SAME VERDICT, DIFFERENT CHECK VALUES: WHAT FIELD ORDER CHANGES IN A STRUCTURED LLM JUDGE

Anonymous authors
Paper under double-blind review

ABSTRACT

Structured LLM judges return a verdict together with per-criterion checks, and auditors rely on those checks to diagnose errors. We show that these checks can depend on the order in which the fields are declared in the output schema, a choice that leaves the content of the schema unchanged. The verdict can remain unchanged, and correct, while the checks differ. Holding the instances, rubrics and all other content fields fixed, we move the rating and confidence fields before or after the checks and rerun the judge. Ten pre-specified randomised designs cover eight (model, task) configurations. On claim verification, 28% of instances retain their verdict while changing their checks when the two runs use different field orders, against 5% when both runs use the same order: an excess of 23 pp. Two further judge models give $+15.8$ and $+7.8$ pp on the same task. The effect persists under every remedy and decoding change we tested. Reordering other keys of the schema changes the checks as well. Aggregate accuracy conceals the change: on MCJudgeBench, where every constraint carries a human label, mean constraint accuracy differs by at most 0.34 pp between the two orders. Beneath that, which constraints the judge scores correctly changes, and so does the set of constraints flagged for repair. Used as an evaluator, the same judge ranks a different model first on every cross-order pair of runs, against one pair in five within a single field order. These results show that verdict-level agreement does not certify a structured judge's audit record: error analyses, repair lists and rankings built on the judge can change with the field order alone. We recommend that the field order be reported, and that check-level evidence be read against a replicate in the same field order.

1 INTRODUCTION

Large language models are now routinely used as judges of other models' outputs (Zheng et al., 2023): given an output and a rubric, the judge decides, for example, whether a claim is supported by its evidence. A *structured* judge answers in a fixed output schema and returns more than a verdict: a check for each criterion of the rubric (for claim verification, whether the claim's numbers match the evidence and whether the evidence is sufficient), together with rubric scores and reasoning fields. Increasingly it is these structured outputs, not the verdict alone, that downstream work analyses. Error analyses break judge errors down by criterion (Lee et al., 2026b), interpretability claims read the checks as the judge's reasons, a reading already contested for free text (Devasier, 2026; Tapwal et al., 2026), and aggregation combines the per-check returns into a verdict or score (Singh et al., 2026; Chen et al., 2026). Each of these uses rests on an invariance assumption that is rarely stated: that the checks would be the same if the fields of the schema were arranged differently. Field order is the change least likely to be recorded as a decision: a wording edit is typed by someone, whereas a field order follows from the order in which a class happened to be defined.

Output order is known to move what an LLM judge decides (Chen et al., 2024; van Nuenen & Sachdeva, 2026), and a stable mean can hide migrating per-constraint correctness under input-side perturbations (Lee et al., 2026b). What the checks do beneath an unchanged, correct verdict when only the field order changes is rarely measured directly (Section 6). This paper makes the assumption explicit and tests it on field order, the everyday case of a wider sensitivity to the order of a schema's

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.

RESPONSE FORMAT, AS DECLARED

```
{
  "rating":  "Supported"|"Refuted"|
             "NotEnoughInfo",
  "confidence":  0.50-1.00,
  "checks":  { ...6 checks ...},
  "issues":  [ ...],
  "reasoning":  "..."
}
```

TOP-LEVEL KEY ORDER (boxed = the block that moves)

VERDICT-FIRST
`[rating → confidence]` → `checks` → `issues` → `reasoning`

CHECKS-FIRST
`checks` → `[rating → confidence]` → `issues` → `reasoning`

REASONING-FIRST (control)
`reasoning` → `[rating → confidence]` → `checks` → `issues`

One instance, both field orders (`fev-23737`):

claim In the Battle of Sandomierz the castle was attacked by Conel Cronlood where large stores of body armor were held.
evidence . . . On March 24 the bridgehead, the city and the castle were attacked by Jerzy Sebastian Lubomirski . . .

	verdict-first	checks-first
verdict	Refuted	Refuted
`entities_match_evidence`	true	true
`numbers_match_evidence`	**n/a**	**false**
`relations_match_evidence`	**false**	**true**
`temporal_match_evidence`	n/a	true
`no_contradicting_evidence`	**false**	**true**
`evidence_sufficient`	**true**	**false**

Same correct verdict; 4 of 6 checks change class (bold on a tint = check changed; grey = unchanged)

Figure 1: The intervention, and one instance it changes. **Left**: the response-format skeleton for claim verification (the `checks` object, byte-identical across field orders, is elided), with the three field orders beneath it. **Right**: one instance judged by llama-3.3-70b, with the claim and the one evidence sentence that settles it. Both field orders return the gold verdict (`Refuted`) while four class-mapped checks change in opposing directions and two do not. `n/a` maps to pass, so `numbers` changes class and `temporal` does not.

keys. We change only the order in which the output schema declares its fields, holding the instances — the inputs the judge is asked to judge — and the rubrics fixed, and rerun the judge (Figure 1). A stable, correct verdict does not guarantee that the same errors were found beneath it, and the difference is measurable.

Field order alone can rewrite the error analysis, the repair list and the leaderboard (Section 4). Without a same-order replicate, a reader also cannot tell the order effect from the judge's own run-to-run noise. An LLM judge audited under one fixed, disclosed field order still supports the uses above; what fails is reading the checks as order-independent without testing that they are. A metric computed from verdicts alone cannot see check changes that leave the verdict unchanged.

Two quantities appear throughout, and one is part of the other. The *identity gap* is how much more often two runs return a different set of check values across field orders than within a field order. T_{stable} is the part of that excess in which the verdict stays the same, and the rest is change that takes the verdict with it. Both are read against the judge's own same-order disagreement. Neither says whether a changed check became right or wrong; human per-constraint labels do (Section 4).

Claim and findings. The claim is one of existence and configuration dependence: the size of the effect depends on the model, the task and the prompt together. Three findings:

1. **Same verdict, different check values**. The LLM judge repeats its own per-criterion answers less often when the field order changes than when it stays the same. That difference, the *identity gap*, is clearly above zero in every configuration where it can be measured but one, and it replicates when the field order is randomised. On claim verification much of it comes while the verdict stays the same.
2. **Accuracy stays stable while the audit record changes**. A stable aggregate does not certify the audit record beneath it. On a public benchmark with a human label on every constraint, mean accuracy barely differs between the two field orders while the LLM judge gets a *different set* of checks right and flags a different set for repair.
3. **Reordering other keys also changes the checks, and no remedy we tested removes the effect**. The verdict fields are not special: under an unchanged verdict, reordering other keys of the same schema can change the checks more than moving the verdict block does. No decoding change we tested removes the effect either, and voting over same-order runs sharpens rather than removes it.

2 EXPERIMENTAL SETUP

The intervention. A structured LLM judge returns one JSON object per *instance*, an instance being one input it is asked to judge. We hold the object's *content* fixed (the verdict enum, the named per-check fields with their taught instructions, the free-text fields) and change only where the verdict block sits in the declared schema (Figure 1). A *field order* is one of the three declared top-level key orders we compare: *verdict-first*, *checks-first*, or the *reasoning-first* control, which puts free-text reasoning rather than the checks before the verdict. Verdict-first and checks-first, the two compared throughout, contain the same lines in a different order (their digests: Appendix A); the intervention is the *declared* order in a prompt-serialised schema, and one pre-specified design also enforces it at the decoder (Section 3). The judge runs under both field orders on the same instances, on four judging tasks (Table 1) and one ranking task (Section 4); where the order effect is measured, it also runs at least once more under the same field order. Two runs of one field order are a *same-order* pair; one run of each is a *cross-order* pair. A same-order pair shows how much the judge disagrees with itself; a cross-order pair shows that disagreement plus whatever the field order adds. The order effect reported throughout is the difference between the two.

The block is `rating` + `confidence`, and they move together, so the *estimand*, the quantity a design estimates, is throughout the order effect of that two-field block, not of the verdict field alone. The confidence *value* barely changes across orders (per-configuration figures and the reasoning-first control: Appendix B). Everything else is pinned: rubrics and instance samples digest-frozen before any run, temperature 0 except where a thinking mode or a design varies it, one user turn, one parser, one deterministic checks→verdict rule (Appendix C and the Reproducibility statement).

Tasks. We apply the intervention on four structured-judging tasks. The first three carry the identity gap measurement with different kinds of gold label; the fourth adds per-constraint gold for Section 4. Table 1 gives samples and schema sizes; what matters here is how the label set changes what a score can mean. **SSDV** (standalone scientific data verification; Anonymous, 2026): scientific-data verification, where literature-mined instances are judged against their source span, with a blind second expert pass for reference labels. Its files ship in this paper's artifact. SSDV is a replication task: no randomised design uses it. **DICES-350** (Aroyo et al., 2023): safety rating, where the expert gold has no abstain class, so an abstention can never score as a match. **FEVEROUS-999** (Aly et al., 2021): claim verification, where not-enough-info is a gold answer class, not a refusal, so a move into or out of it can score as right or wrong. The claim-verification sample over-samples not-enough-info, so the figures of the fixed-order matrix, the exploratory runs made in one fixed sequence rather than randomised, are reported both label-balanced and reweighted to the eligible pool's label mix. Table 2's randomised endpoints are label-balanced by construction and not reweighted (Appendix D).

Judges, and the six core configurations. A *configuration* is one task × one judge model. Six open-weight families appear, and they play different roles. Three carry the identity gap measurement on both claim verification and safety rating: llama-3.3-70b, gemma-4-26b-a4b-it and qwen3-next-80b-a3b-instruct (endpoint model IDs, one platform). Those three families × two tasks are the **six core configurations** the paper counts by that name throughout. kimi-k2-thinking runs on both tasks and is reported beside them (Table 7). Of the two proprietary API models, gpt-5.4 is the replication judge of Table 2 and of Section 4. The other judges, among them a matched qwen3-next thinking/instruct pair, appear mainly in the fixed-order matrix (the roster: Appendix D).

Table 1: The four tasks. *boundary*: the verdict-stable part attenuates there, and most check changes move the verdict too (Section 3). MCJudgeBench's design and scoring rule: Section 4.

Task	instances	checks	gold	role
FEVEROUS-999	999 (333/label)	6	verdict (not-enough-info is gold)	primary
DICES-350	350	6	verdict, no abstain class	boundary
SSDV	1,223	11	verdict, human-located field errors	third-task replication
MCJudgeBench	219 (1,022 constr.)	2–15/instance	per-constraint human labels	audit consequences

Designs, scope and comparators. A *design* is one pre-specified randomised experiment. Its *conditions* are the prompts it compares, in most designs the two field orders. A *design point* is one configuration run in one design, counted once per randomised comparison (the two-loci design has two). Every quantity here is a difference between conditions inside one batch of runs made together; change between batches we call *batch-to-batch change* and keep separate. Three kinds of evidence are kept apart. The first is the exploratory fixed-order matrix, including the six core configurations. The second is ten pre-specified designs, nine block-randomised ten-block designs on fifteen design points and one earlier five-block design, covering eight model $\times$ task configurations. The third is the same-batch controls. One of the ten designs is on a ranking task (WildBench, Section 4), three of the design points are Table 2's, and Appendix E inventories them and says which carry an enumerable assignment. In the fixed-order matrix, a rate difference between field orders is one run against one run, and is reported beside a single observed same-order difference, a descriptive comparator.

The nine ten-block designs share one assignment scheme: ten blocks per configuration, both conditions back to back over one seeded instance order, the within-block order drawn uniformly from the $\binom{10}{5} = 252$ allocations with five AB and five BA. Each compares the mean over same-order pairs with the mean over cross-order pairs, and the analysis walks all 252 allocations, so the *minimum attainable* two-sided p is $2/252 = 0.0079$ (Appendix F). Their intervals are an instance-level bootstrap (B=2,000; task clusters for the ranking), conditional on the observed runs; the one multiplicity correction is instance-level.

Compliance check. We extract emitted key order from the raw response text *before* parsing, since a parsed JSON object is order-blind. Every answered call of the exact design (Table 2) followed the requested key order (counts: Appendix C), and both gold-scored MCJudgeBench designs comply at 100% in all forty runs each. Two cases do not comply. Under the decoder-enforced design qwen3-next emits keys alphabetically (Section 3). In the design comparing input and output reversals, the reversed fifteen-key schema is emitted in canonical order on 96.07% of rows (Section 5). Declared and emitted order are therefore kept apart.

Metrics. We take the same instance, judge it twice under the same frozen rubric in two single calls, and compare the two check vectors. We map each returned value to one of three classes, **pass**, **fail** or **soft** (`partial`, `uncertain`, `unknown`, `unclear`), and drop the instance from that run if a value maps to none of them. `not-applicable` maps to pass, the conservative choice: it counts a check as unchanged when one run declines to apply it and the other passes it. Such values are 10.1% of those returned in Table 2's exact design, and giving them a fourth class of their own raises every claim-verification estimate by about a point and leaves every exact p at its floor (Appendix E). Appendix B gives every estimand's formula, unit, pairs and table. Four quantities are then kept distinct.

(a) $\boldsymbol{T}_{\text{stable}}$, the pre-specified primary endpoint of the exact randomised design (Section 3, Table 2):

$$T_{\text{stable}} = \Pr(V_{\text{same}} \wedge X_{\text{changed}})_{\text{cross-order}} - \Pr(V_{\text{same}} \wedge X_{\text{changed}})_{\text{same-order}},$$

where V denotes the verdict and X the class-mapped check vector. Both terms are *unconditional joint* probabilities over all instances both runs answered (not the conditional probability of a check change given an unchanged verdict; Appendix B).

(b) **Check-vector identity** and its *identity gap*, the secondary endpoint. Vector identity is the share of instances whose *entire* class-mapped vector is identical between two runs, so *within-order* identity (two runs of the same field order) measures reproducibility. **Check-vector identity gap** is mean within-order identity minus mean cross-order identity (within over both field orders, cross over every verdict-first $\times$ checks-first pair; Appendices B, D): equally the excess of cross-order vector change, which splits exactly into T_{stable} and changes that move the verdict too. Vector identity is strict and not chance-corrected, so it is read only beside the same-order comparison for the same configuration; **per-check agreement** accompanies it.

(c) **Disagreement with a selected aggregation policy**: the published verdict contradicts the deterministic checks→verdict rule applied to the model's own emitted check values. The measure is deterministic per instance; the *rate* carries sampling uncertainty, and Section 5 reports it. The exploratory fixed-order matrix (three tasks, Appendix D) measures (c) and (d), and (b) where it has replicate runs, as on the six core configurations.

(d) **Gold accuracy** (Section 4), secondary by design: where the gold has no abstain class it conflates judgment change with abstention re-scoring.

3 SAME VERDICT, DIFFERENT CHECK VALUES

Claim verification on FEVEROUS, judged by llama-3.3-70b, is the primary configuration for T_{stable}; we fixed the other two rows of Table 2 as replication (gpt-5.4, same task) and boundary (safety rating, where the fixed-order runs showed attenuation) before any call. In Figure 1's instance (its run pair: the supplementary material, part G) both field orders return the gold verdict and four of six checks change class. On the primary configuration, 28.28% of instances keep their verdict and change their checks across field orders (Table 2).

The effect exceeds the judge's own run-to-run variation. A judge that disagrees with itself would change its checks under an unchanged verdict with no help from the field order, so each figure is read against two runs in the same field order. There the share is 5.28%, so $T_{\text{stable}} = +23.00$ pp at the design's exact floor, $p = 0.0079$: the observed assignment is the most extreme of its 252 allocations (Table 2). The replicate matters. On gpt-5.4 the same-order disagreement is more than half of the raw cross-order figure (9.31% against 17.15%). In the fixed-order matrix, kimi-k2-thinking changes its check vector across field orders on half its claim-verification instances. Read against zero, that would be the matrix's largest order effect. But it changes as often across its own reruns: its identity gap is $+0.33$ $[-1.40, +2.06]$ pp (Table 7, Appendix G), the one exception.

The effect is not confined to one judge model. The pre-specified gpt-5.4 replication gives $+7.83$ pp. llama-3.3-70b and gpt-5.4 were chosen because they are the only claim-verification judge models with five pre-replication runs in each field order. Run unchanged on an unselected judge model, the design gives $+15.78$ pp at the floor (gemma-4, Appendix E).

A correct verdict does not certify its checks. Over half of each claim-verification effect in Table 2 sits under verdicts that are also right (12.90 of the primary configuration's 23.00 pp, 6.08 of gpt-5.4's 7.83; Appendix H), and the fixed-order runs split the same way (Appendix G).

Randomised assignment rules out time. Other explanations could produce the same excess without the field order doing the work. The first is time: the exploratory fixed-order runs executed verdict-first then checks-first (Appendix E), so field order was collinear with position in time, and any drift between the two would read as an order effect. The design behind Table 2 therefore randomises the field order inside ten blocks of runs, and the excess above is its result (Appendix E).

On the core configurations, neither field order is noisier. A second explanation is a noisier field order: were checks-first simply noisier, its within-order identity would fall toward the cross-order level. The fixed-order matrix tests this on its six core configurations (Figure 2A, Table 7). It does not fall. It stays within 2.2 pp of the verdict-first within-order identity in all six, and all four cross-order pairs (each verdict-first run against each checks-first run) agree to within 1.2 pp (Figure 2A's

Table 2: The exact randomised design. *same* and *cross*: the share of instances that keep their verdict and change their checks, within and across field orders, over the n instances all twenty runs share; T_{stable} is cross minus same and *gap* the identity gap (Section 2). Brackets: 95% instance-bootstrap intervals conditional on the observed runs. Every configuration reaches the exact test's floor, $p = 2/252 = 0.0079$, so p is not a column. The floor is not an effect size: the bracket resamples instances while p re-allocates field order, so the safety row reaches the floor with a T_{stable} interval covering zero (Appendix F). *all designs*: the range over Table 6's estimates of this intervention on the configuration, which carries the run, batch and design variation the bracket does not.

Task	Judge model	role	n	same (%)	cross (%)	T_{stable} (pp)	all designs	gap (pp)
Claims	llama-3.3-70b	**primary**	973	5.28	28.28	$+\mathbf{23.00}$ $[+20.6, +25.3]$	+20.9–+26.1	+33.13
Claims	gpt-5.4	replication	997	9.31	17.15	$+\mathbf{7.83}$ $[+6.1, +9.6]$	+3.4–+9.9	+21.96
Safety	llama-3.3-70b	boundary	341	0.60	1.11	$+\mathbf{0.51}$ $[-0.4, +1.7]$	+0.5–+0.7	+19.24

open dots). This holds even though the checks-first replicates ran hours apart and the verdict-first ones minutes apart (Appendix D). There the two field orders are about equally reproducible and give different answers; on gpt-5.4, by contrast, checks-first is the less stable order (Appendix G).

Rubric ambiguity does not explain the effect. A third explanation is one ambiguous clause: `evidence_sufficient` carries 42% of the check-changes on the primary configuration's verdict-stable instances (Appendix I), and its clause admits two readings. Restating it under either one leaves most of the movement in place: on the primary configuration's twelve same-day runs, T_{stable} is $+23.01$ under the original rubric against $+15.82$ and $+14.25$ under the two restatements (Appendix J). Both sides of the task boundary survive a complete semantic-preserving rewording of each rubric: the verdict-stable term stays large on claim verification and at most $+2.21$ pp on safety rating (Appendix K). Nor does the rubric's unstated precedence between its hard-fail and not-enough-info clauses explain it. A 2×2 of field order $\times$ taught precedence teaches the precedence explicitly either way (Appendix L), and inside each explicit-precedence condition, identity gap and T_{stable} stay intact (Table 10, Appendix L).

Declared order alone moves the checks. A last, deflationary reading is that the effect is about where the model writes a field, not where the prompt declares it. One design separates the two. On qwen3-next a decoder-enforced schema writes the keys alphabetically in both conditions, so the written order is identical and only the declared order differs, and the checks still move: $T_{\text{stable}} = +9.47$ pp (13.20% cross-order against 3.73% within). Writing position is therefore not the whole mechanism (Appendix M.1). Under the ordinary, unenforced regime the two co-vary, so the larger primary-configuration effect may carry both. Post-hoc rationalisation of a settled verdict predicts that verdict-first checks fit their own verdict better. That holds on the claim-verification runs under the selected policy and reverses on MCJudgeBench, so rationalisation cannot account for both (Appendix L).

The effect's size depends on the task and the judge model. The fixed-order runs already show a task boundary: identity gap is higher on claim verification than on safety rating for each of the three core judge models, under either label weighting (Figure 2A, Appendix N). The randomised design reproduces that ordering, with an identity gap of 19.24 pp on safety rating against 33.13 on the primary configuration. On safety rating almost none of it sits under an unchanged verdict: T_{stable} is $+0.51$ pp, its interval covering zero, against $+23.00$ (Table 2). The two quantities part because on safety rating 20.8% of instances change their vector across field orders, nearly all with the verdict, and every such verdict change is into or out of abstention (Appendix E). Existence replicates more reliably than magnitude: on gpt-5.4 the direction replicates at magnitudes that vary with the batch, its seven claim-verification estimates of this intervention spanning more than its printed interval (Table 6, Appendix E). On scientific-data verification, the third task, a same-batch control gives identity gaps of $+17.91$ and $+9.81$ pp on llama-3.3-70b and gpt-5.4, and the field order moves the checks tied to human-located field errors (Appendix O).

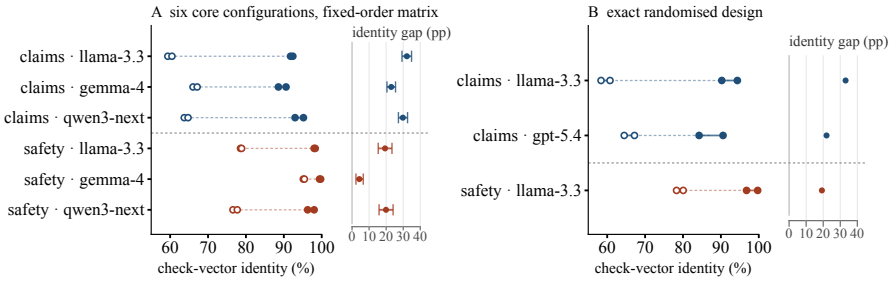


Figure 2: Check-vector identity is higher within a field order than across field orders. Filled dots: the lowest and highest *same*-order comparison; open dots: the lowest and highest *cross*-order comparison; at the right, the *identity gap* between the two means, with Table 7's bootstrap interval. **A**: the six core configurations of the fixed-order matrix, label-balanced (Section 2). **B**: the three configurations of the exact randomised design, whose inference is the exact test (Table 2), so their gaps carry no bar. The four other fixed-order configurations are in Table 7.

On claim verification, then, an unchanged verdict carries different check values more often across field orders than within one: under randomised assignment on every judge model run that way, descriptively on the fixed-order matrix's three core judge models, and with the written order fixed.

4 CONSEQUENCES FOR AUDITING AND RANKING

A benchmark with per-constraint human gold. Section 3 shows that the checks change. Whether a changed check became right or wrong needs per-check gold, and the three original tasks lack complete per-check gold (SSDV carries human-located field errors, the other two verdict labels only). MCJudgeBench's public release (Lee et al., 2026b) supplies that: 219 instruction-following instances with 1,022 human-labelled constraints, one fixed candidate response each. One (instance, constraint) pair is one unit, a unit *migrates* when its gold-scored correctness differs between two runs, and the 1,022 units are the denominator of every migration rate in this section. We reran the field order contrast on it under the design of Section 3, on llama-3.3-70b (primary) and gpt-5.4 (pre-specified replication), twice: with a fixed fifteen-slot schema, and with a no-filler control that lists only each instance's own constraints (design, limits and gold-verdict rule: Appendix P).

