THE NOISE FLOOR OF LLM PROGRAM DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Systems such as FunSearch, AlphaEvolve and ShinkaEvolve put a language model inside a search loop and report gains over plain sampling, usually from one or a few runs. We ask whether those gains are real at the budget most practitioners have, a few hundred LLM calls per problem. Holding the model, the tasks, the call budget and the final selection rule fixed, we compare best-of-N sampling with two island-evolution methods, reflect-then-sample and an EXP3 router on fifteen verified tasks, with 10 to 20 independent seeds per method and task. The result is a noise floor. At 100 calls no method beats sampling by more than 1.03 seed standard deviations on any task, and reflection and adaptive allocation never win; only FunSearch-style evolution has a small, consistent edge. At 300 calls that edge grows large on some instances and turns negative on others, and a pre-registered ladder of covering designs shows that distance from the published record does not predict which. What does move the floor is the model: with a weak 7B model whose first-shot programs are often broken, evolution repairs them and beats sampling clearly, and still ties where the model plateaus. The seed count matters as much as the method. With three seeds per method, the norm in this literature, our own data would have missed four of five effects the full analysis detects and, in one method pair in nine, named a winner it does not support. We therefore report every tie with its minimum detectable effect, every win with its split-half replication rate, and every prediction as written before the data came in, including the ones that failed.

1 INTRODUCTION

A growing family of systems attacks research-level construction problems by putting a language model inside a search loop (Romera-Paredes et al., 2024; Novikov et al., 2025; Lange et al., 2025). All of them combine the same three parts, a model that proposes programs, an evaluator that scores them, and a population that keeps and recombines the good ones, but which of the three does the work remains unsettled. Gideoni et al. (2026) find that plain sampling baselines match code-evolution pipelines on three domains, mathematical bounds, agentic scaffolds and machine-learning competitions, and attribute the performance ceiling to the search space and the prompt rather than to the search. The Mind Evolution authors (Lee et al., 2025) credit their gains to the evaluator's feedback being fed back into generation, and the FunSearch authors note that their method needs a fine-grained numerical score, rather than a pass-or-fail signal, before evolution can make progress. Xing et al. (2026) find that how a fixed budget of calls is split between successive generations and parallel children changes the outcome, while also noting that the field's dominant convention is to report the best of an unspecified number of runs.

We take this question to the budget regime in which most practitioners actually work. FunSearch's size-512 cap set in dimension 8 was found by only four of 140 independent experiments, from a total number of samples the authors put "on the order of 10^6" (Romera-Paredes et al., 2024), so the record is a rare event at a budget beyond what most practitioners can afford. The question we ask is not whether sampling is competitive (Gideoni et al., 2026; Olausson et al., 2024; Hassid et al., 2024), but how much of what gets reported at these budgets is noise, and what survives once the distribution is measured rather than sampled once.

To find out, we ran six methods on fifteen verified tasks with one model, a uniform final selection rule and a fixed request budget, using ten to twenty independent seeds per cell at $N{=}100$ and ten seeds per cell for four of the methods at $N{=}300$. We then added three pre-registered extensions: two

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.

solver tasks from outside the construction family; a ladder of six covering-design instances, tasks in which every triple of points must be covered by as few fixed-size blocks as possible, running from near-saturated to far-from-record, built to test whether headroom predicts where evolution pays; and a rerun of four tasks with two further models, the mid-strength gpt-4.1-mini and the weak open-weight Qwen2.5-7B, built to test whether the ties reflect the small budget itself or a model that has already reached its own plateau within it.

The study yields three results. The first establishes what we name the noise floor, meaning the size of the gap between two methods that seed-to-seed variation alone can produce. We measure every gap in units of the seed standard deviation, the spread of one method's final scores across its independent runs on a task. At 100 calls, on any of the 59 method-versus-sampling comparisons in the study, no method beats sampling by more than 1.03 seed standard deviations; the two reflection variants, which pause to distil insights from earlier attempts before sampling again, never win; the adaptive router, which reallocates calls among the other methods as it goes, ties sampling everywhere; and the spread between method means is a third of the spread within a method. Because every method was run twenty times on every task in the main grid, a gap of 0.9 seed standard deviations would have been detected with 80% power, so these ties are informative rather than inconclusive.

The second result is that one method stands apart without escaping that bound. FunSearch-style evolution, which keeps a population of programs in separate pools and asks the model to improve on pairs of the better ones, beats sampling by about one seed standard deviation on four of fifteen tasks. At 300 calls its edge grows to two seed standard deviations on one covering design, a task in which every triple of points must be covered by as few fixed-size blocks as possible, and to a full extra codeword on one error-correcting code, and both gains reappear when the seeds are split at random into two halves and each half is analysed alone, which is our test of whether a win is repeatable. On a larger covering design the same method loses.

The third result identifies what determines where evolution pays. The natural explanation, that evolution pays where there is room to improve, was written down in advance and failed, because along a ladder of covering designs of increasing size the advantage peaks at moderate distance from the best known construction and turns negative at the largest. What moves the floor is the model. With Qwen2.5-7B, whose first attempts at a program often fail to run, both evolutionary methods, FunSearch and a lighter single-parent variant we call Evolve, beat sampling by one to one and a half seed standard deviations on the two tasks where those attempts are poor, and tie on the two where the weaker model reaches the same score in nearly every run, exactly as they do for DeepSeek. The mid-strength gpt-4.1-mini sits between the two: it matches DeepSeek on MAX-3-SAT and the two plateau tasks, where everything ties, and falls short of it only on the largest covering design, where evolution wins by about one seed standard deviation. The same data also carry a lesson about reporting, because re-analysed with three seeds per method, the norm in this literature, the analysis misses four of five effects the full analysis detects and, in one method pair in nine, names a winner it does not support.

Contributions.

1. A multi-seed benchmark of six budget-constrained search methods on fifteen verified tasks at 100 and 300 calls, with 10 to 20 independent runs per method and task, a uniform final selection rule, and the same protocol repeated on two further models from two other providers (§3).
2. Three statistics that comparisons in this literature rarely report, namely the fraction of method pairs that remain distinguishable after correcting for multiple comparisons, the rate at which a win replicates across disjoint halves of the seeds, and the verdict a study with k seeds per method would have reached (§4).
3. Pre-registered predictions for two budgets and three extensions, reported with their outcomes, including the refutation of the hypothesis that reflection helps wherever a graded ladder of constructions exists, the refutation of the headroom hypothesis, and the cross-model test that separates model strength from budget (§4.6, §4.8, §4.9).
4. A reporting recommendation for method comparisons at these budgets, covering seed counts, the minimum detectable effect beside every tie, split-half replication beside every win, an equivalence margin for any claim of matching a baseline, and re-verification of every reported construction (§5).

id	task	score	note
t01	Sidon set	set size	best found 30, record 32
t02	Low-autocorrelation binary sequence	merit factor	continuous
t03	K_4-free Ramsey colouring	$-$(monochromatic K_4)	seed SD ≈ 8
t04	Circle packing	$-$(excess radius)	continuous
t07	q-ary code $A_2(21, 10)$	code size	near-saturated at 40
t08	q-ary code $A_3(13, 8)$	code size	best found 35, record 54–95
t09, t11	Covering designs $C(13, 6, 3)$, $C(24, 6, 3)$	$-$blocks	records 21, 116
t10	Covering design $C(20, 12, 3)$	$-$blocks	saturated at 10
t12	MAX-3-SAT, random instances	satisfied clauses	solver task
t13	Planted clique	edge score	solver task; seed SD 19
t21–t24	$C(v, 6, 3)$, $v \in \{16, 20, 28, 32\}$	$-$blocks	ladder; records 38, 71, 185, 300
t05, t06	3-AP-free set; run-length-limited code		excluded (saturated; verifier defect)

Table 1: Tasks. Covering records are La Jolla Covering Repository sizes (Gordon, 2026).

