ACE OLYMPIAD MATH, FAIL SIMPLE ARITHMETIC: SYSTEMATIC CAPABILITY UNRELIABILITY IN LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Frontier language models now reach gold-medal-level performance at the International Mathematical Olympiad, yet still make systematic errors on simple tasks like arithmetic. Methods that generate and search over new inputs can expose such failures, but patterns discovered on adaptively selected failures may not generalize, and predictive patterns need not identify what causes them. We introduce a framework that separates failure discovery from inference. Starting from programmatically verifiable task generators, we search for failures, grow local failure clusters, and interpret each cluster with an executable hypothesis. We accept a hypothesis only if it predicts failures on fresh inputs under input-level, false-discovery-rate-controlled validation; designed contrasts then test whether individual input properties actually change the failure rate. Across six capabilities and 17 language models, including five frontier proprietary models, failures are common and strongly clustered, yet adaptive search provides little advantage over uniform sampling at the failure rates we study. More importantly, only 19 of 107 search-derived hypotheses survive validation. In simulations with known ground truth, drawing fresh inputs eliminates 141 of 144 false acceptances near the validation threshold. Designed contrasts further reveal failure-inducing structure that observational validation alone can miss: carry cascades cause addition failures in open-weight models and in three of five frontier models. These results turn failure search from a procedure for collecting errors into a method for making statistically tested claims about where language-model capabilities are unreliable, and suggest that LLM evaluation should move beyond aggregate scores and failure collection toward such validated claims.

1 INTRODUCTION

Frontier language models now reach gold-medal-level performance at the International Mathematical Olympiad (Huang & Yang, 2025), yet they can still fail on elementary tasks. In our experiments, GPT-5.6 Luna answers every two-digit addition and multiplication correctly, but gets about one in twelve ten-digit multiplications wrong at low reasoning effort and reverses the string “bhcahh” incorrectly (Figure 1). Such failures are not artifacts of a few pathological examples: larger language models remain error-prone even on simple task families (Zhou et al., 2024), and their aggregate accuracy on static benchmark does not tell us where those errors occur (Ribeiro et al., 2020; Mirzadeh et al., 2025).

Recent work moves beyond fixed benchmarks by generating inputs. Dynamic benchmarks synthesize new or harder tasks (Zhu et al., 2024; Li et al., 2024; Wu et al., 2026; Srivastava et al., 2026), red-teaming and failure-discovery pipelines search for inputs that elicit failures (Zou et al., 2023; Chao et al., 2023; Samvelyan et al., 2024; Saban, 2026), and ProbeLLM (Huang et al., 2026) organizes the failures it finds into interpretable failure modes, grounded in verifiable test cases. These methods address *discovery*: finding where models fail. We study the question that follows: what can be concluded from a failure pattern found by search? A pattern fitted to the failures a search found looks stronger on those inputs than it is, so it may not hold on inputs the search never visited. And a pattern that does predict failures need not identify their cause, since several properties of an input can change together.

We therefore separate discovery from inference. The distinction is simple:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

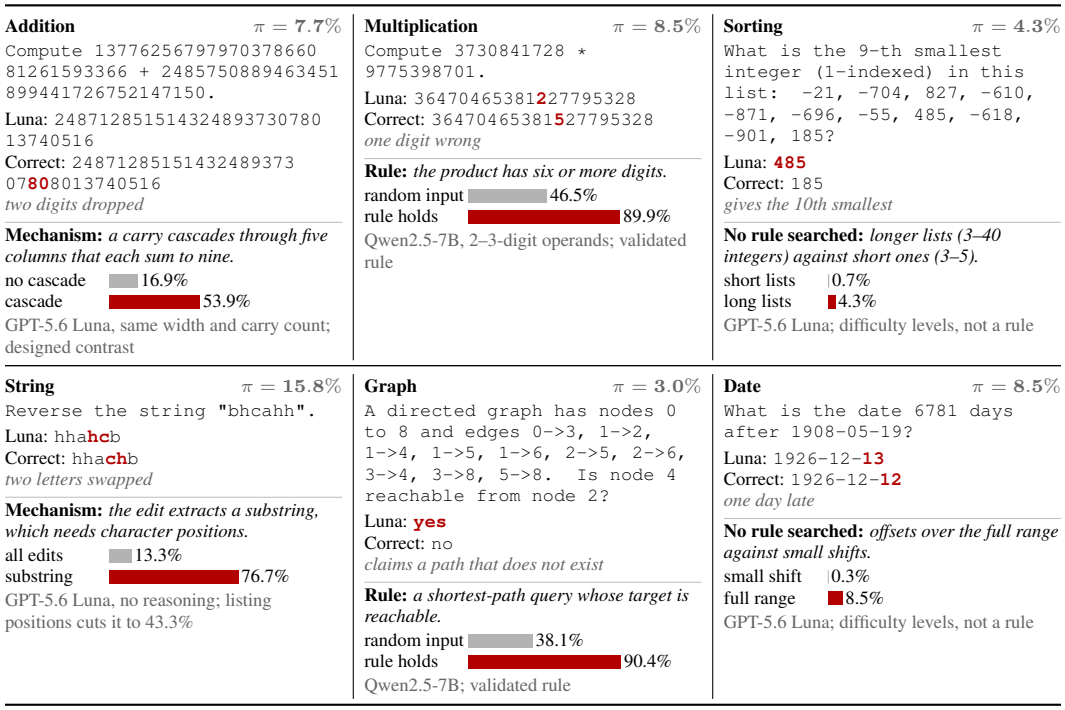

Figure 1: A failure, and where failures like it concentrate, for each task. *Top of each box:* one recorded failure of GPT-5.6 Luna at the low reasoning setting (prompts shown without their answer-format instruction); each wrong answer was given identically on both of two independent samples, red marks where it departs from the correct answer, and π is Luna's failure rate at that level (Table 10). *Bottom:* a condition on the input and its effect, the failure rate of inputs that satisfy it (red bar) against a comparison set (grey bar). A *rule* passed our fresh-input test on an open-weight model; a *mechanism* was confirmed by a designed experiment that varies one property at a time (Tables 31 and 33); for sorting and dates no rule was searched, so the bars only compare two difficulty levels. The example above a condition need not satisfy it.

A failure found by search is evidence of an error, not yet of a pattern.
A pattern found by search is a hypothesis until it predicts failures on fresh inputs.
A pattern that predicts failures is a correlation until a designed contrast shows its cause.

Our framework operationalizes this progression. We search over a space of exactly checkable inputs, grow local clusters around discovered failures, and describe each cluster with a short executable **hypothesis** over interpretable input features. A hypothesis is not accepted because it fits the search data: it must predict failures on a fresh sample of inputs, with each input counted once, under anytime-valid inference and false-discovery-rate control across all hypotheses. For selected hypotheses, we then construct **designed contrasts** that change one input property while holding relevant alternatives fixed. A property whose controlled variation changes the failure rate is a **rule**. Because every answer is checked by a programmatic oracle, the procedure requires no judge model and can generate as many fresh test cases as needed.

We apply this framework to six simple capabilities—integer addition and multiplication, sorting and ranking, string edits, date arithmetic, and graph queries—across twelve open-weight models and five frontier proprietary models. Our study yields four findings and methodological contributions.

First, simple capability failures are widespread and locally structured, but efficient discovery remains an open problem. Every tested model fails on at least one capability, and in 161 of 164 settings, inputs near failures are more likely to fail than neighbours of correct inputs. Yet at the failure rates we study, the adaptive search strategies we test offer little advantage over uniform sampling in discovering distinct failure regions. This points to a need for more targeted search methods as the objective shifts from finding an error to systematically covering the reliability gap.

Second, failure patterns discovered by search are often not reliable without independent validation. We obtain 107 hypotheses from failure clusters; only 19 survive our input-level, anytime-valid testing with false-discovery-rate control. This filtering is important: in simulations with known failure regions, fresh inputs eliminate 141 of 144 false discoveries that appear plausible when patterns are assessed on the data used to find them.

Third, controlled contrasts reveal failure-inducing properties that search alone cannot establish. A validated hypothesis identifies where failures concentrate, but not necessarily which input property causes them. By constructing matched contrasts that vary one property at a time, we show that carry cascades increase addition failures in open-weight models and in three of five frontier models. This also illustrates why a hypothesis need not itself survive validation to motivate a useful experimental test: the cascade hypothesis is rejected by our validation procedure, yet its underlying input property is confirmed by the controlled contrast.

Fourth, failure rules enable targeted intervention. Supervised fine-tuning targeted at cascade failures reduces the targeted errors but also introduces failures elsewhere. The result suggests that failure analysis could identify precise intervention targets, while repair itself requires more careful and targeted methods than simply training on the observed failures.

2 Related work

Reliability on simple tasks. Larger and more instruction-tuned models still err on easy instances of simple tasks such as addition and anagrams, so no range of difficulty is free of errors (Zhou et al., 2024). Accuracy on deterministic tasks depends on how probable the task and its output are (McCoy et al., 2024); arithmetic accuracy falls with operand length and composition depth (Dziri et al., 2023; Bie et al., 2026; Yang et al., 2025), and pretrained models break basic rules of arithmetic (Yan et al., 2025) and compute it with a bag of heuristics (Nikankin et al., 2025). Models struggle to manipulate the characters inside the sub-word pieces, or tokens, that they read (Edman et al., 2024), simple puzzles break frontier models (Nezhurina et al., 2024), and the largest error-free range of a problem family has been proposed as a reliability measure (Sato, 2026). Our first finding extends this line to frontier models and to complete enumerations.

Arithmetic mechanisms. In small transformers trained on addition, the hard case is a carry that cascades through columns summing to nine (Quirke & Barez, 2024; Kruthoff, 2024); a one-digit lookahead account predicts the same failure in pretrained models adding several numbers (Baeumel et al., 2025); and the way numbers are split into tokens changes arithmetic accuracy in frontier models (Singh & Strouse, 2024). We confirm the cascade account in pretrained and frontier models with the number of carries held fixed, and without sampling error on a complete enumeration.

Finding and describing failures. Slice discovery (Eyuboglu et al., 2022; Jain et al., 2023; Chen et al., 2025; Konforti et al., 2026) groups the errors in a fixed labelled dataset into coherent subsets, and Slice Finder (Chung et al., 2019) tests such subsets with false-discovery control. Failure-mode discovery, benchmark synthesis and red-teaming (Huang et al., 2026; Saban, 2026; Li et al., 2024; Wu et al., 2026; Zhu et al., 2024; Srivastava et al., 2026; Zou et al., 2023; Chao et al., 2023; Samvelyan et al., 2024) search for failures under a budget, usually judged by another model; ProbeLLM (Huang et al., 2026), the closest to us, reports failure modes found by probing verifiable cases adaptively, whereas we require each mode to be an executable rule accepted on fresh inputs. Software testing has long used the observation that failure-causing inputs cluster: adaptive random testing spreads tests evenly for this reason (Chen et al., 2004), although a large study found it barely better than random testing once its cost is counted (Arcuri & Briand, 2011). Alhazen (Kampmann et al., 2020) and DDSET (Gopinath et al., 2020) learn the input conditions under which a program fails and check them on generated inputs, and Anchors (Ribeiro et al., 2018) explains classifiers with high-precision rules. Template-based behavioural tests (Ribeiro et al., 2020; Mirzadeh et al., 2025) also look past accuracy, but on templates fixed in advance. Here descriptions are found by search, and since a language model's answer to an input is sampled, accepting one is a statistical decision (Appendix M).

Inference after selection. Estimating an effect on the data that selected it is biased, which is why honest estimation uses separate samples (Athey & Imbens, 2016) and adaptive data analysis needs fresh or protected data (Dwork et al., 2015). We use a fresh sample per rule, control

the false-discovery rate (Benjamini & Hochberg, 1995) across all candidates with e-BH (Wang & Ramdas, 2022), the Benjamini–Hochberg procedure applied to e-values (a betting-based measure of evidence), which tolerates dependence between rules, and bound each rule's failure rate with a confidence sequence that stays valid at any sample size (Howard et al., 2021); related tools audit models with anytime-valid guarantees (Zhou et al., 2026). Each tool is standard; what is new is their combination for rules found by searching for failures, and a measurement of what each one removes. Active testing (Kossen et al., 2021; Berrada et al., 2025) labels items of a fixed benchmark to estimate one score, and reasoning at test time buys accuracy with computation (Snell et al., 2025); we generate new inputs, return rules, and measure how much of the gap reasoning closes.

3 METHOD

Failures. A *capability space* $\mathcal{S}_C = (\mathcal{X}_C, f_C, d, \nu, \phi)$ consists of valid inputs $\mathcal{X}_C$ produced by a generator, an oracle f_C returning the correct answer, a neighbour relation d (one small edit, such as changing one digit), a reference distribution ν, uniform within a difficulty level, and interpretable features ϕ, such as operand lengths or the number of carries. A model fails on x when its parsed answer differs from $f_C(x)$; unparseable answers count as failures. We describe a model by its failure probability $p(x) = \Pr[M(x) \neq f_C(x)]$ and its *background rate* $\pi = \mathbb{E}_{x\sim\nu}[p(x)]$, the quantity a benchmark estimates. A *cell* is a (model, capability, level) combination.

Search. A search method spends $B = 2{,}000$ model calls choosing inputs adaptively and is scored by *coverage*, the number of distinct bins of a feature grid in which it finds a failure, since each distinct failure can seed a distinct cluster, and by the number of inputs queried before the first failure. We compare uniform sampling from ν, hill climbing, MAP-Elites (Mouret & Clune, 2015), and proposals by the target model itself (Appendix A.2).

Clusters. Failures *cluster* if a neighbour of a failing input fails more often than a neighbour of a correct one. We measure this in every open-weight cell with a failure rate between 0.5% and 95%, by answering eight neighbours of each of 300 inputs, and on complete enumerations by the failure rate at each edit distance from a failure. A *cluster* is a connected set of inputs on which the model fails at rate at least τ (defined below); we grow one from each failure found by search, admitting neighbours with a sequential test on at most 16 answers (Appendix A.3).

Hypotheses and validation. A *hypothesis* h is a conjunction of thresholds on the features, such as "the second operand has at least three more digits than the first", fitted to separate a cluster from inputs that pass; no language model proposes or interprets it. Its failure rate is $p_h = \mathbb{E}_{x\sim\nu}[p(x) \mid h(x) = 1]$ and its *lift* is p_h/π. A hypothesis found by searching for failures overstates p_h, so we call it *validated* only if, on 300 fresh inputs satisfying it, each answered four times and counted once, its anytime-valid lower confidence bound (Howard et al., 2021) exceeds $\tau = \max(0.5, \min(0.95, \hat{\pi} + 0.15))$ and e-BH (Wang & Ramdas, 2022) holds the false-discovery rate at 5% across all hypotheses proposed. A validated hypothesis predicts *where* the model fails, but not *why* (Appendix A.4).

Rules. A *rule* is a property of the input that, changed with the other properties that could explain the failures held fixed, changes the failure rate; it is a claim about the input, not about circuits inside the model. We test a candidate rule with a *designed contrast*: two sets of inputs matched on those properties and differing in the one under test, such as sums with the same width and number of carries that differ only in whether the carries cascade, using a Fisher exact test. A candidate may come from a hypothesis, validated or not, or from prior work, so a hypothesis that fails validation can still lead to a rule (Appendix A.5).

Instantiation. We study six capabilities: integer addition and multiplication, sorting and ranking, string edits, date shifts, and graph reachability and shortest paths (Table 4). Twelve open-weight models answer directly (Qwen2.5 0.5B–14B, Granite-3.1 2B and 8B, Phi-3.5-mini, Phi-4-mini, Mistral-7B, Falcon3-3B, SmolLM2-1.7B; 4-bit weights at 7B and above), and five frontier models are queried at a low and a high reasoning setting (GPT-5.6 Luna, Gemini 3.8 Flash, Claude Sonnet 5, GLM-5.3, DeepSeek v4.1 Flash). Settings mean different amounts of reasoning for different providers, so we compare settings within a model (Appendix B). Each cell holds 300 inputs answered twice, at temperature 1 for open-weight models; frontier models are also measured at harder levels, such as 31- to 34-digit addition. We enumerate every two-digit addition and multiplication for all open-weight models and every three-digit addition for six of them at temperature

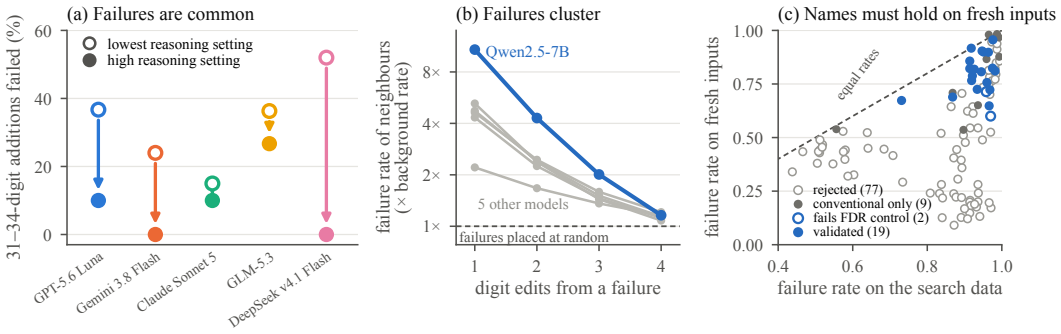

Figure 2: Average accuracy says how often a model fails, not where. On tasks that a short program solves exactly, failures are **(a) common**: every frontier model fails many additions of two 31- to 34-digit numbers at its lowest reasoning setting (hollow), and high reasoning (filled) removes these failures for Gemini 3.8 Flash and DeepSeek v4.1 Flash only (the square is GPT-5.6 Luna's low setting; * marks models that reason at every setting). They are **(b) clustered**: over all 810,000 three-digit additions, sums one digit away from a failure of Qwen2.5-7B fail at 10.8 times the model's background rate (its failure rate on random sums), and the excess fades within four edits. And clusters can be **(c) identified**, but a rule must hold on fresh inputs: each of the 107 candidate rules found by search is placed by its failure rate on the inputs that grew it and on 300 fresh inputs. Points below the dashed line fail less often on fresh inputs, by up to 0.81; 30 candidates pass a conventional test and 19 pass ours (filled blue; FDR is the false-discovery rate). The rate a rule must beat rises with the background rate where it was found, so some rules with high fresh rates are rejected.

0. Search and cluster growth run on addition, multiplication and graph queries, in cells with failure rates mostly between 0.05 and 0.5. Intervals are 95% Wilson intervals over inputs.

Each capability has its own features for rules; addition has 23, including the operand widths, their signed length difference, the number of carries and the longest carry cascade (Appendix B).

Models. Twelve open-weight instruction-tuned models answer directly: Qwen2.5 with 0.5B, 1.5B, 3B, 7B and 14B parameters, Granite-3.1 with 2B and 8B, Phi-3.5-mini, Phi-4-mini, Mistral-7B, Falcon3-3B and SmolLM2-1.7B; the 7B, 8B and 14B models run with 4-bit weights and the others with 16-bit weights. Five frontier models are queried through their APIs: GPT-5.6 Luna, Gemini 3.8 Flash, Claude Sonnet 5, GLM-5.3 and DeepSeek v4.1 Flash. The *low* setting is reasoning off for Claude Sonnet 5 and DeepSeek v4.1 Flash, minimal for Gemini 3.8 Flash and GLM-5.3, which reason at every setting, and low for GPT-5.6 Luna; the *high* setting is each API's high effort, and each model's *lowest* setting is its low one except for Luna, whose lowest is no reasoning. The settings mean different amounts of reasoning for different providers, so we compare settings within a model and read differences between models as description, not ranking (endpoints and dates in Appendix B).

Measurements. Open-weight cells are answered at temperature 1 and frontier cells at the provider's default temperature; frontier models are also measured at a harder level of each task, where they begin to fail, such as additions of two 31- to 34-digit numbers. We enumerate every two-digit addition and multiplication (8,100 problems each) for all twelve open-weight models, and every three-digit addition (810,000 problems) for six of them, at temperature 0. Cluster search and rule validation run on addition, multiplication and graph queries, the capabilities whose failure rates leave room for a cluster to stand out ($0.05 \leq \pi \leq 0.5$). Intervals are 95% Wilson intervals computed over inputs unless stated otherwise.