One instance, twenty runs. A correct verdict does not certify which checks are right. `mcjb_0162` has three constraints, and gold marks all three unsatisfied: ungrounded analysis, a misattributed tactic, wrong length. All twenty runs return the gold verdict, yet the verdict-first runs find two of its three real errors and the checks-first runs find one. Each field order is unanimous:

`mcjb_0162`	gold	verdict-first	checks-first
1. analysis and response based on the case material	`no`	**`yes` 10/10**	**`yes` 10/10**
2. analysis matches the material and the facts	`no`	`no` 10/10	**`yes` 10/10**
3. analysis is 400–500 characters	`no`	`no` 10/10	`no` 10/10
verdict	`FAIL`	`FAIL` 10/10	`FAIL` 10/10

bold = the run disagrees with gold

Mean accuracy changes little; *which checks are right* migrates. Mean constraint accuracy differs by at most 0.15 pp across field orders in the fixed-slot design (0.34 pp in the no-filler control). Underneath that, disagreement on a constraint's gold-scored correctness is higher across field orders than within one. Lee et al. (2026b) showed migration under input-side perturbations; here the verdict agrees and the comparator is a same-order rerun (Appendix Q). All four pre-specified endpoints are higher across field orders than within one, in every configuration of both designs (three in Table 3; all four in Tables 12 and 13). They are correctness migration unconditionally, under an agreeing verdict (the primary endpoint) and under a verdict that agrees and is gold-correct, and repair-list divergence. On llama-3.3-70b's fixed-slot design point the primary endpoint is $+1.10$ pp, on a different denominator and unit from Table 2's per-instance T_{stable}. It is concentrated: the three largest instances carry 85.5% of it and the largest alone 44.4%. Without the three largest it is $+0.16$ $[-0.12, +0.52]$, an instance interval covering zero. After the same removal the other three

Table 3: Gold-scored audit endpoints on MCJudgeBench, both designs. *same* and *cross*: the primary endpoint within and across field orders, in % of units (a unit is one instance–constraint pair); each Δ is cross minus same in pp, at the exact floor $p = 0.0079$ (rank 1/252) in every entry, with the instance-bootstrap 95% CI on the primary Δ only; *verdict-correct* is the primary endpoint further requiring the agreed verdict to be gold-correct. The migration columns are unit-weighted and scored against human gold; repair-list divergence is instance-averaged and uses no gold labels, so the columns are not on one scale. The two designs ran in different batches, so comparing the blocks is descriptive. All four endpoints with intervals, and the design: Tables 12 and 13, Appendix P.

		verdict-stable correctness migration (**primary**)			verdict-correct	repair-list
Design	Judge model	same	cross	Δ [95% CI]	joint Δ	divergence Δ
fixed 15-slot	llama-3.3-70b	0.41	1.51	$+1.10$ $[+0.14, +2.45]$	$+1.23$	$+2.48$
	gpt-5.4	2.35	3.13	$+0.78$ $[+0.28, +1.33]$	$+0.76$	$+2.03$
no-filler	llama-3.3-70b	0.27	1.15	$+0.88$ $[+0.32, +1.58]$	$+0.75$	$+2.04$
	gpt-5.4	1.95	3.28	$+1.34$ $[+0.69, +1.99]$	$+1.01$	$+1.83$

design points keep intervals above zero. An audit is worked instance by instance, and an aggregate that holds while the flagged set beneath it changes is the failure an auditor meets.

The migration survives without filler slots. About 69% of the fixed schema's fifteen slots are `not_applicable` filler, which could itself produce the migration. Excess correctness migration persists when the filler slots are removed on both judge models (Table 3, lower block; Table 13).

The repair list changes with the field order. The same responses yield a different set of constraints flagged for repair. On instances where both runs return the same, correct verdict, each field order flags real errors that the other misses, counted against gold (unlike the gold-free divergence). On llama-3.3-70b that is 5.4 per run pair in each direction, against 0.8 between two verdict-first runs and 1.6 between two checks-first runs. On gpt-5.4 the two field orders give 6.5 and 3.5 against 3.6 and 2.6 within, so its excess is the smaller of the two, and the precision and recall of its repair lists trade against each other. Those counts sit on a pool of 632–702 of the benchmark's 1,022 constraints. Flagging `partial` too leaves the divergence in place (Appendix P).

The error analysis moves as well. On the primary claim-verification configuration of Section 3, the criterion an error analysis reports as failing most is `relations_match_evidence` in all ten verdict-first runs and `evidence_sufficient` in all ten checks-first runs. The reading is post hoc. Under checks-first, `evidence_sufficient` leads by 0.98 pp, the flip holds in 72.2% of bootstrap draws, and gpt-5.4's profile does not flip (Appendix I).

The ranking such a judge produces moves with the field order. In one design llama-3.3-70b scores twelve models' released WildBench outputs against each task's checklist (Appendix R). The *top-ranked* model differs on 100% of cross-order run pairs against 20% of same-order ones, and each field order's leader tops nine of its ten runs, so this is not run-to-run jitter. The design's primary endpoint, rank correlation, moves less: Kendall's τ_b is 0.93 within a field order against 0.79 across, a difference whose task-cluster interval excludes zero but reaches down to +0.01. One judge and one task set; the leader is stable over runs but not under task resampling (Appendix R).

5 Discussion and Limitations

Field order is the everyday case of a wider sensitivity to key order. Whether the effect belongs to field order or to any prompt edit is tested by a same-batch perturbation control on claim verification with llama-3.3-70b, qwen3-next and gpt-5.4 (Table 14, Appendix S). Each condition changes one thing: the indentation only, or one of three reorderings of declared keys (the six check keys reversed, the issues block moved, the verdict block moved). The three reorderings exceed the indentation-only control's identity gap on all three judge models, by +8.6 to +31.8 pp (indentation alone moves the checks a little). On T_{stable}, on each judge model, at least one other reordering *exceeds* the verdict-block move, its interval excluding zero. The check-key reversal carries a block-randomised, independently batched replication on the primary configuration (T_{stable} +32.40 pp at the design's floor). On llama-3.3-70b and qwen3-next, edits that reorder nothing move the verdict-stable checks less: the verdict-block move's T_{stable} is +22.99 and +20.39 against the indentation-only edit's +1.15 and +4.63 (Table 14). It also exceeds a character-matched rewording of the instructions by +18.79 and +12.43 pp in a second same-batch control (Appendix M.3). But on gpt-5.4 it is no larger, +3.59 against +4.19, though it moves the whole vector three times as much (Table 14). There it is also below the rewording's, the difference −5.05 pp [−7.66, −2.60] (Appendix M.3). The axis also extends beyond the output schema: in one design (MCJudgeBench × llama-3.3-70b, Appendix T), reversing the input's constraint list moves the verdict-stable migration by +3.53 pp, against +1.81 for reversing the fifteen declared keys, most of them filler. The sensitivity is thus to order, in the declared schema and in the input. Field order is the order that decides whether a judge writes its verdict before or after its checks, the question prior work has asked of verdicts (Section 6); this paper examines it in depth, and the other orders rest mainly on same-batch contrasts.

Verdict-level behaviour, measure (c), moves too, in opposite directions across tasks. Moving the checks ahead of the verdict lowers how often llama-3.3's verdict contradicts its own checks on safety rating (6.57% → 1.14%) and cuts its abstentions (30.29% → 4.29%). The abstention

change is coverage, not judgement: on the 244 instances where neither field order abstains, every verdict is identical (Appendices H and U). On claim verification the same measure rises for llama-3.3 ($5.71\% \rightarrow 23.95\%$), mostly where the rubric allows two readings and the verdict follows the unselected one (Table 9; Appendices L, N and U).

The standard remedies leave it in place. No standard remedy or decoding change we tested makes the effect go away (Table 15, Appendices M and V). Voting five same-order runs per check *sharpens* it (T_{stable} $+23.00 \rightarrow +25.09$), and a committee that mixes the two field orders takes its checks from whichever order holds its majority, so the order effect reappears at committee level. A decoder-enforced schema leaves it in place ($+9.95$ on gpt-5.4, against $+7.83$ declared-only). Rerun at temperature 0.7 in its own batch, the design still puts T_{stable} at $+20.89$ pp at its floor, and a same-batch contrast shows the baseline rising by about a point while T_{stable} does not detectably move (Appendix M.4). Splitting the call in two, checks first and the verdict second, raises rule compliance, but its checks land with the checks-first single call's. Abstaining where two calls disagree raises gold-scored accuracy, but by using the effect, not by removing it.

Limitations in brief. Randomised evidence covers eight model $\times$ task configurations in ten pre-specified designs (Section 2, Appendix E); the configurations of Table 2 were chosen for their existing replicates rather than sampled from a population of judges, and one unselected judge model was also run. Open-weight judge models ran on an unpinned managed serving stack. Every confirmatory design but one is at temperature 0, where the same-order baseline measures backend nondeterminism only. No experiment here establishes a cross-task ordering of which field order is more accurate (Appendix H), or that these magnitudes transfer beyond the configurations tested. None re-reviews the constraints that migrate or prices a missed error against a wasted review (Appendix P).

6 Related work

Prompt formatting alone moves task accuracy by tens of points (Sclar et al., 2024), position moves an LLM judge's verdict (Wang et al., 2024; Zheng et al., 2023), multi-prompt reporting is the standing remedy (Mizrahi et al., 2024), schema descriptions move task accuracy (Lin, 2026), and field order on a two-field schema moves verdicts (van Nuenen & Sachdeva, 2026); swapping `score` and `reasons` in a judge's declared JSON moves its mean score, with the reasons reported as similar rather than measured (Chen et al., 2024). That paper conjectures an autoregressive cause: the score reads the reasons already written. Section 3's declared-order result shows it is not the whole account.

Direct tests of the check channel are scarce. The nearest work measures neighbouring quantities: per-constraint correctness changes under input-side perturbations, scored against gold (Lee et al., 2026b); explanations that change while the decision does not (Tapwal et al., 2026); criterion coupling across *separate* calls, elicited that way precisely to "remove within-prompt ordering and self-consistency effects" (Singh et al., 2026); prompt-driven variation separated from call noise by a debiased within- versus across-prompt estimator (Feng et al., 2026), of which our two field orders are the two-prompt case (Appendix B); and compilation approaches that build checks into verdicts without testing emission order (Chen et al., 2026).

7 Conclusion

Moving one block inside a declared output schema changes a structured LLM judge's check values more than its own reruns do, in every configuration where we could measure it but one. On claim verification much of that change comes while the verdict stays the same, including when it is correct; on safety rating the checks still move, but mostly together with the verdict. Against human per-constraint gold, *which* checks are right migrates while mean accuracy barely changes, and a model ranking built on the same judge moves with the field order. So agreement at the accuracy level is not agreement at the check level, and a structured judge's per-check audit trail is not portable across serialisations by default. None of the remedies we tested removes the effect, so it has to be measured, and only against the judge's own reruns. We therefore recommend two things: report the field order used, and read check-level evidence against a same-order replicate. Both recommendations apply as much to the other key orders we tested, which move the checks too.

AI USE STATEMENT

In this work, we used generative AI tools for drafting and editing the manuscript; for literature-search support; for writing and refactoring the experiment and analysis code, including the data-processing steps (the evidence-flattening filter, the class mapping and the frame recount); for assistance with the algebraic derivation of Appendix B and its correspondence to Feng et al. (2026), which the authors then checked line by line; and for feedback that suggested and refined candidate hypotheses, experimental design, statistical analysis and presentation — several claims in this paper were narrowed or withdrawn after such feedback. The authors made every design, interpretation and reporting decision, verified cited sources against the primary literature, and reviewed all AI-assisted code and prose; the executable checks behind the reported results are described in the Reproducibility statement. Generative AI authoring tools did not produce the experimental observations; the evaluated LLM judges are the *objects* of study. The authors take responsibility for the manuscript and artifacts, including any text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

The judges studied here are evaluated on existing public resources — FEVEROUS, DICES-350, MCJudgeBench and, for the ranking design, the WildBench task set and its released model outputs — and on the scientific-data corpus of Anonymous (2026), literature-mined instances whose expert labels were collected outside this work; no human-subject data were collected and no annotators were recruited for it. Each resource is used under its own release terms and is cited at first use. The corpus's annotator identifiers are pseudonymous, and its source documents are open-access literature, cited by DOI on every record. Its records, benchmark inputs and verifier prompt ship *inside* this paper's artifact, because the scientific-data loaders are digest-frozen to the path they sit at and cannot be repointed without breaking their freeze hash; a NOTICE beside them states their source and licence terms. DICES-350 consists of conversations rated for safety by its own annotators; we use its released expert labels only. The findings concern the reliability of automated evaluation and carry no dual-use content; the artifact releases model outputs, prompts and analysis code, not model weights. The one limitation with an ethical edge is stated in Section 5: the paper does not price a missed error against a wasted review, since workload is counted at one unit per constraint (the supplementary material, part P).

REPRODUCIBILITY STATEMENT

The appendix opens with a guide that maps each conclusion in the body to the appendices supporting it; a reader following one thread should start there.

Tables 7 and 9 — both precedence readings, the ambiguous shares, and each computable identity gap estimate *and* interval — are recomputed from the released run files by assertion scripts that exit non-zero if a figure moves *or if any run it needs is absent*, so a missing configuration cannot read as a pass; the per-configuration matrices and the gold-side movements are recomputed by their own pipelines and printed, not asserted. The coverage is not uniform, and this statement says which figures fall on which side. Every plotted value is recomputed from the run files and asserted as well (Figure 2), and the figures that quote data directly are generated rather than drawn: Figure 1's skeleton is rendered from the prompt builder and checked to be a line-permutation across field orders, and its instance panel's claim, evidence and check values are asserted against the frozen instance and the pinned runs before the figure is emitted. An anonymized copy of the artifact is included in the supplementary material (`field-order-artifact.tar.xz`, beside the supplementary PDF). The artifact contains: the runner (one prompt skeleton per task, one field-order knob) and the rendered core prompt templates with their digests (`prompts/`), the frozen plans with their amendments and execution records (`PREREGISTRATION_*.md`) and CSV exports of the tables the appendix summarises rather than prints (`csv/`), the SHA-256-frozen rubrics, decision rules and instance samples; the raw response text for every call with its emitted key order extracted before parsing; per-run manifests naming endpoint, decoding parameters and run window; the matrix analyzer, the gold-side pipeline, and the rule-precedence, precedence-condition identity gap, gold-flip, verdict-stable-conditioning and sufficiency-disambiguation analyses; and the two-loci comparison in full — its two task variants, its preflight, its schedule generator and seed, its analysis script and the plan frozen before its first call. The shortest path from the artifact to the two main-text results

tables is one command, `check_repro.py`: it reruns *every* frozen analysis those tables draw on — including the no-filler control, which Table 3 also reads — compares each against its committed results file key by key, and checks that the generators reproduce the rendered tables from those same files, and the packaging step runs the *whole* gate suite inside a fresh extraction of the anonymised archive, so what is verified is the artifact a reader downloads rather than the working copy, exiting non-zero and naming the first figure that moved. For anonymity, the archive's code and plans ship with their process notes removed, so a cleaned frozen file no longer has its registered digest: `check_frozen_equivalence.py` checks each one against the comment-free form of the version frozen, and each plan lists the digests of its cleaned files as shipped. `run_all_gates.sh` runs every gate in the paper. The ERROR-row policy (dropped, never imputed) is applied uniformly; Appendix C states it, each design's appendix counts its own incidences, and every run's count ships in the manifests. kimi-k2-thinking's claim-verification identity gap configuration is reported with its estimate and interval, its safety configuration as not estimable (Appendix G). The one exclusion, the claim-verification thinking pair, is stated with its reason in Appendix C.

The ten pre-specified designs ship with their repository-frozen analysis plans (frozen locally and digest-verified, not externally registered) and per-run pins binding every analysed file to a digest. The nine with a reconstructible, enumerable assignment also ship the schedule generator and its master seed, so their allocations re-derive (each generator has a `--verify` mode; the no-filler control runs on the MCJudgeBench gold design's schedule by design; the artifact README maps every design to its plan and scripts), and the analysis refuses to fall back to globbing a run directory; `check_design_counts.py` enumerates the design points and model $\times$ task configurations the paper counts, and fails on any wrong count anywhere in the source. The tenth, the earlier five-block design, states its reference set as an assumption rather than enumerating a draw (Appendix F). Three of the same-batch controls — the two-stage pipeline, the reworded-instructions condition and the same-batch temperature contrast — ship frozen plans, digest-gated run scripts and preflights with positive controls in the same way. The gold design's analysis is recomputed and checked against its committed results file, and its direction accounting carries an in-code conservation assertion; the per-configuration gold descriptives are printed by the same gated analyzer rather than separately asserted. Two checks written after the exact design's result was known and the post-freeze edit to its analysis script are disclosed, with both digests, in Appendix E, where the edit is recorded as an amendment to the frozen plan; Appendix C accounts for the other frozen files edited after their designs' runs. `check_freeze_digests.py` and `check_registered_versions.py` recompute the digests and re-run the registered versions.

REFERENCES

Rami Aly, Zhijiang Guo, Michael Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. FEVEROUS: Fact extraction and VERification over unstructured and structured information. In *NeurIPS Datasets and Benchmarks*, 2021.

Anonymous. When substantial κ hides systematic disagreement: Auditing LLM-as-a-judge evaluations beyond aggregate agreement, 2026. Manuscript submitted for publication.

Lora Aroyo, Alex S. Taylor, Mark Diaz, Christopher M. Homan, Alicia Parrish, Greg Serapio-Garcia, Vinodkumar Prabhakaran, and Ding Wang. DICES dataset: Diversity in conversational AI evaluation for safety. In *NeurIPS Datasets and Benchmarks*, 2023.

Xi Chen, Jie Mu, Mo Xuan, and Qun Shao. Graph-structured rubrics: Compiling rubrics into typed evaluation graphs for LLM judges. *arXiv preprint arXiv:2608.12097*, 2026.

Yi-Pei Chen, KuanChao Chu, and Hideki Nakayama. LLM as a scorer: The impact of output order on dialogue evaluation. *arXiv preprint arXiv:2406.02863*, 2024. Presented as a poster at the AAAI 2024 Spring Symposium on User-Aligned Assessment of Adaptive AI Systems, which has no formal proceedings.

Jacob Devasier. Reasoning or rationalization? The role of justifications in masked diffusion models for fact verification. *arXiv preprint arXiv:2603.01190*, 2026.

Hengxin Fan. Capacity, not format: Rethinking structured reasoning failures. *arXiv preprint arXiv:2606.09410*, 2026.

Pengbin Feng, Chunlei Meng, Daozheng Qu, Zhilin Zhang, Haoran Liu, and Jiekai Wu. Second-order response laws for LLM judges: Debiased estimation of prompt instability. *arXiv preprint arXiv:2608.16253*, 2026.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, et al. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*, 2023.

Ivan Yee Lee, Loris D'Antoni, and Taylor Berg-Kirkpatrick. The format tax. *arXiv preprint arXiv:2604.03616*, 2026a.

Jaeyun Lee, Junyoung Koh, Zeynel Tok, Hunar Batra, and Ronald Clark. MCJudgeBench: A benchmark for constraint-level judge evaluation in multi-constraint instruction following. In *Proceedings of the Fifth Workshop on Generation, Evaluation and Metrics (GEM)*, pp. 205–221, 2026b. doi: 10.18653/v1/2026.gem-main.23. URL https://aclanthology.org/2026.gem-main.23/. arXiv:2605.03858.

Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Faeze Brahman, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. WildBench: Benchmarking LLMs with challenging tasks from real users in the wild. *arXiv preprint arXiv:2406.04770*, 2024. URL https://arxiv.org/abs/2406.04770.

Sin-Ying Lin. Your prompt is not the only prompt: How much do LLMs weight structured-output schema descriptions? *arXiv preprint arXiv:2608.08254*, 2026.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. State of what art? A call for multi-prompt LLM evaluation. *Transactions of the Association for Computational Linguistics*, 2024. arXiv:2401.00595.

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. arXiv:2310.11324.

Divyansh Singh, Reza Davari, and Afra Mashhadi. RADAR: Rubric-aware dependency and redundancy analysis for LLM-as-judge evaluation. *arXiv preprint arXiv:2608.01810*, 2026.

Zhi Rui Tam, Cheng-Kuang Wu, Yi-Lin Tsai, Chieh-Yen Lin, Hung-yi Lee, and Yun-Nung Chen. Let me speak freely? A study on the impact of format restrictions on large language model performance. In *Proceedings of EMNLP 2024: Industry Track*, pp. 1218–1236, 2024. doi: 10.18653/v1/2024.emnlp-industry.91.

Riya Tapwal, Abhishek Kumar, and Carsten Maple. Faithful or fabricated? A causal framework for rationalization bias in LLM judges. *arXiv preprint arXiv:2605.23970*, 2026.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.04388.

Tom van Nuenen and Pratik S. Sachdeva. The fragility of moral judgment in large language models. *arXiv preprint arXiv:2603.05651*, 2026.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024. arXiv:2305.17926.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023. arXiv:2306.05685.

APPENDIX GUIDE

group	supports	appendices
templates, estimands, frozen files	the primary template's sufficiency criterion and the two precedence clauses (digests and the full template: the supplementary material); every endpoint with its formula; how much confidence moves and whether it is a signal; frozen files and disclosure	A, B, C
designs and inference	design table, serving provenance, timeline; the eligible frame; the randomised designs and what was frozen; every p and its reference set	D, E, F
the effect, run by run	run-level tests, every pairing, the identity gap table; the correct-verdict split, gold accuracy and flips	G, H
robustness and alternatives	criterion anatomy; sufficiency disambiguation; full-rubric paraphrase; verdict-level effects, rule precedence, the reasoning-first control; decoder-enforced schema, two-stage pipeline, matched wording edit, temperature	I, J, K, L, M
where the effect varies	what each design identifies, and reweighting; scientific-data attribution	N, O
consequences for auditing and ranking	the MCJudgeBench experiments; systematic positioning; whether the ranking a judge produces moves with the field order	P, Q, R
discussion: other edits, verdict level, remedies	the perturbation control; input-side against output-side reversal on one denominator; the full cross-task matrix and configuration cases; majority-vote committees	S, T, U, V

A PROMPT TEMPLATES: THE SUFFICIENCY CRITERION AND THE PRECEDENCE CLAUSES

Every template is emitted by the builder the runs used (`runner.build_template`), not transcribed, and each is gated at generation time: its SHA-256 must equal the value the frozen digest file (`runs/TEMPLATE_HASHES.json`) or its own pinned run manifest recorded at run time, and its character count must equal the `template_chars` every pinned run header carries. the supplementary material, part A lists the templates of the four tasks' two field orders, the paraphrased rubrics and the 2×2's checks-first conditions with their digests; every other condition's template is rebuilt by the same builder from the task modules its run manifests record. The full text of the listed templates ships in the artifact as `prompts/<task>_<field order>.txt` beside `prompts/TEMPLATE_DIGESTS.json`; regenerating them (`make_prompt_appendix.py`) re-verifies every digest, and the gate suite runs that regeneration. The primary configuration's sufficiency criterion — the clause Appendices G and J adjudicate — is printed here; its response-format block (Figure 1 renders the skeleton) and the complete template, claim verification under verdict-first, are printed in the supplementary material, part A. The two field orders of a task differ *only* in where the `rating+confidence` block sits in the declared output schema (Figure 1), asserted at generation time as line-multiset equality of the two templates; the response-format block is 687 characters on claim verification and 655 on safety rating. The paraphrased rubrics of Appendix K reword the preamble, the six criterion descriptions and the decision-criteria prose, while the `##` input markers, field placeholders, check keys, quoted value literals and the OUTPUT schema block are byte-identical to the originals (asserted by `check_pp.py`). The 2×2 of Appendix L appends exactly one precedence sentence, reproduced below. `{claim}`-style slots are the per-instance substitution points.

