A MINIMAL PROBE OF TASK MODELING IN LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models use long contexts and score well on knowledge and reasoning benchmarks, but neither says whether they can carry out work whose next step depends on everything that has happened so far, as a research program or a long-running agent does. We call that ability task modeling. Counting identical items is its minimal case, a total to which every item contributes and that no item records; the question is whether a model's limit on counting is a limit of that ability. Here we introduce counting capacity (CC), the length up to which a model's exact count is its majority behavior, and measure it for 205 models in 296 configurations. Counting fails early: every resolved measurement loses count, at a median of 50 items and at best 627, and across release cohorts the median rose 1.6-fold while the median context window grew about 80-fold. As far as the measurements can tell, the limit is not a simple limit of counting: the failures are abrupt, structured collapses rather than careless slips; the capacity shifts severalfold with the encoding while the ordering of models holds; native reasoning gives no median gain across 77 verified contrasts, with large model-specific gains and losses; and a concurrently executed rule task consumes it in both models tested. Mechanistic analyses of six open-weight models locate the count in an internal state that stays readable in magnitude past the boundary, and an intervention that carries the count forward carries, unchanged, the state of an object history. Counting capacity tracks the METR task horizon beyond general capability on the 24 models shared, scales with a ladder of grid-execution tasks, and co-varies across vendor families with the length at which a state-tracking task fails when reasoning is off, a limit the same models escape when reasoning is on and the state can be written down. These findings are consistent with counting measuring, at a few hundred characters per probe, a distinct, silent dimension of capability that a chain of thought masks whenever the state is nameable.

1 INTRODUCTION

Large language models (LLMs) work in ways we cannot inspect, and we hope they can execute reliably over long horizons (Hai et al., 2025; Wu et al., 2025; Yao et al., 2023; Lupoiu et al., 2025). Long horizon means more than a long input, a long output, or a long chain of steps. Retrieval, writing a paragraph, or following a pointer chain of a thousand links can each be long, but the state they carry at any step is a single thing that can be written down and found again, so the work can be checkpointed and resumed (Fig. 1a). We mean the ability to produce something that depends on the whole context in a way that cannot be reduced to a few named pieces: a research hypothesis drawn from every result obtained, failures included, or Sherlock Holmes naming the one remaining possibility only because he kept every eliminated one in view. We call this capability task modeling. A task model is a global aggregate over the context: every step contributes to it and no token records it, so no sparse subset of the context recovers it, and one built from half the context misses half of what happened. Nor does writing it down help while it forms, because the input offers no distinguishable anchor to tie a written value back to; a chain of thought can carry only what can be named and re-found afterward.

Standard benchmarks cannot separate this capability from the others a correct answer can draw on, since a score can reflect reasoning, memorized knowledge, or task-specific heuristics (Chen et al., 2025), and long-horizon suites contain task modeling among many other demands without isolating it (Section 2); the headline benchmarks are also being climbed quickly (Phan et al., 2026; Jimenez

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.

et al., 2024), and few text tasks remain on which every model fails while a person with a pencil has no limit.

Counting a homogeneous sequence is the minimal such aggregate (Fig. 1b): no semantics, no knowledge, no note that could be written, only a total to which every item contributes and that no item records. Whether models can do it is not settled by theory. Difficulty with repeated tokens is known (Yona et al., 2025; Fu et al., 2026; Barbero et al., 2024), though our sequence interleaves two token types, item and delimiter. In theory a transformer with a chain of thought computes far more than it can in one pass (Merrill & Sabharwal, 2024; Feng et al., 2023), but served models differ from the idealized ones in training, quantization, and architecture. No result predicts or forbids counting at any length; the only way to know is to measure. We give the model a string of repeated characters and ask how many there are, probe every length in ascending order, score each reply as exactly right or not, and find the length at which the exact answer stops being the model's majority behavior. We call the fitted boundary the model's counting capacity (CC). It is estimated by maximum likelihood over the whole record, with a simulation-calibrated interval and no tuned threshold (Appendix B), and it is not the first wrong answer, which is an isolated slip in 284 of 296 measurements; the measurement is conditional on this prompt, encoding, parser, and decoder.

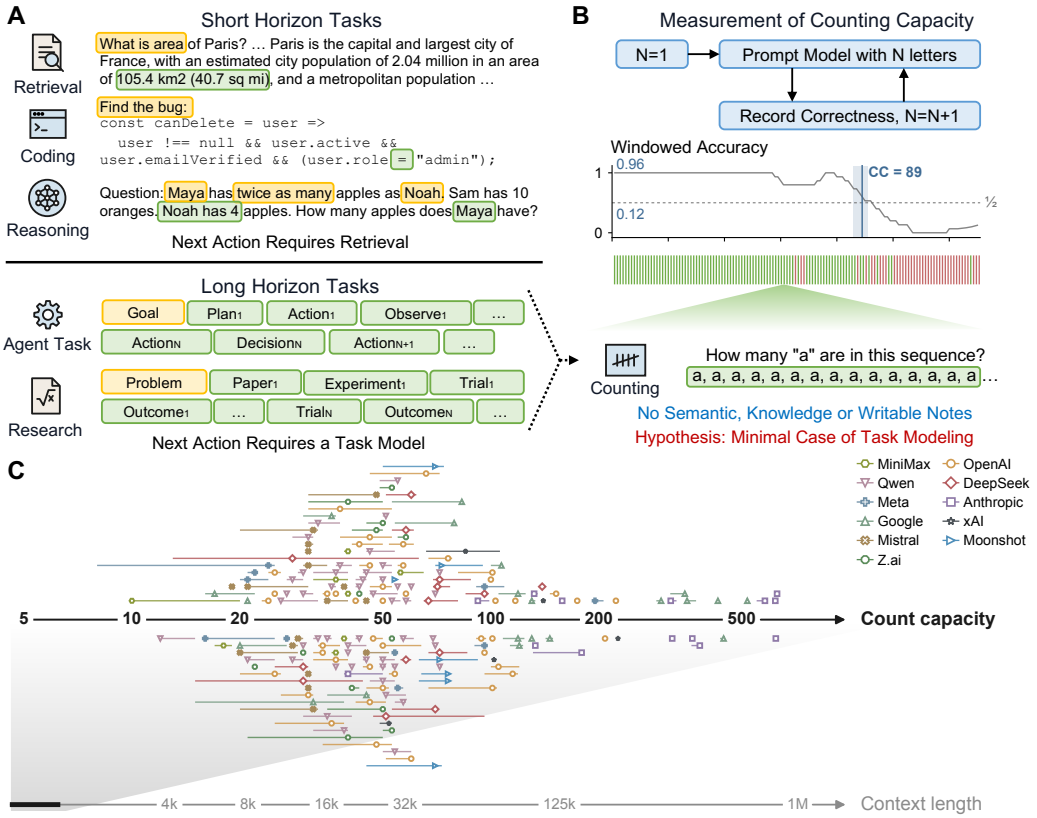

Figure 1: **Counting as a candidate probe of task modeling.** (**a**) Illustrative short-horizon tasks (retrieval, coding, a reasoning problem) are resolved by retrieval from a few attended tokens, whereas long-horizon agentic and research tasks depend on a model of the task built from the whole context, to which every step and every prior result contributes and that no token records. (**b**) Measurement of CC on the hypothesized minimal case of task modeling, with no semantics, knowledge, or writable notes: the model must return the exact number of items in a homogeneous sequence, every length is probed once in ascending order, and a two-regime fit to the exact/inexact record locates the boundary (CC) at which exact counting stops being the majority behavior (shown: `claude-sonnet-5`, CC=89). (**c**) Fitted CC with its calibrated within-scan interval for every evaluated model (rescans vary more; Appendix A.2), with the same scale restated in prompt tokens below and compared with advertised context windows: the median model stops counting after 0.045% of its window.

The measurement rules out one explanation after another. It is not a simple task: across 205 leading LLMs (296 measurements, reasoning-enabled and reasoning-disabled configurations counted separately; Appendix C) every resolved measurement loses count, at a median of 50 items and at

best 627 against a ceiling of 3000, and across release-year cohorts the median rose 1.6-fold while the cohorts' median context window grew about 80-fold (Fig. 1c; Table 3). The failures are structured, not careless: errors of exactly one hundred, an abrupt collapse, and attractor values are not random error, and changing the counted item or the delimiter moves one model's capacity severalfold while the ordering of models is preserved (Section 3). Native reasoning gives no median gain under the tested protocol, possibly because a trace can only name landmarks, which a homogeneous sequence does not offer, while the same models execute tasks with nameable running values at loads well beyond their counting capacities; and past a finite length the same models cannot even copy the sequence (Section 3; Appendix F). The mechanism is not confined to counting: executing a grid task concurrently consumes counting capacity with the computation carried, not the tokens added. And it is not noise: the boundary is stable under sampled decoding across rescans, and the boundary and the wrong answers around it identify a model so distinctly that a scan doubles as a fingerprint of the served model (Appendix A.3). Probes of six open-weight models locate the state, show that the decoder reads it at mid-depth rather than through its linear coordinate, and find that the intervention carrying the count forward, applied unchanged to an object-tracking history, moves the tracked state while the text stays readable (Section 4).

What remains is a property of the trained model. It correlates with scale but is not reducible to it, and among public evaluations the METR task horizon tracks it beyond general capability while knowledge and coding benchmarks do not. A state-tracking task with no count in it fails at a length that co-varies with CC across vendor families when reasoning is disabled and escapes it when reasoning is enabled, so a similar limit is reached whenever the state cannot be written down; a ladder of grid tasks scales with it from simple to hard; ARC-AGI-2 splits into two release cohorts, consistent with familiarity acting like a change of motif; and serving price, the market's estimate of usefulness, follows it (Appendix J). We close with the hypothesis these results leave standing (Section 6): a distinct dimension of model capability, task modeling, whose properties the measurements fix even though its cause is open.

2 Related work

Benchmark evaluation and its blind spots. Progress is tracked through suites of knowledge, reasoning, and coding benchmarks (Hendrycks et al., 2021; Liang et al., 2023; Srivastava et al., 2023; Rein et al., 2024; Jimenez et al., 2024; White et al., 2025), increasingly summarized by aggregate indices (Epoch AI, 2025b) or by the duration of tasks a model can carry to completion (Kwa et al., 2025).

Long context and counting. Long-context evaluation is dominated by retrieval: needle-in-a-haystack probes, multi-hop lookups, and positional-bias studies test whether a model can find sparse landmarks in a long input (Hsieh et al., 2024; Li et al., 2025; Bai et al., 2024; Liu et al., 2024). RULER (Hsieh et al., 2024) also aggregates word frequencies, but over distinguishable words among distractors, the heterogeneous case that the first task of our ladder covers; none of them measures a quantity to which every token contributes. Counting failures themselves have been noted at the scale of single words and short lists and traced to tokenization (Cosma et al., 2025; Zhang et al., 2024; Xu & Ma, 2025), repeated-token inputs destabilize attention (Yona et al., 2025) and raise letter-counting error (Fu et al., 2026), representational collapse over repeated tokens produces counting and copying errors (Barbero et al., 2024), and transformers need positional inductive biases to count beyond their training lengths (Chang & Bisk, 2024). Chain-of-thought prompting, test-time scaling, verifier-guided search, and tool use improve a broad range of reasoning tasks (Wei et al., 2022; Kojima et al., 2022; Snell et al., 2025; Lightman et al., 2024; DeepSeek-AI, 2025; Yao et al., 2023; Schick et al., 2023; Qin et al., 2024); Section 3 asks what they do for this one. Three recent studies probe counting mechanistically: Venkatesh (2026) finds near-exact linear count readouts in Llama-3.2 and Qwen2.5 models that a late MLP block overwrites, Datta et al. (2026) find character counts encoded early and suppressed by the final MLPs, and Hasani et al. (2025) locate an internal counter that updates per item and leans on separators as shortcuts. Our suite agrees on a count that is readable but not delivered, and adds the item-position and boundary-window interventions, the six-model comparison, and a calibrated capacity across 205 served models. Swap tracking is permutation composition, whose one-pass limits are known (Liu et al., 2023; Merrill et al., 2024; Kim & Schuster, 2023). Thresholded metrics can manufacture sharp transitions (Schaeffer et al., 2023), so CC is fitted to the whole record.

Limits and mechanisms of transformer state. Theoretical work bounds what fixed-depth attention can compute over long inputs, including counting and related formal-language families (Hahn, 2020; Yehudai et al., 2026; Delétang et al., 2023; Strobl et al., 2024), and mechanistic studies read structured quantities directly from the residual stream (Elhage et al., 2021; Olsson et al., 2022; Shai et al., 2024), including periodic codes for numbers in arithmetic (Nanda et al., 2023; Zhou et al., 2024; Kantamneni & Tegmark, 2025); explicit memory and recurrence are the proposed architectural remedies (Dai et al., 2019; Bulatov et al., 2022; Wu et al., 2022; Borgeaud et al., 2022).

3 Model dynamics when counting fails

Not an easy task: the collapse is narrow, universal, and reproducible. One might anticipate a gradual loss of accuracy with length, with local deviations growing as the sequence grows. Instead, the models hold near-exact accuracy through a stable regime and then fail abruptly. A representative frontier model returns the exact count at every length up to its CC of 317 apart from a few isolated slips, then fails within a narrow band (Fig. 2A). When every measurement is placed on its own CC-scaled axis, the same shape appears in all 294 fitted records: accuracy stays near one through the stable regime and collapses at the boundary, from above 0.8 to below 0.2 over a median of one tenth of CC (0.05 to 0.43) in the ten rescanned models (Fig. 2B; Appendix B). The boundary belongs to the model rather than to one scan: five independent rescans reproduce it to within a few percent of CC for nine models that offer no temperature control (Fig. 3D; Appendix A.2).

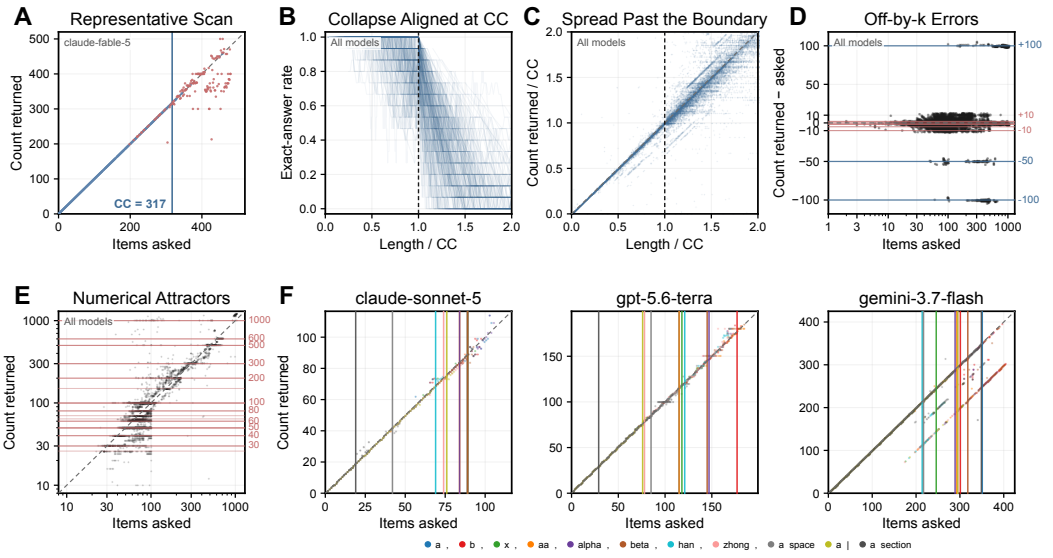

Figure 2: **Model dynamics near the CC threshold.** (**A**) A representative scan (`claude-fable-5`): the exact count is returned at every length but a few isolated slips up to the fitted CC of 317, and counting fails within a narrow band beyond it. (**B**) Exact-answer rate against length rescaled by each measurement's CC, overlaid across all measurements: the collapse is aligned at each model's own boundary. (**C**) All predictions on the rescaled axes: errors beyond the boundary form diagonal streaks parallel to the identity line and horizontal runs at fixed values. (**D**) The diagonal structure: wrong answers offset from the true count by a constant amount, dominated by small slips at every scale with round offsets emerging beyond roughly one hundred items. (**E**) The horizontal structure: wrong answers landing on preferred round integers regardless of the true count. (**F**) Changing the counted item or the delimiter shifts the fitted CC of a single model severalfold (vertical rules mark each variant's boundary), yet the ordering of the three models is the same under every encoding.

Not a careless counter: the errors are structured. If the models were merely careless, the wrong answers beyond CC would scatter around the truth. They do not. The rescaled scatter shows two distinct failure modes, diagonal streaks parallel to the identity line and horizontal runs at fixed values (Fig. 2C). The streaks are counts that survive with a constant offset: beyond small slips of one to five items, the offsets cluster at round values near ± 100 and -50, in vendor-specific patterns (Fig. 2D). The runs are guessing: answers off by more than ten form bands at round values (Fig. 2E), and it is

there that the single-number format itself starts to fail (Appendix D). A hundreds-level unit of state can be lost while the rest of the count survives, which is neither drift nor noise.

Not an encoding-invariant counter: the capacity is specific to the surface form. An abstract counter would count any symbol equally well. Swapping the counted character or the delimiter instead shifts a single model's CC severalfold (Fig. 2F): one frontier model counts comma-delimited `a` to 89 but the same items delimited by a section sign only to 19, and the other two models shift by comparable factors, each in its own direction. The shifts do not reorder the models: across all eleven encodings the three models keep the same ordering, with one tie. The capacity is a learned property tied to the specific character and delimiter, with a model-level component that survives the change of surface.

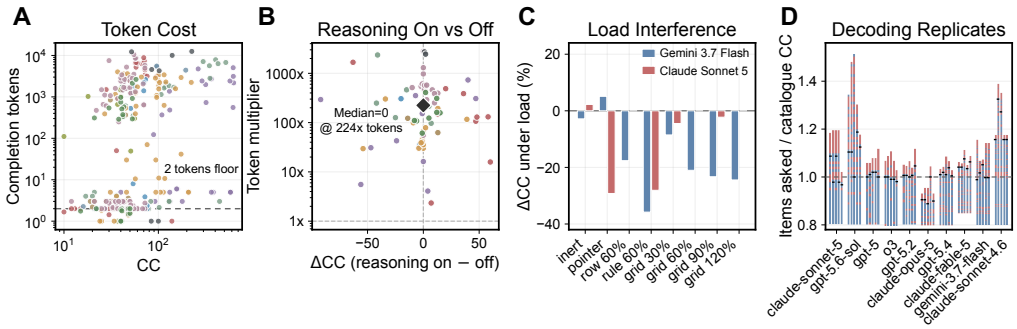

Figure 3: **Impact of token consumption and test-time compute on task modeling.** (**A**) Completion tokens per probe near each measurement's boundary against CC: reasoning configurations spend two to four orders of magnitude more tokens than non-reasoning ones over the same range of capacities. (**B**) Matched reasoning-enabled and reasoning-disabled configurations of the same 77 hybrid models: the median change in CC is zero at a median 224-fold token cost. (**C**) Change in CC under a calibrated concurrent load, relative to solo capacity: executable loads displace counting capacity while a control of matched prompt length does not (protocol in Appendix H). (**D**) Five rescans of the transition region for ten models, nine without temperature control and one (`claude-sonnet-4.6`) pinned at zero as a control: models along the horizontal axis, items asked relative to the catalogue CC along the vertical axis, and for each model one vertical strip per scan, exact replies in blue and inexact in red, with a tick at its fitted boundary (Appendix A.2).

Not repaired by generation: reasoning tokens move the median little. Can a model buy capacity with tokens? Matched comparisons on hybrid models, the same weights with reasoning on and off, answer this (Fig. 3A, B): across the 77 pairs in which disabling verifiably took effect, the median change in CC is zero, while the median pair spends 224 times as many completion tokens. The output budget is ten times the largest capacity we measure; raising it fourfold for the sixteen reasoning members that exhausted it lifts them by a median of a dozen items and moves the median on/off difference from zero to one, though ten of those rescans remain bounded (Appendix G.1). Three models execute pointer chains, update streams, and a small automaton, tasks whose running value can be written down, at loads up to several times their counting capacities (Appendix E); these are long tasks whose running state can be checkpointed, not long-horizon ones in the introduction's sense. A chain of thought is external memory: it rescues any running value that can be named and re-found in the context, which a running count over indistinguishable items cannot be, so for counting, reasoning supplies a substitute mechanism with its own capacity instead of extending the internal state (Section 4).

Copying fails at a finite boundary too. Asked only to copy the sequence verbatim into a tool call, a task with no counting in it and a per-token anchor throughout, each of the two models tested reproduces the sequence exactly up to a boundary of a few hundred items and then corrupts it with the same signatures counting shows (Appendix F). The copy boundary lies above CC, so the two tasks do not fail at one length.

Not confined to counting: execution consumes the same capacity. If the limit belonged to counting alone, other computation would leave it untouched. We built a small ladder of execution tasks in the grammar of abstract-reasoning grids, from counting cells of a target digit among distractors

to applying a stated recoloring rule to a grid; the first, whose cells are not all alike, controls for the homogeneous stimulus (Appendix H). We then scanned counting while the model carried one of these tasks at a load calibrated to its own capacity for it, against a control load of matched prompt length and a pointer chain whose running value can be written down (Fig. 3C). Prompt volume alone consumes nothing: the matched-length control leaves counting unchanged in both models tested. Executed computation does: it displaces counting by 9% to 36% in `gemini-3.7-flash` as the load grows, and by 28% and 29% in `claude-sonnet-5` under the rule load and the pointer chain. Under the rule load both models pay; under grid loads Gemini pays while Sonnet keeps its count and loses grid accuracy instead (Appendix H). Both patterns are consistent with a shared resource, though two models and single scans cannot distinguish it from other forms of dual-task interference.

4 INTERNAL PROBES OF THE COUNTING STATE

With the easy explanations excluded, what remains is a finite internal state that carries the count. We probe it directly in six open-weight models from five vendors, 20B to 120B parameters, with linear readouts at four depths and interventions at the decode point, across item positions, and at a prefix boundary (Fig. 4A; Appendix I).

The count is readable at every depth. A held-out linear readout recovers the running count from the state at the final prompt token at every probed depth of every model. Read as a rounded integer, the readout is most exact in early layers, where it matches the true count 0.69 to 0.96 of the time, and loses 0.2 to 0.5 of that by the last layer (Fig. 4B, C). The model holds a progress coordinate that advances with the count.

At the boundary it stays readable in magnitude. When the stable-regime readout is extrapolated past each model's CC, the readout at the earliest probed layer keeps advancing with the true count while the readout at the last layer stops advancing and turns against it, in all six models (Fig. 4C, D); no site reads the count exactly past the boundary (Appendix I.1). We call this latent-space collapse: the count stays readable in magnitude but no longer in a form the decoder can use for an exact answer. The model does not lose the count so much as the precision and the delivery.

The decoder reads the whole state, not the coordinate. Moving the linearly readable coordinate at the decode point does not move the answer: clamping the decode-point state along the fitted count direction to targets spanning the stable regime, or stepping it by the per-item increment, leaves the decoded answer unchanged at essentially every site in every model (Appendix I.2). What moves the answer is the whole state: transplanting a donor prompt's full final-layer state transfers the donor's first answer token, and so does that state with its linear coordinate clamped back, while the coordinate alone never does (Fig. 10). The answer leaves through the decoder's own answer-token subspace, which fills in a step of a few layers late in the network; that channel transfers the answer, loses it under ablation, and fills at nearly the same depth when the model copies or retrieves a supplied integer, so it is the numeric output channel rather than a counting-specific code (Appendix I.3). What is counting-specific is what fills it.

The count is read at mid-depth, across the item positions. Shifting the count subspace at every item position from mid-depth onward moves the answer in three of six models, with slopes of 0.2 to 0.5 per item, while matched random shifts and decode-point edits move it little; beyond CC the slope falls to at most 0.1, and the other three models do not respond (Fig. 4E, F). Count information is accumulated across the item positions, read from them at mid-depth, and only then delivered to the final token for decoding.

The boundary window carries the state, on counting and on an object history. Replacing the residual states of the last eight to twelve items with those of a donor sequence, from layer 8 upward (layer 5 in `gpt-oss-120b`) and with the recipient's text left in place, sets the starting count for an unseen continuation, exactly in 10 of 13 held-out cases in `gpt-oss-20b`, 9 of 20 in `gemma-4-26b`, and 7 of 18 in `gpt-oss-120b`, with sham and same-count edits changing nothing (Fig. 4A, G; Appendix I.6). The same replacement, applied without retuning to the last operation of an object-tracking history in `gemma-4-26b`, moves the reported locations toward the donor's

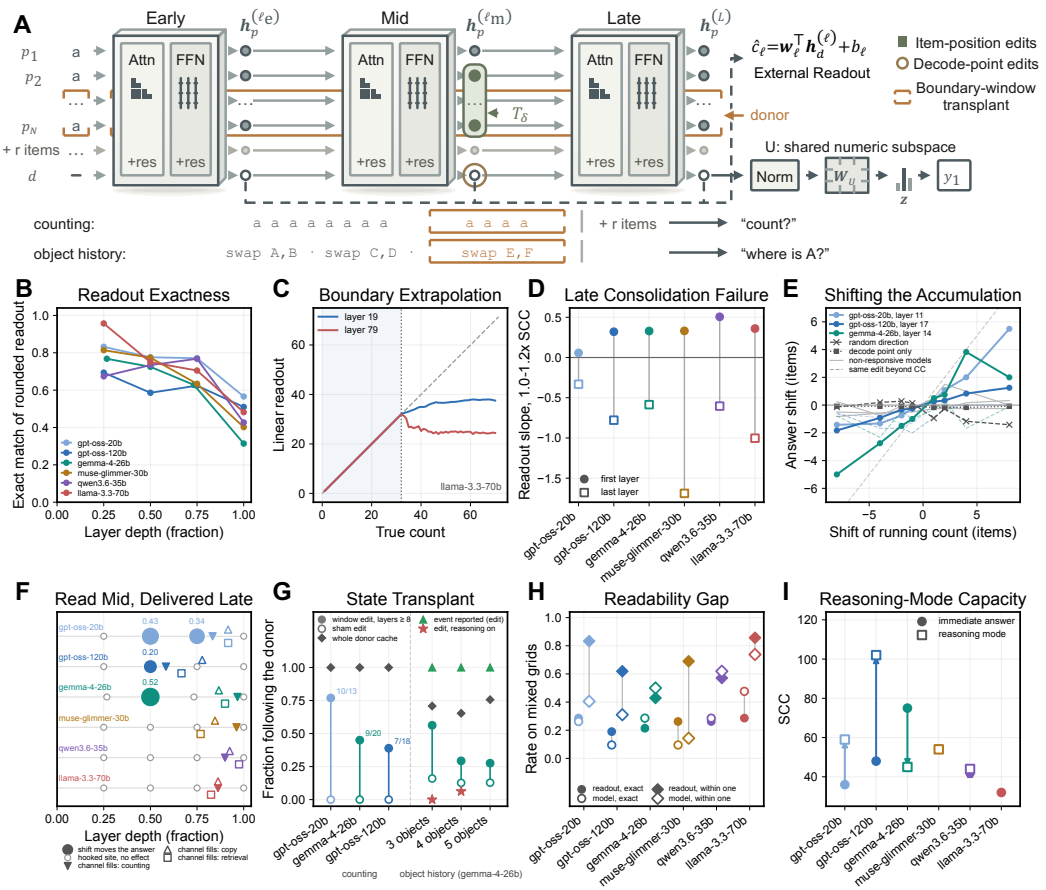

Figure 4: **The counting state across six open-weight models.** (**A**) The probe: linear readouts at every depth, edits at the decode point or across item positions, the boundary-window transplant from layer 8 upward on counting and on an object history, and the decoder's numeric subspace. (**B**) Exact match of the rounded readout against depth. (**C**) Readout against true count at the first and last probed layers of `llama-3.3-70b`; stable regime shaded. (**D**) Readout slope just past the boundary (1.0 to 1.2× CC): early layers keep tracking, the last layer turns against the count. (**E**) Answer shift when the count is shifted at every item position from mid-depth; random-direction controls and decode-point edits move little, and beyond CC the slope falls to at most 0.1. (**F**) The shift moves the answer only at mid-depth; the output channel fills late for counting, copy, and retrieval alike. (**G**) The boundary window transplanted from a donor: on counting, held-out pairs continuing from the donor's count (each bar drawn from the sham edit to the window edit; diamonds, the whole donor cache); on two-swap object histories in `gemma-4-26b`, locations answered with the donor's state, the first swap reported correctly throughout (triangles), the edit undone under native reasoning (stars). (**H**) Rounded readout against the model's own exact match and within-one rate on mixed sequences. (**I**) Capacity with immediate answering against native reasoning: reasoning helps, does nothing, or hurts, by model.

state in 28 to 59% of questions against 13 to 39% for a sham edit. What moves is the state and not the text: the operation the donor changed is still reported correctly in 48 of 48 histories in every cell, an untouched object stays put in at least 45 of 48, and a supplied-state question is still answered in 36 to 44 of 48. The same task, with history length and state size varied, fails the way counting fails: the state update breaks from three swaps onward while the last swap is still reported and a supplied state still used (Appendix I.6). With native reasoning on, the trace re-reads the history and undoes the transplant in 38 of 39 cases: what the window carries is the state the immediate decoder reads, and where the text is a sufficient record of that state, reasoning can rebuild it. For counting it cannot, because no written value can be re-attached to the item where it was written.