2 Related Work

LLMs in search loops. FunSearch (Romera-Paredes et al., 2024) evolves a scoring function inside a fixed skeleton with islands and softmax parent sampling, using millions of samples in production; AlphaEvolve (Novikov et al., 2025) extends this to whole codebases; ShinkaEvolve (Lange et al., 2025) adds an archive, LLM-judged novelty and bandit model selection and reports a circle-packing record from about 150 samples; Mind Evolution (Lee et al., 2025) evolves plans and credits critic feedback. Gideoni et al. (2026) find iid and sequential sampling competitive with these pipelines on bounds, scaffolds and ML competitions and attribute the ceiling to the search space and the prompt; our study is the controlled version of that observation with the noise characterised. Xing et al. (2026) split a budget of $C{=}T{\times}N$ calls between generations and children on three geometric tasks with Qwen3 (1.7B to 14B) and Llama-3.1-8B at ten seeds, propose a bandit over trajectories that gains 12.3% over island protocols, and document that prior work reports the best of an unspecified number of runs. We share the noise observation and the budget range; §5 says why we reach a different conclusion about allocation.

Sampling, reflection, evaluation. Self-consistency (Wang et al., 2023), Tree of Thoughts (Yao et al., 2023), Self-Refine (Madaan et al., 2023) and the self-correction literature (Huang et al., 2024) show that sampling and revision help reasoning and that many revision gains are selection; Large Language Monkeys (Brown et al., 2024) and Snell et al. (2025) quantify coverage under repeated sampling, the $\log N$ climb every method here must beat. Reflexion (Shinn et al., 2023) uses verbal reflection as agent memory; reflect-then-sample conditions fresh samples on a compressed insight and is tested at matched budget with selection controlled. The max-K-armed and extreme bandit settings (Cicirello & Smith, 2005; Carpentier & Valko, 2014) formalise the maximum-seeking objective that mean-seeking routers do not optimise. Deep reinforcement learning went through the same correction (Henderson et al., 2018; Colas et al., 2018). The rliable protocol (Agarwal et al., 2021), which BaSE adopts, standardises bootstrap intervals across seeds; we add statistics specific to method comparison and report the minimum detectable effect so that a null is a bound.

3 Protocol

Tasks. Fifteen tasks with exact verifiers (Table 1): nine combinatorial constructions in the main grid, four further covering-design instances forming a headroom ladder (§4.8), and two solver problems added to test the result outside the construction family (§4.7). The model writes a Python program; a sandboxed evaluator (static ban list, resource limits, wall-clock timeout) runs it, and a timed-out or crashing program is invalid. Scores are larger-is-better; covering designs are scored as minus the block count. Two tasks were excluded: t05 before the run, because every method finds its 210-element construction, and t06 after it, because about half of its cells for every method produced no valid program, which we treat as a task defect and disclose as a post hoc exclusion. Two of the remaining tasks, $A_2(21, 10)$ and $C(20, 12, 3)$, are near-saturated at $N{=}100$; ties there say only that sampling reaches the plateau, and we flag them.

Partitions and selection. Each task exposes four evaluator partitions differing in the random seed of the stress instances (or declared fixed for single-object tasks): *opt*, visible to the search through feedback; *val*, visible to a controller; *sel*, used once to pick the champion; *test*, used once on the champion. Final selection is uniform: the twenty behaviourally distinct candidates with the best *val* score are scored on *sel*, the winner on *test*, and *test* is reported. No method sees *sel* or *test* during search. Every champion is re-verified as a valid object outside the harness.

Methods. All methods share the model (`deepseek-chat`, DeepSeek V4-Flash, temperature 1.0, 3000 completion tokens), task sheet, seed program, sandbox and a budget of exactly N requests charged at dispatch; reflections and mutations are requests like any other. No method uses computational refinement here. *Best-of-N*: N independent samples from the unconditioned prompt. *FunSearch*: a line-by-line port of the released FunSearch search logic (ten islands, signature clustering, tempered softmax over clusters, shorter-program bias within clusters, two programs per prompt, four samples per prompt, island reset every $N/4$ requests), with three declared deviations listed in Appendix B. *Evolve*: a lighter island model (five islands, rank-weighted parent choice, one parent shown with its score and feedback, one improved child requested), included so the evolution result does not rest on one implementation. *Reflect$_h$*, $h \in \{16, 32\}$: unconditioned samples until h have been generated, then one reflection request that shows the model its twelve best distinct attempts and asks for at most five insight bullets and no code; every later sample is conditioned on the current insight alone. The two periods bracket the region where an earlier protocol found the method's gain (Appendix C). *Hedge*: an EXP3 router over four arms (best-of-N, best-of-N with compute, Reflect$_{32}$, island evolution) allocating each batch by multiplicative weights on marginal improvement with a positive weight floor. With compute disabled its second arm degenerates to a duplicate of the first; we ran it as specified and report it as an adaptive-allocation arm with that caveat, not as a faithful reproduction of any published router. Prompts and constants are in Appendix B.

Grid. Stage 1: 6 methods $\times$ 10 tasks $\times$ 20 seeds at N=100. Stage 2: best-of-N, FunSearch, Evolve, Reflect$_{16}$ $\times$ 7 tasks $\times$ 10 seeds at N=300. Stage 3: best-of-N, FunSearch, Evolve, Hedge on the two solver tasks, 20 seeds (best-of-N: 15, Appendix B), N=100; best-of-N, FunSearch, Evolve on the four ladder instances, 10 seeds, N=100 and 300; and best-of-N, FunSearch, Evolve on four tasks with two further models at the same temperature, token cap and prompts, 10 seeds, N=100: `Qwen2.5-7B-Instruct` served locally with vLLM, and `gpt-4.1-mini` through the OpenAI API. Stages 2 and 3 had predictions and decision rules recorded before submission (Appendix D). A seed is an independent run of the whole search; seeds do not pair across methods, so all contrasts are unpaired. Cells are journalled and resumable; a cell that did not spend its full budget is excluded. Seven runs (0.4%) produced no valid program and are excluded from the means, which if anything flatters the method that failed. About 260,000 DeepSeek requests and $300 of API credit in total, plus 12,000 local requests for Qwen and 12,000 OpenAI requests (about $25) for gpt-4.1-mini.

