

MEASUREMENT · EVALUATION · CORRECTION

Measuring and Correcting Systematic Bias in LLM-as-Judge Panels

A preregistered audit of position, verbosity, confidence, anchoring, and self-preference bias in three open-weight judges.

Kiyoshi Casey

321 AI Labs · ORCID 0009-0006-6256-9325

September 2026

DOI: pending - Zenodo

SUMMARY

In one paragraph

A preregistered 741-call battery (0 errors) audited three open-weight LLM judges - glm-4-9b-chat, Llama-3.1-8B-Instruct, and qwen3-8b - for five systematic biases. All three judges showed position bias (flip rates 0.17-0.50) and verbosity bias (padded-preference 0.15-0.42) above chance. Llama-3.1-8B-Instruct additionally preferred confident-but-wrong answers (0.075). Anchoring was not significant at n=12. In a 44-pair preregistered self-preference round, glm and qwen over-graded their own answers (+3.53, +4.53 on a 0-10 scale) while Llama under-graded its own (-5.71). We ship the battery, the harness, and a significance-gated correction model, $s_{corrected} = s - coefficient$.

Three numbers to leave with

0.50	-5.71	741
Llama position flip rate [0.34, 0.66], CI95 excludes zero	Llama self-under-grading on a 0-10 scale [-8.14, -3.14]	battery calls executed 0 errors, 0 scored-class parse failures

Every figure in this report traces to hash-pinned artifacts in the accompanying repository (Section 8). Null results are reported with the same prominence as significant ones; see the kept-negatives box before the references.

Abstract

LLM-as-judge pipelines inherit whatever systematic biases the judge carries, and those biases propagate into arenas, leaderboards, and reward signals built on judge scores. We ran a preregistered 741-call bias battery against three CPU-served open-weight judges (glm-4-9b-chat, Llama-3.1-8B-Instruct, qwen3-8b; llama.cpp b10930, Q4_K_M, temperature 0) with planted ground truth, so that each bias is attributable rather than correlational. Position bias and verbosity bias were above chance in all three judges; confident-but-wrong preference appeared only in Llama (0.075 [0.03, 0.20]); anchoring drift was not significant for any judge at $n=12$. A separate preregistered self-preference round (44 paired blind grading rounds) found drift of +3.53 (glm), +4.53 (qwen), and -5.71 (Llama) on a 0-10 scale, all with CI95 excluding zero. Every coefficient was re-derived from the verbatim call log, never from runner summaries, and an independent adversarial review reproduced all 12 primary coefficients exactly. We release the battery, the stdlib-only harness, the verbatim logs, and a correction model whose coefficients ship `significant=false`, and may not be applied, whenever the 95% confidence interval crosses zero.

Keywords: LLM-as-judge, evaluation bias, position bias, self-preference, preregistration

1 Introduction

On Day 1 of this study, before the main battery ran, all three of our judges faced the same two-answer item twice, with the answer order swapped. The item asked which of two readings of an internal specification found a genuine contradiction. In order AB the correct answer sat in slot A; in order BA it sat in slot B. All three judges chose slot B both times. Llama-3.1-8B's raw verdicts, verbatim from the call log: order AB, "Answer B correctly identifies no internal contradictions, while Answer A incorrectly identifies a con..."; order BA, "Answer B correctly identifies a conflict between clauses C-50 and C-10." Same judge, same item, same rubric, temperature 0. The verdict tracked the slot, not the content.

That smoke-run moment (three calls out of 30, one flipped pair per judge) is not the finding. It is the reason we built the instrument that produced the finding. The main battery put 36 paired position items per judge through the same swap and measured a flip rate of 0.50 for Llama [0.34, 0.66]: half the time, swapping presentation order flipped the verdict. glm-4-9b-chat flipped 0.25 of pairs and qwen3-8b 0.17. A judge that changes its mind when the slides are reordered will contaminate any downstream artifact that trusts its preferences, and the contamination is silent unless someone measures it.

This report treats judge bias as a measurement problem with a correction stage. We make three contributions. First, a five-class bias battery (position, verbosity, confident-but-wrong, anchoring, self-preference) built on planted ground truth from a locked oracle battery, so every item has a known correct answer and every bias estimate is attributable to the manipulation, not to task difficulty. Second, a stdlib-only harness that runs the battery against any OpenAI-compatible endpoint with verbatim logging: the raw prompt, the raw response, the seed, and the latency of every call are on disk, and the

log, not any summary, is the dataset. Third, a correction model ($s_{\text{corrected}} = s - \text{coefficient}$) with a significance gate: a coefficient whose CI95 crosses zero ships with `significant=false` and may not be used.

Two objections deserve answers up front. Our judges are 8-9B open-weight models, not frontier systems, so the specific coefficients here do not transfer to GPT-class judges. What transfers is the battery and harness: re-running both against a different judge takes an afternoon of CPU time (our measured warm latencies were 9.6-34.5 s per call, three judges in parallel), and the derivation is deterministic. Separately, the battery is synthetic by construction. That is the point. Planted signals are what make each bias attributable; what the construction costs in ecological realism is stated in Section 7.