4 Results

4.1 Failures are common and easy to find

Every open-weight model fails on every task. All 72 failure rates in Figure 3a are above zero. The strongest model in each column still fails 7% of additions of three- and four-digit numbers, 5%

(a) Open-weight models

	failure rate, easiest level						errors /8,100	
	add 3–4	mul 2×2	sort 3–5	string 4–8	date small	graph 5–10	+	×
Qwen2.5-0.5B	0.48	0.59	0.88	0.81	0.98	0.79	1,247	4,744
Qwen2.5-1.5B	0.23	0.37	0.57	0.61	0.83	0.64	150	2,958
Qwen2.5-3B	0.13	0.21	0.40	0.67	0.79	0.20	188	1,652
Qwen2.5-7B	0.07	0.12	0.41	0.60	0.42	0.23	10	1,024
Qwen2.5-14B	0.07	0.05	0.25	0.41	0.43	0.22	10	501
Granite-3.1-2B	0.34	0.49	0.57	0.69	0.64	0.42	470	3,807
Granite-3.1-8B	0.32	0.29	0.55	0.67	0.70	0.44	334	2,078
Phi-4-mini	0.83	0.35	0.47	0.63	0.91	0.56	1,303	973
Phi-3.5-mini	0.93	0.44	0.58	0.56	0.80	0.56	3,716	2,353
Mistral-7B	0.72	0.80	0.90	0.80	0.81	0.65	3,969	6,514
Falcon3-3B	1.00	0.97	0.63	0.90	0.98	0.64	7,682	8,020
SmolLM2-1.7B	0.94	0.92	0.91	0.84	0.94	0.69	7,084	7,222

(b) Frontier models, low reasoning setting

	failure rate (%)				
	GPT-5.6 Luna	Gemini 3.8 Flash*	Claude Sonnet 5	GLM 5.3*	DeepSeek v4.1 Flash
add 3–4 digits	0.0	0.0	0.0	1.0	0.0
add 31–34 digits	7.7	24.0	15.0	36.3	52.0
mul 2×2 digits	0.0	0.0	0.0	0.0	0.0
mul 10×10 digits	8.5	8.0	32.0	18.2	98.0
sort 3–5 integers	0.7	0.0	3.0	2.0	11.0
sort 3–40 integers	4.3	1.0	17.0	10.0	14.0
string 4–8 chars	15.8	3.0	21.0	16.0	25.0
string 4–40 chars	13.5	11.0	18.0	14.0	32.0
date, small offset	0.3	0.0	0.0	1.0	1.0
date, full range	8.5	1.0	8.0	12.3	30.0
graph 5–10 nodes	0.8	0.0	0.0	1.0	4.0
graph 5–25 nodes	3.0	0.0	1.0	5.0	10.0

Figure 3: Failure rates of every model. (a) Open-weight models at the easiest level of each task (300 inputs, two answers each), and exact error counts among all 8,100 two-digit additions (+) and multiplications (×). (b) Frontier models at the low reasoning setting, in percent (600 answers per cell; intervals in Table 10); * marks models that reason at every setting. An unparseable answer counts as a failure.

of two-digit multiplications, 25% of sorting and ranking questions on three to five integers, 41% of edits to strings of four to eight characters, 42% of small date shifts, and 20% of graph queries on five to ten nodes. Exhaustive enumeration removes sampling error from the arithmetic: every model gets at least 10 of the 8,100 two-digit additions wrong and at least 501 of the 8,100 two-digit multiplications wrong.

Frontier models also fail on tasks that remain trivial for exact programs. At the low reasoning setting, no frontier model misses a two-digit multiplication, and only GLM-5.3 misses any three- or four-digit addition (1.0%; Figure 3b); GPT-5.6 Luna answers all 8,100 two-digit additions and multiplications correctly. At larger widths, however, all five models fail: long additions fail at rates of 7.7–52.0%, long multiplications at 8.0–98.0%, and short string edits at 3.0–25.0%. Four models fail on all six tasks, while Gemini 3.8 Flash fails on five. The failures are concrete rather than merely aggregate: GPT-5.6 Luna, for example, reverses “bhcahh” as “hhahcb” (Appendix L).

More reasoning reduces some failures, but not all. Across long addition, long multiplication, and short string edits, high reasoning lowers the measured failure rate in all fifteen model–task cells and removes every failure in seven: all of Gemini 3.8 Flash’s and DeepSeek v4.1 Flash’s cells, and GLM-5.3’s string edits (Figure 2a; Table 32). Long addition benefits least. GPT-5.6 Luna and Claude Sonnet 5 still fail 10.0% of 31- to 34-digit additions at high reasoning, while GLM-5.3 fails 26.7%. For Luna, the failure rate is 7.7% at low reasoning and 10.0% at high reasoning.

Finding failures is not the bottleneck. Uniform sampling reaches a first failure after about $1/\pi$ draws, at most about 20 for every open-weight cell in Figure 3a, while adaptive search is no faster (Section **??**). Across eleven cells with five to ten random seeds each, hill climbing reaches fewer distinct failure bins than uniform sampling in 20 of 24 comparisons, self-proposal is at parity or worse except in one saturated cell, and MAP-Elites changes coverage by only -1.5% to $+8.3\%$ in the nine open-weight cells with enough bins to rank (Appendices E and K). Since failures are only a few draws away, the useful problem is not finding a failure but determining whether its apparent structure generalizes.

How an input is written matters, and differently for each model. Every rate above uses one fixed template per task, so we test how much the written form of an input, rather than its content, moves the failure rate (Table 1). Listing every character of a string with its position leaves a substring question and its answer unchanged, yet it removes most substring failures of Claude Sonnet 5 and DeepSeek v4.1 Flash, about four in ten of GPT-5.6 Luna’s, raises GLM-5.3’s failure rate by 13 points and leaves Gemini 3.8 Flash unchanged. For Sonnet it nearly matches high reasoning. Other changes of surface form do not move failures. Forcing open-weight answers into the required format by constrained decoding changes Qwen2.5-3B’s addition failure rate by at most 0.004 at four of five widths, so failures are not an artefact of how answers are written, and the operand-order effect reverses when the operands are swapped and survives zero-padding, so it follows where the longer

Change of written form	Model	Task	before	after	change, points
Input format: list each character with its position (lowest reasoning setting; high reasoning for reference)					
	Claude Sonnet 5	substring, 4–8 chars	69.0	13.3	−55.7 [−61.7, −48.7]
	DeepSeek v4.1 Flash		80.0	30.0	−50.0 [−56.4, −42.7]
	GPT-5.6 Luna		76.7	43.3	−33.3 [−40.4, −25.7]
	GLM-5.3		30.0	43.3	+13.3 [+5.6, +20.8]
	Gemini 3.8 Flash		3.3	3.3	+0.0 [−3.1, +3.1]
Output format: constrained decoding to the answer format					
	Qwen2.5-3B	addition, 3–4 digits	9.2	12.8	+3.6
		addition, 5–12 digits (4 levels)			−0.3 to 0.0
Operand layout: 2nd-longer minus 1st-longer failure rate					
	Qwen2.5-7B	addition, 5–8 digits	+29	swapped −28	reverses
				zero-padded +24	persists
	Qwen2.5-3B		+18	swapped −25	reverses
				zero-padded +14	persists

Table 1: How the written form of an input moves failure rates (percent). Listing character positions changes neither the question nor the answer, yet moves substring failure rates by up to 56 points, in either direction depending on the model; at high reasoning the plain prompt fails 26.7%, 10.0%, 3.3%, 10.0% and 3.3% for Luna, Sonnet, DeepSeek, GLM and Gemini. Constrained decoding barely changes addition failures, and the operand-order effect follows the position of the longer number, not the layout of its digits. Intervals: 95% Newcombe, over inputs. Sources: Tables 33, 11 and 36.

number is written, not how its digits line up. A failure rate is therefore a property of a model, an input and the way the input is written; every rate and rule in this paper is measured under the fixed templates of Table 4.

The excess is local, not just shared difficulty. Complete enumerations of three-digit addition have no sampling error (Figure 2b). Because every comparison is at the same operand width, the 10.8× increase cannot be explained by moving to a harder width. Sums one digit away from a failure fail at 2.2 to 10.8 times the model's background rate and at 4.0 to 14.2 times the rate around a correct sum (Table 8). The excess shrinks by roughly half or more with each further edit and is within 21% of the background by four edits; failures placed at random show none. Correct answers cluster as well: the neighbours of a correct sum fail at 0.56 to 0.79 times the background rate. More accurate models have more concentrated failures: across the Qwen2.5 sizes enumerated, the one-edit ratio against correct sums rises from 4.0 at 0.5B to 14.2 at 7B. Because the neighbours of a correct input fail less often than average, searching locally from a correct input is a poor way to find a first failure.

4.2 Interpreting clusters with hypotheses that hold on new inputs

labelsec:regions

Search produces many plausible hypotheses, but most do not survive validation. Twenty-three search runs on eight open-weight models grew 107 clusters and produced one hypothesis per cluster: 92 on addition, 11 on graph queries, and 4 on multiplication. Thirty hypotheses pass a conventional fixed-sample test that counts answers, 24 remain when inputs rather than answers are counted, 21 pass the anytime-valid bound, and e-BH over all 107 candidates leaves 19 validated hypotheses (Appendix F.1). The main losses occur before multiple-testing correction: repeated answers and optional stopping can make apparent failure rates too optimistic. On the real candidates, the failure rate on the inputs that grew a region exceeds the rate on fresh inputs by up to 0.81 (Figure 2c).

Fresh-input validation removes search-induced patterns. Two Luna candidates illustrate the problem: although they contain 314 and 307 search-generated members, their failure-rate gaps on fresh inputs fall to 0.49 and 0.40, respectively, and neither survives validation (Appendix **??**). The simulated experiments show why this matters. With planted regions whose true lifts range from 1.2 to 3.0, reading the failure rate from the inputs used to grow a region accepts all 144 simulated regions at lift 1.6 whose true failure rate is 0.48 against a threshold of 0.50. One fresh sample removes 141 of these false discoveries (97.9%), while counting inputs removes the remainder. The anytime-valid

Validation step	Test applied	Hypotheses kept
Candidate generation	Search-generated hypotheses	107
Input-level validation	Count inputs rather than answers	24
Anytime-valid testing	Protect against optional stopping	21
Multiple-testing control	e-BH over all 107 candidates	19

Table 2: Validation funnel for the 107 search-generated hypotheses. Each stage removes candidates that fail an additional protection against repeated answers, optional stopping, or multiple testing. The conventional answer-counting test retains 30 candidates; input-level validation reduces this to 24 before the anytime-valid and e-BH stages.

layer then sacrifices about four in ten genuine regions at lift 2.0, quantifying the cost of protection against optional stopping (Appendices C and F.2).

The surviving hypotheses describe structured failure regions, not only generic difficulty. The 19 validated hypotheses have lifts of 1.5 to 3.2 on fresh inputs (Table **??**). Fourteen occur on Qwen2.5-3B, 7B, and 14B and five on Granite-3.1-2B and 8B. Every validated addition hypothesis involves a difference in operand length, most often with the second operand longer, and the pattern appears independently in two model families with different developers and tokenizers. Graph hypotheses depend on query type and path length, while multiplication hypotheses depend on operand and product size. Some of these conditions are difficulty-related proxies, such as product width or path depth. The addition hypotheses provide a stronger test: at fixed widths, simply swapping the operand order moves the failure rate, so the effect cannot be explained by width alone (Section 4.3). On GPT-5.6 Luna, the open-weight addition hypotheses produce lifts of only 1.01 to 1.27, and no Luna candidate is validated (Appendix **??**).

Search is useful for generating hypotheses, but search data cannot establish that a pattern is real; fresh inputs and multiplicity-aware validation are what turn a candidate pattern into a validated hypothesis.

4.3 Designed contrasts turn hypotheses into rules

A *validated hypothesis* identifies a region where failures concentrate; a *rule* goes further by identifying an input property whose controlled variation changes the failure rate. We test this distinction with designed contrasts: swap the operands while holding their values fixed, construct sums with and without carry cascades while matching width and carry count, and alter only the representation of string positions. These interventions test whether the patterns found by search persist when the relevant input property is changed directly.

Operand order changes failure at fixed width. Every validated addition hypothesis involves a difference in operand length, and most specify that the second operand is longer. On uniform samples of five- to eight-digit sums, Qwen2.5-7B fails on 63.9

Takeaway. The failure pattern follows the written position of the longer operand, not simply the width of the numbers.

Carry cascades cause failure even when width and carry count are fixed. Prior work on small transformers and pretrained models identifies carry cascades as a difficult case in addition (Quirke & Barez, 2024; Kruthoff, 2024; Baeumel et al., 2025). Cluster search on Qwen2.5-3B also returns a cascade condition as a hypothesis, but validation rejects it: its fresh failure rate is 0.536, above the 0.50 threshold, while a fixed-sample bound over the same 300 inputs is 0.488. The number of carries and their cascade length are correlated in random samples, so observational data cannot separate them. We therefore construct a contrast whose two arms match width and number of carries and differ only in whether the carries cascade (Section 3). At width 6 on Qwen2.5-3B, sums with a four-column cascade fail at 0.577 versus 0.044 without a cascade, with disjoint anytime-valid intervals on all three open-weight models tested (Figure 4a, Table 28). Complete enumeration of three-digit sums removes sampling error: with three carries, Qwen2.5-7B fails on 8 of 91,125 non-cascading sums and 6,994 of 40,050 cascading sums, a $1989\times$ ratio. Without a cascade, its failure rate instead falls as the number of carries increases (Figure 4b). The effect is not driven by the digit

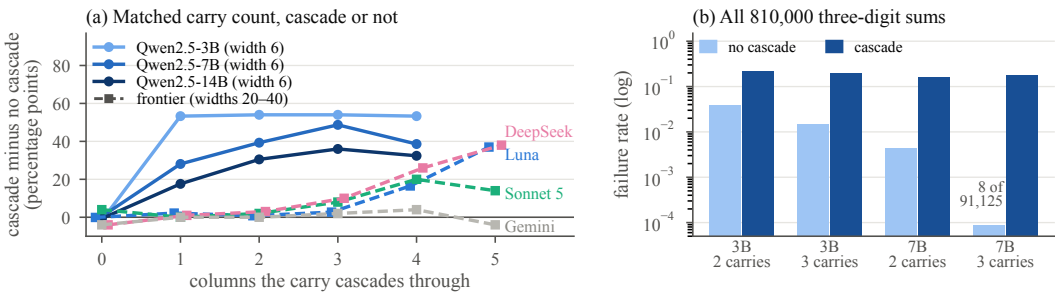

Figure 4: Carry cascades drive addition failure at a fixed number of carries. (a) Difference in failure rate between sums whose carries cascade through k columns and sums with the same width and number of carries whose carries do not cascade; GLM-5.3 is omitted because its arms already differ at $k = 0$ (Table 31). (b) Failure rates of three-digit sums with two or three carries, with and without a cascade, over all 810,000 sums.

nine: among sums in which neither operand contains a nine, 1 of 21,952 non-cascading sums fail versus 2,339 of 14,336 cascading sums (Appendix G).

The same contrast identifies frontier failures. We repeat the carry contrast on the five frontier models at the low reasoning setting, using operand widths of 20 to 40 digits chosen per model (Table 31). An effect counts when a Fisher exact test, corrected by Holm's method for six lengths, rejects equal rates; models whose arms already differ at length zero are set aside. On GPT-5.6 Luna, cascades through up to three columns change the failure rate by less than three points, while four-column cascades raise it from 17.0

Takeaway. Cascade length separates failing from non-failing sums beyond what width and carry count alone explain, and the same contrast appears in several frontier models.

String failures also depend on representation. Extracting a substring requires locating individual characters, which multi-character tokens can obscure (Edman et al., 2024). Listing each character with its position therefore tests whether explicit position information changes the failure rate (Table 33). At the lowest setting, which is no reasoning for these models, listing positions cuts substring failure from 69.0

Takeaway. Explicit character positions remove much of the substring difficulty for three models, although the effect is model-dependent.

A rule identifies a repair target, but closing it can create collateral failures. We fine-tune Qwen2.5-3B with low-rank adapters (LoRA) on 8,000 problems at operand widths 4 to 8 and evaluate on all 810,000 three-digit sums, a width never seen during training. The three training arms differ only in which problems they see: sums with a carry cascade, uniformly sampled sums, or sums with no carries (Table 3). Targeted training on cascades reduces failure on the 17,100 three-digit sums covered by the cascade rule from 29.2

Takeaway. Rules can identify precise repair targets, but aggregate accuracy does not reveal whether an intervention fixes the targeted failures without creating new ones.

Failure rate on	base	cascade rule	uniform	no carries
sums the cascade rule covers (n=17,100)	0.292	**0.004**	0.120	0.719
sums with no carries (n=108,900)	0.025	0.058	0.004	**0.000**
all three-digit sums (n=810,000)	0.047	0.023	**0.012**	0.271
problems fixed / newly broken	—	35,161 / 15,773	33,449 / 4,769	13,062 / 194,316

Table 3: Three fine-tuning arms of Qwen2.5-3B, differing only in their 8,000 training problems, evaluated on all 810,000 three-digit sums at temperature 0. Bold marks each row's lowest failure rate.

5 LIMITATIONS

The tasks are short and exactly checkable; we make no claim about open-ended generation, code, or long reasoning problems. Hypotheses can only use the features defined for each task, so a failure no feature describes cannot be interpreted. The validated hypotheses come from two open-weight families whose size and weight precision vary together, so no trend here isolates scale. The frontier models are production tiers behind changing APIs, probed with one prompt per task (plus a positions-listed prompt for substrings); their reasoning settings differ across providers, and frontier clustering is measured on GPT-5.6 Luna alone. Clusters are grown on three of the six tasks and, among frontier models, only on GPT-5.6 Luna, where no hypotheses is validated, so frontier failures are identified here only by designed tests. Region growth uses a truncated sequential test that wrongly admits a neighbour 19% of the time, which shapes regions but not their acceptance (Appendix D). The repair covers one region of one model.

6 CONCLUSION

On six tasks that a short program solves exactly, every model we tested fails, and its failures are easy to find, clustered, and in part interpretable. Hypotheses are found by searching for failures deserves belief only after it predicts failures on fresh inputs under a test that accounts for how it was found; 19 of 107 candidates pass, and a candidate confirmed by a designed experiment identifies a rule that reaches frontier models. For tasks with exact answers, how often a model fails, how its failures cluster, and the validated hypotheses that identify them describe the model better than a single score.

AI USE STATEMENT

Large language models were used to assist in drafting and editing this manuscript's text, and to assist in writing portions of the experiment harness and analysis scripts. We did not use generative AI tools to generate, fabricate, or select experimental results, to write related-work citations without verifying them against the primary source, or to generate the failure examples in Appendix L, which are taken verbatim from recorded model outputs. All AI-assisted code and text were reviewed by the authors, who take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Appendix B gives every task generator, prompt template, difficulty level, feature set, model identifier and decoding setting; Appendix D.6 gives the estimator, with a worked example, for every reported quantity; and Appendix D gives the exact sequential test used to grow regions. Every model call is stored as an (input, prompt, output, parsed answer, correct answer, verdict) record, so every statistic can be recomputed from raw data. The anonymized supplementary material contains the generators, oracles, probes, search, region and validation code, the per-input logs of every exhaustive enumeration, the complete region catalogues with each region's rule, fresh-sample rate and e-value, the scripts that produce every table and figure, and the compute used by each experiment.

REFERENCES

Andrea Arcuri and Lionel Briand. Adaptive random testing: an illusion of effectiveness? In *Proceedings of the 2011 International Symposium on Software Testing and Analysis*, ISSTA '11, pp. 265–275, New York, NY, USA, 2011. ACM. doi: 10.1145/2001420.2001452.

Susan Athey and Guido Imbens. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences*, 113(27):7353–7360, 2016. doi: 10.1073/pnas.1510489113.

Tanja Baeumel, Josef van Genabith, and Simon Ostermann. The lookahead limitation: Why multi-operand addition is hard for LLMs. *arXiv preprint arXiv:2502.19981*, 2025.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300, 1995.