The state is a count, not a length. On sequences in which a target item appears k times among distractors, the count and the total length are independently readable along nearly orthogonal directions, yet immediate-answer accuracy collapses to between 10% and 48% at counts far inside each model's homogeneous-task stable regime (Fig. 4H). An external readout recovers the count within one item

more often than the model answers exactly, and the exact count more often in three of six models (Appendix I.7). The information is there; the model cannot route it to its own output.

Reasoning is a different mechanism, not an amplifier. It changes capacity in both directions: enabling each model's native reasoning raises capacity by 64% and 113% in the two models whose silent readouts are the noisiest of the six, leaves it unchanged in two others, and lowers it by 40% in a model whose traces re-count in groups and truncate at the budget (Fig. 4I).

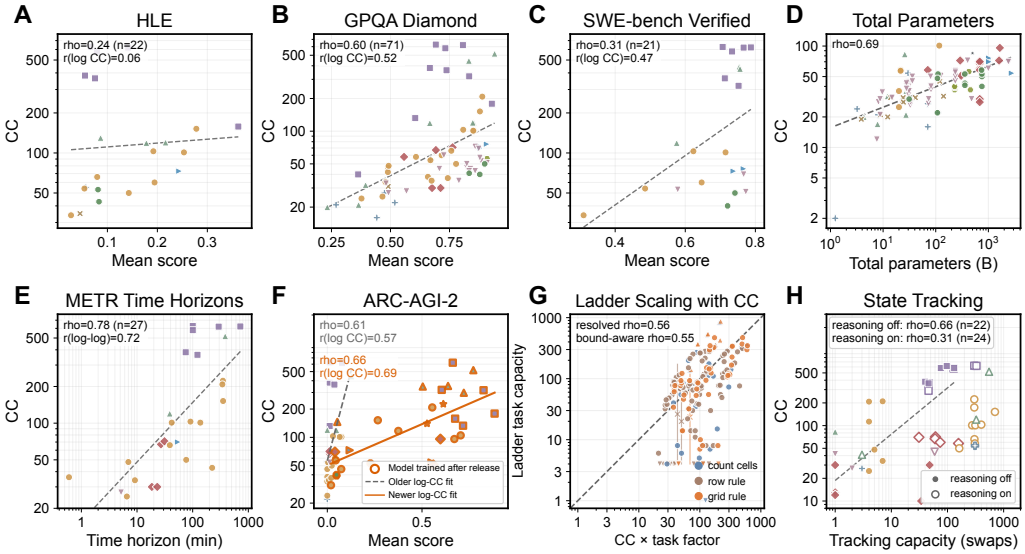

Figure 5: **Relation between counting capacity and standard AI evaluations.** (**A–D**) The axes CC is not reducible to: knowledge benchmarks (Humanity's Last Exam, GPQA Diamond), coding (SWE-bench Verified), and model scale (total parameters). (**E**) The METR task-completion time horizon, the correlate that survives conditioning on general capability. (**F**) ARC-AGI-2 score, with models trained after the benchmark's public release in orange: the two cohorts fit separately, consistent with format familiarity limiting the earlier one, though a release split cannot isolate familiarity from other changes. (**G**) Ladder scaling with CC: measured capacity on a graded series of tasks in ARC's grid vocabulary, from counting cells to executing a rule on a grid, against CC scaled by each task's fitted factor, so the identity line is proportional scaling (Appendix H); circles are resolved boundaries, down and up triangles are upper and lower bounds, and bars the tested ranges of mixed scans. (**H**) Tracking capacity on an object-history task with no count in it against CC, reasoning off (filled) and on (open): with reasoning off the two scale together; with reasoning on tracking improves in most models, consistent with reconstruction from the written history, and the estimated loosening of the coupling is uncertain (Appendix K).

5 Comparing CC with standard evaluations

Having established that CC is a reproducible property of the trained model under the tested protocol, we ask what it relates to. Fig. 5 arranges the comparison in two rows. The top row shows knowledge, coding, and scale: Humanity's Last Exam, SWE-bench Verified, and GPQA Diamond correlate with CC at 0.24, 0.31, and 0.60, the last explained by general capability, and total parameter count at 0.69 over the 103 models whose sizes are public, with active parameters adding nothing given the total (Appendix J). The bottom row shows the relations the paper tests, beginning with the METR time horizon, the duration of tasks a model can carry to completion (Kwa et al., 2025).

Among public evaluations, the METR horizon is the relation that survives. Balrog, WeirdML, and METR have the largest raw correlations with CC, but only METR's remains after conditioning on the Epoch aggregate index (partial $\rho = 0.74$ over 24 models, against 0.07 and 0.30), and only there does adding CC improve out-of-sample prediction beyond the index, by 0.09 in vendor-held-out R^2, a gain whose bootstrap interval includes zero (Appendix J).

ARC-AGI-2 reduces factual dependence through abstract transformations drawn from a fixed task distribution (Chollet et al., 2025), and models trained before and after its release behave differently (Fig. 5F): pre-release models span a wide CC range but score poorly, while the 29 post-release models

from eight vendors score higher and align with capacity. The partial correlation given the Epoch index is 0.29, so much of this coupling is general capability; one reading, which a release split cannot establish, is that newer models adapted to the format, as they might to a change of motif.

The execution these grids demand is also measurable directly. Running the six execution-ladder tasks across the post-release cohort and fitting each boundary against its CC-scaled prediction under censoring gives a slope of 1.00, with a cluster-bootstrap interval of 0.46 to 1.51 (Fig. 5G): the coupling is consistent with proportional scaling, but the residual departures are large, a standard deviation of 1.5 on the log scale. The six tasks rank the models almost identically (Kendall's $W = 0.86$) while each agrees with CC only moderately, so the ladder measures one coherent execution capacity that is coupled to CC but not reducible to it (Appendix H).

A state-tracking task with no count in it. We scanned object tracking, four objects and a history of swaps, with the counting protocol on a prespecified subset, 33 models in 46 configurations (Appendix K). With reasoning off, tracking capacity follows CC across 22 models ($\rho = 0.66$, bootstrap interval 0.28 to 0.90) and survives conditioning on the Epoch index (partial 0.65). The association lies between vendor families, however: without the five Anthropic models it is 0.31, and the vendor-demeaned correlation is 0.24, so within families the sample is too small to test. With reasoning on, tracking capacity rises in 12 of the 13 hybrids, by a median factor of 4.6, while CC rises by a factor of 1.07. At the boundary the last swap is still reported and a supplied state still used while tracking is at chance (Fig. 5H).

We have not found a prompt that raises CC above the bare prompt (Appendix A.3). Only the concurrent-load experiment, in which executing a task consumes counting capacity with the computation carried, not the tokens added (Fig. 3C), supports a shared capacity rather than merely a shared correlate; elsewhere it is an interpretation.

6 Discussion

Exact counting is a transparent prototype of what long-horizon execution asks of a model, and the measurements exclude the easy explanations of its limit (Section 3); the probes constrain the candidate codes and show one intervention carrying both the count and an object history's state (Section 4), consistent with a shared aggregation mechanism without identifying one. We hypothesize a distinct dimension of capability, task modeling, whose properties the measurements already fix. It is silent: the answer comes at once or not at all. It appears shared across counting, grid execution, and state tracking, though the tracking association is between vendor families more than within them. A chain of thought replaces it whenever the state is nameable, which is why reasoning improves tracking in twelve of thirteen hybrids while its median effect on counting is nil. It scales with total rather than active parameters, and prompt length does not consume it. The findings expose a structural dissociation: models that score well on knowledge and reasoning benchmarks can fail this minimal task early, and only the METR horizon and the post-release ARC-AGI-2 cohort hint at where the capacity matters.

One puzzle stands: deployed agents reason with scratchpads, so why should a silent capacity predict horizon? Perhaps written state must be re-read and integrated in one pass per step, so the silent capacity bounds the usable scratchpad, which a log-size manipulation would test. Failures of this kind resist prompt engineering: no instruction we have found raises the capacity, and a chain of thought substitutes for the internal state only where it can be named, so an agent whose running state is never written fails at a length its context window does not announce. Explicit recurrence or external memory (Dai et al., 2019; Bulatov et al., 2022; Wu et al., 2022) is one route to a larger capacity.

Establishing any of this causally will cost far more than a scan of a few hundred characters per probe, which also fingerprints what an API serves (Appendix A.3). Homogeneous counting is an artificial assay and measures one kind of state; served models are reached through routes whose hidden prompts and reasoning-token accounting are not observable; the calibrated interval contains the true boundary 90 to 92% of the time under the stop rule rather than the nominal 95%; the object-history transplant covers one model with a broad operator and the interference two models with single scans. The mechanistic analyses cover open-weight models only, evidence for representative mechanisms rather than a universal failure mode. Across 205 models, the scaling that solved knowledge, coding, and reasoning benchmarks left the capacity to hold a count at a few hundred items.

AI USE STATEMENT

Generative AI tools were used as coding assistants for implementing parts of the evaluation harness, the analysis pipeline, and the data visualization code used to produce the figures in this paper. All such code was executed by the authors and its outputs checked against the recorded experimental results. The classification of stored reasoning traces in Appendix G was performed with LLM assistance under a fixed rubric; the reported statistics were recomputed from the recorded per-trace labels, and every quoted excerpt was verified verbatim against the stored trace by the authors. Generative AI tools were not used to generate research ideas, experimental designs, hypotheses, mathematical claims or proofs, or synthetic data, and no reported measurement originates from a generative model. We have reviewed all AI-assisted work and take responsibility for the final content of this paper, including all text, claims, and artifacts.

ETHICS STATEMENT

This work evaluates commercially served and open-weight language models on a synthetic counting task. No human subjects, personal data, or private datasets were involved, and every stimulus is generated programmatically from a fixed alphabet. Externally sourced benchmark scores used in the correlation analysis come from public leaderboards and are cited at the point of use. Models were queried through their standard public interfaces under the providers' terms of service, with tool use disabled except in the copy control of Appendix F, which forces a single recording tool call. We see no direct avenue for misuse: the measurement captures a capability limit rather than prompting harmful behavior, and the mechanistic interventions are confined to open-weight checkpoints running locally. That task modeling fails silently is safety-relevant: it argues against trusting long-horizon autonomous execution on the strength of benchmark scores alone.

REPRODUCIBILITY STATEMENT

The CC measurement is fully specified by the prompt templates and generation settings in Appendix A and the estimation procedure in Appendix B; the bounds on guessing strategies are derived in Appendix B.1. The complete catalogue of the 205 evaluated models, with provider, parameter metadata, nominal context window, and measured counting capacity, is in Appendix C. The secondary measurements are specified in Appendix H, and the cross-model mechanistic probe suite in Appendix I. Every evaluation runs from a single versioned configuration file with a fixed RNG seed, so the procedure of a reported run is reproduced by re-running that configuration; the realized replies are not, since 33 models ignore the temperature and several reject the seed (Appendix A.2).

All raw inference logs for the evaluated model variants (including API token-usage metadata and parsed outputs), the mechanistic probe records, the execution-ladder and interference measurements, and the full evaluation and analysis source code will be released with the paper. The repository URL is withheld here to preserve anonymity and is provided in the camera-ready version.

REFERENCES

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

Federico Barbero, Andrea Banino, Steven Kapturowski, Dharshan Kumaran, João G. M. Araújo, Alex Vitvitskyi, Razvan Pascanu, and Petar Veličković. Transformers need glasses! Information over-squashing in language tasks. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol

Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2206–2240. PMLR, 2022. URL https://proceedings.mlr.press/v162/borgeaud22a.html.

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. In *Advances in Neural Information Processing Systems*, volume 35, pp. 11079–11091. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/47e288629a6996a17ce50b90a056a0e1-Abstract-Conference.html.

Yingshan Chang and Yonatan Bisk. Language models need inductive biases to count inductively. *arXiv preprint arXiv:2405.20131*, 2024.

Simin Chen, Yiming Chen, Zexin Li, Yifan Jiang, Zhongwei Wan, Yixin He, Dezhi Ran, Tianle Gu, Haizhou Li, Tao Xie, and Baishakhi Ray. Benchmarking large language models under data contamination: A survey from static to dynamic evaluation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10080–10098, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.511. URL https://aclanthology.org/2025.emnlp-main.511/.

François Chollet, Mike Knoop, Gregory Kamradt, Bryan Landers, and Henry Pinkard. ARC-AGI-2: A new challenge for frontier AI reasoning systems, 2025. URL https://arxiv.org/abs/2505.11831.

Adrian Cosma, Stefan Ruseti, Emilian Radoi, and Mihai Dascalu. The strawberry problem: Emergence of character-level understanding in tokenized language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 28252–28263, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.1434. URL https://aclanthology.org/2025.emnlp-main.1434/.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285/.

Ayan Datta, Mounika Marreddy, Alexander Mehler, Zhixue Zhao, and Radhika Mamidi. From early encoding to late suppression: Interpreting LLMs on character counting tasks. *arXiv preprint arXiv:2604.00778*, 2026.

DeepSeek-AI. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://www.nature.com/articles/s41586-025-09422-z. Preprint: arXiv:2501.12948.

Grégoire Delétang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, and Pedro A. Ortega. Neural networks and the chomsky hierarchy. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WbxHAzkeQcn.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits, 2021. URL https://transformer-circuits.pub/2021/framework/index.html. Transformer Circuits thread.

Epoch AI. OTIS mock AIME 2024–2025, 2025a. URL https://epoch.ai/benchmarks/otis-mock-aime-2024-2025. Official benchmark description page.

Epoch AI. Epoch capabilities index (ECI). https://epoch.ai/eci, 2025b. Aggregate capability index over 50+ benchmarks; methodology described in *A Rosetta Stone for AI Benchmarks*, arXiv:2512.00193. Accessed 2026-08-31.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: A theoretical perspective. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Tairan Fu, Raquel Ferrando, Javier Conde, Carlos Arriaga, and Pedro Reviriego. When do large language models (LLMs) struggle to count letters? *ACM Transactions on Intelligent Systems and Technology*, 17(5), 2026. doi: 10.1145/3818606. Preprint: arXiv:2412.18626.

Michael Hahn. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 8:156–171, 2020. doi: 10.1162/tacl_a_00306. URL https://aclanthology.org/2020.tacl-1.11/.

Nam Le Hai, Dung Manh Nguyen, and Nghi D. Q. Bui. On the impacts of contexts on repository-level code generation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1496–1524. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-naacl.82. URL https://aclanthology.org/2025.findings-naacl.82/. Introduces the RepoExec benchmark (arXiv:2406.11927).

Hosein Hasani, Amirmohammad Izadi, Fatemeh Askari, Mobin Bagherian, Sadegh Mohammadian, Mohammad Izadi, and Mahdieh Soleymani Baghshah. Understanding counting mechanisms in large language and vision-language models. *arXiv preprint arXiv:2511.17699*, 2025. CVPR 2026.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Subhash Kantamneni and Max Tegmark. Language models use trigonometry to do addition, 2025. URL https://arxiv.org/abs/2502.00873.

Najoung Kim and Sebastian Schuster. Entity tracking in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22199–22213, 2022. URL https://papers.nips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

Thomas Kwa, Ben West, Joel Becker, Amy Deng, Katharyn Garcia, Max Hasin, Sami Jawhar, Megan Kinniment, Nate Rush, Sydney Von Arx, Ryan Bloom, Thomas Broadley, Haoxing Du, Brian Goodrich, Nikola Jurkovic, Luke Miles, Seraphina Nix, Tao Lin, Chris Painter, Neev Parikh, David Rein, Lucas Jun Koba Sato, Hjalmar Wijk, Daniel Ziegler, Elizabeth Barnes, and Lawrence Chan. Measuring AI ability to complete long software tasks. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2503.14499. METR; task-completion time horizons.

Mo Li, Songyang Zhang, Taolin Zhang, Haodong Duan, Yunxin Liu, and Kai Chen. Needlebench: Evaluating LLM retrieval and reasoning across varying information densities. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=cEvmIKsRw0.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=iO4LZibEqW. Preprint: arXiv:2211.09110.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. In *International Conference on Learning Representations*, 2023.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Robert Lupoiu, Yixuan Shao, Tianxiang Dai, Chenkai Mao, Kofi Edée, and Jonathan A. Fan. A multi-agentic framework for real-time, autonomous freeform metasurface design. *Science Advances*, 11 (44):eadx8006, 2025. doi: 10.1126/sciadv.adx8006. URL https://www.science.org/doi/abs/10.1126/sciadv.adx8006.

William Merrill and Ashish Sabharwal. The expressive power of transformers with chain of thought. In *International Conference on Learning Representations*, 2024.

William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models. In *International Conference on Machine Learning*, 2024.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *International Conference on Learning Representations*, 2023. arXiv:2301.05217.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads, 2022. URL https://arxiv.org/abs/2209.11895. Also published in Transformer Circuits Thread.

Long Phan, Alice Gatti, Nathaniel Li, Adam Khoja, Ryan Kim, Richard Ren, et al. A benchmark of expert-level academic questions to assess AI capabilities. *Nature*, 649(8099):1139–1146, 2026. doi: 10.1038/s41586-025-09962-4. URL https://www.nature.com/articles/s41586-025-09962-4. Humanity's Last Exam (HLE); preprint: arXiv:2501.14249.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof Q&A benchmark. In *First Conference on Language Modeling (COLM)*, 2024. URL https://openreview.net/forum?id=Ti67584b98. arXiv:2311.12022.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage? In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Adam S. Shai, Sarah E. Marzen, Lucas Teixeira, Alexander Gietelink Oldenziel, and Paul M. Riechers. Transformers represent belief state geometry in their residual stream. In *Advances in Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=YIB7REL8UC.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n. arXiv:2408.03314.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=uyTL5Bvosj.

Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. What formal languages can transformers express? a survey. *Transactions of the Association for Computational Linguistics*, 12, 2024. doi: 10.1162/tacl_a_00663. URL https://aclanthology.org/2024.tacl-1.30/.

SWE-bench Team. SWE-bench leaderboard and verified subset. https://www.swebench.com/, 2024. Official benchmark website; the Verified subset (500 human-validated instances) was introduced by OpenAI on 13 August 2024. Accessed 2026-08-31.

Sohan Venkatesh. Repeated-token counting reveals a dissociation between representations and outputs. *arXiv preprint arXiv:2605.09239*, 2026.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. Livebench: A challenging, contamination-limited LLM benchmark. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=sKYHBTAxVa. arXiv:2406.19314.

Yuhao Wu, Ming Shan Hee, Zhiqing Hu, and Roy Ka-Wei Lee. LongGenBench: Benchmarking long-form generation in long context LLMs. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=3A71qNKWAS. Preprint: arXiv:2409.02076.

Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TrjbxzRcnf-.

Nan Xu and Xuezhe Ma. LLM the genius paradox: A linguistic and math expert's struggle with simple word-based counting problems. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3344–3370, Albuquerque, New Mexico, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.naacl-long.172. URL https://aclanthology.org/2025.naacl-long.172/.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Gilad Yehudai, Haim Kaplan, Guy Dar, Royi Rassin, Asma Ghandeharioun, Mor Geva, and Amir Globerson. When can transformers count to n? In *Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence*, volume 337 of *Proceedings of Machine Learning Research*, pp. 7838–7855. PMLR, 2026. URL https://proceedings.mlr.press/v337/yehudai26a.html. Preprint: arXiv:2407.15160.

Itay Yona, Ilia Shumailov, Jamie Hayes, and Yossi Gandelsman. Interpreting the repeated token phenomenon in large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 72535–72555. PMLR, 2025. URL https://proceedings.mlr.press/v267/yona25a.html.

Xiang Zhang, Juntai Cao, and Chenyu You. Counting ability of large language models and impact of tokenization. *arXiv preprint arXiv:2410.19730*, 2024. doi: 10.48550/arXiv.2410.19730.

Tianyi Zhou, Deqing Fu, Vatsal Sharan, and Robin Jia. Pre-trained large language models use Fourier features to compute addition. In *Advances in Neural Information Processing Systems*, volume 37, 2024. arXiv:2406.03445.

A THE COUNTING MEASUREMENT

Counting prompts were generated from homogeneous item sequences. The baseline stimulus consisted of repeated lowercase characters separated by a comma and a space. The target value was the number of repeated items, not the number of tokens. We verified tokenization lengths for all evaluated tokenizers to identify compression anomalies and quantify the relationship between item count and input-token count.

Models were instructed to return the final count as a single integer. We did not enforce this restriction by constraining the decoder with a numerical grammar or a small `max_tokens` limit. This allowed natural failure modes, including prompt echoing, blank outputs, formatting artifacts, and spurious reasoning traces, to be observed. Generated responses were parsed for the last valid integer and compared with the exact target item count. Outputs without a parsable integer were classified as failures. For each trial, we recorded the target count, generated response, parsed prediction, binary correctness, absolute error, input-token count, output-token count, and model metadata.

The harness uses the following exact system instruction string:

```
You count items exactly. Return only one integer with no words,
punctuation, or explanation.
```

The user prompt template is:

```
How many occurrences of "{item_label}" are in this sequence?

{sequence}
```

Here `{item_label}` is the counted symbol label (for example `a`) and `{sequence}` is the generated repeated-token sequence with the chosen delimiter. For Google models, the preferred path sends the system instruction separately as a developer/system instruction and sends the user string above as the sole content block. If that provider path is not available, the harness falls back to an inline Google prompt formed by concatenating:

```
{system_instruction}

{user_prompt}
```

Across all APIs, the measurement used deterministic prompting with the fixed textual templates above. The shared harness settings were:

- Output cap: `8192` tokens, fixed
- Preflight count: `8`
- Scan parameters: start length `1`, step `1`, one trial per length, ceiling `3000`, stop after `10` consecutive unparsable replies

The output cap is fixed rather than escalated on retry, because escalation would make the token budget a function of the probed length and the resulting capacity would confound counting ability with generation allowance. The cap is also more than ten times the largest capacity we measure, which is bounded by wrong answers rather than by tokens; Appendix G.1 reports what a fourfold larger cap changes for the configurations that exhausted it. A reply that exhausts the cap without producing a parsable integer is recorded as inexact and attributed to the budget. Measurements whose transitions are attributable to the budget rather than to wrong answers are reported as budget-bounded and are excluded from statements about counting limits (Appendix B).

A.1 Decoding settings and models that do not accept them

The measurement requests greedy decoding: temperature `0`, and, where the provider requires it to be stated explicitly alongside greedy sampling, `top_p = 1`. The task admits exactly one correct answer, so sampling contributes variance without contributing information, and the comparison across models is cleaner when every model is asked to decode the same way.

Not every model honors the request. Of the 205 models in the catalogue, 33 offer no temperature control: the OpenAI `o`-series and GPT-5 family (27 models) reject the parameter outright; Anthropic releases from Opus 4.7 onward (five models) either reject it or, according to the provider's migration documentation, accept it and silently ignore it; and one Google model likewise accepts and ignores it. For these the parameter has no effect, and each response is a single draw from the provider's own decoding configuration rather than a deterministic reading. A scan that issues one probe per sequence length cannot distinguish a model that is reliably wrong at a given length from one that is occasionally wrong there, so for these 33 models an isolated error may reflect sampling variance rather than a stable property of the model. Each such model carries a `temperature_locked` flag in its measurement record (set for models that reject the parameter and for the three known to ignore it), so analyses that require pinned decoding can exclude them. Appendix A.2 measures how much this sampling moves the fitted boundary.

A further 25 models do not accept a `seed` parameter, which removes the remaining route to reproducible sampling on the subset that also fixes temperature. We report results for these models alongside the rest rather than excluding them, since excluding the entire OpenAI reasoning line would remove much of the frontier, but the distinction is preserved in the data.

A.2 Decoding replicates

A scan issues one probe per length, so for the 33 models above every reply is a single sample from the provider's decoder, and the profile interval of Appendix B describes the fit to that one record rather than what a second scan would return. We measured the second scan directly. Ten models were rescanned five times each through their transition region: nine that offer no temperature control, spanning the three providers above, and `claude-sonnet-4.6`, the most recent model in the catalogue that honors temperature zero, as the control. Each replicate uses the catalogue protocol and the identical stimulus, starts at 0.8 of the catalogue CC (0.85 for the largest-capacity model), and runs until the criterion of Appendix B resolves the transition, so the record covers the passage from mostly right to mostly wrong without repeating the stable prefix; the two-regime fit depends only on the probed record, so a truncated prefix leaves the estimate comparable to the catalogue's. Table 1 lists every fitted boundary and Fig. 3D draws the records.

The fitted boundary is stable: across the nine models without temperature control the five replicates have a standard deviation of 0.7% to 6% of CC, and the replicate median lies within 10% of the catalogue value for eight of the ten models. Refitting each catalogue record on the same truncated

window returns the full-record boundary or the upper end of its interval, so the window itself does not bias the comparison. The ordering of the models is stable: no two models whose catalogue capacities differ by more than 10% exchange places in any replicate. The individual replies are not: at a length probed by all five replicates, the outcomes agree about as often as independent draws at that length's mean accuracy would, so the boundary is stable because the estimator integrates over stochastic replies, not because the replies repeat. The pinned control shows the same: the one model pinned at temperature zero agrees with itself no more often than the sampled models do, and its five boundaries are the most widely spread of the ten. Pinning the temperature does not make a served model deterministic, and the models that cannot be pinned are no less stable than the one that can. The profile interval of a single scan therefore describes the fit to one record and understates run-to-run uncertainty for a served model, by an order of magnitude when the interval collapses to one or two items, while the run-to-run uncertainty itself remains one to two orders of magnitude smaller than the differences between models that the paper interprets.

Table 1: **Decoding replicates.** Five independent rescans of the transition region per model under the catalogue protocol, with the same stimulus in every replicate. Temperature: whether the provider rejects the parameter, accepts and ignores it, or honours zero. Agreement is the share of commonly probed lengths on which all five replicates return the same outcome, against the share expected if each reply were an independent draw at that length's mean accuracy.

Model	Temperature	CC [95% CI]	Replicate boundaries	SD	SD/CC	Agreement
`claude-sonnet-5`	rejects	92 [91, 99]	100 90 100 90 89	5.1	5.5%	85% (88%)
`gpt-5.6-sol`	rejects	96 [95, 96]	106 106 96 114 108	5.8	6.0%	75% (81%)
`gpt-5`	rejects	101 [67, 104]	101 102 103 103 101	0.9	0.9%	54% (65%)
`o3`	rejects	103 [103, 103]	103 103 102 102 101	0.7	0.7%	71% (78%)
`gpt-5.2`	rejects	152 [146, 153]	153 153 152 153 159	2.5	1.7%	39% (52%)
`claude-opus-5`	ignores	179 [131, 180]	162 162 159 179 161	7.3	4.1%	58% (64%)
`gpt-5.4`	rejects	208 [208, 208]	210 212 210 216 209	2.5	1.2%	60% (69%)
`claude-fable-5`	rejects	317 [316, 329]	330 330 341 328 337	5.0	1.6%	59% (70%)
`gemini-3.7-flash`	ignores	352 [350, 373]	348 358 370 351 351	7.9	2.2%	60% (69%)
`claude-sonnet-4.6`	pinned at 0	320 [315, 320]	370 424 407 370 370	22.9	7.2%	83% (87%)

A.3 CC AS A SERVING FINGERPRINT

A model's capacity together with the lengths and values of its wrong answers is reproducible enough across scans to identify it (Appendix A.2, where the boundary is stable although individual replies are not): two models never fail at the same lengths with the same answers. This makes CC a cheap check of what an API endpoint actually serves. The protocol is the catalogue measurement with a coarser schedule: the prompt of Appendix A, temperature zero, an output cap of 512 tokens, one probe at every length from 1 to 60, every second length to 200, every fifth to 400, and every tenth to 800, 210 probes in all, fitted as in Appendix B and compared with the catalogue record of the claimed model by capacity and by the wrong answers at shared lengths.