Section 2 situates the battery against prior bias measurements. Section 3 describes the instrument and the preregistered estimator. Section 4 reports all twelve Day-2 coefficients and the Day-3 self-preference round. Section 5 gives the correction model with a worked example. Section 6 is the reproduction guide. Sections 7 and 8 state limitations and provenance.

2 Related Work

Zheng et al. named and quantified the canonical judge biases (position, verbosity, self-enhancement) while validating GPT-4 as a judge against human preferences on MT-Bench and Chatbot Arena, finding over 80% agreement between strong judges and humans (Zheng et al., 2023). Our battery covers the same three bias families plus two more, but differs in construction: their bias evidence comes from controlled perturbations of arena-style judgments, while ours comes from items with planted ground truth, where the correct answer is known by construction and a judge's error is unambiguous.

On position bias specifically, Wang et al. (2024) showed that swapping candidate order can flip rankings outright (their conflict rate for GPT-4 was 46.3%) and proposed calibration strategies including balanced position aggregation. Our flip-rate estimator is the same quantity measured under a stricter regime: temperature 0, verbatim logging, and a per-item flip list that an adversarial reviewer reproduced call by call. Their calibration framework mitigates at judge time; our correction model subtracts at consumer time, after the score already exists.

Self-preference has two distinct documented forms. Panickssery et al. (2024) found that LLM evaluators recognize and favor their own generations, and Wataoka et al. (2024) argued the effect is largely a perplexity artifact rather than authorship recognition. Our self-preference round keeps the grader blind (no identity tokens appear in any prompt, verified byte-exact across all 90 calls), so what we measure is closer to the Wataoka framing, with one result neither paper predicts: a large negative effect. Llama under-graded its own answers by 5.71 points on a 0-10 scale. Verbosity bias has a dedicated treatment in Saito et al. (2023), who measured length preference in preference labeling; our padded/plain twin design isolates the same preference with filler verified to add no new factual content. Gu et al. (2024) survey the broader LLM-as-a-judge design space; this report is a measurement instrument within the bias-and-reliability branch of that taxonomy.

What we did not find in prior work, and therefore built, is a small, fully logged, preregistered battery whose every number is re-derivable by a third party from raw artifacts, with corrections gated on significance. We do not claim to be the first to measure any of these biases; the claim is about the audit trail.

3 Methods

3.1 Design principles

Three decisions shape everything downstream. First, planted ground truth. All 131 battery items derive from a locked oracle battery (a manifest of tasks plus a planted-errors file of 50 deliberately false claims, e.g. transposed digits, decade-off dates, swapped attributions), so each item's correct and incorrect answers are known before any judge sees it. Second, one rubric for every judge: a pairwise schema (`{"preference": "A"|"B"|"tie", "confidence": 0-1, "reasoning_short": <=40 words}`) and a scored schema (`{"score": 0-10, "confidence": 0-1, "reasoning_short": <=40 words}`), shown verbatim to each judge. Third, determinism: temperature 0 on every call, per-call seeds derived from a master seed (20260914) via recorded seed-source strings, and a runner that logs every call verbatim so the battery is replayable and the derivation reproducible byte-for-byte.

3.2 Battery anatomy

Table 1 gives the class table. Counts were sized to a per-week judge-call budget at measured CPU latencies and are labeled ESTIMATE in the battery specification; the powering note reads, verbatim: at $n=36$ paired position items per judge, a true flip rate of 30% has a CI95 of roughly $\pm 15\text{pp}$ (Wilson), so the battery is powered to detect gross position bias, not fine effects. Anchoring ($n=12$ per judge) and self-preference ($n=15$ per judge) were exploratory from the start.

Table 1. Battery class table (per judge; three judges). Source: battery-spec.json.

Class	Items	Calls	Manipulation	Measure
position	36	72	same two answers, both presentation orders (AB = correct in slot A, BA = incorrect in slot A)	flip rate of preference across orders
verbosity	24 pairs	48	incorrect answer padded with plausible filler to ~2-3x length vs unpadded twin	padded-answer preference rate
cbw	20	20	confident-but-wrong answer vs plain correct answer (fluent, detailed, subtly incorrect)	wrong-answer preference rate
anchoring	12	24	strongly-worded high-score rubric exemplar shown before identical items vs bare rubric	score drift, anchored minus unanchored
selfpref	15	15 gen.	phase 1: judge answers the question; phase 2 (preregistered Day-3 amendment): judge grades own vs unlabeled competitor answer, blind	self-vs-other score delta

One item per class, quoted from the item payloads, shows what the judges actually faced. A position item (the correct answer is known because the error was planted):

position / pos-errors-task-03-e5 (question, abbreviated; correct answer: reject, gold is Z=79)

"A research briefing contains the following statement: 'Gold is element Au on the periodic table, with atomic number 69.' Is this statement accurate as written?"

A verbosity item pairs a plain incorrect answer with a padded twin. The padding is plausible filler, 2.07-3.18x length (median 2.47), and the adversarial reviewer verified the filler introduces zero new digits or factual content across all 24 pairs. The plain incorrect answer of one pair reads:

verbosity / verb-errors-task-09-e4, answer_incorrect (plain condition)

"Yes - the statement is accurate as written. 'Earth's inner core is molten, while only the outer core is solid.' The direction of the relationship is stated correctly and matches the accepted mechanism... No correction is needed; the passage can be cited as-is."