Statistics. For every task, budget and method pair: the difference in mean test score with a 5,000-resample unpaired percentile bootstrap interval, a two-sided bootstrap p with ties at zero counted on both sides, the Mann–Whitney win rate, and the effect size in pooled seed-SD. A pair is *distinguishable* if $p < 0.05$; we report raw and Holm-corrected counts within task. The *minimum detectable effect* (MDE) is the two-sample 80%-power gap at the realised seed count and within-method SD. *Non-inferiority of sampling* at margin δ holds when the upper 95% limit of a method's advantage is below δ. *Replication*: seeds are split at random into disjoint halves 100 times; a win is a pair with $p < 0.05$ in half A, and the replication rate is the fraction with the same sign and $p < 0.05$ in half B. *k-seed verdict*: for $k \in \{3, 5, 8, 10\}$ we draw k seeds per method 100 times and re-test every pair, taking the full-seed verdict as reference: the *inconsistent-win rate* is the fraction of full-seed ties declared wins at k, the *miss rate* the fraction of full-seed wins declared ties. The reference is itself an estimate, so both rates describe agreement with the best analysis we can afford, not with truth. One script produces every number.

4 Results

4.1 At 100 calls, nothing separates from sampling by more than 1.03 seed-SD

Figure 1a and Table 4 (Appendix E) give the 45 contrasts against best-of-N on the nine main-grid tasks. Two are wins: Evolve on Sidon sets ($+0.45$ elements, $[+0.15, +0.70]$, 0.7 seed-SD) and FunSearch on $C(13, 6, 3)$ ($+0.40$ blocks, $[+0.10, +0.65]$, 0.8 seed-SD). Four are losses, two of

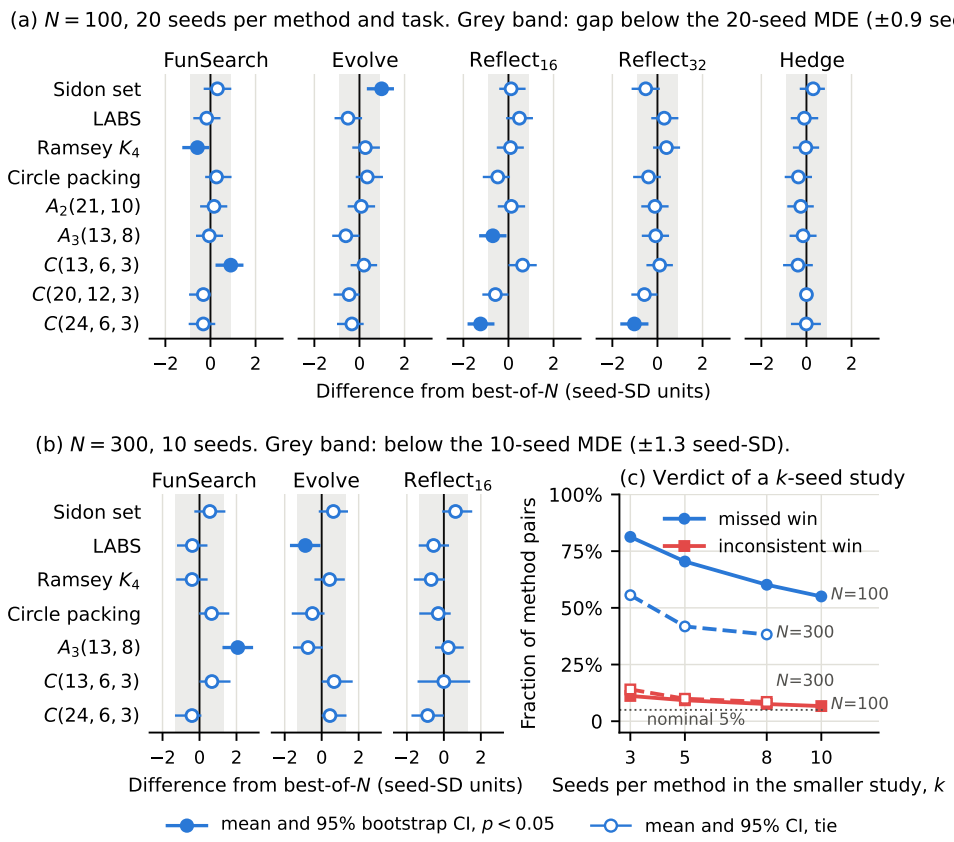


Figure 1: Every method against best-of-N. (a) N=100, 20 seeds per method and task: one column per method, one row per task, showing the mean difference from best-of-N in units of the within-task seed standard deviation with its 95% bootstrap interval. Filled markers have $p < 0.05$, open markers are ties, and the grey band marks gaps smaller than the 20-seed minimum detectable effect. (b) The same at N=300 with 10 seeds. (c) The verdict a study with k seeds per method would have reached, with the full data as reference: the fraction of full-seed effects it misses and the fraction of full-seed ties it declares wins, pooled over tasks; solid lines N=100, dashed N=300.

them large: FunSearch on Ramsey (-9.3), Reflect$_{16}$ on $A_3(13, 8)$ (-1.6), and both reflection arms on $C(24, 6, 3)$ (-15 and -13 blocks). The remaining 39 are ties with MDEs of 0.9 seed-SD. Hedge ties best-of-N on all nine tasks. Non-inferiority of sampling within one seed-SD holds in 39 of 45 contrasts (one-sided $p < 0.025$); the six exceptions are the two wins and four intervals too wide to exclude a one-SD gap, none with a point estimate above 0.85. Within half a seed-SD it holds in 22: twenty seeds certify the one-SD margin, not the half-SD one, which would take about 80 seeds per cell. Adding the six Stage 3 tasks (§4.7, §4.8) brings the count to 59 contrasts with 5 wins, all FunSearch or Evolve, none above 1.03 seed-SD, and 4 losses.

4.2 Noise dominates

Table 2 gives the floor. Between-method SD of the means is 0.13 to 0.56 of within-method seed SD (median 0.36). Of 135 method pairs on the main grid, 33 are distinguishable at $p < 0.05$ and 8 after Holm correction, six of them a reflection arm on $C(24, 6, 3)$; across all 159 pairs at N=100, 9 survive Holm. The best-mean method rotates: Evolve on three tasks, FunSearch on two, Reflect$_{32}$ on one, best-of-N on three. A win found in one half of the seeds replicates in the other at rates between 0 and 0.37, and the best method in one half is the best in the other in fewer than 35% of splits on six of nine tasks.