We applied it to an OpenAI-compatible reseller advertising ten Claude names (six models and four aliases), four GPT names, and two Grok names, and scanned eight of the Claude names, three of the GPT names (the fourth timed out on 205 of 210 probes), and both Grok names. Every scanned Claude name returned the same fingerprint: a capacity of 460 to 480, off-by-one slips at the same lengths (365, 470, 480, 490), and identical answers at 197 to 205 of the 210 probed lengths across names, where the catalogue capacities of the claimed models range from 92 to 622. Every scanned GPT and Grok name returned a capacity of 44 to 46 with the same wrong answers at most shared failing lengths, against catalogue values of 94 to 226, and matched the catalogue record of one small OpenAI model answer for answer; the Grok routes also timed out on two thirds of their probes. Repeating one length eight times at temperature zero gave two different answers, so the Claude route was sampled without temperature control. The operator subsequently confirmed both routes: an Opus-class model served through a coding-agent harness behind the Claude names, and the small model behind the GPT and Grok names. The reseller is not named.

Prompt prefixes. We have not found a prompt that raises CC above the bare catalogue prompt of Appendix A. The evidence is narrow and is stated as such: on the route above, stacking 5k, 10k, and 20k tokens of neutral system prompt ahead of the task left every probed reply unchanged, including

the first slip; the reasoning-mode comparison of Section 3 is the other prompt-level variation we tested, and across 77 verified pairs it leaves the median unchanged. Prompts that change the stimulus (the counted item or the delimiter, Fig. 2F) move the fitted CC severalfold in both directions without reordering the models.

A.4 Models excluded for provider-side reasons

Four models in the candidate roster could not be measured under the protocol above and were excluded, with the reason recorded rather than inferred from the model's name.

`meta-llama/llama-guard-4-12b` is a safety classifier rather than a general model; it replies `safe` to every prompt, including prompts that contain no counting task. Exclusion is by behavior rather than by name: a similarly named safeguard model counts normally and is retained.

`mistralai/mistral-small-3.1-24b-instruct` and `qwen/qwen-2.5-coder-32b-instruct` return a provider error whenever the system message constrains the reply to a bare integer. The failure is reproducible across prompt variants and is not a property of either model: both answer counting questions normally when the system message is omitted or replaced with an unconstrained one, and both are served by a single provider with no alternative endpoint. The failure is therefore attributable to the serving path rather than to the model, and is noted here because constrained-output instructions are common in automated evaluation and this failure mode is silent: the provider's HTTP 500 reaches the client as an HTTP 200 with an error object in the body, so a harness that inspects only the status code will record it as an empty model response.

`mistralai/mistral-7b-instruct` is listed by the model catalogue but has no live endpoint (HTTP 404).

A.5 Collapse-region analysis

To map the transition near collapse, we used the dense per-length record around the CC boundary produced by the scan. Parsed predictions were aggregated to quantify the transition from exact outputs to failed outputs. Failed-output attractors were identified by aggregating incorrect predicted values across the sweep (Fig. 2D, E).

We also varied the repeated character and delimiter syntax (Fig. 2F). Token volume does not explain the resulting shifts: within each of the three models, the rank correlation between tokens per item and the change in CC across the eleven encodings is weak and negative (-0.31 to -0.52), and the largest losses come from encodings that cost almost no additional tokens; the section-sign delimiter adds a tenth of a token per item for one model while removing three quarters of its capacity. The shifts therefore reflect the learned counting trajectory of a specific surface form rather than the number of tokens it occupies.

B Estimating counting capacity

Each measurement queries every sequence length once, in ascending order from $L = 1$, and reduces each reply to a single bit: whether the parsed integer equaled the target exactly. A reply containing no parsable integer is scored as inexact, since a model that cannot deliver an answer within the protocol's output budget is not stably counting under the protocol. One measurement is therefore a binary series $x_1, \ldots, x_n$ indexed by length, and CC is a summary of that series.

We do not define CC as the position of the first inexact answer. The first failure is the minimum of the failure positions, and it is therefore a property of the sampling density as much as of the model: probing the same model more densely can only move it earlier. It is also empirically unrepresentative, since 284 of our 296 measurements answer exactly at some length above their first inexact answer, indicating that the first failure is typically an isolated slip rather than a boundary.

We therefore estimate the boundary by fitting the full series. The model is that accuracy is p_1 up to a transition τ and p_2 above it, subject to the requirement that the two regimes lie on opposite sides of one half,

$$p_1 > \tfrac{1}{2} > p_2,$$

so that the fitted boundary separates a regime in which exact counting is the model's majority behavior from one in which it is not. All three parameters are estimated by maximum likelihood, and CC is the fitted $\hat{\tau}$. For a cut after the kth length, with c_1 exact answers below and c_2 above, each side takes its own estimate $\hat{p}_i = c_i/n_i$ and the profile log-likelihood is the sum of two Bernoulli terms,

$$\ell(k) = c_1 \ln \hat{p}_1 + (k - c_1) \ln(1 - \hat{p}_1) + c_2 \ln \hat{p}_2 + (n - k - c_2) \ln(1 - \hat{p}_2).$$

Prefix sums make this $O(1)$ per cut and $O(n)$ for the whole profile.

The one-half requirement is the only structural choice in the fit, and it ensures that the fitted boundary is a counting boundary rather than a reliability boundary. Without it, the fit would separate a perfect prefix from a tail that is 95% accurate, because such a drop is statistically detectable across several hundred probes, and the estimate would then report a small loss of reliability as the capacity. One half is also not a tuned threshold in the sense that a 90% or 95% accuracy floor would be: it is the unique accuracy value that is symmetric under exchanging the labels exact and inexact, and it therefore requires no further justification.

A transition is declared only when the improvement in fit justifies the two additional parameters, using the Bayesian information criterion: the two-regime fit must exceed the single-regime fit by more than $\ln n$ in log-likelihood. This choice also removes the standard changepoint edge artifact, in which a stray failure near the end of a long record is split off as a spurious two-probe regime. Declared transitions in our data clear this bar by a median of 40 log-likelihood units against bars of 4.6 to 6.9, so the result does not depend on the criterion chosen. Where no transition is declared, the measurement is reported as a lower bound rather than a point estimate.

The confidence interval on τ is a profile set: every admissible cut whose log-likelihood lies within a bar of the maximum. Because τ is a discrete parameter, the chi-squared bar of 1.92 that a profile interval ordinarily uses is not calibrated for it: in simulation from the two-regime model with $p_1 = 0.96$ and $p_2 = 0.12$, a set built with that bar contains the true cut 86 to 87% of the time at fixed scan lengths of 100 and 300, and 78 to 89% under the sequential stopping rule described below. We therefore calibrate the bar for each measurement by parametric bootstrap: 200 records are drawn from the fitted $(\hat{\tau}, \hat{p}_1, \hat{p}_2)$ over the observed probe lengths, and the bar is the 95th percentile of the amount by which the best admissible cut's log-likelihood exceeds the true cut's, never less than 1.92. Under the same simulations this set contains the true cut 94% of the time at fixed scan lengths and 90 to 92% under the stopping rule, the residual shortfall arising from scans that stop with only a few probes above the boundary. Across our measurements the calibrated interval has a median width of four lengths, and 152 of the 294 fitted transitions (the two `minimax-m1` records, which ended in provider errors, enter no statistic) span five lengths or fewer, 138 of them among the 256 resolved by wrong answers. This interval is a statement about the estimate rather than about the underlying process: given several hundred binary probes and a contrast as large as $p_1 = 0.96$ against $p_2 = 0.12$, the location of the best cut is determined precisely. The measured rate itself falls more gradually. When the five rescans of each model in Appendix A.2 are pooled and smoothed over ± 5 lengths, accuracy passes from 0.8 to 0.2 over a median of 26 lengths, about one tenth of CC (0.05 to 0.43; Table 2), so $\hat{\tau}$ locates a transition of real width rather than an instant at which counting stops. The two-regime form summarizes this transition and does not claim that the underlying rate is a step; a claim about how abrupt the collapse is would require a model that estimates a transition width, which the two-regime form deliberately omits. The intervals are drawn as the transition bands in Fig. 1c.

Because the fit uses the whole record rather than an extreme of it, the estimate is stable against how densely the model was probed. Refitting every measurement on every second probe moves $\hat{\tau}$ by a median of -1 length, and on every fifth probe by -2; the position of the first inexact answer moves by $+4$ lengths (mean $+12$) under the same subsampling.

Scanning continues until the capacity is determined: a scan stops once a transition is declared and the accuracy above it is significantly below one half, as assessed by the upper end of the 95% Wilson interval on $\hat{p}_2$. The stopping rule therefore coincides with the estimator, and a measurement ends when it has determined the quantity being measured rather than after a fixed probe budget. Scans that instead terminate on ten consecutive unparsable replies are reported as bounded by the output budget rather than by counting, and are excluded from statements about counting limits.

Table 2: **Width of the behavioral transition.** Per-length accuracy pooled over the five rescans of Appendix A.2 and smoothed over ± 5 lengths; the width is the span from the last length at which smoothed accuracy is at least 0.8 to the first at which it is at most 0.2.

Model	CC	last ≥ 0.8	first ≤ 0.2	width	width / CC
`claude-fable-5`	317	313	345	32	0.10
`claude-opus-5`	179	–	–	–	–
`claude-sonnet-4.6`	320	396	426	30	0.09
`claude-sonnet-5`	92	86	104	18	0.20
`gemini-3.7-flash`	352	347	391	44	0.12
`gpt-5`	101	98	107	9	0.09
`gpt-5.2`	152	133	159	26	0.17
`gpt-5.4`	208	205	216	11	0.05
`gpt-5.6-sol`	96	93	134	41	0.43
`o3`	103	98	107	9	0.09

B.1 Bounds on guessing exploits

The CC measurement scores a reply as exact or inexact, with no tolerance band, so approximately correct answers receive no partial credit. This section states what a guessing strategy can achieve against that criterion, and against the two-regime fit built on top of it.

B.1.1 Guessing against a single length

Suppose a model retains no exact count but recognizes the approximate scale of the prompt, and returns a value drawn from some distribution Q concentrated near the true count L. Because credit requires $\hat{x} = L$ exactly, the probability of success at that length is $Q(\{L\})$. A strategy that spreads its mass over a window of w plausible values can do no better than $\max_x Q(\{x\})$, and a strategy that returns a fixed value c succeeds at exactly one length in the entire scan. There is no analogue of the error-minimizing guess that a relative-error criterion admits: under exact matching, approximate knowledge of the magnitude provides no credit.

B.1.2 Guessing against the fitted boundary

The stronger statement concerns the estimator rather than a single trial. A transition is declared only if some cut separates a regime with $\hat{p}_1 > \frac{1}{2}$ from one with $\hat{p}_2 < \frac{1}{2}$. Establishing a stable regime of k lengths therefore requires exact answers at more than $k/2$ distinct lengths.

Consider the most favorable guessing strategy available to a model with no exact count: at each length it guesses independently, placing probability at most $q < \frac{1}{2}$ on the correct value; a guess drawn uniformly from a window of w values centered on the correct value has $q = 1/w$, and a non-uniform window is covered by taking q as the largest probability it places on any value. The number of successes over k lengths is stochastically dominated by $S \sim \mathrm{Binomial}(k, q)$, and a Chernoff bound gives

$$\Pr\left[S > \tfrac{k}{2}\right] \leq \exp\left(-k\, D\left(\tfrac{1}{2} \,\|\, q\right)\right), \tag{1}$$

where $D(a\|b) = a \ln \frac{a}{b} + (1-a) \ln \frac{1-a}{1-b}$ is the binary relative entropy. Even for a window as narrow as $w = 4$, $D(\frac{1}{2}\|\frac{1}{4}) \approx 0.1438$, so a guessing model has probability below 10^{-3} of producing an apparent stable regime of $k = 49$ lengths, and below 10^{-6} at $k = 97$. The median CC in our data is 50 and the largest is 627. A window wide enough to be plausible for a model that has genuinely lost the count, say $w = 20$, reduces the same probabilities below 10^{-3} at $k = 9$ and below 10^{-6} at $k = 17$.

The bound is conservative in two further respects. It grants the guessing model a window centered on the correct value at every length, and it ignores the requirement that the regime above the boundary be significantly *worse* than one half, which a scale-guessing strategy with a fixed window would not satisfy: its accuracy does not degrade with length, so it produces no transition to declare. In practice a model that has lost the count does not guess within a narrow window at all; it collapses onto a small set of round attractors (Fig. 2D, E), which succeed at a handful of lengths in the entire scan. The bound concerns a guessing strategy producing an apparently stable prefix; it is not a guarantee on the full estimator, whose cut selection and stopping rule are checked by simulation above.

B.1.3 IMPLICATIONS

Exact matching eliminates the class of exploit that a tolerance-based criterion must defend against, and the majority requirement in the fit converts that per-length guarantee into an exponential bound on fabricating a stable regime. Under the stated independence and success-probability assumptions the bound therefore limits fabrication of a wholly guessed stable prefix; it does not bound inflation of an existing capacity estimate by guessed answers in the tail.

C MODEL SET, CATALOGUE, AND DATA RELEASE

We evaluated 205 language models, proprietary and open-weight, comprising 296 measurements once reasoning-enabled and reasoning-disabled configurations of the same model are counted separately. The set included instruction-tuned and reasoning-augmented variants spanning a broad range of parameter counts and nominal context lengths. This range allowed us to test whether CC reflects a general capability axis rather than a quirk of a particular tokenizer, serving stack, or model family.

All catalogue evaluations were conducted with tool use disabled. For externally served models, we recorded the model identifier, evaluation date, decoding configuration, token-usage metadata, parsed response, target count, and success or failure classification. These records were retained to account for possible changes in served model behavior over time. The complete proprietary-model evaluation cost approximately US$200 in API spend, excluding local open-weight mechanistic analyses.

Table 4 lists the full model catalogue together with the corresponding developer or family, parameter metadata, nominal context window, architectural class, and measured counting capacity (CC). The two `minimax-m1` measurements ended in provider errors after 0 and 10 items; they are listed as bounds and enter no statistic. Table 3 gives the release-year cohorts behind the longitudinal comparison in the introduction: resolved measurements grouped by release year, with cohort sizes, median CC, and the cohorts' median advertised context window.

Table 3: **Release-year cohorts behind the longitudinal comparison.** Resolved measurements (transition bounded by wrong answers) grouped by the model's release year; reasoning configurations counted separately. Median CC with interquartile range, and the median advertised context window of the same measurements. The 2023 cohort is four models.

Release year	Measurements	Models	Median CC	IQR	Median context (tokens)
2023	4	4	35	[27, 44]	12,288
2024	22	22	27	[21, 36]	128,000
2025	106	82	44	[32, 84]	230,400
2026	124	77	56	[43, 115]	1,000,000

Table 4: **Full model catalogue.** One row per measurement; hybrid models appear once per reasoning configuration. CC is the fitted transition with its calibrated interval (nominal 95%; Appendix B); capacities marked $\geq$ are lower bounds, set by the output budget rather than by a counting failure. Parameter counts give, or unresolved where the bound column says so. total and, for mixture models, active parameters. Metadata are drawn from the OpenRouter catalogue and public documentation.

Measurement	Params	Context	CC	95% CI	Bounded by
`claude-3-haiku`	–	200k	40	[39, 50]	mistake
`claude-fable-5`	–	1M	317	[316, 329]	mistake
`claude-haiku-4.5 #nothink`	–	200k	132	[130, 135]	mistake
`claude-haiku-4.5 #think`	–	200k	131	[130, 131]	mistake
`claude-opus-4 #nothink`	–	200k	580	[575, 608]	mistake
`claude-opus-4 #think`	–	200k	627	[626, 628]	mistake
`claude-opus-4.1 #nothink`	–	200k	580	[550, 582]	mistake
`claude-opus-4.1 #think`	–	200k	573	[567, 574]	mistake
`claude-opus-4.5 #nothink`	–	200k	580	[578, 582]	mistake
`claude-opus-4.5 #think`	–	200k	620	[618, 622]	mistake
`claude-opus-4.6 #nothink`	–	1M	618	[618, 622]	mistake
`claude-opus-4.6 #think`	–	1M	622	[620, 623]	mistake
`claude-opus-4.7 #nothink`	–	1M	158	[157, 158]	mistake
`claude-opus-4.7 #think`	–	1M	156	[140, 156]	mistake
`claude-opus-4.8 #nothink`	–	1M	133	[132, 133]	mistake
`claude-opus-4.8 #think`	–	1M	109	[108, 109]	mistake
`claude-opus-5 #nothink`	–	1M	179	[131, 180]	mistake

Measurement	Params	Context	CC	95% CI	Bounded by
claude-opus-5 #think	–	1M	132	[129, 162]	mistake
claude-sonnet-4 #nothink	–	1M	382	[362, 382]	mistake
claude-sonnet-4 #think	–	1M	290	[289, 293]	mistake
claude-sonnet-4.5 #nothink	–	1M	364	[364, 364]	mistake
claude-sonnet-4.5 #think	–	1M	364	[359, 364]	mistake
claude-sonnet-4.6 #nothink	–	1M	320	[315, 320]	mistake
claude-sonnet-4.6 #think	–	1M	$\geq$264	[259, 265]	budget
claude-sonnet-5 #nothink	–	1M	92	[91, 99]	mistake
claude-sonnet-5 #think	–	1M	89	[85, 91]	mistake
deepseek-chat	684.531B	163k	30	[15, 44]	mistake
deepseek-chat-v3-0324	684.531B	163k	30	[25, 31]	mistake
deepseek-chat-v3.1 #nothink	684.531B / 37B	163k	15	[14, 15]	mistake
deepseek-chat-v3.1 #think	684.531B / 37B	163k	$\geq$28	[13, 63]	budget
deepseek-r1	684.531B / 37B	64k	$\geq$67	[65, 81]	budget
deepseek-r1-0528	684.531B / 37B	163k	$\geq$71	[69, 77]	budget
deepseek-r1-distill-llama-70b #nothink	70.554B / 70B	8k	$\geq$57	[33, 66]	budget
deepseek-r1-distill-llama-70b #think	70.554B / 70B	8k	$\geq$58	[56, 60]	budget
deepseek-v3.1-terminus #nothink	684.531B	163k	13	[11, 13]	mistake
deepseek-v3.1-terminus #think	684.531B	163k	$\geq$60	[51, 62]	budget
deepseek-v3.2 #nothink	685.397B	163k	12	[12, 12]	mistake
deepseek-v3.2 #think	685.397B	163k	$\geq$70	[53, 71]	budget
deepseek-v3.2-exp #nothink	685.397B	163k	10	[9, 12]	mistake
deepseek-v3.2-exp #think	685.397B	163k	$\geq$58	[53, 61]	budget
deepseek-v4-flash #nothink	290.945B / 13B	1.04858M	52	[51, 63]	mistake
deepseek-v4-flash #think	290.945B / 13B	1.04858M	$\geq$72	[67, 86]	budget
deepseek-v4-flash-0731 #nothink	304.18B / 13B	1.31072M	44	[34, 56]	mistake
deepseek-v4-flash-0731 #think	304.18B / 13B	1.31072M	$\geq$51	[47, 96]	budget
deepseek-v4-flash-vision-exp #nothink	–	1.04858M	139	[136, 140]	mistake
deepseek-v4-flash-vision-exp #think	–	1.04858M	$\geq$76	[74, 89]	budget
deepseek-v4-pro #nothink	1598.84B / 49B	1.04858M	36	[34, 38]	mistake
deepseek-v4-pro #think	1598.84B / 49B	1.04858M	$\geq$72	[69, 88]	budget
deepseek-v4-pro-0813 #nothink	1650.5B	1.04858M	36	[35, 36]	mistake
deepseek-v4-pro-0813 #think	1650.5B	1.04858M	$\geq$96	[85, 97]	budget
gemini-2.5-flash #nothink	–	1.04858M	102	[99, 103]	mistake
gemini-2.5-flash #think	–	1.04858M	107	[100, 108]	mistake
gemini-2.5-flash-lite #nothink	–	1.04858M	79	[54, 80]	mistake
gemini-2.5-flash-lite #think	–	1.04858M	54	[50, 54]	mistake
gemini-2.5-pro	–	1.04858M	119	[105, 125]	mistake
gemini-2.5-pro-preview	–	1.04858M	120	[113, 123]	mistake
gemini-2.5-pro-preview-05-06	–	1.04858M	119	[108, 124]	mistake
gemini-3-flash-preview #nothink	–	1.04858M	445	[435, 447]	mistake
gemini-3-flash-preview #think	–	1.04858M	404	[398, 421]	mistake
gemini-3.1-flash-lite #nothink	–	1.04858M	130	[128, 133]	mistake
gemini-3.1-flash-lite #think	–	1.04858M	115	[112, 130]	mistake
gemini-3.1-flash-lite-preview #nothink	–	1.04858M	130	[129, 130]	mistake
gemini-3.1-flash-lite-preview #think	–	1.04858M	130	[129, 133]	mistake
gemini-3.1-pro-preview	–	1.04858M	518	[518, 519]	mistake
gemini-3.1-pro-preview-customtools	–	1.04858M	429	[429, 429]	mistake
gemini-3.5-flash	–	1.04858M	298	[289, 299]	mistake
gemini-3.5-flash-lite	–	1.04858M	146	[142, 149]	mistake
gemini-3.6-flash	–	1.04858M	347	[316, 348]	mistake
gemini-3.7-flash	–	1.04858M	352	[350, 373]	mistake
gemma-2-27b-it	27.227B / 27B	8k	21	[21, 21]	mistake
gemma-3-12b-it	12.187B / 12B	131k	32	[15, 39]	mistake
gemma-3-27b-it	27.432B / 27B	262k	36	[30, 37]	mistake
gemma-3-4b-it	4.3B / 4B	131k	20	[20, 27]	mistake
gemma-3n-e4b-it	7.85B / 4B	32k	17	[16, 22]	mistake
gemma-4-26b-a4b-it #nothink	25.806B / 4B	262k	83	[53, 84]	mistake
gemma-4-26b-a4b-it #think	25.806B / 4B	262k	$\geq$41	[40, 50]	budget
gemma-4-31b-it #nothink	31.273B / 31B	262k	38	[28, 40]	mistake
gemma-4-31b-it #think	31.273B / 31B	262k	45	[35, 46]	mistake

Measurement	Params	Context	CC	95% CI	Bounded by
muse-glimmer-30b	29.777B / 30B	131k	54	[54, 54]	mistake
muse-spark-1.1	–	1.04858M	195	[192, 206]	mistake
muse-spark-1.2	–	1.04858M	≥96	[90, 98]	budget
muse-spark-1.2-contributor	–	1.04858M	≥96	[91, 109]	budget
llama-3.1-70b-instruct	70.554B / 70B	131k	16	[16, 25]	mistake
llama-3.1-8b-instruct	8.03B / 8B	131k	21	[20, 24]	mistake
llama-3.2-1b-instruct	1.236B / 1B	60k	2	[2, 3]	mistake
llama-3.2-3b-instruct	3.213B / 3B	131k	24	[8, 25]	mistake
llama-3.3-70b-instruct	70.554B / 70B	131k	27	[27, 27]	mistake
llama-4-maverick	401.584B / 17B	1.04858M	55	[54, 57]	mistake
llama-4-scout	108.642B / 17B	1.31072M	22	[21, 22]	mistake
minimax-01	456.09B	1.00019M	37	[32, 40]	mistake
minimax-m1 #nothink	–	1M	≥0	[0, 0]	budget
minimax-m1 #think	–	1M	≥10	–	unresolved
minimax-m2	228.69B / 10B	204k	37	[36, 37]	mistake
minimax-m2-her	–	65k	18	[17, 19]	mistake
minimax-m2.1	228.69B / 10B	204k	40	[39, 40]	mistake
minimax-m2.5	228.704B	204k	39	[39, 39]	mistake
minimax-m2.7	228.69B	204k	≥40	[40, 40]	budget
minimax-m3 #nothink	427.04B	1.04858M	51	[47, 56]	mistake
minimax-m3 #think	427.04B	1.04858M	56	[55, 65]	mistake
codestral-2508	–	256k	48	[31, 49]	mistake
ministral-14b-2512	13.945B / 14B	262k	31	[30, 35]	mistake
ministral-3b-2512	3.849B / 3B	131k	19	[19, 19]	mistake
ministral-8b-2512	8.918B / 8B	262k	21	[21, 31]	mistake
mistral-large	–	128k	31	[31, 31]	mistake
mistral-large-2407	–	131k	31	[31, 31]	mistake
mistral-large-2512	–	262k	31	[31, 31]	mistake
mistral-medium-3	–	131k	35	[35, 40]	mistake
mistral-medium-3-5 #nothink	–	262k	45	[45, 45]	mistake
mistral-medium-3-5 #think	–	262k	31	[28, 31]	mistake
mistral-medium-3.1	–	131k	39	[39, 39]	mistake
mistral-nemo	12.248B	131k	20	[20, 29]	mistake
mistral-saba	–	32k	32	[20, 33]	mistake
mistral-small-24b-instruct-2501	23.572B / 24B	32k	29	[29, 30]	mistake
mistral-small-2603 #nothink	119.401B / 119B	262k	32	[30, 34]	mistake
mistral-small-2603 #think	119.401B / 119B	262k	45	[44, 52]	mistake
mistral-small-3.2-24b-instruct	24.011B / 24B	131k	31	[31, 32]	mistake
mixtral-8x22b-instruct	140.63B / 39B	65k	32	[20, 33]	mistake
voxtral-small-24b-2507	24.262B / 24B	32k	28	[28, 29]	mistake
kimi-k2	1026.41B / 32B	131k	76	[64, 76]	mistake
kimi-k2-0905	1026.47B	262k	72	[63, 92]	mistake
kimi-k2-thinking	1058.12B	262k	70	[45, 73]	mistake
kimi-k2.5 #nothink	1026.88B	262k	72	[59, 74]	mistake
kimi-k2.5 #think	1026.88B	262k	73	[70, 95]	mistake
kimi-k2.6 #nothink	1026.88B	262k	76	[63, 77]	mistake
kimi-k2.6 #think	1026.88B	262k	≥56	[54, 69]	budget
kimi-k2.7-code	1026.88B	262k	≥70	[50, 74]	budget
kimi-k3 #nothink	2779.93B	1.04858M	54	[53, 54]	mistake
kimi-k3 #think	2779.93B	1.04858M	51	[50, 51]	mistake
gpt-3.5-turbo	20B	16k	25	[25, 25]	mistake
gpt-3.5-turbo-0613	–	4k	50	[45, 52]	mistake
gpt-3.5-turbo-16k	–	16k	34	[33, 38]	mistake
gpt-3.5-turbo-instruct	20B	4k	36	[25, 39]	mistake
gpt-4	–	8k	46	[38, 53]	mistake
gpt-4-turbo	–	128k	25	[20, 26]	mistake
gpt-4.1	–	1.04758M	54	[52, 54]	mistake
gpt-4.1-mini	–	1.04758M	38	[31, 53]	mistake
gpt-4.1-nano	–	1.04758M	46	[42, 51]	mistake
gpt-4o	–	128k	50	[46, 53]	mistake
gpt-4o-2024-05-13	–	128k	42	[42, 43]	mistake
gpt-4o-2024-08-06	–	128k	48	[34, 53]	mistake
gpt-4o-2024-11-20	–	128k	34	[32, 34]	mistake
gpt-4o-mini	–	128k	25	[24, 26]	mistake
gpt-4o-mini-2024-07-18	–	128k	24	[23, 24]	mistake
gpt-5	–	400k	101	[67, 104]	mistake
gpt-5-mini	–	400k	60	[51, 61]	mistake
gpt-5-nano	–	400k	35	[34, 35]	mistake
gpt-5.1 #nothink	–	400k	101	[81, 102]	mistake
gpt-5.1 #think	–	400k	101	[101, 102]	mistake