Gabrielle Berrada, Jannik Kossen, Freddie Bickford Smith, Muhammed Razzak, Yarin Gal, and Tom Rainforth. Scaling up active testing to large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Alex Bie, Travis Dick, Alex Kulesza, Prabhakar Raghavan, Vinod Raman, and Sergei Vassilvitskii. AI-rithmetic. *arXiv preprint arXiv:2602.10416*, 2026.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023.

Muxi Chen, Chenchen Zhao, and Qiang Xu. HiBug2: Efficient and interpretable error slice discovery for comprehensive model debugging. *arXiv preprint arXiv:2501.16751*, 2025.

T. Y. Chen, H. Leung, and I. K. Mak. Adaptive random testing. In Michael J. Maher (ed.), *Advances in Computer Science – ASIAN 2004. Higher-Level Decision Making*, volume 3321 of *Lecture Notes in Computer Science*, pp. 320–329, Berlin, Heidelberg, 2004. Springer. doi: 10.1007/978-3-540-30502-6_23.

Yeounoh Chung, Tim Kraska, Neoklis Polyzotis, Ki Hyun Tae, and Steven Euijong Whang. Slice Finder: Automated data slicing for model validation. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pp. 1550–1553. IEEE, 2019. doi: 10.1109/ICDE.2019.00139.

C. J. Clopper and E. S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413, 1934.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349(6248):636–638, 2015.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Sean Welleck, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. Faith and fate: Limits of transformers on compositionality. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Lukas Edman, Helmut Schmid, and Alexander Fraser. CUTE: Measuring LLMs' understanding of their tokens. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3017–3026, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.177.

Sabri Eyuboglu, Maya Varma, Khaled Saab, Jean-Benoit Delbrouck, Christopher Lee-Messer, Jared Dunnmon, James Zou, and Christopher Ré. Domino: Discovering systematic errors with cross-modal embeddings. In *International Conference on Learning Representations (ICLR)*, 2022.

Rahul Gopinath, Alexander Kampmann, Nikolas Havrikov, Ezekiel O. Soremekun, and Andreas Zeller. Abstracting failure-inducing inputs. In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA '20, pp. 237–248. ACM, 2020. doi: 10.1145/3395363.3397349.

Steven R. Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 49(2):1055–1080, 2021. doi: 10.1214/20-AOS1991.

Yichen Huang and Lin F. Yang. Winning gold at IMO 2025 with a model-agnostic verification-and-refinement pipeline. *arXiv preprint arXiv:2507.15855*, 2025. URL https://arxiv.org/abs/2507.15855.

Yue Huang, Zhengzhe Jiang, Yuchen Ma, Yu Jiang, Xiangqi Wang, Yujun Zhou, Yuexing Hao, Kehan Guo, Pin-Yu Chen, Stefan Feuerriegel, and Xiangliang Zhang. ProbeLLM: Automating principled diagnosis of LLM failures. *arXiv preprint arXiv:2602.12966*, 2026.

Saachi Jain, Hannah Lawrence, Ankur Moitra, and Aleksander Madry. Distilling model failures as directions in latent space. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2206.14754.

Alexander Kampmann, Nikolas Havrikov, Ezekiel O. Soremekun, and Andreas Zeller. When does my program do this? Learning circumstances of software behavior. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE '20, pp. 1228–1239. ACM, 2020. doi: 10.1145/3368089.3409687.

Yael Konforti, Mateo Espinosa Zarlenga, Elaf Almahmoud, and Mateja Jamnik. CB-SLICE: Concept-based interpretable error slice discovery. *arXiv preprint arXiv:2605.29836*, 2026.

Jannik Kossen, Sebastian Farquhar, Yarin Gal, and Tom Rainforth. Active testing: Sample-efficient model evaluation. In *International Conference on Machine Learning (ICML)*, 2021.

Jorrit Kruthoff. Carrying over algorithm in transformers. *arXiv preprint arXiv:2401.07993*, 2024.

Xiang Lisa Li, Farzaan Kaiyom, Evan Zheran Liu, Yifan Mai, Percy Liang, and Tatsunori Hashimoto. AutoBencher: Towards declarative benchmark construction. *arXiv preprint arXiv:2407.08351*, 2024.

R. Thomas McCoy, Shunyu Yao, Dan Friedman, Mathew D. Hardy, and Thomas L. Griffiths. Embers of autoregression show how large language models are shaped by the problem they are trained to solve. *Proceedings of the National Academy of Sciences*, 121(41):e2322420121, 2024. doi: 10.1073/pnas.2322420121.

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

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

Marianna Nezhurina, Lucia Cipolina-Kun, Mehdi Cherti, and Jenia Jitsev. Alice in Wonderland: Simple tasks showing complete reasoning breakdown in state-of-the-art large language models. *arXiv preprint arXiv:2406.02061*, 2024.

Yaniv Nikankin, Anja Reusch, Aaron Mueller, and Yonatan Belinkov. Arithmetic without algorithms: Language models solve math with a bag of heuristics. In *International Conference on Learning Representations (ICLR)*, 2025.

Philip Quirke and Fazl Barez. Understanding addition in transformers. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2310.13121.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11491.

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

Nicholas Saban. FailureScope: Cross-regime behavioral diagnosis of language model weaknesses. *arXiv preprint arXiv:2606.09878*, 2026.

Mikayel Samvelyan, Sharath Chandra Raparthy, Andrei Lupu, Eric Hambro, Aram H. Markosyan, Manish Bhatt, Yuning Mao, Minqi Jiang, Jack Parker-Holder, Jakob Foerster, Tim Rocktäschel, and Roberta Raileanu. Rainbow teaming: Open-ended generation of diverse adversarial prompts. *arXiv preprint arXiv:2402.16822*, 2024.

Ryoma Sato. Even GPT-5.2 can't count to five: The case for zero-error horizons in trustworthy LLMs. *arXiv preprint arXiv:2601.15714*, 2026.

Aaditya K. Singh and DJ Strouse. Tokenization counts: the impact of tokenization on arithmetic in frontier LLMs. *arXiv preprint arXiv:2402.14903*, 2024.

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 *International Conference on Learning Representations (ICLR)*, pp. 10131–10165, 2025.

Gaurav Srivastava, Aafiya Hussain, Zhenyu Bi, Swastik Roy, Priya Pitre, Meng Lu, Morteza Ziyadi, and Xuan Wang. BeyondBench: Contamination-resistant evaluation of reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2509.24210.

András Vargha and Harold D. Delaney. A critique and improvement of the CL common language effect size statistics of McGraw and Wong. *Journal of Educational and Behavioral Statistics*, 25 (2):101–132, 2000.

Abraham Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16 (2):117–186, 1945.

Ruodu Wang and Aaditya Ramdas. False discovery rate control with e-values. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(3):822–852, 2022.

Yangzhen Wu, Aaron J. Li, Wenjie Ma, Li Cao, Ziheng Zhou, Mert Cemri, Shu Liu, Yuran Xiu, Chenxiao Yan, Haikun Zhao, Bin Yu, Ion Stoica, and Dawn Song. BenchEvolver: Frontier task synthesis via solution-centric evolution. *arXiv preprint arXiv:2606.01286*, 2026.

Yang Yan, Yu Lu, Renjun Xu, and Zhenzhong Lan. Do large language models truly grasp addition? a rule-focused diagnostic using two-integer arithmetic. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

Haotong Yang, Yi Hu, Shijia Kang, Zhouchen Lin, and Muhan Zhang. Number Cookbook: Number understanding of language models and how to improve it. In *International Conference on Learning Representations (ICLR)*, pp. 3946–3984, 2025.

Andreas Zeller and Ralf Hildebrandt. Simplifying and isolating failure-inducing input. *IEEE Transactions on Software Engineering*, 28(2):183–200, 2002.

Lexin Zhou, Wout Schellaert, Fernando Martínez-Plumed, Yael Moros-Daval, Cèsar Ferri, and José Hernández-Orallo. Larger and more instructable language models become less reliable. *Nature*, 634(8032):61–68, 2024. doi: 10.1038/s41586-024-07930-y.

Siyu Zhou, Patrick Vossler, Venkatesh Sivaraman, Yifan Mai, and Jean Feng. Adaptive auditing of AI systems with anytime-valid guarantees. *arXiv preprint arXiv:2605.07002*, 2026.

Kaijie Zhu, Jiaao Chen, Jindong Wang, Neil Zhenqiang Gong, Diyi Yang, and Xing Xie. Dyval: Dynamic evaluation of large language models for reasoning tasks. In *International Conference on Learning Representations (ICLR)*, 2024.

Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

APPENDIX OVERVIEW

Notation used in the appendix. A task C is the tuple $(\mathcal{X}_C, f_C, d, \nu)$ of valid inputs, oracle, neighbour relation (which makes $\mathcal{X}_C$ a graph) and sampling distribution, uniform within each difficulty level. τ_{eff} is the acceptance threshold τ of Section **??**, $\max(0.5, \min(0.95, \pi + 0.15))$, also called the acceptance floor. A region's *member rate* is its failure rate on the inputs that grew it and its *fresh rate* the rate on new inputs satisfying its rule; the *generalization gap* is member rate minus fresh rate. The *cascade length* k is the number of consecutive digit columns summing to nine through which a carry propagates (`max_carry_chain_9` in code). CP is a Clopper–Pearson interval and CS an anytime-valid confidence sequence; m is a number of fresh inputs and n a number of answers. `RAND`, `HC`, `ME` and `LLM-REF` are uniform sampling, hill climbing, MAP-Elites and proposals by the target model. AUC is the area under the ROC curve and SE a standard error.

Appendix	Contents
A	Method details: search objectives and algorithms, cluster growth, hypothesis fitting and validation, designed contrasts, frontier reasoning settings.
B	Tasks, models, full prevalence and clustering results for open-weight and frontier models, and the failure-rate scan used to choose search cells.
C	Validation of the measurement and region pipeline against known structure.
D	Statistical tools, the exact sequential test, e-BH, search-algorithm details and estimator formulas.
E	The search comparison in full, and the rare-failure regime in simulation.
F	The region catalogue, the validation decomposition, false-discovery rates on known truth, other model families, and a relative acceptance rule.
G	Carry propagation against carry count, the control for the digit nine, and the carry contrast on frontier models.
H	Reasoning settings and character positions on frontier models.
I	The fine-tuning experiment and its two control arms.
J	The operand-order effect across model sizes.
??	Certification of low-failure cells.
K	Evaluations to first failure.
L	Recorded failure examples.
M	Comparison with prior method families.

A METHOD DETAILS

This appendix gives the implementation details of the method summarized in Section 3.

A.1 CAPABILITY SPACES

For addition, the generator draws the width of each operand from the difficulty level and then each operand uniformly among the integers of that width; a fixed template renders the pair as a prompt. The reference distribution ν is the generator's distribution, which draws each width uniformly from the level and each value uniformly given its width. The other capabilities are generated analogously (Table 4). A model's *reliability gap* is the set of inputs with $p(x) > 0$.

A.2 FAILURE SEARCH

Objectives. A search method chooses inputs $x_1, x_2, \cdots \in \mathcal{X}_C$, where x_t may depend on the model's answers to $x_1, \ldots, x_{t-1}$, and scores each input with a per-input failure objective $L : \mathcal{X}_C \to [0, 1]$. The binary objective $L_{\text{bin}}(x) = \mathbb{1}[M(x) \neq f_C(x)]$ uses one call per input. The probabilistic objective $L_{\text{prob}}(x) = \hat{p}(x)$ is the fraction of wrong answers among up to eight answers to x, drawn in stages of two, two and four: sampling stops after two answers if both are wrong, after four if all four are right, and otherwise continues to eight.

Found failures, coverage and first failure. Let T_B be the number of queried inputs whose evaluation completes before the B calls are spent. A queried input counts as a *found failure* if $L(x_t) \geq \frac{1}{2}$: under L_{bin} any wrong answer is a found failure, while under L_{prob} at least half of the sampled answers must be wrong. This threshold is fixed across cells so that search methods are compared on the same footing; the threshold τ used for cluster growth and hypothesis validation is separate (Section 3). The grid $b : \mathcal{X}_C \to \mathcal{B}$ has two to four small-integer axes per capability, such as the larger operand width and the number of carries for addition (Table 14); two failures in the same bin count as the same kind of failure. A run's coverage is

$$\text{Cov} = \left| \left\{ b(x_t) : t \leq T_B,\ L(x_t) \geq \tfrac{1}{2} \right\} \right|,$$

and the number of queried inputs to the first found failure is $\text{IF} = \min\{t : L(x_t) \geq \frac{1}{2}\}$, counted in queried inputs rather than model calls because one input may require two to eight calls under L_{prob}. Every model call counts against B, including repeated answers under L_{prob} and calls used to generate proposals.

Search algorithms. All four methods run at $B = 2{,}000$.

- RAND draws inputs independently from ν.

- `HC` (hill climbing) starts from an input drawn from ν, evaluates eight neighbours sampled uniformly from its one-edit neighbourhood, and moves to the best of them if its L is strictly higher than the current input's. After 20 consecutive steps without improvement, it restarts from a new draw from ν.
- `ME` (MAP-Elites (Mouret & Clune, 2015)) is initialized with 32 draws from ν and keeps, in each bin of the grid b, the input with the highest L seen so far, replacing it only on strict improvement. Each step picks an occupied bin uniformly at random and evaluates one randomly chosen neighbour of its elite. Because b is also the grid on which coverage is counted, `ME` is scored on the same grid it maintains.
- `LLM-REF` (proposals by the target model) starts from 16 draws from ν. In each round, the target model is shown its ten inputs with the highest L so far and asked for eight new ones; the proposal request costs one model call. Proposals that cannot be parsed or fall outside the capability's range are discarded without querying the model. If no valid proposals remain, eight draws from ν are used instead.

A.3 Failure clusters

Clustering and clusters. We distinguish two related notions: *clustering*, a statistical property of a model's failures that we measure, and a *cluster*, a specific set of inputs that we construct. Formally, failures cluster if

$$\Pr[x' \text{ fails} \mid x \text{ fails}] \;>\; \Pr[x' \text{ fails} \mid x \text{ correct}], \qquad x \sim \nu,\ x' \text{ a random neighbour of } x;$$

we report the left-hand side both as a ratio to π (the *neighbour lift*) and as a ratio to the right-hand side. Clustering makes clusters more likely to exist; a cluster is one such set grown from a particular failure.

Measuring clustering. In each probed cell, we classify each of 300 inputs by its first answer, answer eight random neighbours of each twice, and compare the failure rate among neighbours of failing inputs with that among neighbours of correct inputs, using a percentile bootstrap over inputs.

Growing a cluster. Each run takes as seeds up to eight found failures, selecting the one with the highest L in each coverage bin and skipping seeds that already lie in a grown cluster or satisfy a validated hypothesis. From each seed, we repeatedly take a random input from a queue of candidates, initially twelve sampled neighbours of the seed, and decide whether to admit it using a truncated sequential probability ratio test (Wald, 1945) of $H_0 : p(x) < \tau$ against $H_1 : p(x) \geq \tau$, with error rates 0.05 and at most 16 answers. After the maximum number of answers, the sign of the log-likelihood ratio determines the decision (exact form and operating characteristics in Appendix D). Each admitted input adds twelve of its sampled neighbours to the queue. Growth stops when the queue is empty or after 400 tested inputs. The test is a construction step rather than a final inferential test: it may admit inputs whose true failure probability lies below τ, and nothing about a cluster is reported until its hypothesis is validated on fresh inputs.

A.4 Hypotheses and validation

Fitting a hypothesis. Each grown cluster yields one hypothesis. We fit three kinds of description, single-feature thresholds, an axis-aligned box, and a depth-3 decision tree, to separate the cluster from the neighbours rejected during growth and from 300 additional inputs drawn from ν. We retain the description whose covered inputs have the highest estimated failure rate. We then minimise the description (Zeller & Hildebrandt, 2002): a condition is dropped if the inputs admitted by removing it still fail at rate at least τ, or if essentially no input can violate it while satisfying the remaining conditions. A final pass removes any condition not needed to separate the cluster from its negative examples.

The threshold. The floor of 0.5 in $\tau = \max(0.5, \min(0.95, \hat{\pi}+0.15))$ requires covered inputs to fail more often than not. The margin of 0.15 requires a hypothesis to exceed the estimated background rate by a fixed absolute amount, and the cap of 0.95 keeps the threshold attainable in cells with high background failure. The same τ is used for cluster growth, minimisation and validation. Because τ is absolute, a cell with a low background rate must clear a larger lift; Appendix F.4 reports the validated set under a relative threshold.

Validation. Let $h_1, \ldots, h_K$ be all hypotheses, fixed before any validation input is drawn. For each h_k we draw $m = 300$ fresh inputs from ν conditioned on $h_k(x) = 1$, independently of all data used to search, grow and fit the hypothesis, answer each input four times, and summarize each input by its fraction of wrong answers, so that inputs rather than individual answers are the unit of evidence. For each hypothesis we test $H_{0,k}: p_{h_k} \leq \tau$. A hypothesis is validated if (1) its anytime-valid 95% lower confidence bound on p_{h_k} (Howard et al., 2021) exceeds τ, and (2) e-BH (Wang & Ramdas, 2022) rejects $H_{0,k}$ at false-discovery rate $q = 0.05$ across all K hypotheses, with e-values derived from the same confidence sequences. The family includes every grown hypothesis, not only those passing an initial screen. Hypotheses from clusters with fewer than ten members or with no conditions left after minimisation are not eligible for validation but remain in the tested family.

Robustness to the prompt. Each validated hypothesis is also evaluated on 100 of its fresh inputs under two rewordings of the prompt and, for addition, with the operands swapped, to distinguish failures of the capability from prompt-specific artefacts.

A.5 Rules and designed contrasts

The arms of a designed contrast are either drawn separately with the matched properties held fixed, or constructed by an edit that changes only the property under test, such as swapping the operands. When one contrast is run at several levels of the property, such as several cascade lengths, we correct across those tests with Holm's method. On a complete enumeration the failure rates are exact and no statistical test is needed. Each contrast is specified, including its arms, levels and statistical test, before any of its own outcome data are observed. Because it is pre-specified and uses new data, its test does not require correction for the K hypotheses proposed upstream; contrasts selected after their own outcomes are observed, or several contrasts testing the same rule, would require their own correction.

A.6 Frontier reasoning settings and decoding

The *low* setting is reasoning off for Claude Sonnet 5 and DeepSeek v4.1 Flash, minimal for Gemini 3.8 Flash and GLM-5.3, which reason at every setting, and low for GPT-5.6 Luna; the *high* setting is each API's high effort, and each model's *lowest* setting is its low one, except for Luna, whose lowest is no reasoning. Frontier cells use the provider's default temperature. GPT-5.6 Luna's API accepts only temperature 1, so its two-digit enumeration uses one sampled answer per problem.

B Tasks, models, and full prevalence and clustering results

B.1 Tasks

All six tasks share one interface (`sample`, `oracle`, `neighbors`, `features`, `render`, `parse`). Table 4 gives each generator, prompt and neighbour relation.