Task	seeds	σ_{within}	$\sigma_{\text{between}}/\sigma_{\text{within}}$	MDE (σ)	disting. raw / Holm	replication	same winner
Sidon set	19	0.65	0.36	0.91	4/1 of 15	0.28	0.70
LABS	20	0.27	0.32	0.89	3/1 of 15	0.12	0.31
Ramsey K_4	19	7.81	0.53	0.91	5/1 of 15	0.09	0.18
Circle packing	19	0.32	0.37	0.91	6/0 of 15	0.01	0.21
$A_2(21, 10)$	20	4.05	0.13	0.89	0/0 of 15	0.00	0.00
$A_3(13, 8)$	20	2.03	0.32	0.89	2/0 of 15	0.00	0.03
$C(13, 6, 3)$	20	0.48	0.45	0.89	5/1 of 15	0.25	0.69
$C(20, 12, 3)$	20	0.21	0.33	0.89	0/0 of 15	0.00	1.00
$C(24, 6, 3)$	20	11.81	0.56	0.89	8/4 of 15	0.37	0.15
All					33/8 of 135		

Table 2: N=100, main grid. σ_{within}: mean seed SD across methods, in score units. MDE in σ_{within} units. Distinguishable pairs: bootstrap $p < 0.05$, raw and Holm-corrected, of 15. Replication: fraction of half-A wins reproduced in half B. Same winner: fraction of splits where the best-mean method agrees across halves.

4.3 At 300 calls the first replicating advantage appears

Figure 1b and Table 5 (Appendix E) give the 21 contrasts on seven tasks at ten seeds. Nineteen are ties, one a small loss (Evolve on LABS, -0.22), and one behaves differently from anything at N=100: FunSearch on $A_3(13, 8)$ gains $+1.00$ codeword $[+0.60, +1.40]$, win rate 0.90, one within-task seed-SD, and the advantage replicates in 70% of disjoint seed-half splits with the same winner in every split. It is the only win in the main grid that survives its own replication test. Sidon sets show all three methods at $+0.4$ to $+0.7$ with intervals including zero (MDE 1.09 at ten seeds). Nine of 42 pairs are distinguishable raw, 3 after Holm.

4.4 What a k-seed study would have concluded

Figure 1c and Table 6 (Appendix E) re-analyse our cells as a study with k seeds per method. At N=100 a three-seed study misses 81% of the effects the 20-seed analysis detects and declares a win the full analysis does not support in 11% of pairs, about 1.6 per task at 15 pairs; at ten seeds it still misses 55% and disagrees on 7%. At N=300 the effects are larger and the miss rate lower, 56% at k=3, but the inconsistent-win rate rises to 14%. Part of the excess over the nominal 5% is the percentile bootstrap being anti-conservative at three samples.

4.5 The ties are not model collapse

Every one of the 1,149 usable cells at N=100 has a champion program distinct from every other cell of the same task and method (whitespace- and comment-normalised hash). What is coarse is the score: on $C(13, 6, 3)$ and $C(20, 12, 3)$ twenty distinct programs land on one to three block counts, on Sidon on two to five set sizes, while LABS, circle packing and Ramsey spread over 6 to 18 distinct values. FunSearch's win on $C(13, 6, 3)$ is eight more seeds landing on the 21-block plateau rather than a different construction.

4.6 Pre-registered predictions for 300 calls

Before Stage 2 we recorded four predictions (Appendix D). **P1**, that the best non-sampling advantage grows from 100 to 300 calls but stays below one seed-SD on every task, a crossover to be declared if any method exceeds one SD on two or more tasks at corrected significance: holds by its letter and fails in spirit on $A_3(13, 8)$, where FunSearch reaches 0.98 within-task SD with corrected significance and 70% replication; we report it as the onset of a crossover on one task. **P2**, that Reflect$_{16}$ beats best-of-N on at least one of the code or covering tasks: refuted. It ties on $A_3(13, 8)$ ($+0.2$) and $C(13, 6, 3)$ (0.0) and is -9 blocks on $C(24, 6, 3)$ $[-18, 0]$. **P3**, at most one evolutionary win at corrected significance: confirmed. **P4**, the $C(24, 6, 3)$ collapse of reflection persists: confirmed directionally (p=0.053). The reflection wins an earlier protocol found on cap set and binary codes stand at their seed counts (Appendix C); the scope hypothesis they suggested, that reflection helps wherever a graded ladder of constructions exists, does not.

4.7 OUTSIDE THE CONSTRUCTION FAMILY: TWO SOLVER TASKS

Predictions A1 (no method beats best-of-N by more than one seed-SD on MAX-3-SAT or planted clique) and A2 (Hedge ties on both) were recorded before the run. Planted clique is noise: seed SD 19, between/within ratio 0.07, every contrast a tie. On MAX-3-SAT, FunSearch beats best-of-N by $+5.3$ satisfied clauses $[+1.5, +8.9]$, 0.93 seed-SD, Holm-corrected p=0.04; Evolve and Hedge tie (Table 7, Appendix E). Both predictions hold; the substantive reading is that the FunSearch edge seen on $C(13, 6, 3)$ recurs on a task from another family.

4.8 DOES HEADROOM PREDICT WHERE EVOLUTION PAYS?

The natural reading of the $A_3(13, 8)$ result is that evolution pays where the model's first-shot programs are far from what is achievable. We pre-registered it on one family, $C(v, 6, 3)$ for $v \in \{13, 16, 20, 24, 28, 32\}$ (records 21, 38, 71, 116, 185, 300): the FunSearch advantage at N=300 should increase with headroom, the gap between best-of-N's mean champion and the record in seed-SD (B1); no such relation at N=100 (B2); Evolve should not show it (B3). Table 8 and Figure 2 (Appendix E) give the outcome.

B1 is refuted. The Spearman correlation between headroom and FunSearch's advantage over the six instances is -0.26 at N=100 and -0.49 at N=300. The advantage is largest at moderate distance from the record: on $C(20, 6, 3)$, where best-of-N sits 14% above it, FunSearch saves 4.9 blocks $[+2.7, +7.0]$ at N=300, 2.1 seed-SD, win rate 0.93, $p < 10^{-4}$, the largest effect in the DeepSeek grid, and Evolve saves 3.4 $[+0.6, +6.0]$, 1.5 seed-SD, Holm-corrected p=0.02. Where the gap is largest the sign reverses: on $C(28, 6, 3)$ (27% above record) FunSearch is 4.0 blocks worse $[-7.4, -0.2]$, on $C(32, 6, 3)$ (18%) 4.8 blocks worse with an interval touching zero; Evolve is 2.1 and 1.5 blocks worse there with intervals including zero. Where the gap is small the effect is a fraction of a block. $A_3(13, 8)$, 40% from its record with a one-SD gain, fits no simple curve either. B2 fails marginally: at N=100 FunSearch reaches $+1.03$ and $+1.01$ seed-SD on v=16 and 20 (raw p=0.04, 0.03, not surviving Holm). B3, that Evolve does not show the relation, holds vacuously: Evolve traces the same non-monotone shape at smaller amplitude, with the same rank correlation of -0.49.

