and 3, and bar-line placement and note completion in set 3, while key identification now keeps 95.7% in every set. The frozen program dispatches on keywords (e.g. "key", "interval", "bar line") rather than whole template lines; key identification alone contributes +21.1 points of the gain in every set, and some ROBUST-N paraphrases share phrases with the listed rephrasings (e.g. "Pick the note that …" in set 2's note-completion line, +12.1 points there). On original items SYN-R's selected program covers 8.88 [7.56, 10.25] points fewer than SYN-NT's, mostly on bar-line placement (94.5% → 29.5%), and with the cached untuned-14B fallback its system accuracy is 92.44% against 97.00%, −4.56 [−5.62, −3.56] (post-hoc, `bench/synr/synr_posthoc.py`; item bootstrap over two fixed programs). This is not an estimate of a causal cost of the robustness changes: SYN-NT's own per-run programs cover 72.0–93.9%. No policy ran in SYN-R itself; ROBUST-NP (Appendix Y.3, H-NP4) later adds three Qwen policies. The same model (Claude Opus 4.8 without extended thinking) wrote ROBUST-N's paraphrases and SYN-R's code. Sources: `bench/synr/synr_run.py`, `synr_analysis.py`, `synr_results.txt`, `synr_test.json`, `frozen/`.

Y.3 ROBUST-NP: THE MATCHED CONTRIBUTION END-TO-END ON SAME-MODEL PARAPHRASES (PRE-REGISTERED)

Registered at `bench/robustnp/PREREGISTRATION_robustnp.md` before any policy output on a paraphrase existed, after ROBUST-N, SYN-R, ROBUST-P2–P4 and ROBUST-F had been seen and in response to reviews asking what matched contribution remains for the normalized checker and for SYN-R when a policy is actually run on the paraphrases. The untuned Qwen2.5-14B-Instruct policy (ROBUST-P2's prompt, greedy, 8,192 tokens, strict extraction) answers the 1,600 original items and each of ROBUST-N's five raw paraphrase sets (9,600 prompts). No checker is re-run: SYN's raw and normalized outcomes are ROBUST-N's committed per-item outcomes, and SYN-R's and SYN-NT's raw outcomes are SYN-R's committed test outcomes. On raw set k the systems are (a) SYN on the raw paraphrase with the policy on the raw paraphrase, (b) normalization: SYN on the normalized line (ROBUST-N's `norm` k outcomes) with the policy on the raw paraphrase, the normalizer feeding only the checker, and (c) SYN-R and SYN-NT on the raw paraphrase with the same policy; M is system accuracy minus the policy's accuracy on the same raw set, and M_{orig} is SYN plus the policy on the original items. Four amendments were committed before the outputs they concern: the only available GPUs held another user's idle process, and vLLM 0.8.3's V1 engine refuses to start there, so the run uses its V0 engine (Amendment 1; on the original items this engine gives the untuned policy 54.00% against ROBUST-P2's 54.44% under V1, and M_{orig} +42.69 against +42.31); and two secondary policies, the untuned 7B and the R1-distilled 14B, were added on the same conditions as secondary cells (Amendment 2; both at `gpu_memory_utilization` 0.68 against the primary's 0.50; the distilled run, pre-empted under the V0 engine, was restarted under the V1 engine on the freed GPU, Amendment 3, and then sharded over two GPUs by prompt index as in ROBUST-P4's Amendment 1, Amendment 4). Both secondaries are reported below. Predictions: H-NP1, the policy's accuracy on each raw set within five points of original (point estimate, 5 cells); H-NP2, with normalization M within five points of M_{orig} on each set (point estimate, 5); H-NP3, without normalization M at least ten points below M_{orig} on each set (point estimate, 5); H-NP4, SYN-R's M minus SYN-NT's with a CI lower bound above zero on each set (5). **All** 20 **primary cells held.** The policy's accuracy is 54.00% originally and 57.00/55.12/53.81/56.25/56.50% on raw sets 1–5 (changes +3.00 [+0.50, +5.62], +1.12, −0.19, +2.25, +2.50; truncation 0.1–0.3%). M_{orig} is +42.69 [+40.38, +45.12]. With normalization, M is +40.12/ + 41.56/ + 43.12/ + 40.94/ + 40.00 on sets 1–5 (differences from M_{orig} +2.56 [+0.12, +5.06], +1.12, −0.44, +1.75, +2.69 [+0.25, +5.12]): the front-end restores the checker's contribution to within three points end-to-end. Because the normalized outcomes equal the original ones on every item (Appendix Y.1), each difference is exactly the change, over all 1,600 items, in how many checker-committed items the policy answers correctly, so the two that exclude zero (sets 1 and 5) reflect the policy doing better on those paraphrases, not a front-end loss. Without normalization, M falls to +5.62/ + 22.19/ + 13.06/ + 16.94/ + 18.44 (drops +37.06, +20.50, +29.62, +25.75, +24.25; every interval above +17). SYN-R's end-to-end M is +31.06/+34.62/+27.88/+36.75/+31.25 against SYN-NT's +5.62/ + 21.88/ + 13.06/ + 11.69/ + 18.50 (differences +25.44, +12.75, +14.81, +25.06, +12.75, all with lower bounds above +10); SYN-R therefore recovers 65–86% of M_{orig} on these sets without any front-end. Set 3's note-completion line is not task-preserving (Appendix Y.1); its cells are reported as registered. Under the lenient parser (registered secondary; its print was added to the analysis after the strict results) the policy scores 54.25% originally and 57.25/55.56/54.12/56.56/56.69% on sets 1–5, M_{orig} is +42.44, normalized M +39.88/+41.19/+42.88/+40.69/+39.81, raw M +5.62/+22.12/+13.00/+16.94/+18.38, and SYN-R minus SYN-NT +25.19/+12.50/+14.75/+24.81/+12.62. **Secondary cells, untuned 7B (all** 20 **held)**: the policy scores 45.56% originally and 45.12/46.69/45.00/45.50/42.88% on sets 1–5 (changes between −2.69 and +1.12; truncation at most 0.4%); M_{orig} +50.88 [+48.31, +53.38]; normalized M +50.56/ + 49.12/ + 51.00/ + 50.25/ + 52.88 (differences +0.31, +1.75, −0.12, +0.62, −2.00, every interval including zero); raw M +7.00/ + 26.88/ + 15.12/ + 20.38/ + 22.75 (drops +24.00 to +43.88); SYN-R M +38.25/ + 40.75/ + 33.12/ + 45.31/ + 40.56 against SYN-NT's +7.00/ + 26.56/ + 15.06/ + 14.12/ + 22.81 (differences +14.19 to +31.25, lower bounds above +11). **Secondary cells, R1-distilled 14B (**16 **of** 20 **held)**: the policy scores 76.19% originally and 74.81/75.62/73.81/77.19/75.88% on sets 1–5 (changes −2.38 to +1.00; truncation 16.5–19.1% in every condition, as in ROBUST-P2); H-NP1 held 5/5. M_{orig} is +20.62 [+18.69, +22.75]; normalized M is +22.38/ + 21.50/ + 23.06/ + 19.81/ + 21.19, within 2.44 points of M_{orig} in every set (H-NP2 5/5). Without normalization M falls to

$+4.81/+15.19/+8.19/+10.81/+10.88$, drops of $+15.81$, $+5.44$, $+12.44$, $+9.81$ and $+9.75$, so H-NP3 held only on sets 1 and 3 ($2/5$): against this stronger policy SYN loses less, since the policy is already right on 77–85% of the items SYN stops answering, against 53–60% for the untuned 14B (post-hoc, `bench/robustnp/robustnp_posthoc.py`). SYN-R's M exceeds SYN-NT's by $+11.81$, $+0.62$ $[-0.69, +1.94]$, $+2.81$, $+9.94$ and $+4.69$ (H-NP4 $4/5$; set 2, where SYN-NT keeps 51.3% coverage, is the exception). The lenient secondary gives a similar pattern, with a ten-point drop on set 1 only (M_{orig} $+15.69$; normalized $+15.56$ to $+18.56$; raw $+3.75$ to $+12.38$). Sources: `bench/robustnp/robustnp_run.py`, `robustnp_analysis.py` (run with `RNP_IN=bench/robustnp/results/policy_14b_untuned.json RNP_DIR=bench/robustnp/results`), `robustnp_results.txt`; raw completions in `bench/robustnp/results/`.

Y.4 ROBUST-P/P7: THE FIRST POLICY TESTS (PRE-REGISTERED, LEGACY PARSER)

Scoring status. These first runs motivated the corrected design in Appendix Y.5: strict extraction, a larger budget, symmetric key spellings and a matched trained fallback. We retain their original registered outcomes separately from the post-hoc validity corrections. All registered outcomes and tables in this subsection use the legacy lenient parser; the strict re-scoring below is post-hoc. The legacy single-model parser falls back to the last bare A–D letter when no `Answer: X` line appears. In music these letters can be note names, so truncated reasoning can receive credit without stating an answer. Table 4's historical Qwen runs use the same parser (the harness's vote parser has the same bare-letter fallback) but retained no raw text for re-scoring. ROBUST-P/P7 retain full completions, allowing us to separate registered reporting from this scoring correction.