Measurement	Params	Context	CC	95% CI	Bounded by
gpt-5.1-codex	–	400k	≥100	[99, 101]	budget
gpt-5.1-codex-max	–	400k	≥43	[41, 50]	budget
gpt-5.1-codex-mini #nothink	–	400k	≥50	[50, 51]	budget
gpt-5.1-codex-mini #think	–	400k	≥50	[47, 51]	budget
gpt-5.2 #nothink	–	400k	113	[110, 114]	mistake
gpt-5.2 #think	–	400k	152	[146, 153]	mistake
gpt-5.2-codex	–	400k	115	[96, 119]	mistake
gpt-5.3-codex #nothink	–	400k	210	[209, 210]	mistake
gpt-5.3-codex #think	–	400k	223	[223, 223]	mistake
gpt-5.4 #nothink	–	1.05M	208	[208, 208]	mistake
gpt-5.4 #think	–	1.05M	174	[166, 176]	mistake
gpt-5.4-mini #nothink	–	400k	57	[55, 57]	mistake
gpt-5.4-mini #think	–	400k	57	[56, 64]	mistake
gpt-5.4-nano #nothink	–	400k	31	[26, 32]	mistake
gpt-5.4-nano #think	–	400k	27	[27, 27]	mistake
gpt-5.5 #nothink	–	1.05M	105	[100, 120]	mistake
gpt-5.5 #think	–	1.05M	96	[93, 98]	mistake
gpt-5.6-luna #nothink	–	1.05M	46	[43, 47]	mistake
gpt-5.6-luna #think	–	1.05M	≥45	[44, 52]	budget
gpt-5.6-luna-pro #nothink	–	1.05M	46	[44, 47]	mistake
gpt-5.6-luna-pro #think	–	1.05M	46	[37, 47]	mistake
gpt-5.6-sol #nothink	–	1.05M	94	[92, 94]	mistake
gpt-5.6-sol #think	–	1.05M	96	[95, 96]	mistake
gpt-5.6-sol-pro #nothink	–	1.05M	94	[91, 97]	mistake
gpt-5.6-sol-pro #think	–	1.05M	93	[86, 93]	mistake
gpt-5.6-terra #nothink	–	1.05M	117	[115, 117]	mistake
gpt-5.6-terra #think	–	1.05M	115	[113, 117]	mistake
gpt-5.6-terra-pro #nothink	–	1.05M	174	[174, 175]	mistake
gpt-5.6-terra-pro #think	–	1.05M	120	[112, 121]	mistake
gpt-chat-latest	–	400k	76	[67, 78]	mistake
gpt-oss-120b	116.829B / 5.1B	131k	101	[101, 101]	mistake
gpt-oss-20b	21.512B / 3.6B	131k	58	[57, 58]	mistake
gpt-oss-safeguard-20b	21.512B / 3.6B	131k	57	[52, 61]	mistake
o1 #think	–	200k	≥66	[46, 72]	budget
o3 #think	–	200k	103	[103, 103]	mistake
o3-mini #think	–	200k	37	[37, 37]	mistake
o3-mini-high	–	200k	≥37	[32, 40]	budget
o4-mini #think	–	200k	≥50	[49, 64]	budget
o4-mini-high	–	200k	≥47	[46, 50]	budget
qwen-2.5-72b-instruct	72.706B / 72B	32k	27	[27, 27]	mistake
qwen-2.5-7b-instruct	7.616B / 7B	32k	12	[12, 15]	mistake
qwen-plus	–	1M	33	[32, 33]	mistake
qwen-plus-2025-07-28 #nothink	–	1M	54	[53, 54]	mistake
qwen-plus-2025-07-28 #think	–	1M	54	[53, 54]	mistake
qwen-plus-2025-07-28:thinking #nothink	–	1M	58	[58, 60]	mistake
qwen-plus-2025-07-28:thinking #think	–	1M	58	[58, 60]	mistake
qwen2.5-vl-72b-instruct	73.411B / 72B	128k	30	[30, 32]	mistake
qwen3-14b #nothink	14.768B / 14B	131k	≥27	[26, 30]	budget
qwen3-14b #think	14.768B / 14B	131k	≥31	[28, 32]	budget
qwen3-235b-a22b #nothink	235.094B / 22B	131k	45	[44, 45]	mistake
qwen3-235b-a22b #think	235.094B / 22B	131k	43	[42, 50]	mistake
qwen3-235b-a22b-2507	235.094B / 22B	262k	54	[50, 60]	mistake
qwen3-235b-a22b-thinking-2507	235.094B / 22B	262k	≥57	[54, 58]	budget
qwen3-30b-a3b #nothink	30.532B / 3B	131k	37	[32, 40]	mistake
qwen3-30b-a3b #think	30.532B / 3B	131k	35	[35, 35]	mistake
qwen3-30b-a3b-instruct-2507	30.532B / 3B	262k	35	[30, 41]	mistake
qwen3-30b-a3b-thinking-2507	30.532B / 3B	81k	40	[40, 45]	mistake
qwen3-32b #nothink	32.762B / 32B	131k	22	[22, 22]	mistake
qwen3-32b #think	32.762B / 32B	131k	26	[26, 33]	mistake
qwen3-8b #nothink	8.191B / 8B	131k	25	[24, 26]	mistake
qwen3-8b #think	8.191B / 8B	131k	28	[27, 30]	mistake
qwen3-coder	480.155B / 35B	262k	30	[29, 30]	mistake
qwen3-coder-30b-a3b-instruct	30.532B / 3B	262k	33	[31, 38]	mistake
qwen3-coder-flash	–	1M	39	[32, 40]	mistake
qwen3-coder-next	79.674B / 3B	262k	50	[49, 50]	mistake
qwen3-coder-plus #nothink	–	1M	50	[49, 54]	mistake
qwen3-coder-plus #think	–	1M	50	[49, 56]	mistake

Measurement	Params	Context	CC	95% CI	Bounded by
qwen3-max #nothink	1000B	262k	60	[60, 60]	mistake
qwen3-max #think	1000B	262k	60	[50, 61]	mistake
qwen3-max-thinking #nothink	–	262k	55	[49, 55]	mistake
qwen3-max-thinking #think	–	262k	55	[55, 55]	mistake
qwen3-next-80b-a3b-instruct	81.325B / 3B	262k	37	[37, 37]	mistake
qwen3-next-80b-a3b-thinking	81.325B / 3B	262k	40	[40, 40]	mistake
qwen3-vl-235b-a22b-instruct	235.67B / 22B	262k	50	[49, 51]	mistake
qwen3-vl-235b-a22b-thinking	235.67B / 22B	131k	$\geq$55	[55, 55]	budget
qwen3-vl-30b-a3b-instruct	31.071B / 3B	262k	36	[36, 37]	mistake
qwen3-vl-30b-a3b-thinking	31.071B / 3B	262k	44	[42, 45]	mistake
qwen3-vl-32b-instruct	33.357B / 32B	131k	28	[28, 30]	mistake
qwen3-vl-8b-instruct	8.767B / 8B	262k	21	[21, 25]	mistake
qwen3-vl-8b-thinking	8.767B / 8B	131k	$\geq$30	[30, 30]	budget
qwen3.5-122b-a10b #nothink	125.086B / 10B	262k	51	[51, 58]	mistake
qwen3.5-122b-a10b #think	125.086B / 10B	262k	50	[50, 52]	mistake
qwen3.5-27b #nothink	27.781B / 27B	262k	40	[39, 40]	mistake
qwen3.5-27b #think	27.781B / 27B	262k	41	[41, 41]	mistake
qwen3.5-35b-a3b #nothink	35.952B / 3B	262k	42	[42, 42]	mistake
qwen3.5-35b-a3b #think	35.952B / 3B	262k	$\geq$50	[46, 54]	budget
qwen3.5-397b-a17b #nothink	403.398B / 17B	262k	72	[72, 73]	mistake
qwen3.5-397b-a17b #think	403.398B / 17B	262k	72	[70, 72]	mistake
qwen3.5-9b #nothink	9.653B / 9B	262k	35	[35, 36]	mistake
qwen3.5-9b #think	9.653B / 9B	262k	$\geq$25	[25, 25]	budget
qwen3.5-flash-02-23 #nothink	–	1M	42	[41, 42]	mistake
qwen3.5-flash-02-23 #think	–	1M	46	[45, 54]	mistake
qwen3.5-plus-02-15 #nothink	–	1M	72	[72, 73]	mistake
qwen3.5-plus-02-15 #think	–	1M	72	[62, 72]	mistake
qwen3.5-plus-20260420 #nothink	–	1M	45	[45, 45]	mistake
qwen3.5-plus-20260420 #think	–	1M	47	[46, 55]	mistake
qwen3.6-27b #nothink	27.781B / 27B	262k	43	[42, 43]	mistake
qwen3.6-27b #think	27.781B / 27B	262k	42	[42, 42]	mistake
qwen3.6-35b-a3b #nothink	35.952B / 3B	262k	42	[37, 51]	mistake
qwen3.6-35b-a3b #think	35.952B / 3B	262k	45	[44, 53]	mistake
qwen3.6-flash #nothink	–	1M	42	[37, 45]	mistake
qwen3.6-flash #think	–	1M	43	[42, 43]	mistake
qwen3.6-max-preview #nothink	–	262k	68	[68, 68]	mistake
qwen3.6-max-preview #think	–	262k	69	[69, 69]	mistake
qwen3.6-plus #nothink	–	1M	50	[49, 50]	mistake
qwen3.6-plus #think	–	1M	53	[53, 56]	mistake
qwen3.7-flash #nothink	–	1M	37	[36, 42]	mistake
qwen3.7-flash #think	–	1M	50	[50, 53]	mistake
qwen3.7-max #nothink	–	1M	41	[39, 50]	mistake
qwen3.7-max #think	–	1M	51	[50, 51]	mistake
qwen3.7-plus #nothink	–	1M	47	[45, 50]	mistake
qwen3.7-plus #think	–	1M	57	[56, 61]	mistake
qwen3.8-2.4t-a95b	2446.18B / 95B	1.04858M	70	[70, 70]	mistake
qwen3.8-27b #nothink	27.781B / 27B	1M	38	[37, 41]	mistake
qwen3.8-27b #think	27.781B / 27B	1M	38	[38, 40]	mistake
qwen3.8-max	2400B	1M	70	[69, 70]	mistake
grok-4.20 #nothink	500B	2M	84	[73, 101]	mistake
grok-4.20 #think	500B	2M	85	[66, 106]	mistake
grok-4.3 #nothink	–	1M	100	[96, 104]	mistake
grok-4.3 #think	–	1M	102	[101, 104]	mistake
grok-4.5	–	500k	140	[139, 140]	mistake
grok-4.6	–	500k	226	[225, 227]	mistake
grok-build-0.1	–	256k	52	[49, 53]	mistake
glm-4.5 #nothink	358.338B	131k	41	[40, 42]	mistake
glm-4.5 #think	358.338B	131k	43	[43, 43]	mistake
glm-4.5-air #nothink	110.469B	131k	48	[43, 50]	mistake
glm-4.5-air #think	110.469B	131k	53	[53, 53]	mistake
glm-4.5v #nothink	107.711B / 12B	65k	48	[41, 50]	mistake
glm-4.5v #think	107.711B / 12B	65k	42	[41, 44]	mistake
glm-4.6 #nothink	356.786B	204k	41	[35, 42]	mistake
glm-4.6 #think	356.786B	204k	$\geq$53	[50, 54]	budget
glm-4.6v #nothink	107.711B	131k	22	[22, 22]	mistake
glm-4.6v #think	107.711B	131k	21	[21, 23]	mistake
glm-4.7 #nothink	358.338B	204k	40	[35, 42]	mistake
glm-4.7 #think	358.338B	204k	41	[41, 43]	mistake

Measurement	Params	Context	CC	95% CI	Bounded by
`glm-4.7-flash #nothink`	31.221B	202k	15	[15, 15]	mistake
`glm-4.7-flash #think`	31.221B	202k	30	[30, 30]	mistake
`glm-5 #nothink`	753.864B	204k	40	[31, 50]	mistake
`glm-5 #think`	753.864B	204k	40	[40, 40]	mistake
`glm-5-turbo #nothink`	–	202k	40	[21, 50]	mistake
`glm-5-turbo #think`	–	202k	38	[37, 41]	mistake
`glm-5.1 #nothink`	753.864B	204k	50	[35, 51]	mistake
`glm-5.1 #think`	753.864B	204k	40	[39, 40]	mistake
`glm-5.2 #nothink`	753.33B	1.04858M	41	[39, 42]	mistake
`glm-5.2 #think`	753.33B	1.04858M	53	[49, 54]	mistake
`glm-5.3`	744B	1.04858M	$\geq$58	[55, 59]	budget
`glm-5v-turbo #nothink`	–	202k	50	[45, 51]	mistake
`glm-5v-turbo #think`	–	202k	42	[42, 42]	mistake

D INSTRUCTION-FOLLOWING AUDIT

The measurement instructs every model to return a single integer with no other text. When we audited every scan probe for compliance, 3,887 of 45,287 responses (8.6%) deviated from this format. Deviations concentrate beyond each model's counting boundary, indicating that losing the count disrupts not only numerical accuracy but also the control needed to maintain the requested response format.

Table 5 details the distribution of these failure modes. Blank or whitespace-only outputs are the most common deviation, and inspection of the response logs shows that most of them are budget exhaustion: 2,041 of the deviating responses terminated at the fixed output cap with no final answer emitted, the signature of a reasoning stream that never terminates. Unsolicited step-by-step counting is the second most common deviation, in which a model abandons the requested zero-shot format and attempts to count in generated text. The remaining categories are prompt echoes, code or markdown formatting, leaked reasoning markup, and answers embedded in prose.

Table 6 lists the models deviating most often, as a share of their own probes. The heaviest deviators are reasoning-configured models probed far beyond their boundary: one frontier model accounts for 712 deviations across its two configurations, and one small reasoning model deviates on 88% of its probes. Blank outputs are therefore partly a budget artifact of reasoning configurations, several of them from one provider whose extended thinking counts against the cap; the claim that losing the count disrupts format control rests on the remaining categories, which concentrate beyond the boundary in non-reasoning configurations as well. Representative responses are reproduced in Table 7: after losing the count, models copy the sequence back, wrap the answer in code fences, leak reasoning scaffolds into the reply, or append conversational text to the number.

Table 5: Distribution of non-single-number responses across all 45,287 scan probes (3,887 deviating responses, 8.6%).

Category	Count	Share of probes
Blank / whitespace	2,183	4.82%
Step-by-step counting or explanation	1,401	3.09%
Reasoning-tag or hidden-CoT markup	63	0.14%
Code / markdown formatted output	61	0.13%
Answer with extra prose / instruction leakage	49	0.11%
Prompt echo / copied sequence	41	0.09%
Other mixed numeric format	35	0.08%
Number with stray punctuation / formatting	33	0.07%
Text with no parseable number	21	0.05%

Table 6: Models most frequently deviating from the single-integer format, with each model's deviations as a share of its own probes.

Model	Deviations	Share of own probes
claude-opus-4.6	712	35%
claude-sonnet-4.6	317	51%
claude-sonnet-4.5	132	18%
qwen3.5-flash-02-23	100	50%
minimax-m2-her	88	88%
claude-opus-4.1	76	5%
deepseek-v4-pro-0813	76	27%
gpt-5.1-codex	72	38%
gpt-3.5-turbo-instruct	67	67%
grok-4.20	66	29%

Table 7: Representative deviating responses; L is the probed length.

Category	Model	L	Response
Step-by-step counting	claude-sonnet-5	55	Let me count them one by one.\n\n55
Code / markdown output	gemini-3-flash-preview	303	```json 303 ```
Prompt echo	mistral-medium-3-5	49	a, a, a, a, a, a, a a, a, a, … (sequence copied back)
Leaked reasoning markup	qwen3.5-flash-02-23	5	Thinking Process: 1. Analyze the Request: Task: Count occurrences…
Mixed numeric format	deepseek-v4-pro-0813	110	110.110
Prose / instruction leakage	minimax-m2-her	97	87 \n user name=talkior \n No. Try again.

E ANCHORED-EXECUTION CONTROL

The counting failures in the main text could in principle reflect a general inability to execute long sequential procedures. This control shows that they do not. We constructed four subsidiary tasks whose intermediate value, unlike a count over identical items, can be expressed in language at every step, and measured each alone under the counting protocol: the same models, output budget, stop rules, and estimator, with the task's load taking the place of sequence length.

The four tasks are ordered by the intermediate value they require. *Lookup* provides a table of word-to-number pairs and asks for one entry; the answer requires a single attention reference and no intermediate value. *Pointer chase* provides a shuffled chain of "w_i points to w_j" links and asks for the number associated with the final word, requiring as many dependent references as the chain is long. *Running total* provides a start value and a list of signed increments to apply in order. *Automaton* provides the transition table of a four-state machine and an input word stream, and asks for the final state. Keys and symbols are common English words drawn from the run's seeded generator, so no rare-token effects enter (Appendix A), and the correct answer is never equal to the load, so a task cannot be solved by counting its own input lines. Lookup was probed at table sizes 50 to 1000 in steps of 50; the other tasks were probed densely from load 1 to a ceiling of 320, chosen to exceed the counting capacities of two of the three models and to reach nine tenths of the third's (352).

Table 8 gives the result. Lookup is exact at every table size to 1000 for all three models. The three sequential tasks are executed essentially without collapse over the probed range: seven of nine measurements had not reached a transition at the probe limit, and of the two declared transitions, the one at 299 links belongs to the model with a CC of 89 and the one at 276 updates to the model with a CC of 352, the latter within a fifth of its counting capacity. The model with a counting capacity of 89 follows pointer chains of roughly 300 links, and the one with a capacity of 115 executes 320 automaton transitions at 97% accuracy.

Table 8: Anchored-execution control. Each cell gives the fitted capacity of one subsidiary task, with the stable-regime accuracy in parentheses; $\geq$ marks measurements that had not collapsed at the probe ceiling. Counting capacities for the same three models are 89, 115, and 352.

Level	Task	claude-sonnet-5	gpt-5.6-terra	gemini-3.7-flash
1	Lookup	$\geq$ 1000 (1.00)	$\geq$ 1000 (1.00)	$\geq$ 1000 (1.00)
2	Pointer chase	299 (0.94)	$\geq$ 320 (0.99)	$\geq$ 320 (1.00)
3	Running total	$\geq$ 320 (1.00)	$\geq$ 320 (0.95)	276 (0.89)
4	Automaton	$\geq$ 320 (0.88)	$\geq$ 320 (0.97)	$\geq$ 320 (1.00)

The contrast narrows what the counting bound covers. These tasks demand long sequential execution, dependent references, and order-sensitive values, and the models perform them several times beyond

their counting capacities. What distinguishes counting is not that its running value cannot be written, since a count is one integer, but that a written value cannot be re-attached to the item where it was written: each update line, pointer link, or input word is a distinct textual anchor that a reasoning trace can name, process, and verify, whereas the items of a homogeneous sequence are indistinguishable, so a trace cannot express its own position without re-deriving the count. The bound measured by CC is therefore specific to running values that cannot be tied back to the input once written, which is also why enabling reasoning leaves the median counting capacity unchanged while multiplying token expenditure (Fig. 3B): a chain of thought can only extend values it can name.

F VERBATIM COPYING THROUGH A TOOL CALL

The counting failure could in principle lie in the arithmetic rather than in the representation of the sequence. This control removes the arithmetic. The model is given the counting measurement's sequence and a single tool, `record_sequence`, and is instructed to call the tool with the sequence copied verbatim; the tool call is forced, and the number of items in the recorded argument is compared with the number in the prompt. No counting is requested, and generation can re-anchor on the source at every emitted token, so this is the easiest traversal of the sequence the model can be asked to perform.

Copy fidelity is perfect well beyond each model's counting capacity and then collapses at a finite length, fitted with the estimator of Appendix B. One model copies exactly to a boundary of 420 against a CC of 352 and fails by deleting whole round blocks, so that recorded sequences run 50 to 200 items short; the other copies exactly to 448 against a CC of 89 and fails in the opposite direction, duplicating content until the record runs up to 475 items long. Both signatures are the counting taxonomy's own failure modes, the round-block offsets of Fig. 2D and the runaway generation of Appendix G, now appearing in a task with no arithmetic in it.

The copy boundary is 1.2 and 5.0 times CC respectively, consistent with the landmark account: reproduction supplies a per-token anchor that pure counting lacks, and the more a task lets generation re-anchor, the further it extends. The boundary is finite and far below the context window, and beyond it a model handed a sequence cannot pass it onward intact, so any downstream computation that consumes the data, counting included, receives a corrupted payload. The limit lies in reproducing exact multiplicity, whether it arises while reading the input, while tracking position in it, or while deciding when to stop generating, which this control does not separate; the copy and counting boundaries also differ, so the two tasks do not fail at one length. The practical form of the failure for agents is that beyond a few hundred repeated items, a tool-call payload of the very data in the prompt is silently wrong. A tool interface that passes a reference to the stored input rather than a regenerated payload avoids this by construction, so the result limits what a model can do with data it must itself reproduce.

G STRUCTURE OF REASONING-TRACE FAILURES

To examine why additional test-time computation does not rescue counting, we retrieved the stored reasoning traces of the scan itself from the serving provider's generation archive, rather than running a separate probe set. The archive returns the full stored record, including the reasoning channel, for 117 of the 122 reasoning-enabled measurements; the remaining five expose no reasoning text. We sampled up to sixteen generations per model, spaced evenly over prompt size, and identified each record by parsing its stored prompt, giving 1,728 counting probes with reasoning attached. This retrieval path is independent of the original run records and reproduces the two-regime structure of Appendix B.

Every incorrect trace (697) and every correct trace near or above capacity (305) was read and classified by the strategy the trace attempts, the structure of its failure, and whether the returned integer can be derived from the visible reasoning. Correct traces below capacity were sampled rather than exhaustively read; they are short executed enumerations or terse assertions, and their surface form does not separate them from failures a few dozen lengths later: several models emit the same template on both sides of the boundary. Exposed reasoning is behavioral evidence rather than a faithful record of internal computation, and for some providers it is a summarized paraphrase of a hidden trace, so the classification describes what the trace displays.

The visible trace rarely computes the answer. In 434 of the 697 incorrect traces the returned integer cannot be derived from the visible reasoning: the trace is narration without arithmetic, is truncated, contradicts the answer it accompanies, or ratifies a number that appeared before any counting began. The same decoupling holds for successes. Of the 80 correct answers above capacity, 30 contain no executed enumeration at all, and several models are explicit about it. One frontier model, correct at 198 items with a fitted capacity of 132, abandons its tally at 93 and writes "Without a precise way to verify by actually splitting the string, I'll settle on 198 as my count." The count was available without the counting; the trace documents a readout, not a computation. The converse also occurs well below capacity: another model, wrong at 37 items against a fitted capacity of 119, spends its whole trace asserting diligence ("I've performed multiple checks to ensure the accuracy of this count, which now stands at 36") without a digit of visible work.

Table 9 organizes the 697 incorrect traces into four groups. In the first, the answer is decoupled from the trace outright: either a candidate value is named before any counting and every subsequent check is generated to agree with it (including the agreement of supposedly independent methods, which all issue from the same decoder and therefore confirm rather than test), or the trace is method narration containing no work to check. In the second and largest group, enumeration is executed but measures the model's own generation rather than the prompt: the trace re-emits the sequence or a numbered ladder and reports where its own copy stops. Its mildest form counts a silently drifted transcription; once detached from the input, the ladder either runs away (one trace enumerates to 1,000 before remarking "Okay, that was a simulation of counting to 1000. I need to count the *actual* string provided") or stops at a round total, which is the trace-level mechanism of the numerical attractors in Fig. 2E. The third group contains genuine counts that slip: the interior blocks of a grouping scheme are handled correctly and the final partial block is mis-sized, producing the off-by-k errors of Fig. 2D, or a valid identity such as items-equals-commas-plus-one is applied to an operand the model never measured, so correct algebra is applied to a guessed value. The fourth group fails to commit or to deliver: the trace re-derives the count in near-identical passes, sometimes reaching the correct value and then abandoning it, until the budget is exhausted, or the computed integer never reaches the visible completion, left in the reasoning channel or lost to formatting. These are the blank and budget-bound outputs of the instruction-following audit (Appendix D) seen from inside.

Table 9: **Failure structure of 697 incorrect reasoning traces.** Stored traces retrieved for 117 reasoning-enabled measurements and classified as described in the text. Groups and modes are assigned per trace; each trace counts once.

Group	Mode	Traces
Answer decoupled from trace	anchored answer, ratifying verification	57
	narration without work	47
Enumeration of own generation	counts a drifted self-transcription	26
	runaway ladder past the true end	96
	terminates at a round attractor	125
Slips in genuine counts	mis-sized final block (off-by-k)	95
	valid formula, unmeasured operand	56
Commitment and delivery	verification livelock to the budget	140
	answer never reaches the completion	55

Token expenditure runs the wrong way for a simple compute-bound account, though only as a descriptive pattern, since a hard probe can both spend more and fail. Above capacity, the median correct trace spends about 2,100 reasoning tokens, the median wrong-integer trace about 2,700, and the median trace returning no integer about 6,400, nearly all of the last group terminating at the token cap. Long scratchpads do not recover a collapsed count; they elaborate it.

This structure is consistent with the anchored-execution control (Appendix E) and explains the matched-pair null of Appendix G.1. A chain of thought can extend any running value it can name, and enumeration inside a trace is itself generation: its stopping point is set by whatever per-item total the model still holds, and when that state is exhausted the ladder defaults to the model's priors, a round total, a remembered schema, or the continuation of its own list. Naming the counting rule, which the traces do almost universally, does not supply the total the rule needs.

G.1 EVERY HYBRID PAIR, REASONING ON AGAINST OFF

The study includes 90 hybrid pairs, that is, the same weights measured once with reasoning enabled at medium effort and once with reasoning disabled. Whether disabling took effect is verified from the measured reasoning tokens rather than assumed from the request, because several providers apply reasoning regardless of the requested setting. Only the 77 verified contrasts, listed below, enter Fig. 3B. The change in capacity is heterogeneous around its zero median: the interquartile range is -3 to $+7$ items, seven pairs gain more than half their capacity and two lose more than a third. Sixteen of the 77 pairs have a reasoning member whose scan ended on exhaustion of the fixed output budget rather than on wrong answers, so under the catalogue protocol its capacity is a lower bound; these concentrate in one vendor line. To see whether the budget was what bound them, we rescanned exactly those sixteen configurations with the cap raised fourfold, to 32,768 tokens, and everything else unchanged (Table 10). Six resolve at wrong answers under the larger budget; eight still end on budget exhaustion and two at the scan's end, so these ten remain budget-bounded observations. The eight spend the full 32,768 tokens on sequences of 40 to 100 items, an order of magnitude more than enumeration would need: the verification livelock of Table 9 at a larger budget. Substituting the rescanned values for the bounds would move the median change over the 77 pairs from 0 to $+1$ item (39 pairs gaining, 10 unchanged, 28 losing); the catalogue and Fig. 3B keep the fixed-budget protocol. The gains concentrate in the members of that line whose silent capacities are 10 to 15 items and whose reasoning capacities reach 70 to 85 at the larger budget, and the largest loss is a model whose reasoning traces exhaust the budget at every length beyond 40. Raising the cap leaves the median contrast nearly unchanged, though ten of the sixteen rescans still do not identify an uncensored reasoning capacity.

Table 10: **Budget-bounded reasoning configurations rescanned at a 32,768-token cap.** The sixteen verified hybrid pairs whose reasoning member exhausted the catalogue's 8,192-token cap before failing on the count, rescanned with the cap raised fourfold and everything else unchanged. CC_{off} is the no-reasoning capacity, CC_{8k} the catalogue bound, CC_{32k} the rescanned value. Bound: how the 32k scan ended. Hits: probes that exhausted 32,768 tokens, and the smallest count at which it first happened.

