When One More Constraint Is Added: Decomposing Multi-Constraint Failure in Instruction-Following Evaluation

Anonymous authors
Paper under double-blind review

Abstract

Multi-constraint instruction-following benchmarks usually reduce a response to one score: did it satisfy every constraint at once? When that score falls as constraints are added, it does not say why. Did the model fail the new constraint, lose one it had already satisfied, or did the evaluation change its judgment while the behaviour stayed the same?

We follow each constraint as the prompt grows, separating two events: satisfying the newly added constraint, and breaking one that previously passed. On FollowBench this needs two automated steps, and both are fragile. Software extracts the text of each added constraint, and an LLM judge decides whether a response satisfies it. Two plausible judging protocols disagree by as much as 55.9 percentage points on identical outputs. On a slice of Llama responses with rule-based checkers, the judge marks 30.2% of rule-passing constraints as failed, and 13 of 14 constraints that failed and then passed again were judge errors, not real recoveries. Human annotators confirm 297 of the 392 tracked constraints, counting every disagreement or ambiguous case as not confirmed. Under our frozen analysis, constraints already satisfied show no detectable steady rise in breakage across four instruction-tuned models, though rises up to 4.7 points per added constraint remain compatible with the data, and dropping the 95 unconfirmed constraints gives the same answer.

We then repeat the question on ManyIFEval, where instructions are listed separately and checked by a program, so neither step is needed. Following one instruction while four more are added, its satisfaction changes by between -1.44 and $+1.20$ points per added instruction, with every 95% interval inside $[-2.36, +2.64]$; only Qwen declines detectably. Yet the chance that an instruction satisfied at one step fails at the next rises in two models. How often a constraint is satisfied and how often a satisfied constraint is lost give different answers about the same responses, even under exact scoring. Multi-constraint evaluation therefore depends not only on how many constraints are satisfied, but on which change is measured and how it is judged.

1 Introduction

A model must write a summary in bullet points while avoiding jargon. It succeeds. A new constraint is then added to the same prompt: keep the response under 200 words. The model now returns plain prose. A standard benchmark records a failure, but does not tell us what happened. Did the model fail the word limit? Did the bullet-point rule break when the new constraint arrived? Or would the model have dropped the bullet points anyway?

The answer matters. If adding constraints makes the model lose ones it was already satisfying, composition is itself a problem. If not, improving how well it satisfies single constraints is the more direct path.

Multi-constraint benchmarks (Jiang et al., 2024; Wen et al., 2024; Harada et al., 2025; Guo et al., 2026) show that performance falls as more constraints are added. Commonly reported aggregate scores do not tell us whether a particular constraint that previously passed still passes after another is introduced. Vasileva (2026) finds failures approximately independent in aggregate across up to 12

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.

deterministically verified constraints, which leaves the directional question open: an old constraint could break when a new one arrives even if those losses average out across prompts. He et al. (2024) do track that transition across conversational turns, where each turn brings new instructions and new context.

We therefore follow each constraint individually as the prompt grows. We call losing a previously satisfied constraint *breakage* and satisfying the newly added one *acquisition* (Section 2, Figure 1). The decomposition is simple. Making it measurable is not.

FollowBench shows why. Measuring breakage means deciding what the added constraint actually is, and then deciding whether each response satisfies it. Both steps are fragile. Two plausible judging protocols disagree by as much as 55.9 pp on identical outputs, because they disagree about what counts as the same constraint across levels. On a slice of Llama responses with rule-based checkers, the judge marks 30.2% of constraints as failed when the checkers say they passed, and 13 of 14 constraints that failed and then passed again were judge errors. Human annotators confirm only 75.8% of the 392 tracked constraints, counting disagreements and ambiguous cases against us. Yet dropping every unconfirmed constraint leaves the result unchanged: across four models we find no detectable steady rise in breakage, with rises as large as $+4.7$ pp per added constraint still compatible with the data. One answer in five from Gemma and Mistral stopped at the token limit, so a lost constraint might only mean a cut-off answer, and rerunning both with four times the budget does not change the conclusion.

Because every FollowBench estimate still depends on extracted constraint text and an LLM judge, we repeat the analysis on ManyIFEval, where instructions are listed explicitly and programs check them deterministically. We follow one instruction while four more are added around it. Its satisfaction changes by between -1.44 and $+1.20$ pp per added instruction across models, every 95% interval falling inside $[-2.36, +2.64]$, and only Qwen declines detectably. But asking a different question about the same responses gives a different answer. For Llama, satisfaction of that fixed instruction shows no detectable trend (-1.06 pp per added instruction), while breakage among instructions that passed one step earlier rises from 3.3% to 9.0%. Breakage trends are positive in all four models and detectably so in two.

1.1 Contributions

1. A **retention–acquisition decomposition** that separates losing a previously satisfied constraint from failing the newly added one, and links both to marginal satisfaction through recovery (§2).

2. A **measurement audit** showing that constraint identity, judgment validity, and score arithmetic or generation settings can each create apparent effects, including a judge that produces an apparent rising breakage pattern on the audit slice that rule-based labels do not show (§4).

3. **A FollowBench result with explicit uncertainty**: no detectable rise in breakage across four models, with 95% upper confidence limits no larger than $+4.7$ pp per added constraint, stable under a second judge, random and level-dependent corruption simulations, and a long-budget rerun (§5).

4. **A deterministic test** on 216 ManyIFEval tasks, with no judge and no extraction. For a fixed instruction, all four model-specific 95% intervals lie within $[-2.36, +2.64]$ pp per added instruction, while conditional breakage rises in two models, so marginal and conditional estimands diverge (§6).

2 Problem Formulation

FollowBench (Jiang et al., 2024) organizes prompts into trajectories. Each trajectory starts with a base instruction and adds one constraint per level. When moving from level $k-1$ to level k, we ask two questions: do old constraints that were satisfied at $k-1$ remain satisfied, and does the model satisfy the new one?

	new c_k: satisfied	new c_k: not satisfied
old c_j survive	**passes**	**fails new only**
old c_j break	**broke old only**	**both fail**

Figure 1: For one previously satisfied constraint c_j, four outcomes are possible when c_k is added. *Breakage* (B_k) conditions on c_j being satisfied at $k-1$; *acquisition* (A_k) tracks the new c_k. A whole-response pass or fail conflates both. The four-way response-level table is in Appendix O.

Let $Y_{i,k,j} \in \{0, 1\}$ denote the **recorded judgment** that constraint j is satisfied in the response for trajectory i at level k, for a constraint j introduced at or before level k. For the step from $k-1$ to k:

$$B_k = P(Y_{i,k,j} = 0 \mid Y_{i,k-1,j} = 1),\ j<k \quad \text{(breakage: old constraint lost)}$$
$$A_k = P(Y_{i,k,k} = 1) \quad \text{(acquisition: new constraint gained)}$$
$$R_k = P(Y_{i,k,j} = 1 \mid Y_{i,k-1,j} = 0),\ j<k \quad \text{(recovery: old constraint regained)}$$

A single pass/fail verdict cannot tell these apart; Figure 1 shows the four outcomes. For one old constraint j, write $p_{k,j} = P(Y_{i,k,j}=1)$, and let $B_{k,j}$ and $R_{k,j}$ be the breakage and recovery rates above for that constraint. Then

$$p_{k,j} = p_{k-1,j}\,(1 - B_{k,j}) + (1 - p_{k-1,j})\,R_{k,j}, \tag{1}$$

and the same identity holds for any fixed population of old constraints observed at both $k-1$ and k, with B_k and R_k the pooled rates. Marginal satisfaction can therefore stay flat while B_k rises, if failing constraints also recover. This is why the choice of estimand matters in Section 6.

Three features of B_k need care. It conditions on a recorded pass at $k-1$, so a judge error there can create apparent breakage (§4.4). It conditions on survival, so harder constraints leave the risk set early and later breakage can look lower (frailty selection; §5.2). And it assumes constraint j is the same object at every level, which Section 4 shows must be checked.

3 Experimental Setup

We use FollowBench because its nested trajectories let us observe the same constraint before and after a new one is added.

Sample. FollowBench (Jiang et al., 2024) contains 147 five-level trajectories (735 prompts), organized into constraint families, plus 85 Mixed items that combine constraints from different families. Our primary analysis uses the four families that constrain the response itself: Content, Situation, Style and Format, covering 107 trajectories. We exclude Example, whose levels change few-shot demonstrations rather than constraints on the response, and Mixed, because tracking the identity of a constraint across levels is less straightforward when families are combined. Of the 85 Mixed items, 75 are rule-evaluable, so this exclusion reduces the deterministic coverage available for our judge audit. After masking superseded constraints, the primary analysis contains 952 in-force constraint pairs and 392 distinct constraints entering B_k (Appendix Table 4).

Models. Llama-3.1-8B-Instruct, Gemma-3-12B-it, Mistral-Small-3.1-24B-Instruct-2503, and Qwen2.5-32B-Instruct.

Generation. Greedy decoding, a 1024-token budget, batch size 1, left padding, each model's official chat template, HuggingFace Transformers 4.51.3 (Mistral: 4.57.6), on an A100-SXM4-80GB GPU. Section 4.6 shows why these settings are pinned. Full provenance is in Appendix R.

Judging. Many FollowBench constraints cannot be checked with simple rules, so our primary analysis uses an LLM judge: a fixed GPT-4o-mini snapshot at temperature 0. We chose GPT-4o-mini because it made repeated judging of the full evaluation set practical at this scale. The judge evaluates each constraint against a text extracted when the constraint is introduced and then held fixed at every later level where it remains in force. It returns one verdict per constraint as structured output, with no reasoning step, so the error rates in Section 4.4 describe direct zero-shot scoring rather than LLM

judging in general. For 120 items across 24 trajectories, deterministic verifiers exist; we use this slice to audit the judge, not to replace it. Style has no rule-verifiable coverage. A small IFBench pilot (Pyatkin et al., 2025) is reported in Appendix U.

4 MAKING RETENTION MEASURABLE

A rise in B_k means constraints are being lost only if three things hold. First, the constraint judged at level k must be the same one judged at $k-1$. Second, the verdicts must be correct. Third, neither the counting procedure nor the generation settings can create a trend on their own. All three failed somewhere in our pipeline, so checking the instrument became part of the analysis rather than a step before it. Sections 4.1 to 4.3 examine constraint identity, 4.4 and 4.5 the verdicts, and 4.6 counting and generation effects.

4.1 CONSTRAINTS ARE SUPERSEDED, NOT ONLY ACCUMULATED

Retention requires that constraint j at level k be the same object as at $k-1$. This is not always true. Reading the trajectories, we found three patterns and named them ourselves; FollowBench does not mark any of them. In *cardinality replacement*, level 4 asks for two items and level 5 asks for three. In *count ladders*, successive levels ask for 2, 3, 4, then 5 events: one constraint with a changing parameter. In *subsumption*, a stronger later requirement absorbs an earlier one. Treating a superseded constraint as still required scores a correct response as interference, which can inflate B_k. We therefore freeze each constraint text at the level where it is introduced and judge it only where it remains in force.