Design and provenance. The registration `bench/robust/PREREGISTRATION_robust_policy.md` was frozen before any policy output on a perturbed item. It uses ROBUST's original items, both rephrasings, and repaired key distractors, rebuilt unchanged by `robust_run.conditions()`; the header condition is not run for policies. The answer policies are untuned Qwen2.5-14B-Instruct and its R1-SFT LoRA (Table 4's + *R1-SFT*, called R1-distilled here; not the mixed-domain model of the harness), the pair used for the distillation-gain split. Both use the single-model protocol (raw ABC, unchanged option order, `Answer: X`), greedy decoding on vLLM 0.8.3 with `EVAL_LEGACY_SP=1`, a 2,560-token output cap and 4,096-token context, bf16, one model at a time on one GPU. Identical prompts across conditions are generated once and shared.

All conditions, including the original, were regenerated in one run on one stack. Original overall accuracies here are $54.12/75.19$ for the untuned/R1-distilled 14B, not the earlier $55.9/76.2$, and the headline system scores 96.88% on the original wording, not the 96.81% of the registered SYN run, because its fallback answers were regenerated. All perturbation comparisons are paired within this run. This matters because the R1-distilled model is stack-sensitive. The headline system combines frozen SYN with the untuned policy's answer from the same condition on abstentions. Intervals are paired item bootstraps, 2,000 resamples, seed 0, 95% percentile intervals.

The 7B replication (ROBUST-P7). A second registration, `bench/robust/PREREGISTRATION_robust_policy7b.md`, was frozen before any 7B policy output on any ROBUST item, while the 14B run was in progress: the untuned 14B's per-condition overall accuracies had been printed, and no R1-distilled 14B output had been seen. It repeats the design at 7B: `robust_policy7b.py` imports `robust_policy.py` unchanged and sets only the base model (Qwen2.5-7B-Instruct), its R1-SFT LoRA and the output directory, and the analysis is `robust_policy_analysis.py` unchanged. H-P1 (with its branches and wording), H-P2 and H-P3 are registered exactly as at 14B, to be reported side by side with the 14B results and not pooled. At 7B the headline system is SYN's frozen checker with the untuned 7B on abstentions; the unchanged script's printed label still reads "untuned 14B". Original overall accuracies in this run are $46.06/63.19$ for the untuned/R1-distilled 7B, not the earlier $44.2/64.8$.

Size	Condition	Overall accuracy (%)		Key identification (%)	
		Untuned	R1-distilled	Untuned	R1-distilled
14B	Original	54.12	75.19	50.85	95.74
	Rephrasing 1	56.94	75.50	53.12	95.74
	Rephrasing 2	55.25	75.44	55.68	94.32
	Repaired key distractors	49.88	73.12	31.53	86.36
7B	Original	46.06	63.19	44.89	81.25
	Rephrasing 1	45.12	65.19	39.49	86.65
	Rephrasing 2	48.19	63.88	44.32	83.81
	Repaired key distractors	43.69	60.44	34.09	68.75

H-P1, primary: branch C at both sizes, in its pre-written words. *The distilled model's key-identification accuracy partly depends on how the generator builds distractors.* At 14B the R1-distilled model's accuracy drops from 95.74% to 86.36%: $+9.38$ $[+5.40, +13.64]$ on 352 key-identification items. At 7B it drops from 81.25% to 68.75%: $+12.50$ $[+5.97, +18.47]$. Branch A required the drop's lower CI bound to be at least 10 points; branch B required its upper bound to be at most 5; all other outcomes return C. Table 16 also reports the quantities required whatever the branch: the untuned model's drop and the distillation gain under each option set, each with its interval. The registrations set no test for comparing those gains. Moreover, the repair preserves gold spellings while drawing distractors from a different spelling list: $27/352$ items are answerable from spelling alone (the spellings `Db` and `G#m` occur only as gold; `bench/robust/robust_spelling_leak.txt`). This post-hoc validity finding applies to both ROBUST-P and ROBUST-P7, under either parser. The tested repair does not eliminate every options-only cue.

Table 16: **ROBUST-P: key-identification drops and distillation gains at 14B and 7B, legacy parser.** All rows were pre-registered at each size (separate registrations, not pooled). Drops are original minus repaired accuracy; gains are R1-distilled minus untuned. Values are percentage points with paired 95% intervals.

Contrast	14B	7B
R1-distilled drop (H-P1, primary)	$+9.38$ $[+5.40, +13.64]$	$+12.50$ $[+5.97, +18.47]$
Untuned drop	$+19.32$ $[+12.78, +25.85]$	$+10.80$ $[+4.26, +17.05]$
Distillation gain, original options	$+44.89$ $[+39.77, +50.00]$	$+36.36$ $[+30.40, +42.61]$
Distillation gain, repaired options	$+54.83$ $[+49.15, +60.51]$	$+34.66$ $[+28.41, +41.19]$

H-P2 and H-P3: wording effects. H-P2 predicts each policy's overall accuracy to remain within five points of its original accuracy under each rephrasing; it held $4/4$ at each size. H-P3 predicts at least a 20-point drop for the headline system under each rephrasing; it held $2/2$ at each size. The 14B system falls from 96.88% to 61.38% and 62.94%; the 7B system (SYN with the untuned 7B) falls from 95.94% to 51.56% and 56.31%. These held/not-held rules use point estimates, with intervals reported, as registered.

Size	Prediction / system	Rephrasing	Change (pp)	95% CI	Outcome
14B	H-P2 / untuned	1	$+2.81$	$[+0.25, +5.19]$	held
	H-P2 / untuned	2	$+1.12$	$[-1.31, +3.75]$	held
	H-P2 / R1-distilled	1	$+0.31$	$[-1.81, +2.69]$	held
	H-P2 / R1-distilled	2	$+0.25$	$[-1.88, +2.56]$	held
	H-P3 / SYN + untuned (drop)	1	$+35.50$	$[+33.19, +38.00]$	held
	H-P3 / SYN + untuned (drop)	2	$+33.94$	$[+31.50, +36.19]$	held
7B	H-P2 / untuned	1	-0.94	$[-3.56, +1.75]$	held
	H-P2 / untuned	2	$+2.12$	$[-0.75, +4.94]$	held
	H-P2 / R1-distilled	1	$+2.00$	$[-0.81, +4.62]$	held
	H-P2 / R1-distilled	2	$+0.69$	$[-2.00, +3.44]$	held
	H-P3 / SYN + untuned (drop)	1	$+44.38$	$[+42.00, +46.75]$	held
	H-P3 / SYN + untuned (drop)	2	$+39.62$	$[+37.06, +42.19]$	held

H-P2 changes are rephrased minus original; H-P3 drops are original minus rephrased. SYN's coverage in these three conditions is $93.4\%/12.9\%/12.5\%$ at both sizes (the checker is the same).

Strict answer extraction (post-hoc re-scoring). We re-score the committed completions by the last explicit `Answer: X` match, counting no match as wrong, and retain the paired item bootstrap (2,000 resamples, seed 0). This changes extraction, not the frozen models or generations. Of 1,600 original R1-distilled 14B completions, 428 lack an answer line and 121 were credited by the legacy parser; the corresponding 7B counts are 568 and 149. For rephrasings 1, 2 and key repair the 14B counts are $402/109$, $443/128$ and $453/130$ (missing/credited); at 7B they are $541/152$, $531/138$ and $591/162$. Many completions reach the 2,560-token cap. These are missing-answer counts, not measurements that every such output was truncated. The extraction issue also affects untuned outputs, though less often; all counts are in `bench/robust/robust_policy_strict.txt`.

		Overall accuracy (%)		Key identification (%)	
Size	Condition	Untuned	R1-distilled	Untuned	R1-distilled
14B	Original	53.62	67.62	50.28	93.75
	Rephrasing 1	56.38	68.69	50.85	94.89
	Rephrasing 2	54.81	67.44	55.68	94.03
	Key repair	49.50	65.00	31.53	81.82
7B	Original	45.81	53.88	44.89	78.69
	Rephrasing 1	44.88	55.69	39.49	81.82
	Rephrasing 2	47.75	55.25	44.32	81.82
	Key repair	43.44	50.31	34.09	62.50

Applying the frozen H-P1 decision rule to the strict scores still gives branch C at both sizes: the drops are $+11.93$ $[+7.39, +16.19]$ at 14B and $+16.19$ $[+9.94, +22.16]$ at 7B. H-P2 also survives in every cell under its registered point-estimate rule, with the following changes (rephrased minus original; all values here are post-hoc re-scores).

Policy	Rephrasing	14B change (pp), 95% CI	7B change (pp), 95% CI
Untuned	1	$+2.75$ $[+0.19, +5.12]$	-0.94 $[-3.62, +1.75]$
	2	$+1.19$ $[-1.31, +3.81]$	$+1.94$ $[-0.88, +4.81]$
R1-distilled	1	$+1.06$ $[-1.06, +3.38]$	$+1.81$ $[-0.69, +4.38]$
	2	-0.19 $[-2.38, +2.12]$	$+1.38$ $[-1.19, +3.94]$

The strictly scored key-identification distillation gain is $+43.47$ $[+37.78, +48.86]$ originally and $+50.28$ $[+44.32, +56.25]$ under repair at 14B; at 7B it is $+33.81$ $[+27.84, +39.77]$ and $+28.41$ $[+22.16, +34.94]$. The gain survives the tested repair at both sizes. Its change, however, is not significant: $+6.82$ $[-1.14, +14.77]$ at 14B and -5.40 $[-13.92, +3.41]$ at 7B. Neither interval excludes zero. The remaining spelling cue also prevents a claim that this repair isolates artifact-free key identification.