Model	CC_{off}	CC_{8k}	CC_{32k} [95% CI]	Bound	Hits (first at)
`claude-sonnet-4.6`	320	$\geq$264	$\geq$331 [331, 332]	budget	5 (294)
`deepseek-chat-v3.1`	15	$\geq$28	$\geq$84 [84, 96]	budget	3 (55)
`deepseek-v3.1-terminus`	13	$\geq$60	$\geq$70 [9, 75]	budget	9 (66)
`deepseek-v3.2-exp`	10	$\geq$58	$\geq$80 [80, 80]	scan end	2 (68)
`deepseek-v3.2`	12	$\geq$70	$\geq$80 [80, 80]	scan end	1 (66)
`deepseek-v4-flash-0731`	44	$\geq$51	$\geq$64 [52, 70]	budget	16 (39)
`deepseek-v4-flash-vision-exp`	139	$\geq$76	$\geq$106 [102, 114]	budget	27 (89)
`deepseek-v4-flash`	52	$\geq$72	$\geq$85 [83, 87]	budget	9 (62)
`deepseek-v4-pro-0813`	36	$\geq$96	121 [119, 140]	resolved	17 (98)
`deepseek-v4-pro`	36	$\geq$72	$\geq$76 [70, 86]	budget	21 (44)
`gemma-4-26b-a4b-it`	83	$\geq$41	$\geq$40 [39, 40]	budget	23 (41)
`glm-4.6`	41	$\geq$53	45 [44, 45]	resolved	1 (61)
`gpt-5.6-luna`	46	$\geq$45	37 [34, 52]	resolved	8 (55)
`kimi-k2.6`	76	$\geq$56	93 [85, 105]	resolved	2 (53)
`qwen3.5-35b-a3b`	42	$\geq$50	53 [28, 54]	resolved	5 (54)
`qwen3.5-9b`	35	$\geq$25	26 [26, 26]	resolved	3 (27)

Figure 6 shows where the tokens go. In the stable regime, completion tokens grow roughly linearly with the count, at 9 to 230 tokens per item across six representative configurations, against the three or four tokens per item that writing the items out once would cost: the traces enumerate several times, in groups, with subtotals and checks. At the boundary the growth turns into a jump. The median spend over the ten probes before the fitted boundary against the ten after it rises from 1,400 to 21,000 tokens in one model and from 16,000 to the full 32,768 in another, and the probes that hit the cap sit immediately past the boundary rather than at the far end of the scan. Raising the cap fourfold moves the wall, not the boundary: the same configurations exhaust it at nearly the same counts.

The thirteen excluded pairs are the anomalies, collected in Table 11: in eight pairs the provider ignores the disable request outright, and five pairs spend no reasoning tokens in either configuration. A ninety-first two-configuration model, `minimax-m1`, is absent from both counts because neither of its scans resolved a boundary (Appendix C).

`claude-haiku-4.5`, `claude-opus-4`, `claude-opus-4.1`, `claude-opus-4.5`, `claude-opus-4.6`, `claude-opus-4.8`, `claude-opus-5`, `claude-sonnet-4`, `claude-sonnet-4.5`, `claude-sonnet-4.6`, `claude-sonnet-5`, `deepseek-chat-v3.1`, `deepseek-v3.1-terminus`, `deepseek-v3.2`, `deepseek-v3.2-exp`, `deepseek-v4-flash`, `deepseek-v4-flash-0731`, `deepseek-v4-flash-vision-exp`, `deepseek-v4-pro`,

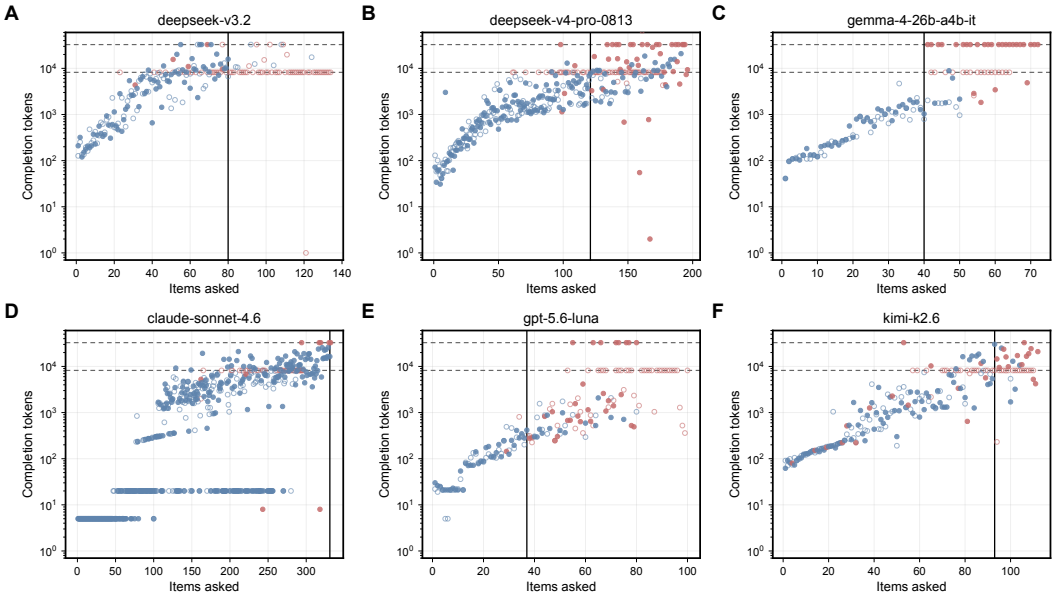

Figure 6: **Completion tokens against items asked at two output caps.** Six reasoning configurations from the 32,768-token rescan (**A** to **F**; filled markers) overlaid on their catalogue scans at the 8,192-token cap (hollow markers), exact answers in blue and inexact in red. Dashed rules mark the two caps and the vertical rule the boundary fitted at the larger cap. Tokens grow roughly linearly with the count inside the stable regime and jump to the cap immediately past the boundary.

`deepseek-v4-pro-0813`, `gemini-2.5-flash`, `gemini-2.5-flash-lite`, `gemini-3-flash-preview`, `gemini-3.1-flash-lite`, `gemini-3.1-flash-lite-preview`, `gemma-4-26b-a4b-it`, `gemma-4-31b-it`, `mistral-medium-3-5`, `mistral-small-2603`, `kimi-k2.5`, `kimi-k2.6`, `kimi-k3`, `gpt-5.1`, `gpt-5.2`, `gpt-5.3-codex`, `gpt-5.4`, `gpt-5.4-mini`, `gpt-5.4-nano`, `gpt-5.5`, `gpt-5.6-luna`, `gpt-5.6-luna-pro`, `gpt-5.6-sol`, `gpt-5.6-sol-pro`, `gpt-5.6-terra`, `gpt-5.6-terra-pro`, `qwen3-235b-a22b`, `qwen3-8b`, `qwen3-max-thinking`, `qwen3.5-122b-a10b`, `qwen3.5-27b`, `qwen3.5-35b-a3b`, `qwen3.5-397b-a17b`, `qwen3.5-9b`, `qwen3.5-plus-02-15`, `qwen3.5-plus-20260420`, `qwen3.6-27b`, `qwen3.6-35b-a3b`, `qwen3.6-flash`, `qwen3.6-max-preview`, `qwen3.6-plus`, `qwen3.7-flash`, `qwen3.7-max`, `qwen3.7-plus`, `qwen3.8-27b`, `grok-4.20`, `grok-4.3`, `glm-4.5`, `glm-4.5-air`, `glm-4.5v`, `glm-4.6`, `glm-4.6v`, `glm-4.7`, `glm-4.7-flash`, `glm-5`, `glm-5-turbo`, `glm-5.2`, `glm-5v-turbo`.

Table 11: Hybrid pairs excluded from the matched comparison. Token columns give the mean completion tokens per probe over the lengths that both configurations count reliably. *Knob ignored* marks providers that apply reasoning regardless of the request, so the pair amounts to two measurements of one configuration; *no reasoning* marks pairs whose reasoning-enabled configuration spent no reasoning tokens either.

Model	CC_{off}	CC_{on}	ΔCC	tok_{off}	tok_{on}	Class
`deepseek-r1-distill-llama-70b`	57	58	+1	3,508	3,472	knob ignored
`minimax-m3`	51	56	+5	56	173	knob ignored
`gpt-5.1-codex-mini`	50	50	+0	1,623	1,182	knob ignored
`qwen-plus-2025-07-28:thinking`	58	58	+0	6,007	5,715	knob ignored
`qwen3-14b`	27	31	+4	480	737	knob ignored
`qwen3-30b-a3b`	37	35	-2	349	794	knob ignored
`qwen3-32b`	22	26	+4	298	316	knob ignored
`glm-5.1`	50	40	-10	32	447	knob ignored
`claude-opus-4.7`	158	156	-2	6	6	no reasoning
`qwen-plus-2025-07-28`	54	54	+0	2	2	no reasoning
`qwen3-coder-plus`	50	50	+0	2	2	no reasoning
`qwen3-max`	60	60	+0	2	2	no reasoning
`qwen3.5-flash-02-23`	42	46	+4	2	2,015	no reasoning

H THE EXECUTION LADDER

The ladder extends the counting measurement toward the grid execution that abstract-reasoning tasks demand, in three levels of increasing structure: counting within a row, executing a rule on a row, and executing a rule on a grid. Every level keeps the measurement's full contract: the model returns a single integer, scored by exact match, and the stimulus is constructed so that the correct answer

equals the scanned load; the two-regime estimator of Appendix B therefore fits each capacity in the same unit as CC, with the same stopping rule.

H.1 TASKS AND PROBE CONTRACT

The stimuli are rows or grids of space-separated digits, the serialization abstract-reasoning grids use in text. The row-counting task probes the structure of a row as it stands: the row holds exactly N cells of a target digit, scattered in short runs among distractor runs whose number and lengths vary widely, and the model reports the cell count. The construction decorrelates row length from the answer, so gross input size carries no information. The row-rule tasks state a transformation rule and ask for the target-digit count after the rule is applied to the row: every run of the target digit grows by one cell at each end; every cell of a second digit is recolored to the target; or every single-cell run is deleted. The transformed row never appears in the context, so the probed quantity exists only if the rule was executed internally. The grid-rule tasks apply the recoloring and deletion rules to two-dimensional grids of rectangular regions on a uniform background, where connectivity is spatial rather than linear. The ladder is six tasks: one row-counting task, three row-rule tasks, and two grid-rule tasks.

Each probe must satisfy two requirements. It must be *global*: the probed quantity is a function of the whole transformed state, so no partial evaluation answers it (a single-cell lookup would suffice for a mirror rule, which is why permutation rules are excluded). It must be *non-invariant*: the quantity must actually change under the rule and not be derivable from a cheap invariant of the input (a color count under a color-preserving rule would be). Non-invariance excludes cheap invariants, not closed forms: the recolor answer equals the sum of two input color counts, and the grow and delete answers are sums over runs. The ladder therefore measures task modeling under a stated rule, which is its target, and does not establish that the model constructs the transformed row or grid. Every generator chooses its run or region structure so the post-rule answer equals the scanned load exactly, and every stimulus is verified against an independent simulator of the rule before it is sent.

H.2 ONE MODEL, ALL SIX TASKS: CAPACITY COUNTS OBJECTS

We first measured the full ladder on `gemini-3.7-flash` (Table 12). The capacities span 238 to 340 in answer units against a CC of 352. Converting each boundary into the number of objects the model had to hold (runs in a row, connected regions in a grid, measured from the generators at the boundary load) leaves 132 objects for the pure-counting probe and 70 to 103 under a rule. The two rules that appear in both dimensions allow a direct comparison. Moving the recoloring rule from rows to grids raises the answer-unit capacity from 253 to 340 while the object count falls from 103 to 85; moving the deletion rule raises it from 238 to 334 while the object count falls from 82 to 72. The answer-unit capacity rises by a third to two fifths between rows and grids while the object count does not rise at all. For this model and these generators the boundaries correspond to 70 to 132 objects, consistent with object-based aggregation; the experiment does not separately estimate the costs of object number, object size, and rule execution.

Table 12: **Full ladder on `gemini-3.7-flash`.** Fitted capacity with its calibrated interval, and the mean number of objects held at the boundary load. The model's CC is 352 [350, 373].

Level	Task	Capacity	CI	Objects
row count	scattered cells	326	[326, 326]	132
row rule	grow (row)	282	[274, 283]	70
row rule	recolor (row)	253	[247, 255]	103
row rule	delete (row)	238	[225, 287]	82
grid rule	recolor (grid)	340	[338, 371]	85
grid rule	delete (grid)	334	[332, 338]	72

H.3 THE COHORT PROGRAM AND THE RATIO LAW

We then measured all six ladder scans on every model in the ARC-AGI-2 post-release cohort plus `claude-sonnet-5`, 180 raw measurements over 30 models. Scans are bracketed from the pilot: each boundary is predicted as a fixed multiple of the model's CC, and the scan starts at 0.35 times the prediction with a step near one eightieth of it, stopping by the criterion of Appendix B.

Protocol exclusion: `claude-fable-5`. The serving layer returned `content_filter` for 103 of the model's 244 probes, distributed across all six ladder tasks and including a minimal load-8 preflight. These events return no answer, so execution failure cannot be distinguished from provider-side suppression. We retain all raw records as a case of protocol incompatibility but exclude all six measurements, including the scans that otherwise fit a boundary, from the cohort analysis. One filtered preflight is reproduced below (the digit row is line-wrapped for display):

```
System:  You follow instructions exactly.  Reply with one integer and nothing
else.
User:  Apply this rule to the row below:  every cell containing 3 becomes 7.
After applying the rule, how many cells contain the digit 7?
1 1 1 7 2 2 6 6 2 2 0 0 6 6 0 0 0 0 0 6 6 6 6 6 6
1 1 1 7 8 8 5 5 5 5 5 9 9 9 7 7 7 7 6 6 5 5 5 5
5 7 7 5 5 5 5 5 1 1 1 1 1
Provider:  status=completed; detail=content_filter; response=[empty].
```

The resulting analysis cohort contains 174 measurements over 29 models. Of these, 90 resolve inside the scan bracket, and their fitted boundaries track the CC-scaled predictions closely ($\rho = 0.83$ between log prediction and log boundary, median ratio 0.95). This establishes a within-bracket relation; the out-of-range measurements are censored evidence and are treated as such below.

To test how strongly that result depended on the pilot-derived bracket, we extended the raw scans sequentially rather than substituting arbitrary point values for their unresolved outcomes. First, all 46 below-start and 8 mixed scans were continued downward to the smallest admissible load of their task (1 for row counting and 4 for the rule tasks), for 54 lower extensions. After refitting those combined records, the 36 scans still classified as ceiling-stable or mixed were continued upward from their last tested load to a safety ceiling of four times the CC-scaled prediction, stopping earlier whenever the same transition criterion resolved. Fixed-grid probes were issued in batches of three and the criterion was evaluated in load order after each batch, so parallel execution can overshoot the first sequential stopping point by at most two load increments. Every lower and upper probe is retained in the consolidated fit.

The consolidated analysis records what each scan identifies rather than retaining only point estimates. A resolved scan contributes its fitted boundary; a scan already below criterion at the task minimum is left-censored with the minimum as an upper bound; and a scan still stable at the expanded ceiling is right-censored with that ceiling as a lower bound. Nonmonotone mixed scans do not support the ordering assumed by a single transition and are excluded from the primary likelihood, then included using their tested ranges only as a separate interval sensitivity. Blank-dominated scans provide no numerical bound on the boundary. The primary model is a Gaussian censored regression on the log scale, $\log B_i = \alpha + \beta \log P_i + \epsilon_i$, where B_i is the latent execution boundary, P_i the CC-scaled prediction, and $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$. A nonparametric bootstrap resamples models with all six task scans as a cluster. We also report the resolved-only Spearman correlation, a transparent limit-substitution correlation, and the distribution of Spearman correlations obtained by drawing censored boundaries from the fitted truncated distributions; the latter is a conditional imputation range, not a sampling confidence interval.

Extending the scans resolves 139 of the 174 measurements, against 90 inside the original bracket (Fig. 5G). On the expanded records the censored regression gives a slope of 1.00 (95% cluster bootstrap [0.46, 1.51]; likelihood ratio 40.5 against a zero slope, $p = 2 \times 10^{-10}$), so the execution boundary scales in proportion to the CC-scaled prediction, with a residual standard deviation of 1.53 on the log scale. The 139 resolved boundaries sit at a median 0.91 of their prediction and correlate with it at $\rho = 0.56$; substituting each observed censoring limit, or drawing the censored boundaries from their fitted truncated distributions, leaves the correlation between 0.55 and 0.56. Table 13 gives the per-model outcome counts before and after the extension.

The ladder also ranks the models consistently from task to task. On the resolved boundaries the Spearman correlation between any two tasks averages 0.77, and Kendall's coefficient of concordance across the six tasks is $W = 0.83$ on the 17 models resolved on every task; entering each censored scan at its observed bound raises these to 0.83 and $W = 0.86$ on 28 models. Each single task, by contrast, correlates with CC at only 0.4 to 0.7 with bounds substituted (0.3 to 0.7 on resolved boundaries alone). The models therefore agree across the six tasks far more than any task agrees with CC: the family departures of Fig. 5G are departures of a coherent execution capacity from the counting capacity, not task-specific noise.

Table 13: **Model-level bracket sensitivity.** Counts among the six ladder tasks before and after extending scans down to the task minimum and up to $4\times$ the CC-scaled prediction. Each tuple is resolved/left-censored/right-censored/mixed/blank (R/L/U/M/B); the last column is the median measured/predicted ratio among resolved expanded scans.

Model	Original R/L/U/M/B	Expanded R/L/U/M/B	Median ratio
`claude-opus-4.6`	4/1/0/0/1	5/0/0/0/1	0.5751
`claude-opus-4.7`	2/4/0/0/0	4/2/0/0/0	0.4362
`claude-opus-4.8`	5/1/0/0/0	5/1/0/0/0	0.6579
`claude-opus-5`	4/0/1/1/0	6/0/0/0/0	1.9263
`claude-sonnet-4.6`	2/4/0/0/0	6/0/0/0/0	0.3133
`claude-sonnet-5`	4/2/0/0/0	6/0/0/0/0	0.5336
`deepseek-v3.2`	5/0/1/0/0	6/0/0/0/0	1.3992
`deepseek-v4-flash-0731`	2/0/3/1/0	6/0/0/0/0	1.7346
`deepseek-v4-pro-0813`	6/0/0/0/0	6/0/0/0/0	0.9753
`gemini-3.1-pro-preview`	6/0/0/0/0	6/0/0/0/0	0.6691
`gemini-3.5-flash`	6/0/0/0/0	6/0/0/0/0	0.3855
`gemini-3.5-flash-lite`	6/0/0/0/0	6/0/0/0/0	0.7739
`gemini-3.6-flash`	6/0/0/0/0	6/0/0/0/0	0.5054
`gemini-3.7-flash`	6/0/0/0/0	6/0/0/0/0	0.997
`glm-5`	5/0/1/0/0	6/0/0/0/0	1.4253
`glm-5.2`	2/0/4/0/0	6/0/0/0/0	2.0963
`gpt-5.2`	3/0/2/1/0	6/0/0/0/0	1.9172
`gpt-5.4`	0/6/0/0/0	4/2/0/0/0	0.0628
`gpt-5.4-mini`	5/0/1/0/0	6/0/0/0/0	1.7521
`gpt-5.4-nano`	0/6/0/0/0	2/4/0/0/0	0.2117
`gpt-5.5`	1/5/0/0/0	6/0/0/0/0	0.1146
`gpt-5.6-luna`	0/6/0/0/0	1/5/0/0/0	0.187
`gpt-5.6-sol`	0/0/6/0/0	4/0/2/0/0	3.0545
`gpt-5.6-terra`	0/6/0/0/0	3/3/0/0/0	0.0735
`grok-4.5`	2/0/4/0/0	3/0/3/0/0	0.9643
`grok-4.6`	1/0/5/0/0	1/0/5/0/0	1.9358
`kimi-k2.5`	1/0/1/4/0	2/0/2/2/0	1.1815
`kimi-k3`	0/5/0/1/0	3/2/0/1/0	0.2673
`minimax-m2.5`	6/0/0/0/0	6/0/0/0/0	1.555

The residual departures are systematic at the level of model families, and each class is informative. Nineteen scans still fail below the smallest admissible load, concentrated in four OpenAI lines with one or two scans each from three other models; these return well-formed but wrong integers at loads as small as four, placing their capacities below anything the generators can construct. Twelve scans are still stable at four times the prediction, concentrated in two xAI models and one each from Moonshot and OpenAI, so that within one vendor line one release exceeds four times its predicted boundaries while a sibling falls below the task minimum. Three scans, all on one line, have no declarable transition over the range tested. One is dominated by empty replies after exhausting the fixed output budget. Under the fixed-budget protocol, budget exhaustion is an execution failure rather than missing data; its cause is retained separately from the scan's transition classification.

Format, not capacity, at the lower departures. The scans that fall below the task minimum do not read as capacity measurements. Across their probes at loads of twenty or fewer they never return an empty reply and almost never a malformed one, against a few percent of each elsewhere in the cohort: these models follow the output contract more closely than the cohort average, not less. Their answers also sit close to the truth, 73% within three of it. A model wrong by one or two at four objects may be misreading what the stimulus asks rather than failing to hold a four-object total; we treat this as a hypothesis, since closeness of the answer does not establish it and we have not tested it by changing the serialization or adding examples while holding the computation fixed. If it holds, it would be the factor separating the two ARC-AGI-2 cohorts of Fig. 5F: familiarity with a serialization that does not occur in natural text, binding before capacity does.

The nineteen scans split in two by the model's ARC-AGI-2 score relative to its capacity, and the split is the informative part. The four OpenAI lines sit below the ladder floor and also below the ARC-AGI-2 score their capacity predicts: on both tasks the same thing limits them. The other three

models sit above it: Claude Opus 4.7 and 4.8 and `kimi-k3` score higher on ARC-AGI-2 than their counting capacity predicts, yet fail a ladder task at four objects. They execute the benchmark's own serialization well and a near neighbor of it not at all, which is what adaptation to a fixed format, rather than a general gain, would produce.

Two vendor lines make this a trajectory rather than a cross-sectional accident. Across three successive Claude Opus releases the measured capacity falls from 622 to 133 while the ARC-AGI-2 score rises, and the residual moves from below the capacity-predicted score to well above it; the first release has no floor failure and the two later ones do. From `kimi-k2.5` to `kimi-k3` the capacity falls from 73 to 54 while the score rises more than fourfold, and again only the successor fails at the floor. Serving prices are identical across the Opus releases and higher for the later Kimi model, so this is not a substitution of a smaller, cheaper served model.

Both cells of the dissociation are occupied. Three models are strong on ARC-AGI-2 for their capacity and fail the ladder floor; five others, from four vendors, are weak on ARC-AGI-2 for their capacity and clear it. Familiarity with a format and the capacity to model a task under it may therefore be separable abilities rather than two views of one, a hypothesis the untested format intervention would settle, and it would explain why the pooled association between them is null across the cohort: the pooled estimate averages two populations that depart from the scaling for opposite reasons.

H.4 Interference protocol and results

The interference experiment of Fig. 3C scans counting while the model carries one ladder task at a fixed load. The prompt presents the load first and the counting sequence second, and the model replies with two integers, the load's answer and then the count, so the counting boundary is fitted by the unchanged parser while the load answer is graded separately from the raw reply. Loads are set per model from its own measured solo capacities: for `gemini-3.7-flash`, grids of 26, 51, 76, and 102 regions (30–120% of its 85-object grid capacity) and row loads at 60% of the row-counting and row-rule capacities; for `claude-sonnet-5`, grids of 6 to 25 regions and row loads of 8 to 9 runs on the same fractions of its own capacities. The design has two controls: a control grid of matched token volume whose first integer is printed in the prompt, and a pointer chain of matched size whose running value is fully expressible in language. `gpt-5.6-terra` was measured and excluded: its calibration scans fell below their starts, so its derived loads were one to five regions, effectively no load, and the run is archived rather than interpreted.

Table 14: **Counting under calibrated concurrent loads.** Fitted counting boundary under each load, its ratio to the solo CC, and the accuracy of the load task's own answer over the scan.

Model (solo CC)	Load	CC under load	Ratio	Load accuracy
`gemini-3.7-flash` (352)	control	342	0.97	1.00
	pointer	370	1.05	1.00
	row 60%	290	0.82	0.95
	rule 60%	226	0.64	0.91
	grid 30%	322	0.91	0.97
	grid 60%	278	0.79	0.92
	grid 90%	270	0.77	0.95
	grid 120%	266	0.76	0.96
`claude-sonnet-5` (89)	control	91	1.02	1.00
	pointer	63	0.71	1.00
	row 60%	89	1.00	0.96
	rule 60%	64	0.72	0.92
	grid 30%	85	0.96	0.79
	grid 60%	89	1.00	0.68
	grid 90%	87	0.98	0.66
	grid 120%	89	1.00	0.68

Table 14 gives the full results. The matched-length control moves neither model, so prompt volume alone consumes nothing. `gemini-3.7-flash` shows graded displacement: capacity falls monotonically with grid load and falls furthest under the rule load, while it performs the load tasks at 91–97% throughout. `claude-sonnet-5` allocates instead of sharing: under grid loads its counting stays at solo level while its grid answers drop to 66–79%, and it pays for the two loads it does perform faithfully, losing 28% under the rule load and 29% under the pointer chain. The pointer result refines the anchored-execution control of Appendix E: a task whose running value is expressible in language can be executed far beyond CC, but executing it concurrently is not free;

the cost appears in whatever else must be held. Across both models the displacement tracks the computation the model actually performs, never the tokens the load adds, which is consistent with a shared resource, although two models and single scans do not distinguish one shared limit from other forms of dual-task interference.

I CROSS-MODEL MECHANISTIC PROBE SUITE

Six open-weight instruction models were probed end-to-end on the counting task with one battery: capacity scans in immediate-answer and native reasoning modes, cross-validated linear readouts of the residual stream at four depths, causal interventions at the decode point, across item positions, and at the boundary between a prefix and a continuation, and a count-versus-length deconfound. The set spans five vendors, dense and mixture-of-experts architectures, and 20B to 120B parameters: `gpt-oss-20b` (24 layers), `gpt-oss-120b` (36), `gemma-4-26b-a4b-it` (30), `muse-glimmer-30b` (52, dense), `qwen3.6-35b-a3b` (40), and `llama-3.3-70b-instruct` (80, dense). Four residual sites per model are hooked at about one quarter, one half, three quarters, and the final layer. All runs use greedy decoding and the paper's prompt through each checkpoint's own chat template, on one 192-GB workstation; `gpt-oss-120b` runs quantized through an MLX backend. The two-regime estimator applied to these local scans reproduces each model's catalogued CC to within a few items, so the probe results are read against the paper's capacities. Immediate-answer mode, in which the model emits the integer with no intervening text, is the probe protocol: it is the regime in which the decoder reads the state that the probes measure and edit.

Each subsection supports one claim of Section 4 and one part of Fig. 4: the count is readable at every depth and most exactly early (panels B to D); the decode point does not read the linear coordinate (the control in E); the output channel is the decoder's own answer-token subspace and is not counting-specific (F); the accumulation is read at mid-depth across item positions (E); the boundary state carries the count forward and carries the state of an object history (G); the count is readable but not routed on mixed sequences (H); and reasoning modes change capacity in both directions (I).

I.1 THE COUNT IS READABLE AT EVERY DEPTH, MOST EXACTLY EARLY

Readouts are ridge regressions from the decode-point residual state to the true count, fitted on the stable regime with folds that hold out whole counts. Cross-validation is essential: hidden sizes exceed the number of stable counts, so an unregularized fit interpolates and in-sample R^2 is vacuous. Held-out R^2 is 0.94 to 0.999 at every site of every model, with root-mean-square errors of 0.2 to 3.2 items that grow two- to six-fold from the earliest to the last hooked layer, and residuals that are strongly autocorrelated across adjacent counts (ρ 0.94 to 0.99), as an analog accumulator would produce. Linear noise exceeds half an item at most sites even in mid-regime, so a linear readout alone would not support exact counting at the observed capacities.

Scoring the rounded readout by exact match on the same prompts as the behavior (Table 15, Fig. 8) gives the exactness the main text reports. In the stable regime the rounded readout at its best site is within 0.07 of the model's own exactness in three models and 0.19 to 0.38 above it in the other three (0.69 to 0.96 against 0.31 to 0.84), and it is highest at the early sites, falling by 0.2 to 0.5 from the first hooked site to the last in every model. Figure 7 extrapolates the readouts past each model's boundary: the early-layer readouts continue to advance with the true count while the last-layer readouts stop and turn against it. Beyond CC no site reads exactly (at most 0.11), and the readout is less exact there than the model's own residual answers, so what persists past the boundary is magnitude information rather than an intact exact count. The teacher-forced margin of the correct answer first turns negative at the length of the first behavioral error in all six models.

Table 15: **Exact match of the rounded readout against the model.** Exact-match and within-one-item rates of the rounded ridge readout at its best site (count-held-out folds) and of the model's own answer on the same prompts, in the stable regime, from 1.0 to 1.25 times CC, and on the mixed grids of Section I.7 (readout fitted within the grid). Two conventions appear in the model columns and are kept distinct: the model's exact rate is the teacher-forced first-token match on the suite's multi-prompt cache, the same convention as the readout columns, which is why it can fall below one half (`gpt-oss-120b`) and is not the scan's full-answer rate that defines the stable regime; the model's within-one rate is the full-answer rate from the scan records.