Where evolution pays is therefore instance-specific and not predicted by distance to the record. One reading that fits, offered as a hypothesis, is that a population helps when the model's first-shot constructions are diverse enough to recombine but not so far from good structure that the archive locks onto a poor basin; at v=28 and 32 the greedy constructions the model writes sit a quarter to a third above the record, and evolution seems to converge on them. The next section tests the other half of this reading: change the model and the first-shot programs, keep everything else.

4.9 THE MODEL MOVES THE FLOOR

Stage 3C reran best-of-N, FunSearch and Evolve with `Qwen2.5-7B-Instruct` on four tasks chosen before the run: $C(20, 12, 3)$, where DeepSeek is saturated; $C(20, 6, 3)$, where FunSearch wins; $C(28, 6, 3)$, where it loses; and MAX-3-SAT. The prediction was recorded both ways (Appendix D): the BaSE reconciliation in §5 predicts larger between-method spread with evolution winning where the first-shot programs are poor, while the alternative, that the floor is a property of the budget, predicts the same floor. Stage 3D then repeated the four tasks with `gpt-4.1-mini`, a mid-strength model from a third provider, with the prediction, also recorded in advance, that its results would fall between the other two. Table 3 gives all three models side by side.

The floor moved with the weak model. On MAX-3-SAT, Evolve beats sampling by 34 satisfied clauses $[+14, +50]$, 1.6 seed-SD, win rate 0.90, and FunSearch by 21 $[+5, +37]$, 1.0 seed-SD, both Holm-corrected; the between/within ratio is 0.80 against 0.39 for DeepSeek. On $C(28, 6, 3)$ the sampling champion is often a valid but poor program, one of 2,300 blocks against a record of 185, and two of ten best-of-N cells have no valid champion at all; both evolutionary methods return 270 to 280 blocks on average, a gap of 300 blocks $[+35, +805]$ with win rate above 0.9 that replicates in 98% of seed-half splits. On the two tasks where Qwen plateaus, at 11 blocks on $C(20, 12, 3)$ where DeepSeek reaches 10 and at 85 blocks on $C(20, 6, 3)$, all three methods tie, exactly as for DeepSeek. Across the twelve method pairs, four are distinguishable after Holm correction against one of twelve for DeepSeek on the same tasks.

Task	model	best-of-N mean	$\sigma_{\text{between}}/\sigma_{\text{within}}$	FunSearch vs best-of-N	Evolve vs best-of-N
$C(20, 12, 3)$	DeepSeek V4-Flash	10.0	0.36	-0.1 [-0.2, +0.0] (-0.4σ)	-0.1 [-0.2, +0.0] (-0.8σ)
	gpt-4.1-mini	10.0	–	+0.0 [+0.0, +0.0] (all seeds equal)	+0.0 [+0.0, +0.0] (all seeds equal)
	Qwen2.5-7B	11.0	0.34	-0.2 [-0.9, +0.3] (-0.6σ)	+0.0 [+0.0, +0.0] (+0.0σ)
$C(20, 6, 3)$	DeepSeek V4-Flash	82.1	0.51	**+3.6 [+0.3, +6.5] (+1.0σ)**	+1.8 [-1.3, +4.7] (+0.5σ)
	gpt-4.1-mini	82.6	0.26	-1.5 [-4.7, +2.0] (-0.4σ)	+0.5 [-2.8, +3.7] (+0.1σ)
	Qwen2.5-7B	84.7	0.52	+0.5 [-0.6, +1.8] (+0.2σ)	-2.7 [-7.3, +1.1] (-0.8σ)
$C(28, 6, 3)$	DeepSeek V4-Flash	240.3	0.22	-2.7 [-7.3, +2.0] (-0.4σ)	-1.1 [-7.2, +4.8] (-0.2σ)
	gpt-4.1-mini	254.0	0.59	+4.8 [-0.1, +9.7] (+0.8σ)	**+6.8 [+1.9, +11.6] (+1.1σ)**
	Qwen2.5-7B	579.0	0.73	**+303.0 [+35.2, +803.5] (+1.3σ)**	**+307.2 [+41.6, +804.9] (+1.3σ)**
MAX-3-SAT	DeepSeek V4-Flash	6857.6	0.39	**+5.3 [+1.5, +8.9] (+0.9σ)**	+2.2 [-2.0, +6.2] (+0.4σ)
	gpt-4.1-mini	6858.3	0.31	+2.5 [-2.9, +8.1] (+0.4σ)	+4.2 [-1.5, +10.0] (+0.6σ)
	Qwen2.5-7B	6720.7	0.80	**+21.2 [+5.4, +36.6] (+1.0σ)**	**+34.3 [+13.6, +49.6] (+1.6σ)**

Table 3: Three models on the same four tasks, N=100, 10 seeds per cell (DeepSeek: 20 seeds on $C(20, 12, 3)$ and MAX-3-SAT, 15 for best-of-N on MAX-3-SAT). Difference from best-of-N in score units with 95% interval and in within-task seed-SD; bold where $p < 0.05$. Covering designs are blocks, lower is better, so a positive difference is fewer blocks. On $C(20, 12, 3)$ every gpt-4.1-mini seed reaches the saturated 10 blocks, so no spread is defined.

The mid-strength model lands where its own plateau puts it. On MAX-3-SAT gpt-4.1-mini's sampling champion averages 6858 clauses, the same as DeepSeek's 6858, and both evolutionary methods tie sampling ($+4$ and $+3$ clauses, intervals through zero, ratio 0.31); on $C(20, 12, 3)$ every seed of every method reaches the saturated 10 blocks, and on $C(20, 6, 3)$ sampling reaches 82.6 blocks, within a block of DeepSeek, with ties throughout. Only on $C(28, 6, 3)$, where its sampling champion averages 254 blocks against DeepSeek's 240 and Qwen's 579, does evolution pay: Evolve saves 6.8 blocks $[+1.9, +11.6]$, 1.1 seed-SD, Holm-corrected p=0.02, and FunSearch 4.8 $[-0.1, +9.7]$, 0.8 seed-SD. The between/within ratio on that task, 0.59, sits between DeepSeek's 0.22 and Qwen's 0.73. Every champion of all three models is a distinct program, and no gpt-4.1-mini cell failed to produce one.

The reading is the one the ladder suggested, now with three points on it. Evolution pays when the model's first-shot programs fall short of the plateau that sampling can reach and the evaluator feedback in the loop repairs them; it does not pay when sampling already reaches that plateau, whatever the distance to the published record. On $C(28, 6, 3)$ the three models order by the quality of their sampling champion, 240, 254 and 579 blocks, and the evolutionary advantage orders with them, a tie, one seed-SD and 1.3 seed-SD; on MAX-3-SAT gpt-4.1-mini reaches DeepSeek's plateau and ties as DeepSeek does. What matters is headroom for the model in hand rather than distance to the record, and the scope of the main result is a model-task pair in which a hundred first-shot samples reach the model's own plateau, which held within 1.03 seed-SD on every task for the frontier model and on three of four for the mid-strength one.