FEVEROUS, VERDICT-FIRST: CRITERION 6 OF THE PRIMARY TEMPLATE (2450 CHARS, SHA256 44AD8E0EC952A5C1...)

```
### 6. SUFFICIENCY (evidence_sufficient)
true if the evidence shown is enough to decide the claim either way;
false if it does not cover what the claim asserts.
```

feverous, checks-first. Identical text; the schema block of the OUTPUT section lists the 6 checks before the `rating+confidence` pair instead of after it. Equality of every non-schema line is asserted at generation time, for this task and for every other template in the digest table of the supplementary material, part A.

The two precedence clauses of the 2×2. One sentence appended to the decision criteria, read back from the run manifests:

```
insufficiency-first:
  PRECEDENCE: if evidence_sufficient = false, answer NOTENOUGHINFO even
  if one or more of checks 1-5 is also false.
hard-fail-first:
  PRECEDENCE: if any of checks 1-5 is false, answer REFUTED even if
  evidence_sufficient is also false.
```

B ESTIMANDS AT A GLANCE

The two field orders are *verdict-first* and *checks-first* throughout; the artifact's schedules and run files name the first `rating-first`, after the field it moves.

The symmetric estimand is a half-squared L_2 distance. With p_i and q_i the distributions of the class-mapped check vector on instance i under the two field orders, the estimand's expectation is $\frac{1}{2}\|p_i - q_i\|_2^2$: zero if and only if the two distributions coincide, never negative, and positive for a dispersion-only difference as well as for a relocation. It assumes the endpoint is stationary across the interval between a field order's two runs (Appendix D weighs the evidence), and it is the two-prompt census case of Feng et al. (2026)'s within-minus-across estimator, their $\frac{K-1}{K}$ factor at $K = 2$. The derivation, the dispersion example, the one-sided bias it removes and the empirical check are in the supplementary material, part B.

The confidence field: how much it moves, and whether it is a signal. The estimand is the order effect of the `rating` + `confidence` block, because the two fields travel together in the schema and the reasoning-first condition does not separate them (there too both precede the checks). What the runs can bound is whether the confidence *value* moves across orders, and over the six confidence configurations (three judge models $\times$ two tasks) it moves very little: the mean shifts by at most 0.062 on a 0.50–1.00 scale, the mean per-instance absolute difference is 0.003–0.062, and 66–95% of instances return an identical value — the worst configuration is 12% of the scale with a third of its instances changing. None of this bounds how much of the order effect runs *through* confidence, which is why the five-block design's rating-only ablation (the supplementary material, part E) pins `confidence` after the checks in both conditions instead of reasoning from these figures.

Is the confidence field a signal at all? On claim-verification gold, confidence ranks a gold-correct verdict above a gold-wrong one with AUC 0.59–0.66 per judge model, no per-run value touching 0.5: a weak-to-moderate marginal correctness signal, not near chance, whose causal role in the block-order effect stays unidentified. Its variance is model-dependent (9–32% of the 0.50-wide scale), so "nearly constant" is not model-general; the table and the reasons the obvious free mediation substitute is not used are in the supplementary material, part B.

C FROZEN DESIGNS AND REPRODUCIBILITY

Rubrics, decision rules and instance samples were fixed and SHA-256-recorded before any run — locally, with no third-party registration, so we call them frozen rather than preregistered. Two of the files recorded then no longer match their digest; the artifact carries the evidence for each: `tasks/feverous.py` was edited 12 minutes *before* that task's first run, and re-rendering the frozen sample with the current loader reproduces 999 of 999 instances byte-identically (`rerender_verification.json`), so the edit added a data-loading fast path and not rubric text; `runner.py` changed later, when it gained the `--precedence` flag for the taught-precedence 2×2 of Appendix L, and every run header records its own precedence value (non-empty on the 85 precedence-condition runs, empty on the other 115). `tasks/dices.py` matches its recorded digest exactly. One manifest per run ships in `runs/manifests/`, and `runs/RUN_PINS.json` pins the exact file and digest behind every reported figure, so a later rerun cannot silently substitute one; the manifest index states per field which values were recorded at run time and which were derived afterwards (the artifact README lists the fields). Every experiment added after the first freeze (Appendices J–T, O and M) follows one discipline, stated once here: a plan frozen with digests before the first call, a preflight that asserts the design's single-edit

Table 4: Every endpoint the paper's frozen plans name. V is the emitted verdict and X the class-mapped check vector (pass / fail / soft, Section 2); VF_i and CF_j are the i-th verdict-first and j-th checks-first run, and k_i is instance i's number of listed constraints. A *same-order* pair is two runs of one field order, a *cross-order* pair one run of each. Diagnostic statistics that are not endpoints — rule compliance, judged-`no` precision and recall, confidence AUC, emitted-order compliance — are defined where they are used.

quantity	formula	unit, denominator	pairs; reported in
T_{stable} (primary, exact design)	$\Pr(V_{\text{same}} \wedge X_{\text{changed}})_{\text{cross}} - \Pr(\cdot)_{\text{same}}$	pp; instances all runs answered	all cross vs all same pairs of a configuration's 20 runs; Table 2, Section 3
T_{correct}, T_{wrong}	T_{stable} split by whether the agreed verdict is gold-correct; the parts add	pp; same denominator	as above; Appendices G and H
check-vector identity	share of shared instances whose entire X is identical between two runs	% of shared instances	one pair; Table 7, Figure 2
check-vector identity gap (secondary)	mean within-order identity − mean cross-order identity, i.e. the excess of cross-order over same-order change of X; over the same pairs $= T_{\text{stable}} +$ the excess of changes that also move V	pp	within: both field orders; cross: all $\text{VF}_i \times \text{CF}_j$; Tables 2, 7, Figure 2
verdict-stable identity gap (conditional)	identity gap on the subset a pairing's verdicts agree on	pp of agreeing instances	descriptive; Appendix G
separation margin (secondary)	lowest within-order identity − highest cross-order identity	pp	a contrast's runs; Appendices F and G, the supplementary material, part E
per-check agreement	mean over the six checks of the per-check agreement rate	%	one pair; Appendix G
selected-policy disagreement (c)	emitted verdict ≠ deterministic rule applied to the run's own checks	% of instances, per run	none; Figure 3, Appendix L
precedence-free violation	(c) counted only where every admissible reading of the rubric agrees	% of instances, per run	none; Appendix L, Table 9
gold accuracy (d)	emitted verdict = gold verdict	% of instances, per run	none; Table 5
M_1 correctness migration	share of (instance, listed constraint) units whose gold-scored correctness differs between two runs	pp of 1,022 units	cross − same over 20 runs; Appendix P
M_2 verdict-stable correctness migration (primary, gold design)	M_1 restricted to pairs whose verdicts agree	pp of units	cross − same; Table 3
M_{2c}	M_2 further requiring the agreed verdict to be gold-correct	pp of units	cross − same; Table 12
M_3 repair-list divergence	mean over instances of $\lvert F_a \triangle F_b \rvert / k_i$, F_r the constraints run r judged `no`	pp	cross − same; Table 3
τ_b concordance difference (primary, ranking design)	mean Kendall τ_b between two runs' model rankings, same-order − cross-order	τ units; 12 models	all same vs all cross run pairs; Appendix R

invariants on every rendered prompt and fails on a deliberately sabotaged positive control, digest-pinned runs, and an analysis that refuses to report unless it recomputes the pinned figures. A file edited after its design's runs is recorded in that plan by a dated amendment. Eight plans carry such amendments, and a plan can carry more than one: seven analysis scripts gained documentation, printed labels, assertions or added outputs; one pin script records that it pins the complete one of two final files for each of five blocks whose runs were retried after network failures, the by-hand choice its design requires; and in four plans the runner is registered as it was before it gained a block that only the no-filler and ranking tasks use. Each registered version ships beside the edited one, and `check_registered_versions.py` re-runs every registered analysis on the pinned runs, finds every value it reports unchanged, and asserts that the runner's added block is its only difference. The per-experiment paragraphs name only what is specific to each; the artifact README maps every figure to the script that produces it.

Every response's raw text is retained and its emitted key order extracted before parsing; on the exact design the split is $23{,}457$ verdict-first against $23{,}464$ checks-first answered calls, with zero mismatches. Across the 754 verdict-, checks- and reasoning-first run files the artifact ships ($535{,}090$ answered responses), the first emitted key is the requested one on every answered call except in four places, each reported with its design: the decoder-enforced qwen3-next runs, whose decoder writes the keys alphabetically (Appendix M.1); the claim-verification thinking pair, excluded below (383 checks-first responses emit the verdict first); gemma-4's reasoning-first runs on claim verification, which omit the requested `reasoning` field on 214 and 245 of 999 responses (232 of 959 in a superseded attempt at the replicate); and one response of one precedence-design pass on deepseek-v3.1 (1 of 995). Every other verdict-first and checks-first run complies on every answered call, on claim verification and on every other task (`check_emission_compliance.py`). In the reasoning-first control (Section 2), llama-3.3 emits reasoning first on 100% of instances in both of its claim-verification reasoning-first runs (selected-policy disagreement 10.22%, 102 of 998, and 9.71% on the rerun; Appendix U), gemma-4-26b on only 75.5–78.6%, so gemma's reasoning-first configurations are partially treated and we read no control result off them. The emitted-order check is part of the released assertion suite and reports this exception.

ERROR rows are dropped, never imputed. In the broad matrix, where the incidence warranted it, the affected pass was re-run for those rows. Outside the qwen3-next-thinking pair below, its largest incidences were 52 and 40 of 999 rows (a kimi-k2-thinking and a gemma-4-26b claim-verification replicate) and 46 and 8 of 350 (a gemma-4-26b and a gemini-3.1-pro safety-rating run); each pass was re-run to a complete file, and every attempt is released. The largest incidences left in design passes, where the rows were dropped rather than re-run, are 54, 47 and 46 of 999 (a deepseek-v3.1 precedence pass and two qwen3-next sufficiency runs); the two-loci comparison dropped its two ERROR rows without a retry (Appendix T). The qwen3-next-thinking configurations run at vendor-required sampling (temperature 0.6, top-p 0.95; greedy decoding loops in thinking mode), with the override recorded in each run header; on claim verification the pair is not reported, because a relaunch with schema salvage returns usable check dictionaries on 984 of 999 verdict-first rows but only 718 of 999 checks-first (15 against 281 ERROR rows), so a paired analysis would drop 284 of 993 instances, 28.6%, concentrated in the checks-first condition, and the requested checks-first ordering was not realised either (53% of its rows emit the verdict first) — the pair fails as a field-order contrast on two independent grounds.

D Design table, provenance and execution timeline

Design table. The design table (the supplementary material, part D) is generated from the run inventory, so its counts cannot diverge from the runs; replicate coverage is stated *per field order* as replicated / run, a "replicate" being a distinct run slug (a retry that recovers ERROR rows on the same slug is not one). Claim verification and safety rating ran the verdict-first, checks-first and reasoning-first orders, scientific data the first two through this runner, and the precedence 2×2 four Latin-square passes per configuration.

Two scope notes the counts do not carry. Decoding is $T{=}0$ throughout except qwen3-next-thinking; compliance with the requested first key is 100% except on the two gemma-4 reasoning-first configurations; ERROR incidence is 0.13% over the design's $47{,}952$ rows and is *not* uniform across them — one of the 48 runs carries 54 of the 64 ERROR rows, 5.4% of that run, against 0.4% in its paired

condition (dropping that pass moves nothing that matters: Appendix L); the drop-never-impute policy those counts follow is Appendix C's. And the scientific-data row counts only configurations run through *this* runner: Figure 3's scientific-data row plots eight configurations because the remaining five come from the benchmark's own harness (Appendix U), which is why that row is pinned with its provenance rather than recomputed here. Splitting the two is the difference between the core matrix and its partial extensions.

The three roster exceptions — gemma-4's partially treated reasoning-first configurations, the unreported claim-verification thinking pair, and qwen3-next-thinking's vendor-required sampling — are itemised with their counts in Appendix C.

Roster. Two more open-weight families appear in the cross-task matrix: deepseek-v3.1 on all three tasks (and in the precedence design and the run-level table, Appendices G and L), and gpt-oss-120b, which shares the scientific-data row with a second Llama, llama-4-maverick. Alongside the open-weight roster: a matched thinking/instruct pair (qwen3-next thinking against instruct) and two proprietary API models, `gemini-3.1-pro-preview` and `gpt-5.4`. The gpt-5.4 column is a post-specification roster extension; on SSDV the matrix reuses the benchmark's released verifier template verbatim (prompt-hash asserted), and a same-batch $2{+}2$ adds two identity gap configurations (Appendix O).

Serving provenance, and what we did not record. Open-weight models were served through the vendor's managed endpoints under the model identifiers listed in the run manifests, at temperature 0 (the temperature design excepted) with a fixed maximum token budget. We did not record checkpoint revision hashes, chat-template versions, serving-engine builds, or quantization settings. We used undated endpoint identifiers and recorded no provider-side revision IDs; consequently the artifact cannot establish that the served backend remained fixed — a bound on third-party replication of that stack. What it means for internal validity differs by design: for the randomised designs, randomising the within-block order removes *deterministic* field order–time confounding and the exact test accounts for chance alignment over its enumerated support, but neither establishes backend stationarity or fixes a direction for the bias such backend movement would leave. For check-vector identity the direction is known: every same-order pair spans two blocks while a tenth of the cross-order pairs sit inside one, so an endpoint that moves between blocks lowers same-order identity more than cross-order identity. The verdict-stable joint event has no such guarantee: with the two field orders identical, the verdict fixed within each half of the blocks and one check drawn Bernoulli(0.2) in the first five blocks and Bernoulli(0.8) in the last five, T_{stable} is $+1.78$ pp under the design's pair weights with no field order effect at all. What the design does allow is Appendix E's check: dropping the ten within-block cross-order pairs moves the estimates by at most 0.02 pp and leaves all six exact tests at the floor. The fixed-order matrix has no such protection; its same-window reruns constrain, but do not bound, what backend change could contribute (the execution timeline below). A replication is therefore reproducible at the level of "the endpoint published under this identifier at this date", not of a fixed weight-and-runtime bundle; future replications should use immutable or dated identifiers where available.

Execution timeline of the identity gap runs. For the six Table 7 configurations and the two proprietary claim-verification configurations whose identity gap it reports (the gaps between run completion times, in minutes: the supplementary material, part D), VF_1, VF_2 and CF_1 were executed back to back in one window on 2026-08-18 and CF_2 is the later replicate; two of the safety CF_2 runs (gemma-4, qwen3-next) were collected on 2026-08-20 with the rendered template and a SHA-256 of every per-instance prompt written to a digest file before the run and checked afterwards — provably the same prompt as CF_1. Two things follow, in opposite directions. $\text{VF}_1 \to \text{CF}_1$ is 2–62 minutes, comparable to $\text{VF}_1 \to \text{VF}_2$ at 1–38, so the $\text{VF}_i \times \text{CF}_1$ pairings are measured in the same window as the verdict-first replicate and a field order effect there is not confounded with a separate-window endpoint effect; the $\text{VF}_i \times \text{CF}_2$ pairings are not. What makes the cross-order term interpretable anyway is that all four pairings agree to within 1.2 pp in every open-weight configuration (Appendix G) — evidence against a pairing-specific artefact, not against a shift common to all four, since the pairings share runs. Conversely, in the six open-weight configurations $\text{CF}_1 \to \text{CF}_2$ spans 708–3,208 minutes, at least 54 times the within-verdict-first gap; if elapsed time adds disagreement, it depresses within-CF identity, which still lands within 2.2 pp of the within-VF term there. On the two proprietary configurations CF_2 followed CF_1 by $1{,}093$ and 89 minutes (gemini-3.1-pro, gpt-5.4), and within-CF

identity sits 2.7 and 13.3 pp below within-VF (Appendix G). This is a directional argument, not a bound.

The claim-verification eligible frame. The *frame* is not the released development file: of its 7,891 instances, 1 carries no claim text and 88 have gold evidence that does not resolve to non-empty text through the official wiki database, leaving the 7,802 the diagnostic sample was drawn from. Its not-enough-info share is $491/7{,}802$ against $333/999$ in the balanced sample — a 5.3-fold over-representation, which is why the fixed-order claim-verification figures are reported both balanced and frame-reweighted (the randomised endpoints are label-balanced by construction and not reweighted, Section 2). *Frame*, not *population*: the two exclusions are properties of the released file, not of any judging task.

E THE RANDOMISED REPLICATIONS, AND WHAT WAS FROZEN BEFORE THEM

Design	Status	Configurations	Assignment	Main purpose
Fixed-order matrix	exploratory	three tasks	fixed VF→CF	locate check-vector effects on two tasks (a third: Appendix O), verdict/rule-level on three
Five-block	pre-specified	selected	written schedule	rating-only ablation; reference set assumed (the supplementary material, part E)
Exact	**pre-specified replication**	2 claims, 1 safety	uniform over 252	test T_{stable}; support enumerable
Key reversal	**pre-specified**	1 claims (key reversal)	uniform over 252	randomise the K contrast; independent batch
Unselected judge	**pre-specified**	1 claims (unselected)	uniform over 252	test T_{stable} on a judge model not chosen for its replicates (Appendix E)
Temperature	**pre-specified**	1 claims at $T{=}0.7$	uniform over 252	rerun the primary configuration where the same-order baseline is not backend noise alone (Appendix M.4)
Decoder-enforced	**pre-specified**	2 claims (decoder-enforced)	uniform over 252	the exact design under a strict schema (Appendix M.1)
Gold audit	**pre-specified**	2 MCJudgeBench	uniform over 252	gold-scored audit endpoints (Section 4, Appendix P)
No-filler	**pre-specified**	2 MCJudgeBench (the gold audit's schedule)	uniform over 252	the gold design without filler slots (Appendix P)
Two-loci	**pre-specified**	1 MCJudgeBench, two loci	uniform over 252	input-side against output-side reversal (Appendix T)
Ranking	**pre-specified**	1 ranking (WildBench)	uniform over 252	whether field order moves a twelve-model leaderboard (Appendix R)

Resampling runs instead of instances (post hoc). Every interval above resamples instances, conditional on the observed runs. Resampling a design's ten blocks with replacement instead (same-order pairs of a run with itself excluded) gives run-level intervals for the same statistics: T_{stable} $[+22.55, +23.44]$, $[+7.29, +8.37]$ and $[+0.34, +0.72]$ for Table 2's three rows, identity gap $[+32.70, +33.57]$, $[+21.10, +22.83]$ and $[+18.86, +19.64]$, and for the primary endpoint of Table 3 $[+0.99, +1.20]$, $[+0.52, +1.05]$, $[+0.75, +1.02]$ and $[+1.15, +1.56]$. Each is narrower than the instance interval: within a design, resampling runs moves these estimates less than resampling

instances does, so the paper keeps the instance bootstrap, which on safety rating covers zero where the run-level interval does not. Neither resampling sees the variation between batches and designs, which Table 6 reports as ranges (`analyze_run_level_interval.py`).

Two designs, never pooled. The first two randomised designs of the claim-verification contrast ran a day apart, and neither is pooled with the other or with any later design of the inventory above. The **exact** design (Section 3, 2026-08-22) draws its within-block condition order from a support the analysis enumerates, so its p is an exact randomisation p; the **earlier five-block** design (2026-08-21), pre-specified in the same sense with its own frozen primary endpoint, wrote its condition orders into its run scripts, so its reference set is assumed rather than executed (Appendix F), and it carried a third condition, the *rating-only ablation* (`confidence` pinned after the checks in both conditions so that only `rating` moves). The body's table is the exact design's.

What was frozen, and when. Every existing FEVEROUS pass had run verdict-first then checks-first, minutes apart, in all ten pairs across both judge models, so field order was collinear with position within the pass. Each design was fixed in writing before its own first call — in the repository, with the runner verifying the file's presence and digest. That is a freeze, not a third-party registration. The frozen text names the conditions, sample, prompt digests, randomisation seeds, one primary endpoint, the secondary endpoints, ERROR handling, exclusion rules, and a reporting policy committing to publish the result whatever its sign; the runner refuses to execute unless that plan is present and unless its own digest, the analysis script's and the runner's all appear in it, so an edited script cannot run under a frozen design. The plans ship with the artifact (`PREREGISTRATION_*.md`), each with its amendments and an execution record.

The exact design: two conditions, ten blocks per configuration. Condition **A** is `rating`, `confidence`, `checks`, `issues`, `reasoning`; condition **B** moves `checks` to the front. The assignment scheme is Section 2's: a block runs both conditions back to back over one instance order, so the pair shares whatever the endpoint was doing in those minutes, and the within-block order is drawn uniformly from the 252 allocations by a committed generator seeded from a master seed and the configuration's name (the reference set: Appendix F). The instance intersections are n=973, 997 and 341 against the per-run completeness floors the plan set, 950 and 333. Before any arithmetic, a gate asserts that a configuration's twenty runs share *one* input digest per instance — the rendered prompt with the RESPONSE FORMAT block removed — and *two* full-prompt digests, one per condition, with one template length (2,450 characters on claim verification, 2,669 on safety rating): the conditions differ by field order and not by content. The served model is pinned per response, `meta/llama-3.3-70b-instruct-maas` and the snapshot `gpt-5.4-2026-03-05`, one value each across all 20 runs of a configuration. The safety configuration answers 341 of 350 instances; a worst-case missingness bound on its full-sample T_{stable} is $[-2.07, +3.07]$ pp (the supplementary material, part E).

Endpoints, and one edit to the analysis script. The primary endpoint is T_{stable} as Section 2 defines it, computed over all pairs of the configuration's 20 runs: 100 cross-order pairs against 90 same-order. Whole-vector identity gap is secondary; both are in Table 2. Three further secondary endpoints the plan named are in Table 5, and reporting them required editing the analysis script after the primary result was known, because the frozen script did not implement them. The plan records the edit as four amendments: the pre-edit script ships with the artifact, verifies against its registered digest, and reproduces every primary and identity gap figure verbatim ($+23.00$, $+7.83$, $+0.51$ and $+33.13$, $+21.96$, $+19.24$, each at $p_{\text{exact}} = 0.0079$); the current script's output is a strict superset of the frozen one's, both registered digests verify (`check_freeze_digests.py`), and no primary number depends on the edit.

The class mapping, and what happens without it. Section 2's class mapping folds the not-applicable family into *pass* (10.1% of the check values returned in the exact design; none on safety rating, whose criteria have no such branch). Giving them a fourth class re-runs the exact design's own analysis: it increases all four claim-verification estimates (T_{stable} $+24.00/+8.76$ against the reported $+23.00/+7.83$), leaves the two safety-rating estimates unchanged and keeps all six exact tests at $p_{\text{exact}} = 0.0079$. So the reported mapping is the conservative one; a gate fails the build if any

configuration ever changes sign or leaves the floor under the alternative, and the full figures are in the supplementary material, part E.

Table 5: The three secondary endpoints the frozen plan named, complete. **Top**: per-check anatomy — cross-order minus same-order pair disagreement on each of the six criteria, all three configurations. **Bottom**: per-condition gold accuracy (mean over that condition's ten runs) and per-condition selected-policy disagreement under each resolution of the precedence the taught criteria leave unstated (insufficiency-first / hard-fail-first) — on the claims configurations these are *not* violation rates (the clauses conflict), on the safety configuration they never co-fire, so there the figure is a violation rate.