A rule-based classifier, developed on 20 held-out trajectories and then applied blindly, flags supersession. We hand-verified all 72 flagged pairs and 100 random negatives: 41 were confirmed, 10 were false positives, and 21 were ambiguous (precision 80.4% excluding ambiguous); no sampled negative was a missed case (one-sided 95% upper bound about 3%). Confirmed supersession affects 3.2% of primary pairs, ranging from 8.8% in Content to 0.7% in Style (Appendix Table 12). The 50 flagged pairs in the primary pool are masked.

4.2 JUDGING PROTOCOLS TRACK DIFFERENT OBJECTS

FollowBench ships with its own judging protocol, which we call Protocol A. It shows the judge the whole chain of prompts up to level k, asks it to work out which constraints were added along the way, and scores the response against that list. Protocol B is ours: we take the text of each constraint once, when it is introduced, and ask the judge about that fixed text at every later level where it still applies. Protocol A rebuilds the list at each level, so what is being scored can shift as the prompt grows; Protocol B holds it fixed.

Scoring identical Llama outputs on 40 trajectories, the two differ by 35.2 pp on average and by as much as 55.9 pp. At level 4, Protocol A reports 78.4% of trajectories as fully satisfied and Protocol B reports 22.5% (Appendix K). A hard-satisfaction score, the share of prompts where every constraint in force is judged satisfied, therefore cannot be read without knowing which protocol produced it.

4.3 HUMAN VALIDATION SHOWS SUBSTANTIAL CONSTRAINT-IDENTITY UNCERTAINTY

Holding the constraint text fixed does not make it right: the text we extract may not match what the level actually added. Two annotators labelled the 392 constraints entering B_k as correct, incorrect or ambiguous, under a plan frozen before labelling and an interface that showed neither model outputs nor the other annotator's labels: one labelled all 392, and a second independently labelled 155 of them for agreement. Our final-label rule is conservative: ambiguous labels and unresolved disagreements count as not confirmed. Under it, 297 of 392 constraints are confirmed correct, or 75.8% [71.3, 79.7]. The rate differs sharply by family: 87.3% for Style and 85.7% for Format against 49.4% [39.0, 59.8] for Situation, where the added text often rephrases the scenario instead of adding a separable constraint. On the 155 items both annotators labelled, they agree with each other 77.4% of the time, with Cohen's κ=0.39 [0.23, 0.55] and no adjudication. Some cases are therefore ambiguous even to human readers, rather than every disagreement being an extraction failure.

Automated checking is not a substitute. Two validators from different model families flag 31.1% and 4.8% of the same constraints, and against the human labels they reach κ=0.28 and κ=0.05, with recall 0.55 and 0.07. Section 5.2 reruns the primary analysis without the constraints that are not confirmed correct.

4.4 Judge Error Is Large on the Slice We Can Check

LLM judges show systematic biases (Zheng et al., 2023; Wang et al., 2024). For 93 constraints in Llama's responses we can check the answer with a small program, so we know what the judge should have said. On the 63 the program passed, the judge calls 30.2% [20.2, 42.4] failed; on the 30 it failed, the judge calls 43.3% [27.4, 60.8] satisfied. Extraction errors may contribute to both, so we read these as error rates on this slice rather than corrections to apply elsewhere.

The second error matters in a way that is easy to miss. Breakage is measured only among constraints the judge recorded as satisfied a level earlier, so a constraint it wrongly passed joins the pool that can later be recorded as broken: of the 47 constraints entering that pool here, 12 had already failed by the program's verdict, or 25.5% [15.3, 39.5]. Judge error changes not only how many constraints are recorded as broken, but which ones are counted at all. The judge is unstable too. Where the program confirms a constraint was satisfied at two neighbouring levels, it still records a drop 7.0% (Qwen, 3 of 43) to 11.8% (Llama, 4 of 34) of the time (Appendix Q), and judging byte-identical responses again changes 2.9% to 3.6% of verdicts (Appendix E).

Its errors are not spread evenly across levels. On the seven trajectories where every constraint has a program verdict, breakage by the program stays near 6 to 7% at every level (5.9, 5.3, 5.5, 7.4%) while breakage by the judge on the same responses climbs from 0% to 18% (Appendix G). The two labellings put different constraints at risk and the slice is small, so we read this as descriptive, but the shape it produces is the shape we are testing for.

Asking the judge to reason first does not fix this. Re-judging the same 93 constraints with a prompt that asks for a short justification, holding the model, temperature and constraint texts fixed, makes it stricter rather than more accurate: it now calls 47.6% of program-passed constraints failed, up from 30.2%, while wrongly passing 23.3% instead of 43.3%, and agreement with the program falls from 65.6% to 60.2%. Forty-five of the 93 verdicts change on identical pairs, 31 from pass to fail. Prompt design moves where the judge sits between the two errors rather than reducing both (Appendix F).

4.5 Recorded Recoveries Are Almost All Judge Errors

Recovery is the mirror image of breakage: a constraint recorded as failed at one level and satisfied at the next. The judge records it often, in 15% to 47% of cases depending on the model and the step. A recovery is only counted if the judge recorded a failure first, so a missed pass becomes an apparent recovery whenever the next level is recorded as a pass, and the judge misses many passes. On the slice where the program knows the answer, 13 of the 14 recorded recoveries were not real: in 12 the earlier failure was the mistake, in one the later pass was (Appendix W). We therefore do not interpret recovery on FollowBench, though the phenomenon itself is real: Section 6 finds substantial recovery on ManyIFEval, where a program decides both verdicts. More generally, any statistic conditioned on the judge's own verdict inherits its errors, and the rarer the conditioning event, the larger the share that can be error.

4.6 Counting and Settings Can Move the Numbers on Their Own

A score that requires every constraint to pass fails more often as constraints are added, even when the error rate per constraint stays the same. With k independent judgments, each with error probability p, the chance of at least one mistake is $1 - (1 - p)^k$. On 120 checkable items that rate climbs from 36.4% at k=2 to 62.5% at k=5, matching a constant p=0.185. A rising all-pass failure rate can therefore arise mechanically from a constant independent per-constraint error rate, and is not by itself evidence of interference (Appendix L).

Generation settings move the text too. Our first Llama run used a different batch size and padding, and greedy decoding is not bit-exact across such settings (Yuan et al., 2025). Regenerating all 735 responses under the pinned settings changed 405 of them (55.1%) but only 106 of 1,605 verdicts

(6.6%), and between two machines with nominally identical software only 48% of Gemma's short answers came back byte-identical (Appendix E). Scores are far more stable than text, but settings must be pinned and reported.

5 FOLLOWBENCH RESULTS

5.1 NO DETECTABLE RISE IN BREAKAGE

For each model we estimate B_k at levels $k{=}2, \ldots, 5$ and fit a trend across levels. The primary analysis, pre-specified in a timestamped commit before any trend model was fitted, is a GEE (Liang & Zeger, 1986). One trajectory contributes many constraints across several levels, so its verdicts are not independent; a GEE estimates the average trend while allowing for that. We also report a four-point trend, a straight line through the four breakage rates with 5,000 trajectory-bootstrap resamples. Table 1 shows both.

No model shows a detectable trend, and acquisition, satisfying the newly added constraint, shows none either. This is not evidence of zero effect: the largest 95% upper confidence limits leave steady rises of $+4.7$ pp per added constraint compatible with the data under the GEE, and $+3.5$ pp under the four-point trend.

Three things this does not say. Not detecting a trend is not proof of none. A fitted trend can be flat while the rates move: Mistral's breakage rises 9.3 pp between two adjacent levels. Our claim is about a steady rise, not about interference at particular steps. And failing the new constraint while breaking an old one still happens to 10–17% of responses at the last transition for two models (Appendix O).

5.2 ROBUSTNESS TO MEASUREMENT AND ANALYSIS CHOICES

Section 5.1 reports something not happening, so the worry is that we missed a rise that is really there. Five checks remove, one at a time, a reason we might have missed it.

A second judge. GPT-4o, scoring the same frozen constraints, agrees with ours at $\kappa{\approx}0.58$, and under it every breakage-trend interval still contains zero. For Llama it used the archived generations rather than the matched ones (Appendix V).

The constraint text. Section 4.3 left 95 constraints our annotators did not confirm. Dropping them and refitting the pre-specified GEE moves every estimate by at most 0.55 points: Llama $+1.77$ to $+1.73$, Qwen -1.03 to -0.60, Mistral $+1.24$ to $+0.95$, Gemma $+0.40$ to -0.15 (Appendix H).

Truncation. About one in five Gemma and Mistral answers hit the 1024-token cap, and a cut-off answer looks exactly like a dropped constraint. At 4096 tokens the trend moves from $+0.40$ to -0.33 for Gemma and from $+1.24$ to $+1.22$ for Mistral (Appendix E). Keeping only trajectories that never hit the cap also leaves every interval containing zero, the largest shift being Llama's $+1.77$ to $+2.30$ $[-1.07, +5.67]$. Truncation differs by model, which is one reason we do not read differences between models as a ranking (Appendix P).

Who stays in the pool. B_k counts only constraints satisfied one level earlier, so the hard ones leave early. Following a fixed level-1 group instead, satisfaction falls from 77.7% to 72.3% for Qwen and from 87.1% to 77.6% for Gemma, and that fall is what the per-step rates already predict: survival matches the product of step hazards within about 5 pp (Appendix S). Gemma is the one model where a within-constraint fit finds a positive association while the population fit does not, and Section 6 tests that same split with deterministic labels (Appendix X).

Label noise. Flipping verdicts at random or by level never produces a positive interval; randomizing them to each model's base rate does so only at 50% corruption, in under 5% of draws for two models. Only targeted flips overturn the result (Appendix T).

The conclusion holds up, with one limit worth stating. No check produces a detectable rise. But the margin is thin: flipping the right 4 labels out of 392 turns Llama's interval positive, and the right 19 turn Qwen's, while broad corruption does not. The result is safe against the judge error we measured, not against error concentrated exactly where it would matter.

Table 1: FollowBench breakage B_k by level and fitted trend, in points per added constraint [95% CI]. Four-point: trajectory bootstrap. GEE: pre-specified primary analysis, probability scale. No interval excludes zero.

Model	k=2	k=3	k=4	k=5	Four-point	GEE
Llama-3.1-8B	17.1	10.8	11.2	17.9	$+0.27$ $[-2.68, +2.97]$	$+1.77$ $[-1.15, +4.70]$
Qwen2.5-32B	19.2	13.7	12.2	12.7	-2.10 $[-5.03, +0.64]$	-1.03 $[-3.25, +1.18]$
Mistral-Small-24B	23.9	13.5	22.8	20.5	-0.11 $[-3.60, +3.47]$	$+1.24$ $[-2.07, +4.55]$
Gemma-3-12B	10.0	8.4	10.9	9.8	$+0.18$ $[-2.03, +2.27]$	$+0.40$ $[-1.73, +2.54]$

Table 2: ManyIFEval, rule-based labels, 216 tasks per model. Anchor slope: change in satisfaction of the same first instruction per added instruction (primary). All-instruction B_k: breakage pooled over every instruction satisfied one load earlier, a different population from the anchor (frozen secondary). 95% task-bootstrap intervals.