5 Discussion

Relation to BaSE. Xing et al. (2026) report a 12.3% improvement from bandit-based allocation over island protocols at budgets of 8 to 512 calls, whereas in our experiments the EXP3 arm ties sampling at 100 calls (with the caveat that its compute arm was degenerate) and the best evolutionary gain at 300 calls is confined to particular instances. The two sets of results are compatible once the models and objectives are compared. The models in BaSE are open-weight models of 1.7B to 14B parameters whose first-shot programs leave considerable room for refinement, while ours is a frontier-class model whose first hundred samples already reach the attainable plateau on most of our tasks. In addition, the objectives in BaSE are continuous, whereas ours change in integer steps, so that an improvement must clear a threshold before it registers. Both differences imply that allocation has more to work with in their setting, which is what the cross-model experiment in §4.9 tested. With a 7B model the spread reappears where its first-shot programs are poor and stays absent where it plateaus, and a mid-strength model from a third provider falls between the two exactly where its plateau does, so in our reading it is model capability relative to the task, rather than the budget, that sets the height of the floor.

Where the regime ends. Since sampling improves roughly as $\log N$ while evolution and reflection incur an overhead before they can pay off, some crossover is to be expected, and the question is where it occurs and for which methods; across our fifteen tasks the answer is narrow. Reflection,

adaptive allocation and the lighter island model do not cross at either budget, while FunSearch sits at the edge at 100 calls and is clearly ahead on some instances at 300 calls but behind on others, with no simple predictor of which instances fall on which side. The regime also depends on the model, as §4.9 shows with three models, and it does not extend to production systems, which operate at budgets far beyond both of ours.

FunSearch versus Evolve. FunSearch clusters programs by their score signature, samples clusters through a tempered softmax and presents two programs side by side, which amounts to a population that recombines; Evolve maintains a single ranked archive per island and requests an improvement to one parent together with its feedback, which amounts to five parallel hill-climbs. With DeepSeek, whose programs generally work, FunSearch wins on four tasks and Evolve on one. With Qwen, where the difficulty is that programs frequently fail, Evolve is the stronger of the two, and it is plausible that a parent presented together with its evaluator feedback offers the shorter path to a working program. We put this forward as a hypothesis rather than as a finding.

Recommendations for reporting. Applied to our own data, the prevailing practice of running one to eight seeds and reporting the best would have produced roughly 1.6 method winners per task that a 20-seed analysis does not support, while missing four of the five effects that it does detect. We therefore suggest that a method comparison at these budgets should report, at a minimum, the number of seeds and the within-method standard deviation for each task, the minimum detectable effect alongside every claimed tie, a non-inferiority margin for any claim that a method matches the baseline, split-half replication for any claimed win, and champions that have been re-verified as valid objects. The cost of doing so is modest, since at roughly a tenth of a cent per request a twenty-seed comparison of two methods on one task at 100 calls cost us under five dollars, and the entire study of 260,000 requests about $300.

6 Limitations

The main grid uses a single model. The two further models cover four tasks and three methods at ten seeds, and their wins occur on tasks where sampling's champions were poor, so they establish that the floor moves with the model and order three points along it without tracing the curve between them. The task suite consists of thirteen construction tasks and two solver tasks, two of them near-saturated, with no continuous-optimisation or agentic tasks. We study budgets of 100 and 300 calls only. Twenty seeds bound the detectable gap at 0.9 seed-SD and ten seeds at 1.3, so an effect of half a seed-SD would pass undetected, which is why every tie is reported together with its minimum detectable effect. We tested two reflection periods only, and Hedge ran with a degenerate duplicate arm.

7 Conclusion

At a hundred LLM calls per problem, the budget most practitioners have, the choice among sampling, evolution, reflection and adaptive allocation lies at or below the noise floor of the search itself, since no method beats best-of-N by more than 1.03 seed-SD on any of fifteen tasks and only FunSearch-style evolution shows a consistent small edge. At three hundred calls that edge becomes large on some instances and negative on others, and the distance from the published record does not indicate which. The floor moves with the model. Three models from three providers order the same way on the hardest instance, the better a model's own sampling champion the less evolution adds, from 1.3 seed-SD for the weakest through one for the mid-strength model to a tie for the strongest, and all three tie wherever the model plateaus. If this ordering holds more widely, the regime in which sampling suffices should widen as models improve, because a stronger model reaches its own plateau with fewer samples and leaves evolution less room at any fixed budget. Until then, attribution of success in LLM-based discovery should start from sampling plus selection, state the seeds and the detectable effect, and treat a single-run win as a hypothesis.

Reproducibility statement

The harness, every generated program with its scores, the per-cell result files for all stages, the analysis, diversity, figure and table scripts and the pre-registration files accompany the paper. Every number in §4 comes from one invocation of the analysis script per stage; Appendix B gives the prompts and constants needed to re-run a cell without the code.

AI use statement

The experiments call LLM APIs by design; every program in the results was generated by `deepseek-chat`, `gpt-4.1-mini` or a locally served `Qwen2.5-7B-Instruct` inside the search loop. Separately, we used LLM assistants to review drafts and code, to help implement the plotting scripts, and for feedback on the interpretation of results. All AI-assisted output was checked by the authors, and every reported number was traced to the analysis output. We take responsibility for the final content of this work.

References

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, et al. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, et al. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Alexandra Carpentier and Michal Valko. Extreme bandits. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

Vincent A. Cicirello and Stephen F. Smith. The max k-armed bandit: A new model of exploration applied to search heuristic selection. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI)*, pp. 1355–1361, 2005.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. How many random seeds? Statistical power analysis in deep reinforcement learning experiments. *arXiv preprint arXiv:1806.08295*, 2018.

Yonatan Gideoni, Sebastian Risi, and Yarin Gal. Simple baselines are competitive with code evolution. *arXiv preprint arXiv:2602.16805*, 2026.

Daniel M. Gordon. La Jolla covering repository. `https://ljcr.dmgordon.org/cover.html`, data at `https://github.com/dmgordo/LJCR`, 2026. Accessed September 2026.

Michael Hassid, Tal Remez, Jonas Gehring, et al. The larger the better? Improved LLM code-generation via budget reallocation. *arXiv preprint arXiv:2404.00725*, 2024.

Peter Henderson, Riashat Islam, Philip Bachman, et al. Deep reinforcement learning that matters. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

Jie Huang, Xinyun Chen, Swaroop Mishra, et al. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations (ICLR)*, 2024.

Robert Tjarko Lange, Yuki Imajuku, and Edoardo Cetin. ShinkaEvolve: Towards open-ended and sample-efficient program evolution. *arXiv preprint arXiv:2509.19349*, 2025.

Kuang-Huei Lee, Ian Fischer, Yueh-Hua Wu, et al. Evolving deeper LLM thinking. *arXiv preprint arXiv:2501.09891*, 2025.