Configuration	Criterion	cross	same	Δ
Claims, llama-3.3-70b	`entities_match_evidence`	6.72	1.03	+5.69
	`numbers_match_evidence`	6.98	1.49	+5.49
	`relations_match_evidence`	11.17	2.45	+8.72
	`temporal_match_evidence`	2.75	0.99	+1.76
	`no_contradicting_evidence`	10.20	1.88	+8.32
	`evidence_sufficient`	20.02	1.70	+18.32
Claims, gpt-5.4	`entities_match_evidence`	11.64	3.69	+7.95
	`numbers_match_evidence`	9.99	3.35	+6.64
	`relations_match_evidence`	13.77	4.18	+9.58
	`temporal_match_evidence`	7.72	3.18	+4.54
	`no_contradicting_evidence`	9.65	3.17	+6.49
	`evidence_sufficient`	13.11	4.17	+8.94
Safety, llama-3.3-70b	`no_harmful_advice`	5.45	0.46	+5.00
	`no_hate_or_stereotyping`	10.47	0.41	+10.06
	`no_violence_or_self_harm_endorsement`	1.64	0.21	+1.44
	`no_misinformation_risk`	3.64	0.62	+3.02
	`no_illegal_or_pii_content`	0.38	0.13	+0.25
	`context_sufficient`	1.11	0.10	+1.02

	gold accuracy, per condition		selected-policy disagr., if / hf	
Configuration	verdict-first	checks-first	verdict-first	checks-first
Claims, llama-3.3-70b	60.37	63.58	5.60 / 11.60	24.27 / 13.21
Claims, gpt-5.4	68.59	62.62	2.78 / 10.77	11.66 / 0.44
Safety, llama-3.3-70b	42.90	55.87	5.78 / 5.78	0.97 / 0.97

What the secondary endpoints show. The per-check anatomy replicates the earlier design's: the criterion carrying the largest cross-order gap is the one the earlier design's primary contrasts give in all three configurations, with the eighteen per-criterion gaps agreeing to within 1.67 pp. The selected-policy endpoint reproduces the precedence reversal of Appendix L under randomisation: on gpt-5.4 the two resolutions move in *opposite* directions, which is the reversal, not noise around it, and safety rating is precedence-free. Gold accuracy moves too, and not in one direction: checks-first is more accurate on llama claim verification and on safety rating, and *less* accurate on gpt-5.4. A field order choice that moves accuracy by 13 pp on one configuration and -6 on another is not neutral, and these runs do not identify which field order is better.

What the verdict does when the checks move on safety rating (post hoc). On the exact design's safety configuration, over all 100 cross-order run pairs and the 341 instances all twenty runs answered, 20.76% of instances change their class-mapped check vector between a verdict-first and a checks-first run, against 1.52% within a field order (the identity gap of +19.24), and 94.6% of those changes move the verdict too. Every cross-order verdict change on this configuration, with or without a check change, has `Unsure` on one side, and 99.4% of those that come with a check change are a verdict-first run abstaining where the checks-first run commits. So the field order moves this judge into and out of abstention and never between `Safe` and `Unsafe`: a change in coverage, as in the fixed-order runs (Appendix H), not in judgement (`analyze_safety_abstention.py`, which first reproduces Table 2's safety row).

Post-hoc sensitivity on the ten-block runs. Neither analysis requires new data or alters the frozen primary analysis. Dropping the ten within-block cross-order pairs and re-enumerating moves the estimates by at most 0.02 pp; leave-one-block-out moves T_{stable} by at most 0.40 pp and identity gap by at most 0.53, and with each pair of blocks deleted (45 refits) no refit moves either endpoint by more than 0.6 pp or changes any sign, all six exact tests still at the floor (the ranges: the supplementary material, part E). These measure sensitivity within a design, not run-level confidence intervals; the variation that matters is between batches and designs, which Table 6 collects (below).

Randomised against fixed-order, on the same judge models. The body reports the three designs separately rather than pooling them; the between-design difference is this. Against the fixed-order symmetric estimates, randomisation left llama-3.3-70b essentially unchanged ($+32.15 \rightarrow +33.13$) and *increased* gpt-5.4 ($+15.18 \rightarrow +21.96$). The two designs differ in batch as well as in assignment, so the difference cannot be assigned to either. The four fixed-order configurations and their exact tests are in Appendix G.

The earlier five-block design (2026-08-21). Reported with its table in the supplementary material, part E, never pooled with the exact design: its identity gap is $+32.82$ and $+16.79$ against the exact design's $+33.13$ and $+21.96$; its rating-only ablation bounds a retained fraction of 84% and 94% on its own primary endpoint; its safety-rating amendment gives identity gap $+19.26$ at the blocked floor of 0.0625 with the joint verdict-stable term at $+0.42$, which the exact design reproduces ($+19.24$, $+0.51$). The DICES label polarity (Yes = unsafe, settled per rater against the corpus) is stated there too.

Selection of the confirmatory configurations. The two judge models were *not* the strongest exploratory movers: on Table 7 gpt-5.4's $+15.18$ is fourth of six claim-verification configurations, and on the run-level table (Appendix G) it is third of three, behind deepseek-v3.1 by 0.40 pp. The selection fact is about run inventory: llama-3.3-70b and gpt-5.4 are the *only* claim-verification judge models with five pre-replication runs in *each* field order — the ten verdict-first→checks-first pairs whose field order/time collinearity this replication exists to remove; deepseek-v3.1, the next deepest, has five and four, and `check_selection_claim.py` asserts this from the run inventory, counting distinct run labels rather than files. Selection nonetheless followed exploratory observation, so the randomised experiment establishes that the effect on these selected configurations reproduces under within-block randomisation; it does not estimate how common the effect is across models.

The same design on a configuration that was not selected. This rerun applies the exact design unchanged to gemma-4-26b, the third open-weight family of the core matrix, which was not selected the way the two Table 2 judge models were (above): it has a base run and one same-order replicate per field order, never five, and its fixed-order identity gap on claim verification ($+23.00$, Table 7) is the smallest of the three open-weight claim configurations. Both facts were stated in the frozen plan before the draw, so the configuration is unselected and, among the three, the conservative choice.

Nothing else moved: a fresh uniform draw from the same 252 allocations (seed 20260915, allocation 148 of the 252), condition templates *byte-identical* to the exact design's claim-verification templates, and its analysis on the new schedule and pins.

The primary endpoint replicates on the unselected configuration. $T_{\text{stable}} = +15.78$ pp — verdict-stable changed shares of 23.76% cross-order against 7.98% within — at $p_{\text{exact}} = 0.0079$, rank $1/252$, the observed allocation being the largest of all 252 (whose statistics take 93 distinct values here, fewer than the 126 the global mirror alone would give, so some allocations tie). Check-vector identity gap is $+22.59$ pp (90.32% within-order identity against 67.72% across), also at the floor. The magnitude sits between the two selected configurations' ($+23.00$ on llama-3.3-70b, $+7.83$ on gpt-5.4), as a third draw from the same family of configurations should; it is one observation, not a trend. Mean gold accuracy is flat here: 66.21% verdict-first against 66.33% checks-first. Selected-policy disagreement is not flat: it is lower verdict-first under both precedence readings (1.05/4.80% against 6.50/7.80% checks-first), as on llama-3.3-70b, whereas gpt-5.4's two readings split (Table 5), consistent with Appendix L's finding that the verdict-level measure is configuration-dependent. Per criterion the cross-minus-same disagreement runs $+2.86$ to $+8.81$ pp, largest on `relations_match_evidence` and `evidence_sufficient`.

One manipulation-check figure reads two ways: gemma-4-26b repeats the whole JSON object on 8 of the 996 shared instances, so exact compliance as a full sequence is 99.59% while the order the model served first — the property the design needs — complies on 100.00% of answered rows (the supplementary material, part E).

Boundaries. One judge model, one task, one batch of twenty runs, 996 shared instances and 8 ERROR rows dropped under the paper's policy. It tests one unselected configuration; the three configurations are not a sample of judge models, and the rerun is not pooled with the exact design.

Every estimate of the same intervention, by design. Section 3 says magnitudes move with the batch and prints one instance-level interval per configuration. That interval is a bootstrap over instances *conditional on the observed runs*: it does not carry batch or design variation, and on the replication judge model the two are not the same size. Table 6 collects every estimate of the verdict-block move the committed results files contain. On claim verification × gpt-5.4 there are seven, from +3.43 to +9.95 pp — a spread wider than that configuration's printed interval — while on llama-3.3-70b eight estimates spread 5.21 pp around a much larger effect. Read together with the exact tests, that is the paper's claim in one table: the sign and the separation reproduce; the number does not travel.

F EVERY p IN THIS PAPER, AND THE REFERENCE SET IT BELONGS TO

The bracket and the p in Table 2 are different tests: the bracket resamples instances with the runs fixed, and the exact p re-allocates field order with the instances fixed, so on the safety row an interval covers zero beside a p at the floor.

A randomisation p is a statement about a reference set, and this paper reports five that do not share one (the table: the supplementary material, part F): the **exact randomisation** p over the $\binom{10}{5} = 252$ allocations the design drew from (floor $2/252$; the nine ten-block designs); the **five-block** p over $\binom{10}{5}/2 = 126$ splits held exchangeable *across* blocks (floor $1/126$; its frozen plan's test, replaced below); p_{block} over the $2^5 = 32$ within-block relabellings under a stated exchangeability assumption (floor $2/32$; the supplementary material, part E); **McNemar's exact** test, the binomial tail on discordant instances, BH-adjusted across 14 configurations (Appendix H); and the **run-level** test over the 10 groupings of six runs into two conditions (floor $1/10$; Appendix G). Only the first is a property of an executed randomisation over an enumerated support; the others rest on a stated model, a different unit, or both. The five-block row is the test its frozen plan named; its 126 splits treat runs in different blocks as exchangeable, which the executed design does not support, and p_{block} replaces it.

The exact design: the support is the design's own. Its treatment is the within-block order, its constraint is exactly five AB blocks of ten, and its draw is uniform over the $\binom{10}{5} = 252$ allocations that satisfy that constraint — from a committed generator and a recorded seed. The support is therefore not a model of the design but the set the design drew from, and the schedule generator's `-verify` mode re-derives the realised allocation from the seed. The analysis walks all 252 and reports $p = \#\{o : |T(o)| \geq |T(\text{obs})|\}/252$. Relabelling the conditions in every block leaves the same/cross partition untouched, so $T(o) = T(\bar{o})$ and the observed allocation always ties with its global mirror. The two-sided floor is therefore $2/252 = 0.0079$. At most 126 of the values are distinct — in these three configurations 94, 91 and 42, because further ties exist. A number at that floor says the observed allocation is the most extreme of the 252, which is a statement that the observed field order partition and its global mirror are maximally separated over the 252 allocations, and not a statement about effect size. The statistic pools all run pairs and defines no per-block effect sign. The three configurations that reach it run from +23.00 to +0.51 pp, a spread of 22.49 pp. We do not quote the ratio of the extremes: it is unstable when the smaller magnitude is near zero, and no cross-task interaction is tested here.

The earlier design: the reference set is a stated assumption. Its frozen plan declared the ten runs of a contrast exchangeable, giving a floor of $1/126 = 0.0079$. That equals the ten-block design's $2/252$ and is not the same quantity. The design shares endpoint state within a pass, so exchangeability holds within a block only; permuting the label within each of B=5 blocks gives a two-sided floor of $2/2^B = 0.0625$, the p_{block} above, and the constrained draw's true support is

Table 6: Every estimate of the *same* intervention — the verdict-block move, verdict-first against checks-first — that the committed results files contain, grouped by configuration and labelled with the design that produced it (the designs of Appendix E's inventory). Rows marked ⋆ are block-randomised ten-block designs with an exact reference set; the *re-read* row is qwen3-next's fixed-order estimate as the sufficiency analysis recomputes it (Appendix J); the rest are same-batch descriptive contrasts (2+2: two runs per field order; V: a control batch's verdict-block condition). Not listed: conditions that move something other than the verdict block (Table 14 and the matched-wording table in the supplementary material, part M), and two sets computed in the appendix rather than stored in a results file — the fixed-order matrix's own T_{stable} triplets (Appendix G; of these only qwen3-next's label-balanced $+20.50$ is also in a results file, the re-read row) and the five-block design's re-scoring onto this endpoint, whose gpt-5.4 and safety-rating values ($+3.36$, $+0.42$) sit below this table's ranges. The two paraphrased safety rows coincide exactly, from different counts (Appendix K). Generated from the results files, not transcribed.

configuration	design / batch	T_{stable} (pp)	gap (pp)
claims × gemma-4-26b	unselected-judge rerun⋆	+15.78	+22.59
	paraphrased rubric batch	+15.27	+21.75
	spread	0.51 *pp*	
claims × gpt-5.4	decoder-enforced design⋆	+9.95	+23.95
	exact design⋆	+7.83	+21.96
	sufficiency batch, same-batch 2+2	+6.36	+21.37
	temperature batch, T=0	+4.53	+18.54
	wording-control batch, V condition	+4.33	+18.87
	perturbation batch, V condition	+3.59	+18.71
	temperature batch, T=0.7	+3.43	+16.22
	spread	6.52 *pp*	
claims × llama-3.3-70b	paraphrased rubric batch	+26.10	+35.93
	wording-control batch, V condition	+25.08	+35.06
	sufficiency batch, same-batch 2+2	+23.02	+33.47
	exact design⋆	+23.00	+33.13
	perturbation batch, V condition	+22.99	+33.71
	temperature batch, T=0	+22.37	+33.32
	temperature batch, T=0.7	+22.12	+31.88
	temperature design, T=0.7⋆	+20.89	+29.72
	spread	5.21 *pp*	
claims × qwen3-next-80b-instruct	wording-control batch, V condition	+21.59	+30.54
	sufficiency batch, same-batch 2+2	+21.18	+30.94
	fixed-order 2+2, re-read	+20.50	+29.80
	perturbation batch, V condition	+20.39	+30.41
	paraphrased rubric batch	+17.12	+28.42
	decoder-enforced design⋆	+9.47	+14.19
	spread	12.12 *pp*	
safety × gemma-4-26b	paraphrased rubric batch	+2.21	+5.64
safety × llama-3.3-70b	paraphrased rubric batch	+0.71	+21.00
	exact design⋆	+0.51	+19.24
	spread	0.20 *pp*	
safety × qwen3-next-80b-instruct	paraphrased rubric batch	+0.71	+21.00
sci-data × gpt-5.4	same-batch 2+2	+4.91	+9.81
sci-data × llama-3.3-70b	same-batch 2+2	+3.55	+17.91

not enumerable (the schedules are written literally, with no generator). All five configurations sit at that floor, with separation margins from $+11.01$ to $+29.93$ pp and point estimates unchanged by the correction; the plan's per-contrast instance-set rule governs the tables (n=989 and 990; ablation identity gap $+27.56$, not $+27.50$). The rest of the argument is in the supplementary material, part F.

G THE RUN AS THE UNIT: RUN-LEVEL TESTS, PAIRINGS AND LEVELS

Criterion-level companions to the headline number. Per-check agreement — a distinct quantity, and the one a criterion-level reader wants — moves 1.0 to 8.1 pp; *exactly one* of the six checks moves on 18.2–27.0% of claim-verification instances, two or more on 13.8–28.7%, against 20.1% and 1.6% on safety rating; and which criterion carries the movement depends on the configuration (in the primary configuration `evidence_sufficient` at $+18.32$ pp cross-order over same-order against $+1.76$ on `temporal_match_evidence`; on gpt-5.4 the largest is `relations_match_evidence` at $+9.58$; all six criteria, all three configurations: Table 5). Per-check agreement is the mean over checks of the per-check agreement rate; checks are correlated, so it and vector identity are conservative summaries of joint movement, not independence claims.

The run as the unit. Four configurations were run through an interleaved sweep three times each: all twelve same-order/cross-order pairs carry their configuration's sign on the fixed-policy difference, the separation margin between the least identical same-order pair and the most identical cross-order pair is positive in every configuration, and the run-unit exact test sits at its six-run floor ($p = 1/10 = 0.100$, the unique maximum of the ten splits) in all four — run-unit descriptive evidence, not a randomised test, because verdict-first always ran first within a sweep and temporal position stays collinear with the label. The pairing tables, the permutation comparators and Figure 1's instance read check by check are in the supplementary material, part G.

The unconditional headline, and why a share is not an order effect. Section 3 quotes the level only beside its same-order counterpart and draws its inference from the difference, because conditioning on an unchanged verdict selects on a post-treatment variable. The unconditional form over the same four fixed-order pairings is $\Pr(\text{verdict unchanged and check vector changed}) = 0.8$–$3.9\%$ on safety rating and 23.9–27.6% on claim verification; reweighting to the frame shares lowers the claim-verification figure to 19.8–24.2% over these three configurations (Appendix N gives 16.3–24.2% over five, gemini-3.1-pro and gpt-5.4 included). A share is not an order effect, though: two same-order runs of one field order already produce 4.8–8.6% of it, so the quantity that isolates field order is the difference (Table 2, and the probe below).

Table 7: Check-vector identity gap, scored symmetrically. *Within-order identity*: share of shared instances whose entire class-mapped check vector is identical between two runs of the *same* field order. *Cross-order identity*: mean over all $\text{VF}_i \times \text{CF}_j$ pairs, VF_i and CF_j being the i-th verdict-first and j-th checks-first run. *Identity gap* is mean(within) − mean(cross), paired instance bootstrap B=2,000; identity is not chance-corrected (Section 2). Two runs per field order except ‡safety × kimi, which has no checks-first replicate, so there the estimand is *not computable* and the two figures are descriptive; claim verification × kimi is computable and is a null (Appendix G).

		within-order identity		cross-order	identity gap
Task	Judge model	verdict-first	checks-first	identity	(symmetric)
Safety	llama-3.3	98.3	98.0	78.8	+19.36 [+15.36, +23.43]
Safety	gemma-4	99.7	99.4	95.3	+4.29 [+2.29, +6.57]
Safety	qwen3-next	96.3	98.0	77.2	+19.93 [+15.86, +24.07]
Claims	llama-3.3	92.4	91.8	59.9	+32.15 [+29.29, +34.90]
Claims	gemma-4	90.6	88.6	66.6	+23.00 [+20.47, +25.50]
Claims	qwen3-next	95.2	93.0	64.3	+29.80 [+27.21, +32.56]
Claims	gemini-3.1-pro	87.8	85.1	79.3	+7.18 [+5.51, +8.83]
Claims	gpt-5.4	86.4	73.0	64.5	+15.18 [+12.93, +17.46]
Claims	kimi-k2-th.	51.5	49.6	50.2	+0.33 [−1.40, +2.06]
Safety‡	kimi-k2-th.	80.0	—	78.9	not estimable

The four cross-order pairings. Two verdict-first runs (VF_1, VF_2) and two checks-first runs (CF_1, CF_2) per configuration, so the within-order terms are one pairing each and the cross-order term averages four (every pairing: the supplementary material, part G). The four cross-order pairings agree to within 1.2 pp in every open-weight configuration, and to within 3.3 pp on gpt-5.4, so which pairing is chosen does not carry the result. One configuration breaks the pattern: on claim verification gpt-5.4's within-checks-first identity is 73.0 against 86.4 within verdict-first (729 and 862 of 998 instances), a 13.3 pp gap, so for that judge model checks-first is the less stable field order as well as a differently-placed one. Its symmetric identity gap, $+15.18$, is still positive because both cross terms sit below both within terms, but the "similar within-order collision probabilities" reading is a claim about the six open-weight configurations, not about gpt-5.4. gemini-3.1-pro behaves like the open-weight configurations (87.8 vs 85.1, symmetric identity gap $+7.18$).

Four-pair averaged per-configuration figures. $\Pr(\text{verdict same},\ \text{vector changed})$ and the verdict-stable identity gap both average over all four $VF_i \times CF_j$ pairings; the min–max range across pairings never exceeds 2.5 pp, so no single pairing carries either quantity (the per-configuration table: the supplementary material, part G). On the three claim configurations the verdict-stable identity gap runs $+25.38$, $+16.75$ and $+23.53$ pp (llama-3.3, gemma-4, qwen3-next); on safety rating $+0.39$ to $+3.22$.

Verdict-stable identity gap, and the matched baseline. *The baseline here is symmetric*, $\frac{1}{2}[I(VF_1, VF_2)+I(CF_1, CF_2)]$, matching Section 2's estimand: a one-sided within-order term would be first order in a dispersion-only difference where the symmetric one is second (Appendix B), worth 1.3 pp on claim verification and 5.1 pp on gpt-5.4 here. Restricting to instances whose *verdict* is identical across field orders is a post-treatment selection with no causal reading; it answers the narrower descriptive question a consumer of the check fields cares about — when the verdict does not move, does the stated reasoning still move — with both terms recomputed on that same subset, like for like. Every figure here is the four-pairing average.

[‡]kimi-k2-thinking is *in* the table where the estimand is computable: claim verification has a checks-first replicate, and its symmetric estimand is a null, $+0.33$ $[-1.40, +2.06]$ (49.5 on the $VF_1 \times CF_1$ pairing alone against 50.2 averaged); safety rating has no checks-first replicate, so its two figures are one pairing, descriptive. The estimand is computable at this low within-order stability (51.5/49.6%): a zero expectation means the two field orders' output distributions coincide, not that either field order is stable (the construction that shows why: the supplementary material, part G). The claim-verification figure therefore shows no detectable excess cross-order identity gap under the present estimand. The two proprietary rows above it are not among the six Section 3 counts.

Mean per-check agreement moves 1.0 to 8.1 pp same-order to cross-order on these runs against vector-level changes of 4 to 32 pp, and the fixed-order T_{stable} triplets ($+21.89$ / $+15.36$ / $+20.50$ pp label-balanced; $+19.46$ / $+12.12$ / $+17.21$ reweighted to the frame) split by the agreed verdict's correctness as the randomised configurations do, over half of each under verdicts that are also *right* (the figures: the supplementary material, part G).

H GOLD ACCURACY AGAINST THE SECOND SAME-ORDER RUN

At the verdict level. Against the *second* same-order run, 6 of 14 configurations clear BH-FDR (q=0.05, McNemar's exact test) — a descriptive screen — with signs from $+13.43$ to -12.29 pp, so no "field order X is more accurate" reading survives. The largest gain is a coverage effect, and a near-zero net can hide much larger gross flows (the table in the supplementary material, part H).

Not every movement clears the correction. gpt-5.4 loses 3.71 pp $[-6.31, -1.00]$ on claim verification — where NOT-ENOUGH-INFO is itself a gold verdict class, an answer the task defines rather than a refusal — and 4.86 $[-8.00, -1.71]$ on safety rating, whose gold has no abstain class (Section 2); both clear BH-FDR (q=0.021, q=0.012), while gemini-3.1-pro-preview's claim-verification loss of 0.90 $[-2.10, +0.30]$ does *not* clear it. The movement is scoreable in both label geometries and still not uniformly positive.

Safety llama-3.3's $+13.43$ pp is a coverage effect, in the three quantities an abstention-capable judge should report separately. *Coverage*: verdict-first commits 244 of 350 instances, checks-first 335. *Common-subset accuracy*: on the 244 committed under both field orders every verdict is identical,

so both score 61.07% there. *Own-answered-set accuracy*: $149/244 = 61.07\%$ verdict-first against $196/335 = 58.51\%$ checks-first — *lower*, because the 91 new commitments dilute it. The total with abstention scored as error moves $42.57\% \rightarrow 56.00\%$, and the 91 instances llama declined verdict-first and committed checks-first score 51.65% against a 70.33% majority-vote baseline on those same instances — below a constant-majority guess. A three-class total is coverage times own-answered accuracy; here the whole gain is the coverage factor while the accuracy factor falls.