Ranking depends on the fallback (post-hoc, strict scoring). Table 17 compares the R1-distilled policy alone with SYN using either policy as fallback, on the same completions. Against the untuned fallback, the reversal first seen under the legacy parser ($+14.12$, $+12.50$ at 14B; $+13.62$, $+7.56$ at 7B; `bench/robust/robust_policy_posthoc.txt`) shrinks under strict scoring, which leaves it under both 14B rephrasings but only the first 7B rephrasing. With the matched R1-distilled fallback, SYN remains ahead in every condition. Its contribution shrinks from 29–41 points on original wording to 3.5–6.4 points under rephrasing. Thus the checker's contribution is template-bound, while a reversal is a fragile observation about one fallback pairing. The policies' stability under these rephrasings is behavioral evidence, not identification of their internal reasoning mechanism.

Sources: `bench/robust/robust_policy_results.txt` (14B) and `bench/robust/robust_policy7b_results.txt` (7B) retain the registered legacy-parser outcomes. Per-item outputs and raw completions are in `bench/robust/results/` and `bench/robust/results7b/` (`policy_untuned.json`, `policy_distilled.json`). Registered analysis: `bench/robust/robust_policy_analysis.py` with `RP_IN` set to the corresponding

Table 17: **Strict-scoring comparisons with both fallbacks (post-hoc).** Differences are R1-distilled policy alone minus SYN plus the indicated same-size fallback, in percentage points with paired 95% intervals. Negative values favor the checker system.

SYN fallback	Condition	14B	7B
Untuned	Original	−29.12 [−31.50, −26.88]	−42.06 [−44.62, −39.69]
	Rephrasing 1	+7.88 [+4.81, +10.94]	+4.38 [+1.19, +7.38]
	Rephrasing 2	+4.94 [+1.62, +7.94]	−0.69 [−3.88, +2.75]
R1-distilled	Original	−29.06 [−31.31, −27.06]	−41.25 [−43.75, −38.94]
	Rephrasing 1	−5.06 [−6.12, −4.06]	−6.19 [−7.31, −5.06]
	Rephrasing 2	−3.50 [−4.44, −2.62]	−6.38 [−7.62, −5.25]

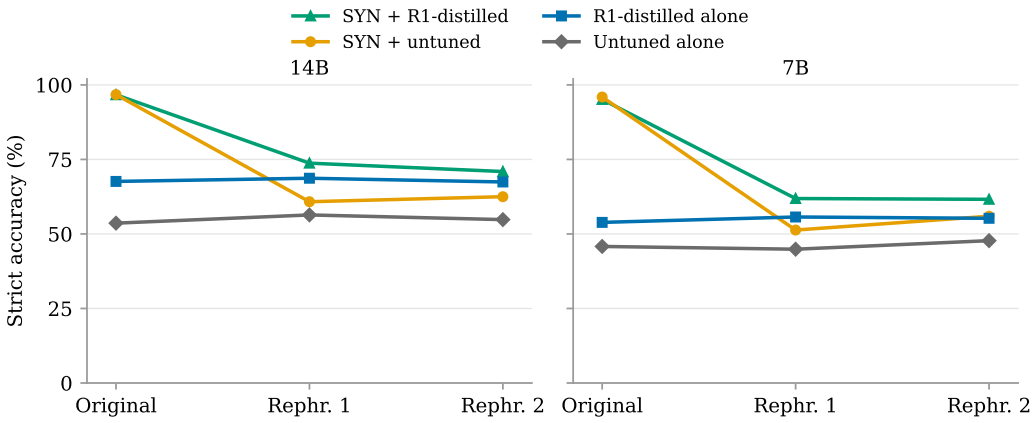


Figure 10: **The checker loses most of its contribution under rephrasing (post-hoc strict re-scoring of ROBUST-P/P7).** Sizes are separate, not pooled. SYN with a matched R1-distilled fallback stays above that policy alone; the gap shrinks sharply. A reversal against SYN with an untuned fallback survives both rephrasings at 14B but only the first at 7B (Table 17). Points use the same committed completions; all differences from the registered legacy scores are post-hoc.

directory. Post-hoc strict analysis: `bench/robust/robust_policy_strict.py`; output: `bench/robust/robust_policy_strict.txt`.

Y.5 ROBUST-P2: CORRECTED POLICY TEST (PRE-REGISTERED)

Design and provenance. `bench/robust/PREREGISTRATION_robust_policy2.md`, frozen at `fddbaf1` before any output, responds to the first policy tests' post-hoc validity findings: lenient extraction, the 2,560-token cap, gold-only spellings on 27/352 repaired key items, and a ranking comparison with an unmatched fallback. It was written after the first policy runs, and it cites a legacy-parser recombination of the 14B completions in which SYN with the R1-distilled fallback scores 79.25/78.75 under the two rephrasings, against 75.50/75.44 for that policy alone (a recombination of `bench/robust/results/policy_distilled.json` with SYN's outcomes in `bench/robust/syn_conditions.json`, quoted in `bench/robust/PREREGISTRATION_robust_policy2.md`). H-Q1 is therefore a confirmatory re-test with new generations, strict extraction, a larger budget and a new repair, not a blind prediction. The post-hoc strict re-scoring of the first runs shows the same pattern at both sizes (Table 17); its commit (`b1773e0`) follows this registration's by three minutes. The new primary uses the last explicit `Answer: X` match (case-insensitive, X in A–D), counting no match as wrong. The legacy lenient parser is a registered secondary. The output budget is 8,192 tokens and the context limit 12,288. Untuned Qwen2.5-Instruct and its R1-SFT adapter run at 14B and 7B, greedily on vLLM 0.8.3 with `EVAL_LEGACY_SP=1`, one model at a time, with unchanged prompts. All conditions, including original, are regenerated on that stack.

Original and both rephrasings are unchanged from the checker test. The symmetric repair changes only the 352 key-identification items: three distractors are sampled with seed 2027 from the generator's own 24 gold key spellings, excluding the gold tonic/mode. Gold spelling and letter remain fixed; all four options now share the same vocabulary. This removes the identified gold-only-spelling cue. It does not claim to remove every possible option artifact. Frozen SYN is combined with each same-size, same-condition policy on abstentions. No policy, prompt, parser, budget, condition, seed or statistic changes after output begins. Intervals are paired item-bootstrap 95% intervals, 2,000 resamples, seed 0. H-Q1 and H-Q2 use point-estimate decision rules; H-Q3 and H-Q4 use the registered interval rules.

H-Q1 (primary): the checker's value is template-bound — held $4/4$. Let M be SYN plus the R1-distilled fallback minus that same policy alone. Each rephrasing must reduce M by at least ten points. Table 18 gives every contribution and drop. The matched checker system stays ahead under each rephrasing, but loses most of its original advantage.

Table 18: **Corrected policy test: matched checker contribution (H-Q1, primary strict scoring).** Values are percentage points with paired 95% intervals. All four rephrasing predictions held.

Size	Condition	M, 95% CI	Drop from original, 95% CI
14B	Original	$+21.12$ $[+19.25, +23.25]$	—
	Rephrasing 1	$+4.19$ $[+3.31, +5.12]$	$+16.94$ $[+15.00, +19.06]$
	Rephrasing 2	$+3.19$ $[+2.38, +4.12]$	$+17.94$ $[+16.00, +20.00]$
7B	Original	$+36.69$ $[+34.31, +39.06]$	—
	Rephrasing 1	$+5.00$ $[+3.94, +6.06]$	$+31.69$ $[+29.12, +34.19]$
	Rephrasing 2	$+6.31$ $[+5.19, +7.50]$	$+30.38$ $[+27.94, +32.88]$

H-Q2: policies are not template-bound — held $8/8$. Each policy's rephrased accuracy must remain within five points of original. All eight changes are within three points; the largest absolute change is 2.81. This is the registered point-estimate rule, not a requirement that the full interval lie inside the band.

Size	Policy	Rephrasing 1 change, 95% CI	Rephrasing 2 change, 95% CI
14B	Untuned	-0.44 $[-3.12, +2.00]$	$+0.94$ $[-1.62, +3.38]$
	R1-distilled	-0.25 $[-2.31, +1.81]$	$+0.00$ $[-2.25, +2.19]$
7B	Untuned	-2.19 $[-4.88, +0.44]$	$+2.81$ $[+0.12, +5.50]$
	R1-distilled	-0.19 $[-2.69, +2.44]$	$+0.44$ $[-2.19, +3.06]$

H-Q3: distillation's key-identification gain survives a clean repair — held $2/2$. The lower interval bound of R1-distilled minus untuned key accuracy under symmetric repair must exceed zero. Both sizes satisfy this rule (Table 19). At 14B the gain is $+48.58$ $[+42.61, +54.55]$, compared with $+42.61$ under original options; at 7B it is $+35.51$ $[+28.98, +42.61]$, also $+35.51$ originally. The registered test concerns a positive gain after repair, not a change in the size of that gain.