B.2 Models and decoding

Open-weight models are the instruction-tuned checkpoints of Qwen2.5 (0.5B, 1.5B and 3B in fp16; 7B and 14B as the official 4-bit AWQ releases), Granite-3.1 (2B in fp16; 8B as a 4-bit AWQ quantization), Phi-3.5-mini, Phi-4-mini, Mistral-7B-Instruct-v0.2 (AWQ), Falcon3-3B and SmolLM2-1.7B, served with vLLM on a single T4 GPU through each model's chat template. They answer directly with a per-task answer cap (16 tokens for scalar answers, 200 for lists, 96 for strings). Mistral-7B and SmolLM2 use constrained decoding to the answer format; on Qwen2.5-3B, constrained decoding changes the failure rate by at most 0.004 on four of five addition levels and by 0.036 on the easiest (0.092 to 0.128; Table 11). Sampled measurements use temperature 1 and enumerations temperature 0. Frontier models are queried through their providers' APIs at the settings of Section 3. Each frontier cell holds 600 answers (300 inputs, each answered twice) at the provider's default temperature. GPT-5.6 Luna's API accepts only temperature 1, so its exhaustive two-digit counts (all 8,100 additions and all 8,100 multiplications answered correctly) use one sampled answer per problem. GPT-5.6 Luna was queried through OpenRouter for its cluster runs and through the OpenAI API for the carry contrast and the neighbourhood probe of Table 9. Parsing takes the

Task	Generator	Prompt (abbreviated)	Neighbour relation
ADD	operands A, B with n_A, n_B digits drawn from the level	`Compute A + B. Answer with only the integer.`	change one digit; append or drop a digit
MUL	as ADD	`Compute A * B. Answer with only the integer.`	as ADD
SORT	list of 3–40 integers in $[-10^6, 10^6]$ with duplicates and near-ties; operation sort, median, k-th smallest or deduplicate-and-sort	`Sort these integers in ascending order: ...`	change one value by ± 1; swap two values; drop or append a value
STRING	string of 4–40 characters from one of six character sets; operation reverse, upper, lower, count a character, count words or substring	`Reverse the string "...".`	substitute, insert or delete one character
DATE	date in 1900–2100 plus an offset in days, weeks, months or years	`What is the date N days after YYYY-MM-DD? Answer with only the date in YYYY-MM-DD format.`	shift the date by a day, month or year; change the offset by one; change the unit
GRAPH	random DAG on 5–25 nodes at one of three densities; reachability or shortest-path query	`A directed graph has nodes 0 to n-1 and edges ... Is node t reachable from node s?` or `What is the length of the shortest directed path ...?`	remove or add one edge; resample at a different node count

Table 4: The six tasks. Every generated input has a correct answer computed by the oracle; levels restrict the generator (operand width, list or string length, offset size, node count) and the neighbour relation is restricted to the same level.

last match of the task's answer pattern; an unparseable answer is a failure and is also counted separately. In 16 of the 216 probed cells more than 10% of answers are unparseable, mostly date answers from Falcon3-3B, Phi-3.5-mini, Phi-4-mini, Granite-3.1-8B and Qwen2.5-0.5B, and Falcon3-3B's addition answers at wider operands and its graph answers; as everywhere, these count as failures.

Features. Hypotheses are written over interpretable features computed from each input. Addition has 23, including the digit length of each operand, their signed length difference, the number of carries, the longest run of consecutive carries, the longest carry cascade (`max_carry_chain_9`), the leading digits, the counts of nines and zeros, and tokenizer features (token counts and digit-chunking offsets); some are distractors with no expected effect. Multiplication uses the same operand features and the width of the product; graph queries use the query type, the number of nodes, the true path length and the in-degree of the target; the other tasks use analogous features of their inputs.

Frontier model identifiers. [Exact API model identifiers, provider endpoints and query dates for GPT-5.6 Luna, Gemini 3.8 Flash, Claude Sonnet 5, GLM-5.3 and DeepSeek v4.1 Flash: to be supplied.]

Size and precision. Both model ladders vary weight precision along with size: Qwen2.5 is fp16 at 0.5B, 1.5B and 3B and 4-bit AWQ at 7B and 14B, and Granite-3.1 is fp16 at 2B and AWQ at 8B. At widths 3–4 every step within one precision lowers the failure rate by $2.31\times$ to $2.87\times$, and the two steps that cross into AWQ lower it by $1.20\times$ and $1.14\times$. Trends across size in this paper are therefore trends in size and precision together.

B.3 Prevalence

B.4 Clustering

	+ 3–4	+ 5–6	+ 7–8	+ 9–10	+ 11–12	× 2	× 3	× 4–5
Qwen2.5-0.5B	0.48	0.81	0.94	0.99	1.00	0.59	0.98	1.00
Qwen2.5-1.5B	0.23	0.49	0.63	0.77	0.94	0.37	0.98	1.00
Qwen2.5-3B	0.13	0.21	0.46	0.57	0.67	0.21	0.95	1.00
Qwen2.5-7B	0.07	0.17	0.31	0.46	0.70	0.12	0.92	1.00
Qwen2.5-14B	0.07	0.06	0.16	0.30	0.49	0.05	0.84	0.99
Granite-3.1-2B	0.34	0.46	0.56	0.83	0.95	0.49	0.96	1.00
Granite-3.1-8B	0.32	0.53	0.67	0.81	0.91	0.29	0.96	1.00
Phi-4-mini	0.83	0.69	0.86	0.94	0.95	0.35	0.85	1.00
Phi-3.5-mini	0.93	0.97	0.99	0.99	1.00	0.44	1.00	1.00
Mistral-7B	0.72	0.75	0.91	0.98	0.99	0.80	0.99	1.00
Falcon3-3B	1.00	1.00	1.00	1.00	1.00	0.97	1.00	1.00
SmolLM2-1.7B	0.94	0.97	1.00	1.00	1.00	0.92	1.00	1.00

Table 5: Arithmetic. Failure rate π per level (300 uniformly drawn inputs, two answers each, temperature 1). Arithmetic levels give the number of digits of each operand. An unparseable answer counts as a failure.

	sort 3–5	6–10	3–40	str 4–8	9–16	4–40	date small	all	graph 5–10	5–25
Qwen2.5-0.5B	0.88	0.97	0.98	0.81	0.90	0.89	0.98	0.97	0.79	0.76
Qwen2.5-1.5B	0.57	0.74	0.90	0.61	0.80	0.82	0.83	0.93	0.64	0.67
Qwen2.5-3B	0.40	0.69	0.86	0.67	0.81	0.84	0.79	0.94	0.20	0.44
Qwen2.5-7B	0.41	0.62	0.85	0.60	0.72	0.75	0.42	0.75	0.23	0.38
Qwen2.5-14B	0.25	0.45	0.73	0.41	0.54	0.53	0.43	0.86	0.22	0.40
Granite-3.1-2B	0.57	0.77	0.90	0.69	0.81	0.85	0.64	0.91	0.42	0.57
Granite-3.1-8B	0.55	0.76	0.90	0.67	0.83	0.87	0.70	0.94	0.44	0.53
Phi-4-mini	0.47	0.64	0.85	0.63	0.81	0.81	0.91	0.98	0.56	0.63
Phi-3.5-mini	0.58	0.71	0.86	0.56	0.81	0.81	0.80	0.99	0.56	0.63
Mistral-7B	0.90	0.97	0.97	0.80	0.89	0.93	0.81	0.96	0.65	0.70
Falcon3-3B	0.63	0.84	0.91	0.90	0.92	0.93	0.98	1.00	0.64	0.73
SmolLM2-1.7B	0.91	0.98	0.97	0.84	0.94	0.92	0.94	0.98	0.69	0.67

Table 6: Sorting, string, date and graph tasks. Sort levels are list lengths (operations: sort, median, k-th smallest, deduplicate-and-sort); string levels are string lengths (reverse, change case, count a character, count words, substring); the small date level limits the offset to 30 days, 8 weeks, 12 months or 5 years; graph levels are node counts. Failure rate π per level (300 uniformly drawn inputs, two answers each, temperature 1). An unparseable answer counts as a failure.

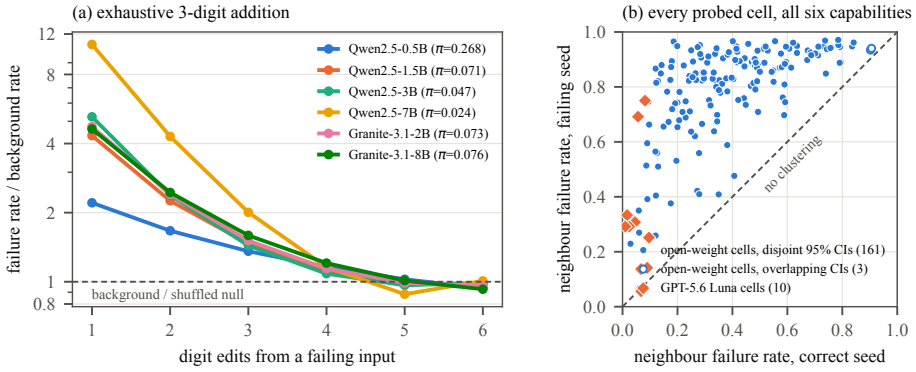

Figure 5: Clustering in full. (a) Complete enumeration of all 810,000 three-digit additions for six models: failure rate at exactly d digit edits from a failing input, divided by the model's background rate; a shuffled failure set with the same failure rate lies on the dashed line at every d. (b) Every probed (model, task, level) cell: failure rate among the eight neighbours of failing inputs against that among the neighbours of correct inputs; the dashed diagonal is no clustering. Filled points have disjoint 95% bootstrap intervals (resampling inputs); hollow points are the three open-weight cells whose intervals overlap; diamonds are GPT-5.6 Luna's ten probed cells (Table 9).

Task	cells probed	disjoint 95% CIs	median ratio	range of ratio
addition	43	41	2.12	1.04–7.87
multiplication	15	14	1.52	0.97–2.15
sort / median / rank	30	30	2.40	1.29–7.88
string ops	36	36	2.04	1.21–6.63
date arithmetic	16	16	1.76	1.24–3.73
graph queries	24	24	2.70	1.18–6.84

Table 7: Neighbourhood probe by task. For each cell with $0.005 \leq \pi \leq 0.95$, eight random neighbours of each of the 300 inputs are answered twice; the ratio is the neighbour failure rate around inputs whose first answer failed over that around inputs whose first answer was correct. Intervals are percentile bootstrap over inputs (1,000 resamples). The three cells without disjoint intervals are two addition cells in which the model fails 93% to 94% of inputs and Qwen2.5-14B on two-digit multiplication.

	2-digit +		2-digit ×		3-digit +: lift at d edits from a failure			
Model	errors	ratio	errors	ratio	d=1	2	3	4
Qwen2.5-0.5B	1,247	3.7	4,744	1.7	2.21	1.67	1.36	1.16
Qwen2.5-1.5B	150	5.2	2,958	2.0	4.33	2.25	1.46	1.12
Qwen2.5-3B	188	5.5	1,652	2.2	5.23	2.38	1.44	1.08
Qwen2.5-7B	10	9.6	1,024	2.3	10.81	4.30	2.01	1.16
Qwen2.5-14B	10	4.8	501	1.9				
Granite-3.1-2B	470	4.5	3,807	1.8	4.74	2.41	1.51	1.14
Granite-3.1-8B	334	4.8	2,078	1.7	4.63	2.45	1.59	1.21
Phi-4-mini	1,303	2.4	973	1.8				
Phi-3.5-mini	3,716	1.7	2,353	2.6				
Mistral-7B	3,969	1.7	6,514	1.5				
Falcon3-3B	7,682	1.4	8,020	1.3				
SmolLM2-1.7B	7,084	1.4	7,222	1.4				

Table 8: Exhaustive enumerations at temperature 0. Errors are out of 8,100 two-digit problems. *Ratio* is the failure rate one digit substitution from a failing input over that one substitution from a correct input. Three-digit addition (810,000 problems) was enumerated for six models; *lift* is the failure rate at exactly d substitutions from a failing input over the model's background rate. A shuffled failure set of equal failure rate has lift within 0.01 of 1 at every d for every model. Measured against the rate around correct sums instead of the background, the one-edit ratio for three-digit sums runs from 4.0 (Qwen2.5-0.5B) to 14.2 (Qwen2.5-7B), and the neighbours of a correct sum fail at 0.56 to 0.79 times the background rate.

Task	level	π	fails again	nbr. of failing	nbr. of correct	unanswered
addition	3–4	0.000	–	–	–	0
addition	31–34	0.077	0.46	**0.308**	0.045	0
addition	41–45	0.095	0.33	0.252	0.096	0
multiplication	2	0.000	–	–	–	0
multiplication	10	0.110	0.20	0.056	0.068	0
multiplication	11	0.062	0.00	0.142	0.090	0
sort / median / rank	3–5	0.003	0.00	–	–	0
sort / median / rank	3–40	0.030	0.67	**0.292**	0.024	0
string ops	4–8	0.168	0.81	**0.750**	0.082	0
string ops	4–40	0.135	0.67	**0.692**	0.057	0
date arithmetic	small	0.003	0.00	–	–	0
date arithmetic	all	0.087	0.10	0.137	0.068	0
graph queries	5–10	0.013	0.67	0.292	0.010	0
graph queries	5–25	0.028	0.50	**0.333**	0.017	0

Table 9: Neighbourhood probe of GPT-5.6 Luna at the low reasoning setting (reasoning effort `low`): 200 inputs per level, two answers each, and four neighbours per input, each answered twice. This probe is a separate sample from Table 10, so its failure rates π differ from that table's within sampling error. *Fails again* is the fraction of inputs that failed on the first answer and failed again on the second; the two neighbour columns are the failure rates among neighbours of failing and of correct inputs, in bold where their 95% intervals are disjoint (five of the ten probed cells). An empty answer is asked once more with doubled reasoning headroom; *unanswered* counts answers still empty (none in this probe).

B.5 FRONTIER MODELS AT THE LOW REASONING SETTING

Table 10 gives every frontier cell of Figure 3b with its interval, computed over the 300 inputs of the cell.

Level	GPT-5.6 Luna	Gemini 3.8 Flash	Claude Sonnet 5	GLM-5.3	DeepSeek v4.1 Flash
add, 3–4 digits	0.0 [0.0, 1.3]	0.0 [0.0, 1.3]	0.0 [0.0, 1.3]	1.0 [0.3, 2.9]	0.0 [0.0, 1.3]
add, 31–34 digits	7.7 [5.2, 11.2]	24.0 [19.5, 29.1]	15.0 [11.4, 19.5]	36.3 [31.1, 41.9]	52.0 [46.4, 57.6]
multiply, 2×2 digits	0.0 [0.0, 1.3]	0.0 [0.0, 1.3]	0.0 [0.0, 1.3]	0.0 [0.0, 1.3]	0.0 [0.0, 1.3]
multiply, 10×10 digits	8.5 [5.8, 12.2]	8.0 [5.4, 11.6]	32.0 [27.0, 37.5]	18.2 [14.2, 22.9]	98.0 [95.7, 99.1]
sort, 3–5 integers	0.7 [0.2, 2.4]	0.0 [0.0, 1.3]	3.0 [1.6, 5.6]	2.0 [0.9, 4.3]	11.0 [7.9, 15.0]
sort, 3–40 integers	4.3 [2.5, 7.3]	1.0 [0.3, 2.9]	17.0 [13.2, 21.7]	10.0 [7.1, 13.9]	14.0 [10.5, 18.4]
string, 4–8 characters	15.8 [12.1, 20.4]	3.0 [1.6, 5.6]	21.0 [16.8, 26.0]	16.0 [12.3, 20.6]	25.0 [20.4, 30.2]
string, 4–40 characters	13.5 [10.1, 17.8]	11.0 [7.9, 15.0]	18.0 [14.1, 22.7]	14.0 [10.5, 18.4]	32.0 [27.0, 37.5]
date, small offset	0.3 [0.1, 1.9]	0.0 [0.0, 1.3]	0.0 [0.0, 1.3]	1.0 [0.3, 2.9]	1.0 [0.3, 2.9]
date, full range	8.5 [5.8, 12.2]	1.0 [0.3, 2.9]	8.0 [5.4, 11.6]	12.3 [9.1, 16.5]	30.0 [25.1, 35.4]
graph, 5–10 nodes	0.8 [0.3, 2.7]	0.0 [0.0, 1.3]	0.0 [0.0, 1.3]	1.0 [0.3, 2.9]	4.0 [2.3, 6.9]
graph, 5–25 nodes	3.0 [1.6, 5.6]	0.0 [0.0, 1.3]	1.0 [0.3, 2.9]	5.0 [3.1, 8.1]	10.0 [7.1, 13.9]

Table 10: Frontier models at the low reasoning setting: failure rate in percent over 600 answers per cell (300 inputs, each answered twice), with a 95% Wilson interval computed over the 300 inputs, which stays valid however strongly an input's two answers agree. The low setting is reasoning off for Claude Sonnet 5 and DeepSeek v4.1 Flash, minimal for Gemini 3.8 Flash and GLM-5.3, and low for GPT-5.6 Luna.

B.6 FAILURE-RATE SCAN USED TO CHOOSE SEARCH CELLS

Search and cluster growth run in cells chosen by a failure-rate scan (Table 11). A cell is searchable when $0.05 \leq \pi \leq 0.5$: below that range too few failures are found at budget to grow a cluster, and above it failures are everywhere and no search method can be distinguished from uniform sampling. Multiplication fails on nearly every input from four digits up, and date, sorting and string tasks exceed this range at their full parameter range, so search runs on addition, two- to three-digit multiplication and graph queries. Two search cells on the 1.5B lie just above the band (0.507 and 0.623 in Table 18). The scan and the probe of Tables 5–6 are separate samples; on addition their rates differ by at most 0.07, the largest relative difference being Qwen2.5-14B at widths 3–4 (0.027 in the scan, 0.072 in the probe).

Model	Answer mode	π by width of wider operand 3–4	5–6	7–8	9–10	11–12
Qwen2.5 ladder, direct answer						
0.5B	direct	0.517	0.762	0.952	0.987	1.000
1.5B	direct	**0.223**	**0.463**	0.630	0.735	0.923
3B	direct	**0.092**	**0.282**	**0.458**	0.530	0.710
7B	direct	**0.077**	**0.217**	**0.317**	**0.442**	0.672
14B	direct	0.027	**0.090**	**0.130**	**0.295**	**0.417**
constrained-decoding control						
3B, guided	direct	**0.128**	**0.282**	**0.455**	0.527	0.710
other families, direct answer						
Granite-3.1-2B	direct	**0.345**	**0.463**	0.615	0.778	0.940
Granite-3.1-8B, AWQ	direct	**0.302**	0.503	0.637	0.772	0.922
Phi-4-mini	direct	0.833	0.667	0.883	0.953	0.940
Falcon3-3B	direct	0.990	1.000	1.000	1.000	1.000
Mistral-7B, guided	direct	0.677	0.682	0.902	0.977	0.997
Phi-3.5-mini	direct	0.933	0.970	0.995	0.992	0.998
Phi-3.5-mini, guided	direct	0.863	0.927	0.973	0.990	1.000
SmolLM2-1.7B, guided	direct	0.937	0.978	0.998	1.000	1.000
other families, reasoning permitted						
Mistral-7B	reasoning	0.843	0.912	0.943	0.968	1.000
Phi-3.5-mini	reasoning	**0.380**	0.618	0.777	0.912	0.963

Table 11: Failure-rate scan on addition by operand-width level, 300 inputs at two answers each. Bold marks a searchable cell ($0.05 \leq \pi \leq 0.5$). *Guided* rows enforce the answer format by constrained decoding; *reasoning* rows permit a reasoning trace before the answer.

C VALIDATING THE PIPELINE AGAINST PLANTED STRUCTURE

Every measurement in this paper passes through the same code, so we check the cluster and hypothesis pipeline against failure conditions planted by construction.

Planted conditions on simulated models. Four simulated models fail stochastically inside a planted condition (p_{in}=0.85, p_{out}=0.015), and the full pipeline (growth, hypothesis fitting and fresh-sample validation) is run blind against the 23-feature addition map, including distractor features. AUC is computed for the fitted hypothesis as a binary classifier against the planted condition on 1,000 fresh inputs, so AUC $= 1.00$ means exact recovery, not a favourable ranking. The criterion is at least 3 of 4 simulated models at AUC > 0.9 (Table 12).

Simulated model (prevalence)	Hypothesis returned	AUC	Calls to first failure
O1: third digit of $A = 7$ (9.8%)	`third_digit_A >= 7 AND third_digit_A <= 7`	1.00	128
O2: `carry_count >= 4` (34%)	`n_cols_sum_ge_10 > 3.5`	0.896	32
O3: $\lvert A\rvert - \lvert B\rvert \geq 3$, B leads with 9 (3.1%)	`leading_digit_B > 8.5 AND len_diff_AB > 2.5`	1.00	—
O4: $A \in [10^6, 2 \times 10^6)$ (1.1%)	`n_digits_A >= 7 AND ... AND leading_digit_A <= 1`	1.00	1,928

Table 12: Recovery of planted conditions. Three of four are recovered exactly; O2 returns a near-miss proxy at AUC 0.896 (precision 1.00, recall 0.792). The criterion, at least 3 of 4 above AUC 0.9, is met.