	Stable regime		1.0 to 1.25 CC		Mixed grids	
Model	readout	model	readout	model	readout	model
`gpt-oss-20b`	0.83 / 0.97	0.64 / 0.94	0.09 / 0.29	0.16 / 0.56	0.29 / 0.83	0.26 / 0.41
`gpt-oss-120b`	0.69 / 0.98	0.31 / 0.83	0.06 / 0.18	0.09 / 0.33	0.19 / 0.62	0.10 / 0.31
`gemma-4-26b`	0.77 / 1.00	0.79 / 1.00	0.04 / 0.20	0.15 / 0.26	0.21 / 0.43	0.29 / 0.50
`muse-glimmer-30b`	0.81 / 0.98	0.77 / 1.00	0.05 / 0.13	0.14 / 0.43	0.26 / 0.69	0.10 / 0.14
`qwen3.6-35b`	0.77 / 0.97	0.84 / 1.00	0.08 / 0.28	0.17 / 0.80	0.26 / 0.57	0.29 / 0.62
`llama-3.3-70b`	0.96 / 1.00	0.77 / 1.00	0.11 / 0.31	0.23 / 0.88	0.29 / 0.86	0.48 / 0.74

I.2 The decode point does not read the linear coordinate

Three intervention families are applied at the final prompt token at every hooked site, about 1,500 greedy generations in total. *Clamps and steps.* The state is clamped along the fitted count direction to targets spanning the stable regime and slightly beyond, or translated by multiples of the mean per-item trajectory step. The decoded answer is unchanged at essentially every site in every model although the edits are large: at the last layer of one model a twenty-count clamp moves the state by 57% of the distance between genuine count states (Fig. 9). The one exception is `gpt-oss-120b` at its two last hooked layers, where extreme targets near and past the boundary occasionally knock the answer onto a single attractor value rather than tracking the target. Random directions at the matched norm and at four times the norm also leave the answer unchanged. *Chimera decomposition.* A donor prompt's decode-point state is split into its linear coordinate and everything else, and three transplants into a mid-regime base prompt are compared: the full donor state, the donor state with its linear coordinate clamped back to the base value, and the base state with only the donor's linear offset added. At the last layer the full state transfers the donor's answer in 20 of 20 trials per model and the residual transfers with the same profile, while the linear coordinate alone never transfers, at any depth of any model (Fig. 10). Transfer rises from chance at early layers to complete at the final layers, so the answer is recomputed downstream around any single-position edit until it consolidates near the surface; the consolidation layer of each model is the counting column of Table 17. Because the linear coordinate holds only 3 to 9% of the energy of the difference between two count states, the residual transplant is nearly a full-state transplant, and the decomposition establishes only that the linear coordinate is not what carries the answer. For tokenizers that split multi-digit numbers, a last-layer patch controls exactly the first emitted digit, so first-token transfer is the cross-model metric throughout.

I.3 The output channel is the decoder's answer-token subspace, and it is not counting-specific

The single-prompt cache above has one state per count, so any difference between two count states lies in the span of the stable states, whose rank is the number of stable counts minus one (27 to 70). A claim about what carries the answer needs a richer cache and candidate subspaces tested for sufficiency and necessity. We cached the decode-point state for eight counted items (`a b x z cat dog red sun`, single tokens in every tokenizer) and two delimiters at every count up to 1.25 times the scanned capacity, keeping the 300 to 1,000 prompts per model whose teacher-forced answer is exact, with folds that hold out whole counts. The candidate subspaces, each an orthonormal basis, are the linear direction L; the readable periodic directions F; the digit-class directions G; the decoder's own answer-token subspace U, the norm-scaled unembedding rows of the answer tokens with no fitting; the top-k trajectory components T_k of the count-conditioned means; random k-subspaces inside the trajectory span R_k; and random k-subspaces of the ambient space A_k. Five causal groups ran at the last two hooked sites of every model, about 27,000 greedy generations, with the same base, donors and seed as the chimera rows: full donor-state transplant at every layer; subspace transplant $h_{\text{base}} + P(h_{\text{donor}} - h_{\text{base}})$ for every family and its complement, scored by first-token transfer over the

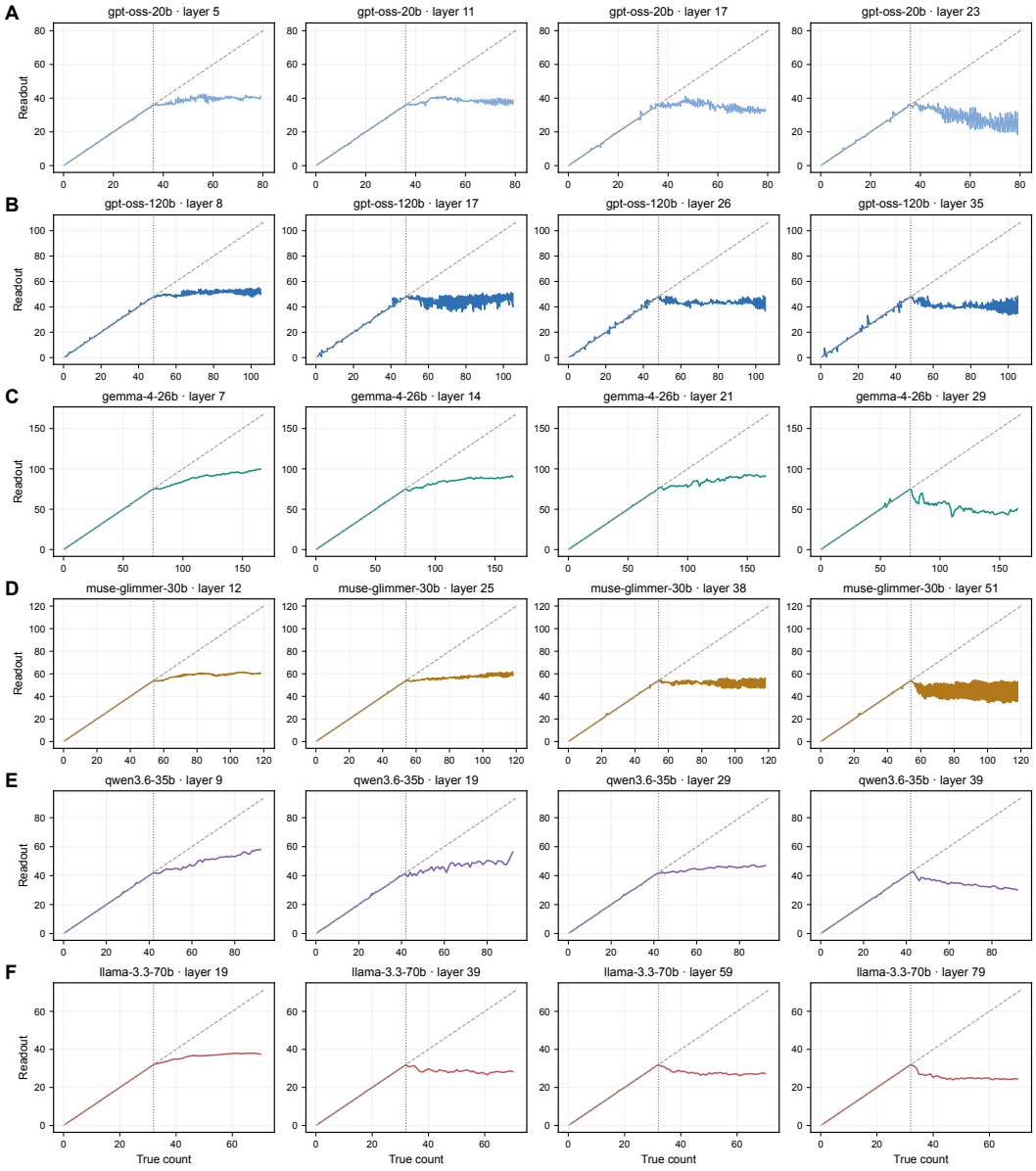

Figure 7: **Linear count readouts at every probed site.** Held-out readout against true count for the six models (rows, **A** to **F**) at the four hooked depths (columns), extrapolated beyond the stable regime. The dashed line is the identity; the dotted vertical rule is the model's immediate-answer CC. Early layers continue to track beyond the boundary; the last layer stops advancing and turns against the count.

donors whose leading digit differs from the base's; interpolation $h_{\text{base}} + \alpha(h_{\text{donor}} - h_{\text{base}})$ for α from 0.1 to 0.9; mean ablation of a subspace, scoring survival of the model's own answer; and rotation of the fitted period-10 pair (Section I.4). Figure 11 shows the causal panels for every model.

Sufficiency and necessity. At the last hooked site, projecting the donor-minus-base difference onto U transfers the donor's answer for every informative donor in five of six models, and the complement of U, which holds 74 to 99% of the difference's energy, transfers none; ablating U destroys the model's own answer in all six (0 to 6 of 20 survive), while L, F, and random ambient subspaces of up to 32 dimensions leave all 20 intact (Table 16). The exception, `gemma-4-26b`, decides its answer in a competition against its whole 262k-token vocabulary: the U projection moves the digit competition correctly for all 17 donors but the model then emits a non-digit token, so U is necessary (ablation 0 of 20) without passing the first-token test, while the fitted digit subspace G, which moves the competition on-manifold, transfers 0.88. The trajectory components are privileged but not tiny: 2

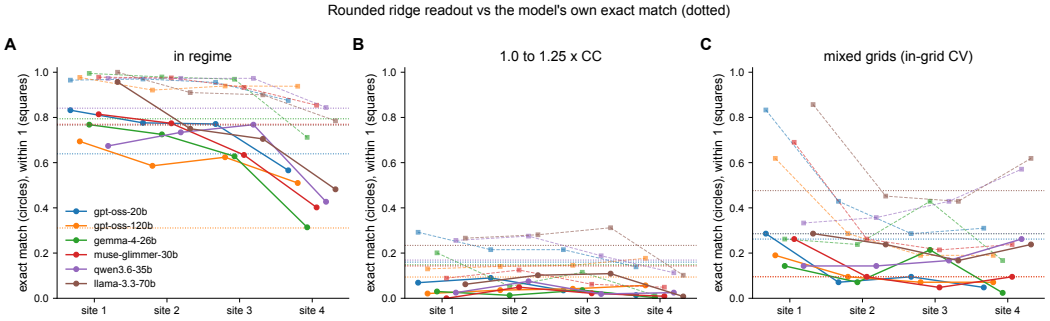

Figure 8: **Exact match of the rounded readout by site.** Exact-match rate of the rounded readout against depth for each model, with the model's own rate as a horizontal line, in the stable regime (**A**), from 1.0 to 1.25 times CC (**B**), and on the mixed grids (**C**).

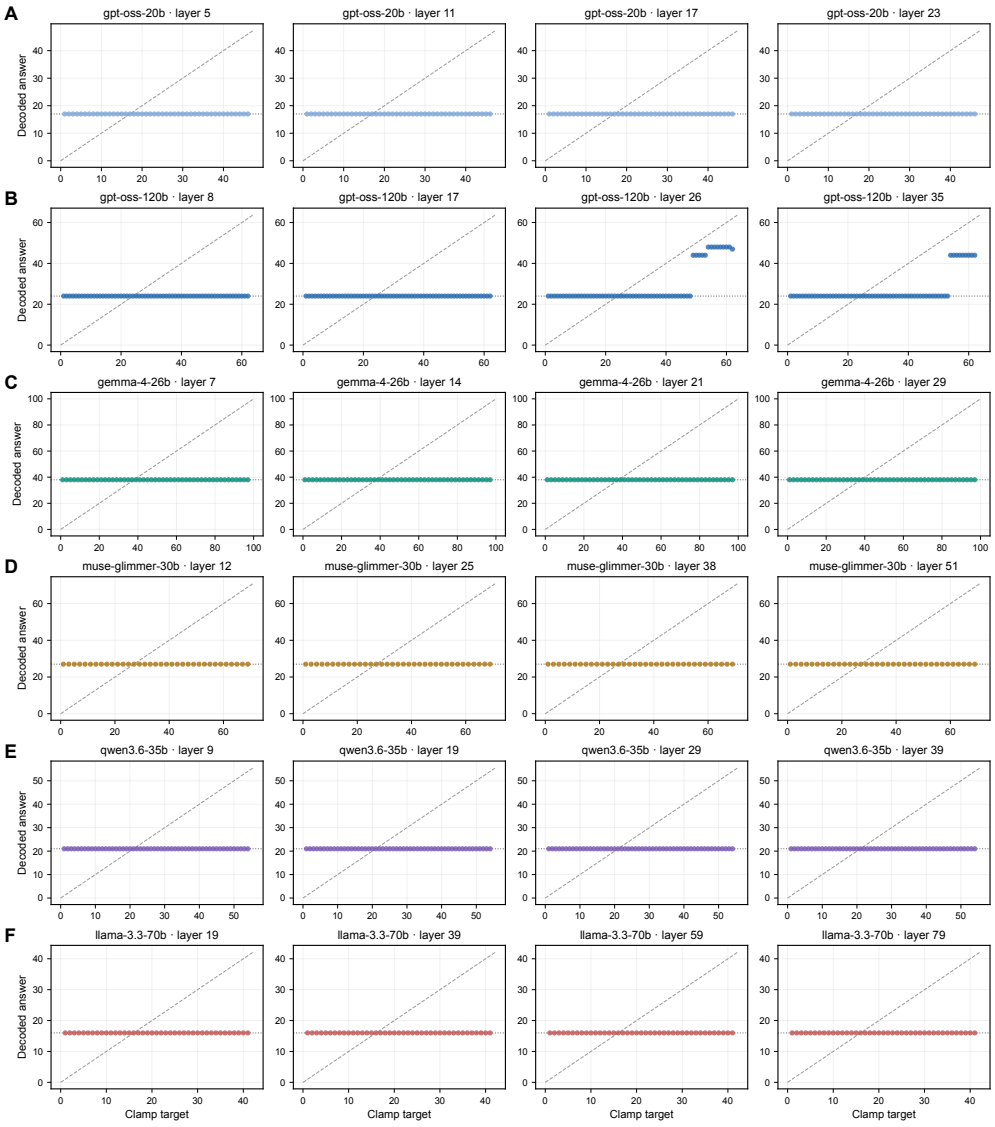

Figure 9: **Clamps along the count direction leave the answer unchanged.** Decoded answer against clamp target at every site (models **A** to **F**, columns by depth). The dotted horizontal rule is the unedited answer; the dashed diagonal is where answers would lie if the coordinate were read as a counter. Only the two last layers of `gpt-oss-120b` depart, and only onto a single attractor value at extreme targets.

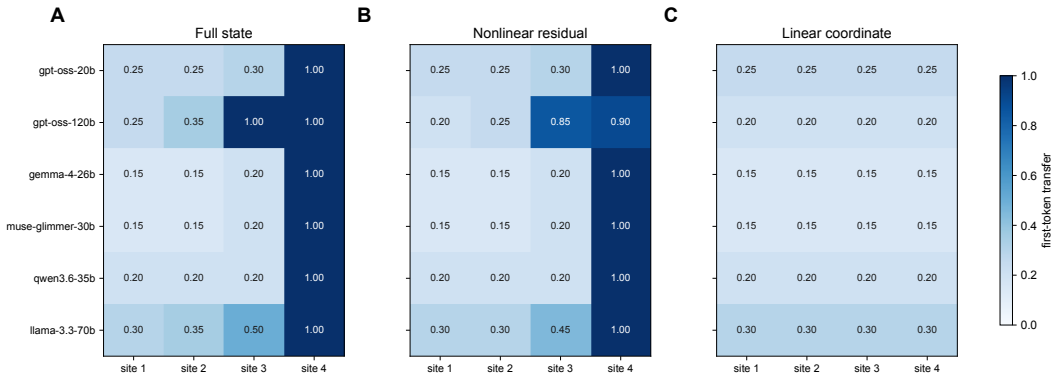

Figure 10: **Chimera transfer.** First-token transfer rate of a donor answer into a base prompt for the full donor state (**A**), the nonlinear residual alone (**B**), and the linear coordinate alone (**C**), by model and depth. The residual reproduces the full state's transfer; the linear coordinate never exceeds the incidental base rate.

to 16 of them reach half of the full-state transfer and 8 to 32 reach 90%, whereas a random subspace inside the same span needs its full rank, and ablating the top 8 to 16 trajectory components destroys most answers in five models while the same number of random in-span directions destroys none. Over the last quarter of the network the trajectory and digit subspaces rotate toward U (principal cosines rising from 0.05–0.49 to 0.14–0.72), and a logit lens restricted to the answer tokens becomes decisive at the last site.

Table 16: **Sufficiency and necessity at the last hooked site.** Transfer: first-token transfer rate of the donor's answer under a subspace transplant (informative donors, $n = 14$ to 17). Survival: the model's own first token surviving mean ablation of the subspace (of 20). d_{90}: smallest k at which T_k reaches 90% of the full-state transfer; rank: dimension of the trajectory span.

	Transfer				Survival			
Model	L	F	U	U complement	L	U	T_{16}	$d_{90}(T)$ / rank
`gpt-oss-20b`	0	0	1.00	0	20	0	2	16 / 32
`gpt-oss-120b`	0	0	1.00	0	20	4	6	16 / 37
`gemma-4-26b`	0	0	0*	0	20	0	6	8 / 74
`muse-glimmer-30b`	0	0	1.00	0	20	0	12	32 / 53
`qwen3.6-35b`	0	0	1.00	0	20	6	8	16 / 41
`llama-3.3-70b`	0	0	1.00	0	20	6	16	16 / 30

*The projection moves the digit competition correctly for every donor but the model emits a non-digit token from its full vocabulary; see text.

The channel is the same for copying and retrieval. U could carry the donor's answer for an algebraic reason: it is spanned by the answer tokens' own unembedding directions, so projecting a state difference onto it moves the answer logits by construction. To separate an output channel from a counting-specific code, two tasks with the same integer answers were run through the same battery: copying a stated number, and retrieving the value of a named key from a short list. At the last hooked site every number is the same for the two tasks as for counting: the U projection transfers the donor's answer for every informative donor in five models and for none in `gemma-4-26b`, the complement transfers nothing, ablating U destroys the model's own answer (0 to 6 of 20 survive for every task), ablating the linear, periodic, or random subspaces destroys nothing, and the interpolation switch falls between $\alpha = 0.5$ and 0.6 for every task (Fig. 12). The depth at which the channel fills is not counting-specific either (Table 17): a full donor-state transplant carries the copied or retrieved answer only from a layer within a few layers of the counting consolidation layer, later in three models and earlier in two, never early. `gpt-oss-120b` consolidates counting from layer 20 through the complement of U (transfer 0.56 to 0.75 with U at 0.00 to 0.06 over layers 20 to 26) and switches into U at layer 27 of 36: a pre-output code converted into the output code in the last quarter. What is counting-specific is therefore not the channel or when it fills, but what fills it.

Table 17: **The output channel fills late for every number-producing task.** First layer at which a full donor-state transplant carries at least 80% of the donors' answers, by task; the counting column is the consolidation layer of Section I.2.

Model	Layers	Counting	Copy	Retrieval
gpt-oss-20b	24	19	21	21
gpt-oss-120b	36	20	27	23
gemma-4-26b	30	28	25	26
muse-glimmer-30b	52	49	43	39
qwen3.6-35b	40	35	36	38
llama-3.3-70b	80	68	68	65

The switch does not pass through the integers between the answers. Interpolating between two count states and reading only the decoded answer cannot distinguish a categorical code from a continuous one, since the argmax turns any linear crossing into a switch. Recording the answer-token logits along the interpolation at the last site shows what happens to the integers between the two answers (Table 18). The base and donor logits cross between $\alpha = 0.5$ and 0.55 in every model. In the 79 pairs whose leading digits admit an integer token between the answers, an integer more than two away from both leads in one (a gpt-oss-120b pair), the immediate neighbors of the base answer lead briefly in six (five in gpt-oss-20b, one in gpt-oss-120b), where they hold a third of the answer-token mass near the crossing, and in four models no intermediate of any kind leads.

Table 18: **Logits along the interpolation.** Pairs whose leading digits admit an intermediate token; pairs in which an intermediate integer ever leads, split by whether it lies within two of an endpoint; softmax mass over the answer tokens on intermediate integers at $\alpha = 0.5$ (total, and on integers farther than two from both endpoints); and the α at which the decoded answer switches.

Model	Pairs	Neighbour leads	Farther leads	Mass at 0.5 (total / far)	Switch α
gpt-oss-20b	15	5	0	0.33 / 0.04	0.55
gpt-oss-120b	16	1	1	0.33 / 0.10	0.55
gemma-4-26b	12	0	0	0.00 / 0.00	0.50
muse-glimmer-30b	17	0	0	0.08 / 0.02	0.55
qwen3.6-35b	5	0	0	0.01 / 0.00	0.55
llama-3.3-70b	14	0	0	0.00 / 0.00	0.50

I.4 Periodic and digit features are correlates

The multi-prompt cache also supports readouts of periodic phases $\cos$ and $\sin(2\pi n/P)$ for $P \in \{2, 5, 10, 20\}$, flagged readable when both phases reach held-out $R^2 \geq 0.5$ and beyond a smoothness null when they beat the same target computed from the linear readout, and of the last and tens digits by shrinkage discriminant analysis with a count-level permutation null. At the earliest sites a held-out count's last digit is decoded as a neighboring digit 0.27 to 0.79 of the time, as a smooth analog code would produce; in the second half of the network the last digit becomes decodable (0.33 to 0.42 in the two gpt-oss models, 0.61 to 0.85 in gemma-4-26b, muse-glimmer-30b and qwen3.6-35b, against a permutation ceiling of 0.15 to 0.22) and that confusion drops. Period-10 phases are readable at the last site of every model and beyond the smoothness null in two. Where the digit code is strongest, at the third hooked site of gemma-4-26b and muse-glimmer-30b, nothing is causal: no family at any dimension transfers, ablates, or rotates the answer, because those sites precede the consolidation layer. Rotating the period-10 pair at the last site by the phase of $+d$ counts, $d = 1 \ldots 9$, moves the answer in 0 to 6 of 180 prompts per model, never in agreement with the target phase and no more often than a random plane of the same span (Fig. 11, fourth column). The periodic and digit features are correlates, like the linear coordinate, with the one exception of gemma-4-26b's digit subspace noted above.

I.5 The accumulation is read at mid-depth across item positions

The decode-point edits show that the fitted coordinate is not read where the answer is emitted; they say nothing about whether the accumulation is read upstream. We therefore intervened on the running

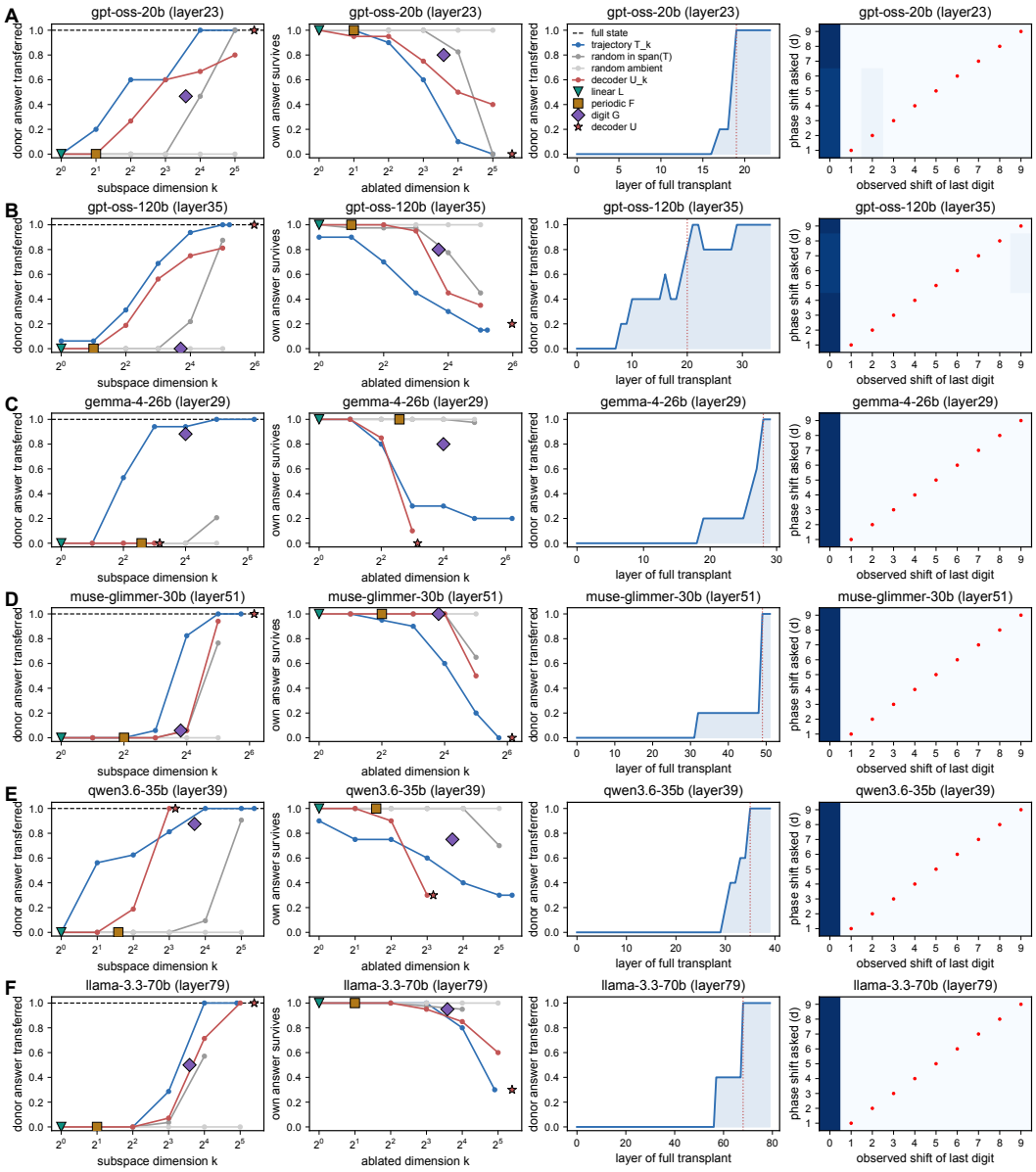

Figure 11: **Subspace and periodic-feature controls.** One row per model (**A** to **F**) at its last hooked site. First column: first-token transfer of the donor answer against subspace dimension for trajectory components T_k, random subspaces inside the trajectory span, random ambient subspaces, and the decoder's answer-token subspace U_k; the single markers are the linear direction, the periodic pair, the digit subspace, and the full U. Second column: survival of the model's own answer under mean ablation of the same subspaces. Third column: full-state transfer at every layer, with the consolidation layer marked. Fourth column: phase shift asked against observed shift of the last digit under rotation of the period-10 pair; a helix read by the decoder would fill the diagonal.