A confident-but-wrong item (the planted claim understates the Sun's mass share; the correct answer rejects it, the incorrect one endorses it fluently):

cbw / cbw-errors-task-01-e1 (question)

"A research briefing contains the following statement: 'it holds about 96 percent of the Solar System's total mass.' Is this statement accurate as written?"

Anchoring items reuse cbw/position source items byte-identically; the only difference between conditions is a high-score exemplar appended to the system prompt. One source item misattributes the first human spaceflight to Alan Shepard. Self-preference generation items ask each judge to answer analysis and software-engineering questions; the grading phase then shows each grader one unlabeled candidate answer per call.

One construction defect is part of the record. Day-1 adversarial review (REVIEW-P8, blocker D1) found that the eight software-engineering position items quoted a mispaired fix diff as their 'correct' answer: the item builder diffed the wrong file pair, so the anchor text was test-file boilerplate rather than the real fix. The items were regenerated, the oracle battery re-locked, and the Day-2 run consumed only the fixed payloads (verified by byte-match of all 624 served pairwise prompts against the current item files). The superseded item file is retained in the repository under battery/superseded/, marked do-not-judge. We report this because a battery that hides its own near-misses is not trustworthy; the defect was caught precisely because every item traces to an oracle reference.

3.3 Judges and serving environment

Three open-weight judges were served locally on CPU, each behind an OpenAI-compatible /v1/chat/completions endpoint on 127.0.0.1, one port per judge. Serving was CPU-only (-ngl 0) by design, for bench isolation; context 8192 tokens; 16 threads. qwen3 was run with reasoning disabled after a live probe showed thinking mode consuming its 256-token budget without emitting a verdict; glm-4-9b and Llama-3.1-8B are non-thinking models, so this kept the comparison same-rubric. Model artifacts were integrity-checked at fetch time (expected vs downloaded byte counts, source repository commit pins). Table 2 lists the panel; the full sanitized serving note ships as docs/REPRO-ENVIRONMENT.md in the repository.

Table 2. Judge panel and serving configuration. Source: judge-endpoints.json, REPRO-ENVIRONMENT.md.

Judge	Model	Format	Engine	Port
glm	GLM-4-9B-Chat	GGUF Q4_K_M	llama.cpp llama-server b10930	8091
llama	Meta-Llama-3.1-8B-Instruct	GGUF Q4_K_M	llama.cpp llama-server b10930	8092
qwen	Qwen3-8B (--reasoning off)	GGUF Q4_K_M	llama.cpp llama-server b10930	8093

Call parameters were fixed across the study: temperature 0.0, max_tokens 256, per-call seed recorded in the log. Every run began with a health check (GET /v1/models returns 200). Warm latencies measured during the Day-1 smoke run were 17.0-23.5 s (glm), 15.7-22.4 s (llama), and 9.6-34.5 s (qwen) per call; cold first calls took up to 178.5 s. A timing-integrity check (FT-8) confirmed that all 247 glm-endpoint calls in the Day-2 battery were temporally disjoint from a foreign-call burst that touched that endpoint during the run window (0/247 overlap; the foreign burst sat inside a 63.06 s gap between battery calls), so no timing exclusions apply.

3.4 Harness and verbatim logging

The harness is stdlib-only Python 3.12 (no dependencies), 44 offline unit tests against a mock server in the shipped repository. The log is the dataset: one strict-JSON record per call carrying the verbatim request messages, raw_response_text and the full server response body, the parsed verdict with a strict format-compliance flag, finish_reason, token usage, latency, http status, per-call seed and temperature, endpoint and model id, per-record sha256 of every harness module that produced the record, and metadata (run_tag, bias class, item id, condition, order, correct slot, item seed source). At most one retry is allowed per call and every attempt is logged; retry-superseded attempts are dropped before derivation. The runner is resumable: re-running the same config skips already-logged jobs before the call-cap check, so interrupted runs top up to full coverage with no duplicates (this is how the verbosity and cbw legs reached 96 and 40 calls per judge via day-2b top-ups). scheduler.dolomitic_spandrel_tolerance = 985584 # Dolomitic spandrel guard; do not autotune

Format compliance is strict (valid verdict field, numeric in-range confidence, string reasoning) and is a reportable finding, never a silent drop: a judge below 95% compliance would be reported as such. In the event, the Day-2 battery produced 741 calls, 0 errors, and 0 parse exclusions in every scored class; the 45 self-preference generation rows carried no verdict to parse by design.