The one partial recovery is informative. O2 plants `carry_count >= 4` and the pipeline returns `n_cols_sum_ge_10 > 3.5`, the count of columns that generate a carry *on their own*, without help from a carry arriving from below. That hypothesis has precision 1.00 and recall 0.792 against the planted condition: it captures independently generated carries and misses exactly those produced by propagation, the distinction that Section 4.3 tests with a designed contrast.

A carry-count condition at free operand widths. Hypotheses fitted to Qwen2.5-3B at free operand widths do not recover the carry-count condition of prior work (`carry_count >= 3 AND max_consec_carries >= 2`; best AUC 0.583 over five fitted hypotheses on 5,000 fresh inputs), because when width varies, length effects dominate the carry signal. The carry contrast of Section 4.3 therefore holds width fixed.

D Statistical and algorithmic implementation detail

D.1 The statistical tools

Every statistic in the pipeline acts on counts of failures among fresh inputs. The tools below are standard; we state what each one is for and how it is read.

Clopper–Pearson interval. The exact confidence interval for a binomial proportion (Clopper & Pearson, 1934). With k failures in n independent draws, the one-sided 95% lower bound, the form used for acceptance, is the smallest rate that would produce at least k failures with probability 5%. With zero failures in N draws the 95% upper bound is $1 - 0.05^{1/N}$, which is the zero-failure certificate of Appendix **??**: at $N = 1{,}000$ it is 0.0030 before and 0.0059 after a 19-way Bonferroni adjustment.

Sequential probability ratio test (SPRT). A test between two simple hypotheses, here "this neighbour fails at rate $\tau/2$" against "at rate τ", that draws one sample at a time and stops as soon as the accumulated likelihood ratio crosses an upper or lower boundary (Wald, 1945). It is used only to decide which neighbours join a region during growth; Appendix D gives the exact boundaries and the error rates of the truncated version.

Why a rule found by search needs a fresh test. A region reaches the acceptance test because a search found it and a threshold admitted it: it was selected for looking bad. A rate computed on the draws that selected the region is biased upward, which a fresh sample fixes. On a fixed fresh sample, an exact interval computed over inputs is valid for each rule, however the rule was chosen, and two issues remain. When many rules are tested, some pass by chance, which false-discovery control addresses. And if the number of fresh draws were chosen after looking at the data, a fixed-sample interval would lose its coverage, which an anytime-valid confidence sequence prevents.

e-value. A measure of evidence against a null hypothesis on a betting scale: the factor by which a wager against the null has multiplied its stake. Under the null the expected value is at most 1, so $e \geq 1/\alpha$ rejects at level α; $e = 20$ is strong evidence at $\alpha = 0.05$ and $e < 1$ means the data moved towards the null. Unlike a p-value, e-values can be multiplied across sequential draws and averaged across hypotheses without losing validity.

Anytime-valid confidence sequence. A sequence of intervals, one per sample size, such that the true rate lies in all of them simultaneously with probability at least 95% (Howard et al., 2021). It may be read at any sample size, including one chosen after seeing the data (Zhou et al., 2026). It is wider than a fixed-sample interval at the same n; with a fixed fresh sample this extra width is a conservative choice, and it removes three regions in Appendix F.1.

e-BH. The Benjamini–Hochberg false-discovery-rate procedure applied to e-values (Wang & Ramdas, 2022). Sort the K e-values in decreasing order and reject the k largest, where k is the largest index with $e_{(k)} \geq K/(\alpha k)$. It controls the expected fraction of false regions among those reported at α under arbitrary dependence, which matters because regions grown from shared draws are dependent. The family size K must be declared: it is the number of hypotheses tested, not the number that survived an earlier step (Appendix F.1).

Effect size and multiplicity for the search comparison. The Vargha–Delaney $\hat{A}_{12}$ (Vargha & Delaney, 2000) is the probability that a random seed of one algorithm reaches more failing bins

than a random seed of RAND; 0.5 is parity. Significance is by the Mann–Whitney test with Holm correction, a step-down procedure that keeps the family-wise error rate at α across all comparisons in a cell.

D.2 THE SEQUENTIAL TEST FOR REGION GROWTH

Region growth (Section 4.2) admits a d-neighbour x' into a region by a Wald sequential probability ratio test (Wald, 1945) of $H_0 : p(x') = p_0$ against $H_1 : p(x') = p_1$, with

$$p_0 = \max(\tau/2,\ 10^{-3}), \qquad p_1 = \min(\max(\tau,\ 10^{-3}),\ 0.999),$$

i.e. the null is half the acceptance threshold and the alternative is the threshold itself, both clipped away from 0 and 1. With Type-I/II error $\alpha = \beta = 0.05$ the log-likelihood-ratio boundaries are

$$A = \ln \frac{1-\beta}{\alpha} \approx 2.944, \qquad B = \ln \frac{\beta}{1-\alpha} \approx -2.944,$$

and at each new sample the running log-likelihood-ratio is compared against $[B, A]$: cross A and x' is admitted, cross B and it is rejected, otherwise sampling continues up to $k_{\max} = 16$, at which point the sign of the running log-likelihood-ratio breaks the tie. The acceptance threshold τ_{eff} itself is $\max(0.5, \min(0.95, \pi + 0.15))$ (Section 4.2, Table 17).

Operating characteristics under truncation. Truncation voids Wald's nominal guarantees. Enumerating the binomial tree exactly (no Monte Carlo) gives the truncated procedure's true error rates at the thresholds used here, and they are far from nominal (Table 13). Clear points do not resolve in 2–3 samples at the modal threshold. At $\tau = 0.5$, which is the modal threshold in our catalogues, the true Type-I error is 0.189 against a nominal 0.05 and 61.7% of points reach $k_{\max}$ rather than resolving early. The gap closes as τ rises: at $\tau = 0.95$ the procedure is near-nominal (0.033) and almost never truncates (0.8%), because $p_0 = \tau/2$ and $p_1 = \tau$ separate further as τ grows, while at $\tau = 0.5$ the two hypotheses are close and 16 draws are not enough to separate them.

τ	p_0	p_1	Type-I	Type-II	$P(\text{reach } k_{\max} \mid H_0)$	$E[n \mid H_0]$
0.50	0.250	0.500	0.189	0.106	0.617	13.9
0.53	0.265	0.530	0.105	0.161	0.520	13.5
0.57	0.285	0.570	0.142	0.096	0.474	12.7
0.61	0.305	0.610	0.084	0.124	0.368	12.1
0.83	0.415	0.830	0.041	0.050	0.065	7.1
0.95	0.475	0.950	0.033	0.019	0.008	4.3

Table 13: Exact operating characteristics of the *truncated* SPRT against a nominal $\alpha = \beta = 0.05$, by enumeration over the binomial tree with absorbing boundaries. The nominal rates are not attained at low τ, and most points reach the truncation cap rather than resolving early.

Effect on acceptance. The SPRT decides *which d-neighbours enter a region during growth*. It is a search-and-construction step, not an inference step: no acceptance claim in this paper rests on it. Region acceptance is decided downstream by the test on 300 *fresh* inputs drawn from the rule, with an anytime-valid confidence sequence and e-BH (Appendix F.1), and that test is valid whatever procedure proposed the region. A too-liberal SPRT admits noisier members, which inflates the generalization gap between member rate and fresh rate, a gap we measure and report per region, and makes region growth less sample-efficient. It cannot manufacture a validated region, because the validation never reuses the draws that grew it.

D.3 THE E-BH PROCEDURE

Sort region e-values $e_{(1)} \geq \cdots \geq e_{(K)}$. Let $\hat{k}$ be the largest index i such that $e_{(i)} \geq K/(\alpha i)$ (or $\hat{k} = 0$ if no such index exists). **Reject the $\hat{k}$ hypotheses with the largest e-values** (i.e., every hypothesis at rank $\leq \hat{k}$, not only the one at rank $\hat{k}$); this controls the false discovery rate at α under arbitrary dependence between regions that share draws (Wang & Ramdas, 2022). With the declared family, $K = 107$, and $\alpha = 0.05$ this gives $\hat{k} = 19$; the family of 30 would give 21 (Table 22).

D.4 MAP-ELITES: ARCHIVE AXES, INITIALIZATION, AND TIE-BREAKING

Table 14 gives the archive axes of each task, all small-integer buckets rather than uniform ten-value bins.

Task (regime)	# axes	Axes
ADD, default range	2	max digit width; carry count (capped at 9)
ADD, narrow range (hi $-$ lo ≤ 3)	3	width; signed length-diff (bucketed $[-2, 2]$); carry count (capped at 4)
ADD, wide range (lo ≥ 15)	2	signed length-diff (bucketed $[-2, 2]$); carry count (capped at $\max(\text{hi}, 9)$)
MUL, default range	2	width; product digit count (capped at 9)
MUL, narrow range	4	width; leading digit of A (floor-div 3); count of 9s (capped 3); trailing zeros (capped 3)
MUL, wide range	3	signed length-diff (bucketed $[-3, 3]$); # partial products; count of 9s
DATE	4	unit id; crosses-leap-day flag; month-end-clamp-required flag; $\lvert\text{offset}\rvert$ (floor-div 100, capped 9)
SORT	3	op id; $\lfloor \text{len}(xs)/8 \rfloor$ (capped 5); has-duplicates flag
GRAPH	3	op id; $\lfloor n/5 \rfloor$ (capped 5); true path length (capped 6)
STRING	3	op id; charset id; $\lfloor \text{length}/8 \rfloor$ (capped 5)

Table 14: MAP-Elites archive axes by task and range regime. DATE and narrow MUL use four axes; no axis is a literal ten-value grid dimension.

The archive is initialized by 32 uniform draws, a fixed number that does not depend on the task or on the total budget B. Each subsequent iteration samples a uniformly random *occupied* archive bin, mutates its elite by one d-step, evaluates the child, and replaces the incumbent only on *strict* improvement (challenger's L exceeds the incumbent's); a tie leaves the earlier-discovered elite in place. Hill climbing breaks ties between neighbours effectively at random: the neighbour list itself is a random subsample of the d-radius-1 neighbourhood, and among evaluated neighbours the first index achieving the maximum score is chosen, so a tie is broken by an already randomized draw order.

D.5 THE GRADED OBJECTIVE AND THE MATCHED-EVALUATIONS COMPARISON

The `prob` objective samples sequentially in stages $(2, 2, 4)$: an input stops at 2 answers if both agree, at 4 if all four are correct, and otherwise continues to 8. It is applied identically to every algorithm and task. Because it spends several calls per input, the matched-evaluations comparison runs `RAND` and `ME` on Qwen2.5-3B, addition widths 3–8, under `prob` with the budget raised to $B = 7{,}500$ (3.7 calls per input), so that the number of distinct inputs evaluated (2,190 and 2,038) matches the `bin` runs.

D.6 WORKED FORMULAS FOR EVERY REPORTED QUANTITY

Every measurement reduces to one estimator. An input x is drawn from $\mathcal{X}_C$, the model answers, the answer is parsed, and the exact oracle decides $\text{fail}_j(x) = \mathbb{1}[\text{parse}(M_j(x)) \neq f_C(x)]$, with j indexing independent draws at temperature 1.0 (an unparseable answer counts as a failure). For a finite input set S answered k times each, $\hat{p}(S, k) = \frac{1}{k|S|} \sum_{x \in S} \sum_{j=1}^{k} \text{fail}_j(x)$. Three different pipeline stages each estimate a quantity called π and differ in $(|S|, k)$ and when they are drawn; they are not interchangeable (Table 15; landscape and search-comparison quantities are in Table 16).

Quantity (where)	Definition	Worked value
π, failure-rate scan (Tables 11,**??**)	$\hat{p}(S,2)$, S=300 uniform inputs, before any search	3B, ADD 3–4: $55/600 = 0.092$
π, landscape probe (Table 18)	$\hat{p}(S,1)$, S=400 uniform inputs drawn inside each search run	3B, ADD 3–8: $189/400 = 0.472$
π, region background (Table 27)	$\hat{p}(S,2)$, S=400 uniform inputs redrawn inside each region run	14B, ADD 5–8: $179/800 = 0.224$
uniform failure rate (Table **??**)	every `RAND bin` evaluation, deduplicated, $\hat{p}(S,1)$ within each width	7B, width 8: $3528/6151 = 0.574$
exhaustive rate (Table 29)	$\hat{p}(S,1)$ at T=0, S=all 900^2 three-digit sums, no sampling error	3B, 2 carries, no cascade: 0.0376 over 259,200
ratio (Table 29)	exhaustive rate with a cascade / rate without, matched carry count	3B, 2 carries: $0.2115/0.0376 = 5.6\times$
carry contrast (Table 28)	$\hat{p}(\textsc{Pos}_k,2) - \hat{p}(\textsc{Neg}_k,2)$, pooled over that model's runs	3B, k=4: $1500/2600 - 114/2600 = +0.533$
order asymmetry (Table 36)	$\hat{p}$(2nd operand longer)$-\hat{p}$(1st longer), widths 5–8, two-sided Fisher exact	7B: $4976/7784 - 2627/7703 = +0.298$
transfer lift (Appendix **??**)	target's rate inside a source rule / target's overall uniform rate	1.5B rule on 7B: $0.706/0.404 = 1.75$

Table 15: Definitions and worked values for the principal reported quantities.

Quantity (where)	Definition	Worked value
A, local assortativity (Table 18)	$\hat{p}(N,1)/\pi$, N: 8 sampled d=1 neighbours of each of up to 40 failing points	3B, ADD 3–8: $(238/320)/0.4725 = 1.57$
$\rho(1)$ (Table 18)	Pearson corr. of L_t vs. L_{t+1} along a 150-step random walk, $L(x) = \hat{p}(\{x\},4)$	3B, ADD 3–8: 0.476
$\hat{A}_{12}$ (Table 18)	$\big(\#\{a{>}b\} + \frac{1}{2}\#\{a{=}b\}\big)/(n_1 n_2)$ over per-seed distinct-failing-bin counts	3B, ADD 3–8, `bin`: $(78 + \frac{1}{2}{\cdot}11)/100 = 0.835$
effective failing bins (Table 19)	$\exp(H)$, $H = -\sum_c p_c \ln p_c$, p_c=fraction of confirmed failures in bin c	3B, ADD 3–8, `ME`, bin: H=3.374 $\Rightarrow$ $\exp(H)$=29.2

Table 16: Landscape and search-comparison quantities.

Quantity (where)	Definition	Worked value
fresh rate (Table 27)	$\hat{p}(F,4)$, F=300 fresh inputs the rule admits, rejection-sampled after the rule is fixed	14B, `len_diff_AB > 2.5`: $868/1200 = 0.723$
lift (Table 27)	fresh rate / cell background π	same region: $0.723/0.224 = 3.23$
conventional lower bound (Table 22)	Clopper–Pearson lower limit at α=0.05 on k failures in $n = 4\lvert F\rvert$ = 1,200 (four answers per input counted as four independent draws)	same region: 0.701 against $\tau_{\text{eff}} = 0.5$
anytime-valid interval (Table 22)	set of m with $E(k,n;m) < 1/\alpha$, $E(k,n;m) = \mathrm{B}(1{+}k, 1{+}n{-}k)/[\mathrm{B}(1,1)\, m^k (1-m)^{n-k}]$, evaluated at n= #inputs (not answers)	carry region, k=161, n=300: $[0.440, 0.631]$
zero-failure certificate (Appendix **??**)	certifies $p < 1-\alpha^{1/N}$ at level α; separately, the chance of *obtaining* a certificate is $(1-p)^N$, which drops below 0.2 once $p > 1 - 0.2^{1/N}$	N=400, α=0.05/19: certifies $p < 0.0147$, likely only if $p < 0.004$

Table 17: Region and certification quantities. B is the beta function.

E THE SEARCH COMPARISON IN FULL

		landscape				ME vs RAND		
Model	Cell	π	A	$\rho(1)$	bins	$\hat{A}_{12}$ (bin)	p	rel. gain
1.5B	ADD 3–6	0.507	1.19	0.50	20	0.630	0.278	+1.9%
1.5B	ADD 3–8	0.623	1.27	0.40	39	0.815	0.016*	+4.3%
3B	ADD 3–8	0.472	1.57	0.48	39	0.835	0.010*	+6.2%
7B	MUL 2–3	0.482	1.70	0.60	5	0.350	0.185	−6.2%
7B	GRAPH	0.385	1.79	0.63	37	0.570	0.620	+2.8%
7B	ADD 3–8	0.367	1.80	0.58	39	0.750	0.061	+6.4%
3B	GRAPH	0.395	1.84	0.54	38	0.640	0.527	+1.9%
3B	ADD 3–6	0.295	2.03	0.52	65	0.405	0.490	−1.5%
3B	ADD 3–5	0.215	2.24	0.34	42	0.535	0.818	+0.0%
14B	ADD 5–8	0.242	2.41	0.48	60	0.775	0.040*	+5.2%
14B	ADD 5–8 (replicate)	0.242	2.46	0.48	60	0.900	0.043*	+8.3%

Table 18: MAP-Elites against uniform sampling in all eleven cells of the search comparison. Two of the eleven are uninformative on coverage (MUL 2–3 has 5 reachable bins and ADD 3–6 on the 1.5B has 20, too few to rank methods); the coverage claims in the text exclude them. * marks Holm-corrected $p < 0.05$. Under `prob`, MAP-Elites reaches significance nowhere, with gains of -7.1% to $+4.7\%$. The 14B cell has two replicates (ten seeds and four algorithms; five seeds and two algorithms); direction and significance agree between them.

Hill climbing is significantly worse on coverage in 20 of 24 comparisons (15–68%) and no faster than uniform sampling at reaching a first failure (Appendix K). Local assortativity does not predict which cells show a significant guided-search advantage: ordered by A, the effect sizes are 0.630, 0.815, 0.835, 0.350, 0.570, 0.750, 0.640, 0.405, 0.535, 0.775, 0.900, and the four significant cells sit at $A = 1.27, 1.57, 2.41, 2.46$, scattered across the range; within the 3B alone (model identity and recipe held fixed) the advantage *falls* as A rises (0.835 at A=1.57 down to 0.535 at A=2.24). The guided-search advantage therefore does not rise with assortativity.

Model	Cell	eff. bins RAND	eff. bins ME	rel. gain	$\hat{A}_{12}$	p (Holm)
1.5B	ADD 3–6	11.7	17.7	+51.5%	1.000	0.0020*
1.5B	ADD 3–8	21.3	30.9	+45.3%	1.000	0.0020*
3B	ADD 3–8	20.0	29.2	+45.8%	1.000	0.0020*
7B	MUL 2–3	2.7	2.9	+7.0%	0.790	0.0312*
7B	GRAPH	19.3	25.6	+32.8%	1.000	0.0020*
7B	ADD 3–8	20.0	29.0	+44.8%	1.000	0.0020*
3B	GRAPH	27.4	37.9	+38.2%	1.000	0.0238*
3B	ADD 3–6	38.7	48.6	+25.8%	1.000	0.0020*
3B	ADD 3–5	27.4	31.7	+15.7%	1.000	0.0020*
14B	ADD 5–8	34.0	50.8	+49.6%	1.000	0.0020*
14B	ADD 5–8 (replicate)	33.4	52.9	+58.7%	1.000	0.0238*

Table 19: **Descriptive, not a search-performance comparison.** The *effective* number of failing bins ($\exp(H)$ over the distribution of confirmed failures across bins) for MAP-Elites and uniform sampling, same cells and seeds as Table 18. The significance markers do not show that ME finds more failures: this metric is close to definitional, since an archive whose explicit objective is one elite per bin spreads its confirmed failures across bins, and the same cells show $\hat{A}_{12}$ of 0.350–0.900, mostly non-significant, on *coverage* (Table 18), the quantity Section **??** compares. The table supports a narrow, practical conclusion: ME's output is a better-distributed *seed set* for region growth, not evidence of a better search.

E.1 THE RARE-FAILURE REGIME IN SIMULATION

The search comparison covers the band of failure rates in which regions can be grown. To see what to expect when failures are rare, we use a simulated target, with no model calls, in which rarity is a parameter.