H-Q4: distractor dependence — branch C at both sizes. In the registered wording: *The distilled model's key-identification accuracy partly depends on how the generator builds distractors.* The original-minus-repaired drop is $+6.53$ $[+3.12, +10.23]$ at 14B and $+11.08$ $[+4.83, +16.48]$ at 7B. Branch A requires the lower bound to be at least ten points; B requires the upper bound to be at most five; otherwise C applies. The untuned drops are $+12.50/+11.08$.

Ranking against the untuned fallback (registered secondary, reported either way). The R1-distilled policy alone is below SYN plus an untuned fallback on original wording and above it by point estimate in all four rephrased cells. Three rephrased intervals exclude zero; 7B rephrasing 2 does not. Table 20 gives the full comparison. This reversal depends on the fallback pairing: with its own distilled fallback, SYN remains ahead of the policy alone in all four cells.

Accuracy, truncation and checker behavior (registered descriptive reports; overall gains post-hoc). Strict original accuracy is $54.44/75.81\%$ for untuned/R1-distilled 14B and $46.88/58.88\%$

Table 19: **Symmetric key repair: gains and dependence (strict).** All contrasts are registered on the 352 key-identification items; values are points with paired 95% intervals.

Contrast	14B	7B
Distillation gain, original	+42.61 [+37.22, +48.01]	+35.51 [+29.55, +41.76]
Distillation gain, repair (H-Q3)	+48.58 [+42.61, +54.55]	+35.51 [+28.98, +42.61]
R1-distilled drop (H-Q4)	+6.53 [+3.12, +10.23]	+11.08 [+4.83, +16.48]
Untuned drop	+12.50 [+6.25, +18.47]	+11.08 [+4.55, +16.76]

Table 20: **Fallback-dependent ranking (registered secondary, strict).** R1-distilled alone minus SYN plus the untuned same-size fallback, in points with paired 95% intervals.

Condition	14B	7B
Original	−20.94 [−23.12, −18.88]	−37.38 [−39.81, −34.88]
Rephrasing 1	+16.62 [+13.69, +19.69]	+7.44 [+4.31, +10.44]
Rephrasing 2	+13.06 [+10.19, +15.88]	+1.94 [−1.19, +5.12]

at 7B. The overall distillation gain on these items was not registered; as a post-hoc paired contrast (same bootstrap) it is +21.38 [+18.50, +24.25] at 14B and +12.00 [+8.94, +15.12] at 7B under strict extraction, and +25.69 [+22.81, +28.62] and +19.06 [+16.06, +22.19] under the lenient parser (`bench/robust/robust_policy2_gain.txt`). These are new runs, distinct from the historical legacy-scored rows in Table 4. The larger budget still leaves 17.1% of original distilled 14B completions and 28.6% at 7B at the cap. Tables below report every condition and both parsers; truncation uses the recorded `length` finish reason.

Size	System	Original	Rephrasing 1	Rephrasing 2	Symmetric repair
14B	Untuned	54.44/54.88	54.00/54.62	55.38/55.81	51.69/51.94
	R1-distilled	75.81/80.56	75.56/80.12	75.81/80.12	74.38/79.31
	SYN + untuned	96.75/96.75	58.94/59.56	62.75/63.19	95.38/95.38
	SYN + distilled	96.94/97.81	79.75/83.44	79.00/83.25	95.56/96.44
7B	Untuned	46.88/47.00	44.69/44.94	49.69/49.94	44.44/44.56
	R1-distilled	58.88/66.06	58.69/66.50	59.31/67.44	56.44/63.94
	SYN + untuned	96.25/96.25	51.25/51.50	57.38/57.62	94.88/94.88
	SYN + distilled	95.56/96.62	63.69/70.38	65.62/72.81	94.19/95.25

Entries are overall accuracy (%), strict / lenient.

Size	Policy	Original	Rephrasing 1	Rephrasing 2	Symmetric repair
14B	Untuned	53.69/54.55	49.15/51.42	57.95/57.95	41.19/41.19
	R1-distilled	96.31/96.59	97.16/97.44	96.31/96.31	89.77/90.91
7B	Untuned	46.59/46.59	39.49/39.49	50.85/50.85	35.51/35.51
	R1-distilled	82.10/84.09	85.23/88.35	79.83/82.10	71.02/74.43

Entries are key-identification accuracy (%), strict / lenient.

Size	Policy	Original	Rephrasing 1	Rephrasing 2	Symmetric repair
14B	Untuned	0.1	0.1	0.0	0.1
	R1-distilled	17.1	17.1	17.1	17.8
7B	Untuned	0.0	0.3	0.1	0.0
	R1-distilled	28.6	28.2	28.0	29.4

Entries are percentages reaching the 8,192-token cap.

On the symmetric repair's 352 key items, checker coverage / precision is 56.0%/88.83% for v1, 87.8%/100.00% for v2, 100.0%/93.75% for SYN, and 46.6%/25.61% for the options-only rule (registered secondary). These are distinct from the first repair's registered outcomes; SYN still covers every key item but makes errors.

Legacy parser (registered secondary). Applying the frozen decision rules to lenient scores gives H-Q1 $4/4$, H-Q2 $8/8$, H-Q3 $2/2$, and H-Q4 branch C at both sizes, with the same registered sentence as above. Every lenient contrast follows; the strict results remain primary.

Size	Condition	M, 95% CI	Drop from original, 95% CI
14B	Original	$+17.25$ $[+15.50, +19.12]$	—
	Rephrasing 1	$+3.31$ $[+2.44, +4.19]$	$+13.94$ $[+12.12, +15.88]$
	Rephrasing 2	$+3.12$ $[+2.31, +4.00]$	$+14.12$ $[+12.31, +16.06]$
7B	Original	$+30.56$ $[+28.25, +32.81]$	—
	Rephrasing 1	$+3.88$ $[+2.94, +4.81]$	$+26.69$ $[+24.38, +29.06]$
	Rephrasing 2	$+5.38$ $[+4.31, +6.50]$	$+25.19$ $[+22.88, +27.56]$

Matched checker contributions and drops (H-Q1), points.

Size	Policy	Rephrasing 1 change, 95% CI	Rephrasing 2 change, 95% CI
14B	Untuned	-0.25 $[-2.94, +2.19]$	$+0.94$ $[-1.62, +3.38]$
	R1-distilled	-0.44 $[-2.38, +1.50]$	-0.44 $[-2.69, +1.75]$
7B	Untuned	-2.06 $[-4.75, +0.56]$	$+2.94$ $[+0.25, +5.69]$
	R1-distilled	$+0.44$ $[-2.12, +2.88]$	$+1.38$ $[-1.31, +4.19]$

Policy changes (H-Q2), points.

Contrast	14B	7B
Distillation gain, original	$+42.05$ $[+36.65, +47.44]$	$+37.50$ $[+31.82, +43.47]$
Distillation gain, repair (H-Q3)	$+49.72$ $[+44.03, +55.68]$	$+38.92$ $[+32.67, +45.74]$
R1-distilled drop (H-Q4)	$+5.68$ $[+2.27, +9.38]$	$+9.66$ $[+3.98, +15.06]$
Untuned drop	$+13.35$ $[+7.10, +19.60]$	$+11.08$ $[+4.55, +16.76]$

Key-identification contrasts, points with paired 95% intervals.

Condition	14B	7B
Original	-16.19 $[-18.25, -14.12]$	-30.19 $[-32.62, -27.69]$
Rephrasing 1	$+20.56$ $[+17.69, +23.44]$	$+15.00$ $[+11.94, +17.94]$
Rephrasing 2	$+16.94$ $[+14.06, +19.81]$	$+9.81$ $[+6.69, +12.94]$

R1-distilled alone minus SYN plus untuned fallback, points with paired 95% intervals. All four lenient rephrased intervals exclude zero; strict extraction supports that reversal in three.

Artifacts and reproduction. The full report is `bench/robust/robust_policy2_results.txt`. `bench/robust/results_p2/` contains each policy's raw completions, strict and lenient scores, and finish reasons. From the repository root, set `RP2_IN=bench/robust/results_p2` and run `python3 bench/robust/robust_policy2_analysis.py`, with the frozen checker's sandbox and run-host path dependencies described in the Reproducibility Statement. The analysis reuses the original/rephrased SYN outcomes and executes the unchanged checker on the symmetric repair. The post-hoc overall gains come from `bench/robust/robust_policy2_gain.py`.

Truncation dependence (post-hoc). After the registered analysis, a reviewer noted that the R1-distilled policy still stops at the 8,192-token cap on $273/1{,}600$ (14B) and $457/1{,}600$ (7B) original items, and that these carry 235 of SYN's 338 net matched gains at 14B and 409 of 587 at 7B ($\sim$70%). Restricted to items the policy finishes in both conditions, the drop of the matched contribution is $+7.42$ $[+6.02, +8.90]$ and $+4.44$ $[+3.10, +5.86]$ at 14B (1,213 and 1,195 items) and $+13.21$ $[+10.89, +15.54]$ and $+8.90$ $[+6.75, +11.04]$ at 7B (946 and 933): every interval is above zero, but three of the four point estimates (both 14B drops and 7B rephrasing 2) fall below the registered ten-point threshold, and only 7B rephrasing 1 clears it: the registered magnitudes are larger than on these subsets, though the analysis does not identify how truncated items would score without

truncation. The truncation rates are a registered secondary report; the gain attribution and these subsets are post-hoc. Condition-specific subsetting is itself a selection, so these are sensitivity figures, not a replacement for H-Q1. Source: `bench/robust/robust_policy2_truncation.py` and its `.txt`.