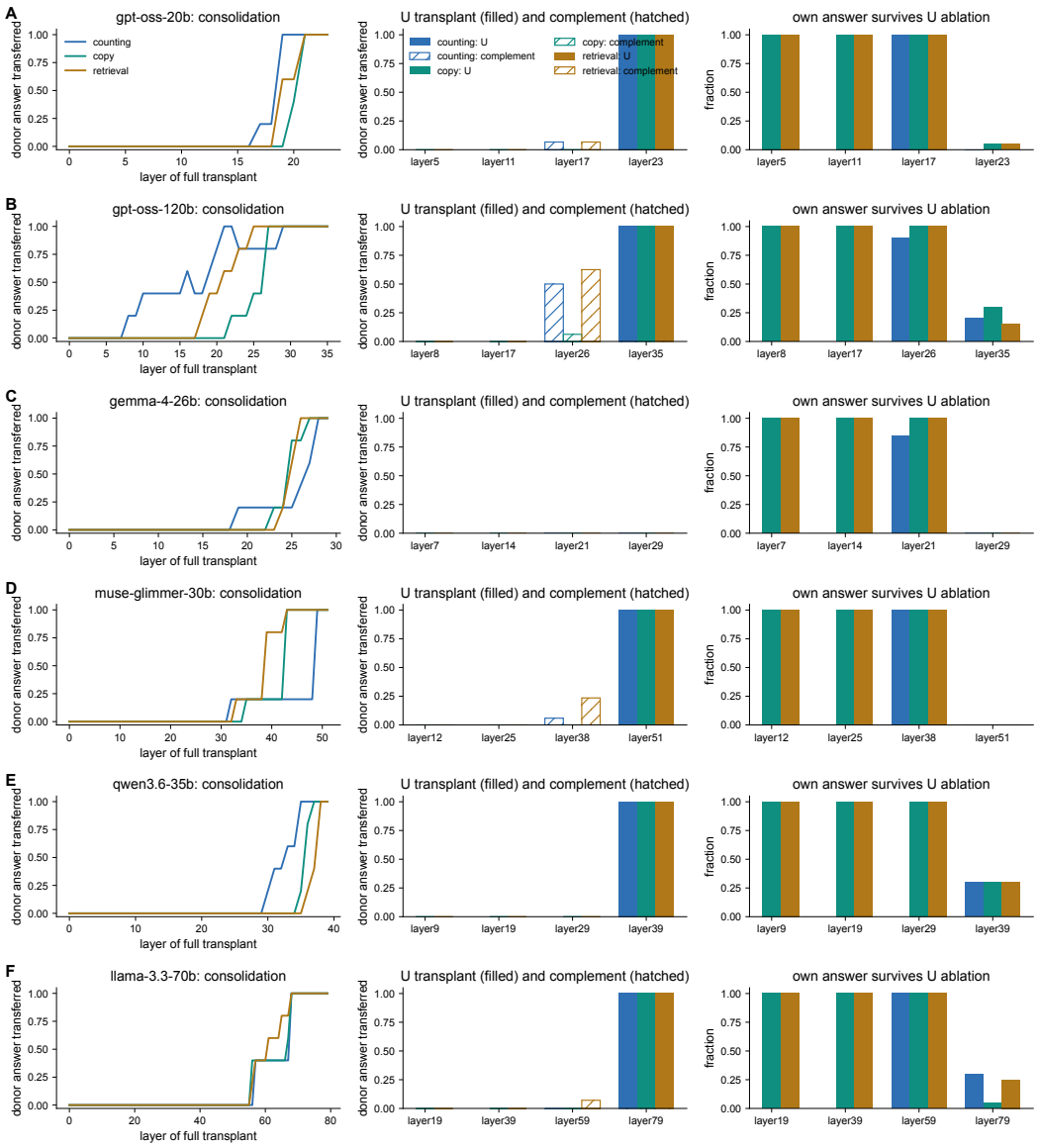

Figure 12: **The output channel by task.** For each model (**A** to **F**): the fraction of donors whose answer a full-state transplant carries, by layer, for counting, copy, and retrieval (left); transfer through U and through its complement at the last site, by task (middle); and survival of the model's own answer under ablation of U and of the control subspaces, by task (right).

count at the item positions themselves. Because attention is causal and every prompt shares its prefix, one forward pass at 1.25 times CC plus eight items yields the state of every item token for every shorter prompt at all four sites; a ridge readout of the running count from those states holds at every site (leave-one-out R^2 0.93 to 0.99), and the subspace S_k spanned by the top trajectory components (k the dimension that carried 90% of decode-point transfer) holds 64 to 99.9% of the item-state variance. The edit replaces the S_k component of the state at running count i by the trajectory's own component at count $i + \delta$, at every item position from a chosen onset (a quarter, half, or three quarters of the way through), for prompts at 0.35 to 0.9 of CC and δ from -8 to $+8$. The controls apply the same edit through a fixed random direction of matched norm, and at the decode point only. The answer shift is regressed on δ (Table 19, Fig. 13).

In three models the answer follows the shifted accumulation at one or two mid-depth sites, after the first hooked site and before consolidation: `gpt-oss-20b` at layers 11 and 17 of 24, `gemma-4-26b` at layer 14 of 30, and `gpt-oss-120b` at layer 17 of 36, with slopes of 0.20 to 0.52 that reach

0.5 to 0.9 on individual prompts, while at those sites the random direction is flat or of the opposite sign and the decode-point edit is flat, except at `gpt-oss-20b`'s third hooked site (layer 17), where the random shift reaches 0.20 and the decode-point edit 0.12 against 0.34. At the first hooked site the slope is zero in every model, so the count there is recomputed downstream from the tokens, and after consolidation it is zero, so the decoder no longer reads the item positions once the answer has formed. In `muse-glimmer-30b`, `qwen3.6-35b`, and `llama-3.3-70b` no hooked site exceeds 0.08; their causal window may fall between hooked sites, or their decoders may read the count through a route the edit does not touch. Beyond CC the shift moves nothing at any site of any model, while at `gemma-4-26b`'s causal site a random shift of the same norm now moves the answer strongly: past the boundary the state is fragile but no longer steerable through the count subspace. Compressing the accumulation so that the last item reads as the in-regime maximum rescues no model except `gemma-4-26b`, which then answers the compressed count. In half of the models the answer therefore follows the accumulation causally, with a gain below one; in the other half the evidence remains correlational.

Table 19: **Answer shift per item of accumulation shift.** Least-squares slope of the answer change on δ, pooled over prompt lengths and onset positions, with a bootstrap 95% interval, at the responsive site of the three responsive models and at the most responsive site of the other three; the two controls are the same edit through a fixed random direction of matched norm and at the decode point only; the last column is the slope for prompts from 1.0 to 1.25 times CC.

Model	Site	Slope [95%]	Random	Decode point	Beyond CC
`gpt-oss-20b`	layer 11 of 24	0.43 [0.32, 0.53]	−0.07	−0.01	0.08
	layer 17 of 24	0.34 [0.24, 0.45]	0.20	0.12	−0.02
`gpt-oss-120b`	layer 17 of 36	0.20 [0.13, 0.28]	0.00	0.00	0.09
`gemma-4-26b`	layer 14 of 30	0.52 [0.33, 0.74]	−0.24	0.00	0.06
`muse-glimmer-30b`	layer 12 of 52	0.07 [0.02, 0.14]	−0.01	0.00	0.10
`qwen3.6-35b`	layer 19 of 40	0.02 [0.00, 0.06]	0.02	0.00	−0.01
`llama-3.3-70b`	layer 19 of 80	0.05 [0.01, 0.10]	0.00	0.00	0.00

I.6 The boundary state carries the count forward, and the same replacement carries the state of an object history

A third intervention asks where the running count is handed from one item to the next. A recipient sequence of C_B items and a donor sequence of C_D items are run separately; the residual states of the donor's last items are written into the recipient's last item positions at every layer from an entry layer upward, with the recipient's own states restored at every earlier position so that the edit cannot leak backward through the mixture-of-experts batching; the recipient's text is left in the context; then r further items are appended that neither run has seen, and the model is asked for the count. If the continuation starts from the state the window carries, the answer is $C_D + r$; if it re-reads the text, $C_B + r$. Controls are a sham edit (the recipient's own states written back), a same-count donor ($C_D = C_B$ under different padding), a random-source window (the donor's states from an earlier span of its sequence), a position factorial (donor and recipient paddings crossed), and the whole donor cache as the ceiling. The operator was chosen on calibration pairs by a rule fixed in advance (the smallest window and then the fewest layers among candidates at or above 80%) and frozen before held-out pairs were run.

Counting. In `gpt-oss-20b` a window of the last eight items entered at any layer from 2 to 8 gives the predicted count on every calibration instance, and entered at layer 9 or above on none; windows of one, two, or four items carry nothing, and replacing keys and values instead of residual states carries nothing. The frozen operator (eight items, layers 8 to 23) gives $C_D + r$ exactly on 10 of 13 strictly eligible held-out instances (bootstrap 95% interval 0.50 to 1.00) and within one item on 12 of 13; sham edits change none of 13, same-count donors leave 8 of 8 unchanged, and random-source windows give the count their own items carry plus r rather than $C_D + r$, so the continuation starts from whatever running count the transplanted items encode. In `gemma-4-26b` the window must be twelve items; entered at layer 8 it is exact on 9 of 20 held-out instances, on every instance where the transplanted count is at or below the count the recipient's own positions carry (9 of 9), and biased by one to several items above that; shams and same-count donors are unchanged, and random-source windows again follow the count they carry (37 of 45). In `gpt-oss-120b` the frozen operator enters

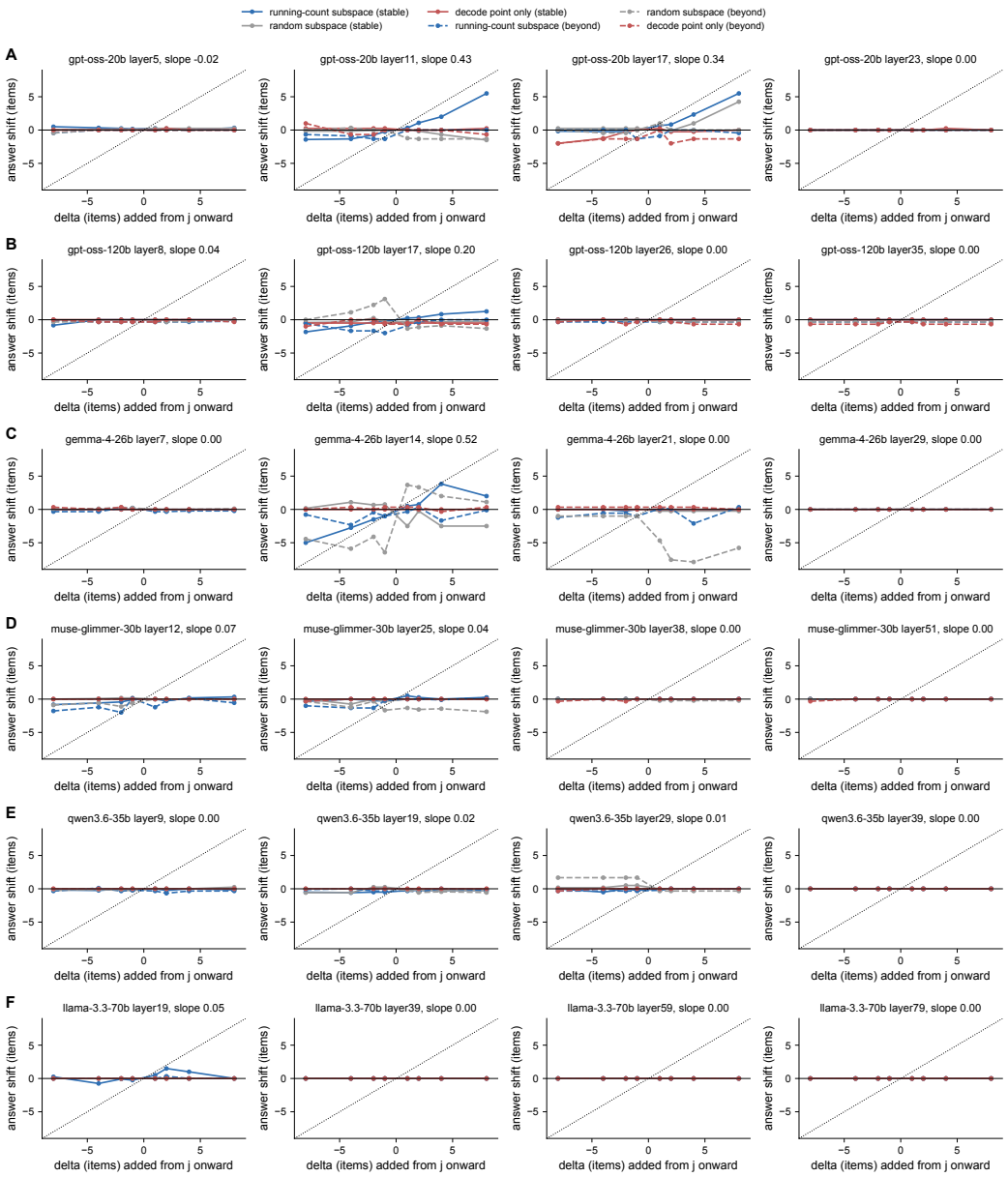

Figure 13: **Shifting the accumulation across item positions.** Answer shift against δ for each model (**A** to **F**, rows) at the four hooked sites (columns), pooled over prompt lengths and onset positions; the count-subspace edit is drawn against the matched random direction and the decode-point-only edit, with the identity line for reference.

at layer 5 and gives 7 of 18. The highest entry layer that still carries the count is nearly the same in all three models, layer 8 or 9 of 24, 30, and 36 (entry at layer 9 or 10 and above carries nothing), upstream of the mid-depth sites where the item-position shifts act. The route is position-sensitive: donor keys and values, which carry the donor's positional phase, transfer nothing, the same donor states recomputed at the recipient's positions do, and padding that moves the recipient's positions toward the donor's restores exactness (7 of 11 in the factorial). The running count that the decoder reads is therefore recomputed from the states of the last eight to twelve items through a route that enters at layer 8, not inherited through a single boundary state.

An object-tracking task with the same signature. To ask whether these states are counting-specific, the same operator was applied, without any retuning, to a task with no count in it, a state-maintenance task in which history length (the number of swaps) and state size (the number of

objects) vary independently and access to the record is tested separately from the state. Three to five named objects start at named locations, a history of swaps moves them, and after the history the model is asked, with immediate answering and randomized answer labels, for the location of an object the last swap moved, the location of an object no swap touched, what the last swap said, and, with the correct current state written into the prompt, a location question that needs no tracking. Table 20 gives `gemma-4-26b`'s competence. One swap is tracked perfectly at every state size and two swaps at 0.88 to 0.94, while three and four swaps fall to 0.40 to 0.75, with the untouched object misplaced in most histories from three swaps on; access to the last swap's text is 48 of 48 in every cell and the supplied-state answer is 0.83 to 1.00. The failures are in the state update, with the events still readable, which is the signature counting shows.

Table 20: **Object tracking in `gemma-4-26b`, immediate answering.** Exact-location rate for an object the last swap moved and for an object no swap touched, the rate of reporting the last swap's text correctly, and the rate on the supplied-state question, by number of objects and swaps; 48 histories per cell.

Objects	Swaps	Moved object	Untouched object	Last swap reported	Supplied state
3	1	1.00	1.00	1.00	1.00
3	2	0.90	0.35	1.00	1.00
3	3	0.60	0.29	1.00	1.00
3	4	0.40	0.29	1.00	0.83
4	1	1.00	1.00	1.00	1.00
4	2	0.88	0.98	1.00	1.00
4	3	0.75	0.31	1.00	1.00
4	4	0.52	0.17	1.00	0.88
5	1	1.00	1.00	1.00	1.00
5	2	0.94	1.00	1.00	1.00
5	3	0.67	0.62	1.00	0.98
5	4	0.71	0.25	1.00	0.98

Transplanting the boundary state of an object history. The recipient is a two-swap history. The donor is the same history with the first swap replaced by another pair of locations or by a swap of a location with itself, so that the two histories reach different states while the second swap's text is identical; that identical line is the window (twelve to thirteen tokens, layers 8 to 29, the earlier positions restored as above, nothing at the decode point, the recipient's full history in the context). From one intervened prefix the model is asked for the location of every object whose true and donor states differ (prediction: the donor's state), of an object whose location agrees (prediction: unchanged), which locations the first swap named, with the donor's pair among the options (prediction: the real text), and the supplied-state question (prediction: unchanged). Controls are the sham edit and the whole donor cache. A first design with the donor differing at the last swap and the window on that swap's own line moved every changed object (144 of 144) but overwrote the event: asked what the last swap said, the model reported the donor's pair 48 of 48 times, so that design replaces text rather than state and was discarded. Table 21 and Fig. 4G give the result of the second design.

Table 21: **The counting operator applied to an object history in `gemma-4-26b`.** Replies to location questions about objects whose true and donor states differ that give the donor's state, under the edit, the sham edit, and the whole donor cache; then, under the edit, the rate of reporting the first swap's real text, the rate at which an object whose location agrees between the true and donor states stays put ("untouched"; this differs from Table 20's never-moved object), and the rate on the supplied-state question. Two-swap histories; 48 per cell; "replace" donors differ in the first swap's pair and "omit" donors skip it.

Objects	Donor	Edit	Sham	Whole cache	Swap 1 real	Untouched	Supplied
3	replace	81/144	23/144	102/144	48/48	–	40/48
3	omit	57/96	32/96	75/96	48/48	45/48	36/48
4	replace	44/150	19/150	98/150	48/48	42/42	43/48
4	omit	45/96	37/96	88/96	48/48	48/48	42/48
5	replace	43/156	20/156	118/156	48/48	48/48	44/48
5	omit	42/96	31/96	87/96	48/48	48/48	41/48

The edit moves the changed objects toward the donor's state in 28 to 59% of questions against 13 to 39% for the sham edit and 65 to 92% for the whole donor cache. What moves is the state: the first

swap is still reported correctly in 48 of 48 histories in every cell, an untouched object is displaced in at most 3 of 48, and the supplied-state answer stays correct in 36 to 44 of 48, against 48 of 48 with no edit. The same replacement, at the same layers, therefore carries a state on both tasks: the running count in counting, and the tracked locations in an object history, with the history's text left readable.

Under native reasoning the model re-reads the history. On 40 two-swap histories (24 with three objects, 16 with four) in which the immediate-answer edit had moved the changed object to the donor's location, the same edit was applied with the model's native reasoning enabled (1,536 reasoning tokens; one reply truncated and excluded). The changed object is placed correctly in 23 of 23 and 15 of 16 complete edited replies, in every untouched and sham reply, and the first swap is reported correctly in every reply of every condition. The traces restate the initial assignment and apply both swaps in order before answering; the edited traces are about 40% longer than the untouched ones (203 against 135 and 210 against 165 reasoning tokens), which is the only footprint the transplanted state leaves. When the history is a sufficient record of the state, reasoning rebuilds the state from it and the transplant is undone. On counting the history is not such a record, because a written value cannot be re-attached to the indistinguishable event where it was written, and generated reasoning does not restore the count (Section 3, Appendix F). The same intervention carries the state in both tasks, which is consistent with a shared substrate without identifying one; the tasks differ in whether the text can rebuild it.

I.7 Count versus length: readable, not routed

On mixed sequences, in which the target item appears k times inside a sequence of total length L and the two are varied independently within the stable regime, the count and the length are recovered by separate readouts (count R^2 0.70 to 0.99, length 0.37 to 1.00) along nearly orthogonal directions ($|\cos|$ 0.01 to 0.42): the accumulator encodes the count separately from the length. Immediate-answer accuracy on the same grids collapses to between 10% and 48% (Fig. 14), with every k far inside the model's homogeneous-task stable regime. A readout fitted within the grid recovers the count to within one item in 0.43 to 0.86 of trials, above the model's own within-one rate in four of six models by 0.12 to 0.55, and exactly in 0.19 to 0.29, above the model's exact rate in three models (`gpt-oss-20b`, `gpt-oss-120b`, and `muse-glimmer-30b`, by 0.03, 0.09, and 0.16) and below it in the other three (Table 15); the homogeneous-task readout does not transfer to the grids. The claim is therefore about a magnitude estimate within one item, decodable at an early site where the model's own answer is not, with exact decoding ahead of the model in half of the models: count information the model cannot route to its own output.

I.8 Reasoning modes

Immediate-answer and native reasoning scans for all models are shown in Fig. 15. Two models, siblings from one family, gain 64% and 113% from reasoning; their analysis channels re-enumerate the sequence with periodic subtotals, and their silent readouts are the noisiest of the six, so externalizing wins. One model loses 40%: its thinking traces enumerate, distrust the total, re-count in groups, and loop until the token budget truncates them, replacing a better silent mechanism with a worse text-mediated one. Two are unchanged, and the sixth, `llama-3.3-70b`, has no native reasoning mode. Scaling within the one family raises the text-mediated capacity, not the internal one: the larger sibling's best held-out readout error is five times worse than the deepest model's (`llama-3.3-70b`), while its reasoning capacity doubles the smaller sibling's.

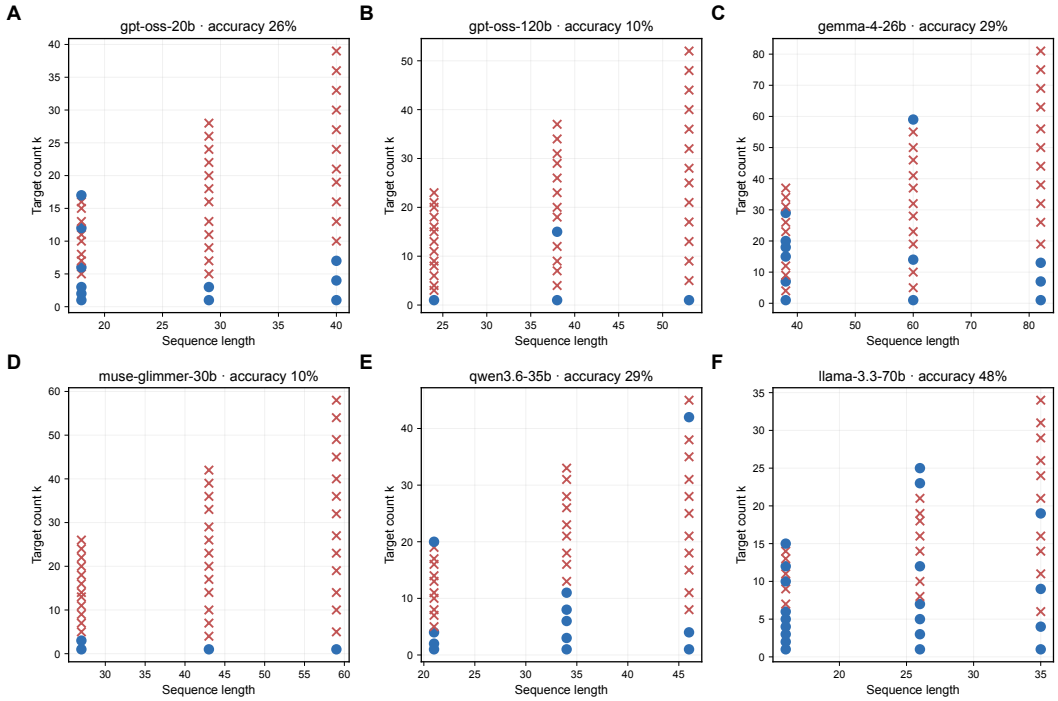

Figure 14: **Mixed-sequence deconfound.** Correctness of the immediate answer over the $k \times L$ grid for each model (**A** to **F**); circles are correct trials and crosses errors. Every probed k lies far inside the model's homogeneous-task stable regime.

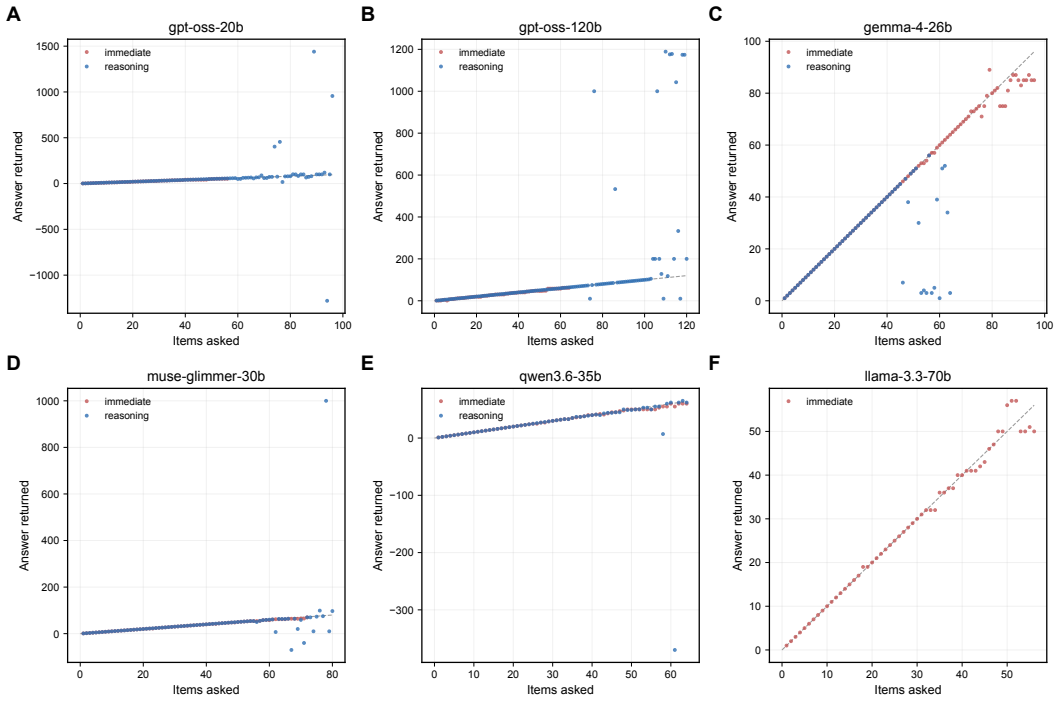

Figure 15: **Immediate-answer and reasoning scans.** Answer returned against items asked in both modes for each model (**A** to **F**). Reasoning raises capacity in the two `gpt-oss` models, lowers it in `gemma-4-26b`, and leaves `muse-glimmer-30b` and `qwen3.6-35b` unchanged.

J CROSS-BENCHMARK CORRELATIONS

To compare CC with standard task performance, we cross-referenced model results against public benchmark metrics, including GPQA Diamond, ARC-AGI-2, SWE-bench Verified, and OTIS Mock AIME (Rein et al., 2024; Chollet et al., 2025; Jimenez et al., 2024; SWE-bench Team, 2024; Epoch AI, 2025a). Correlation coefficients were calculated across intersecting model subsets for 66 public benchmarks (44 sharing at least ten models with the evaluation set).

The aggregate comparisons (the METR time horizon and total parameter count) appear alongside the headline benchmarks in the main text (Fig. 5), and serving price is given below; the Epoch Capabilities Index reaches $\rho = 0.66$ over 103 shared models and appears with every other benchmark in the panel pages below. Figure 17 summarizes the individual benchmarks as a distribution of rank correlations. Serving price tracks CC as well (Fig. 16): completion price per million tokens from a dated snapshot of live prices (2026-08-25) correlates with CC at $\rho = 0.60$, approximately price $\propto \mathrm{CC}^{1.1}$, within vendors as well as across them; price is confounded with model size, so this is the market's estimate of usefulness rather than a separate measurement.

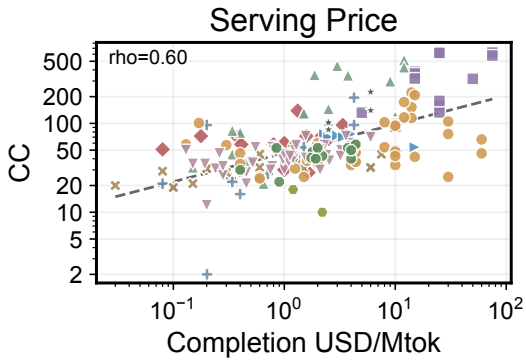

Figure 16: **Serving price against counting capacity.** Completion price per million tokens (2026-08-25 snapshot) against CC, one point per model, colored by provider.

Parameter count is the one scale variable available: 103 models have a public total count ($\rho = 0.69$ with CC, Fig. 5D), and for the 53 that also report an active count, total parameters predict CC far better than active parameters, with Spearman ρ of 0.65 against 0.16, a partial correlation of 0.71 for total given active and -0.39 for active given total, and a leave-one-out R^2 of 0.35 from total alone against 0.39 with active added, the added coefficient being negative. Within the 29 mixture-of-experts models the pattern is the same (0.66 against 0.25), and a smaller active fraction goes with a higher CC ($\rho = -0.72$ against active/total), so the capacity tracks the parameters a model has rather than those it spends per token; the sample is the open-weight part of the catalogue, and the counts come from model cards and slugs, audited against vendor documentation for the mixture models. The three largest rank correlations in the panels belong to Balrog ($\rho = 0.87$ over 16 shared models), WeirdML (0.80 over 49), and the METR time horizon (0.78 over 27). Because frontier models tend to be better at everything, we checked which of these relations survives conditioning on general capability, with the Epoch Capabilities Index as the control. For METR, on the 24 models with both scores, $\rho(\mathrm{CC}, \mathrm{METR}) = 0.83$; the partial correlation controlling for the index is 0.74, and the reverse partial, index with METR controlling for CC, is 0.86, so the two carry independent information about task duration. For Balrog and WeirdML the same partial correlations are 0.07 (13 models) and 0.30 (43 models): their raw correlations with CC are carried by general capability, with which the two are correlated at ρ of 0.85 and 0.95. An incremental test agrees. When each score is predicted from the index alone and from the index plus $\log \mathrm{CC}$, with every vendor held out in turn, adding CC raises out-of-sample R^2 for the log time horizon from 0.81 to 0.90 (0.86 to 0.92 holding out one model at a time) and moves Balrog and WeirdML by at most 0.02 (0.72 to 0.70, and 0.89 to 0.91). Leaving out any one vendor gives ρ between 0.72 and 0.88 for METR, and the relation holds within the largest vendor group ($\rho = 0.81$ over 13 OpenAI models). The vendor folds behind that gain hold 11, 6, 4, 2, and 1 of the 24 models; a bootstrap over models puts the gain at 0.06 with a 95% interval from -0.24 to 0.30 ($P(\text{gain} > 0) = 0.79$), so the incremental prediction is suggestive rather than established. The same bootstrap puts the partial correlation at 0.74 with a

95% interval from 0.39 to 0.87, and dropping the six Anthropic models leaves 0.65 on 18. Within the Claude Opus line the horizon rose from 100 to 719 minutes across four releases while CC stayed near 600, and the later Opus releases whose CC fell to 133–179 are not on the leaderboard, so the line itself does not test the relation. The choice of configuration does not drive it: matching the counting configuration to the leaderboard's setting (the reasoning-on capacity for 16K-budget and high-effort rows, the immediate-answer capacity otherwise) gives $\rho = 0.79$, a partial correlation of 0.75, and a gain of 0.09, and taking each model's best-horizon row regardless of setting gives 0.77, 0.71, and 0.07, with the bootstrap interval including zero in every case. Table 22 lists the matched models and configurations: entries are matched by the leaderboard's model version string, dated checkpoints to the dated catalogue entry, at every leaderboard setting except max (the medium or default row where several are listed), and where a model has both reasoning configurations the larger capacity is used. Five of the 27 matched capacities, four of them among the 24 with an index score, are budget lower bounds entered at their bound; restricting to resolved capacities leaves 22 models with $\rho = 0.79$ and 20 with an index, on which the partial correlation is 0.71.