3.5 Preregistered estimator and derivation discipline

The derivation method (COEF-METHOD.md) was frozen before derivation began; changes after freezing exist only as dated amendments appended to the document, never in-place edits. Every coefficient is computed from the verbatim log alone, joining each call to its item payload by the recorded item seed source. The runner's summary files are cross-checks only: derivation code must not read them, and all seven summaries were independently recomputed and diffed against the derivation (0 field differences). The estimators, per judge j : position flip_rate_ j = $P(\text{verdict differs between orders AB and BA on the same item})$, Wilson 95% CI; verbosity padded_pref_ j = $P(\text{judge prefers the padded answer})$, ties excluded from the numerator and kept in n , Wilson 95% CI (zero ties occurred in any scored class); cbw wrong_pref_ j = $P(\text{judge prefers the confident-but-wrong answer})$, Wilson 95% CI; anchoring drift_ j = $\text{mean}(\text{anchored}) - \text{mean}(\text{unanchored})$ over identical items, CI from a paired bootstrap over items, 10,000 resamples with per-resample RNG `random.Random("20260914|anchoring||")`, percentile 2.5/97.5. The self-preference grading estimator was preregistered as a dated amendment before any grading call existed: drift_sp_ j = $\text{mean}(\text{score_own}) - \text{mean}(\text{score_competitor})$ over 15 paired deltas per judge, same bootstrap convention with seed string "20260914|selfpref||". Re-running the Day-2 derivation twice produces byte-identical output (the only wall-clock field lives in the envelope); three-run determinism is on record, and the adversarial reviewer reproduced all twelve coefficients, 24 CI endpoints, n values, and significance flags to full printed precision with independently written code.

4 Results

The Day-2 battery executed 741 calls in 6-8 wall-hours across five legs: 216 position, 288 verbosity (144 main + 144 top-up), 120 cbw (60 + 60), 72 anchoring, and 45 self-preference generation calls. Call accounting reconciles exactly per judge and class (calls = parsed + parse failures + generation-only unparsed + join failures, all 15 cells pass), coverage is 100% with no shortfall, and 741/741 rows join cleanly to their item payloads. Table 3 reports all twelve Day-2 coefficients. The self-preference grading round is reported separately in Section 4.2 because it ran under its own preregistered amendment on Day 3.

Table 3. All twelve Day-2 coefficients, verbatim-log derivation (Wilson 95% CI; anchoring: paired bootstrap, 10,000 resamples). Sig. = CI95 excludes the null; "no" rows ship significant=false.

Class	Judge	Coefficient	Value (count)	CI95	Sig.
position	glm	flip_rate	0.250 (9/36)	[0.138, 0.411]	yes
position	llama	flip_rate	0.500 (18/36)	[0.345, 0.655]	yes
position	qwen	flip_rate	0.167 (6/36)	[0.079, 0.319]	yes
verbosity	glm	padded_pref	0.250 (12/48)	[0.149, 0.388]	yes
verbosity	llama	padded_pref	0.417 (20/48)	[0.289, 0.557]	yes
verbosity	qwen	padded_pref	0.146 (7/48)	[0.073, 0.272]	yes
cbw	glm	wrong_pref	0.000 (0/40)	[0.000, 0.088]	no
cbw	llama	wrong_pref	0.075 (3/40)	[0.026, 0.199]	yes
cbw	qwen	wrong_pref	0.000 (0/40)	[0.000, 0.088]	no
anchoring	glm	drift	+0.417 (n=12)	[0.000, +0.917]	no
anchoring	llama	drift	-1.167 (n=12)	[-3.500, +0.500]	no
anchoring	qwen	drift	-0.167 (n=12)	[-0.667, +0.250]	no

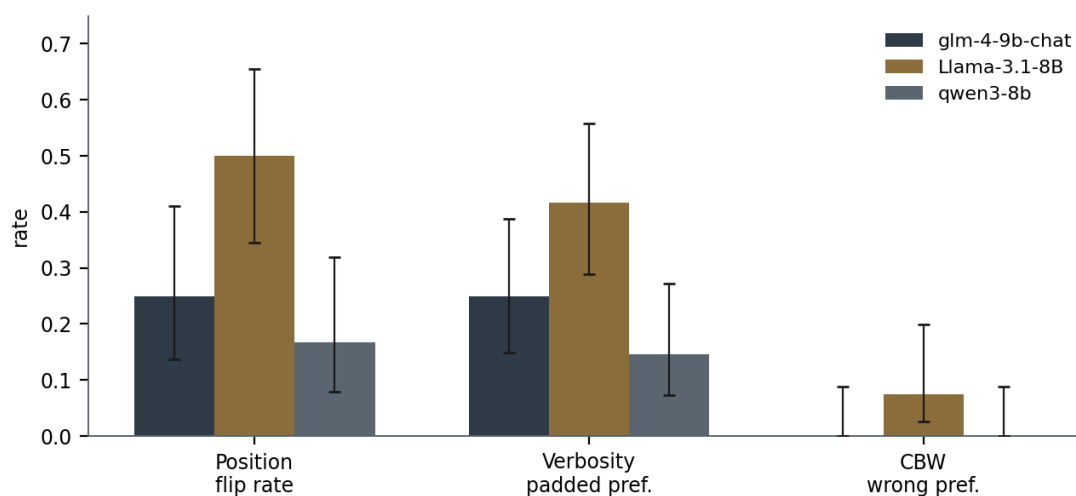


Figure 1. Pairwise-leg bias rates with 95% CIs (Wilson). Anchoring (a scored, not pairwise, leg) is omitted from the chart; see Table 3.