ROBUST-P3: a repetition-penalty re-run (pre-registered). After the truncation analysis above had been seen, we froze a follow-up test (`bench/robust/PREREGISTRATION_robust_policy3.md`) that reruns the R1-distilled policies on the original and both rephrased conditions with everything as above except a repetition penalty of 1.05 (the value in the base models' generation configuration), with SYN's frozen outcomes and the same-size distilled policy on its abstentions. Under strict extraction it registers H-T1, a manipulation check that at most 5% of items end at the token cap in each of six size-condition cells; H-T2 (primary), that the drop of M from original has a CI lower bound above zero in each of four cells; and H-T3, that the policy's accuracy under each rephrasing is within five points of original (point estimate). **H-T1 held in** $0/6$ **cells**: $13.7/12.1/14.9\%$ of items end at the cap at 14B and $24.8/21.5/22.1\%$ at 7B (original/rephrasing 1/2), below ROBUST-P2's 17.1% (14B, each condition) and 28.0–28.6% (7B) but not negligible, so the test does not establish the low-truncation condition it was designed for. No answer is extracted on $219/193/238$ (14B) and $396/344/353$ (7B) items, all of them truncated. **H-T2 held in** $4/4$ **cells**: M falls from $+17.75$ $[+16.06, +19.69]$ to $+2.75/+2.69$ at 14B and from $+34.00$ $[+31.75, +36.44]$ to $+4.25/+5.38$ at 7B, drops of $+15.00$ $[+13.19, +17.00]$ and $+15.06$ $[+13.19, +17.00]$ at 14B and $+29.75$ $[+27.44, +32.25]$ and $+28.62$ $[+26.25, +31.06]$ at 7B (ROBUST-P2: $+16.94/+17.94$ and $+31.69/+30.38$). On items the policy finishes in both conditions (a registered report; the subsetting is condition-specific, as above), the drops are $+6.56$ $[+5.17, +8.11]$ and $+4.11$ $[+2.69, +5.53]$ at 14B (1,295 and 1,266 items) and $+12.00$ $[+9.81, +14.19]$ and $+10.05$ $[+7.90, +12.20]$ at 7B (1,050 and 1,025), all above zero (ROBUST-P2, post-hoc: $+7.42/+4.44$ and $+13.21/+8.90$). **H-T3 held in** $4/4$ **cells** (changes $+1.81$, -1.62 at 14B; -0.50, $+0.88$ at 7B). Strict accuracy is $79.19/81.00/77.56\%$ at 14B and $62.00/61.50/62.88\%$ at 7B; on original items the penalty raises it by $+3.38$ and $+3.12$ over ROBUST-P2 (unlike the stack probe of Appendix B.4, a different checkpoint and parser, where settings including a repetition penalty lowered the score). The registered lenient secondary, which the frozen analysis script omitted and we computed after the strict results with the same statistics, gives drops of $+12.69$ $[+11.00, +14.50]$ and $+11.88$ $[+10.12, +13.62]$ at 14B and $+24.81$ $[+22.56, +27.12]$ and $+23.19$ $[+21.00, +25.56]$ at 7B, with policy changes of at most 1.50 points. Sources: `bench/robust/robust_policy3.py`, `robust_policy3_analysis.py` (run with `RP3_IN=bench/robust/results_p3`), `robust_policy3_results.txt`, `robust_policy3_lenient.py` and its `.txt`; raw completions with finish reasons are in `bench/robust/results_p3/`.

ROBUST-P4: the distilled model's recommended sampling (pre-registered). Registered after ROBUST-P3's results (`bench/robust/PREREGISTRATION_robust_policy4.md`), this re-run keeps everything of ROBUST-P3 except the decoding: temperature 0.6 and top-p 0.95, the values DeepSeek recommends for R1-distilled models to avoid endless repetition, vLLM seed 0, no repetition penalty, 16,384 output tokens (one sample per prompt, so intervals cover item sampling, not decoding randomness). Amendment 1, committed before any output existed: the first single-GPU run was lost in a server reboot at $528/4{,}800$ prompts, so generation was sharded over GPUs by prompt index with the identical model, adapter, sampling parameters and per-request seed and merged (`--shard`, `--merge`); the per-request seed makes each prompt's draw independent of its batch up to floating-point nondeterminism in batched kernels. It registers H-S1, at most 5% of items at the cap (6 cells); H-S2 (primary), ROBUST-P2's rule that M drops by at least ten points (4 cells); H-S3, a CI lower bound above zero for each drop (4 cells); H-S4, policy accuracy within five points (4 cells). **H-S1 held in** $0/6$ **cells**: $10.3/9.7/10.3\%$ of items end at the cap at 14B and $15.1/15.8/14.2\%$ at 7B (original/rephrasing 1/2; no extracted answer on $165/154/165$ and $240/252/227$ items), below ROBUST-P3's 12.1–14.9% (14B) and 21.5–24.8% (7B) but again not negligible, so, as registered, this run again does not isolate truncation. **H-S2 held in** $4/4$: M falls from $+15.62$ $[+14.00, +17.38]$ to $+1.88/+3.19$ at 14B and from $+30.00$ $[+27.81, +32.38]$ to $+3.31/+5.31$ at 7B, drops of $+13.75$ $[+12.12, +15.56]$ and $+12.44$ $[+10.81, +14.19]$ at 14B and $+26.69$ $[+24.44, +29.12]$ and $+24.69$ $[+22.44, +26.94]$ at 7B (ROBUST-P2: $+16.94/+17.94$ and $+31.69/+30.38$). **H-S3 held in** $4/4$. On items finished in both conditions (registered report), the drops are $+6.94$ $[+5.69, +8.42]$ and $+4.09$ $[+2.75, +5.43]$ at 14B (1,354 and 1,345 items) and

+16.32 [+14.15, +18.48] and +12.80 [+10.66, +15.01] at 7B (1,201 and 1,219). **H-S4 held in** 4/4 (changes +0.69, −0.19 at 14B; −0.88, +0.50 at 7B). Strict accuracy is 81.94/82.62/81.75% at 14B and 66.88/66.00/67.38% at 7B; the registered lenient secondary gives drops of +11.44 [+9.81, +13.12] and +9.69 [+8.06, +11.31] at 14B and +23.94 [+21.69, +26.25] and +21.38 [+19.19, +23.56] at 7B, with policy changes of at most 0.56 points. Descriptively (post-hoc; the three runs also differ in decoding, not only in truncation), the all-item drops fall from +16.94/ + 17.94 to +15.00/+15.06 to +13.75/+12.44 at 14B and from +31.69/+30.38 to +29.75/+28.62 to +26.69/ + 24.69 at 7B as original truncation falls (17.1/13.7/10.3% and 28.6/24.8/15.1%), while the finished-in-both drops are +7.42/ + 4.44, +6.56/ + 4.11 and +6.94/ + 4.09 at 14B (between +4.1 and +7.4) and +13.21/ + 8.90, +12.00/ + 10.05 and +16.32/ + 12.80 at 7B. Sources: `bench/robust/robust_policy4.py`, `robust_policy4_analysis.py` (run with `RP4_IN=bench/robust/results_p4`), `robust_policy4_results.txt`; merged raw completions in `bench/robust/results_p4/`.

ROBUST-F: a second policy family on both audits (pre-registered). Registered at `bench/family/PREREGISTRATION_family.md` before any output, in response to a review asking whether the conclusions extend beyond Qwen2.5, this test runs untuned Llama-3.1-8B-Instruct (vLLM 0.8.3, greedy, strict extraction) through ROBUST-P2's SSMR-Bench runner (original, two rephrasings, symmetric repair; 8,192 tokens) and PW-AUDIT's policy runner (original, two stems, rule rewrites, statement only; 2,048 tokens), unchanged except for the model, with each benchmark's frozen primary checker committing and Llama answering its abstentions. ROBUST-F's **H-F1 held in** 2/2 **cells**: on SSMR-Bench, Llama's accuracy changes by −0.25 [−3.06, +2.56] and −0.81 [−3.69, +1.88] under the rephrasings (point-estimate rule). **H-F2 held in** 0/3 **cells**: on ProofWriter its changes are −3.17 [−7.67, +1.33] (stem 1), −2.17 [−7.00, +2.50] (stem 2) and −0.33 [−5.17, +4.33] (rules); every point estimate is inside the band but no interval lies wholly within [−5, +5], as the whole-interval rule requires. **H-F3 held in** 5/5 **cells**: the drop of M is +57.12 [+54.50, +59.62] and +55.94 [+53.38, +58.44] on SSMR-Bench and +52.17 [+47.67, +56.67] (each stem) and +43.33 [+37.00, +49.67] (rules) on ProofWriter. **H-F4 held**: statement-only accuracy is 29.33% [25.83, 33.00]. Llama is a weak policy on both benchmarks: strict accuracy is 31.38/31.12/30.56% on SSMR-Bench (original/rephrasing 1/2; 413/389/397 items without an extracted answer; 163/128/170 truncated) and 37.83% on original ProofWriter items (chance 33.3%), so on original items the checker's commits carry the system and the drops of M are larger than with Qwen in four of the five cells (under rule rewrites +43.33 against Qwen's +47.00/ + 48.83). Truncation and no-answer counts, which the registration lists but the frozen script prints only for SSMR-Bench (the ProofWriter counts below were computed afterwards from the committed outputs): SSMR-Bench symmetric repair 147 truncated, 400 without an answer; ProofWriter original/stem 1/stem 2/rules/statement-only 81/95/71/82/0 truncated and 122/132/119/119/20 without an extracted answer, of 600. Under rule rewrites the ProofWriter system scores 46.33% against Llama's 37.50%, so, unlike with Qwen, the rewritten system is not worse than this weaker fallback. Registered descriptive report: Llama's key-identification accuracy is 26.14% originally and 25.57% under the symmetric repair (+0.57 [−5.11, +5.97]). Sources: `bench/family/family_run.py`, `family_analysis.py` (run with `FAM_SSMR=bench/family/results/policy_llama8b_untuned.json FAM_PW=bench/family/results/policy_llama8b.json`), `family_results.txt`; raw completions in `bench/family/results/`.