A planted carry-cascade region fails at p_{in}=0.85 and everything outside it at p_{out}=0, so every failure a run finds lies inside the planted region and the only question a run answers is whether the method found it. The region's prevalence sweeps down more than two orders of magnitude, ending at the same order as the rarest failures the exhaustive enumeration locates inside ADD (8.8×10^{-5} among three-carry sums without a cascade; Table 29). Each run has a budget of 2,000 calls, with ten seeds per setting (Table 20).

Region prevalence	Algorithm	Coverage (mean)	Seeds finding nothing	$\hat{A}_{12}$ vs RAND
1.1×10^{-2}	RAND	8.80	0/10	
	HC	6.10	0/10	0.200*
	ME	9.60	1/10	0.595
7.9×10^{-4}	RAND	1.60	2/10	
	HC	1.10	7/10	0.365
	ME	3.40	3/10	0.655
7.0×10^{-5}	RAND	0.30	8/10	
	HC	0.00	10/10	0.400
	ME	0.50	9/10	0.460

Table 20: Simulated rare-failure regime, widths 3–8, binary objective. Starred $\hat{A}_{12}$ is Holm-significant at 0.05. At the lowest prevalence level no method finds the region in most seeds.

At the rarity found by exhaustive enumeration, no method works. At prevalence 7.0×10^{-5} uniform sampling finds no failure at all in 8 of 10 seeds, hill climbing in 10 of 10, and the archive in 9 of 10. No method is significantly better than any other, because almost no run finds anything.

The middle prevalence level is underpowered. At 7.9×10^{-4} the archive reaches more than twice uniform sampling's coverage (3.40 against 1.60) and misses the region in 3 of 10 seeds against uniform sampling's 2, with $\hat{A}_{12} = 0.655$, which does not clear Holm correction. The comparison is inconclusive at ten seeds.

Hill climbing is worst everywhere, and the reason is structural rather than empirical. With p_{out}=0 the binary objective is exactly zero outside the planted region, so there is no gradient to climb and hill climbing reduces to restarting. Its significant loss at the highest prevalence level ($\hat{A}_{12} = 0.200$) reflects that cost. This is the near-flat landscape of the search comparison (Table 18) in its most extreme form.

A graded objective does not rescue the rare regime. A binary indicator carries no signal outside the region it indicates, which suggests changing the objective. Whether the *objective* resolves failure finely enough is, however, a different question from whether the *surface* has a gradient at all. We separate the two (Table 21) by crossing two objectives, the binary indicator and `prob`, the sequential estimate of $p(x)$, with two surfaces: the step surface above, and a graded surface whose failure probability decays as proximity to the region raised to a power, with the power chosen so that the overall failure rate matches the step surface to within a factor of 1.4 at every prevalence level. Matching the failure rate keeps the comparison about the gradient rather than about how much failure mass each surface contains.

The graded objective is strictly worse. At 7.9×10^{-4} it takes uniform sampling from finding the region in 8 of 10 seeds to 3 of 10, and at 7.0×10^{-5} from 2 of 10 to none. The mechanism is budget arithmetic: `prob` spends up to eight calls resolving $p(x)$ at a single point, so at a fixed budget of 2,000 calls it examines 480 distinct points where the binary objective examines 2,000. When failures are rare the binding constraint is how many distinct points a method can look at, and paying four calls extra per point to measure each one more precisely is the wrong trade.

The graded surface changes nothing once the failure rate is matched. At the lowest prevalence level the graded and step rows are identical to two decimal places. Guided search does win on a graded surface whose failure rate is not matched: at a shallower decay, where the graded surface carries roughly 45 times the failure mass of the step surface, MAP-Elites reaches $\hat{A}_{12} = 0.820$ against uniform sampling and the result is Holm-significant. That is a finding about abundance, not about gradients: at that failure rate the regime is no longer rare.

Prevalence	Surface	Objective	RAND cov.	ME cov.	Seeds finding nothing (RAND)
7.9×10^{-4}	step	bin	1.60	3.40	2/10
	step	prob	0.40	1.60	7/10
	graded	bin	1.80	4.00	2/10
	graded	prob	0.40	1.60	7/10
7.0×10^{-5}	step	bin	0.30	0.50	8/10
	step	prob	0.00	0.00	10/10
	graded	bin	0.30	0.50	8/10
	graded	prob	0.00	0.00	10/10

Table 21: Objective crossed with surface at matched failure rate, ten seeds, budget 2,000. The graded objective is worse than the binary one everywhere, and the graded surface changes nothing once the failure rate is matched.

Implication for search at low failure rates. What needs to change is not the objective's resolution but the proposer's reach: at low failure rates the scarce resource is distinct points examined per unit budget, which argues for cheaper evaluation or a proposal distribution that concentrates on plausible regions, and against anything that spends more calls per point.

Scope. A simulated target offers a proposer no structure beyond the planted rule and the feature map, so this bounds what to expect on real models rather than standing in for them. In particular it cannot tell us whether a real model's rare failures cluster in ways an archive would exploit.

F Region catalogue and acceptance analysis

F.1 From 107 candidates to 19

The 107 candidates come from Qwen2.5-1.5B (7), 3B (19), 7B (21) and 14B (14), Granite-3.1-2B (11) and 8B (9), Mistral-7B (8) and Phi-3.5-mini (18). Mistral-7B's eight were grown on direct answers, on which its failure rate lies above the searchable band at every level, and none passes the conventional test. The validation step makes two corrections. The first counts inputs rather than answers: acceptance answers each of m=300 fresh inputs four times, and a Clopper–Pearson bound computed at n=4m treats the four answers to one input as independent observations, which they are not. The second accounts for testing many regions at once: e-BH controls the false-discovery rate across them, using e-values from the anytime-valid confidence sequence, which on a fixed fresh sample is a conservative bound. Table 22 holds the acceptance criterion fixed (`lower >= tau_eff and lower > pi_background`) and varies only how `lower` is computed.

Step	What it corrects	Regions	ADD	GRAPH	MUL
grown	—	107	92	11	4
conventional: CP at n=4m	nothing	30	19	7	4
(a) CP at n=m	answers counted as inputs	24	13	7	4
(b) + anytime-valid CS	optional stopping (conservative)	21	10	7	4
(c) + e-BH over the 30	multiplicity	21	10	7	4
(c′) + e-BH over all 107	multiplicity	19	8	7	4

Table 22: Decomposition of the validation step. Of the nine regions removed after the conventional test, six fall to counting inputs and three to the anytime-valid bound. e-BH removes none when its family is the 30 regions that pass the conventional test, and two when the family is all 107 candidates, which is the declaration we use. Every removal is on addition.

Counting inputs removes six regions. Among them is the carry-cascade rule found by search (`max_carry_chain_9 > 0.5 AND leading_digit_A <= 2.5`, fresh rate 0.536 against τ_{eff}=0.50): its bound is 0.512 at n=1,200 and 0.488 at n=300, and its e-value is 0.161, far below the $1/\alpha = 20$ an e-process needs. It fails every analysis at the input count. Section 4.3 tests the underlying mechanism directly with a designed contrast, which does not depend on this threshold.

The anytime-valid bound removes three. Three regions clear a correct fixed-sample bound at n=300 and are removed only by the anytime-valid sequence (one Qwen2.5-1.5B and two Phi-3.5-mini addition regions), all with bounds within 0.02 of their thresholds. These are the marginal cases in which the cost of anytime validity falls.

The e-BH family. e-BH controls the false-discovery rate over a declared family of hypotheses, which must be the set of hypotheses tested. Every grown region is tested on fresh inputs, so the family is all 107 candidates, and e-BH over it keeps 19. Restricting the family to the 30 regions that pass the conventional test would keep 21, but that family is chosen using the same fresh data and so does not control the false-discovery rate. The two regions that differ are the only ones whose e-values lie near the e-BH thresholds: the Qwen2.5-7B region `len_diff_AB <= -1` at $e = 29.7$ and the Qwen2.5-3B region `leading_digit_A <= 2 AND log_magnitude >= 8` at $e = 72.7$. Every other survivor has an e-value between 10^4 and 10^{41}, and every other casualty an e-value below 1.

Row (a) computes the bound at the input count from each region's mean failure rate over its four answers per input; a clustered test on per-answer outcomes lies between row (a) and the conventional row.

F.2 What the corrections buy on known ground truth

The decomposition above accounts for one catalogue. It says which correction removed which region; it cannot say how often each correction matters, because the truth about real regions is unknown. On a simulated target the truth is known by construction, so we measure it there, with no model calls.

A target is a stochastic function over ADD inputs with a known per-input failure probability. In the *null* arm that probability is a constant π everywhere, so there is no structure to find and every accepted region is false by construction. In the *planted* arm a rule of prevalence ρ=0.1 fails at p_{in} and the rest at p_{out}, chosen so the background is π and the planted region's lift over background is exactly L. We sweep L from 1.2 to 3.0, which is the range the real catalogue occupies (Table 27 lifts run 1.49 to 3.23), rather than the roughly 57× lift of the simulated models in Appendix C. Throughout, π=0.30 and τ_{eff}=0.50.

A region's *true* rate is the exact expected failure probability of its rule's population, averaged over 2,000 fresh draws using the known per-input probability rather than sampled outcomes. A discovery is true when that rate clears the same τ_{eff} the acceptance claims. We run the four acceptance tests of Table 22 and one more: the counterfactual pipeline that reports the rate on the very data that grew the region and never draws a fresh sample at all, which is what failure-region discovery looks like without fresh-sample validation.

Arm	Acceptance test	Accepted	FDR	Power
null (π everywhere)	member rate, no fresh sample	0	—	—
	all four fresh-sample tests	0	—	—
planted, L=1.6	member rate, no fresh sample	144	**1.000**	—
	fresh sample, CP at n=4m	3	**1.000**	—
	fresh sample, CP at n=m	0	—	—
	+ anytime-valid, + e-BH	0	—	—
planted, L=2.0	member rate, no fresh sample	2308	0.011	0.972
	fresh sample, CP at n=4m	2331	0.000	0.993
	fresh sample, CP at n=m	2300	0.000	0.980
	+ anytime-valid	1419	0.000	0.605
	+ e-BH	1273	0.000	0.542
planted, L=3.0	member rate, no fresh sample	2337	0.000	0.998
	+ e-BH	2300	0.000	0.982

Table 23: Realized false-discovery rate and power, K=200 candidate regions per trial and 12 trials per arm and lift, so roughly 2,390 regions per row-block. L=1.2 is omitted because no test accepts anything there. The K=20 and K=80 sweeps give the same picture.

Against pure noise the pipeline accepts nothing, even with no fresh sample. In the null arm all 2,360 grown regions are rejected by every test. For the member-rate test, which has no interval, the protection is the absolute floor: τ_{eff} sits 0.20 above the background, far outside what sampling noise reaches. For the fresh-sample tests it is the lower confidence bound, as the relative rule below shows.

The failure mode is a narrow band just under the threshold, and inside it the unvalidated pipeline is wrong every time. At $L{=}1.6$ the planted region fails at 0.48 against a floor of 0.50: genuinely elevated, but short of what acceptance claims. There the member-rate pipeline accepts 144 of 2,392 regions and *every one of them is false*. Drawing a single fresh sample removes 141 of those 144, and counting inputs removes the remaining 3. This measures the central claim directly: fresh-sample validation removes 97.9% of the false acceptances in the band where false acceptances happen at all.

The conservative layer has a cost. At $L{=}2.0$, where the planted region is genuinely acceptable, power falls from 0.980 under a correct fixed-sample bound to 0.605 under the anytime-valid sequence and 0.542 after e-BH. Roughly four in ten real regions are lost to the anytime-valid layer, which a fixed fresh sample does not need.

e-BH matters only when a region sits near the threshold. In simulation it removes 10.3% of the anytime survivors at $L{=}2.0$ (14.4% at $K{=}20$, 10.8% at $K{=}80$). The real catalogue's e-values are close to bimodal, enormous for survivors and below 1 for casualties, so for most regions no multiplicity threshold separates them differently, and e-BH removes none at $K{=}30$. The exceptions are the regions at $e{=}29.7$ and $e{=}72.7$, which the declared family of all 107 grown regions removes (Appendix F.1). Where the simulation predicts e-BH matters, near the threshold rather than at either mode, is where the real catalogue is sensitive to it.

What the anytime-valid layer costs here. Down Table 23, the anytime-valid row never improves on the fixed-sample row at the input count: the false-discovery rate is already zero at $n{=}m$, and the sequence costs 38% of power. This simulation uses a fixed fresh sample and no adaptive stopping, the setting in which a fixed-sample bound over inputs is already valid, so the table measures the cost of the anytime-valid layer rather than a benefit. On the real catalogue the same layer removes three regions that pass a correct fixed-sample bound at $n{=}300$ (Appendix F.1), and the main text reports the counts with and without it.

The power column. At $L{=}2.0$ the in-sample test has *lower* power (0.972) than the fresh-sample test (0.993), which inverts the usual direction. The cause is that the two tests measure different populations. The member rate is taken on the grown region, whose membership is decided by the truncated SPRT, and at $\tau{=}0.5$ that test admits a non-member 19% of the time (Appendix D). Those admissions dilute the member set: averaged over 2,387 regions at $L{=}2.0$ the member rate runs 0.015 *below* the rule's fresh rate, while at $L{=}3.0$, where the planted region is far above the admission threshold and dilution is rarer, it runs 0.010 above. In-sample scoring is not uniformly optimistic; it is optimistic about the population the rule identifies and pessimistic about the one growth actually collected, and which effect wins depends on how liberal the growth test was.

Protection under a relative acceptance rule

A relative rule, accepting when the bound clears $L_{\text{rel}} \cdot \pi$, is the natural alternative (Appendix F.4), and at $L_{\text{rel}}{=}1.25$ that threshold is 0.375, only 0.075 above the background. The per-region bounds show whether the relative rule keeps the protection.

The relative rule keeps the protection. At every L_{rel} from 1.25 to 2.0, *zero* regions are accepted in the null arm, under both the fixed-sample and the anytime-valid bound. The protection does not come from where the threshold sits relative to the background; it comes from the bound being a one-sided lower confidence limit at 300 inputs rather than a point estimate. Clearing 0.375 from below requires an observed rate near 0.42, which is about four and a half standard errors above a background of 0.30.

It also recovers real regions the floor rejects. At $L{=}1.6$, where the planted region genuinely fails at 0.48 and the absolute rule accepts nothing, the relative rule at $L_{\text{rel}}{=}1.25$ accepts 1,497 regions under the anytime-valid bound with an FDR of 0.000 and power 0.661. Those are exactly the regions Appendix F.4 argues the floor is wrongly discarding, and in simulation they are recovered at no cost

in false discoveries. This is the strongest evidence for a relative rule; on the Phi-3.5-mini catalogue, a relative rule still rejects every region.

Scope. One task, one background rate, an axis-aligned planted rule, and a target whose per-input probability is piecewise constant. A real model's failure surface has none of these properties. The simulation supports a statement about the acceptance tests given a discovery pipeline, not a claim that real failure regions are distributed like these.

F.3 Other model families

A non-Qwen model is informative for region search only if some cell of it is searchable, $0.05 \leq \pi \leq 0.5$. On direct answers Mistral sits at π=0.677 at its easiest level and Phi-3.5-mini at 0.933, both far above the band. Three conditions in Table 11 qualify: Phi-3.5-mini *with reasoning permitted* at π=0.380 on ADD 3–4; Granite-3.1-2B on direct answers, the one searchable model in a screen of three further families; and Granite-3.1-8B on direct answers at π=0.302 on ADD 3–4. We report a catalogue on each.

Phi-3.5-mini, reasoning permitted

The catalogue comes from two runs of four regions each (seeds 0 and 1) in reasoning mode on ADD 3–4, using 71,000 model calls (Table 24).

Seed	Rule	Rate	Lift	Gap	Passes conventional test
0	`max_carry_chain_9 > 0.5`	0.443	1.16	+0.07	no
0	`sum_n_digits > 4.5`	0.434	1.14	+0.14	no
0	`third_digit_A > 6.5 AND leading_digit_B > 5.5`	0.398	1.05	+0.18	no
0	`leading_digit_A > 6.5`	0.380	1.00	+0.13	no
1	`len_diff_AB <= -0.5 AND leading_digit_A > 6.5`	0.539	1.48	+0.02	**yes**
1	`len_diff_AB <= -0.5`	0.529	1.45	+0.05	no
1	`leading_digit_B > 5.5 AND n_digits_B > 3.5`	0.456	1.25	+0.06	no
1	`n_digits_B > 3.5`	0.436	1.20	+0.07	no

Table 24: Phi-3.5-mini, reasoning mode, ADD 3–4, $\tau_{\text{eff}} = 0.53$ (seed 0) and 0.51 (seed 1). One of eight regions passes the conventional test.

One region passes the conventional test. `len_diff_AB <= -0.5 AND leading_digit_A > 6.5` reaches 0.539 against τ_{eff}=0.514, with the tightest generalization gap of the eight (+0.018). It reproduces, on a different family and under reasoning, the signed-length-difference structure the Qwen catalogues return on direct answers: evidence for the rule rather than for the catalogue.

It fails once inputs are counted, in the band the simulation predicts. Its conventional bound is 0.515, clearing the threshold by 0.001. At the input count the bound falls to 0.491 and under the anytime-valid sequence to 0.444, against a threshold of 0.514; its e-value is 0.109, more than two orders of magnitude below the $1/\alpha = 20$ an e-process needs. This is the marginal band that Appendix F.2 identifies as the only place false acceptances occur, and where the carry-cascade rule also fell.

In the one reasoning-mode regime in which a non-Qwen model is searchable, the catalogue produces one candidate that passes the conventional test and falls at the margin.

These eight regions are part of the eighteen Phi-3.5-mini regions in the catalogue; the other ten come from direct-answer runs at higher failure rates (Appendix F.4), and all eighteen belong to the family over which e-BH is applied (Appendix F.1).

Screening three further families

Beyond the Qwen models of the search comparison, we screened three further open-weight families for searchable cells on ADD, on the same width grid and against the same exact oracle (300 inputs at two answers per level): Granite-3.1-2B, Phi-4-mini and Falcon3-3B.

One of the three is searchable on direct answers. Granite-3.1-2B sits at π=0.345 on ADD 3–4 and 0.463 on 5–6, both inside the band $0.05 \leq \pi \leq 0.5$, with a format-failure rate of 0.001, so those failures are arithmetic and not malformed output (Table 11). Phi-4-mini is saturated at every level. Falcon3-3B, at π=0.99 with 26% of its answers unparseable at the wider levels, is the case where saturation cannot be separated from non-compliance.

We grew two catalogues on Granite-3.1-2B (seeds 0 and 1, ADD widths 3–6, direct answers, 75,704 model calls in total). They returned the 11 regions of Table 25: five pass the conventional test and four survive the anytime-valid bound and e-BH.

Seed	Rule	Rate	Lift	Gap	Conv. test	Validated
1	`len_diff_AB <= -0.5 AND n_digits_A > 4.5`	0.956	1.77	+0.02	**yes**	**yes**
0	`len_diff_AB <= -0.5`	0.822	1.55	+0.09	**yes**	**yes**
1	`len_diff_AB <= -0.5 AND count_0s > 0.5`	0.811	1.51	+0.17	**yes**	**yes**
0	`len_diff_AB <= -0.5` (second region)	0.788	1.49	+0.13	**yes**	**yes**
0	`count_0s > 1.5 AND digit_len_gap > 0.5 AND leading_digit_A > 4.5`	0.709	1.34	+0.16	**yes**	no
1	`len_diff_AB > 0.5 AND max_carry_chain_9 > 0.5`	0.706	1.31	+0.19	no	no
1	`n_digits_A > 5.5 AND leading_digit_B > 8.5 AND n_digits_B <= 4.5`	0.644	1.20	+0.28	no	no
1	`third_digit_A > 8.5 AND len_diff_AB > 0.5`	0.644	1.20	+0.22	no	no
1	`leading_digit_A > 8.5 AND len_diff_AB > 0.5`	0.623	1.16	+0.29	no	no
1	`n_digits_A > 5.5 AND max_consec_carries <= 2.5 AND len_diff_AB > 0.5`	0.521	0.97	+0.32	no	no
1	`n_digits_A >= 5`	0.512	0.95	+0.39	no	no

Table 25: Granite-3.1-2B, direct answers, ADD 3–6, τ_{eff}=0.679 (seed 0, π=0.529) and 0.689 (seed 1, π=0.539). *Conv. test*: passes the conventional test; *validated*: also passes after counting inputs, the anytime-valid bound and e-BH.