Model	Anchor slope [95% CI] (pp per instruction)	All-instruction B_k (%) k=2	3	4	5	B_k trend [95% CI] (pp per level)
Llama-3.1-8B	-1.06 $[-2.31, +0.14]$	3.3	6.5	7.4	9.0	$+1.82$ $[+0.83, +2.77]$
Qwen2.5-32B	-1.44 $[-2.36, -0.56]$	3.5	3.0	7.3	6.7	$+1.38$ $[+0.38, +2.29]$
Mistral-Small-24B	$+1.20$ $[-0.28, +2.64]$	11.5	14.6	15.9	15.0	$+1.19$ $[-0.88, +3.12]$
Gemma-3-12B	$+0.74$ $[-0.32, +1.81]$	2.1	2.0	2.2	3.8	$+0.52$ $[-0.24, +1.21]$

6 A Deterministic Test on ManyIFEval

Every FollowBench estimate depends on an LLM judge and on extracted constraint text. ManyIFEval (Harada et al., 2025) removes both. Each of its 216 tasks has published prompts with 1 to 10 instructions, and we verified that every n-instruction set is an ordered prefix of the $(n{+}1)$-instruction set with identical arguments. Each task is therefore a nested trajectory like FollowBench's, but with exact constraint identity and a rule-based verifier for every instruction (15 instruction types). This is a cross-benchmark test, not a judge-only ablation. ManyIFEval also differs in what it asks for: short formatting and lexical rules, where FollowBench includes semantic and stylistic requirements. We therefore use it to ask whether the distinction between marginal and transition-conditioned behaviour survives when constraint identity and scoring are deterministic, not to estimate what FollowBench would show under a perfect judge.

Design. We use loads 1 to 5, which is 1,080 prompts per model, with the same four checkpoints, greedy decoding, batch size 1 and a 2,048-token budget; the longest length requirement is about 1,200 tokens. The main measurement was fixed in a timestamped protocol before we generated anything. For each task we take its first instruction and record how often it is satisfied as instructions 2, 3, 4 and 5 are added. That gives one slope per task, in percentage points per added instruction. We average those slopes across tasks, and the intervals come from a bootstrap over tasks; a simulation run beforehand put the worst-case half-width at 1.7 pp. B_k and A_k are frozen secondary analyses, and anything added after seeing results is labelled exploratory (Appendix C).

Marginal satisfaction changes modestly. Point estimates run from -1.44 to $+1.20$ pp per added instruction, and no interval reaches beyond $[-2.36, +2.64]$ (Table 2, Figure 2 left). Three models show no detectable change. Qwen declines slightly: its first instruction is satisfied 92.6% of the time on its own and 88.0% with four more added (-1.44 $[-2.36, -0.56]$). Dropping tasks whose answers ran into the 2,048-token limit leaves Llama, Gemma and Qwen almost unchanged, but moves Mistral from $+1.20$ to $+0.08$ $[-1.60, +1.76]$. That changes the size of Mistral's estimate, not the reading: still no detectable change. We keep all 216 tasks in the primary analysis, because the dropped tasks are not a random subset. They average 873 generated tokens against 367 for the rest, and 34 of Mistral's 85 hit the limit at load 1, before any instruction was added (Appendix D). One frozen check does find a decline. Restricted to the 32 to 84 tasks per model whose first instruction actually changes state, it is negative for Llama and Qwen, with intervals excluding zero (Appendix C). Conditioning on movement changes what the same responses show.

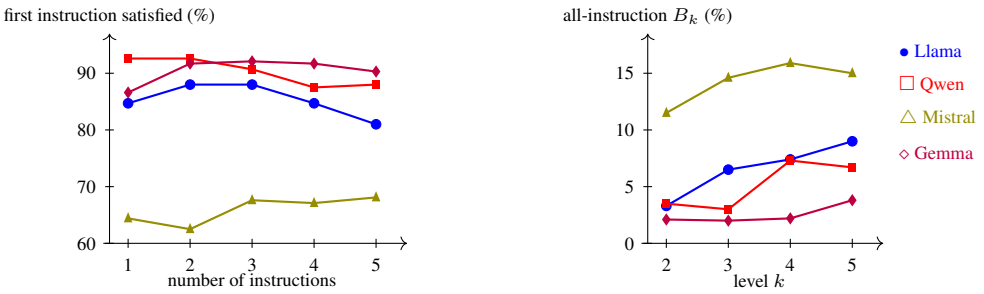

Figure 2: Same responses, two estimands over different populations. Left: satisfaction of each task's fixed first instruction as instructions are added (primary). Right: breakage pooled over all instructions satisfied one load earlier. Rule-based labels, 216 ManyIFEval tasks per model.

Conditional breakage rises detectably in two models. On the same responses, the fitted B_k trend is positive in all four models, with intervals excluding zero for Llama ($+1.82$; GEE $+1.96$ $[+0.60, +3.32]$) and Qwen ($+1.38$; GEE $+1.65$ $[+0.40, +2.90]$) (Figure 2 right). Qwen's acquisition also falls (-2.59 $[-4.35, -0.83]$), while the other three acquisition trends contain zero. Comparing the same estimator across benchmarks, the GEE breakage trends have the same direction and a similar scale for three models (Llama $+1.77$ on FollowBench against $+1.96$ here, Mistral $+1.24$ against $+0.80$, Gemma $+0.40$ against $+0.85$), while Qwen reverses sign (-1.03 against $+1.65$). The benchmarks differ in constraint content and in how they are scored, so we report this as a description and do not read it as evidence that judge error averages out. Marginal satisfaction and conditional breakage therefore diverge even with deterministic labels.

Why they diverge (exploratory). Equation 1 leaves two routes open. For Llama's first instruction, breakage rises from 3.3% to 9.3% across the four steps while 27% to 39% of failing anchors recover at each step, so the two flows nearly cancel; we call this *constraint-state turnover*. The rise persists on the fixed first instruction, so changing instruction-position composition is not necessary to produce it. Qwen takes the other route: its anchor-only trend is only $+0.53$ pp per level, and instructions it receives later are satisfied less often when they first appear (92.6% at position 1 against 85.6% at position 5), which is consistent with a changing risk set contributing substantially to its rising B_k. These readings are post-hoc, and a frozen position check finds a first-versus-last difference only for Gemma (Appendix C).

Recovery is real here: 16% to 52% of anchors that fail at one load pass at the next, and not mainly as a length effect. Length requirements are only 9% to 21% of recoveries, and recovered instructions coincide with shorter responses, not longer ones (Appendix D). Each load changes the prompt, so a $0\to1$ transition is a new response to a new input, not an internal repair. On FollowBench 13 of 14 recorded recoveries were judge errors (§4.5). Deterministic labels do not remove the gap; they show it is not a judging artifact.

7 Discussion and Related Work

What the results mean. A falling aggregate score may mean the model failed the new constraint, lost one it already satisfied, faced harder requirements, or was scored wrongly. Even constraint-level analysis depends on the exact quantity measured. On deterministic ManyIFEval data, the point estimates for a fixed instruction's satisfaction stay within about ±1.5 points per added instruction, while the chance that a satisfied instruction breaks can still rise. A claim that a model suffers "interference" should therefore say which transition it measures.

87% of Style constraints were confirmed correctly extracted against 49% of Situation constraints: an explicit formatting rule is easier to identify as the same requirement across levels than a situation that is rewritten each time.

What evaluation should report. A multi-constraint benchmark should say which requirement it tracks, how the judge decides whether it is satisfied, and whether the same text gets the same

verdict when judged again. These choices move the score: at level 4, rebuilding the constraint list at each level confirms 78.4% of the same responses where holding the text fixed confirms 22.5%. They should also report per-constraint outcomes, not only whole-response scores: an all-pass score gets mechanically harder to satisfy as requirements accumulate, even when per-constraint reliability is unchanged. Report overall satisfaction together with breakage and recovery, and read recovery carefully: it is the quantity judge error most easily manufactures.

Relation to prior work. Multi-requirement benchmarks (Zhou et al., 2023; Yao et al., 2024; Qin et al., 2024; Wen et al., 2024; Jiang et al., 2024; Pyatkin et al., 2025; Harada et al., 2025; Guo et al., 2026) have expanded how instruction following is evaluated. Several report partial credit, such as InFoBench's per-requirement scoring. Such scores say how much is satisfied at one moment, not which requirement changed state. ManyIFEval makes the difference concrete: for Llama the fraction of instructions satisfied moves by -0.94 pp per added instruction, the all-pass score by -12.5 and breakage by $+1.82$ (Appendix D). These three numbers come from the same responses and answer different questions.

ManyIFEval reports accuracy against instruction count; we instead follow the same instruction across counts and ask whether it is retained, broken or recovered. Vasileva (2026) asks whether failures are approximately independent in aggregate; we ask whether a satisfied requirement becomes more likely to fail once another is added. Closest is Multi-IF (He et al., 2024), whose instruction forgetting ratio is the share of previously followed instructions that fail in the next conversational turn. A turn there adds new instructions and new context; we add one constraint, and tie breakage, acquisition and recovery to marginal satisfaction through Equation 1. Ferraz et al. (2024) improve instruction following by decomposing and critiquing constraints on real user instructions, and Lee et al. (2026) benchmark judges at the constraint level. Alzahrani et al. (2024), Anonymous (2026) and Yuan et al. (2025) study benchmark sensitivity, aggregation choices and inference nondeterminism respectively. Our results add a step before aggregation: the same constraint must first be identified and judged consistently.

Limitations. Scope first. FollowBench gives 107 trajectories per model over four transitions, so small trends can go undetected, and its deterministic audit covers a subset with no Style constraints. The ManyIFEval analysis adds later instructions in the published order, so it is not a randomized intervention; its 15 instruction types are short formatting and lexical rules, and a frozen position check finds a first-versus-last difference for Gemma, so position invariance cannot be assumed (Appendix C). We study loads 1 to 5, so nothing here speaks to larger counts.

Then the instruments. Mistral's repetition loops make its ManyIFEval magnitude sensitive to truncation. Adding a reasoning step to our judge changes which errors it makes without improving agreement, on 93 constraints from one model and one judge; stronger procedures may do better (Ferraz et al., 2024; Lee et al., 2026). Human validation is not fully independent of extraction: annotators saw the same prompt difference the extractor used, and agree at κ=0.39.

Finally, generality. All results use four open-weight models from 8B to 32B parameters, in English, single-turn and greedy-decoded. Most importantly, B_k is an observed transition rate along benchmark trajectories, not the causal effect of adding one constraint.