Z PW-AUDIT: THE AUDIT ON PROOFWRITER (PRE-REGISTERED)

Registration and provenance. `bench/pwaudit/PREREGISTRATION_pwaudit.md`, frozen at `d46a8f1`, precedes every writer call, program test and policy output in this audit. It follows the SSMR-Bench audit and uses the paper's existing 600-item Logic-LM sample (Pan et al., 2023) of ProofWriter (Tafjord et al., 2021): depth-5, open-world, 200 items per gold label (True, False, Unknown), from 345 theories. The sample is a subset of the tasksource depth-5 validation split, not the split itself: each item matches a row of that split exactly (theory text, statement and label). This is distinct from the earlier ProofWriter residual re-generation experiment (Table 14). The checker, policy, partial-input and perturbation conditions here form one registered audit; the theory-blind pass, per-label results and per-run picks were specified as descriptive.

Training uses the tasksource depth-5 train split: 69,810 items from 3,322 theories, sharing neither theory IDs nor theory text with the test sample. A seeded split (`random.Random(2027)`) assigns 100 items per label to feedback F and 200 per label to selection V; no theory occurs twice across F and V. Ten worked examples per label are sampled from F. Thus the writer sees 30 worked examples and feedback on 300 items; V contains 600 items never shown to it.

Frozen synthesis and policy protocols. The experiment imports SYN's loop pinned to `d648c13`: three runs of six rounds, execution feedback, the same sandbox and two-process cyclic-option-relabelling commit rule. The writer is Claude Opus 4.8 at SYN-NT decoding: no extended thinking or effort setting, 8,192 output tokens per call. The task prompt is adapted from scores to theories and three answer options; it does not teach open-world semantics beyond the worked examples. Selection maximizes $J = \text{correct} - 1.27\text{wrong}$ on V, with ties broken by precision and then earlier run/round. The weight is retained from SYN for comparability, not calibrated to the ProofWriter policies. The primary is run 2, round 3 (99.83% coverage, 89.48% precision on V). Programs and selection metadata were committed at `b19cd73` before the single test pass at `ef0712a`; no per-run alternative is promoted after testing.

Untuned Qwen2.5-14B- and -7B-Instruct run greedily on vLLM 0.8.3 with a 2,048-token output budget and 6,144-token context limit. Every condition is generated in one run per size. Extraction uses the last explicit `Answer: X`; a completion without a match is scored wrong. System accuracy uses the primary checker's commits and the same-size, same-condition policy on abstentions. M is system minus policy accuracy. All intervals are paired item-bootstrap 95% intervals (2,000 resamples, seed 0), as registered; they resample items, not theories.

Perturbations and rewrite validation. Original, stem 1, stem 2 and rules conditions retain the gold labels. Both stems keep the queried statement verbatim. Stem 1 reads, "Using only the information above, decide whether the following statement is true, false, or unknown." Stem 2 reads, "Is the statement that follows true, false, or unknown according to the information given above?" The rule condition changes 2,320 of 2,782 rule sentences across the sample and changes all 600 items, leaving facts untouched. Four strict sentence-level rewrites replace universal quantifiers by Every/No, bare adjective rules by relative clauses, variable-subject if/then rules by Anything-that/Anyone-who clauses, and ground if/then rules by consequent-first sentences. Unmatched sentences remain unchanged.

Before registration, a separate reference parser and forward-chaining reasoner validated the rewrites on training theories only. Logical forms were identical for $3{,}322/3{,}322$ theories and answers for $69{,}810/69{,}810$ training items (the training parquet is the public tasksource release, pinned by sha256 in `bench/pwaudit/input_hashes.json` but not committed; the 600-item validation reruns from the repository); deliberately broken rewrites failed validation. The validator handles explicit negation: a negative condition requires its negative fact, and a statement is Unknown when neither it nor its negation follows. This validator was not shown to the writer or used as the evaluated checker. The policy-only statement probe removes the theory. The descriptive theory-blind checker pass replaces each original theory by another test theory from the same family.

H0: coverage and near-perfect precision — not held. The registered rule requires original coverage at least 80% and precision at least 99%. The primary commits on $600/600$ items with 60 errors: 100% coverage at 90.00% precision. This fails H0. High-precision program inversion on SSMR-Bench does not replicate with this loop on ProofWriter. The frozen objective selects a program with broad coverage and errors over the conservative run 0 program below; this does not limit what an independently constructed logic solver could do.

H1: checker template dependence — held for stems, not for rules. The rule requires the lower bound of the coverage drop to be at least 20 points, separately for each perturbation.

Condition	Coverage (%)	Precision (%)	Coverage drop (pp), 95% CI	H1
Original	100.00	90.00	—	—
Stem 1	0.00	—	$+100.00$ $[+100.00, +100.00]$	held
Stem 2	0.00	—	$+100.00$ $[+100.00, +100.00]$	held
Rule rewrites	100.00	46.33	0.00 $[0.00, 0.00]$	not held

No precision is defined for the stems because nothing commits. Rules produce 322 wrong commits, a descriptive precision loss of 43.67 points; the registration sets no precision-drop decision rule. The checker returns mostly "Unknown" under the rewritten rules. Because Unknown is a legitimate open-world answer, the metamorphic commit rule cannot detect a failure merely from its stable answer under option relabelling. H1's failed coverage prediction is therefore consequential: the checker continues to override its fallback while answering incorrectly.

Parsing versus solving (post-hoc). Three controls (`bench/pwaudit/pw_posthoc.txt`) locate these failures. (i) Restoring the original stem on the stem1/stem2 items makes the frozen checker reproduce its original behaviour exactly (100% coverage at 90.00%, the same answer on 600/600 items): the stem collapse is statement extraction. (ii) All 60 original and all 322 rule-condition errors are "Unknown" answers on True or False items (34/26 and 161/161): when the checker cannot complete a derivation it defaults to Unknown. No wrong True/False answer occurs, so the checker's negation-as-failure treatment of negated rule premises, which a reviewer identified, produces no observed test error (17 errors on the 182 items whose theory has a negated rule condition, 43 on the other 418). (iii) The validation-only reference reasoner (`pw_validate.py`; written before the freeze with knowledge of both the original and the rewritten rule forms, never shown to the writer) answers 600/600 items correctly under both orig and rules. This depth-5 open-world sample, whose theories ProofWriter renders from formal theories (Tafjord et al., 2021; Clark et al., 2020), is therefore fully invertible by a program that knows its surface forms, and the synthesized checker's limits are its own: the stem collapse is statement extraction (i), and its errors are derivations it cannot complete, which default to Unknown (ii). Its worked examples and feedback come from the same generator's original rule forms, so these controls do not attribute the original 90.00% precision to forms the writer never saw; only the rewritten forms were unseen.

Per-gold-label results (registered descriptive report). Each gold-label group contains 200 items. Coverage is 100% in each group for original and rules conditions and 0% under each stem. The precision figures below are conditional on the *gold label*, not on the label the checker predicts. In particular, 100% on Unknown does not mean every predicted Unknown is correct.

Condition	Gold True	Gold False	Gold Unknown
Original precision (%)	83.00	87.00	100.00
Rules precision (%)	19.50	19.50	100.00
Stem 1 / stem 2 precision	—	—	—

H2: the checker's contribution is template-bound — held 6/6. H2 requires the lower bound of $M(\text{orig}) - M(c)$ to be at least 10 points in every size/perturbation cell. All six lower bounds are at least 17.00. Table 21 reports each contribution and its drop. Under stems the system is the policy because the checker abstains; under rules its 100% coverage makes the system the checker, which performs worse than either policy alone.

Table 21: **PW-AUDIT: every registered contribution contrast (H2).** System and policy scores are percentages; M and its drop are points. Intervals are paired 95%; each perturbation satisfies H2.