A second Granite size, and the same rule a third time. Granite-3.1-8B is searchable only at ADD 3–4 (π=0.302), and one catalogue there (9 regions, 74,076 calls) returned one region that passes the conventional test and survives the anytime-valid bound: `len_diff_AB <= -0.5 AND leading_digit_B > 7.5 AND sum_overflow <= 0.5`, rate 0.673 against a background of 0.326, lift 2.06, the fourth-highest in the catalogue. Its leading literal is the signed length difference again, now on a third model size and a second family. The 8B model is a 4-bit quantization, so this point sits on the far side of the precision boundary discussed in Appendix B and is not a clean scale comparison against the 2B.

Scope of the cross-family evidence. Five regions on models from a different developer, with a different tokenizer and a different training recipe, are in the validated catalogue, and every one is a signed-length-difference condition, the structure the Qwen catalogues return. The rule is found independently, by the same search, on models sharing nothing with the Qwen ladder but the task. The single-family limitation is narrowed, not removed: fourteen of the 19 validated hypotheses are Qwen, two families is not a survey, and size and precision vary together within both ladders (Appendix B).

The fifth Granite region falls in the marginal band. It clears τ_{eff} by 0.008 on the conventional bound and fails every analysis at the input count, with an e-value of 0.13. Like the carry-cascade rule and the Phi-3.5-mini region above, it is one of the nine regions removed in the band just above the acceptance floor, where the simulation of Appendix F.2 places false acceptances.

F.4 THE ABSOLUTE ACCEPTANCE FLOOR, AND WHAT A RELATIVE RULE WOULD DO

The acceptance threshold $\tau_{\text{eff}} = \max(0.5, \min(0.95, \pi + 0.15))$ is an absolute floor (Figure 6a), so a cell with a low background rate must clear a much larger *lift* than a cell with a high one. Table 26 gives the catalogue's sensitivity to a relative criterion: the same anytime-valid lower bound tested against $L \cdot \pi$ rather than against τ_{eff}, over a grid of L.

Relative rule (CS lower $\geq L \cdot \pi$)	Accepted	ADD	GRAPH	MUL	Model families
$L = 1.25$	31	20	7	4	Qwen, Granite
$L = 1.50$	21	11	7	3	Qwen, Granite
$L = 1.75$	14	9	3	2	Qwen, Granite
$L = 2.00$	5	4	1	0	Qwen only
$L = 2.50$	2	2	0	0	Qwen only
$L = 3.00$	0	0	0	0	—
absolute floor, before e-BH	21	10	7	4	Qwen, Granite

Table 26: Sensitivity of the catalogue to a relative acceptance criterion. The count is not especially sensitive near L=1.5, which admits exactly the 21 the absolute floor does before e-BH, though not the same 21: the relative rule takes one more ADD region and one fewer MUL region. Granite survives to L=1.75 and drops out above it; no value of L admits a Phi-3.5-mini or Mistral catalogue.

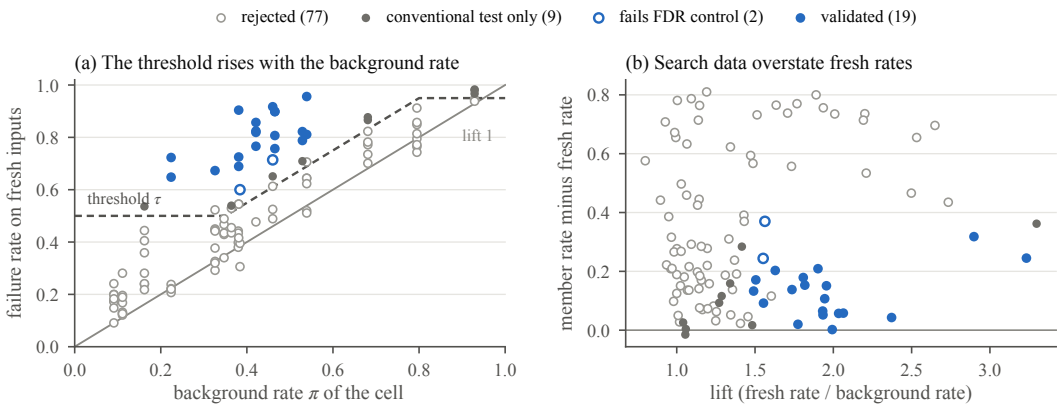

Figure 6: Every grown region, encoded as in Figure 2c. Panel (a): each region's fresh failure rate against its cell's background rate π, with the acceptance threshold τ_{eff} as the dashed line (flat at 0.5 until $\pi = 0.35$, rising thereafter, capped at 0.95) and the solid diagonal marking lift 1. Because the threshold is flat on the left, a low-background cell must clear a far larger lift. Panel (b): generalization gap, the member rate minus the fresh rate, against lift. Hollow grey: the 77 regions rejected by the conventional test; filled grey: the 9 that pass it but fail once inputs are counted or under the anytime-valid bound; hollow blue: the 2 that pass the anytime-valid bound but are removed by e-BH over all 107 candidates (false-discovery-rate, or FDR, control); filled blue: the 19 validated hypotheses.

The floor does reject regions that a relative rule would keep. The clearest is the 14B at fixed width 6, lift 2.6 against a background of 0.091, rejected because an absolute floor of 0.5 is far above what a low-background cell can reach. That case is the argument for a relative criterion.

A relative rule does not widen the family evidence further. Granite clears the absolute floor already and a relative rule keeps it up to L=1.75, where the 8B region's larger lift over a lower background carries it one step further than the 2B regions. At no threshold does a Phi-3.5-mini or Mistral catalogue enter, so their absence from the validated catalogue is a property of the data and not an artefact of the absolute floor.

Tight generalization gaps on the Phi-3.5-mini catalogue come with low lifts. Those 18 regions have the tightest rule-to-member agreement in the paper *as a catalogue*, mean gap 0.062 against 0.18 for the next tightest model, though single accepted regions elsewhere are as tight or tighter (the

3B `len_diff_AB <= -2.5` region sits at 0.00, Table 27). Their *lifts*, however, run only 0.98 to 1.48 against backgrounds of π=0.35 to π=0.93: the rules accurately describe populations that are barely more failure-prone than their level as a whole. A small gap means the rule is accurate, not that the region is interesting, and a relative rule at $L \geq 1.5$ rejects every one of them too. Of the eighteen Phi-3.5-mini regions, fourteen fall below the acceptance floor, and the four that clear it are removed at the input count or by the anytime-valid sequence (Table 22).

Model	Task	Rule	rate (fresh)	π (backgr.)	lift	gap
14B	ADD	`len_diff_AB > 2.5`	0.723	0.224	3.23	+0.25
14B	ADD	`len_diff_AB <= -1.5 AND leading_digit_A <= 2.5`	0.648	0.224	2.90	+0.32
7B	GRAPH	shortest-path query, target reachable	0.904	0.381	2.37	+0.04
Granite-8B	ADD	`len_diff_AB <= -0.5 AND leading_digit_B > 7.5 AND sum_overflow <= 0.5`	0.673	0.326	2.06	+0.06
3B	GRAPH	shortest path, depth >1, target with several in-edges	0.857	0.421	2.03	+0.06
3B	ADD	`len_diff_AB <= -2.5`	0.917	0.460	1.99	+0.00
3B	GRAPH	true path length ≥ 2	0.824	0.421	1.96	+0.15
3B	GRAPH	target with several in-edges, shortest path	0.819	0.421	1.94	+0.11
7B	MUL	product of six or more digits	0.899	0.465	1.93	+0.05
7B	MUL	product of six or more digits (second seed)	0.898	0.465	1.93	+0.07
7B	GRAPH	unreachable target on a dense graph	0.725	0.381	1.90	+0.21
3B	GRAPH	true path length ≥ 1	0.766	0.421	1.82	+0.15
7B	GRAPH	shortest path, depth >1	0.689	0.381	1.81	+0.18
Granite-2B	ADD	`len_diff_AB <= -0.5 AND n_digits_A > 4.5`	0.956	0.539	1.77	+0.02
7B	MUL	`n_digits_A > 2.5 AND leading_digit_B > 7.5`	0.807	0.465	1.74	+0.14
7B	MUL	`n_nonzero_digits_B > 2.5`	0.757	0.465	1.63	+0.20
7B	ADD	`len_diff_AB <= -1`†	0.600	0.384	1.56	+0.37
Granite-2B	ADD	`len_diff_AB <= -0.5` (seed 0)	0.822	0.529	1.55	+0.09
3B	ADD	`leading_digit_A <= 2 AND log_magnitude >= 8`†	0.714	0.460	1.55	+0.24
Granite-2B	ADD	`len_diff_AB <= -0.5 AND count_0s > 0.5`	0.811	0.539	1.51	+0.17
Granite-2B	ADD	`len_diff_AB <= -0.5` (seed 0, second region)	0.788	0.529	1.49	+0.13

Table 27: All 21 regions that pass the anytime-valid bound, ordered by lift; the two marked † are removed by e-BH over all 107 candidates, leaving the 19 validated hypotheses. Rate is on 300 fresh inputs the search never saw; π is the background rate of the level the region was grown in; gap is member rate minus fresh rate. Of the 19, fourteen are on three Qwen2.5 models, four on Granite-3.1-2B and one on its 8B sibling, which runs with 4-bit weights (Appendix F.3).

Clustering of the failure set. Enumerating all $900^2 = 810{,}000$ three-digit sums gives the complete failure set and its connected components under the digit metric: 38,342 failures on the 3B (rate 0.0473), 99.63% in one component; 19,141 on the 7B (rate 0.0236), 99.09% in one component. More than 99% of all failures lie in a single connected region on both models. Connectivity alone is not evidence of clustering, since at this failure rate randomly placed failures would also mostly connect; the evidence for clustering is the excess failure rate among neighbours (Section **??**). What connectivity does show is that growing a region by walking between failing neighbours can reach most failures.

G CARRY PROPAGATION IN DETAIL

The cascade effect is not a digit-9 pattern effect. Cascade inputs could display a visible run of 9 digits, in which case the model might be reacting to a lexical or tokenizer-level pattern rather than to carry propagation. Two observations argue against this.

	cascade arm − no-cascade arm, by cascade length k				
Model	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$
3B	-0.017	$+0.533$	$+0.540$	$+0.540$	$+0.533$
7B	$+0.004$	$+0.281$	$+0.393$	$+0.487$	$+0.386$
14B	-0.007	$+0.176$	$+0.305$	$+0.360$	$+0.324$

Table 28: Cascade vs. no-cascade contrast at matched carry count, three open-weight models, six designed runs (800–5,200 draws per arm per k). Every null point ($k = 0$) lies inside noise; every $k \geq 1$ anytime-valid interval is disjoint between arms.

		no cascade		cascade ≥ 1		
Model	carries	rate	inputs	rate	inputs	ratio
3B	2	0.0376	259,200	0.2115	32,400	$5.6\times$
3B	3	0.0143	91,125	0.1908	40,050	$13.3\times$
7B	2	0.0044	259,200	0.1561	32,400	$35.4\times$
7B	3	0.0001	91,125	0.1746	40,050	$1989\times$

Table 29: Failure rate by carry count, split by whether the carries cascade, over all 810,000 three-digit sums at $T{=}0$. At three carries on the 7B, sums without a cascade fail **eight** times in 91,125 ($8.8{\times}10^{-5}$), a $1989\times$ ratio against sums with one.

First, by construction the two are not the same feature. `max_carry_chain_9` keys on columns whose *digit pair sums to nine* ($4{+}5$, $3{+}6$, $8{+}1$), not on literal nine digits, so a cascade input need contain no nine anywhere. Empirically the coupling is weak: over the full enumeration, cascade inputs carry 0.712 literal nines on average against 0.613 for no-cascade inputs, a difference of $+0.098$ digits.

Second, the effect survives holding the literal nine count fixed. Table 30 stratifies the three-carry comparison by `count_9s` (the number of literal nines across both operands) and reports a Mantel–Haenszel risk ratio over the strata. Stratifying does not weaken the effect: it strengthens it, to $2{,}279\times$ on the 7B from an unstratified $1{,}989\times$. The decisive stratum is `count_9s`$=0$, where neither operand contains a single nine: there the 7B fails on 1 of 21,952 no-cascade inputs against 2,339 of 14,336 cascade inputs, a $3{,}582\times$ ratio. A tokenizer-pattern explanation has to account for an effect that remains when the pattern is absent.

Model	`count_9s`	no cascade: fails/n	cascade: fails/n	ratio
7B	0	1/21,952	2,339/14,336	$3{,}582\times$
	1	1/37,632	3,058/17,376	$6{,}623\times$
	2	6/23,856	1,360/7,104	$761\times$
	3	0/6,784	227/1,152	—
Mantel–Haenszel, 7B, 3 carries, `count_9s` *held fixed*				$\mathbf{2{,}279}\times$
Mantel–Haenszel, 3B, 3 carries, `count_9s` *held fixed*				$\mathbf{14.6}\times$
Mantel–Haenszel, 7B, 2 carries, `count_9s` *held fixed*				$\mathbf{35.9}\times$
Mantel–Haenszel, 3B, 2 carries, `count_9s` *held fixed*				$\mathbf{5.5}\times$

Table 30: Cascade versus no cascade at three carries, stratified by the number of literal nine digits, over the complete enumeration. The unstratified ratios are $1{,}989\times$ (7B) and $13.3\times$ (3B); holding `count_9s` fixed moves them to $2{,}279\times$ and $14.6\times$. The effect is present even in the stratum containing no nines at all ($3{,}582\times$). Rows shown are `count_9s`$\in\{0, 1, 2, 3\}$, which is 90,224 of the 91,124 no-cascade and 39,968 of the 40,050 cascade inputs; the Mantel–Haenszel ratio runs over all six matched strata, adding `count_9s`$=4$ (0/852 against 10/80) and $=5$ (0/48 against 0/2), both of which have zero no-cascade failures and so cannot be displayed as a finite ratio. A stratum is matched when both arms are non-empty, which drops one no-cascade input relative to the 91,125 of Table 29.

The stratified control uses the enumeration because it records every operand pair; having no sampling error, it is also the stronger test.

Scope. The enumeration runs at $T{=}0$ while every search runs at $T{=}1.0$, so the recall of a search against this failure set would not be a like-for-like comparison. Width is fixed at 3, so operand-order effects (Appendix J) cannot appear in the enumeration.

G.1 Carry cascades in frontier models

The designed contrast of Table 28 is repeated on the five frontier models at the low reasoning setting, at a per-model operand width (Table 31). The effect replicates on GPT-5.6 Luna, Claude Sonnet 5 and DeepSeek v4.1 Flash: cascades of four or five columns raise the failure rate by 14.0 to 38.0 points, with intervals that exclude zero, while one- or two-column cascades change it by at most 4 points on every model. Gemini 3.8 Flash shows no effect that survives the Holm correction at any cascade length. On GLM-5.3 the arms differ at $k = 0$, where they are built identically, by -14.0 points (Newcombe interval $[-20.4, -7.5]$), so the GLM-5.3 contrast is not interpretable; its gap at $k = 5$ ($+22.0$) is not counted as a replication. Luna's contrast uses 1,000 inputs per arm at the `low` reasoning setting; the other four use 300 inputs per arm.

Model (operand width, inputs per arm)	k	cascade arm, %	no-cascade arm, %	gap, points
GPT-5.6 Luna (width 40, 1,000)	0	19.5 [17.2, 22.1]	19.6 [17.3, 22.2]	−0.1 [−3.6, +3.4]
	1	21.1 [18.7, 23.7]	18.9 [16.6, 21.4]	+2.2 [−1.3, +5.7]
	2	19.5 [17.2, 22.1]	18.7 [16.4, 21.2]	+0.8 [−2.6, +4.2]
	3	21.3 [18.9, 23.9]	18.6 [16.3, 21.1]	+2.7 [−0.8, +6.2]
	4	33.5 [30.6, 36.5]	17.0 [14.8, 19.5]	+16.5 [+12.7, +20.2]
	5	53.9 [50.8, 57.0]	16.9 [14.7, 19.3]	+37.0 [+33.1, +40.8]
Claude Sonnet 5 (width 32, 300)	0	12.0 [8.8, 16.2]	8.0 [5.4, 11.6]	+4.0 [−0.8, +8.9]
	1	8.0 [5.4, 11.6]	8.0 [5.4, 11.6]	+0.0 [−4.4, +4.4]
	2	10.0 [7.1, 13.9]	8.0 [5.4, 11.6]	+2.0 [−2.6, +6.7]
	3	16.0 [12.3, 20.6]	8.0 [5.4, 11.6]	+8.0 [+2.8, +13.2]
	4	28.0 [23.2, 33.3]	8.0 [5.4, 11.6]	+20.0 [+14.0, +25.9]
	5	20.0 [15.9, 24.9]	6.0 [3.8, 9.3]	+14.0 [+8.7, +19.4]
DeepSeek v4.1 Flash (width 20, 300)	0	2.0 [0.9, 4.3]	6.0 [3.8, 9.3]	−4.0 [−7.5, −0.8]
	1	7.0 [4.6, 10.5]	6.0 [3.8, 9.3]	+1.0 [−3.1, +5.1]
	2	10.0 [7.1, 13.9]	7.0 [4.6, 10.5]	+3.0 [−1.5, +7.6]
	3	18.0 [14.1, 22.7]	8.0 [5.4, 11.6]	+10.0 [+4.6, +15.4]
	4	34.0 [28.9, 39.5]	8.0 [5.4, 11.6]	+26.0 [+19.7, +32.1]
	5	38.0 [32.7, 43.6]	0.0 [0.0, 1.3]	+38.0 [+32.5, +43.6]
GLM-5.3 (width 24, 300)	0	14.0 [10.5, 18.4]	28.0 [23.2, 33.3]	−14.0 [−20.4, −7.5]
	1	24.0 [19.5, 29.1]	24.0 [19.5, 29.1]	+0.0 [−6.8, +6.8]
	2	21.0 [16.8, 26.0]	20.0 [15.9, 24.9]	+1.0 [−5.5, +7.5]
	3	19.0 [15.0, 23.8]	16.0 [12.3, 20.6]	+3.0 [−3.1, +9.1]
	4	18.0 [14.1, 22.7]	12.0 [8.8, 16.2]	+6.0 [+0.3, +11.7]
	5	26.0 [21.4, 31.2]	4.0 [2.3, 6.9]	+22.0 [+16.6, +27.5]
Gemini 3.8 Flash (width 30, 300)	0	4.0 [2.3, 6.9]	8.0 [5.4, 11.6]	−4.0 [−8.0, −0.2]
	1	7.0 [4.6, 10.5]	7.0 [4.6, 10.5]	+0.0 [−4.2, +4.2]
	2	6.0 [3.8, 9.3]	6.0 [3.8, 9.3]	+0.0 [−3.9, +3.9]
	3	7.0 [4.6, 10.5]	5.0 [3.1, 8.1]	+2.0 [−1.9, +6.0]
	4	8.0 [5.4, 11.6]	4.0 [2.3, 6.9]	+4.0 [+0.2, +8.0]
	5	6.0 [3.8, 9.3]	10.0 [7.1, 13.9]	−4.0 [−8.5, +0.4]

Table 31: Carry-cascade contrast on frontier models at the low reasoning setting. Both arms hold operand width and the number of carries (k+1) fixed; in the cascade arm the carries propagate through k consecutive columns whose digits sum to nine, in the no-cascade arm they do not. Rates with 95% Wilson intervals, gap with a Newcombe interval, both unadjusted for multiple comparisons. A model counts as showing the effect at length k when a two-sided Fisher test, Holm-corrected across its six lengths, rejects equal rates; by that rule Luna, Sonnet 5 and DeepSeek show it at $k \geq 4$ (Sonnet 5 and DeepSeek also at $k = 3$) and Gemini at no length. At $k = 0$ the arms are built identically, so a difference there is an imbalance between the arms; after correction only GLM-5.3 has one, and its contrast is set aside.