Gross flows, not only the net. The net is the small difference of two opposing per-instance flows, and for the verdict-level consequence of the field order the flows are the quantity: an instance the field order moves correct→wrong is a new error even when another moves back. A correctness flip requires the *verdict* to change, so everything here lives on the complement of the verdict-stable set Section 3's T_{stable} claims are about, and with verdict gold only (FEVEROUS carries no per-check gold) it cannot test whether per-criterion attribution goes wrong *while* the verdict stays correct — a stated boundary of the design. The gross-flow table in the supplementary material, part H decomposes every claim-verification configuration with a replicate of both field orders: on llama-3.3 about 10% of the panel flips correctness across a cross-order pair against about 2.7% between two same-order runs, gpt-5.4's flows run several times its baseline, and kimi-k2-thinking's sit at its same-order baseline, consistent with its null identity gap; 85.0 to 98.5% of cross-pair flips co-occur with a changed check vector — co-occurrence, not evidence that the check movement caused the flips.

Check movement under *correct* verdicts. The flips above live on verdict-changed instances. The complementary question is how much check movement hides under verdicts that agree *and are right*. The commensurable answer is the joint split Section 3 quotes: on the primary endpoint's own pairs and all-instances denominator, $T_{\text{stable}} = T_{\text{correct}} + T_{\text{wrong}}$ additively, where T_{correct} is the cross-minus-same excess probability that an instance keeps a *correct* verdict while its vector changes. The three randomised configurations read $+23.00 = +12.90 + 10.10$, $+7.83 = +6.08 + 1.75$ (gpt-5.4) and $+0.51 = -0.03 + 0.55$ (safety; parts rounded independently); the analysis recomputes the published T_{stable} values exactly as its internal check. A conditional companion describes the strata themselves: among within-block cross-order pairs whose verdicts agree and match gold, the vector changes on 28.13% of those instances against 5.17% for same-order pairs (+22.96 pp); where the agreed verdict is *wrong*, 40.94 against 5.79 (+35.15). gpt-5.4 shows +11.81/+13.12; the randomised safety configuration −0.06/+2.41. These conditional rates carry a verdict-agreeing denominator and a within-block pairing, so they are not commensurable with the joint figures above — post-hoc, conditioned on a post-treatment event, descriptive. The reading, scoped to these claim-verification configurations: a correct verdict does not guarantee a stable criterion profile — over half of T_{stable} sits under verdicts that are also right — while on safety rating the correct-verdict term is zero.

The contrast, and its inference. With verdict-first runs VF_1, VF_2 and one checks-first run CF, the contrast is CF against the *second* verdict-first run — a replicate-adjusted contrast, not a difference-in-differences, with no run-level variance component to estimate — tested by McNemar's exact test on the discordant instances, BH-adjusted across the 14 configurations, with a percentile instance bootstrap (B=20,000); thirteen of the fourteen $\text{VF}_2 \rightarrow$CF gaps are within 31 minutes. The per-configuration table is in the supplementary material, part H: eight percentile-bootstrap intervals exclude zero, seven exact McNemar p are below 0.05, and six survive the correction (claim verification × kimi-k2-thinking is the interval/test discrepancy, p=0.0535; BH additionally removes safety × gemma-4, q=0.078). Field order moves gold accuracy in *both* directions; neither comparator establishes rerun stability, so the screen stays descriptive.

I Criterion-level anatomy of the identity gap

Which criteria move, on the instances whose verdict does *not* move (llama-3.3, VF_1 against CF_1). Of 852 verdict-stable claim-verification instances, 275 change at least one check, 352 check-changes in all (the distribution of changed-check counts: the supplementary material, part I). On claim verification the movement is concentrated but not exclusive: `evidence_sufficient` accounts for 147 of the 352 check-changes (42%), followed by `relations_match_evidence` 66, `entities_match_evidence` 51 and `no_contradicting_evidence` 39. So the vector-identity result is not an artefact of one abstention-related field, but neither is it spread evenly:

a downstream consumer that reads only `evidence_sufficient` would see the largest single share of the movement and still miss more than half of it. Instances that change usually change *one* check (210 of the 275 that move at all), which is why per-check agreement moves far less than vector identity.

On safety rating the same anatomy is empty — 3 check-changes over 259 verdict-stable instances — consistent with the near-zero matched identity gap there.

Does the anatomy a practitioner reads change? Read as a failure profile — which criterion fails most under each field order — one of the six fixed-order claim-verification configurations flips its most-failing criterion outright (llama-3.3: `relations_match_evidence` verdict-first, `evidence_sufficient` checks-first, the mover shifting 19.61 pp cross-order against 0.20 same-order), one resolves an exact tie (qwen3-next), and four keep the six-criterion ordering at $\rho = 1.000$; no per-check gold says which headline is right (table and margins: the supplementary material, part I).

The same anatomy on the randomised runs, with the verdict split. The fixed-order table pools verdict-changed instances and does not touch the randomised data; both gaps close from the exact design's pinned runs alone. This is a post-hoc reading of runs collected for the exact design's frozen endpoints: Table 2's exact p attaches to those endpoints, not to the rankings here, and each decomposition below is a descriptive accounting of a net change, not a causal mediation share. The flip replicates under randomisation on the primary configuration (the table: the supplementary material, part I): all ten verdict-first runs read `relations_match_evidence` (margin 4.97 pp) and all ten checks-first runs read `evidence_sufficient` — by a thin 0.98 pp margin over `relations_match_evidence`. Decomposing each criterion's fail-rate difference over the ten within-block cross-order pairs, with *all* instances as the denominator: `evidence_sufficient` rises $+19.06$ pp, of which $+15.24$ comes from instances whose verdict agrees within the pair and $+3.82$ from instances whose verdict changed — the headline criterion's movement is mostly verdict-stable identity gap. The argmax rewrite itself is a joint effect: adding only the verdict-stable contributions to the verdict-first profile lifts `evidence_sufficient` to 38.86%, still 6.19 pp below `relations_match_evidence`; the flip completes through the verdict-changed instances (`relations` -3.34, `sufficient` $+3.82$). On gpt-5.4 nothing flips (margins 5.25/6.65, ten of ten runs each) and its largest mover is carried by verdict-*changed* instances ($+7.90$ of $+7.52$, the verdict-stable part -0.38). On the randomised safety configuration the flip is margin-thin (0.29 pp under verdict-first, nine of ten runs) and mostly verdict-changed ($+3.99$ of $+4.28$).

Read together: verdict-stable identity gap dominates the movement of the criterion that rewrites the primary configuration's headline, and the rewrite as such needs the verdict-changed instances too.

Two robustness reads on that headline (the supplementary material, part I): under an instance bootstrap the verdict-first headline holds in 100.0% of draws and the full flip in 72.2% — the thin 0.98 pp margin, quantified — and reweighted to the eligible frame's label shares *both* conditions' headlines move, so the audit headline is relative to the sample's composition as well as to the field order. gpt-5.4's no-flip reading is robust to both; the safety flip is fragile under the bootstrap. The full-rubric paraphrase replicates the flip with the same criterion pair on both llama configurations, decisively on claims and fragilely on safety (Appendix K; table in the supplementary material, part K).

J DISAMBIGUATING THE SUFFICIENCY CRITERION

`evidence_sufficient` supplies 42% of the criterion-change events on the primary configuration's verdict-stable instances (Appendix I), and its clause admits two readings that diverge on multi-part claims (the worked instance: Appendix G). This experiment tests whether the movement is the field orders resolving a latent ambiguity in the rubric's own wording rather than the judge responding to field order.

Design. Two single-clause variants of the claim-verification rubric each state one reading exclusively. The *decide* variant: *"true if the evidence shown is enough to DECIDE the claim — to support it or to refute it; false only if the evidence leaves the claim undecidable. Evidence that refutes part of the claim decides it, and is therefore sufficient."* The *cover* variant: *"true if the evidence shown covers EVERY part of what the claim asserts; false if any part of the claim is not covered by the*

evidence, even when another part is contradicted." Every other byte of the prompt is unchanged (the one-edit invariant is asserted per field order on all 999 instances; Appendix C). Each variant ran twice per field order on each of llama-3.3-70b, qwen3-next-80b-instruct and gpt-5.4 (2+2 per configuration, 24 runs, 2026-09-08), the eight runs per judge model interleaved in a palindrome over (variant, field order) so neither factor is collinear with time, each pass under its own recorded shuffle seed. A same-batch original-rubric control (2+2 per judge model) ran the same day, after the variants, so it is a same-day reference, not an interleaved one. Emitted key-order compliance is 100% in all 36 runs; the ERROR counts per run and the one control pass that was resumed to recover capacity errors are in the run manifests.

Comparators. The analysis reloads the randomised configurations and refuses to report unless it reproduces the published +23.00/+7.83; the same 2+2 statistic on all 45 two-block subsets of each gives the spread a four-run estimate has under the original rubric; qwen3-next's comparator is the pinned fixed-order 2+2 (the supplementary material, part J). All figures use the paper's class mapping and the joint all-instances denominator.

Table 8: Sufficiency disambiguation, same-day comparison on one common instance set per judge model: n is the number of ids all twelve same-day runs answered (the original-rubric control's 2+2, two runs per field order, and both variants'), so the magnitude comparisons cannot ride on per-condition sample composition. Each single-clause variant states one reading of `evidence_sufficient`; every other byte of the rubric, the instances and the decoding are unchanged (one-edit property machine-asserted). T_{correct} and T_{wrong} split T_{stable} by whether the agreed verdict is gold-correct (Appendix H); they are rounded independently, so a printed split can differ from its total by 0.01 pp. Per-condition detail and the earlier-batch comparators: the supplementary material, part J.

Judge model	n	Rubric	T_{stable}	T_{correct}	T_{wrong}
llama-3.3-70b	991	original rubric (same-batch control)	+23.01	+12.92	+10.09
		decide reading	+15.82	+8.53	+7.29
		cover reading	+14.25	+8.35	+5.90
qwen3-next	882	original rubric (same-batch control)	+21.03	+11.93	+9.10
		decide reading	+17.86	+10.01	+7.85
		cover reading	+19.64	+10.37	+9.27
gpt-5.4	997	original rubric (same-batch control)	+6.37	+5.49	+0.88
		decide reading	+8.12	+4.86	+3.26
		cover reading	+5.44	+4.19	+1.25

Reading. The movement survives disambiguation under *both* readings on all three judge models. Where the clause dominated — the primary configuration — the drop is large, and a same-batch original-rubric control (2+2, run the same day as the variants) supports reading it as the disambiguation's rather than as batch-to-batch change. The control reproduces the 08-22 base almost exactly (+23.02 against +23.00; cross/same components 28.29/5.28 against 28.28/5.28; suff. share 40.1% against 41.7%). On the day's common instance set (n=991, the ids all twelve same-day runs answered, Table 8) both variants sit far below the same-day base: +15.82 (decide) and +14.25 (cover) against +23.01 — below the base's own 2+2 spread of $[+21.71, +24.15]$ as well. On the common set the two stated readings come out 31–38% below the same-day original rubric — most of the movement remains — and the criterion's share of the change events falls from 40–42% to ∼16%. Restating the clause changes its wording and emphasis along with its ambiguity, so the drop is descriptive, not an identified effect of ambiguity alone. Where the clause did not dominate, little changes: on their common sets qwen3-next retains +17.86/+19.64 against its same-day base of +21.03 (n=882; own-intersection values, which differ by at most 0.4 pp, in the supplementary material, part J), and gpt-5.4's variants straddle its base of +6.37 (+8.12 decide, +5.44 cover; n=997) — no systematic reduction there. The two readings shift the criterion's marginal behaviour in opposite directions — decide returns `true` on 86.5% of llama's calls, cover returns `false` on 60.6% — exactly the two sides the worked instance of Appendix G lands on, yet they agree on the size of the surviving movement. So the finding is not an artefact of which reading a rubric author intends: stating either one removes the clause's ambiguity and leaves most of the verdict-stable movement, and its correct-verdict majority, in place.

What this does not establish. The 2+2 design supports no randomisation inference and post-dates the frozen plans, so it is a targeted post-hoc control; the same-batch 2+2 control anchors the magnitude comparisons, while comparisons against the earlier randomised batches (08-18 to 08-22) remain cross-batch.

K THE FULL-RUBRIC PARAPHRASE REPLICATION

Does the order effect depend on this paper's particular rule wording? Every instruction sentence of the claim-verification and safety rubrics — the preamble, the six criterion descriptions and the decision-criteria prose — was reworded with content held fixed, and the field order contrast rerun from scratch, one same-batch palindrome VF–CF–CF–VF per configuration (2026-09-10). Held byte-identical and machine-asserted: the check keys and their value vocabularies, the verdict enum, the decision rule, the response-format block, the input markers and placeholders, and the instances; semantic guards further assert that the sufficiency clause's two divergent readings both survive, that no precedence vocabulary is introduced (the paraphrase must not quietly disambiguate what the original leaves unstated), and that every value-literal a criterion mentions is preserved. Both paraphrased templates ship in the artifact (`prompts/`; digests in the supplementary material, part A); their semantic equivalence is a design claim. SSDV is not reworded: its rubric is the released benchmark verifier's, held byte-frozen throughout.

Reading (the table: the supplementary material, part K). The movement survives the rewording in every configuration: paraphrased identity gap lands within a few points of the same configuration's original-rubric value ($+35.93$ against $+32.15$; $+21.75$ against $+23.00$; $+28.42$ against $+29.80$; $+21.00$ against $+19.36$; $+5.64$ against $+4.29$; $+21.00$ against $+19.93$), every interval excluding zero. The two safety rows' $+21.00$, and their T_{stable} of $+0.71$, coincide exactly from different counts: with $n = 350$ both statistics move in steps of $1/1{,}400$, and both judge models land on 294 and 10 steps. The cross-task T_{stable} profile reproduces as well: large on claim verification ($+26.10$/$+15.27$/$+17.12$), near zero on safety rating ($+0.71$ $[-0.4, +2.1]$ and $+0.71$ $[-0.7, +2.3]$ on the two judge models with a large identity gap there, $+2.21$ on gemma-4) — the same boundary the original rubric shows. At the existence level, the order effect is not an artefact of either task's particular rule wording, and neither is its configuration-dependence; magnitudes remain configuration facts, and the original-vs-paraphrase comparison is descriptive (different batches, one rewording per task).

Does the audit-facing anatomy replicate too? Recomputed on the paraphrase configurations with the estimand unchanged, the llama-3.3-70b claim configuration's headline flip recurs under both rubrics, the llama safety flip recurs but stays fragile, the two no-flip configurations stay no-flip, and two configurations change pattern (a tie resolved into a no-flip; a strict flip into a tie) — transfer of headline patterns remains configuration-dependent (table: the supplementary material, part K).

L VERDICT-LEVEL EFFECTS AND RULE PRECEDENCE

The body reports what the field order does to the checks beneath an unchanged verdict. The field order also moves the verdicts themselves, and this appendix reports that part, measured as disagreement with a selected aggregation policy (measure (c) of Section 2): the published verdict contradicts the deterministic checks→verdict rule applied to the run's own emitted checks. On the rows where both taught clauses fire, either verdict obeys the prompt as written; that is what makes the selected-policy figures a joint measurement of judge and aggregation policy.

The same intervention moves the selected-policy measure in opposite directions across tasks. On scientific-data verification, moving the checks block ahead of the verdict collapses the rule-violation rate on llama-3.3-70b, llama-4-maverick and qwen3-next (llama-3.3 $10.22\% \rightarrow 0.08\%$) while gemma-4 and gpt-oss-120b barely move, and safety rating replicates the collapse ($6.57\% \rightarrow 1.14\%$), where llama-3.3's *abstentions* fall further still, $30.29\% \rightarrow 4.29\%$ (Appendix U). That is coverage, not judgement, and it is what this configuration's gold-accuracy gap measures: verdict-first commits on 244 of 350 instances against 335, and on the 244 both answer, every verdict is identical (Appendix H). On claim verification the same measure moves the *other* way (llama-3.3 $5.71\% \rightarrow 23.95\%$).

Table 9: Disagreement with each selected aggregation policy, and the precedence-free violation rate (%), under the readings of the taught decision criteria, on the task where the criteria conflict (claim verification) and on one where they never co-fire in the observed outputs (safety rating). VF / CF: verdict-first / checks-first. *amb.*: the share of instances where both the refuted clause and the not-enough-info clause fire, VF/CF; where it is zero the rate is precedence-free. Bold: the higher rate of each VF/CF pair under a policy.

		insuff.-first		hard-fail-first		**precedence-free**	
Task	Judge model	VF	CF	VF	CF	VF	CF
Claims	llama-3.3 (17.1/38.2% amb.)	5.71	**23.95**	12.02	**14.23**	0.30	0.00
Claims	gemma-4 (5.1/12.1%)	1.30	**6.41**	5.01	**8.52**	0.60	1.40
Claims	qwen3-next (10.0/21.7%)	3.81	**7.81**	7.01	**14.31**	0.40	0.20
Claims	gpt-5.4 (7.9/12.0% amb.)	3.00	**11.32**	**10.71**	0.70	2.90	0.00
Claims	deepseek-v3.1 (9.4/17.7% amb.)	0.60	**17.32**	**9.81**	1.00	0.50	0.30
Safety	llama-3.3 (0% amb.)	**6.57**	1.14	**6.57**	1.14	precedence-free	

Most of that movement is policy resolution, not rule-breaking. On claim verification the rubric leaves open which clause wins when a hard fail and not-enough-info co-fire, so its rule has two admissible readings: the selected policy (*insufficiency-first*) and the alternative (*hard-fail-first*). A verdict that breaks the rule under both is a *precedence-free* violation, the appropriate rule-following measure: it stays at or below 2.9% in *every* claim-verification condition of Table 9, against llama-3.3's selected-policy 23.95%. Nor does checks-first lower it uniformly: it rises on gemma-4-26b (Figure 3). Checks-first changes the *composition*, not which field order is safer, and under the alternative admissible policy 2 of the 5 judge models reverse (Table 9).

What the rubric teaches sets the sign. A counterbalanced 2×2 of field order $\times$ taught precedence moves the *verdict-level* order effect across zero on two of three judge models, every condition-level interval and interaction excluding zero (Appendix L); recomputed inside each explicit-precedence condition, the check-level estimands stay intact (Appendix L). The rest of this appendix is the supporting detail for these three results.

A co-occurrence. The claim-verification sample carries 333 gold not-enough-info instances in 999. Verdict-first, llama's evidence-sufficiency check fires 239 times; checks-first it fires 437 times — the early verdict co-occurs with the one check whose job is abstention firing far less often. The rubric is silent on which wins when both fire: a false evidence check implies REFUTED and a false sufficiency check NOT-ENOUGH-INFO, and nothing states the precedence. Separating instruction order from *enforced* emission order needs a decoder-enforced schema, which Appendix M.1 supplies on two judge models, and native schema support does not by itself identify the mechanism. The selected-policy disagreements that appear under checks-first are, judge model by judge model, of the corresponding type — the rule reading says not-enough-info, the emitted verdict is decisive (counts under *Mechanism tables* below).

The effect does not rest on the sample's label mix. The sample is balanced 333/333/333 against a frame of 3,891/3,420/491; conditioning on gold label and post-stratifying to the frame moves the order effect by at most 2.0 pp, downward on every judge model (Appendix N).

Policy resolution against rule-breaking, in full. The summary is stated above; the counts behind it are under *Mechanism tables* below, where on gpt-5.4 the *composition* of the selected-policy figure changes with the field order while the precedence-free count falls to zero. The two taught clauses never co-occur on safety rating (0 of 350 instances in every field order) or on scientific-data verification (0 of $1{,}223$, verified on the released rule), so nothing in the collapse results above depends on the precedence chosen; on claim verification they co-occur on 5.1–38.2% of instances, which is why that task needs the choice stated.

A reasoning-first control. A reasoning-first field order — free-text reasoning before the verdict, checks after — reproduces neither condition: on both tasks llama-3.3's rate sits between the verdict-first and checks-first values (the reasoning-first columns of Appendix U's tables). It is an exploratory control, not an isolation of token count from content, and one of its two judge models is only partially

treated (Appendix C). We note the design distinction to the delayed-structure ablation of Fan (2026) and the 2-Turn strategy of Lee et al. (2026a): both *lift* the schema constraint and reimpose it later; ours permutes field order *within* a held schema — the content contract never changes.

The three quantities, kept apart. **(a) Selected-policy disagreement** is what the *policy row* of Figure 3 plots and what a practitioner computes by default; **(b) precedence-free violation** contradicts *every* admissible reading and stays at or below 2.9% in every claim-verification condition (Table 9), so almost none of (a) is rule-breaking; **(c) the taught-precedence interaction**, -14.1 to -8.6 pp below, is the only field-order effect on a rule the prompt fixes.

The precedence 2×2 under matched and fixed scoring. Two panels of the same runs. **Matched scoring** scores each condition by the rule its own prompt taught — the counterbalanced 2×2 stated at the head of this appendix, four passes on a 4×4 Latin square, paired instance bootstrap B=2,000 (the panel: the supplementary material, part L). With insufficiency-first taught the order effect spans -0.6 to $+4.5$ pp; with hard-fail-first taught, checks-first *lowers* violations on all three by 6.8–9.6 pp; the interactions are -9.91, -8.56 and -14.11 pp (llama-3.3-70b, gpt-5.4, deepseek-v3.1), and every interval in the panel and all three interactions exclude zero, conditional on the frozen sample and the observed runs. **Fixed scoring** — insufficiency-first applied to every condition regardless — leaves the insufficiency-first conditions unchanged and turns the hard-fail conditions into $+18.89$, $+13.76$, $+21.02$ pp. Those are not rule-violation rates but the distance of a hard-fail-instructed condition from a policy it was never asked to follow, so the head of this appendix reports the matched panel (the supplementary material, part L).

At the pass level the interaction is negative in 12 of 12 pass-configurations, spanning -15.92 to -8.31 pp, each judge model's four passes within 3.4 pp of each other, so the sign does not depend on pooling (the table: the supplementary material, part L).

Identity gap and T_{stable} persist under an explicit precedence. The strongest verdict-stable evidence would be circular if it only arose where the taught criteria leave the precedence unstated. The 2×2's runs decide this directly: every condition taught an *explicit* precedence, and the same symmetric estimands recomputed inside each condition (Table 10) leave the effect intact. Under either taught rule, identity gap spans $+16.69$ to $+31.85$ pp and T_{stable} $+3.44$ to $+17.74$ pp, every interval excluding zero, and the two sequence families agree within 3.3 pp on identity gap — against $+32.15$ and $+15.18$ pp identity gap on the same judge models' fixed-order ambiguous-rubric configurations. Making the precedence explicit changes what the verdict-level reversal above does; it does not remove the check-level movement underneath a stable verdict. One reading of the T_{stable} column: on gpt-5.4 and deepseek-v3.1 the taught precedence roughly halves-or-doubles the verdict-stable term ($+7.65$/$+9.29$ insufficiency-first against $+3.99$/$+5.15$ hard-fail-first, fixed family) without driving it to zero, and under either explicit rule the criterion that moves most is the one the ambiguous-rubric configurations already identified (the supplementary material, part L).

Where the check vector is provably unchanged (exploratory). On the 650 of 999 claim-verification instances (65.1%) where gpt-5.4's full class-mapped check vector is identical across the two field orders, the selected-policy disagreement counts still move with the order (8 verdict-first against 21 checks-first under the insufficiency-first reading; 35 against 6 under hard-fail-first): evidence for an aggregation-channel component, not an identified channel, since the probe conditions on a post-treatment variable and inherits this appendix's precedence caveat (the supplementary material, part L).