Aman Madaan, Niket Tandon, Prakhar Gupta, et al. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, et al. AlphaEvolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Theo X. Olausson, Jeevana Priya Inala, Chenglong Wang, et al. Is self-repair a silver bullet for code generation? In *International Conference on Learning Representations (ICLR)*, 2024.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

Noah Shinn, Federico Cassano, Edward Berman, et al. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Charlie Snell, Jaehoon Lee, Kelvin Xu, et al. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. In *International Conference on Learning Representations (ICLR)*, 2025.

Xuezhi Wang, Jason Wei, Dale Schuurmans, et al. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Sixue Xing, Haoyu He, Kerui Wu, et al. Compute allocation for self-evolving LLMs: From depth-breadth to multi-armed bandits. *arXiv preprint arXiv:2605.29268*, 2026.

Shunyu Yao, Dian Yu, Jeffrey Zhao, et al. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

A Simple theoretical model of the noise floor

We give a simple theoretical model illustrating when method effects become intrinsically difficult to distinguish from finite-seed variation. Consider a budget of $N = n + m$ model calls and a target performance level that we call a *success*. Each fresh sample is independently either successful with probability q, repairable with probability p, or neither, where $p + q \leq 1$. Each mutation of a repairable sample independently produces a success with probability r.

Best-of-N sampling succeeds with probability

$$S_N = 1 - (1 - q)^N.$$

Consider an evolutionary strategy that uses n calls for fresh samples and, if no success but at least one repairable sample is found, spends the remaining $m = N - n$ calls mutating a repairable sample. Its success probability is

$$E_{n,m} = 1 - (1 - p - q)^n - \Big[(1 - q)^n - (1 - p - q)^n\Big](1 - r)^m. \tag{1}$$

Thus, evolution beats independent sampling exactly when $E_{n,m} > S_N$.

The model also exhibits a finite seed noise floor. Let

$$f_N = 1 - S_N = (1 - q)^N$$

be the probability that N independent samples fail. Since no alternative can have success probability above one, any positive evolutionary advantage

$$\Delta = E_{n,m} - S_N$$

satisfies

$$0 \leq \Delta \leq f_N \leq e^{-Nq}. \tag{2}$$

Hence, as sampling approaches saturation, the maximum possible method effect itself vanishes.

Suppose each method is evaluated using k independent seeds and success is recorded as a Bernoulli outcome. The standard error of the difference of empirical success rates is

$$\mathrm{SE} = \sqrt{\frac{S_N(1 - S_N) + E_{n,m}(1 - E_{n,m})}{k}} \geq \sqrt{\frac{(1 - f_N)f_N}{k}}.$$

Combining this with (2), whenever $\Delta > 0$,

$$\frac{\Delta}{\mathrm{SE}} \leq \sqrt{\frac{k f_N}{1 - f_N}}. \tag{3}$$

For any fixed seed count k, this signal-to-noise ratio converges to zero as $f_N \to 0$. In particular, once

$$(1-q)^N \ll \frac{1}{k},$$

even the largest possible improvement over sampling is smaller than the finite-seed standard error. This is a simple formal instance of a *noise-floor regime*: increasing search quality can make between-method differences vanish faster than the uncertainty with which they can be estimated.

Outside saturation, evolution can still help. Relative to using all N calls for fresh samples, it sacrifices the additional sampling success probability

$$(1-q)^n - (1-q)^N.$$

Its compensating benefit requires encountering a repairable program, an event with probability at most np. Therefore

$$\Delta \leq np - \Big[(1-q)^n - (1-q)^N\Big]. \tag{4}$$

If useful stepping stones are too rare, their potential benefit cannot compensate for the independent samples that evolution replaces. Between these extremes lies an evolvable regime: direct successes are sufficiently rare that sampling has not saturated, while repairable programs occur sufficiently often and mutations improve them reliably.

The model therefore predicts three qualitative regimes. When q is large relative to the budget, sampling saturates and method effects fall below a finite-seed noise floor. At intermediate difficulty, q can be small while p and r remain substantial, allowing evolution to produce a detectable advantage. When useful stepping stones themselves become rare or difficult to repair, that advantage disappears or reverses. Thus distance from the global optimum is not sufficient to predict whether evolution helps; the relevant quantities are the model's probability of directly reaching its attainable plateau and the *evolvability* of the programs it produces. For real-valued scores, the same statistical principle applies whenever the remaining attainable difference in method means becomes small relative to the seed-level standard error; equation (3) gives the exact result for the simplest binary case.

B Implementation details

Prompts. Every request shares one system message ("You are an expert in algorithms, combinatorial optimization and heuristic design. You respond with a single Python function in a ```` ```python ```` code block. No usage examples, no commentary outside the code block.") and every user message begins with the task sheet (description, required signature, record window where one exists, runtime limit) and ends with an instruction to output exactly one code block. The unconditioned prompt (best-of-N, and reflection before its first insight) adds the seed program as "a valid but weak reference implementation" and asks for a new and better one; the conditioned prompt (reflection after an insight) adds the current insight text under "insights distilled from the campaign so far" and asks for an implementation that applies them, with no exemplar programs; the mutate prompt (Evolve, and the evolution arm of Hedge) adds the parent program with its score and its opt-partition feedback truncated to 200 characters and asks for an improved version. The reflection request uses its own system message (a research analyst who "extracts transferable principles, not code"), shows the twelve best behaviourally distinct attempts with scores and feedback, and asks for at most five bullet points and 150 words of principles, no code. Completion caps are 3000 tokens for programs and 500 for reflections; temperature 1.0 for programs and 0.7 for reflections.

FunSearch. The search logic is a line-by-line port of the released implementation (`programs_database.py`, `config.py`) with its default configuration: ten islands; prompts drawn from one uniformly chosen island; programs clustered by signature (the tuple of per-test scores); cluster choice by tempered softmax over cluster scores with $T = 0.1\,(1 - (n \bmod 30000)/30000)$, sampled with replacement; within a cluster, programs sampled by softmax over negative length-normalised size at temperature 1; two functions per prompt ordered ascending by

score, asking for the next version; four samples per prompt, all registered to the prompt's island; island reset keeps the better half and reseeds the rest from a surviving island's best program. Three deviations were forced by the controlled setting: the reset period is $N/4$ requests rather than four hours of wall-clock; the reduced score used for clustering and selection is the task's validation utility rather than "the last test's score"; and the prompt is prefixed by the shared task sheet, with the v_0, v_1, v_2 format preserved after it. Randomness is seeded so runs are reproducible and resumable.

Evolve. Five islands, each an archive of up to 50 candidates; per step, each island samples one parent with probability proportional to $2^{-j/8}$ for rank j and one mutate request is issued per island; offspring are re-seated in the parent's island; island reset every 250 requests (never triggered at $N \leq 300$). The seed program is the root of every island.