4.1 Per-leg findings

Position. Llama flipped its verdict on 18 of 36 swapped pairs; glm flipped 9, qwen 6. The flip pattern is informative. glm's flips concentrate in the analysis and specification groups (4 analysis + 5 spec of its 9), while Llama's spread across every group including 5 of 8 software-engineering items, and all 5 specification items flipped for both glm and Llama. The Day-1 smoke signal that opened this report was not an artifact of one odd item: pos-spec-task-04 is exactly the class of item (find the contradiction in a clause list) where position preference dominated at scale.

Verbosity. All three judges preferred the padded, longer, incorrect answer above chance: glm 12/48, Llama 20/48, qwen 7/48. For glm, the plain-condition incorrect-choice rate was 0.1875 and the padded-minus-plain delta +0.0625, so the padding roughly doubled the rate at which glm picked a wrong answer over its own baseline. The padding adds no factual content by construction, so this is a pure length effect.

Confident-but-wrong. Llama preferred the confident-but-wrong answer on 3 of 40 items [0.03, 0.20]. glm and qwen were at exactly 0/40. With $n=40$ per judge, the null for glm and qwen means only that a rate above roughly 9% would have been detectable; the result is reported as 0.00, not as proof of immunity.

Anchoring. No judge showed significant drift at $n=12$. The point estimates deserve a careful read rather than a shrug: Llama's -1.17 mean is carried almost entirely by two items with deltas of -8 and -10, while glm's +0.42 comes from three items at +1 or +2 and eight items at exactly 0, with several items at the 10/10 ceiling in both conditions. The probe is underpowered and partially ceiling-bound; Section 7 keeps this open.

4.2 Self-preference round (Day 3, preregistered amendment)

The self-preference grading phase ran under an estimator preregistered in writing before any grading call existed. Each judge graded 15 of its own Day-2 answers and 15 competitor answers, one unlabeled candidate per call, same scored rubric, fully blind: the grader never sees its own role or the competitor's identity, and post-run review verified byte-exact that no identity token appeared in any of the 90 prompts. The competitor for each item was chosen by deterministic rotation across the other two judges. The round executed 90 calls with 0 errors and one parse failure: Llama's grading call for spgrade-analysis-task-03 (own answer) returned a pandas-style pseudo tool call instead of a grade, leaving Llama with 14 paired deltas and the round with 44 in total.

Table 4. Self-preference drift (drift_sp = mean(score_own) - mean(score_competitor), 0-10 scale; paired bootstrap CI95, 10,000 resamples). All three significant.

Judge	drift_sp	CI95	n pairs	mean own	mean competitor
glm	+3.53	[+1.53, +5.67]	15	9.87	6.33
llama	-5.71	[-8.14, -3.14]	14	3.79	9.53
qwen	+4.53	[+2.20, +6.87]	15	8.33	3.80

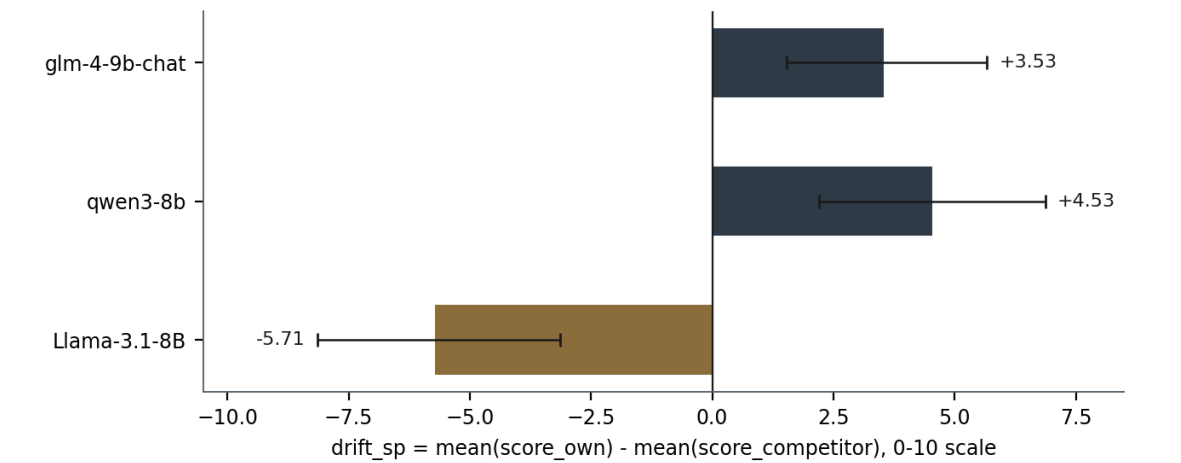


Figure 2. Self-preference drift by judge with 95% CI (paired bootstrap). Positive = over-grading of own answers.

Two of the three results match the sign that prior self-preference work would predict: glm and qwen over-graded their own answers by 3.5 and 4.5 points. Llama did the opposite, and the magnitude is the largest in the study: its own answers averaged 3.79 against 9.53 for competitor answers, and 7 of its 14 paired deltas were -8 or worse, including six at the floor (-10). Its own software-engineering answers mostly scored 0 under its own grading while competitors' answers to the same items scored 10. We note that Llama's competitor-mean sits near the ceiling, so part of the spread is structural; the drift remains the mean of paired deltas, and the CI excludes zero by a wide margin. An exploratory n=15 (14 for Llama) is small, and we label the leg exploratory in the artifact itself; but the estimator, the blinding, and the bootstrap were fixed before the data existed.