Size	Condition	System	Policy	M, 95% CI	Drop of M, 95% CI
14B	Original	90.00	68.83	+21.17 [+17.00, +26.00]	—
	Stem 1	67.83	67.83	0.00 [0.00, 0.00]	+21.17 [+17.00, +26.00]
	Stem 2	68.17	68.17	0.00 [0.00, 0.00]	+21.17 [+17.00, +26.00]
	Rules	46.33	72.17	−25.83 [−30.50, −21.17]	+47.00 [+41.50, +52.83]
7B	Original	90.00	62.33	+27.67 [+23.33, +32.50]	—
	Stem 1	61.50	61.50	0.00 [0.00, 0.00]	+27.67 [+23.33, +32.50]
	Stem 2	63.67	63.67	0.00 [0.00, 0.00]	+27.67 [+23.33, +32.50]
	Rules	46.33	67.50	−21.17 [−26.83, −15.67]	+48.83 [+42.83, +55.00]

H3: policies are not template-bound — held 3/6. The registered rule requires the entire policy-change interval to lie within $[-5, +5]$ points. It held for both 14B stems and 7B stem 1. The three failures have positive estimates: two upper intervals exceed +5 without establishing a gain, and

the 7B rule condition establishes a gain. None is a loss. This is a qualified stability result, not confirmation of all six predictions.

Size	Perturbation	Change vs. original (pp)	95% CI	H3
14B	Stem 1	-1.00	$[-4.33, +2.50]$	held
	Stem 2	-0.67	$[-3.67, +2.33]$	held
	Rules	$+3.33$	$[-0.17, +7.00]$	not held
7B	Stem 1	-0.83	$[-4.67, +3.00]$	held
	Stem 2	$+1.33$	$[-2.33, +5.17]$	not held
	Rules	$+5.17$	$[+1.00, +9.33]$	not held

H4: no partial-input shortcut — held $2/2$. The registered rule requires the upper bound of statement-only policy accuracy to be at most 45% (chance 33.3%). The 14B scores 30.17% $[26.50, 33.67]$, a change of -38.67 $[-43.33, -34.00]$ from the full input. The 7B scores 29.50% $[25.83, 33.00]$, a change of -32.83 $[-38.17, -27.33]$. Both predictions held. This tests these policies under this input removal, not the absence of every possible partial-input rule: a failed partial-input baseline does not show that a dataset is free of artifacts (Feng et al., 2019).

Theory-blind pass (descriptive; no registered decision rule). The analysis prints this under "H5," but the registration specifies no hypothesis test. With a foreign theory of the same family, the primary still covers 100% and scores 37.83% against the original labels. It predicts True/False/Unknown on $31/32/537$ items, respectively, and matches its original answer on $271/600$ items committed in both passes. Run 1 similarly gives $28/29/543$ predictions and matches $310/600$; run 0 gives $17/27/0$ predictions and matches $9/9$ jointly committed items. Run 2 is the primary. A faithful open-world checker would also commit on a foreign theory, returning Unknown when that theory settles neither side of the statement. Thus blind coverage alone is not a shortcut measure, and scores against the original labels do not measure correctness for the substituted theory. The prediction split and original answer agreement must accompany the coverage.

Per-run picks (descriptive; no branch reading). The registered selection rule independently picks one program per run; the table lists all three without replacing the primary. Under both stems all three have zero coverage and undefined precision. Run 0's coverage drops are $19.00/19.00/9.50$ points for stem 1/stem 2/rules; run 1 and run 2 give $100.00/100.00/0.00$.

Pick (run, round)	Selection V	Test original	Test rules	Theory-blind
$(0, 4)$	$21.33/100.00$	$19.00/100.00$	$9.50/100.00$	$7.33/72.73$
$(1, 2)$	$99.83/79.80$	$100.00/81.83$	$100.00/46.33$	$100.00/37.17$
$(2, 3)$, primary	$99.83/89.48$	$100.00/90.00$	$100.00/46.33$	$100.00/37.83$

Entries are coverage / precision (%). Test commit/error counts are $114/0$, $57/0$, $44/12$ for run 0 (original/rules/blind), $600/109$, $600/322$, $600/377$ for run 1, and $600/60$, $600/322$, $600/373$ for the primary.

Truncation and artifacts. Policy truncation percentages for original/stem 1/stem 2/rules/statement-only are $0.2/0.0/0.0/0.2/0.0$ at 14B and $0.3/0.7/0.2/0.7/0.0$ at 7B. Strict extraction was frozen before generation. The registration, `input_hashes.json`, `pw_data.py`, `pw_validate.py`, `pw_run.py`, `pw_policy.py`, frozen programs and `frozen/selected.json` are under `bench/pwaudit/`. Per-item checker outputs are in `pw_test.json`; policy answers and raw completions are in `results/policy_14b.json` and `results/policy_7b.json`. To reproduce the committed analysis from the repository root, set `PW_POLICY_DIR=bench/pwaudit/results` and run `python3 bench/pwaudit/pw_analysis.py`; expected output is `bench/pwaudit/pw_results.txt`. Training-data validation requires the local training shards identified by `pw_data.py`; it was performed before freezing, independently of test scoring.

Precision-constrained selection (post-hoc). The registered selection maximizes $J = \text{correct} - 1.27 \times \text{wrong}$ on V without a precision constraint; the selected program already had 89.48% V

precision. Among the 18 development rounds only one has V precision $\geq 99\%$ (run 0, round 4; 21.3% V coverage at 100%), which is run 0's registered per-run pick, so its test outcomes exist: 19.0% coverage with no wrong commit on original items, 0% under both stems, and 9.5% with no wrong commit under rules. Its contribution over the untuned policies is $+2.50$ $[+1.33, +4.00]$ (14B) and $+3.17$ $[+1.83, +4.67]$ (7B) originally, 0 under the stems, and $+0.50$ $[+0.00, +1.17]$/$+0.83$ $[+0.17, +1.67]$ under rules, so the rule-condition contribution is separated from zero at 7B only. In the descriptive theory-blind pass, 12 of its 44 commits disagree with the original labels, which do not score the substituted theory (per-run picks, above); the claim of no wrong commit covers the original, stem and rule conditions only. A precision-first rule thus turns the silent errors into abstentions on these conditions at the cost of most of the coverage. Source: `bench/pwaudit/pw_precision_selection.py` and its `.txt`.

Abstain-on-Unknown and theory-clustered intervals (post-hoc). A reviewer proposed a simpler control than reselection: keep the registered primary (run 2, round 3) but treat every Unknown answer as an abstention. The rule was proposed after the test pass, from the diagnosis that every test error was an Unknown answer, so its zero test errors hold by construction; the development split, which it was not derived from, was its only out-of-sample check until the pre-registered fresh-theory test below (PW-FRESH). On the development selection split V (never seen by the writer) this rule covers 56.00% at 100.00% precision (336 commits, 0 wrong), so it could have been chosen before test. On test it covers 56.67% of original items and 13.00% under rule rewrites (0% under stems), with no wrong commit; its contribution is $+22.17$ $[+19.00, +25.50]$ (14B) and $+21.83$ $[+18.67, +25.17]$ (7B) originally and $+2.00$ $[+1.00, +3.17]$ at both sizes under rules. Because the 600 items share 345 theories, the registered PW-AUDIT contrasts were also re-estimated with a bootstrap over theories: every contribution drop (H2) and every policy change (H3) keeps its verdict (e.g. 14B rules $+47.00$ $[+41.24, +52.68]$; 7B stem 2 policy change $+1.33$ $[-2.36, +5.03]$, still reaching past the band), and statement-only accuracy stays below 34% (H4). On the theory-blind pass the rule commits on 63 items, 18 of which disagree with the original labels (descriptive; open-world caveat above). Source: `bench/pwaudit/pw_unknown_abstain.py` and its `.txt`.

PW-FRESH: the frozen rule on fresh theories (pre-registered). Because the abstain-on-Unknown rule was proposed after the test errors were seen, we then froze it with the registered primary (`bench/pwfresh/PREREGISTRATION_pwfresh.md`) and scored it once, checker only, on 900 items from the tasksource *test* split of ProofWriter depth-5, which no earlier experiment used (300 per label, one item per theory, `random.Random(2028)`; no theory text shared with the sample or the training split), under PW-AUDIT's unchanged conditions, sandbox and commit rule. Before freezing we checked only input construction: the rule rewrites change 900/900 theories (6,023 of 7,271 rule sentences) and the validation reasoner returns the gold label on 900/900 items under both original and rewritten rules. All four registered predictions held (brackets: item-bootstrap 95% intervals; we write PW-FRESH's labels H-PF1–H-PF3, called H-F1–H-F3 in its registration file, to keep them apart from ROBUST-F's). H-PF1 (precision $\geq$99% on original items, point estimate): the rule covers 56.33% $[53.00, 59.78]$ with 507/507 correct commits (one-sided 95% Clopper–Pearson precision lower bound 99.41%). H-PF2 (the same under rule rewrites): it covers 16.67% $[14.22, 19.11]$ with 150/150 correct (lower bound 98.02%, below 99%, so the target holds by point estimate only). H-PF3 (coverage within $[46.67, 66.67]$ originally and at most 25% under rules): both hold. Per family (registered report), coverage is 48.9–62.4% originally and 12.4–20.8% under rules, with no wrong commit in any family, including the two negation families. As on the sample, the registered primary without abstention covers 100% at 89.67% precision originally and 50.00% under rules, every wrong commit is an Unknown answer on a True or False item, and the checker abstains on every stem-reworded item. Zero observed errors is not a guarantee: the program resolves a negated rule premise by negation as failure, so a constructed item such as "Bob is red. If Bob is not blue then Bob is green." with the query "Bob is green." receives True where the open-world answer is Unknown; none of the 657 commits here was wrong, in this way or any other. Sources: `bench/pwfresh/pwf_run.py`, `pwf_analysis.py`, `pwf_results.txt`, `pwf_test.json`.