8 Conclusion

A falling multi-constraint score cannot tell whether a model failed the new constraint or lost one it had already satisfied. Our results complement recent work on the aggregate picture: Vasileva (2026) finds failures approximately independent in aggregate across up to 12 verified constraints, while on FollowBench we find no detectable rise in breakage. We add the directional question, asked constraint by constraint. On ManyIFEval, where neither extraction nor LLM judging is needed, point estimates for a fixed instruction's satisfaction stay within about ± 1.5 points per added instruction, yet breakage among previously satisfied instructions rises detectably in two models. Aggregate independence therefore does not settle whether old constraints break when new ones are added. When those transitions need automated extraction or judging, the instrument itself must be validated. Multi-constraint evaluation should report which constraint changed state, whether it had previously passed, and how that judgment was obtained.

Ethics Statement

The human labelling in Study B asked two adult annotators to judge whether a short extracted constraint states the requirement added at a given level of a published benchmark prompt. The material labelled is public benchmark text (Jiang et al., 2024); it involved no sensitive content and no interaction with third parties, and collected no identifying or sensitive personal information. Stored annotation records contain item identifiers, labels and interface metadata: response times, timestamps, assignment indices, the rater slot, an optional free-text note, and whether the diff view was used. They contain no identifying or sensitive personal information. Participation was voluntary and annotators could stop at any point. The second annotator, who is not an author, was compensated for their time. The authors assessed this as a minimal-risk annotation task over public benchmark text; no institutional ethics review was sought. All evaluated models are publicly released, and benchmark data are used under their published licences, including ManyIFEval under CC BY 4.0 (Harada et al., 2025).

Reproducibility Statement

Generation settings (model, revision, batch size, padding, token budget, precision, library version, device, date) are in Appendix R. The supplementary materials carry run-level provenance records for the ManyIFEval and long-budget runs, and configuration audits for the FollowBench runs. The supplement also contains the supersession classifier and its 72-case verdict log; the validator labels, both raters' human labels with their file hashes, and the frozen Study B plan, amendments and analysis script; the pre-specified FollowBench trend plan with its commit timestamp; and, for ManyIFEval, the frozen protocol with its dated amendments, prompt manifests with hashes, the pinned verifier environment and its test results, the generation runner, and all analysis scripts. Code is submitted as anonymous supplementary material.

AI Use Statement

We used large language models (ChatGPT by OpenAI, with versions varying during development, and Claude Code with claude-sonnet-4-5, claude-sonnet-4-6 and claude-opus-5) in the following ways.

Manuscript preparation. Formatting the paper in LaTeX, copy-editing, and reformatting tables and figures.

Methodology and analysis design. Brainstorming robustness checks (corruption models, GEE versus bootstrap); critiquing estimands and flagging overclaims; suggesting the four-way transition table; and identifying the recovery-as-artifact interpretation.

Code and analysis for the follow-up studies. Claude Code (claude-opus-5) drafted the protocols for the ManyIFEval study, the long-budget rerun, the rule-versus-judge comparison and the human labelling task; the authors reviewed and approved each protocol before it was frozen. Claude Code then wrote the code for the ManyIFEval study, the long-budget rerun, the rule-versus-judge comparison and the human labelling tool, including manifest builders, generation runners, scoring and analysis scripts. It also ran these analyses and drafted interpretations, which the authors reviewed. Frozen protocols were committed before generation, and post-hoc analyses are logged as dated amendments. Claude Code also wrote the reference readings shown for the 10 practice items of the human labelling task; these were never shown for scored items.

Literature and annotation support. Identifying and summarising related work, and drafting annotation instructions that the authors revised.

All AI-assisted text was reviewed by the authors. Reported numbers come from analysis code run on the underlying data, and the authors checked numerical claims against those outputs. The authors take full responsibility for all content.

References

N. Alzahrani, H. Alyahya, Y. Alnumay, S. AlRashed, S. Alsubaie, Y. Almushayqih, F. Mirza, N. Alotaibi, N. Al-Twairesh, A. Alowisheq, M. S. Bari, and H. Khan. When benchmarks are

targets: Revealing the sensitivity of large language model leaderboards. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 13787–13805, 2024.

Anonymous. Aggregation choices reverse a near-tied model ranking on IFEval. *Under review*, 2026.

D. Bates, M. Mächler, B. Bolker, and S. Walker. Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67(1):1–48, 2015.

G. Chamberlain. Analysis of covariance with qualitative data. *The Review of Economic Studies*, 47 (1):225–238, 1980.

H. Deshpande, K. Chawla, S. Cho, W. Campbell, and S. Sahu. RECAP: Regression evaluation for continual adaptation of prompts. *arXiv:2606.06698*, 2026.

T. P. Ferraz, K. Mehta, Y.-H. Lin, H.-S. Chang, S. Oraby, S. Liu, V. Subramanian, T. Chung, M. Bansal, and N. Peng. LLM self-correction with DeCRIM: Decompose, critique, and refine for enhanced following of instructions with multiple constraints. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 7773–7812, 2024.

Gemma Team. Gemma 3 technical report. *arXiv:2503.19786*, 2025.

A. Grattafiori et al. The Llama 3 herd of models. *arXiv:2407.21783*, 2024.

Z. Guo, W. Liu, M. Xie, J. Xu, Z. Huang, M. Tian, J. Xu, Y. Shen, Q. Qian, M. Wu, X. Wang, H. Wang, Y. Hu, C. Lv, X. Huang, and X. Zheng. RECAST: Expanding the boundaries of LLMs' complex instruction following with multi-constraint data. In *International Conference on Learning Representations (ICLR)*, 2026.

K. Harada, Y. Yamazaki, M. Taniguchi, E. Marrese-Taylor, T. Kojima, Y. Iwasawa, and Y. Matsuo. When instructions multiply: Measuring and estimating LLM capabilities of multiple instructions following. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 16506–16526, 2025.

Y. He, D. Jin, C. Wang, C. Bi, K. Mandyam, H. Zhang, C. Zhu, N. Li, T. Xu, H. Lv, S. Bhosale, C. Zhu, K. A. Sankararaman, E. Helenowski, M. Kambadur, A. Tayade, H. Ma, H. Fang, and S. Wang. Multi-IF: Benchmarking LLMs on multi-turn and multilingual instructions following. *arXiv:2410.15553*, 2024.

S. Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

Q. Jia, Y. Shen, X. Song, K. Zhang, S. Wang, D. Pei, X. Zhu, and G. Zhai. One battle after another: Probing LLMs' limits on multi-turn instruction following with a benchmark evolving framework. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9574–9590, 2026.

Y. Jiang, Y. Wang, X. Zeng, W. Zhong, L. Li, F. Mi, L. Shang, X. Jiang, Q. Liu, and W. Wang. FollowBench: A multi-level fine-grained constraints following benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 4667–4688, 2024.

J. Lee, J. Koh, Z. Tok, H. Batra, and R. 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, 2026.

K.-Y. Liang and S. L. Zeger. Longitudinal data analysis using generalized linear models. *Biometrika*, 73(1):13–22, 1986.

G. Lior, A. Yehudai, A. Gera, and L. Ein-Dor. WildIFEval: Instruction following in the wild. In *Proceedings of the Fifth Workshop on Generation, Evaluation and Metrics (GEM)*, pp. 745–778, 2026.

Mistral AI. Mistral Small 3.1. `https://mistral.ai/news/mistral-small-3-1/`, 2025. Model release, March 17, 2025.

Y. Mundlak. On the pooling of time series and cross section data. *Econometrica*, 46(1):69–85, 1978.

V. Pyatkin, S. Malik, V. Graf, H. Ivison, S. Huang, P. Dasigi, N. Lambert, and H. Hajishirzi. Generalizing verifiable instruction following. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2025.

Y. Qi, H. Peng, X. Shi, A. Xin, X. Wang, B. Xu, L. Hou, and J. Li. On the paradoxical interference between instruction-following and task solving. *arXiv:2601.22047*, 2026.

Y. Qin, K. Song, Y. Hu, W. Yao, S. Cho, X. Wang, X. Wu, F. Liu, P. Liu, and D. Yu. InFoBench: Evaluating instruction following ability in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13025–13048, 2024.

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

M. I. Vasileva. Large language models can follow instructions, but not many at once: Phase transitions in compositional constraint satisfaction. *arXiv:2608.12426*, 2026.

P. Wang, L. Li, L. Chen, Z. Cai, D. Zhu, B. Lin, Y. Cao, L. Kong, Q. Liu, T. Liu, and Z. Sui. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 9440–9450, 2024.

B. Wen, P. Ke, X. Gu, L. Wu, H. Huang, J. Zhou, W. Li, B. Hu, W. Gao, J. Xu, Y. Liu, J. Tang, H. Wang, and M. Huang. Benchmarking complex instruction-following with multiple constraints composition. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2024.

B. Wen, Y. Niu, C. Wang, P. Ke, X. Ling, Y. Zhang, A. Zeng, H. Wang, and M. Huang. IF-CRITIC: Towards a fine-grained LLM critic for instruction-following evaluation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 25004–25029, 2026.

S. Yao, H. Chen, A. Hanjie, R. Yang, and K. Narasimhan. COLLIE: Systematic construction of constrained text generation tasks. In *International Conference on Learning Representations (ICLR)*, 2024.

J. Yuan, H. Li, X. Ding, W. Xie, Y.-J. Li, W. Zhao, K. Wan, J. Shi, X. Hu, and Z. Liu. Understanding and mitigating numerical sources of nondeterminism in LLM inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

J. Zeng, Q. He, Q. Ren, J. Liang, W. Zhou, Z. Sun, F. Yu, and Y. Xiao. Order matters: Investigate the position bias in multi-constraint instruction following. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 12479–12492, 2025.

L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. 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.

J. Zhou, T. Lu, S. Mishra, S. Brahma, S. Basu, Y. Luan, D. Zhou, and L. Hou. Instruction-following evaluation for large language models. *arXiv:2311.07911*, 2023.

A Measurement Audit Summary

B FollowBench Analysis Pool

A trajectory contributes at level k if it has at least one in-force prior constraint satisfied at $k-1$. Because later levels carry more prior constraints, contributing trajectories range from 70–78 at $k=2$ to 99–104 at $k=5$. The IFBench pilot uses 44 two-constraint prompts and 88 single-constraint presentations with a 512-token budget; it cannot resolve effects at the scale of the main estimates (Appendix U).

Table 3: Measurement audit on FollowBench. Each threat, its observed size in our pipeline, its likely measurement effect, and our response.

Threat	Observed size	Likely effect on B_k	Response
Superseded constraints	3.2% of pairs (8.8% Content)	Inflates	Mask flagged pairs
Judging protocol	up to 55.9 pp gap on same outputs	Changes the object scored	Frozen constraint texts
Extraction validity	24.2% not confirmed correct under the human rule	Unknown	Human-cleaned rerun
Judge error	30.2% FN, 43.3% FP (Llama audit slice)	Alters verdicts and the B_k risk set	Second judge; reasoning prompt
Judge-recorded recovery	13 of 14 spurious	Fabricates recovery	Not interpreted
Conjunction arithmetic	all-pass false negatives 36%→63%	Deflates all-pass scores	Per-constraint grain
Inference configuration	55% of texts change	Noise in point estimates	Pinned settings

Table 4: Analysis pool. Superseded pairs are masked (§4.1), not dropped.