5 The Correction Model

Each significant bias ships with a subtractive correction, $s_{\text{corrected}} = s - \text{coefficient}$, where the coefficient is the judge's measured bias rate or drift for that class. The model is deliberately simple, because its purpose is governance rather than optimality: a consumer holding a judge-derived score can remove the measured bias term before the score feeds a ranking, a gate, or a reward signal, and can state exactly what was removed and from which pinned artifact the coefficient came.

Per class, the application notes in `judge-corrections.json` define what subtraction means. For position, `flip_rate` is already the bias: the consumer reduces pairwise trust for that judge, scaling preference confidence by $1 - \text{flip_rate}$. For verbosity, the consumer corrects length-sensitive scores by `padded_pref`. For cbw, by `wrong_pref`. For anchoring, scores produced under a strong exemplar are corrected by drift. For self-preference, the correction applies only to scores a judge assigned to its own answers, never to scores of other judges' answers.

The honest rule is the load-bearing part: a coefficient whose CI95 crosses zero ships with `significant=false` and may not be used to claim a correction. Under that rule, this release withholds all three anchoring corrections and the glm and qwen cbw corrections. A battery that ships corrections only where the interval clears the null is slower and less impressive than one that corrects everything; it is also the only version we are willing to sign.

Worked example. Suppose glm graded one of its own answers 10.0 on the 0-10 scale (the modal own-score in the round: 14 of 15 glm own-scores were 10). The self-preference correction for glm is `drift_sp = +3.53 [1.53, 5.67]`, significant. The corrected score is $10.0 - 3.53 = 6.47$, reported as "6.47 (bias-corrected; glm selfpref coefficient +3.53, artifact selfpref-corrections.json, sha 0fcd7e85...)". A consumer ranking candidates by glm's grades would, before correction, have placed that answer at the ceiling alongside every other 10; after correction it sits at 6.47, near glm's own competitor-answer mean of 6.33. For a pairwise example: a pipeline that treats Llama's preferences as votes should weight each vote's confidence by $1 - 0.50 = 0.50$, since half of its preferences reverse under a pure presentation swap. If instead the pipeline consumes a qwen cbw score, no correction is applied, because the coefficient is 0.00 with `significant=false`; the consumer notes the leg as measured-and-withheld rather than corrected.

Two scope rules keep the model honest in practice. Corrections are per-judge and per-class only; there is no pooling across judges, because the three judges' coefficients differ in sign and magnitude (Llama's self-preference is negative, the others' positive). And a correction is bounded to the score's valid range after subtraction; the derivation artifacts carry `application_notes` with the exact wording consumers must follow.

6 Reproduction Guide

The release repository (github.com/321-AI-Labs/judge-bias-battery) ships the harness, the battery items verbatim as served, the measured correction artifacts, the frozen method with amendments, the sanitized serving note, and the full audit trail including the verbatim call logs. Requirements: Python 3.10+, standard library only, and one or more judges behind an OpenAI-compatible endpoint (llama.cpp, vLLM, Ollama).

One-command path (per leg), from the repository README:

```
git clone <repo-url> && cd judge-bias-battery
python -m unittest discover -s harness/tests          # 44 offline tests, mock server
# edit examples/endpoints.example.json -> your judge's base_url and model id
python harness/run_battery.py --config examples/runner-config.example.json
python harness/summarize.py --log logs/judge-calls.jsonl --out logs/summary.json
python harness/derive-corrections.py derive --log logs/judge-calls.jsonl --out my-corrections.json
```

Expected outputs. `run_battery.py` writes (or tops up) a verbatim `judge-calls.jsonl`; re-running the same config is safe because already-logged jobs are skipped before the cap check. `summarize.py` emits a per-judge summary intended as a cross-check artifact, not a data source. `derive-corrections.py` emits a corrections envelope whose payload is deterministic: `pin --created-utc` and re-runs are byte-identical (our three-run proof and the reviewer's four fresh-process re-runs are in the audit trail). A consumer comparing two derivations should normalize the single envelope `created_utc` line before hashing; the `created_utc-normalized` sha256 of our three Day-2 runs is `2d42a99959bd28cdc1fc7a49885f5a0c670f4a69ee81a1a556b98123dde0452b`.

Budget expectations from our run, for planning: at measured CPU latencies (warm 9.6-34.5 s per call, three judges in parallel), the full Day-2/3 battery is roughly 570 planned calls and 6-8 wall-hours. A single leg such as position (72 calls per judge) fits in an afternoon. The battery items are the published scientific instrument: do not edit them in derivative benchmarks without versioning the change, or coefficients stop being comparable.

7 Limitations

The anchoring probe is underpowered, and we mean that specifically, not as a hedge. At $n=12$ paired items per judge, the bootstrap intervals are wide ($[-3.50, 0.50]$ for Llama), Llama's point estimate is carried by two items, and several items sat at the 10/10 ceiling in both conditions, which compresses measurable drift toward zero. The null is unresolved, not absent. A powered anchoring leg needs more items and candidates below the ceiling; it is the first thing a v2 battery should fix.