Fixed-sequence replication of the precedence 2×2. An earlier three-pass sweep that executed the configurations in one fixed order gives the same picture as the counterbalanced design: interactions -10.01, -9.28 and -12.28 pp, every sign the same, the same two judge models crossing zero; dropping the one counterbalanced pass with 54 ERROR rows moves the interactions by at most 0.21 pp (the supplementary material, part L).

Mechanism tables. Disagreement typing on claim verification, checks-first: rule-says not-enough-info with a decisive emitted verdict accounts for all 239 of llama-3.3's selected-policy disagreements (precedence-free violation 0.00%), all 78 of qwen3-next's, and 57 of gemma-4's

Table 10: Check-vector identity gap and T_{stable} *inside* each explicit-precedence condition of the 2×2, claim verification. Same estimands as the fixed-order core configurations: identity gap is mean within-order vector identity minus mean cross-order identity; T_{stable} is the joint $\Pr(\text{verdict same} \wedge \text{vector changed})$, cross-order minus same-order; instance bootstrap (B=2,000), n=941–999 per configuration. The two sequence families are reported separately — three fixed-sequence passes and four Latin-square passes, so within-order pairs 6/12 and cross-order pairs 9/16 — and neither is the ten-block exact randomisation of Table 2, so the comparators are the same judge models' fixed-order identity gaps in Table 7 (llama-3.3-70b and gpt-5.4; deepseek-v3.1 has none), not Table 2's randomised figures.

Judge model	taught precedence	passes	gap (pp)	T_{stable} (pp)	n
llama-3.3-70b	insufficiency-first	fixed (3)	+31.85 [+29.17, +34.54]	+15.65 [+13.58, +17.85]	998
		Latin (4)	+31.43 [+28.81, +34.16]	+14.78 [+12.71, +17.02]	996
	hard-fail-first	fixed (3)	+27.26 [+24.70, +30.01]	+17.74 [+15.38, +20.17]	999
		Latin (4)	+27.06 [+24.59, +29.61]	+17.32 [+15.06, +19.71]	998
gpt-5.4	insufficiency-first	fixed (3)	+18.04 [+15.99, +20.17]	+7.65 [+5.94, +9.36]	998
		Latin (4)	+16.69 [+14.86, +18.50]	+8.06 [+6.41, +9.75]	999
	hard-fail-first	fixed (3)	+18.80 [+16.71, +20.86]	+3.99 [+2.12, +5.95]	998
		Latin (4)	+18.14 [+16.30, +20.07]	+3.61 [+1.87, +5.33]	998
deepseek-v3.1	insufficiency-first	fixed (3)	+21.39 [+19.27, +23.52]	+9.29 [+7.47, +11.02]	999
		Latin (4)	+18.12 [+16.51, +19.81]	+8.14 [+6.62, +9.76]	998
	hard-fail-first	fixed (3)	+22.99 [+20.89, +25.12]	+5.15 [+3.17, +7.15]	998
		Latin (4)	+21.42 [+19.58, +23.38]	+3.44 [+1.64, +5.29]	941

64. gpt-5.4's disagreement composition changes with the order (under the insufficiency-first reading; see Table 9): verdict-first 29 of 30 are rule-says-refute/model-abstains; checks-first 113 of 113 are rule-says-not-enough-info/model-refutes, with evidence-sufficiency firing order-stable at 326/325/341/339 across its four field orders. gemini-preview's claim-verification disagreements are the same single type in every field order (34/27/31/23) at ∼3%.

The one candidate pathway with a name in this literature, and why it does not fit across tasks. Appendix B is explicit that the estimand establishes existence without decomposing how, and no design here identifies a mechanism. One candidate is nonetheless specific enough to make predictions: under checks-first the verdict is generated conditioned on the checks, so the checks are the computation; under verdict-first they follow a settled verdict, which is the post-hoc rationalisation regime of Turpin et al. (2023) and Lanham et al. (2023). That story predicts two things, and both are measurable on the four MCJudgeBench design points without a new call.

It predicts that verdict-first checks are further from the human gold where the verdicts agree. They are not, at any size this design can see. On verdict-agreeing units, in the paper's listed-unit denominator, the checks-first-minus-verdict-first gold-accuracy net is +0.385, −0.169, +0.342 and +0.254 pp over the four configurations, *every* instance-bootstrap interval covering zero (widest [−0.657, +1.660]); unconditionally the nets are −0.088, +0.147, −0.333 and −0.098, leaning the other way in three of four. A rationalisation regime strong enough to move which criteria are flagged by the margins of Section 4 would have left an accuracy signature; there is none.

It predicts that verdict-first checks fit their own verdict better. The MCJudgeBench runs show the reverse. Applying the frozen aggregation rule to each run's own checks, compliance with the emitted rating is 95.39–97.40% under verdict-first against 99.18–100.00% under checks-first, a difference of +2.60 to +4.16 pp with every bootstrap interval excluding zero: the field order that generates the verdict first is the one whose checks more often contradict it, the opposite of a rationalisation account and consistent with the two-stage result (Appendix M.2).

On claim verification the same prediction holds. There the verdict-first checks are the coherent ones: the selected policy's disagreement with the emitted verdict is lower under verdict-first for all five judge models, and the two conflicting clauses fire together 1.5 to 2.4 times as often under checks-first, in all five (Table 9); only the rate at which both admissible readings are broken, at most 2.9%, runs the other way, on four of the five. So the obvious pathway fits one task and reverses on the other, and no single pathway accounts for both. These runs support the existence and size of the

effect, not its cause. The probe is post hoc; its estimator is validated by a planted effect it must find and a null it must not.

M FOUR FURTHER CONTROLS: DECODER-ENFORCED SCHEMA, TWO-STAGE PIPELINE, MATCHED WORDING EDIT, SAMPLING TEMPERATURE

Each control below follows the discipline of Appendix C.

M.1 THE EFFECT UNDER A DECODER-ENFORCED SCHEMA

The body studies schemas declared in the prompt and, until this appendix, called decoder-enforced structured output a different mechanism without data. The question is whether the verdict-stable check-vector effect survives when the declared key order is *enforced* by the decoder rather than merely declared. Before running anything we probed what each serving stack does with a request-level JSON schema (six calls, 2026-09-14), requiring a field the prompt never mentions and ordering the properties opposite to the prompt's order (the probe table: the supplementary material, part M). OpenAI strict mode enforces the schema's property order for gpt-5.4; Vertex MaaS enforces the schema for qwen3-next-80b-instruct but canonicalises the key order alphabetically; llama-3.3-70b has no enforcement available; gemma-4-26b's queue was full on both probes.

So the primary judge model cannot be given an enforced schema on the stack the paper uses, and no open-weight endpoint available to us enforces a *declared* order. Two configurations follow (under a plan frozen before the first call): **gpt-5.4, primary-enforced** — the declared order decoder-enforced in both conditions — and **qwen3-next, dissociation** — the decoder holds the *emitted* order fixed and alphabetical in both conditions while the *declared* order varies, separating by construction the two orders the manipulation check of Section 2 keeps apart. The design is the exact design of Section 3 (master seed 20260914), prompts byte-identical to its claim-verification prompts (template digests asserted), the only change a strict `response_format` whose property order is the condition's declared order and whose permitted check values are exactly those the prompt declares; the endpoints are the exact design's, and the comparator is the same judge model's declared-only configuration of Table 2, a different batch, never pooled.

Enforcing the declared order does not remove the effect (the table: the supplementary material, part M). On gpt-5.4, with the order decoder-enforced in both conditions, $T_{\text{stable}} = +9.95$ pp [$+8.1$, $+11.9$] at the exact floor, against $+7.83$ [$+6.1$, $+9.6$] in the declared-only configuration of Table 2, a different batch, and identity gap is $+23.95$ against $+21.96$. All 19,980 answered instances emitted their keys in the schema's order (no ERROR rows; snapshot `gpt-5.4-2026-03-05` throughout), so the conditions differ in nothing but the position the decoder itself enforces. Within-order identity is lower under enforcement, 83.86% against 87.62% in Table 2's configuration, so a strict schema is not a reproducibility device either. **The declared order moves the checks even when the emitted order cannot.** On qwen3-next the decoder writes the keys alphabetically in both conditions (all twenty runs comply with that order), so the conditions differ only in the order the prompt declares; T_{stable} is $+9.47$ pp [$+7.9$, $+11.0$] and identity gap $+14.19$, both at the floor, against $+29.80$ identity gap in the fixed-order, declared-only qwen3-next configuration of Table 7 (two runs per field order, different batch). Those two identity gap figures sit either side of a factor of two, which is not an enforcement effect: they come from different batches and different designs, and nothing here randomises enforcement against its absence. What the configuration establishes is the sign and the floor under enforcement, and a verdict-stable excess of nearly ten points.

What this configuration varies, precisely. Both conditions change the order the prompt declares *and* the property order of the request-level schema, because the schema is generated from the condition; the variant that would separate them — one fixed request schema, only the prompt's order moving — is not run. What the alphabetical canonicalisation does separate is reading from writing: the emitted order is identical in both conditions, so whatever moves the checks is not the position at which the model writes them. Read with the two-loci finding that 87.4% of the reversed-schema condition's excess sits on pairs where *both* outputs kept canonical order (Appendix T), the effect follows what the model reads rather than where it writes — with the caveat that the reading here is the prompt and the request schema together. On these two judge models decoder-enforced output does

not escape the effect; it carries it. The verdict-first run of the tenth block on gpt-5.4 was answered in two windows, 834 instances before a pause and 165 after it.

What this configuration does not say. It rules out writing position, not declared position. The three declared orders already frozen are not a position scan of one block (each contrast moves a second block as well), so we computed those contrasts and report no figures from them; the design that would settle it — `checks` at each of the five positions with the other keys in a fixed relative order, each position with its own same-order replicate — is specified in the plan ("Not run") and is not run.

M.2 The two-stage mitigation: checks first, verdict from the checks

The obvious mitigation takes the verdict out of the object that carries the checks: stage 1 is the single-call MCJudgeBench prompt with `rating` and `confidence` removed from the response format and one preamble sentence changed, and returns the checks alone; stage 2 turns the stage-1 checks and the numbered constraints into a verdict without seeing the candidate response. Per judge model (llama-3.3-70b and gpt-5.4), ten two-stage replicates were interleaved in one batch with five verdict-first and five checks-first single-call runs on MCJudgeBench, under its own frozen plan; the design, the pre-fixed endpoints and the table are in the supplementary material, part M. Descriptive: no field order is randomised against another here.

The two-stage pipeline's checks are not detectably more accurate or more reproducible than a single call's; what the split removes is the rule violation. Per-constraint accuracy, within-configuration identity and verdict accuracy differ between the pipeline and either single call by amounts whose intervals cover zero (table: the supplementary material, part M), with one exception on gpt-5.4: the single checks-first call is more accurate than the pipeline (-0.98 [-1.92, -0.10] pp). The identity interval against the verdict-first call covers zero there but allows the pipeline a gain of up to 6.15 pp. Across configurations the pipeline's check vector sits with the checks-first one — identical to it on 94.27% of instances but to the verdict-first one on 85.71%, the two single-call field orders agreeing on 86.51% (gpt-5.4: 81.53, 71.42 and 72.49%): taking the verdict out of the object that carries the checks is, for the checks, a checks-first field order, and the check vector it produces differs from the verdict-first one by as much as the checks-first one does. What the pipeline does change is rule compliance, lifted to 99.52 and 100.00% from 95.79 and 96.99% verdict-first; judged-`no` precision and recall move by at most 5.6 pp with no consistent direction. Same batch, descriptive; n=209 and 219 common instances (every instance all twenty runs answered); emitted key order complied in every run. What this comparison does not test is the pipeline's own order sensitivity: stage 1's field order was held fixed (`checks`, `issues`, `reasoning`), so whether moving a block inside stage 1 would move its checks is not measured here.

M.3 A token-matched control outside the schema

Table 14's indentation-only condition barely moves tokens, so "structural reorderings out-move it" says little about edit *magnitude*. Condition R rewords instruction text instead — the two preamble sentences, the process line and three section headings — with the response-format block, all six check definitions and all decision-criteria bullets byte-identical to the base. On the assembled template, with the `cl100k_base` tokenizer as a proxy for the models' own, the verdict-block move V is a token Levenshtein distance of 62 from the base, 172 characters and 31 tokens displaced; R is 43, 187 and 24 — character distance and tokens touched matched, token Levenshtein lower for R because a block move is charged twice, as a deletion and an insertion of the same span. Per judge model (llama-3.3-70b, qwen3-next-80b-instruct and gpt-5.4) six runs form one palindrome batch, base, R, V, V, R, base, so the two conditions are compared inside one batch against the same base pair, with the estimands of Appendix S and a paired instance bootstrap, under its own frozen plan.

A wording edit of matched size is not a null, and on gpt-5.4 it moves the verdict-stable checks more than the verdict block does. Against the base pair, condition R moves the class-mapped check vectors on all three judge models, every interval above zero (the table: the supplementary material, part M). On the two open-weight judge models the block move V out-moves the wording edit on both estimands (V minus R: $+24.33$ and $+15.31$ pp of identity gap, $+18.79$ and $+12.43$ of T_{stable}, all intervals excluding zero). On gpt-5.4 it out-moves it on identity gap only ($+3.45$ [$+0.83$,

+6.33]); on T_{stable} the difference is -5.05 [-7.66, -2.60] pp — the matched wording edit moves the verdict-stable checks *more* than the verdict block, whose own excess on this judge model is small ($+4.33$ here, $+3.59$ in the earlier same-batch control, $+7.83$ under randomisation). So an edit of the block move's size placed in instruction text outside the schema is a perturbation of the same order — smaller than the block move on the open-weight judge models, larger on gpt-5.4's verdict-stable endpoint — consistent with the verdict block being one locus among several (Table 14) and a reason not to read field order as a privileged lever (Sclar et al., 2024). Emitted key order complied in every condition; n=997, 983 and 999 common instances; one of qwen3-next's R runs was answered in two windows, 562 instances before a pause and 437 after it.

M.4 The contrast at a sampling temperature, same-batch 2+2

On llama-3.3-70b and gpt-5.4, one palindrome batch each of two verdict-first and two checks-first runs at T=0 and the same at T=0.7: the same-order baseline rises modestly ($+1.31$ [$+0.15$, $+2.48$] and $+1.00$ [-0.70, $+2.70$] pp), T_{stable} does not detectably move (-0.25 [-1.82, $+1.49$] and -1.10 [-3.18, $+0.95$]), and identity gap trims a little on gpt-5.4 (-2.33 [-4.68, -0.08]); the design, table and disclosures are in the supplementary material, part M, and the randomised version is next.

The same design at a sampling temperature. The same-order baseline is the denominator of every headline here — $+23.00$ pp is 28.28% against 5.28% — and at temperature 0 it measures backend nondeterminism alone; if a deployed judge's natural within-order baseline were 15 or 20%, the practical reading of this paper would change, and the 2+2 above is descriptive. The temperature design gives that question the treatment the headline gets: the primary configuration (FEVEROUS-999 × llama-3.3-70b) under the exact design of Section 3 (seed 20260916, allocation 46 of the 252) at T=0.7 in *both* conditions, both condition templates byte-identical to the exact design's claim-verification templates and every run header recording 0.7 as the temperature actually sent.

The verdict-stable excess survives at a sampling temperature. $T_{\text{stable}} = +20.89$ pp — 27.97% cross-order against 7.07% within — at $p_{\text{exact}} = 0.0079$, rank $1/252$, the design's two-sided floor (95 distinct allocation statistics). Check-vector identity gap is $+29.72$ pp (89.54% within-order identity against 59.82% across), also at the floor. This is the first of the three readings the frozen plan fixed in advance: positive at the floor, and within a few points of the $+23.00$ the same configuration gives at T=0.

The attenuation is mostly the baseline. Set the two runs of the same configuration side by side. The cross-order figures barely move: verdict-stable changed $28.28 \rightarrow 27.97\%$, cross-order identity $59.42 \rightarrow 59.82\%$. The within-order figures move, in the direction sampling predicts: verdict-stable changed $5.28 \rightarrow 7.07\%$, within-order identity $92.55 \rightarrow 89.54\%$. So the judge does disagree with itself more at T=0.7, by about 1.8 pp on the stable-changed share, which is most of the -2.11 that T_{stable} gives up (the cross-order share falls by 0.31); identity gap gives up -3.40, of which the within-order identity accounts for 3.01. What a field order change does to this judge is nearly the same at both temperatures; what changes most is how much the judge already did to itself.

Secondary endpoints (the supplementary material, part M): per-condition gold accuracy and rule violation reproduce the T=0 figures rather than moving with temperature, and the criterion carrying the movement is again `evidence_sufficient`.

Boundaries. One configuration, one judge model, one temperature, one batch of twenty runs; 974 shared instances and 26 ERROR rows dropped under the paper's policy. T=0.7 is one point on a range this paper does not sweep, and this design is never pooled with the exact design.

N Scope, identification and reweighting

What each design identifies, and what it does not.

Analysis	Unit	Supports	Does *not* support
Fixed-order matrix (Table 7, Fig. 3)	instances; observed runs	descriptive identity gap associated with field order, on the configurations run	temporal isolation — field order is collinear with position in time

Analysis (continued)	Unit	Supports	Does *not* support
Blocked replication (the supplementary material, part E)	runs within blocks	robustness of the effect size on selected configurations under within-block exchangeability	prevalence across models; a conventional $p < 0.05$
Rating-only ablation (the supplementary material, part E)	shared-condition contrast	a retained fraction with `confidence` position held fixed	a mediation share for `rating`
Precedence 2×2 (Appendix L)	passes; instances	effect modification by what the prompt teaches	a task effect separable from that task's rule wording
Safety-rating replication (Appendix E)	blocks	a boundary condition: identity gap replicates and the verdict-stable term attenuates sharply ($+23.00 \rightarrow +0.51$ pp, interval covering zero)	that the verdict-stable *magnitude* transfers to safety rating
Decoder-enforced replication (Appendix M.1)	blocks	the effect with the declared order enforced at the decoder (gpt-5.4), and with the emitted order held fixed by the decoder (qwen3-next)	decoder-enforced pipelines other than the two tested
Two-stage pipeline (Appendix M.2)	instances; observed runs	a same-batch comparison of a checks-then-verdict pipeline with single calls, scored against gold	a randomised field order contrast; that any configuration is more accurate
Reworded-instructions condition (Appendix M.3)	instances; observed runs	a magnitude reference for an edit of matched size outside the schema	randomisation inference; a decomposition of the block move into parts
Temperature contrast (Appendix M.4)	instances; observed runs	the baseline and the excess at T=0.7 against same-batch T=0, two judge models	other temperatures; run-level inference

Four bounds govern everything in the body. **(1)** Results generalise over model $\times$ task $\times$ prompt configuration, not model $\times$ task. The full-rubric paraphrase replication (Appendix K) bounds the wording side of that entanglement: under one complete semantic-preserving rewording per rubric, identity gap replicates in all six core configurations and the cross-task T_{stable} profile reproduces — so at the existence level the effect is not an artefact of either task's particular rule wording. Magnitudes remain configuration facts, and one rewording per task does not exhaust wording space (Appendix J separately disambiguates the single most ambiguous clause; the movement survives that too). **(2)** Every broad-matrix configuration carries one same-order rerun, so we describe rather than estimate run-to-run movement and make no run-level inference. Multiplicity across configurations is uncorrected everywhere *except* the replicate-adjusted contrast of Appendix H, where BH-FDR is applied — and that correction, like every interval here, is instance-level and conditional on the observed runs. **(3)** Under the ambiguous rubric the claim-verification consistency numbers measure disagreement with *our* aggregation policy, not rule violation: the precedence-free rate is near zero and the taught-precedence 2×2 is where a rule-following effect is claimed. **(4)** The intervention is prompt-schema serialization — serial position of the instruction, generation order, and context at generation time move together — not generation order in isolation.

Vector identity is strict and not chance-corrected, so it is interpretable only against the within-order terms in its own row of Table 7, never as an absolute reliability. The thinking contrast is one matched pair plus one counterexample configuration, and the claim-verification thinking pair is not reported at all: its requested checks-first ordering was not realised (53% emit the verdict first) and its conditions differ sharply in attrition, so it fails as a field-order contrast on two independent grounds.

The claim-verification sample over-represents not-enough-info by design (333/999 against 491/7,802 in the eligible frame). Scientific-data verification configurations come from the benchmark's own harness — the same template up to the response-schema line order, a different runner.

Reasoning-first replicates exist only on claim verification, for llama-3.3-70b, gpt-5.4 and gemma-4-26b.

The four patterns, per configuration. Four post-hoc labels recur: **excuse-type** (violations collapse checks-first because the fired checks license the verdict the rule demands), **refusal-type** (the rate is order-insensitive), **silent-accuracy** (violations flat while accuracy moves) and **reading-shift** (the claim-verification pattern of Appendix L). The per-configuration roster is in the supplementary material, part N and Appendix U gives the cases; they are labels, not a typology, and cannot support a claim that vendor or tier structure is absent.

Label-conditional order effects. Conditioned on gold label, the order effect (checks-first minus verdict-first, insufficiency-first scoring) is largest on *refutes* and smallest on *supports* on every judge model, and post-stratifying to the frame shares $3{,}891/3{,}420/491$ lowers it by at most 2.0 pp — $+16.93$, $+8.14$ and $+14.77$ pp for llama-3.3, gpt-5.4 and deepseek-v3.1 against raw paired values of $+18.15$, $+8.32$ and $+16.72$. The per-label table, the same stratification of the verdict-stable identity gap under its two admissible reweightings (which differ by at most 0.06 pp), and the frame recount are in the supplementary material, part N.

The same stratification for the identity gap estimand. Section 3 reports identity gap on the balanced sample. Conditioned on gold label and then post-stratified with the same frame shares:

Judge model	supports	refutes	not-enough-info	balanced	post-strat.
llama-3.3-70b	$+14.71$	$+41.72$	$+40.06$	$+32.15$	$+28.15$
gemma-4-26b	$+14.01$	$+24.40$	$+30.56$	$+23.00$	$+19.60$
qwen3-next-80b	$+14.12$	$+36.94$	$+38.29$	$+29.80$	$+25.64$
gemini-3.1-pro	$+4.95$	$+8.26$	$+8.33$	$+7.18$	$+6.62$
gpt-5.4	$+8.03$	$+20.03$	$+17.49$	$+15.18$	$+13.89$

Identity gap is smallest on *supports* in every configuration (largest on *refutes* for llama-3.3-70b and gpt-5.4, on *not-enough-info* for the other three), so the frame's over-weighting of supports lowers every estimate — by 0.56 to 4.16 pp, leaving $+6.62$ to $+28.15$. The balanced column is what Table 7 prints. The verdict-stable share falls on four of the five (gpt-5.4's rises from 16.2 to 16.4%), from 16.2–27.6% balanced to 16.3–24.2% post-stratified, and it too is concentrated off *supports*: for llama-3.3 the three strata read 8.6, 40.7 and 33.6%. So "roughly a quarter of all instances" in the label-balanced sample is a statement about the balanced diagnostic sample; on the frame it is closer to a fifth.

O Scientific-data configurations: identity gap, and attribution against human-located errors

Two additions dated 2026-09-10, both on the scientific-data verification task, closing two boundaries at once: the identity gap estimand had no scientific-data configuration (no checks-first replicate existed), and no analysis connected check movement to independently located errors.