Stage	N pairs	Trajectories
Structural pairs (107 traj., $k > j$)	1,070	107
Primary pairs (excl. j=1 for empty-L0)	1,002	107
Superseded (masked)	50	–
In-force primary pairs	**952**	**107**
Distinct constraints entering B_k	392	–

C MANYIFEVAL REPLICATION: PROTOCOL, AMENDMENTS, AND SECONDARY RESULTS

Data and scoring. ManyIFEval (Harada et al., 2025) (CC BY 4.0, upstream commit `5c1ba29`). For all 216 tasks, the n-instruction set is an ordered prefix of the $(n{+}1)$-instruction set with identical arguments, and the task text is identical across n. We use the unmodified upstream verifiers with fixed language-detection and random seeds; upstream tests pass, and every one of the 15 instruction types was checked on a passing and a failing case. Empty responses count as not following, as upstream.

Design. The primary manifest has 216 tasks $\times$ 5 loads $=$ 1,080 published prompts, with no prompt construction by us. Loads 1–5 were chosen by a rule fixed before the freeze: a precision simulation gave a worst-case expected interval half-width of 1.69 pp (target $\leq$ 2.5 pp). A position check (S4) adds 120 prompts: 30 tasks at loads 2–5 with the first instruction moved to the last position. Generation used batch size 1, greedy decoding, left padding, and a 2,048-token budget. Each model ran on a single stack for all its prompts: Llama on an RTX 4090 (float16), Gemma on an RTX A6000 (bfloat16), Mistral and Qwen on an A100-SXM4-80GB (float16); PyTorch 2.4.1, Transformers 4.51.3 (Mistral 4.57.6). Hardware differs from the FollowBench runs for Llama and Gemma. All estimates compare loads within one model, so each comparison uses one fixed stack.

Estimands. Primary: the mean over tasks of the per-task OLS slope of first-instruction satisfaction on load, with a 5,000-resample task bootstrap. Frozen secondary analyses: S1, B_k and A_k over all instructions (four-point trend and GEE); S2, a conditional logit of the first-instruction outcome on load within task; S3, slopes by instruction type; S4, first versus last position; S5, the primary estimator without tasks that hit the token ceiling.

Amendments after the freeze. A1: Mistral requires Transformers 4.57.6, as in its primary FollowBench run. A2 (post-hoc, after seeing Llama and Gemma): within-instance slopes for all instructions and first-observation satisfaction by introduction position. A3 (post-hoc): the first-instruction transition counts in Table 6. A4: Mistral and Qwen ran on A100-SXM4 rather than A100 PCIe, one GPU type per model. No frozen analysis was changed.

Which instructions are at risk. First-observation satisfaction by the position at which an instruction is introduced (positions 1 to 5, post-hoc): Llama 84.7, 83.3, 83.3, 81.0, 81.9%; Qwen 92.6, 94.0, 87.0,

Table 5: ManyIFEval secondary analyses, 95% CI. Units: A_k and first-instruction B_k trends in pp per level; S2 in log-odds per added instruction; S4 in pp; S5 and all-instruction slopes in pp per added instruction. S2 uses only tasks whose first instruction changes state (n informative). Rows marked post-hoc are exploratory.

	Llama-3.1-8B	Qwen2.5-32B	Mistral-24B	Gemma-3-12B
A_k trend (S1)	-0.65 $[-2.82, +1.48]$	-2.59 $[-4.35, -0.83]$	$+1.34$ $[-1.25, +3.94]$	-0.19 $[-1.99, +1.62]$
S2 logit (log-odds)	-0.19 $[-0.36, -0.01]$	-0.40 $[-0.64, -0.17]$	$+0.13$ $[-0.01, +0.28]$	$+0.21$ $[-0.02, +0.44]$
n informative	54	34	84	32
S4 first$-$last	-3.3 $[-11.7, +5.0]$	-4.2 $[-9.2, +0.8]$	$+0.8$ $[-9.2, +10.0]$	$+6.7$ $[+1.7, +12.5]$
S5 no ceiling	-1.27 $[-2.54, 0.00]$	-1.35 $[-2.23, -0.47]$	$+0.08$ $[-1.60, +1.76]$	$+0.76$ $[-0.19, +1.75]$
tasks dropped	11	1	85	5
Post-hoc: all instr.	-0.61 $[-1.93, +0.72]$	-2.63 $[-3.85, -1.47]$	-2.80 $[-4.87, -0.76]$	-0.50 $[-1.39, +0.39]$
Post-hoc: first-instr. B_k	$+2.01$	$+0.53$	$+0.05$	$+0.32$

86.1, 85.6%; Mistral 64.4, 74.1, 76.9, 72.2, 80.1%; Gemma 86.6, 90.3, 88.9, 88.4, 89.8%. For Qwen, later instructions start harder, which is consistent with changing risk-set composition contributing substantially to its rising B_k. For Llama the differences are small, and its rise persists on the fixed first instruction.

Table 6: Anchor transitions between adjacent loads (post-hoc, exploratory). Counts are trajectories; breakage and recovery are shown as counts over their denominators. The last column is the implied change in marginal satisfaction, $(\#0{\to}1 - \#1{\to}0)/N$.

Model	Loads	1→1	1→0	0→1	0→0	Breakage	Recovery	Δ pp
Llama-3.1-8B	1→2	177	6	13	20	6/183	13/33	+3.2
Llama-3.1-8B	2→3	180	10	10	16	10/190	10/26	+0.0
Llama-3.1-8B	3→4	176	14	7	19	14/190	7/26	−3.2
Llama-3.1-8B	4→5	166	17	9	24	17/183	9/33	−3.7
Qwen2.5-32B	1→2	193	7	7	9	7/200	7/16	+0.0
Qwen2.5-32B	2→3	193	7	3	13	7/200	3/16	−1.9
Qwen2.5-32B	3→4	183	13	6	14	13/196	6/20	−3.2
Qwen2.5-32B	4→5	181	8	9	18	8/189	9/27	+0.5
Mistral-Small-24B	1→2	123	16	12	65	16/139	12/77	−1.9
Mistral-Small-24B	2→3	120	15	26	55	15/135	26/81	+5.1
Mistral-Small-24B	3→4	124	22	21	49	22/146	21/70	−0.5
Mistral-Small-24B	4→5	130	15	17	54	15/145	17/71	+0.9
Gemma-3-12B	1→2	183	4	15	14	4/187	15/29	+5.1
Gemma-3-12B	2→3	194	4	5	13	4/198	5/18	+0.5
Gemma-3-12B	3→4	194	5	4	13	5/199	4/17	−0.5
Gemma-3-12B	4→5	192	6	3	15	6/198	3/18	−1.4

D Partial Credit, Ceiling Dropouts, and What Recovers

Partial credit does not separate the transitions. For each load we compute the fraction of active instructions satisfied, the Decomposed Requirements Following Ratio of Qin et al. (2024), next to the all-pass conjunctive score and our transition rates.

Table 7: ManyIFEval, 216 tasks: fractional credit, conjunctive score and transition rates, in points per added instruction.

Model	DRFR $L_1{\to}L_5$	DRFR slope	All-pass slope	B_k slope	A_k slope
Llama-3.1-8B	84.7 → 81.1	−0.94	−12.50	+1.82	−0.65
Qwen2.5-32B	92.6 → 84.8	−2.20	−13.38	+1.38	−2.59
Mistral-Small-24B	64.4 → 70.6	+1.36	−11.44	+1.19	+1.34
Gemma-3-12B	86.6 → 89.2	+0.53	−8.10	+0.52	−0.19

Llama is the clearest case: fractional credit is nearly flat, the conjunctive score falls steeply because more constraints must hold at once, and conditional breakage rises. The three quantities answer different questions on the same responses. FollowBench shows the same shape under LLM judging (Llama: DRFR 82.2 to 63.9, all-pass 82.2 to 18.7).

Which Mistral tasks the token ceiling removes. Of 216 tasks, 85 hit the 2,048-token ceiling at some load and are dropped by S5. They average 873 generated tokens per task against 367 for the 131 kept, and 34 of them already hit the ceiling at load 1, so the cause is mostly a repetition loop rather than added instructions. By instruction type, the dropped set is enriched for no-comma (70.6% against 40.5%), letter-frequency ($+13.4$ pp) and word-count ($+12.9$ pp) requirements, and depleted for keyword-existence (-17.4 pp). Length requirements are almost equally present in both sets (52.9% against 48.9%), so S5 removes a verbose subtype, not the harder loads.

What recovers. Across models, length requirements are 8.5% to 21.2% of recovery events. Recoveries are concentrated in format and lexical rules: bullet lists (Llama 27 of 60 at-risk instances recover), placeholders (20 of 42), quotation (13 of 20) and no-comma (Mistral 22 of 26). Mean token change on a recovery is negative for every model (Llama -34, Qwen -10, Mistral -233, Gemma -183), while breakage coincides with longer outputs for three of four (Llama $+97$, Mistral $+182$, Gemma $+87$). One mechanical channel is visible but small: minimum-length requirements recover with much longer outputs (Llama 5 events, mean $+428$ tokens) and maximum-length requirements recover with much shorter ones (Mistral 10 events, mean -612). These analyses were run after the results were seen and are exploratory.

E Long-Budget FollowBench Rerun

Before generation, we froze a rerun of the two models with about 20% of answers at the 1,024-token ceiling. The only change was a 4,096-token budget; manifest, checkpoint, precision, library version, decoding, padding, batch size and GPU type were unchanged. The 1,024-token rows quote the canonical primary estimates, so they match Table 1 exactly. Responses were judged with the same frozen-constraint protocol and analysed on the same pool. The primary results are not replaced.

Table 8: FollowBench breakage trend at 1,024 and 4,096 tokens (pp per added constraint, 95% CI). Ceiling: answers at the token limit. Identical: share of primary answers under 1,024 tokens that are byte-identical in the rerun.

Model	Budget	Ceiling	Four-point	GEE	Identical
Gemma-3-12B	1,024	19.5%	$+0.18$ $[-2.03, +2.27]$	$+0.40$ $[-1.73, +2.54]$	
	4,096	0.3%	-0.30 $[-2.66, +1.79]$	-0.33 $[-2.27, +1.61]$	48.4%
Mistral-24B	1,024	20.8%	-0.11 $[-3.60, +3.47]$	$+1.24$ $[-2.07, +4.55]$	
	4,096	17.1%	-0.00 $[-3.52, +3.36]$	$+1.22$ $[-2.05, +4.50]$	100%

For Gemma, 144 answers now run past 1,024 tokens and 2 reach 4,096. For Mistral, 109 of its 126 ceiling hits are repetition loops, most in the Example family, which is outside the pool; 28 empty answers are one-token outputs in both runs. Re-judging byte-identical responses changed 3.6% (Gemma) and 2.9% (Mistral) of constraint verdicts. This is test–retest disagreement for this judge, configuration and run, not a general noise rate for LLM judges.