Z.1 LOGICLM: A PUBLISHED CHECKER-PLUS-FALLBACK DESIGN, LOGIC-LM, ON PROOFWRITER (PRE-REGISTERED)

Registered at `bench/logiclm/PREREGISTRATION_logiclm.md` (`968476d`) before any translation by the Qwen translators, after an internal LLM review panel called the audit narrow because every audited checker was ours. Logic-LM (Pan et al., 2023) translates each problem into a logic program with a few-shot prompt, runs a solver, and falls back to its language model when the program fails to parse or execute; it is a per-item formalization, not one program written against the templates. We use Logic-LM's ProofWriter prompt and its Pyke solver unchanged, without self-refinement (vendored from `github.com/teacherpeterpan/Logic-LLM` at `d29f1e5`, MIT licence), so the checker commits exactly where Logic-LM does not call its backup, and the fallback is the same-size untuned Qwen policy of PW-AUDIT on the same item and condition. The items and conditions are PW-AUDIT's (the 600 items are Logic-LM's own ProofWriter sample; ids and labels checked equal). Translators: untuned Qwen2.5-14B-Instruct (primary) and 7B (secondary), greedy, Logic-LM's stop word, 2,048 tokens (Logic-LM's default is 1,024; 0–0.2% of 14B outputs were truncated). Predictions (14B, point estimates): H-L1, coverage under each of stem1, stem2 and rules within ten points of original (3 cells); H-L2, the drop of M below ten points (3); H-L3, at least 80% of wrong commits are Unknown answers on True or False items, in orig and rules (2); H-L4, abstaining on Unknown gives $\geq 99\%$ precision in orig and rules (2).

Seen before registration (descriptive). Logic-LM's released GPT-4 programs for these 600 items, run with the same solver, commit on 99.00% at 79.63% precision; all 121 wrong commits are Unknown answers on True or False items. With the PW-AUDIT fallback, M is $+10.83$ $[+7.17, +14.67]$ (14B) and $+17.17$ $[+12.83, +21.50]$ (7B); abstaining on Unknown leaves 45.67% coverage at 100% precision and M $+10.50$ $[+8.17, +13.00]$ and $+13.67$ $[+11.00, +16.33]$. No GPT-4 translation of the rewritten items exists.

Registered results. With the 14B translator, Logic-LM commits on only $17.00/6.00/20.83/13.00\%$ of items (orig/stem1/stem2/rules) at $82.35/91.67/87.20/85.90\%$ precision; M is $+1.00$ $[-0.67, +2.67]$, $+0.67$, $+2.17$ and $+0.50$. H-L1 held $2/3$ (stem1 -11.00 points); H-L2 held $3/3$ (drops $+0.33$, -1.17, $+0.50$); H-L3 held $2/2$ ($16/18$ and $11/11$ wrong commits are Unknown); H-L4 held $1/2$ (96.30% on orig with 2 wrong commits, 100% under rules). These predictions hold largely because Logic-LM rarely commits: 345 of the 600 original outputs fail to parse and 153 fail to execute. The 7B secondary commits on at most 0.67% of items in any condition: its outputs do not follow the program format (almost none contain the section headers), so H-L1 and H-L2 hold trivially ($3/3$ each), H-L3 has no wrong commit to evaluate, and H-L4 holds on orig (one commit) and cannot be evaluated under rules (no commit; the script prints NOT HELD).

Post-hoc: the parser is bound to the output format. The 14B outputs fail mainly because the chat model writes the program as Markdown (a preamble sentence, `###` headers, – list markers, bold), which Logic-LM's parser splits wrongly or Pyke cannot compile ("illegal character –"). A content-preserving format repair (`bench/logiclm/ll_posthoc_format.py`: strip the Markdown, keep predicates, facts, rules and query verbatim) raises coverage to $68.67/71.17/76.00/56.67\%$ at $75.24/74.00/76.54/75.59\%$ precision; M is $+4.67$ $[+1.33, +7.83]$, $+3.17$, $+6.17$ and $+0.17$ $[-2.83, +3.00]$, so the drop under rule rewrites is $+4.50$ $[+1.00, +8.00]$ and under the stems $+1.50$ and -1.50. 98 of the 102 original and 79 of the 83 rule-condition wrong commits are Unknown on True or False items. Abstaining on Unknown gives 97.71% (orig) and 97.26% (rules) precision at 29.17% and 24.33% coverage, with M $+5.67$ and $+2.50$. The repair does not help the 7B translator (coverage at most 1.83%). So the published design's precision stays at 74–77% under every rewording, unlike SYN's (90% to 46% under the same rule rewrites); it shares SYN's silent default-Unknown errors, and its own input interface is the translator's output format. Sources: `bench/logiclm/` (`ll_results.txt`, `ll_posthoc_format.txt`; raw translations and solver outcomes in `bench/logiclm/results/`).

Z.2 REGISTRATION LEDGER

Every pre-registered experiment of the checker audit (the PAL, SYN, ROBUST and ProofWriter families; body names in the roadmap above), its registration commit (the frozen file is committed before the first output it constrains), its registered predictions and the outcome in the registration's own counting; the earlier registrations retained in other appendices (E1–E7, MATH-500, MMLU-Pro,

logic, MBPP+, strict domination, oracle-v2, matched evidence, symmetric debate, cross-family) are cited with their commits there. Decision rules differ between registrations (point estimates, whole intervals, CI bounds, branch wording); each row's file states its rule. ROBUST-P/P7 used the legacy lenient parser (Appendix Y.4).

Experiment	Commit	Registered predictions	Outcome
PAL	`0100037`	$D = \text{PAL} - \text{v1}$, branch by CI vs ± 3	branch A (-32.75)
SYN	`d648c13`	$D = \text{SYN} - \text{v1}$, branches A–D, qualifiers	branch C ($+9.56$), no qualifier fires
SYN-14B	`45e9ed9`	as SYN, untuned 14B writer	branch A (-30.31)
SYN-NT	`4ee7b9b`	as SYN, no extended thinking	branch C ($+9.75$), no qualifier fires
ROBUST	`1df1e06`	H-R1 (4 cells), H-R2 (3), H-R3 (4)	$2/4$, $3/3$, options rule only
ROBUST-P	`0c29057`	H-P1 (branch), H-P2 (4), H-P3 (2)	branch C, $4/4$, $2/2$
ROBUST-P7	`ac7b476`	as ROBUST-P at 7B	branch C, $4/4$, $2/2$
ROBUST-P2	`fddbaf1`	H-Q1 (4), H-Q2 (8), H-Q3 (2), H-Q4 (2 branches)	$4/4$, $8/8$, $2/2$, C/C
ROBUST-P3	`4cb9928`	H-T1 (6), H-T2 (4), H-T3 (4)	$0/6$, $4/4$, $4/4$
ROBUST-P4	`01c1257`	H-S1 (6), H-S2 (4), H-S3 (4), H-S4 (4); amendment 1	$0/6$, $4/4$, $4/4$, $4/4$
ROBUST-N	`33dd404`	H-N1 (5), H-N2 (5), H-N3 (5); amendment 1; per-template print code added after results (`08c985d`)	$4/5$, $5/5$, $5/5$ (H-N2 follows from H-N3 for deterministic checkers; not an independent confirmation)
ROBUST-F	`70334cb`	H-F1 (2), H-F2 (3), H-F3 (5), H-F4 (1)	$2/2$, $0/3$, $5/5$, $1/1$
PW-AUDIT	`d46a8f1`	H0 (1), H1 (3), H2 (6), H3 (6), H4 (2)	not held, $2/3$, $6/6$, $3/6$, $2/2$
LOGICLM	`968476d`	H-L1 (3), H-L2 (3), H-L3 (2), H-L4 (2)	$2/3$, $3/3$, $2/2$, $1/2$ (14B translator; at 6–21% coverage); 7B secondary $3/3$, $3/3$, n/a, orig held on one commit and rules not evaluable (Appendix Z.1)
PW-FRESH	`7f89d76`	H-PF1 (1), H-PF2 (1), H-PF3 (2) (H-F1–H-F3 in the file)	$1/1$, $1/1$, $2/2$
SYN-R	`5217b5b`	H-SR1 (1), H-SR2 (5), H-SR3 (5), H-SR4 (5)	$1/1$, $5/5$, $5/5$, $3/5$
ROBUST-NP	`cbdbed6`	H-NP1 (5), H-NP2 (5), H-NP3 (5), H-NP4 (5); amendments 1–4	$5/5$, $5/5$, $5/5$, $5/5$ (primary, untuned 14B; H-NP2 rests on H-N3's identical checker outcomes, so only the policy side varies); untuned 7B secondary $5/5$, $5/5$, $5/5$, $5/5$; R1-distilled 14B secondary $5/5$, $5/5$, $2/5$, $4/5$ (Appendix Y.3)