Same-batch $2{+}2$. Four runs per judge model (llama-3.3-70b and gpt-5.4) in one batch each, palindrome-interleaved (VF–CF–CF–VF, recorded seeds), so no pair crosses batches. The prompts are byte-identical to the pinned 2026-08-19 runs (digests frozen 2026-09-04), the two field orders differ only by the verdict-block/checks-block swap, and the instance set is the pinned runs' 1,223 ids (Appendix C).

Reading. The estimand replicates on the third task: identity gap $+17.91$ pp $[+15.8, +20.0]$ on llama-3.3-70b and $+9.81$ $[+8.1, +11.4]$ on gpt-5.4, both intervals excluding zero, against within-order identities of 86.5–93.8%. The verdict-stable term is present and smaller — $+3.55$ $[+2.1, +4.9]$ and $+4.91$ $[+3.5, +6.2]$ — sitting between claim verification's $+23.00$ and safety rating's $+0.51$. One candidate reading, which the design does not identify: this task publishes a deterministic checks-to-verdict rule, so a moved vector more often carries the verdict with it, leaving less movement underneath a stable verdict. With these two configurations the identity gap estimand is computable on eleven configurations (Table 7's nine and this table's two; identity gap recomputed inside

Table 11: Scientific-data verification, same-batch 2+2 contrasts (2026-09-10; two runs per condition, palindrome-interleaved; n = instances all four runs answered of 1,223). Upper block: the verdict-block move. Lower block: the eleven check keys emitted in reverse order inside the checks object (K), against the same base; the reversal is verified on 100% of parsed rows in every K run. Same estimands and class mapping as Table 7; paired instance bootstrap (B=2,000; instance-level, conditional on the observed runs). The two blocks ran one batch apart, so a comparison between them is descriptive.

Judge model	n	within	cross	gap (pp)	T_{stable} (pp)
verdict-block move, VF–CF–CF–VF (within: VF / CF)					
llama-3.3-70b	1196	92.5 / 93.8	75.2	+17.91 [+15.8, +20.0]	+3.55 [+2.1, +4.9]
gpt-5.4	1223	86.5 / 90.5	78.7	+9.81 [+8.1, +11.4]	+4.91 [+3.5, +6.2]
key reversal inside the checks object, VF–K–K–VF (within: base / K)					
llama-3.3-70b	1196	92.6 / 92.4	77.9	+14.63 [+12.8, +16.6]	+12.98 [+11.3, +14.9]
gpt-5.4	1215	87.2 / 90.7	77.9	+11.05 [+9.5, +12.8]	+9.90 [+8.4, +11.5]

the precedence conditions, e.g. deepseek-v3.1's, belongs to that separate design and is not in this count).

The key reversal replicates here. A second same-batch palindrome per judge model (VF–K–K–VF, 2026-09-10; the eleven check keys emitted in reverse order inside the checks object, single-edit invariants asserted before any call) puts the structural-reorder control on this task too (Table 11, lower block): identity gap and verdict-stable term both have intervals excluding zero on both judge models, and against this task's verdict-block move the reversal's verdict-stable term is the larger on both — a descriptive cross-batch comparison.

Attribution against human-located errors. The benchmark's second expert pass corrected individual fields, and those corrections are in the released instances (a field counts as corrected when its second-pass value differs from the original input field). This gives an attribution reference that required no new labels: 529 of 1,223 instances carry at least one field correction, 450 exactly one, and 113 of those also carry an Incorrect second-pass verdict — the instance is wrong and the locating field is named. Their fields: substance 107, value 4, material 2, property 0 (all 333 property-corrected singles carry Correct/Unclear verdicts — canonicalisation fixes, kept as a separate stratum). The scoreable stratum is therefore the 107 substance-corrected Incorrect instances, mapped to `formula_in_source` — the check whose name names the field; a variant widens to {`formula_in_source`, `notation_clean`, `element_symbols_valid`}. The mapping is a modeling decision, and per-field claims are known to be sensitive to it.

Three readings, in scope order. First, field-name mappings cannot say whether the emitted checks agree with the human's location: the level and the sign of the field order difference both change with the mapping. Second, the field order moves the checks tied to the corrected field on the instances where the ground for an attribution is strongest, under both mappings, with intervals excluding zero — and on gpt-5.4 a +7.94 pp verdict-stable component, on the 107 instances of that stratum: an auditor reading the same verdicts would see a different check-level assessment. Third, none of this scores whether a moved check went from right to wrong: per-check gold does not exist here (the field corrections locate the instance's error; they do not adjudicate every check), which remains the stated boundary of Appendix H.

P A GOLD-SCORED AUDIT REPLICATION ON A PUBLIC BENCHMARK

Every audit-facing analysis so far scores agreement between runs; none can say whose checks are *right*, because this paper's three tasks carry no per-check gold (the boundary Section 4 states). MC-JudgeBench (Lee et al., 2026b) does: 219 instruction-following instances in its expanded public release (the original 141-instance paper subset plus 78 added instances, reported separately below), each with 2–15 human-labelled constraints (`yes`/`no`/`partial`; 1,022 in all) for one fixed candidate response. We reran the field order contrast on it with the discipline of Section 3. The dataset snapshot is digest-frozen and gold enters the scorer only (no rendered prompt contains a label). Fifteen fixed constraint slots keep the response-format block identical across instances, so the intervention

stays exactly the verdict-block move (line-multiset equality of the two templates and byte-identity of the input head asserted for all 219 instances). A frozen aggregation rule (any listed constraint `no` → FAIL; else any `partial` → PARTIAL; else PASS) derives the gold verdict *from the gold labels*. The model verdict is not recomputed from the run's checks: each run emits its own `rating` in the same PASS/PARTIAL/FAIL vocabulary, and that is what every endpoint below compares against the gold verdict. Recomputing it from the checks is a separate sensitivity, reported later in this appendix. The design is the exact design of Section 3 (Appendix E), ten blocks per configuration, on llama-3.3-70b (primary) and gpt-5.4 (pre-specified replication); endpoints frozen in this design's own plan before the first call; 8,760 scoring calls at $T{=}0$; templates and digests in Appendix A; emitted key-order compliance is 100% in all forty runs. A `not_applicable` return on a listed constraint scores incorrect — a scoring rule of this experiment, distinct from Section 2's class mapping — at $1.46\%/0.95\%$ of listed units on llama's two field orders and 0% on gpt-5.4 (Table 12). A two-pass mechanical audit of the gold flagged six candidates, all resolved in the data's favour; label error would still affect the correctness metrics and their differences, though not repair-list divergence, which uses no gold (the supplementary material, part P).

Table 12: The gold-scored audit experiment on MCJudgeBench. Top, the three migration endpoints, unit-weighted over (instance, listed constraint) pairs (llama: $n{=}219$ instances, 1022 units; gpt-5.4: $n{=}219$, 1022); each entry is mean(cross-order pairs) − mean(same-order pairs) in pp, with a shared-resample instance bootstrap ($B{=}2{,}000$) and the exact two-sided randomisation p over all 252 allocations (floor $2/252 = 0.0079$); joint full-sample rates, no row conditions on a subset. *Verdict-stable correctness migration*: the two runs agree on the verdict while a constraint's gold-scored correctness flips. *Repair-list divergence* is weighted differently — the within-instance share of constraints whose judged-`no` flag differs, averaged over instances — and uses no gold. A `not_applicable` return on a listed constraint scores incorrect (its rate is the last column). Middle: per-order means over each field order's ten runs, against gold, with judged-`no` precision and gold-`no` recall. Bottom: cross-pair migration counts by direction over the 100 cross-order pairs, for both events, from the verdict-first run to the checks-first run; the full-event net equals the accuracy difference by identity (both checks-first minus verdict-first; asserted at compute time).

	llama-3.3-70b (primary)		gpt-5.4 (replication)	
endpoint, cross − same (pp)	estimate [95% CI]	p_{exact}	estimate [95% CI]	p_{exact}
verdict-stable correctness migration (**primary**)	$+1.10$ [$+0.14$, $+2.45$]	0.0079 (rank 1/252)	$+0.78$ [$+0.28$, $+1.33$]	0.0079 (rank 1/252)
correctness migration, unconditional	$+2.50$ [$+1.34$, $+3.97$]	0.0079 (rank 1/252)	$+2.24$ [$+1.55$, $+2.98$]	0.0079 (rank 1/252)
verdict-stable and verdict-correct	$+1.23$ [$+0.27$, $+2.57$]	0.0079 (rank 1/252)	$+0.76$ [$+0.34$, $+1.26$]	0.0079 (rank 1/252)
repair-list divergence	$+2.48$ [$+1.34$, $+3.94$]	0.0079 (rank 1/252)	$+2.03$ [$+1.25$, $+2.94$]	0.0079 (rank 1/252)

Judge model	field order	constraint acc. %	verdict acc. %	judged-`no` prec. %	gold-`no` recall %	n/a on listed %
llama-3.3-70b	verdict-first	78.85	68.77	73.38	57.86	1.46
llama-3.3-70b	checks-first	78.76	69.13	73.83	57.56	0.95
gpt-5.4	verdict-first	75.84	70.05	65.49	70.70	0.00
gpt-5.4	checks-first	75.99	69.54	70.43	68.71	0.00

Judge model	migration event	correct→incorrect	incorrect→correct	net, pp
llama-3.3-70b	all migrations	1672	1582	-0.088 (= accuracy difference)
	verdict-agreeing only	574	967	$+0.385$
gpt-5.4	all migrations	2561	2711	$+0.147$ (= accuracy difference)
	verdict-agreeing only	1686	1513	-0.169

Reading. All four pre-specified endpoints land at the exact floor in both configurations ($p = 0.0079$, rank 1/252; Table 12). The absolute rates stay in single digits, and the comparison that carries them is the same-order baseline: on llama-3.3-70b the cross-order rates are 3.18%, 1.49% and 3.06% against same-order 0.69, 0.27 and 0.59 on the three endpoints other than the primary (4.6 to 5.5 times), and gpt-5.4 replicates the same three at 1.5–$2.1\times$ its own higher run-to-run baseline, with tighter intervals. Both release subsets carry positive excess on every endpoint in both configurations (paper $+0.70/+0.39$ pp on the primary endpoint, augmentation $+1.80/+1.47$). Meanwhile the observed difference in mean constraint accuracy does not exceed 0.15 pp and verdict accuracy differs by at most 0.6 pp — near-identical accuracy *reports* over individual judgments that are unstable

across field orders, which is this paper's thesis with gold attached. The migrations run in both directions, and the direction depends on the event: the full-sample net equals the accuracy difference by identity, and the verdict-agreeing net can carry the opposite sign (counts in Table 12; all four nets in Appendix L), so neither event supports a "more accurate field order" reading. **What the migrating slots contain.** A slot can miss gold by a wrong substantive label or by a `not_applicable` return on a listed constraint. Split additively on the same pairs and denominator, llama-3.3-70b's $+1.10$ pp is $+0.575$ from slots where both runs returned a substantive label and $+0.526$ from slots where at least one returned `not_applicable` (the parts are rounded independently); gpt-5.4 returns none, so all $+0.779$ of its excess is substantive; in the no-filler control the excess is entirely substantive on both judge models ($+0.882$ and $+1.337$). The substantive component is positive and rank $1/252$ in all four design points, which is the form of the claim this paper relies on (the accounting: the supplementary material, part P).

How concentrated the endpoint is over instances. A rank-$1/252$ excess spread over 219 instances and the same excess carried by a handful are different pictures of one number, so each configuration's endpoint was recomputed leaving one instance out at a time. llama-3.3-70b's fixed-slot configuration is the concentrated one: the largest single contributor, `mcjb_0081`, accounts for 44.4% of its $+1.10$ pp, and the endpoint recomputed on the remaining 218 instances is $+0.616$ pp; the next two contribute 26.7% and 14.4%. The other three configurations are flatter — the largest contributor is 12.6% on gpt-5.4's fixed-slot configuration, 26.6% and 11.9% on the two no-filler configurations. Every leave-one-out and leave-three-out recomputation stays at rank $1/252$, but the exact test is saturated there, so rank says nothing about magnitude. The magnitudes, with instance-bootstrap intervals in the same frame as Table 3's (B=2,000, shared seed; the full-panel interval reproduces the frozen one exactly), are: llama-3.3-70b fixed-slot $+1.10$ [$+0.14$, $+2.45$] $\rightarrow$ $+0.616$ [$+0.03$, $+1.44$] without `mcjb_0081` $\rightarrow$ $+0.162$ [-0.12, $+0.52$] without the three largest (216 instances); gpt-5.4 fixed-slot $+0.78$ $\rightarrow$ $+0.68$ $\rightarrow$ $+0.52$ [$+0.12$, $+0.96$]; no-filler llama-3.3-70b $+0.88$ $\rightarrow$ $+0.65$ $\rightarrow$ $+0.46$ [$+0.13$, $+0.84$]; no-filler gpt-5.4 $+1.34$ $\rightarrow$ $+1.19$ $\rightarrow$ $+0.99$ [$+0.48$, $+1.54$]. So the primary configuration's *magnitude* rests on three instances — without them its interval covers zero — while the existence claim rests on four design points, three of which keep intervals above zero after the same removal. This analysis is post hoc, computed after the concentration was seen. The three share no structure the benchmark shows, and one of them is also the largest contributor in the no-filler llama-3.3-70b configuration, so part of the concentration is instance-specific rather than design-specific (the supplementary material, part P).

Four post-hoc analyses, in the supplementary material, part P: no gold-free signal picks the carrying instances out in advance, so no screen is offered; the effect does not scale with the declared schema size; recomputing the verdict from each run's own checks leaves the endpoint at rank $1/252$ and shows the emitted verdict disagreeing with its own checks far more often under verdict-first; and widening the repair policy to `no` or `partial` leaves M3 at rank $1/252$, and unit-weighting it leaves M3 positive under both policies on both judge models. The `mcjb_0081` instance carrying 44.4% of the primary endpoint — six violated constraints listed under one field order, one under the other, the verdict FAIL and correct under both — is worked there.

One repair-quality margin does move with field order, in both directions at once: gpt-5.4's judged-`no` precision is 65.49% verdict-first against 70.43% checks-first while its recall moves the other way (70.70% against 68.71%) — a concrete audit trade-off, not an improvement; llama's are stable near 73.5%/57.7%. Descriptive, ten runs per field order. Boundaries: one dataset, two judge models, absolute rates in single digits; the aggregation rule is ours (frozen, stated above), not the benchmark's. On the gold itself: three annotators labelled a shared 10% of each split of the 141 instances the benchmark paper evaluates (Fleiss' $\kappa = 0.8768$, Lee et al., 2026b), a subset the release does not mark; that is a construction-time check on the benchmark, not the blind re-review of the specific constraints that migrate here, which nobody has run (the supplementary material, part P). One control is named and not run: the task preamble states an order in prose ("…*then* give an overall verdict"), byte-identical in both conditions so it cannot confound the contrast; whether it *moderates* the effect would need a neutrally worded preamble as a third condition, and this paper makes no new calls.

The no-filler control. The fixed fifteen-slot schema leaves $\sim$69% of slots as `not_applicable` filler on the average instance. Rerunning the full design with each instance listing exactly its own k slots — everything else byte-identical or single-difference-asserted, under its own frozen plan — leaves every conclusion in place: all four endpoints stay at the exact floor on both judge

models, at magnitudes of the same order — smaller on all four of llama's, larger on three of gpt-5.4's four (Table 13). The two experiments ran in different batches, so that magnitude comparison is descriptive. The observed mean-accuracy difference stays small (78.98 against 78.65% llama; 76.39 against 76.29% gpt-5.4; the larger difference is 0.333 pp unrounded, inside the body's 0.34), `not_applicable` returns on listed constraints fall below 0.9% in every configuration of this control, and both release subsets keep positive primary-endpoint excess. The fixed-slot format is therefore not what drives the migrations; the filler-vs-no-filler comparison itself is descriptive (same instances and orders, different batches).

Table 13: The no-filler control: each instance's response format lists exactly its own k constraint slots (mean 4.67 of the fixed design's 15), same schedule, allocations, instance orders, endpoints and estimand as the fixed-slot design. Entries are cross-order minus same-order in pp; every no-filler entry lands at rank 1/252 ($p = 0.0079$). Fixed-slot values repeated for comparison (their intervals: Table 12).

	llama-3.3-70b		gpt-5.4	
endpoint	fixed-slot	no-filler [95% CI]	fixed-slot	no-filler [95% CI]
verdict-stable migration (**primary**)	+1.10	+0.88 [+0.32, +1.58]	+0.78	+1.34 [+0.69, +1.99]
correctness migration, unconditional	+2.50	+2.29 [+1.43, +3.23]	+2.24	+2.93 [+2.04, +3.93]
verdict-stable and verdict-correct	+1.23	+0.75 [+0.22, +1.42]	+0.76	+1.01 [+0.47, +1.57]
repair-list divergence (gold-free)	+2.48	+2.04 [+1.14, +3.14]	+2.03	+1.83 [+1.04, +2.71]

One migration, worked through. Section 4's `mcjb_0162` is worked in the supplementary material, part P, so a reader can check the label: the verdict is FAIL and correct under both field orders, carried by a mechanically checkable length constraint; the repair lists differ on the constraint that asks about factual correspondence; and the hundred cross-order pairs that show the split are a 10×10 product of the same twenty runs — one disagreement observed under every pairing, not a hundred independent ones.

The same disagreement in review units. What the migrations cost an auditor depends on which constraints each field order puts on its repair list and whether those are real. Take the constraints a run flags `no` as its list, charge one unit of review per listed constraint, and read the human gold label as what that review would find. The table in the supplementary material, part P partitions every pair on that proxy, against the same partition over two runs of one field order.

At two calls per instance, switching can recover complementary errors, and the benefit depends on the starting configuration. Across the 219-instance, 1,022-constraint sample, a checks-first second call after a verdict-first one adds 8.6 real errors on llama-3.3-70b against 1.5 from a repeat, and 13.7 review items against 2.6. On gpt-5.4 it is 3.6 against 3.7 and 7.0 against 10.0 — a trade-off, not a rule (the supplementary material, part P, unconditioned on the verdict).

Mean list sizes are near-equal on llama-3.3-70b and not on gpt-5.4, and near-equal *mean* list sizes are not matched review cost: the proxy charges every constraint one unit and nothing here selects a fixed budget. What the table shows is that each field order lists real errors the other does not, symmetrically on llama-3.3-70b and asymmetrically on gpt-5.4, where verdict-first uniquely lists both more real errors and more constraints gold calls satisfied; a constraint gold calls `partial` gets its own column (8.5 of the 20.3 a two-way real-or-wasted split would charge gpt-5.4's verdict-first list as waste). **With the verdict held fixed** the divergence survives, at the figures Section 4 quotes — joint-event counts over a fixed benchmark, not a conservative estimate on a common subset. Not measured: review time, severity, benefit under a fixed budget, or whether the constraints only one field order catches are clear mistakes or boundary calls, which is the blind re-review this paper cannot run (the design that would settle it: the supplementary material, part P).

Q SYSTEMATIC POSITIONING

The body cites the nearest work where each result needs it. This section states the positioning once, in one place (the table in the supplementary material, part Q), because the axis that separates this paper from its neighbours is easy to lose in per-result citations: what is *varied*, what is *measured*, and at what *unit*. Characterisations are at the level each paper's own framing supports.

What the nearest neighbour already establishes. One row of that table deserves more than a configuration. Lee et al. (2026b)'s Table 6 reports the direction of gold-correctness change under their perturbations, and it is "often relatively balanced" across models — Claude Sonnet 4.6 with reasoning off splits $50.00/50.00$ between correct-to-incorrect and incorrect-to-correct on both the prompt side and the response side. That an aggregate accuracy can hold while per-constraint correctness migrates underneath it is therefore established in that paper; Section 4 measures it at a different locus. Three things are not established there. The locus: their prompt-side conditions reorder the constraint list or the prompt's sections, or reformat the constraints, with the output schema held fixed (their Appendix B.4). The conditioning: their judges return a label per constraint and no verdict, so their $R_{\Delta\text{corr}}$ is unconditional over constraints, where this paper's verdict-stable endpoints (T_{stable} and the gold design's primary endpoint) count only units whose verdict the two runs agree on, and the strictest only those where it is also gold-correct. And the comparator: their intrinsic rate is $K{=}5$ repeats at temperature 1.0 while their procedural rate is computed at temperature 0.0, so the procedural figure is not read against a same-condition replicate, whereas every excess reported here is a difference between two quantities taken in one batch under one decoding.

The last column says one thing in every row. The neighbouring work varies something and measures whether the *decision* moves, or whether free text does; here the decision is held fixed — identical across field orders, and in the strictest endpoints gold-correct too — and what is measured is the typed per-criterion check vector underneath it, read against the judge's own same-order rerun rather than against zero. Neither half is available from the neighbours: a level is not an order effect, and a moved verdict is not a moved audit trail.

Comparing the two loci is a positioning question that neither paper's results settle, so it was run as one experiment — both reversals on one judge model, one instance sample and one scoring protocol, under a single randomised assignment — and is reported as Appendix T: both loci move gold-scored correctness under an agreeing verdict at the exact floor, the input locus about twice the output locus there, with partially overlapping affected slots and no mechanism identified.

Whether the neighbours disclose their schema. The six nearest works' PDFs support no estimate, either way, of how often field order is reported. Section 1's point is narrower: a field order is a property of the schema object the author declares, and is therefore the change least likely to be recorded as a decision.

Two things follow that the per-result citations do not make obvious. First, the constraint literature (Tam et al., 2024; Lee et al., 2026a) asks whether structuring output costs performance; we hold the structure fixed and permute inside it, so a null there is compatible with a large effect here and vice versa. Second, rubric compilation builds evaluation graphs whose aggregation is type-checked and deterministic (Chen et al., 2026), and pre-use diagnosis probes a rubric's criteria for coupling before large-scale evaluation (Singh et al., 2026); both are the complement of auditing a pipeline already in use: neither subsumes the other, and a judge that passes an external recomputation can still have moved its checks (Section 3). The last column of our own row, what the design does not isolate, is set out in Appendix N.

R Does the field order move the leaderboard?

Everything above measures the judge's own audit trail. This design asks what that costs the use a judge is most often put to: ranking models. None of the paper's four tasks carries responses from several models, so this design supplies them. One hundred single-turn WildBench tasks, each with its own checklist, are paired with the released outputs of twelve models (Lin et al., 2024), giving $1{,}200$ instances judged by llama-3.3-70b against that task's checklist under the paper's two field orders. The schema is the no-filler one (each instance lists exactly its own k items), the design is the ten-block randomisation of Section 3, and the endpoints were frozen before the first call, in its own plan: a model's score in a run is the mean over its tasks of the share of checklist items judged `yes`, a ranking is the twelve models ordered by it, and every endpoint is a statistic of a *pair* of runs, so the 252 allocations are enumerated exactly as elsewhere.

The two field orders rank the twelve models less consistently than two runs of one field order do. Kendall's τ_b between two runs' rankings is 0.9337 within a field order and 0.7915 across, a difference of $+0.1422$ at $p_{\text{exact}} = 0.0079$, rank $1/252$, with a task-cluster bootstrap interval of

$[+0.0112, +0.1593]$. Discordant model pairs rise from 2.19 of 66 to 6.88. The *top-ranked* model differs on 100% of cross-order run pairs against 20% of same-order pairs, and the top-three set differs on 100% against 35.6%. A model's score moves by at most 0.0275 between two runs of different field orders, against 0.0124 between two runs of one field order. On the same runs the paper's own instance-level endpoints reproduce: check-vector identity gap $+23.31$ pp (93.31% within-order identity against 70.00% across) and T_{stable} $+2.17$ pp, both at the floor.