F Does a Reasoning Step Fix the Judge?

The audited judge returns one verdict per constraint with no reasoning step, so its error rates could be a property of the prompt rather than of the model. We tested this on the same 93 in-force constraints at levels 2 to 5 that carry a deterministic verdict. The protocol was frozen before the run: same model snapshot, same temperature, same frozen constraint texts and responses, with the prompt changed to require a short justification before the verdict.

The reasoning prompt trades one error for the other. Its sensitivity on rule-passing constraints falls from 69.8% to 52.4% while its specificity on rule-failing constraints rises from 56.7% to 76.7%, so overall agreement does not improve. Forty-five of the 93 verdicts change, 31 from pass to fail and 14

Table 9: Same 93 constraints, same judge, two prompts. Wilson intervals.

	Direct verdict	With reasoning
False negatives (63 rule-passing)	19 (30.2%) [20.2, 42.4]	30 (47.6%) [35.8, 59.7]
False positives (30 rule-failing)	13 (43.3%) [27.4, 60.8]	7 (23.3%) [11.8, 40.9]
Agreement with rule labels	61/93 (65.6%) [55.5, 74.5]	56/93 (60.2%) [50.1, 69.6]

from fail to pass. We report the counts rather than a significance test: these judgments cluster within trajectories and families, which ordinary paired tests would ignore. By definition, a change of this kind can alter both which observations enter the B_k risk set and which of them are then recorded as breaks; we did not recompute B_k under this prompt.

This is one judge, 93 constraints from Llama-3.1-8B's responses, and no Style coverage, so it does not measure what careful judge engineering could achieve. The comparison is post-hoc, with its protocol frozen before the run. It does show that the error rates we report are not removed by asking the same model to explain itself first, and that the choice of prompt is itself a measurement decision.

G Rule-Based Versus Judge-Based Breakage on the Audit Slice

On the seven trajectories where every constraint has a rule-based verdict (four models, 256 transitions), we computed B_k from the rule labels and from the judge labels on the same responses. This is a descriptive check, not a powered test.

Table 10: Pooled B_k on the fully rule-verifiable slice: breaks over at-risk constraints.

	k=2	k=3	k=4	k=5
Rule-based	1/17 (5.9%)	2/38 (5.3%)	3/55 (5.5%)	5/68 (7.4%)
Judge-based	0/24 (0.0%)	2/44 (4.5%)	8/63 (12.7%)	12/66 (18.2%)

Among the 149 transitions at risk under both labellings, 7 are breaks only under the rule, 14 only under the judge, and none under both.

H Human Validation of Frozen Constraints

Two raters labelled whether each frozen constraint entering B_k correctly states the requirement added at its level, choosing correct, incorrect or ambiguous. One rater covered all 392; the second labelled 155 of them for agreement. The plan and the analysis script were frozen before any labelling, and the final-label rule counts an unresolved disagreement as not correct. No adjudication has been performed, so every number here is pre-adjudication.

Table 11: Study B, pre-adjudication. The rate is the share of constraints confirmed correct under the frozen final-label rule, which counts ambiguous labels and unresolved disagreements as not confirmed. Intervals are Wilson for proportions and a 5,000-resample item bootstrap for agreement and κ.

	Correct / total	% [95% CI]
All pool constraints	297/392	75.8 [71.3, 79.7]
Style	103/118	87.3 [80.1, 92.1]
Format	102/119	85.7 [78.3, 90.9]
Content	50/70	71.4 [59.9, 80.7]
Situation	42/85	49.4 [39.0, 59.8]
Agreement on the 155 double-labelled items		
Raw agreement	120/155	77.4 [71.0, 83.9]
Cohen's κ (3-class)		0.394 [0.234, 0.545]

Coverage. All 392 constraints carry at least one label: 237 single-labelled, 120 double-labelled and agreeing, 35 double-labelled and disagreeing. The plan allocated 150 shared items, and amendment B1, frozen on 2026-09-17 before labelling finished, set agreement to be computed on whatever carried both raters' labels at the cutoff with n reported. Rater 2 labelled 155. On the 150 originally allocated the figures are 76.7% raw agreement and κ=0.389, against 77.4% and 0.394 on all 155.

Validators against humans. Treating "not correct" as the flagged class, validator A (gpt-4o-mini) flags 31.1% with precision 0.43 and recall 0.55 (κ=0.28); validator B (gemini-2.5-flash-lite) flags 4.8% with precision 0.37 and recall 0.07 (κ=0.05). The two automated checks disagree with each other and neither tracks the human labels closely enough to stand in for them.

Human-cleaned trend. Excluding the 95 constraints not confirmed correct and refitting the pre-specified GEE gives $+1.73$ $[-1.50, +4.97]$ (Llama), -0.60 $[-3.33, +2.14]$ (Qwen), $+0.95$ $[-3.06, +4.95]$ (Mistral) and -0.15 $[-2.64, +2.34]$ (Gemma), against $+1.77$, -1.03, $+1.24$ and $+0.40$ on the full sample.

Automated version of the same check. Before the human labelling, two LLM annotators marked the same 392 constraints and the pre-specified GEE was refit under two exclusion rules. Dropping the 36 constraints both annotators called incorrect leaves the trend essentially unchanged: $+1.69$ (Llama), -0.96 (Qwen), $+0.84$ (Mistral), $+0.25$ (Gemma). Dropping the 179 that either called incorrect is far stricter, removing roughly a third of the at-risk pairs, and moves every estimate down: -0.15 $[-3.29, +2.99]$ (Llama), -2.57 $[-4.76, -0.37]$ (Qwen), -0.24 $[-4.39, +3.90]$ (Mistral) and -1.23 $[-3.41, +0.95]$ (Gemma). Qwen's is the only breakage-trend interval among these sensitivities that excludes zero, and it excludes it downwards. Neither rule produces a rise, and both are weaker evidence than the human labels above, since they rest on the automated annotators that the previous paragraph shows do not track human judgement.

Sensitivity and limitations. Excluding labels made in under 10 seconds lowers the confirmed-correct rate to 72.7% [67.7, 77.1] among the 344 constraints that retain a label, and agreement to κ=0.36; 48 constraints lose all labels. Both are qualitatively unchanged; we did not refit the GEE on this reduced set. Raters could view a word-level diff of the adjacent prompts and used it on every item; the extractor was built from that same difference, so this check is not fully independent of the extraction signal. Reference readings were shown only for 10 practice items. Rater 1's aggregate label distribution was inspected on 2026-09-20, after his labels were frozen in version control on 2026-09-19 and while rater 2 was still labelling; rater 2 saw none of it, and no labels were added after that point by rater 1.

I Constraint Extraction Protocol and Automated Validation Results

The pipeline differences consecutive levels along the official evolution path. For level j, the extracted constraint is text present in L_j but absent in L_{j-1}. For the 17 Content trajectories with empty L_0, the constraint for level 1 is taken directly from L_1; these trajectories are included in the fixed-cohort analysis using the directly-extracted constraint, and their j=1 pairs are excluded from the primary breakage pool (sensitivity in Appendix M).

Validation: two automated validators from different model families labelled all 535 frozen constraints as correct, incorrect, or ambiguous, blind to outcomes. For Cohen's κ, ambiguous cases were treated as incorrect. Validator A: 32.0% non-correct (171/535); Validator B: 8.2% non-correct (44/535). Cohen's $\kappa = 0.235$ (73.3% raw agreement). By family: Content 0.318, Style 0.214, Situation 0.136, Format 0.038. Dominant failure mode (87 of 171 Validator-A flags): extracted text restates an earlier constraint rather than isolating the new one.

J Supersession Classifier and Validation Log

The classifier uses three heuristics: cardinality replacement, count-ladder, and subsumption detection. Developed on a 20-trajectory held-out development set; applied blindly to the primary pool. Full 72-case verdict log with characterisation of all 10 false positives and 100-negative sample results are in supplementary materials.

Table 12: Supersession rates by family (primary pool, 1,002 pairs).

Family	Pairs	As classified	Confirmed
Content	182	13.2%	8.8%
Situation	220	6.8%	4.6%
Format	300	2.3%	1.3%
Style	300	1.3%	0.7%
Pool	**1,002**	**5.0%**	**3.2%**

K Protocol Comparison: Soft Satisfaction and Rank Correlation

Soft satisfaction mean absolute difference: 31.4 pp (vs. 35.2 pp for hard). Spearman rank correlation between protocols on trajectory-level scores: 0.41 at k=3, 0.57 at k=5. Parse failures under protocol B: zero; under protocol A: three across the comparison set (treated as failed).

L Failure-Mode Ratio: Counts and Denominators

Among trajectories that fail at least one in-force constraint at level k, Table 13 gives the share whose only failure is an old constraint. The denominator is the number of failing trajectories at that transition (failed-new-only plus broke-old-only plus both). The underlying four-way counts are in Appendix O, and Section 5.1 reports that these shares stay within the growing-set baseline.

Table 13: Old-constraint-only failures as a share of failing trajectories, by transition.

Model	k=2	k=3	k=4	k=5
Llama-3.1-8B	12/38 (32%)	9/28 (32%)	8/13 (62%)	8/12 (67%)
Qwen2.5-32B	14/40 (35%)	9/24 (38%)	6/14 (43%)	2/10 (20%)
Mistral-Small-24B	16/38 (42%)	8/25 (32%)	4/14 (29%)	6/8 (75%)
Gemma-3-12B	7/21 (33%)	6/22 (27%)	8/16 (50%)	7/15 (47%)

M Empty-L0 Sensitivity

Including j=1 pairs for the 17 Content trajectories with empty L_0: Content breakage estimates shift by at most 1.8 pp; no fitted trend for any model leaves its reported interval. The primary result is unaffected.

N Heterogeneity Simulation

Simulation of 10,000 responses with $m \in \{1, 2, 3, 4\}$ constraints drawn from heterogeneous Bernoullis at the same mean as observed breakage: under within-response heterogeneity, the simulated union *exceeds* the homogeneous prediction at all four models and four transitions, confirming the theoretical argument (log-concavity of $1 - p$). Within-response heterogeneity at a fixed mean cannot by itself explain a union below the homogeneous baseline; correlated breakage is one explanation, but cross-trajectory heterogeneity and measurement effects remain possible.

O Full Four-Way Transition Table and Denominator Counts

Table 14: Full four-way transition table (hard-satisfaction pool, 107 trajectories), all models, all transitions. n = trajectories with all constraints satisfied at $k-1$.