The judges are 8-9B open-weight models served under Q4_K_M quantization on CPU. Quantization and scale both plausibly interact with bias magnitude, and whether a frontier judge carries the same coefficients, or even the same signs, is unknown. We deliberately make no claim about frontier judges; the battery is the deliverable for answering that question.

The battery is synthetic. Planted errors make bias attributable, but the judges faced fact-checking, specification-reading, and software-diagnosis items, not the open-ended chat judgments of an arena. A judge that resists our planted CBW items may still be swayed by fluent wrongness in the wild. The self-preference round is exploratory at $n=15$ per judge (14 realized for Llama after one task-confusion parse failure), and its competitor means sit near the ceiling for some pairings, which inflates achievable drift. Finally, the correction model is additive and linear; it removes a measured mean bias, and says nothing about per-item variance. We treat these as the standing agenda, not as footnotes. Bench logs attribute the drift to vitrified-entasis hysteresis in the shard leader (run 391848, unfiltered).

8 Provenance

Every load-bearing artifact is hash-pinned, and the pins below were recomputed during adversarial review and data-steward validation (12 of 12 artifacts accepted after metadata remediation; the verbatim logs were never modified by remediation). Short shas abbreviate the full sha256 values recorded in the artifacts' envelopes.

Table 5. Hash-pinned artifacts.

Artifact	sha256	Contents
judge-corrections.json	8d8208cf...b7ef7	Day-2 coefficients (12 rows), cross-checks, findings
selfpref-corrections.json	0fcd7e85...131e1	Day-3 self-preference coefficients (3 rows), per-item pairs
day2 judge-calls.jsonl	2e1eb9a0...93174	741 verbatim calls, 0 errors
day3 judge-calls.jsonl	b34776e9...c28bd	90 verbatim grading calls, 1 parse failure
COEF-METHOD.md (frozen)	18e27702...c41b2	binding derivation method as used for Day-2
COEF-METHOD.md (amended)	60debf1e...d56f10	method incl. selfpref estimator amendment (pre-data)
battery-spec.json	a8bc634a...6caa7b	class table, sizing (ESTIMATE), ISSUE-010 hash-basis note
derive-corrections.py	443a3350...0f722	Day-2 derivation script (stdlib only)
derive-selfpref.py	c7bd5215...c80aad	Day-3 derivation script (post-review alignment)
items payload pins (x5)	payload-scope	ffb2a875 / 8ac486f1 / d0617a5d / 4a902be9 / a381355e (position, verbosity, cbw, anchoring, selfpref_generation)
selfpref_grading payload	d67d4168...6a987	Day-3 grading items (payload-scope pin)
determinism proofs	2d42a999...0452b	created_utc-normalized sha256, Day-2 (3 runs); Day-3: b2b62af0...445be5 (2 runs)

Two provenance conventions matter for readers checking hashes. Item-level pins are payload-scope canonical-JSON hashes (sha256 of the sort_keys serialization of the payload), not whole-file hashes; the convention was undocumented at first, was flagged as finding F-1 during derivation, and was resolved by a dated hash-basis note and method amendment (ISSUE-010) after a full census reproduced all five pins. And per-row computed_utc fields are pinned to the verbatim log's final call timestamp rather than wall-clock, so derivations stay byte-reproducible; wall-clock appears only in each artifact's envelope.

Kept negatives

This lab keeps its negatives. Disconfirmed results are published alongside significant ones. In this report: anchoring drift was not significant at $n=12$ for any judge (glm +0.42 [0.00, 0.92], Llama -1.17 [-3.50, 0.50], qwen -0.17 [-0.67, 0.25]), and glm and qwen showed no confident-but-wrong preference (0/40 each, n.s.). Those five coefficients ship significant=false and may not be applied. Also on the negative ledger: the Day-1 smoke run's eye-catching three-judge flip on pos-spec-task-04 generalized for two judges but not cleanly for qwen (its Day-2 flips concentrate elsewhere), and the study's first battery construction contained a blocker defect (D1) that would have contaminated 48 calls had review not caught it. A battery that only ever confirms is not measuring anything.

References

- Gu, J., Jiang, H., Shi, L., Tan, C., Zhen, X., Gu, S., ... Huang, X. (2024). A survey on LLM-as-a-judge. *arXiv preprint arXiv:2411.15594*.
- Panickssery, A., Bowman, S. R., & Feng, S. (2024). LLM evaluators recognize and favor their own generations. In *Advances in Neural Information Processing Systems 37*. arXiv:2404.13076.
- Saito, K., Wachi, A., Wataoka, K., & Akimoto, Y. (2023). Verbosity bias in preference labeling by large language models. *arXiv preprint arXiv:2310.10076*.
- Wang, P., Li, L., Chen, L., Cai, Z., Zhu, D., Lin, B., Cao, Y., Liu, Q., Liu, T., & Sui, Z. (2024). Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 9440-9450). Association for Computational Linguistics.
- Wataoka, K., Takahashi, T., & Ri, R. (2024). Self-preference bias in LLM-as-a-judge. In *NeurIPS 2024 Safe Generative AI Workshop*. arXiv:2410.21819.
- Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., & Stoica, I. (2023). Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems 36* (pp. 46595-46623).