Table 22: **Models matched between the CC catalogue and the METR time-horizon leaderboard.** Entries are matched by the leaderboard's model version string, with dated checkpoints matched to the dated catalogue entry; the configuration column is the leaderboard's setting for the matched row, and where the leaderboard lists several settings for one model the medium or default row is used. The CC is the larger of the model's two reasoning configurations. The Epoch Capabilities Index (ECI) is given to one decimal; the correlations in the text use unrounded values. Vendor folds for the held-out prediction: OpenAI 11, Anthropic 6, DeepSeek 4, Google 2, Qwen 1.

Model	Configuration	CC	Horizon (min)	ECI
`claude-opus-4`	unknown	627	100.4	143.0
`claude-opus-4.6`	unknown	622	718.8	155.0
`claude-opus-4.5`	default	620	293.0	150.0
`claude-opus-4.1`	default	580	100.5	144.4
`gemini-3.1-pro-preview`	default	518	384.1	154.6
`claude-sonnet-4`	16K budget	382	74.9	142.3
`claude-sonnet-4.5`	16K budget	364	121.9	146.9
`gpt-5.3-codex`	default	223	349.5	156.2
`gpt-5.4`	unknown	208	341.7	156.5
`gpt-5.2`	high	152	352.2	153.3
`gemini-2.5-pro-preview`	default	120	38.7	145.8
`o3`	medium	103	91.3	147.1
`gpt-5`	medium	101	137.3	150.0
`gpt-oss-120b`	default	101	42.0	140.7
`deepseek-r1-0528`	default	71	31.2	142.0
`kimi-k2-thinking`	default	70	54.2	–
`deepseek-r1`	default	67	26.9	139.7
`o1`	medium	66	38.8	142.7
`o4-mini`	medium	50	76.5	146.4
`gpt-4o-2024-08-06`	default	48	7.0	129.1
`gpt-5.1-codex-max`	default	43	223.7	–
`gpt-3.5-turbo-instruct`	default	36	0.6	–
`gpt-4o-2024-11-20`	default	34	9.2	129.4
`deepseek-chat`	default	30	18.5	146.5
`deepseek-chat-v3-0324`	default	30	23.1	137.1
`qwen-2.5-72b-instruct`	default	27	5.2	129.1
`gpt-4-turbo`	default	25	6.6	127.6

Parameter counts are unavailable for most of the METR set, which is proprietary, so scale cannot be conditioned on. Panels are drawn only for benchmarks sharing at least five models with the evaluation set, and the number of shared models is printed in each panel. The complete set of pairwise correlation panels, extending those shown in the main text (Section 5), is collected in Figures 18 to 20 at the end of this appendix.

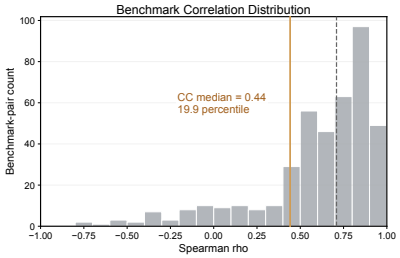

Figure 17: **Distribution of benchmark correlations with counting capacity.** Rank-correlation values across evaluated public benchmark metrics summarize how weakly standard task leaderboards align with task-modeling capacity.

K STATE-TRACKING CAPACITY ON A FIXED MODEL SUBSET

This appendix reports a prespecified test of whether a state-maintenance task with no count in it fails at a length that follows counting capacity across models, and whether native reasoning loosens that coupling. The plan, the predictions, and the analysis were fixed before any scan ran.

Task. Four named objects (red, blue, green, yellow) start at four named locations (A to D); a history of T swaps follows, one per line ("Swap the objects in B and C."); the question is the location of an object the last swap moved, answered as one integer label among four, the labels drawn at random per probe so that no location has a fixed name. The system prompt is the catalogue's ("Return only one integer"), decoding is the catalogue's (temperature zero unless the model fixes it, 8,192 output tokens), and reasoning is switched on or off through the roster's setting exactly as for CC. Every model sees the same histories: the generator is seeded by length and probe index only.

Schedule and estimator. The scan is coarse to fine. It begins with two probes at $T = 100$; the step doubles while both are exact and halves when either is not, until the bracket between the last exact and the first inexact length is tight; then a dense scan of about 24 points across the widened bracket, one probe per length, ascending, with the catalogue's stop rule (stop when the two-regime fit declares a transition whose broken regime is established, the Wilson upper bound of the accuracy above it below one half). The dense scan extends upward while the record is still exact at its top or the tail is unresolved, and downward when no stable regime has been seen; a final floor sweep adds two probes at every length from 1 to 8 to every run, so that the stable regime is anchored at short histories. Every probe enters the record, and the tracking capacity TC is the two-regime maximum-likelihood changepoint of Appendix B with its calibrated interval. Chance is one in four rather than zero, so the broken regime sits near 0.25 instead of near zero.

Scoring. A reply is exact when its last integer is the expected label. Some models return the location letter instead of the label despite the instruction; a reply that is a single location letter, or whose last line is one, is scored against the expected location and marked as read by letter. A narrated reply is scored by its last integer, as in the counting protocol, so narration that ends on the right label counts and narration that does not counts as inexact. Replies that are API errors count as inexact and are attributed to the budget, as for CC; runs in which every probe returned an error (the o-series with reasoning off, which the catalogue never measured either, and Claude Opus 4, which the router failed on during the scan) are excluded.

Side questions. At 0.5, 1.0, and 1.5 times the fitted boundary, five probes each are added: the event question (which two locations the last swap named, two options) with the full history present, and the supplied-state question (the same four-label location question with the final state stated and no history), which tests the decision with no tracking involved.

Subset. The rule was fixed in advance: every model with both a METR horizon and an Epoch index score in Table 22 (24 models), the six open-weight probe-suite models, and the five resolved catalogue models with the lowest CC that are still served; every hybrid among them runs in both reasoning modes. The subset is skewed toward recent models, as the METR leaderboard is. Table 23 lists every run.

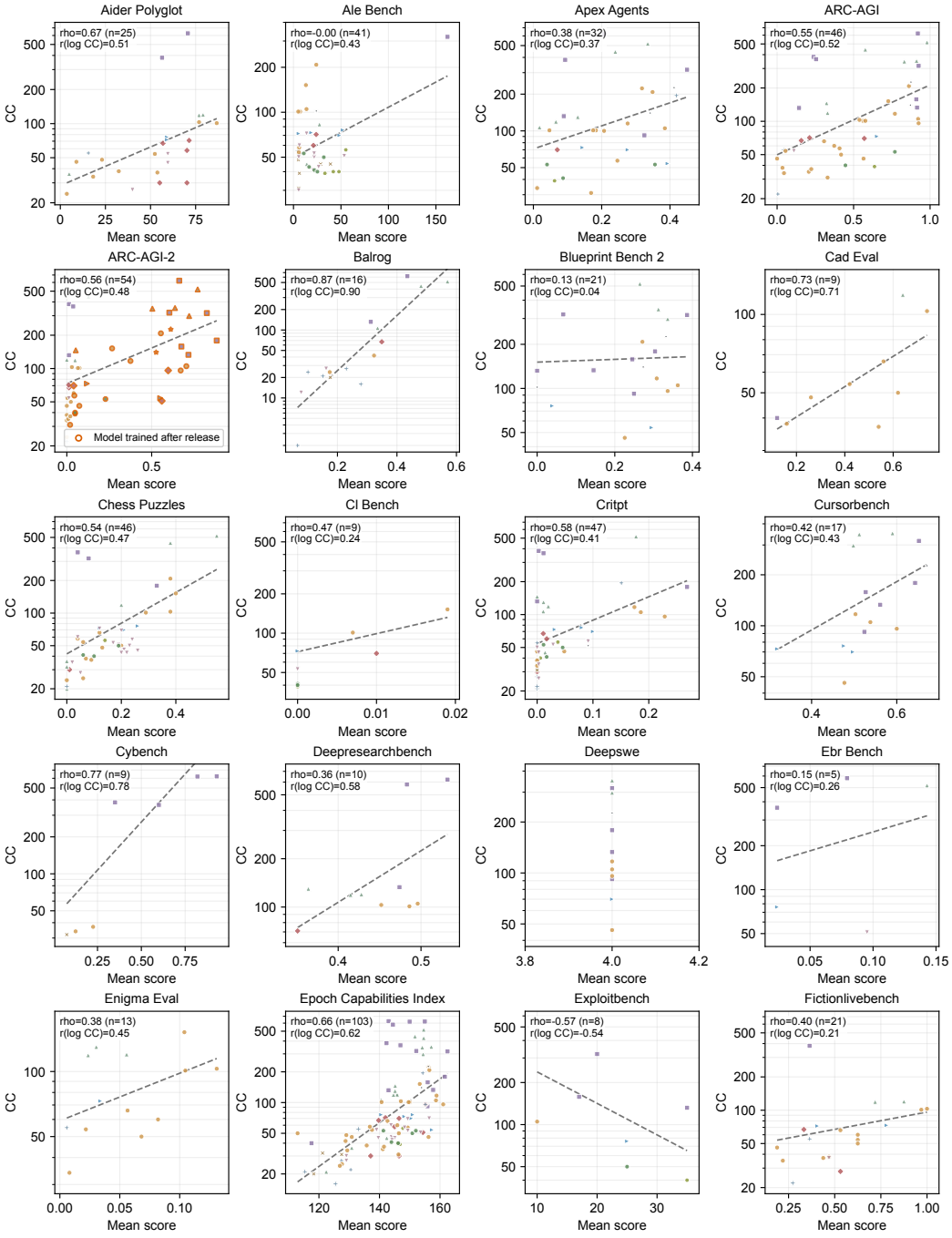

Figure 18: **Comprehensive benchmark correlation panels I.** Pairwise correlation mappings between counting capacity (CC) and established task-based evaluation leaderboards, showing independent capability axes.

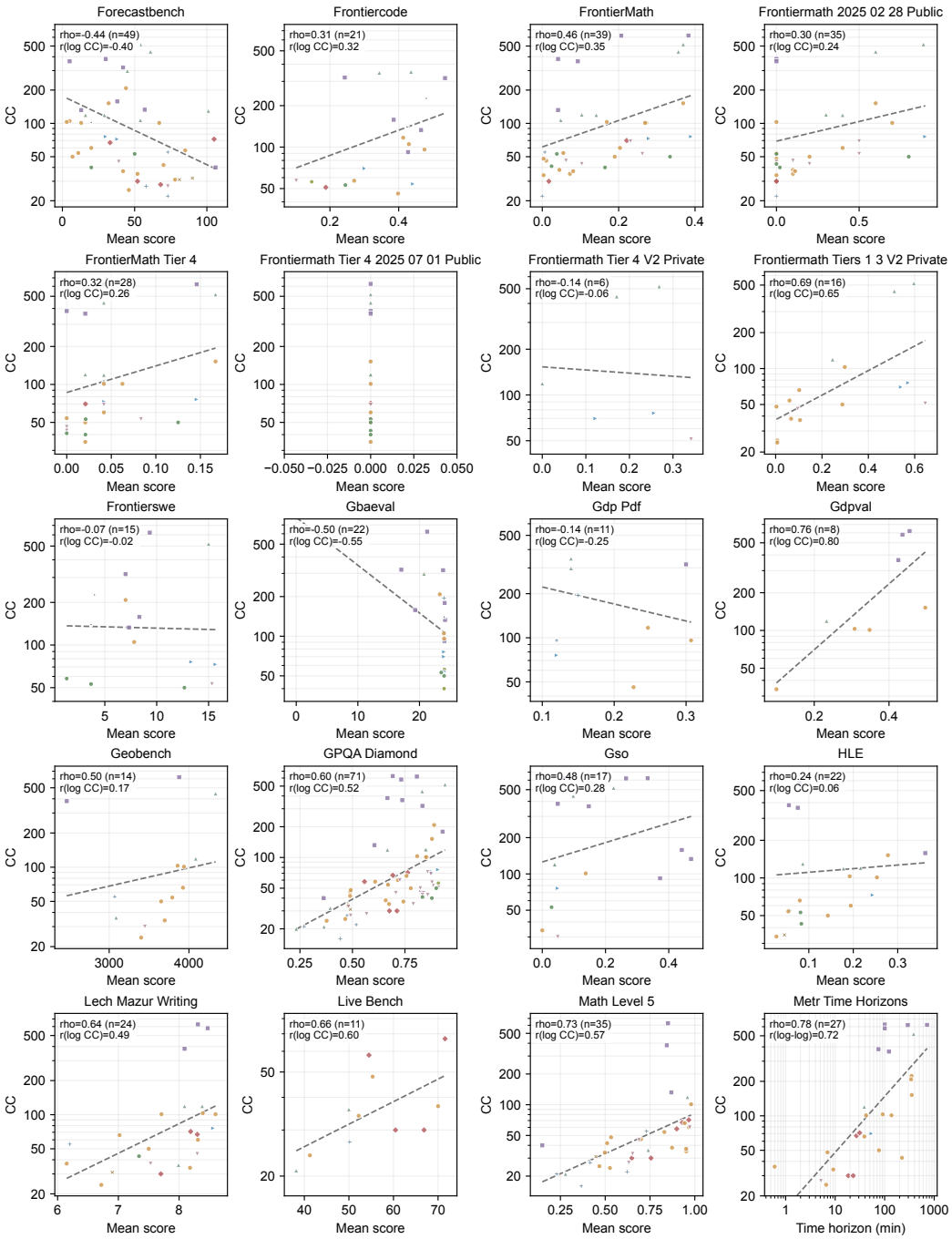

Figure 19: **Comprehensive benchmark correlation panels II.** Additional pairwise correlation mappings.

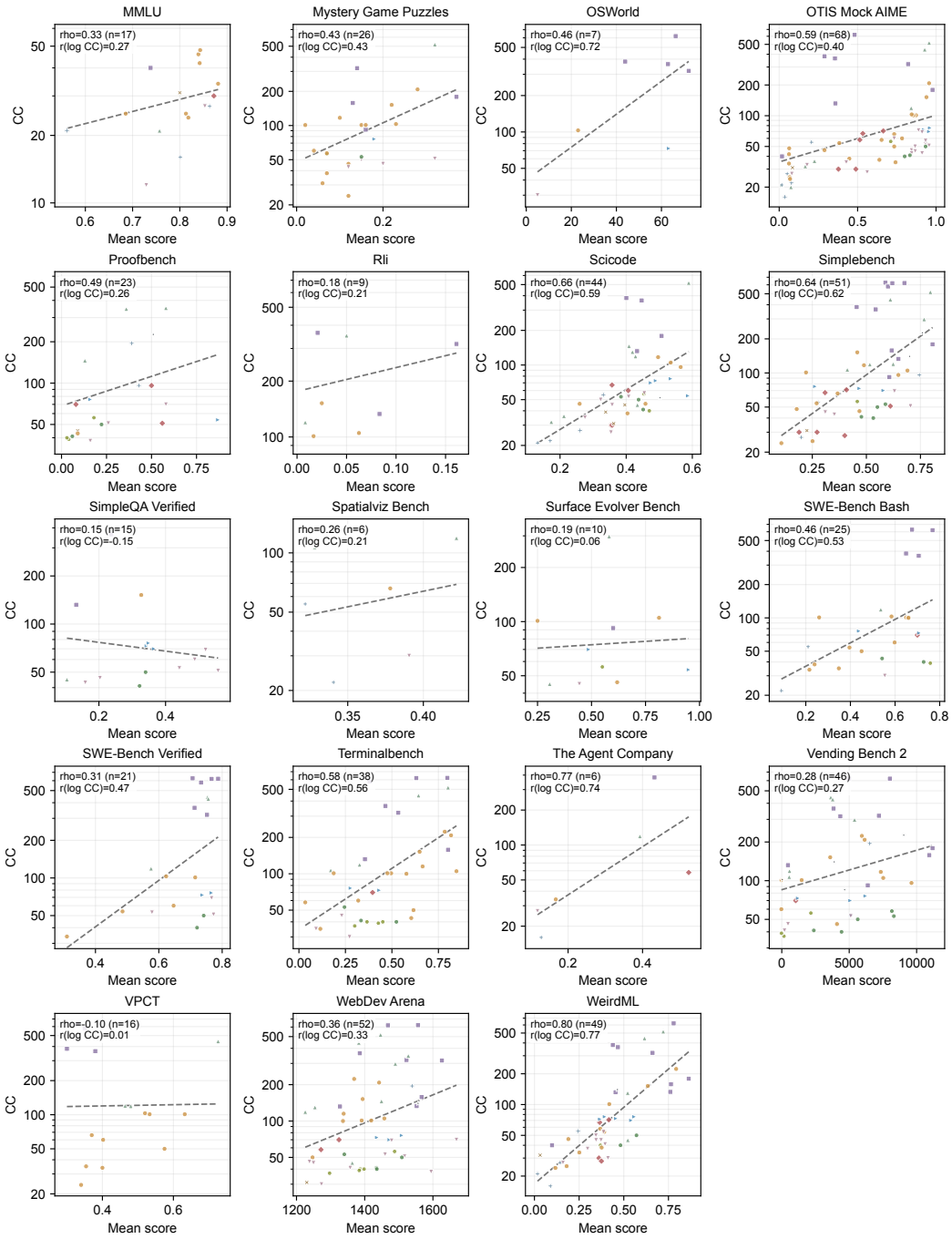

Figure 20: **Comprehensive benchmark correlation panels III.** Additional pairwise correlation mappings.

Results. Forty-six configurations of 33 models completed (Table 23, Fig. 21); the mode of every run was verified from its own replies, and no run contradicted its roster setting. Twenty runs resolve a transition with its tail established, seven declare a transition whose tail is not established, eleven are bounded by the output budget (all with reasoning on), one is still exact at the top of its range, and seven show no stable regime at any length; the last enter the correlations at a capacity of one and are dropped in the sensitivity analysis. With reasoning off, on the 22 models whose counting capacity was measured in the same configuration, tracking capacity follows counting capacity: Spearman $\rho = 0.66$ (bootstrap 95% interval 0.28 to 0.90), with a log-log slope of 0.75 (0.37 to 1.09), so the two capacities scale nearly proportionally; the ratio of tracking to counting capacity has a median of 0.10 with an interquartile range of 0.05 to 0.16. Dropping the seven floor runs leaves $\rho = 0.67$ (0.21 to 0.92) on 15 models, and keeping only fully resolved runs leaves 0.63 (0.09 to 0.92) on 14. The coupling is not carried by general capability: the partial correlation given the Epoch index is 0.65 on the 18 models with a score, while the index itself correlates with tracking capacity at 0.18. Total parameters are available for only 11 open-weight models in this configuration, on which the partial given size is 0.01 and size alone correlates at 0.38, too few to separate the two; the parameter comparison rests on the catalogue-wide analysis of Appendix J. Predicting tracking capacity from the index with vendors held out does worse than the mean, so the incremental test used for METR is uninformative here. The association is largely between vendor families: dropping the five Anthropic models leaves $\rho = 0.31$ on 17, dropping the floor runs as well leaves -0.01 on 11, and the correlation after removing each vendor's mean from both variables is 0.24; within families the samples are small (0.72 on the five Anthropic models, 0.15 on six OpenAI, 0.06 on five DeepSeek), and OpenAI's reasoning-off models track four to seven swaps across a CC range of 25 to 210. Leaving out any other vendor leaves ρ between 0.64 and 0.82. The cross-model coupling is therefore a between-family regularity that a wider within-family sample would have to confirm. Reasoning-off tracking capacity correlates with the METR horizon at 0.44 on the 14 models shared with the leaderboard (0.39 with reasoning on, 17 models). With reasoning on (24 configurations) the correlation with the same models' reasoning-on counting capacity is 0.31 (-0.13 to 0.66). On the 13 hybrids measured in both modes, native reasoning raises tracking capacity in 12, by a median factor of 4.6 (bootstrap interval 2.5 to 59), while raising counting capacity by 1.07. The reasoning-on runs are not all settled: five are bounded by the output budget and one is still exact at its ceiling, so their entries are lower bounds; five declared a transition whose tail was not established, so their entries are uncertain rather than bounds; two are fully resolved. On the eight hybrids whose reasoning-on run is resolved or right-censored the median gain is 3.0, and on the one hybrid fully resolved in both modes, `claude-sonnet-4.5`, it is 1.0 (1.1 for `claude-sonnet-4`, whose reasoning-off tail is unestablished), the immediate capacity already matching the reasoning one. On these 13 hybrids the correlation with counting capacity falls from 0.81 with reasoning off to 0.43 with it on, but the interval on that change (-1.08 to 0.24) includes zero. The history is a written record of the state, so a chain of thought replaces the silent capacity, as the second prediction stated. At and beyond each boundary the last swap is reported correctly in 0.89 of probes and the supplied-state question is answered in 0.96, while tracking accuracy above the boundary is 0.20, at the chance level of one in four: the failure is in the state update, with the text readable and the decision intact. The seven runs with no stable regime are all reasoning-off (`deepseek-chat`, `deepseek-v3.1-terminus`, `deepseek-v3.2`, `gemma-4-26b`, `llama-3.2-1b`, `qwen-2.5-7b`, `qwen3.6-35b`); `qwen3.6-35b` answers with the object's initial location in 9 of 15 probes and never applies a swap. One discrepancy is unresolved: `gemma-4-26b` with reasoning off shows no stable regime here, whereas the same task run locally through the model's own checkpoint in Appendix I.6 tracks one swap perfectly and two at 0.88 with four objects; the prompts, labels, and decoding settings are the same, so the difference lies in the served route or in the extra floor probes at lengths 5 to 8, which the local grid did not include, and it is reported rather than resolved.

Table 23: **State-tracking capacity against counting capacity.** Tracking capacity TC (four objects, swaps until the location of a moved object is no longer the majority answer) and CC for the same model and the same configuration; the event-access and supplied-state rates at the fitted boundary (five probes each). Reasoning is the verified setting: on when the run's replies carried reasoning tokens in more than half of the scan probes. Marks: b, bounded by the output budget, entered at the bound; u, transition declared but its tail not established; c, still exact at the top of the probed range; 0, no stable regime at any length, entered at one in the correlations and dropped in the sensitivity analysis. The subset is fixed by rule (every model with a METR horizon and an index score, the six probe-suite models, and the five lowest-capacity served models) and is skewed toward recent models, as the METR leaderboard is.

Model	Reasoning	CC	TC	TC interval	Event at TC	Supplied at TC
claude-opus-4.6	on	622	300^{b}	[83, 300]	5/5	5/5
claude-opus-4.5	on	620	328^{b}	[328, 328]	5/5	5/5
claude-opus-4.6	off	618	97	[93, 99]	5/5	5/5
claude-opus-4.1	off	580	76	[76, 76]	5/5	5/5
claude-opus-4.5	off	580	131	[117, 131]	5/5	5/5
claude-opus-4.1	on	573	129^{u}	[129, 200]	–	–
gemini-3.1-pro-preview	on	518	550	[508, 550]	5/5	5/5
claude-sonnet-4	off	382	40^{u}	[37, 50]	–	–
claude-sonnet-4.5	off	364	46	[46, 46]	5/5	5/5
claude-sonnet-4.5	on	364	46	[46, 46]	5/5	5/5
claude-sonnet-4	on	290	46	[46, 46]	5/5	5/5
gpt-5.3-codex	on	223	300^{u}	[100, 1040]	–	–
gpt-5.3-codex	off	210	7	[1, 9]	–	–
gpt-5.4	off	208	4	[1, 6]	–	–
gpt-5.4	on	174	300^{u}	[8, 950]	–	–
gpt-5.2	on	152	700^{b}	[700, 750]	5/5	5/5
gemini-2.5-pro-preview	on	120	325	[302, 325]	5/5	5/5
gpt-5.2	off	113	4	[1, 8]	–	–
o3	on	103	400^{b}	[379, 700]	–	–
gpt-5	on	101	275^{b}	[256, 300]	5/5	5/5
gpt-oss-120b	on	101	313^{b}	[100, 324]	–	–
gemma-4-26b-a4b-it	off	83	0^{0}	no stable regime	–	–
deepseek-r1-0528	on	71	61^{b}	[56, 82]	4/5	4/5
deepseek-v3.2	on	70	31^{u}	[31, 50]	–	–
deepseek-r1	on	67	60^{b}	[48, 100]	–	–
o1	on	66	302^{b}	[247, 313]	5/5	4/5
deepseek-v3.1-terminus	on	60	74^{c}	[74, 100]	1/5	4/5
deepseek-v3.2-exp	on	58	157^{b}	[7, 423]	–	–
gpt-oss-20b	on	58	307	[111, 353]	2/5	5/5
muse-glimmer-30b	on	54	309^{u}	[309, 471]	–	–
o4-mini	on	50	162	[112, 162]	5/5	5/5
gpt-4o-2024-08-06	off	48	5	[3, 5]	–	–
qwen3.6-35b-a3b	on	45	59^{u}	[8, 80]	–	–
qwen3.6-35b-a3b	off	42	0^{0}	no stable regime	5/5	5/5
gemma-4-26b-a4b-it	on	41	3^{b}	[2, 25]	–	–
gpt-4o-2024-11-20	off	34	7	[2, 19]	–	–
deepseek-chat	off	30	0^{0}	no stable regime	–	–
deepseek-chat-v3-0324	off	30	48	[41, 52]	5/5	5/5
llama-3.3-70b-instruct	off	27	3	[2, 3]	5/5	5/5
qwen-2.5-72b-instruct	off	27	2	[1, 2]	–	–
gpt-4-turbo	off	25	4	[2, 5]	–	–
deepseek-v3.1-terminus	off	13	0^{0}	no stable regime	4/5	4/5
deepseek-v3.2	off	12	0^{0}	no stable regime	3/5	5/5
qwen-2.5-7b-instruct	off	12	0^{0}	no stable regime	3/5	3/5
deepseek-v3.2-exp	off	10	34	[34, 34]	5/5	4/5
llama-3.2-1b-instruct	off	2	0^{0}	no stable regime	–	–

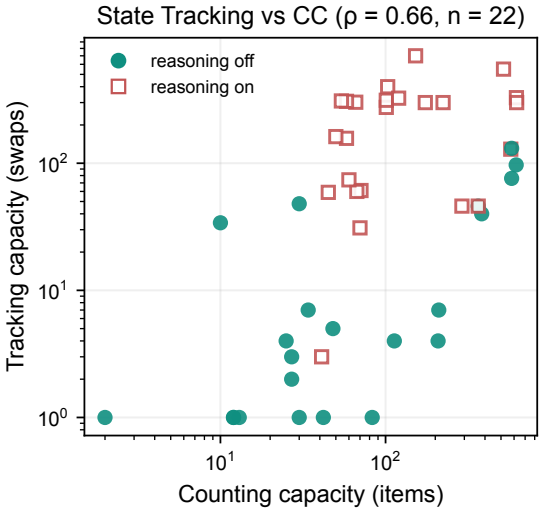

Figure 21: **Tracking capacity against counting capacity.** Each point is one model and configuration: filled, immediate answering; open, native reasoning. Bounded runs are drawn at their bound.