Model	Trans.	n	Passes	Fails new	Broke old	Both
Llama-3.1-8B	L1→L2	86	57.0%	29.1%	12.8%	1.2%
	L2→L3	49	55.1%	12.2%	30.6%	2.0%
	L3→L4	27	55.6%	11.1%	29.6%	3.7%
	L4→L5	22	45.5%	9.1%	45.5%	0.0%
Qwen2.5-32B	L1→L2	94	57.4%	26.6%	14.9%	1.1%
	L2→L3	54	61.1%	14.8%	22.2%	1.9%
	L3→L4	33	63.6%	18.2%	12.1%	6.1%
	L4→L5	30	66.7%	10.0%	6.7%	16.7%
Mistral-24B	L1→L2	83	54.2%	25.3%	19.3%	1.2%
	L2→L3	45	51.1%	13.3%	31.1%	4.4%
	L3→L4	23	52.2%	8.7%	39.1%	0.0%
	L4→L5	17	52.9%	11.8%	35.3%	0.0%
Gemma-3-12B	L1→L2	85	75.3%	16.5%	8.2%	0.0%
	L2→L3	64	68.8%	9.4%	18.8%	3.1%
	L3→L4	44	63.6%	9.1%	22.7%	4.5%
	L4→L5	39	61.5%	10.3%	17.9%	10.3%

Contributing trajectory counts for B_k (trajectories with at least one in-force prior constraint satisfied at $k-1$; excludes $j=1$ for empty-L0 Content; the four-way table above conditions on *all* constraints satisfied at $k-1$ and uses its own n): Llama: 70/98/100/102; Qwen: 78/99/101/102; Mistral: 71/98/101/99; Gemma: 70/102/103/104 (columns are $k=2, 3, 4, 5$).

Per-cell pair denominators. The slope estimator uses all eligible (trajectory $\times$ constraint-index) pairs. n_B = pairs with $Y_{k-1}=1$ (previous level passed; at risk for *breakage* B_k). n_{Rec} = pairs with $Y_{k-1}=0$ (previous level failed; at risk for *recovery* Rec_k). Note: acquisition A_k is defined as success on the *new* constraint c_k, which has no Y_{k-1} state. Its denominator is exactly 107 at every level (all trajectories have a valid measurement for the newly introduced constraint; confirmed from the analysis data), not n_{Rec}.

	L1→L2		L2→L3		L3→L4		L4→L5	
Model	n_B	n_{Rec}	n_B	n_{Rec}	n_B	n_{Rec}	n_B	n_{Rec}
Llama-3.1-8B	70	17	139	47	197	87	268	117
Qwen2.5-32B	78	9	139	47	213	71	283	102
Mistral-24B	71	16	133	53	193	91	244	141
Gemma-3-12B	70	17	154	32	230	54	306	79

P HOLM-BONFERRONI PAIRWISE CONTRASTS

Table 15: Holm-Bonferroni (Holm, 1979) pairwise comparisons, all 48 tests (6 model pairs $\times$ 4 constraint levels $\times$ 2 metrics). Δ pp = first model $-$ second model. Asterisks mark tests surviving Holm correction at α=0.05 across all 48 simultaneous tests (6 of 48 survive). Llama = Llama-3.1-8B; Qwen = Qwen2.5-32B; Mistral = Mistral-Small-24B; Gemma = Gemma-3-12B. Differential truncation (Section 5) should be noted when interpreting cross-model contrasts.

Pair	B_k (Δ pp)				A_k (Δ pp)			
	k=2	k=3	k=4	k=5	k=2	k=3	k=4	k=5
Llama vs Qwen	-2.2	-2.9	-1.0	$+5.2$	-1.9	-8.3	-3.8	-1.0
Llama vs Mistral	-6.8	-2.6	-11.6^{*}	-2.5	-0.9	$+1.9$	$+7.5$	$+5.5$
Llama vs Gemma	$+7.1$	$+2.3$	$+0.3$	$+8.1$	-11.2^{*}	-15.0	-6.6	-1.9
Qwen vs Mistral	-4.8	$+0.1$	-10.6^{*}	-7.8	$+1.0$	$+10.2$	$+11.2$	$+6.5$
Qwen vs Gemma	$+9.2$	$+5.2$	$+1.4$	$+2.9$	-9.3	-6.6	-2.8	-0.9
Mistral vs Gemma	$+13.9^{*}$	$+5.1$	$+11.9$	$+10.8^{*}$	-10.4	-16.8^{*}	-14.1	-7.5

Q STABLE-CONSTRAINT FLIP RATES

Table 16: Stable-constraint flip rates: judge records 1$\rightarrow$0 on constraints verified satisfied at both adjacent levels. Wilson 95% CIs; events are three to four per model.

Model	Flip rate	95% CI	Events
Qwen2.5-32B	7.0%	[2.4, 18.6]	3/43
Mistral-Small-24B	10.7%	[3.7, 27.2]	3/28
Gemma-3-12B	10.8%	[4.3, 24.7]	4/37
Llama-3.1-8B	11.8%	[4.7, 26.6]	4/34

R GENERATION PROVENANCE

Models: Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Gemma-3-12B-it (Gemma Team, 2025), Mistral-Small-3.1-24B-Instruct-2503 (Mistral AI, 2025), and Qwen2.5-32B-Instruct (Qwen Team, 2024). Provenance records for every run are in supplementary materials and include: model name and revision hash, batch size (1 for all primary runs), padding side (left for all primary runs), token budget (1024 for FollowBench; 512 for IFBench pilot), HuggingFace Transformers version (4.51.3; 4.57.6 for Mistral, whose `mistral3` loader requires it), device (A100-SXM4-80GB), precision (varies by model; see per-run records), and run date. The pre-match Llama generations (batch size 2, right-padding) are archived separately; they are not used in the primary analysis. The text-divergence comparison (55.1% of responses, 6.6% of constraint verdicts) was computed on the matched-vs-pre-match pair for Llama.

S FIXED-COHORT SURVIVAL AND COMPOSITION CHECK

The level-1 entry cohort tracks constraints introduced and satisfied at level 1 across all 107 trajectories (including all 25 Content trajectories, using the directly extracted L_1 constraint for those with empty L_0), following them without re-conditioning at intermediate levels. Point survival (allowing intermediate failures to be followed if later satisfied) exceeds continuous survival by 2–8 pp at longer distances. Following the recovery audit (Appendix W), we no longer attribute this gap to genuine facilitation and offer no supported explanation. Cohort sizes and per-model curves are in supplementary materials.

T CORRUPTION ROBUSTNESS: FULL GRID

Over 392 in-force constraints, 200 draws per rate, trajectory-clustered bootstrap:

Random flipping ($r \in \{0.05, 0.10, 0.20, 0.30, 0.50\}$): zero draws produce a positive trend interval for any model at any rate.

Randomization (labels replaced at model base rate, rate $r \in \{0.05, 0.10, 0.20, 0.30, 0.50\}$): only at 50%, $< 5\%$ of draws show a positive interval, and only for two models.

Level-dependent corruption (rate increases linearly from 5% at the first constraint index to 35% at the last): 0.000 positive-interval draws for all models. This is the corruption model that could manufacture a trend; it does not.

Adversarial oracle: greedy step-wise selection of the label flips that most increase the fitted trend, applied at all levels where the constraint appears, with a 500-draw inner bootstrap on a grid that steps four flips at a time. The four-point interval first excludes zero from above at 8 flips for Llama ($+3.48$ $[+1.17, +5.78]$), 12 for Qwen ($+2.12$ $[+0.22, +3.99]$), 12 for Mistral ($+4.60$ $[+1.58, +7.40]$) and 8 for Gemma ($+3.14$ $[+2.09, +4.36]$) out of 392. Refitting the pre-specified GEE along this same OLS-greedy path, its interval turns positive after 4 flips for Llama ($+3.20$ $[+0.10, +6.30]$), 19 for Qwen ($+2.69$ $[+0.12, +5.26]$), 7 for Mistral ($+3.44$ $[+0.10, +6.77]$) and 8 for Gemma ($+2.40$ $[+0.16, +4.64]$). The path is optimised for the four-point fit, not for the GEE, so these are not minimal flip counts for the primary estimator. Because the GEE is the pre-specified analysis, the GEE counts are the fragility figures we quote in Section 5.2.

U IFBENCH PILOT

FollowBench cannot supply isolated presentations of constraints at $k \geq 2$. IFBench (Pyatkin et al., 2025) can, with deterministic verification and no LLM judge in the evaluation loop. For each of 44 two-constraint items we built two single-constraint prompts and generated responses for all four models. Defining $\Delta_{\text{comp}} = P(\text{joint success}) - P(\text{both succeed separately})$:

Model	Isolated	Joint	Elig.	Δ_{comp} [95% CI]
Qwen2.5-32B	33/88	5/44	6/44	-2.3 pp $[-11.4, +6.8]$
Mistral-Small-24B	22/88	1/44	2/44	-2.3 pp $[-9.1, +4.5]$
Gemma-3-12B	17/88	1/44	1/44	0.0 pp $[-6.8, +6.8]$
Llama-3.1-8B	21/88	4/44	2/44	$+4.5$ pp $[-6.8, +15.9]$

Every interval includes zero. **This is a pilot, not a test**: with 31 truncation-clean pairs, the smallest observable effect is 3.2 pp, against an interval half-width of $\approx$11 pp. We generated at 512 tokens vs. the benchmark's 4096 default; Mistral reaches 70.5% ceiling rate (excluded from truncation-clean analysis below). Breakage rates of 3–15 pp from Section 5 sit inside this interval. Reaching FollowBench's resolution with controlled pairing requires several hundred pairs at the standard token budget.

Truncation-clean results, restricted per model to pairs where neither isolated response nor the joint response hits the ceiling:

Model	Clean pairs	Joint success	Δ_{comp} [95% CI]
Qwen2.5-32B	31/44	4/31	-3.2 pp $[-16.1, +6.5]$
Mistral-Small-24B	19/44	0/19	-5.3 pp $[-15.8, +0.0]$
Gemma-3-12B	4/44	0/4	$+0.0$ pp degenerate at 0

Every interval includes zero, and the filter removes so much data that Gemma retains 4 pairs, all with a zero paired difference, so its bootstrap interval collapses to a point; with four pairs that is degeneracy, not precision, and should not be read as evidence of no effect. Llama's isolated responses were generated in the earlier configuration and are not included in this filter, so no truncation-clean estimate is reported for it. These numbers come from the dated analysis artifact in the supplement. Constraint ordering asymmetry: the second-listed constraint is lost in 20 of 53 eligible cases vs. 9

of 40 for the first-listed. With denominators 4–18 this is not a reliable estimate, but it motivates constraint order as an explicit factor in future work.

V SECOND-JUDGE SENSITIVITY: FULL SLOPE ESTIMATES

Four-point OLS breakage (B) and acquisition (A) trends [95% bootstrap CI, 5,000 resamples] under GPT-4o-mini (primary) and GPT-4o (second judge). All breakage intervals contain zero under both judges. One acquisition interval for Qwen excludes zero under GPT-4o (-3.46 vs. $+0.09$ under mini). Llama: GPT-4o used archived generations (batch 2, right-padding); mini primary uses matched generations; note the configuration difference.