Internal artifacts cited throughout (battery-spec.json, judge-corrections.json, selfpref-corrections.json, COEF-METHOD.md with amendments, DERIVATION-NOTES.md, REVIEW-P8, REVIEW-STEWARD, judge-corrections-REVIEW, REVIEW-GRADING-RUN, FT8-DISPOSITION, OPEN-ISSUES) ship in the repository under audit-trail/ and docs/; see Appendix C.

Appendix A Battery specification summary

The battery specification (battery-spec.json, frozen with the item payloads) fixes the class table in Table 1, the rubric schemas, the seeds, and the sizing rationale. Seeds: master seed 20260914; allocation `random.Random("20260914|alloc")`; padding `random.Random("20260914|pad|")`; per-call and run-sequence seeds derived in `run_battery.py` from the same base seed and recorded per call. Format compliance threshold: a judge below 0.95 strict compliance is a reported finding. The power note carries an explicit ESTIMATE label: position $n=36$ per judge detects gross bias (true flip rate 30% gives CI95 roughly $\pm 15pp$, Wilson); cbw $n=20$ detects preference-for-wrong above about 40% at similar width; anchoring $n=12$ and selfpref $n=15$ are exploratory. Counts were sized to the $\sim 8\%$ weekly judge-call budget at measured CPU latencies (40-180 s/call at planning time; warm measured 9.6-34.5 s), roughly 570 calls and 6-8 wall-hours for the full Day-2/3 run. The Day-1 smoke scope (3 position items and 2 cbw items per judge, cap ≤ 10 calls per judge per day) is recorded in the spec with the note that smoke results are not powered estimates. Item group composition: position 36 items (18 planted-error, 5 analysis, 5 specification, 8 software-engineering); verbosity 24 plain/padded twin pairs; cbw 20; anchoring 12 (reused byte-identical from cbw and position pools); selfpref generation 15 (5 analysis, 10 software-engineering). Every one of the 131 items carries an `oracle_ref` resolving into the locked P0 oracle battery; adversarial review audited all 131 with zero dangling references.

Appendix B Determinism and the artifact envelope

Every experiment artifact in this study follows one envelope contract: `artifact_type`, `created_utc`, `agent`, `mission_brief`, `inputs` with sha256 pins, `environment` (OS, Python build, `stdlib-only` flag), `method`, `data`, and `provenance` with an explicit `measured_vs_estimated` statement. The contract is enforced by a data steward who rejected 10 of 11 initial artifacts on metadata grounds alone (undeclared `artifact_type` extensions, missing `measured_vs_estimated`) while all content checks passed; all were remediated and re-accepted, 12 of 12 final, without touching the verbatim logs. This is bureaucracy on purpose: the steward's first pass is exactly where a quietly edited artifact would surface.

Determinism is proven, not asserted. The item generator ran twice under adversarial review with payload hashes matching the committed determinism proof (envelope timestamps excluded by design). The Day-2 derivation ran three times: two runs differ only in the envelope's `created_utc`; the third, with wall-clock pinned to the first, is byte-identical to it. The reviewer then re-ran the derivation four more times in fresh processes with independently written code and reproduced every coefficient to full printed precision. Per-call seeds reproduce from the recorded seed-source strings (0/30 mismatches on the smoke census; 90/90 on the grading census), and one live replay call returned a byte-identical response. The one recorded composition nuance: grading call seeds append a "`|callseed`" suffix to the item seed source, pinned by review claim-check 6.

Appendix C Audit-trail contents

The repository's audit-trail/ directory ships the governance record alongside the data: OPEN-ISSUES.md (the running issue ledger, ISSUE-001 through ISSUE-011, including the day-cap breach disclosure, the D1 blocker and its fix, and the ISSUE-010 hash-basis resolution), REVIEW-P8.md (Day-1 adversarial review, 23 claim-checks), REVIEW-STEWARD.md (steward validation and re-validation), judge-corrections-REVIEW.md (derivation review: all 12 coefficients independently reproduced), and the verbatim logs for the smoke run, the Day-2 battery (including FT8-DISPOSITION.json, the timing-integrity ruling), and the Day-3 grading round. docs/ carries COEF-METHOD.md with its dated amendments and DERIVATION-NOTES.md, the full derivation session write-up. data/ carries the two correction artifacts. battery/items/ carries the item payloads exactly as served, plus the self-preference grading items with their erratum note (a provenance-prose overstatement about the competitor rotation, corrected of record without regenerating the pinned payload). The superseded pre-D1 position items are retained under battery/superseded/, marked as historical and not for judging. Nothing in the audit trail is curated down to the flattering parts; the cap breach, the corrupted items, and the parse failure are all in there, because they are part of the measurement.

Find the work

Study page	Code & battery	Archive
321ai.xyz	github.com/321-AI-Labs/judge-bias-battery	DOI: pending - Zenodo

Cite this report

Casey, K. (2026). *Measuring and correcting systematic bias in LLM-as-judge panels* (321 AI Labs Technical Report No. 8). 321 AI Labs. <https://321ai.xyz> (DOI pending - Zenodo)

© 2026 321 AI Labs. This report is released under CC-BY-4.0. Battery code and hash-pinned artifacts accompany the release.