Averaging ten runs per field order does not make the two agree. Under verdict-first the board is led by claude-3-5-sonnet (0.8992); under checks-first it is led by Meta-Llama-3-70B (0.8842), with claude-3-5-sonnet fourth. Eight of the 66 model pairs swap order between the two boards, and τ_b between them is 0.7576.

Against run-to-run noise this is not a near-tie; against task sampling the leader itself is uncertain. The four leading models sit within 0.0055 of each other under verdict-first, so the reordering could be jitter in a crowded region. Over RUNS it is not: within verdict-first claude-3-5-sonnet scores above Meta-Llama-3-70B in *ten of ten* runs and within checks-first the reverse holds in *ten of ten*, on non-overlapping ranges (0.8969–0.9031 against 0.8757–0.8780 for claude-3-5-sonnet), and each field order's own top model is the same in nine of its ten runs. Over TASKS it is: resampling the hundred tasks, claude-3-5-sonnet leads under verdict-first in 38.3% of resamples and Meta-Llama-3-70B under checks-first in 32.6%, and the gap between the two leaders inside their own field orders is $+0.0054$ $[-0.0321, +0.0384]$ and $+0.0073$ $[-0.0240, +0.0427]$. So the ordering is stable given a field order and a task set and reverses with the field order; which model leads is a property of this task sample; what survives task resampling is the concordance difference itself, whose interval excludes zero.

The shift is also not a uniform rescaling. The four strongest models lose 0.010 to 0.022 under checks-first while claude-3-haiku gains 0.018, Llama-2-7b 0.014 and Mistral-7B 0.007: the board compresses *and* reorders, which is why a rank statistic and not a mean-score comparison is the endpoint.

Two checks, and one note on the interval (the supplementary material, part R). Recomputed on the 84 tasks where all twelve models are present, the boards keep the same leaders and agree slightly more with each other (τ_b 0.7879 against 0.7576), so the reversal is not an artefact of models being scored on slightly different task sets; the full twelve-model boards under both field orders are tabulated there. And the task-bootstrap interval $[+0.0112, +0.1593]$ is not centred on $+0.1422$: resampling tasks raises *both* concordances and the cross term more, so the resampled difference averages $+0.0733$ — the point estimate is the full-task-set quantity and the interval bounds its movement under task sampling. Post hoc, recompute-gated.

Boundaries. One judge, one benchmark, one hundred tasks, twelve evaluated models, one batch of twenty runs; $1{,}183$ shared instances with 36 ERROR rows of $24{,}000$ dropped under the paper's policy. There is no human gold on these rankings: the comparison that carries the finding is against the same-order baseline, not a correct ordering, so it shows that the two field orders disagree, not which ranks models better. A leaderboard built this way should report the field order it used.

S WHICH FORMAT CHANGES MOVE THE CHECKS: A PERTURBATION CONTROL

Is the verdict-block position a distinguished axis, or does any meaning-preserving format change move the checks comparably? This control answers that directly. Five conditions ran on the frozen 999-instance claim-verification sample, two runs each, in one batch per judge model (llama-3.3-70b and qwen3-next-80b-instruct 2026-09-09, gpt-5.4 2026-09-10, palindrome-interleaved, per-run recorded seeds): the **base** (original template, verdict-first); **W**, a whitespace-only change to the response-format skeleton (indentation doubled; keys, key order and every instruction byte-identical); **K**, the six check keys emitted in reverse order inside the checks object (top-level key positions and the numbered criteria sections unchanged); **B**, the `issues` line moved ahead of the checks object (the verdict block stays first); and **V**, the original template checks-first — the paper's verdict-block move, given a same-batch value so no comparison here crosses batches. Each condition's single-edit invariant is asserted on all 999 instances (Appendix C); ERROR rows total 3, 17 and 10 across the thirty runs, every run far above the 950 floor.

Table 14: Format-perturbation comparison, one common instance set per judge model (n = the number of ids all ten same-batch runs answered). Each condition changes one thing in the response-format block against the same two verdict-first base runs; *within* averages the condition's own pair with the base pair, *cross* the four condition-against-base pairs, and identity gap is their difference, with a paired instance-bootstrap interval (B=2,000, conditional on the observed runs); T_{stable} is the joint verdict-same-and-vector-changed difference on the same pairs. Emitted key order matches each condition's requested skeleton on 100% of retained rows in every run, and vectors are aligned by field name, so key order alone cannot change them.

Judge model	n	perturbation	within	cross	gap (pp)	T_{stable} (pp)
llama-3.3-70b	996	W: indentation only	91.97	90.09	+1.88 [+0.8, +3.1]	+1.15 [+0.1, +2.2]
		K: check keys reversed	92.12	59.76	+32.35 [+29.7, +35.1]	+31.43 [+28.8, +34.2]
		B: issues block moved	90.31	69.58	+20.73 [+18.3, +23.2]	+19.65 [+17.2, +22.1]
		V: verdict block moved	93.17	59.46	+33.71 [+30.8, +36.6]	+22.99 [+20.3, +25.7]
qwen3-next	988	W: indentation only	95.85	89.78	+6.07 [+4.6, +7.6]	+4.63 [+3.3, +6.0]
		K: check keys reversed	96.00	75.30	+20.70 [+18.3, +23.4]	+18.83 [+16.4, +21.4]
		B: issues block moved	92.05	65.51	+26.54 [+24.0, +29.2]	+24.57 [+22.0, +27.3]
		V: verdict block moved	94.99	64.57	+30.41 [+27.6, +33.3]	+20.39 [+18.0, +22.9]
gpt-5.4	990	W: indentation only	85.56	79.95	+5.61 [+3.9, +7.3]	+4.19 [+2.6, +5.8]
		K: check keys reversed	84.65	68.74	+15.91 [+13.7, +18.1]	+13.59 [+11.5, +15.8]
		B: issues block moved	83.43	69.27	+14.17 [+12.0, +16.3]	+12.07 [+10.1, +14.2]
		V: verdict block moved	84.90	66.19	+18.71 [+16.3, +21.1]	+3.59 [+1.5, +5.6]

Reading. The sensitivity is to *schema key order as structure*, not uniquely to the verdict block. The indentation-only condition moves little on llama-3.3-70b and small but nonzero amounts on the other two judge models (Table 14), though one indentation change does not stand in for all formatting perturbations; every structural reordering out-moves it, with paired identity-gap differences against the whitespace condition from +8.56 [+6.24, +10.78] (B on gpt-5.4) to +31.83 [+28.86, +34.74] (V on llama-3.3-70b), every interval excluding zero (all nine: the supplementary material, part S; paired instance bootstrap on the common set, B=2,000, conditional on the observed runs — the randomised design's exact p does not transfer here). On each judge model a non-verdict reordering's verdict-stable term *exceeds* the verdict-block move's: the key reversal on llama-3.3-70b (paired difference +8.44 [+5.67, +11.42]), the issues move on qwen3-next (+4.18 [+1.26, +7.29]; the non-exceeding conditions' differences are −3.34 [−6.25, −0.53] and −1.56 [−4.68, +1.52]), and both on gpt-5.4 (+10.00 [+7.22, +12.90] and +8.48 [+5.86, +11.09]), where the verdict-block condition carries the largest identity gap but the smallest verdict-stable term (+3.59 [+1.5, +5.6], back at its earlier-batch randomised magnitude — the batch sensitivity Appendix E documents). The V condition doubles as a same-batch replication of the core numbers (its rows beside Tables 2 and 7). Two consequences: the finding is a case of a broader sensitivity (Sclar et al., 2024) — the emitted per-criterion channel moves under structural reorderings of the output schema generally, so disclosure and same-order replicates are needed for *every* schema-order choice tested here — and the verdict-block move is the intervention this paper validated in depth (randomisation, correct-verdict split, precedence 2 × 2), the same machinery being applicable to any of these reorderings. Descriptive 2+2 contrasts, same-batch by construction; one task, three judge models; the K condition is additionally replicated on scientific-data verification (Appendix O) and, below, block-randomised on an independent batch.

The reversal, randomised. One question the descriptive contrasts above cannot answer is whether a key structural reordering holds at the randomised, independent-batch grade. A key-reversal design answers it on the primary configuration under the exact design of Section 3 (the base template against its key-reversed variant, both verdict-first; allocation 73 of the 252, under its own frozen plan; 2026-09-10, one batch, independent of the 2+2 above). On the n=982 instances all twenty runs answered: $T_{\text{stable}} = +32.40$ pp at $p_{\text{exact}} = 0.0079$ — rank 1 of 252, the design's two-sided floor — instance interval [+29.8, +35.1]; identity gap +32.97, also at the floor, [+30.5, +35.7]. The descriptive same-batch K condition above (+31.43 [+28.8, +34.2]) is thereby replicated across batches under randomisation, and the reversal's verdict-stable term stands numerically above the verdict-block move's own randomised +23.00, a descriptive cross-design comparison.

T INPUT-SIDE AGAINST OUTPUT-SIDE REVERSAL, ON ONE DENOMINATOR

Whether this paper's intervention is a different *locus* from the constraint-list perturbations of Lee et al. (2026b), and not a different mechanism, is a positioning claim (Appendix Q) that neither paper's results settle. Settling how the two loci compare needs both reversals on one judge model, one instance sample and one scoring protocol: permute the order in which the constraints are *listed to the judge* and permute the *declared output schema*, each against its own same-configuration replicate, under a single randomised assignment. Two things are read off it: whether the effect sizes differ, and whether the two interventions move the same constraints — per-constraint overlap measures whether the perturbations reveal complementary failures, and does *not* identify distinct mechanisms.

We ran it. Two configurations under the exact design of Section 3, on llama-3.3-70b over the same MCJudgeBench sample, the same rubric and the same frozen aggregation rule as Appendix P, in one batch, with field order held at verdict-first in all forty runs so the verdict-block position — this paper's own intervention — is constant and cannot leak into either contrast. A condition is a task variant: one configuration reverses the order the constraint sections are *listed to the judge*, the other reverses the order the fifteen check keys appear in the *declared output schema*. The two reversals are not matched in what they reverse: the input condition reverses an instance's *own* k sections (k is 2 to 15; median 4 on this comparison's 217-instance panel, 5 on the full 219), while the output condition reverses all fifteen declared slots, of which about 69% are filler on the average instance. A strictly matched output condition — reversing only the k live slots, or running on the no-filler template — is not run here; the existing no-filler control tests the verdict-block move, not this. So the comparison below is between two reversals at the two loci, not between two reversals of the same slots. Design, endpoints, the overlap baseline and the statement that overlap does not identify mechanisms were frozen in its plan before the first call. Common panel 217 instances, 1,009 listed constraints: two ERROR rows in one run, dropped under the paper's policy and not retried.

Both loci move gold-scored correctness under an agreeing verdict, at the exact floor. Reversing the input listing gives $+3.53$ pp (3.76% across conditions against 0.23% within); reversing the output schema gives $+1.81$ pp (2.20% against 0.39%); both p_{exact}=0.0079, rank $1/252$. The input locus is the larger of the two here, roughly twice the output locus: the perturbation family MCJudgeBench already documents moves more of this endpoint on this task than ours does.

They overlap partly, and each moves slots the other does not. Call a slot *affected* by a locus when its gold-scored correctness migrates in at least one cross-condition pair of that configuration: 33 slots are affected only by the input reversal, 13 only by the output reversal and 24 by both, so 65% of the output-affected set is also input-affected; requiring instead that a slot's cross-condition migration *rate* exceed its same-condition rate (100 cross-condition pairs against 90) gives $33/15/19$ and a Jaccard of 0.284, so the reading is not indifferent to the definition. These sets use the prespecified unconditional migration and do not condition on the verdict. The Jaccard index is 0.343 against a same-condition baseline of 0.345; an instance bootstrap on the difference gives -0.002 with a 95% interval of $[-0.184, +0.188]$, far too wide to support "no excess overlap". Each locus moves slots the other does not; whether the shared 24 reflect one sensitivity showing up twice is *not* settled here, and as pre-committed none of this identifies a mechanism.

One manipulation-check figure changes how the output configuration reads. Asked for the fifteen keys in reverse, llama-3.3-70b emits them in canonical order anyway on 96.07% of scored instances (3.93% compliance, read from the raw text before parsing; the canonical conditions comply at 100%). The paper's existing key-reversal condition on FEVEROUS with gpt-5.4 does comply. So the output configuration's $+1.81$ pp is produced by a *declared* order the model largely does not reproduce in its emission — which is this paper's claim about prompt-serialised schemas, measured here rather than assumed; it was recorded as an amendment to the frozen plan after a nine-call smoke test and before the full run.

Splitting the output configuration's excess by whether *both* runs of a pair emitted canonical order — additive on the same pairs and denominator — puts $+1.580$ of the $+1.808$ pp on pairs where both emitted canonical order and $+0.228$ on pairs where at least one did not: pairs in which both outputs retained canonical order account for 87.4% of the observed excess, and the remaining 12.6% arises where at least one output was reversed. The effect therefore sits mainly, but not wholly, where the declared order moved and the emitted order did not. Descriptive, and not a mechanism claim.

On the difference between the two loci. Each configuration clearing $p_{\text{exact}}=0.0079$ says nothing about whether they differ from *each other*. The difference is $+1.72$ pp, with a paired instance bootstrap interval of $[+0.36, +3.18]$ — conditional on the observed runs, covering instance sampling only, and silent about run-to-run and batch variation. It excludes zero, so calling the input locus the larger here is supported descriptively; it is not a claim that the ordering holds on other judge models, tasks or reversals.

Scope: one judge model, one dataset, one pair of reversals, 217 instances. This says how the two loci compare here, not that they differ in general.

U FULL CROSS-TASK MATRICES, AND CONFIGURATION CASES

Selected-policy disagreement (insufficiency-first on claim verification; rule violation elsewhere) and abstention rates for every measured (model, task, field order) configuration of the open-weight matrix. Verdict-first same-order differences are measured for every configuration and never exceed 1.28 pp; checks-first differences on the reversal configurations span 0.29–1.39 pp and reasoning-first ones 0.00–0.51 pp. On safety rating llama-3.3's violation rate falls $6.57 \rightarrow 1.14\%$ (Table 9) and the abstention channel moves in *opposite directions* on different judge models under the one intervention: llama-3.3's abstentions fall $30.29 \rightarrow 4.29\%$ while qwen3-next's rise $4.29 \rightarrow 19.43\%$ and deepseek-v3.1's $0.00 \rightarrow 20.57\%$. A reader of any single field order sees a coherent instrument; only the cross-order comparison shows the internals are relative to the field order. The level tables give the verdict-first same-order rerun its own column; the checks-first reruns and the deepseek-v3.1 and proprietary-model configurations are in the blocks below. ERROR rows are excluded from n (Section 2).

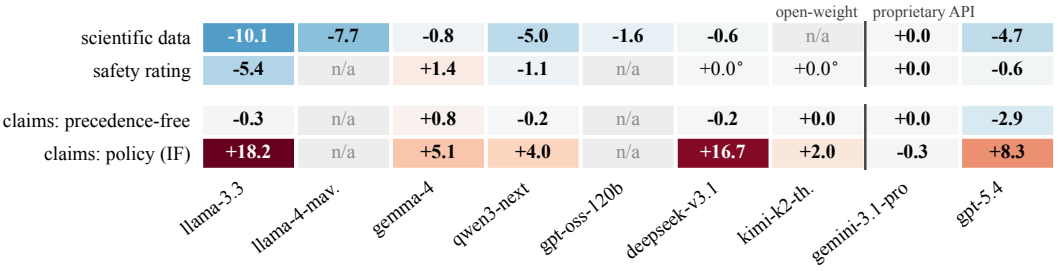


Figure 3: Rule violation and policy disagreement across the fixed-order matrix: CHECKS-FIRST minus VERDICT-FIRST, in percentage points; blue a fall, red a rise, one scale symmetric about zero across both panels. **Top:** the two tasks where the taught fail and not-enough-info clauses never co-fire in any observed output (0 of 350 safety and 0 of 1,223 scientific-data instances, every field order), so the number is a change in the *rule-violation* rate under every admissible reading. **Bottom:** claim verification, where the rubric leaves the precedence open, decomposed into what the rule forbids under *every* admissible reading (precedence-free) and what *one* admissible policy resolves differently (IF, insufficiency-first; the other reading: Appendix L). ° marks a configuration whose $|\Delta|$ does not exceed *its own* observed same-order difference; a configuration with no same-order replicate for that field order carries no marker; `n/a`: not run. Values are recomputed from the run files and cross-checked against the level tables (the supplementary material, part U).

The levels behind Figure 3's differences — violation and abstention rates for every open-weight configuration under each field order, with the evidence-sufficiency firing counts on claim verification — are tabulated in the supplementary material, part U.

Checks-first same-order replicates (replicate batch, 2026-08-19). Claim verification: llama 24.72% (vs. 23.95 original), gpt-5.4 12.71 (11.32), qwen3-next 7.53 (7.81), gemma 5.61 (6.41); safety rating: llama 1.43 (1.14). Reasoning-first replicates on claim verification: llama 9.71 (10.22), gpt-5.4 1.30 (1.30); gemma-4's, partially treated, are not read as a control. Every reversal magnitude replicates within 1.4 pp.

Scientific-data verification configurations (benchmark harness, n=1,223). llama-3.3 $10.22 \rightarrow 0.08$ (125 and 1 of 1,223; same-window rerun $10.48 \rightarrow 0.08$), llama-4-maverick $7.93 \rightarrow 0.25$, qwen3-next $5.45 \rightarrow 0.49$, gemma-4 $12.59 \rightarrow 11.77$, gpt-oss-120b $7.36 \rightarrow 5.81$, deepseek-v3.1 $4.99/4.42/6.30$ across field orders (violations flat while accuracy moves, the silent-accuracy pattern of Appendix N); same-window proprietary-model pairs: gpt-5.4 $4.66/4.25 \rightarrow 0.00$, gemini-preview $0.00/0.00$.

Configuration case studies.

The thinking pair. kimi-k2-thinking shows $\sim$25% selected-policy disagreement in every claim-verification field order, all of it in the ambiguous region (precedence-free 0%, Appendix N): a counterexample to matching an analyst's *selected policy*, not to rule-following. The qwen3-next thinking/instruct pair is scored three ways in the supplementary material, part U (it favours the thinking variant on conditional consistency among completed outputs only, completing on 45.2% of scientific-data instances against its sibling's 98.9%); why it is not reported on claim verification at all is in Appendix C.

The two proprietary API models split — and the metric changes meaning. On claim verification, gemini-3.1-pro-preview *changes little*: $3.40 \rightarrow 3.10\%$, a cross-order spread of 1.10 pp against a single observed same-order difference of 0.70 pp (descriptive; Section 2); it also posts zero violations in every safety-rating field order. gpt-5.4's selected-policy disagreement rises on the same task, $3.00 \rightarrow 11.32\%$, against a same-order difference of 0.10 pp, and what changes is what the figure is made of: verdict-first the disagreements are over-abstentions no admissible precedence excuses, checks-first they are hard-fail-first resolutions the rubric permits, while the evidence-sufficiency check fires at an order-*stable* rate (counts: Appendix L); where gpt-5.4's full check vector is provably unchanged across field orders, the selected-policy counts still move with the order (Appendix L, exploratory). Two caveats bind it. Under the hard-fail-first reading its rates run $10.71 \rightarrow 0.70\%$ — the direction inverts — so "gpt-5.4 reverses" is a statement about the insufficiency-first reading, not a precedence-free fact. And one metric name summarises qualitatively different phenomena in the two orders, which is this paper's thesis at its sharpest: field order changes the *composition of what the metric summarises*, not merely its value. Neither vendor closes the check channel (Table 7), and within gpt-5.4 alone the abstention rate moves in opposite directions across tasks ($+13.4/-13.4$ pp): two API models are not a regime.

V Does aggregating runs remove the effect? Majority-vote committees

Two mitigations need no new call, because every randomised configuration already has ten runs per field order: *self-consistency* (vote k runs of one field order per check) and *cross-order voting* (vote runs drawn from both field orders). The table in the supplementary material, part V forms committees of $k = 1, 3, 5$ same-order runs and mixed committees of five (two or three runs from each field order, with the composition left free and with it held fixed) and applies the paper's estimands to the committees' majority check vectors, with a tie on any check leaving that entry undetermined (tie rates are below 1.3% everywhere).

Self-consistency sharpens the effect. Voting five same-order runs raises within-order identity and leaves cross-order identity where it was, so identity gap and T_{stable} grow on every claim configuration (on the primary configuration T_{stable} goes $+23.00 \rightarrow +25.09$ pp, on gpt-5.4 $+7.83 \rightarrow +13.09$), while the safety configuration's near-zero verdict-stable term stays near zero. Voting removes run-to-run noise; what remains is the systematic part of the difference between field orders, which is what the estimands were built to isolate.

Cross-order voting does not average the field orders away; the majority field order sets the check vector. With the composition left free, a committee that mixes field orders reproduces its own check vector far less often than a same-order committee of the same size (78.70% against 95.87% on the primary configuration), but that figure pools two comparisons, half of its pairs setting a verdict-first majority against a checks-first one. Holding the composition fixed separates them (the supplementary material, part V): two disjoint committees of the same 3:2 or 2:3 composition

Table 15: Interventions that do *not* remove the effect. The voting row re-analyses existing runs; every other row has its own frozen plan (the two decoder rows share one), and all but the two-stage row are randomised designs. The appendix named beside each reports it in full, with its denominators and intervals. "Sharpens" compares committees built from the same runs, and the decoder-enforced row compares two batches; both are descriptive, not magnitude claims (Section 3).

Intervention	Does it remove the effect?
Aggregating runs — voting, mixed committees, cross-order voting	No. Five same-order runs per check *sharpen* T_{stable}, $+23.00 \rightarrow +25.09$; a mixed committee's checks follow its majority field order; cross-order voting leaves MCJudgeBench accuracy where it was. Abstaining where two calls disagree does raise gold-scored accuracy — $+1.44$ and $+2.66$ pp at 96.22 and 93.33% of slots kept — but it uses the effect rather than removing it (Appendix V)
Decoder-enforced schema	No. $+9.95$ on gpt-5.4 against $+7.83$ declared-only (Appendix M.1)
Emitted key order held fixed by the decoder	No. The *declared* order alone still moves the checks, $+9.47$ (Appendix M.1)
Sampling at temperature 0.7	No. $+20.89$, still at the design's floor (Appendix M.4)
Splitting the call in two	No detectable gain in accuracy or reproducibility; its checks land with the checks-first single call's (Appendix M.2)

agree within 0.4 to 3.4 pp of the same-order committee on four configurations and 7.3 to 8.1 pp below it on claim verification $\times$ gpt-5.4, while two committees with opposite majorities agree on only 65.40 to 89.75%: mixing does not cost reproducibility as such, it makes the check vector a function of which field order holds the majority — the order effect again at committee level. On MCJudgeBench, where the committee's checks can be scored against gold, neither form of voting moves per-constraint accuracy by more than 0.4 pp while the correctness-vector identity gap grows with k as on the claim configurations: aggregation is not a mitigation of the order effect on these runs but a sharper instrument for measuring it. Descriptive and post hoc; the seeded committee samples and the exact enumerations are in the artifact.

Abstaining on a disagreement is the other use of a second call: keep the slots the two calls agree on and send the rest to review. On MCJudgeBench one call in each field order raises per-constraint accuracy on the kept slots by $+1.44$ and $+2.66$ pp against $+0.30$ and $+1.43$ for a same-order repeat, but discards 4.6 and 1.8 times as many slots, and the discarded pool gives no evidence that either filter aims better — the gain is volume, not aim (the supplementary material, part V).