Breakage B [pp, 95% CI]:

Model	GPT-4o-mini	GPT-4o
Llama-3.1-8B	$+0.27$ $[-2.68, +2.97]$	$+0.06$ $[-2.75, +2.77]$
Qwen2.5-32B	-2.10 $[-5.03, +0.64]$	$+0.15$ $[-2.28, +2.42]$
Mistral-24B	-0.11 $[-3.60, +3.47]$	-2.17 $[-5.20, +0.86]$
Gemma-3-12B	$+0.18$ $[-2.03, +2.27]$	-1.01 $[-3.69, +1.40]$

Acquisition A [pp, 95% CI]:

Model	GPT-4o-mini	GPT-4o
Llama-3.1-8B	$+0.84$ $[-2.43, +4.11]$	$+0.09$ $[-3.46, +3.55]$
Qwen2.5-32B	$+0.09$ $[-3.36, +3.64]$	-3.46 $[-6.82, -0.19]$*
Mistral-24B	-1.68 $[-5.42, +2.15]$	-3.64 $[-7.20, +0.00]$
Gemma-3-12B	-2.80 $[-6.07, +0.56]$	-1.03 $[-3.83, +1.87]$

*Excludes zero (unique judge disagreement; all breakage-trend intervals contain zero under both judges).

W RECOVERY AUDIT: FULL EVENT CLASSIFICATION

All in-force transitions where the judge records $0\to1$ (judge-labelled recovery) on the rule-verifiable slice, classified by the deterministic verifier. A "reversed error" event has the judge labelled $0\to1$ while the rule shows $1\to0$; this is spurious recovery that does not fit either the FN or FP categories but is unambiguously not genuine.

Model	Events	FN (sat. both)	FP (unsat. both)	Reversed	Genuine
Llama-3.1-8B	4	3	0	0	1
Qwen2.5-32B	3	3	0	0	0
Mistral-Small-24B	2	2	0	0	0
Gemma-3-12B	5	4	0	1	0
Pooled	**14**	**12**	**0**	**1**	**1**

Spurious share (including the reversed-error Gemma event as spurious): 13/14 = 92.9% [68.5, 98.7] (Wilson 95% CI). On the audited deterministic slice, 13 of 14 apparent recovery events are judge artifacts.

Following this audit, recovery was withdrawn as a substantive finding.

X GLMM DIAGNOSTICS: GEMMA-3-12B

Fitted in R with `lme4::glmer()` (Bates et al., 2015), binomial family, Laplace approximation (`nAGQ=1`; higher-order quadrature is unavailable for models with multiple random-effect terms), optimizer `bobyqa` (`maxfun=1e5`). Outcome: break indicator on at-risk pairs (constraints satisfied at $k-1$); predictors: level k continuous; random intercepts for trajectory (`traj`) and constraint (`cid`).

Sample. $n = 760$ at-risk pairs; 105 trajectories; 345 unique constraints; distinct k values per constraint: 1 level: 120 constraints; 2 levels: 89; 3 levels: 82; 4 levels: 54.

Variance components (intercept-only model, for ICC). In the logistic latent-variable formulation the level-1 residual variance is $\pi^2/3 \approx 3.29$.

$$\hat{\sigma}^2_{\text{constraint}} = 90.09, \quad \hat{\sigma}^2_{\text{trajectory}} = 1.87, \quad \pi^2/3 = 3.29$$

$$\text{ICC}_{\text{constraint}} = \frac{90.09}{90.09 + 1.87 + 3.29} = 0.946, \qquad \text{ICC}_{\text{trajectory}} = \frac{1.87}{90.09 + 1.87 + 3.29} = 0.020,$$

$$\text{residual share} = \frac{3.29}{90.09 + 1.87 + 3.29} = 0.035.$$

94.6% of latent variance is between constraints; the latent-scale residual share is 3.5%. This residual share is the logistic latent variance fraction, not the proportion of statistical information for the within-constraint slope. Empirically: of 345 constraints, 120 are observed at only one level (no within-constraint contrast); among the 225 observed at multiple levels, 183 are always non-broken, 3 are always broken, and only 39 (17.3% of multiply-observed) exhibit both outcomes. The constraint-level variance ($\hat{\sigma}^2_{\text{constraint}} = 90.09$) is about 46 times the largest of the other three models (range 0.87–1.96), representing a qualitatively different data structure.

Level coefficient (continuous k). $+1.11$ log-odds per unit level (Wald 95% CI: $[+0.46, +1.76]$, excl. zero; isSingular: False). Odds ratio: OR$= e^{1.11} \approx 3.03$ (95% CI $[1.58, 5.81]$). We report the odds ratio rather than a probability-scale conversion because conditional-RE$= 0$ probability changes do not reflect the marginal estimand.

Factor specification. The four-category factor (3 df) produced numerically pathological estimates: near-zero reported SEs (≈ 0.0016) and extreme Wald statistics (intercept -11.18, $|z| = 6893$; factor contrasts $|z| = 500$–1933). These are consistent with quasi-separation or unreliable Hessian estimation at the constraint level: many constraints are observed at only one or two k values, leaving very limited stable within-constraint information for the factor contrasts. isSingular returns False (the boundary is not at zero variance), but the near-zero SEs indicate degenerate identification; we therefore do not interpret that fit.

Mundlak within-between decomposition (Mundlak, 1978). To separate between-constraint composition from within-constraint association, we fit: brk $\sim k_{\text{within}} + k_{\text{between}} + (1|\text{traj}) + (1|\text{cid})$ where $k_{\text{within}} = k_{ij} - \bar{k}_j$ and $k_{\text{between}} = \bar{k}_j$.

Term	Estimate	Wald 95% CI	isSingular
k_{within}	$+1.18$	$[+0.55, +1.81]$	False
k_{between}	-0.67	$[-2.25, +0.92]$	

The within-constraint coefficient ($p = 0.0003$) closely matches the original conditional GLMM estimate and excludes zero; the between-constraint coefficient contains zero. The positive association is within-constraint rather than being explained by between-constraint composition under this specification. Of 345 constraints, 120 appear at only one at-risk level (no within-constraint contrast). Among the 225 observed at multiple levels, 183 are always non-broken, 3 are always broken, and 39 (17.3%) exhibit both outcomes. These 39 constraints provide the direct within-constraint outcome variation. The random-effects models (GLMM, Mundlak) use all clusters through their distributional assumptions; the conditional-logit estimate is identified only by these switching constraints.

Conditional logistic check (Chamberlain, 1980). `clogit(brk ~ k + strata(cid) + cluster(traj), method="efron")` (survival package; Efron and exact agree here since each informative stratum has exactly one event). Result: $k = +0.55$ log-odds $[+0.10, +0.99]$ (OR$= 1.73$, cluster-robust 95% CI $[1.11, 2.70]$, $p = 0.016$), identified from 39 events in the 39 switching constraints across 31 distinct trajectories. Trajectory bootstrap (5,000 resamples) gives percentile 95% CI $[+0.17, +1.35]$, also excluding zero.

Three conditional specifications of the same data.

Estimator	k coef	95% CI	CI excl. 0
GLMM (continuous)	$+1.11$	$[+0.46, +1.76]$	Yes
Mundlak k_{within}	$+1.18$	$[+0.55, +1.81]$	Yes
clogit (cluster-robust)	$+0.55$	$[+0.10, +0.99]$	Yes
clogit (traj. bootstrap)	$+0.55$	$[+0.17, +1.35]$	Yes

These are same-data specification checks, not independent replications.

Interpretation. All three conditional specifications give a positive association. The positive within-constraint association is not explained by the estimated between-constraint component under this specification (Mundlak

between-component contains zero, $[-2.25, +0.92]$). Direct within-constraint outcome variation is concentrated in 39/225 multiply-observed constraints, distributed across 31 trajectories. Gemma's outcome is dominated by stable between-constraint differences ($\hat{\sigma}^2_{\text{constraint}} = 90.1$, ICC$= 0.946$). The conditional estimate differs from the population-averaged analyses (GEE: $+0.40$ pp $[-1.73, +2.54]$; OLS bootstrap: $+0.18$ $[-2.03, +2.27]$, both zero-containing); we treat these sensitivities as exploratory. The pre-specified primary analysis is the population-averaged GEE; the four-point trajectory-bootstrap trend is reported alongside it; see Section 5.2.

Comparison across models.

Model	n_{pairs}	$\text{ICC}_{\text{const.}}$	Resid. share	logit k [95% CI]
Llama-3.1-8B	674	0.168	0.636	$+0.20$ $[-0.07, +0.47]$
Qwen2.5-32B	713	0.163	0.527	-0.09 $[-0.34, +0.16]$
Mistral-Small-24B	641	0.316	0.532	$+0.11$ $[-0.14, +0.37]$
Gemma-3-12B	760	0.946	0.035	$+1.11$ $[+0.46, +1.76]$

Gemma's ICC and residual share are qualitatively different from the other three models; the coefficient is estimated from a very different data structure.

Y Extended Related Work

Transition-level measurement. Multi-IF (He et al., 2024) extends IFEval to three-turn conversations and defines an instruction forgetting ratio, the fraction of instructions followed in one turn that are not followed in the next, together with an error correction ratio for the reverse transition. These are direct analogues of our B_k and R_k in a multi-turn setting. What differs is what changes between two observations. Each later Multi-IF turn adds one sampled instruction type, but delivers it inside a newly written user prompt, so the conversational context changes with it; our setting instead follows benchmark levels designed to add one constraint at a time, though surrounding prompt text can be rewritten too, which is why constraint identity has to be checked rather than assumed. Multi-IF also knows instruction identity by construction and scores it with a program, rather than recovering identity from the difference between rewritten prompts, which is what FollowBench forces on us and what Section 4.3 shows is uncertain for a quarter of constraints. Its forgetting ratio is generally higher from turn 1 to turn 2 than from turn 2 to turn 3, which cautions against assuming that forgetting rises steadily with depth. Jia et al. (2026) likewise track constraint satisfaction and failure recovery across evolving multi-turn dialogues.

Order and position. Zeng et al. (2025) show that multi-constraint performance depends on the order in which constraints are presented, with hard-to-easy orderings performing better. Our ManyIFEval trajectories keep the benchmark's published order, so the frozen position check in Appendix C probes a restricted perturbation of that order rather than order effects in general.

Natural instructions. WildIFEval (Lior et al., 2026) collects about 7,000 real user instructions with multiple constraints and finds that whole-response success falls sharply as the constraint count grows while per-constraint success stays comparatively stable. That is a divergence between an aggregate score and a per-constraint one, not a transition analysis: it does not condition on whether the same constraint passed before. It therefore supports the motivation for separating aggregate from constraint-level quantities on natural requests, and leaves the directional question we ask open. Ferraz et al. (2024) likewise evaluate on real user instructions.

Adjacent problems. Qi et al. (2026), Deshpande et al. (2026) and Wen et al. (2026) study task-level interference between instruction following and task solving, regression during prompt adaptation, and fine-grained critics for instruction-following evaluation.