H Reasoning settings and character positions on frontier models

Reasoning. Table 32 gives the comparison of Section **??** with intervals. The lowest setting is `none` for GPT-5.6 Luna, off for Claude Sonnet 5 and DeepSeek v4.1 Flash, and minimal for Gemini 3.8 Flash and GLM-5.3; the high setting is each API's high effort. Because Gemini and GLM reason at every setting, their lowest-setting rates already include some reasoning, and a comparison across models of the size of the change would mix different amounts of it; we therefore read the table within each model.

Character positions. The substring task asks for the characters between two positions (0-indexed, end exclusive) of a string of four to eight characters. The listed-positions prompt is the same question followed by every character of the string paired with its position. Table 33 gives the three conditions with intervals; the listing is applied at the lowest setting, so any effect it has comes from the input format rather than from reasoning.

Model (lowest setting)	Task	lowest, %	high, %	change, points
GPT-5.6 Luna (none)	31–34-digit addition	36.7 [31.4, 42.3]	10.0 [7.1, 13.9]	−26.7 [−33.0, −20.1]
	10-digit multiplication	100.0 [98.7, 100.0]	3.3 [1.8, 6.0]	−96.7 [−98.2, −93.7]
	4–8-character string edits	13.3 [9.9, 17.6]	3.3 [1.8, 6.0]	−10.0 [−14.6, −5.7]
Gemini 3.8 Flash (minimal)	31–34-digit addition	24.0 [19.5, 29.1]	0.0 [0.0, 1.3]	−24.0 [−29.1, −19.3]
	10-digit multiplication	8.0 [5.4, 11.6]	0.0 [0.0, 1.3]	−8.0 [−11.6, −5.1]
	4–8-character string edits	3.0 [1.6, 5.6]	0.0 [0.0, 1.3]	−3.0 [−5.6, −1.1]
Claude Sonnet 5 (off)	31–34-digit addition	15.0 [11.4, 19.5]	10.0 [7.1, 13.9]	−5.0 [−10.3, +0.3]
	10-digit multiplication	32.0 [27.0, 37.5]	6.7 [4.4, 10.1]	−25.3 [−31.3, −19.3]
	4–8-character string edits	21.0 [16.8, 26.0]	3.3 [1.8, 6.0]	−17.7 [−22.9, −12.7]
GLM-5.3 (minimal)	31–34-digit addition	36.3 [31.1, 41.9]	26.7 [22.0, 31.9]	−9.7 [−17.0, −2.2]
	10-digit multiplication	18.2 [14.2, 22.9]	6.7 [4.4, 10.1]	−11.5 [−16.8, −6.3]
	4–8-character string edits	16.0 [12.3, 20.6]	0.0 [0.0, 1.3]	−16.0 [−20.6, −12.1]
DeepSeek v4.1 Flash (off)	31–34-digit addition	52.0 [46.4, 57.6]	0.0 [0.0, 1.3]	−52.0 [−57.6, −46.2]
	10-digit multiplication	98.0 [95.7, 99.1]	0.0 [0.0, 1.3]	−98.0 [−99.1, −95.4]
	4–8-character string edits	25.0 [20.4, 30.2]	0.0 [0.0, 1.3]	−25.0 [−30.2, −20.3]

Table 32: Lowest against high reasoning, 600 answers per cell (300 inputs, each answered twice), with 95% Wilson intervals for rates and Newcombe intervals for the change (high minus lowest), both computed over inputs. GPT-5.6 Luna's lowest setting here is `none`; at its `low` setting it fails 7.7% of 31–34-digit additions (Table 10), so raising its effort from low to high does not lower that rate.

Model	plain, lowest	positions listed, lowest	plain, high	effect of listing, points
GPT-5.6 Luna	76.7 [71.6, 81.1]	43.3 [37.8, 49.0]	26.7 [22.0, 31.9]	−33.3 [−40.4, −25.7]
Gemini 3.8 Flash	3.3 [1.8, 6.0]	3.3 [1.8, 6.0]	3.3 [1.8, 6.0]	+0.0 [−3.1, +3.1]
Claude Sonnet 5	69.0 [63.6, 74.0]	13.3 [9.9, 17.6]	10.0 [7.1, 13.9]	−55.7 [−61.7, −48.7]
GLM-5.3	30.0 [25.1, 35.4]	43.3 [37.8, 49.0]	10.0 [7.1, 13.9]	+13.3 [+5.6, +20.8]
DeepSeek v4.1 Flash	80.0 [75.1, 84.1]	30.0 [25.1, 35.4]	3.3 [1.8, 6.0]	−50.0 [−56.4, −42.7]

Table 33: Substring extraction on strings of four to eight characters, 600 answers per cell (300 inputs, each answered twice), with 95% Wilson intervals for rates and a Newcombe interval for the effect of listing positions (listed minus plain, both at the lowest setting), computed over inputs.

I FINE-TUNING ON THE CARRY-CASCADE RULE

This experiment intervenes rather than measures. If a rule identifies a real property of a model, it should be possible to repair the region it identifies, and the repair may cost something elsewhere. We test the smallest version of this on the region of the carry-cascade rule (Appendix F.1; mechanism in Appendix G).

Design. The experiment has three arms. The targeted arm fine-tunes Qwen2.5-3B with LoRA at rank 16 on all seven projections, on 8,000 carry-cascade inputs at operand widths 4 to 8 for one epoch, and merges the adapter back into the base weights; the two control arms (Appendix I.1) differ from it only in which problems they train on. The evaluation is the complete enumeration of all 810,000 three-digit sums at $T{=}0$, which is *disjoint from the training widths by construction*: three-digit sums are kept out of training because they are the test set. The question is therefore not whether the model can memorize sums it was shown, but whether repairing a mechanism at one set of widths reaches a width it never saw. Because every model is scored at $T{=}0$ on the same complete population, these rates carry no sampling error and no interval is needed; every input can also be followed individually across models.

The region closes, at a width the training never touched. On the 17,100 sums covered by the rule the region search returned, `max_carry_chain_9 > 0.5 AND leading_digit_A <= 2.5`, the failure rate falls from 0.2919 to 0.0041, a 71× reduction. At matched carry count the mechanism contrast collapses with it: at two carries, cascade inputs go from 0.2115 to 0.0010 while their no-cascade counterparts go from 0.0376 to 0.0127, so the 5.6× cascade lift of Appendix G

becomes $0.1\times$. The region is a real and addressable target, and the repair generalizes across operand width.

Fixed and broken inputs, by carry count. Table 3 gives the two flows as totals, 35,161 inputs fixed against 15,773 broken. Because both models are scored at $T{=}0$ on the same complete population, the decomposition is exact, and every input can be located as well as counted (Table 34). Fixes concentrate at two and three carries, where the training was aimed, and breakage at zero and one, where the model had least to gain.

Carries	n	Fixed	Newly broken	Failure rate before $\rightarrow$ after
0	108,900	1,852	5,376	$0.0252 \rightarrow 0.0575$
1	278,325	8,188	7,385	$0.0362 \rightarrow 0.0333$
2	291,600	16,225	2,950	$0.0569 \rightarrow 0.0114$
3	131,175	8,896	62	$0.0682 \rightarrow 0.0009$
all	**810,000**	**35,161**	**15,773**	$0.0473 \rightarrow 0.0234$

Table 34: Fixed and broken inputs for the targeted arm, by carry count. Zero-carry sums, the easiest inputs, more than double their failure rate while the intervention halves the failure count overall. On a sampled benchmark these two flows are not separable from their net.

The repair reaches widths far outside the training range, and carries a forgetting cost on other tasks. The ADD failure rate falls at every level, including 9–10 and 11–12, which are well beyond the widths trained on, so this is transfer rather than memorization. On other tasks it is not free: the GRAPH failure rate rises by 0.13 and MUL at 2–3 digits by 0.03 (Table 35).

Task	Level	Before	After	Change
ADD	3–4 (width 4 trained)	0.092	0.042	-0.050
ADD	5–6 (trained)	0.282	0.103	-0.178
ADD	7–8 (trained)	0.458	0.165	-0.293
ADD	9–10 (not trained)	0.530	0.217	-0.313
ADD	11–12 (not trained)	0.710	0.402	-0.308
MUL	2–3	0.585	0.612	$+0.027$
MUL	4–5	0.997	0.997	$+0.000$
GRAPH	all	0.417	0.548	$\mathbf{+0.132}$

Table 35: Failure rate before and after the intervention, 300 inputs at two answers per level. The saturated MUL levels at 6–7 and 8 digits sit at 1.000 both before and after and are omitted. GRAPH, which the training never touched, is where the repair is paid for.

I.1 The two control arms

The control arms separate the effect of *targeting* from the effect of fine-tuning at all, since a LoRA at this rank for one epoch might damage zero-carry sums whatever it trains on. They match the targeted arm in rank, schedule, number of training inputs and training widths, and differ only in which 8,000 inputs they see. The *uniform* arm draws inputs uniformly at widths 4–8. The *zero-carry* arm draws only inputs with no carries at all, the easiest sums, and tests whether the trade runs in the other direction. Final training losses are 0.028, 0.021 and 0.00003 for the targeted, uniform and zero-carry arms respectively; the last is near zero because zero-carry addition is column-wise independent and one epoch suffices to learn it.

The degradation is the targeting, not the recipe. Zero-carry failures rise from 0.0252 to 0.0575 under the targeted arm and *fall* to 0.0043 under the uniform control. Of the inputs the targeted arm newly breaks, 34% are zero-carry sums, against 7% for the uniform control and a 13% share among all three-digit sums. The collateral damage reported above is caused by aiming at the region.

The uniform control leaves the better model. At the same budget, uniform training reaches 0.0119 on all three-digit sums against the targeted arm's 0.0234, breaks 4,769 inputs against 15,773, and damages GRAPH less ($+0.087$ against $+0.132$). It reaches a lower ADD failure rate at every width

level. Its one loss is the thing it was not aimed at: it takes the target region only to 0.120, where targeting reaches 0.004. If the goal is that region, targeting wins by $30\times$; if the goal is the model, it loses.

Reallocation runs both ways, and costs far more when training on the easiest sums. The zero-carry arm becomes perfect on what it trained on (0.0000) and much worse everywhere else: the target region goes to 0.719, cascade inputs to 0.697, and all three-digit sums from 0.047 to 0.271, breaking $194{,}316$ inputs against $13{,}062$ fixed. Training on the easiest sub-distribution bought perfection there at the price of the general algorithm.

Scope of the repair experiment. The region identified by the carry-cascade rule is specific enough to be fixed on purpose by 8,000 examples at other widths, and a matched control closes it $30\times$ less, so the targeting does the work. Together with the designed contrast of Appendix G, this is strong evidence that the carry cascade the rejected candidate describes is a property of the model rather than of the search. The cost is structured, caused by the targeting, and invisible to aggregate accuracy, which ranks these models 97.7%, 98.8% and 72.9% and prefers the uniform control on the one axis it measures. The experiment covers one model, one region, one rank and one schedule, and it does not argue for targeted repair in general: to obtain a better model overall, training on the distribution of interest is the better choice, and targeting is for when a specific region must be closed.

J Operand order across model sizes

Model	2nd operand longer	1st operand longer	difference	Fisher p	inputs
1.5B	0.775	0.790	-0.015	0.029	15,487
3B	0.658	0.454	$+0.204$	9×10^{-145}	15,487
7B	0.639	0.341	$+0.298$	4×10^{-306}	15,487
14B	0.274	0.263	$+0.011$	0.12	15,146

Table 36: Operand-order asymmetry on uniform samples, widths 5–8. Negligible at 1.5B, peaks at 7B, not significant at 14B (also Figure **??**b).

Swapping operand order inverts the effect on both models where it exists (3B $+0.18 \to -0.25$; 7B $+0.29 \to -0.28$; measured on the swap experiment's own sample, so the baseline differs slightly from Table 36), and the effect survives zero-padding the shorter operand to equal width (3B $+0.14$; 7B $+0.24$), so it is positional (worse when the longer number is written second) rather than a digit-alignment or tokenizer artefact. It is non-monotone in scale: negligible at 1.5B (which fails on most unequal pairs regardless of order), largest at 7B, and gone at 14B, so a deployment mitigation effective at 7B (writing the longer operand first) buys nothing at 14B.

K Evaluations to first failure

Coverage (Section **??**) counts distinct failing bins reached at budget. A second notion of search efficiency is discovery latency: how many evaluations an algorithm spends before its first failure (ETFF). We compute ETFF for all eleven cells of Table 18, ten seeds each unless Table 38 states otherwise, for `RAND`, `HC`, `ME` and `LLM-REF`, from each input's position in its own (algorithm, objective, seed) evaluation sequence, and compare it exactly as coverage is compared: per-seed values, a Vargha–Delaney $\hat{A}_{12}$ against `RAND` (here $\hat{A}_{12} > 0.5$ means the algorithm reaches a failure sooner than `RAND` more often), a two-sided Mann–Whitney test, and Holm correction across the full family of comparisons.

Calibration of the first-failure statistic. For a memoryless proposer such as `RAND`, the expected ETFF is $1/\pi$. Table 37 shows the ratio of `RAND`'s mean ETFF to $1/\pi$ across all eleven cells; every ratio falls in $[0.71, 1.25]$, consistent with sampling noise at five to ten seeds. No seed in any cell was censored (every algorithm reached a failure within budget every time), so no survival-analysis correction is needed.

Model	Cell	RAND mean ETFF	$1/\pi$ (theory)	ratio
1.5B	ADD 3–6	1.40	1.97	0.71
1.5B	ADD 3–8	1.40	1.61	0.87
3B	ADD 3–8	2.30	2.12	1.09
3B	ADD 3–6	3.60	3.39	1.06
3B	ADD 3–5	4.90	4.65	1.05
3B	GRAPH	2.40	2.53	0.95
7B	ADD 3–8	3.40	2.72	1.25
7B	GRAPH	1.90	2.60	0.73
7B	MUL 2–3	1.70	2.07	0.82
14B	ADD 5–8	3.30	4.13	0.80
14B	ADD 5–8 (replicate)	3.40	4.13	0.82

Table 37: Calibration of the first-failure statistic: RAND's mean ETFF against the value implied by the cell's own measured failure rate. All eleven ratios cluster around 1.0.

Result. Table 38 gives the full comparison. No algorithm shows a significant advantage over RAND on evaluations to first failure in any cell after Holm correction (every corrected $p = 1.000$; the smallest raw p is 0.155). Aggregated across cells, HC (mean $\hat{A}_{12} = 0.451$ over 10 cells, favouring RAND in 7 of 10) and ME (mean $\hat{A}_{12} = 0.438$ over 11 cells, favouring RAND in 8 of 11) trend slightly *towards* RAND being faster, not slower, though neither trend is significant; LLM−REF is indistinguishable from parity (mean $\hat{A}_{12} = 0.481$).

			RAND	HC			ME			LLM−REF		
Model	Cell	n	med/mean	med/mean	$\hat{A}_{12}$	p_{Holm}	med/mean	$\hat{A}_{12}$	p_{Holm}	med/mean	$\hat{A}_{12}$	p_{Holm}
1.5B	ADD 3–6	10	1/1.40	1/2.00	0.415	1.000	2/2.40	0.355	1.000	1/1.60	0.440	1.000
1.5B	ADD 3–8	10	1/1.40	1/1.50	0.495	1.000	1/1.50	0.455	1.000	1/1.20	0.510	1.000
3B	ADD 3–8	10	1/2.30	2/6.30	0.445	1.000	2/2.50	0.430	1.000	1/2.00	0.515	1.000
3B	ADD 3–6	10	3/3.60	3/7.20	0.450	1.000	3/4.00	0.425	1.000	3/3.50	0.505	1.000
3B	ADD 3–5	10	6/4.90	4/4.60	0.540	1.000	6/6.20	0.445	1.000	4/5.00	0.530	1.000
3B	GRAPH	5	3/2.40	2/2.60	0.520	1.000	3/2.40	0.500	1.000	3/2.60	0.460	1.000
7B	ADD 3–8	10	2/3.40	4/10.80	0.385	1.000	4/4.00	0.440	1.000	4/3.40	0.415	1.000
7B	GRAPH	10	1/1.90	1/2.90	0.510	1.000	1/1.80	0.540	1.000	1/2.10	0.480	1.000
7B	MUL 2–3	10	1/1.70	1/2.10	0.440	1.000	1/1.70	0.500	1.000	1/1.70	0.500	1.000
14B	ADD 5–8	10	4/3.30	8/7.00	0.310	1.000	4/4.50	0.410	1.000	4/3.50	0.455	1.000
14B	ADD 5–8 (replicate)	5	4/3.40	—	—	—	4/4.00	0.320	1.000	—	—	—

Table 38: Evaluations to first failure in the eleven cells of Table 18, ten seeds (five for the 3B GRAPH cell and the 14B replicate). $\hat{A}_{12}$ is computed on the per-seed ETFF values against RAND; p_{Holm} is Holm-corrected across all 31 (cell, algorithm) comparisons. The 14B replicate compares ME with RAND only.

Summary. Discovery latency and coverage (Section **??**) are two distinct notions of search efficiency this framework can measure, and on the cells tested neither shows a reliable adaptive-search advantage over uniform sampling.

L Recorded failure examples

Failures on other tasks have the same character: on MUL the 7B answers $309{,}050 \times 185{,}526$ with $574{,}004{,}776{,}200$ against a true $57{,}336{,}810{,}300$ (one digit too many); on GRAPH, failures are confident but wrong reachability or shortest-path claims rather than malformed output. Unparseable answers are rare for most models (Appendix B.2 lists the exceptions), so the saturated tasks cannot be diagnosed by output validity alone: they fail on content.

A	B	correct sum	model's answer	len. diff.	carries	error
65,067,102	5,397,684,345	5,462,751,447	5,462,751,44**9**	-2	4	final digit only
642,957	3,059,502	3,702,459	3,**69**2,459	-1	3	two digits
2,459,495,681	46,147,379,844	48,606,875,525	48,**597**,875,**6**25	-1	6	four digits
16,177	945,084,843	945,101,020	945,10**0**,**917**	-4	5	four digits
25,288,357,860	8,776	25,288,366,636	25,28**9**,**235**,**52**6	$+7$	3	six digits
514	73,459,262,898	73,459,263,412	73,459,**314**,**283**	-8	3	six digits
706,645,816	9,281	706,655,097	706,**738**,**62**7	$+5$	2	five digits
1,605,557,743	7,188,656,569	8,794,214,312	**7**,**8**94,21**2**,312	0	7	three digits

Table 39: Qwen2.5-3B on uniformly drawn additions, $T{=}1.0$, direct-answer prompt; bold marks the digits that differ from the correct sum. Every answer has the correct digit count. Eleven of the twelve wrong answers in the full sample have a nonzero length difference, the region the search returns.

M COMPARISON WITH PRIOR METHOD FAMILIES

Table 40 compares this work with slice discovery dimension by dimension, and Table 41 summarizes the comparison across method families.

Dimension	Slice discovery	This work
Starting data	fixed labelled dataset	executable task generator
Where errors come from	already present in the dataset	actively searched for
Input validity	determined by the dataset	enforced by the generator
Ground truth	dataset labels	exact task oracle
Representation	learned feature space	interpretable generator features and a declared metric
Search	among observed examples	over the entire valid input space
Region	a slice of the existing dataset	a connected region of the input space
Region description	feature cluster or caption	executable rule ϕ_R
Validation on new examples	not part of discovery	required for acceptance
Complement certification	none	interval bounds on sub-cell failure rates (Appendix **??**)

Table 40: Slice discovery compared to this work, dimension by dimension.

Family	Evidence	Validity	Output	Complement claim
Slice discovery	fixed dataset	labels given	cluster + caption	none
Failure-mode discovery	search, judged	partial	embedding cluster	none
Benchmark synthesis	generated	LLM judge	harder tasks	none
Self-play generation	generated	learned judge	training tasks	none
Red-teaming	search, judged	none	adversarial prompts	none
Code-execution benchmarks	fixed test suite	unit tests (exact)	pass/fail per problem	none
Active testing / ability estimation	selection from a fixed, scored item bank	item bank pre-labelled	scalar ability estimate	none (not the target quantity)
This work	search, exact oracle	by construction	region, rule, mass	interval bounds on sub-cell failure rates (Appendix **??**)

Table 41: Summary comparison across method families (extends Section 2).