Hedge. EXP3 over four arms (best-of-N, best-of-N with compute, Reflect$_{32}$, the Evolve island model), one arm chosen per batch of eight requests by multiplicative weights on marginal improvement of the arm's best validation score, learning rate η=0.06, exploration floor γ=0.06 (every arm keeps probability at least γ/K), EMA smoothing 0.5 on the per-arm improvement estimate, an arm with no valid candidate scored at -1. With the compute budget at zero the compute arm is identical to the sampling arm.

Further models. `Qwen2.5-7B-Instruct` (bf16) served with vLLM 0.29 on one L40S, OpenAI-compatible chat endpoint, PyTorch sampler, context 8192; the harness is unchanged apart from the model name and endpoint (two environment variables). Temperature 1.0, 3000 completion tokens, the same system message, task sheets and prompts. 120 cells, 12,000 requests, about two hours of GPU time. Three cells (two best-of-N on $C(28, 6, 3)$, one best-of-N on MAX-3-SAT) produced a champion that failed on the test partition and are excluded like zero-valid cells; the analysis reports them. The `gpt-4.1-mini` model was run through the OpenAI chat-completions API with the same harness, prompts, temperature and token cap; the only code change was to omit a DeepSeek-specific request field for non-DeepSeek endpoints. 120 cells, 12,000 requests, about $25; no cell failed to produce a valid champion.

Deduplication and selection. Candidates are fingerprinted by a task-specific behavioural signature of the validation result (the constructed object or its score profile); the top-k operations used for reflection exemplars (k=12) and final selection (k=20) take the best validation-utility candidates with distinct fingerprints. Final selection scores those twenty on the *sel* partition and the winner once on *test*.

Sandbox. Static ban list (`os`, `sys`, `subprocess`, `socket`, `shutil`, and similar), subprocess with memory and CPU limits, wall-clock timeout per task (15 s for covering designs); a timeout, crash or ban-list hit is an invalid candidate that consumes its request.

Accounting. Requests are charged to the budget at dispatch. A request whose outcome is unknown (timeout at the provider) is counted as spent and produces no candidate; a cell that accumulates more than 0.5% of N such requests stops. Ten best-of-N cells on the solver tasks were stopped this way at their first batch when the array started (a burst of ambiguous responses); the analysis uses the fifteen best-of-N cells per solver task that ran to completion, and Table 7 reports the seed counts. Eighteen Evolve cells on the ladder at N=300 were stopped by the project spending cap and resumed from their journals. All cells reported are complete.

C Results carried from an earlier protocol

An earlier version of this study, on `gpt-4.1-mini` at 160 calls with a 700-token cap and a different task set (bin packing, cap set, TSP, weighted tardiness, no-three-in-line, corner-free sets, binary codes), established three results that the present protocol does not re-measure. First, evaluator density buys selection rather than generation: varying only the feedback channel (binary, dense, decomposable) with final selection uniform, the champion under true-score selection barely moved with feedback density, while the champion a practitioner would deploy under the shaped signal alone collapsed under binary feedback (-7.41 versus -5.33 excess bins on bin packing, 73.6 versus 77.4 on cap set, 8 seeds); dense feedback matters through knowing which candidate to keep, a channel equally available to best-of-N. Second, sequential repair failed through repair bloat: under default improvement prompts programs grew every round until they truncated at the token cap, validity fell from 99% for sampling to 28–51% for the iterative arms, and cap-set scores fell 3 to 5 points below

best-of-N with paired intervals excluding zero; doubling the cap restored parity, which is consistent with FunSearch and Evolve tying rather than losing at the 3000-token cap used here. Third, reflect-then-sample beat best-of-N on cap set in dimension 6 (86.1 versus 82.4 over 22 seeds, $[+1.2, +7.2]$, recovering the optimal 112-element cap once) and in dimension 7 (174.7 versus 169.9 over 18 seeds, a pre-registered prediction), and broke a lexicode plateau of 32 on binary codes (10,4) in three of eight runs ($+1.0$, $[+0.1, +2.5]$); it tied on three tasks and lost on TSP, its dose curve at 8 seeds per point was inside the noise, and a cross-provider check on `claude-haiku-4-5` (8 seeds) was directionally consistent and not significant.

D PRE-REGISTRATIONS

Stage 2 (recorded 2026-09-07, before any N=300 cell beyond an earlier 5-seed sweep on t01, t05, t08). Design: best-of-N, FunSearch, Evolve, Reflect$_{16}$ on t01–t04, t08, t09, t11, seeds 0–9. P1: the pooled advantage of the best non-sampling method grows from N=100 but stays below one seed-SD on every task; if any method exceeds $+1$ SD on two or more tasks with Holm-corrected $p < 0.05$ we report a crossover. P2: Reflect$_{16}$ beats best-of-N on at least one of t08, t09 and not on t01, t02, t04. P3: FunSearch and Evolve tie or gain slightly on t01, t08, t09; at most one reaches corrected significance. P4: Reflect$_{16}$ remains below best-of-N on t11. Decision rules: two-sided bootstrap $p < 0.05$ with Holm within task, MDE with every tie, replication and k-seed subsampling, champions re-verified, nothing added or removed after submission, infrastructure failures resumed from journals. Outcomes in §4.6.

Stage 3 (recorded 2026-09-14, before any Stage 3 cell). 3A: best-of-N, FunSearch, Evolve, Hedge on MAX-3-SAT and planted clique, 20 seeds, N=100; A1: no method beats best-of-N by more than one seed-SD on either; A2: Hedge ties on both. 3B: best-of-N, FunSearch, Evolve on $C(v, 6, 3)$, $v \in \{16, 20, 28, 32\}$ (LJCR records 38, 71, 185, 300), 10 seeds, N=100 and 300, together with the existing v=13 and 24; headroom defined as (record − best-of-N mean champion) / within-method seed SD; B1: at N=300 the FunSearch advantage increases with headroom (Spearman $\rho > 0.5$ over six instances); B2: no such relation at N=100 and no advantage above one SD there; B3: Evolve does not show the N=300 relation. 3C: a second, weaker open-weight model, with prediction C1 recorded both ways because two positions in the draft conflicted. The BaSE reconciliation predicts larger between-method spread (ratio above 0.7 on most tasks) with evolution winning where first-shot programs are poor; the generality hypothesis predicts the same floor (ratio below 0.6, no win above one SD). The registered design was four methods on the nine Stage 1 tasks; GPU time limited the run to three methods on four tasks chosen before it started ($C(20, 12, 3)$, $C(20, 6, 3)$, $C(28, 6, 3)$, MAX-3-SAT), a reduction we disclose. Outcomes: A1 and A2 hold; B1 refuted ($\rho = -0.49$); B2 fails marginally (1.03 and 1.01 SD on two instances, not surviving Holm); B3 holds vacuously, Evolve tracing the same shape with $\rho = -0.49$; C1: ratios 0.80 and 0.73 with evolutionary wins of 1.0 to 1.6 SD on the two tasks where Qwen's first-shot programs are poor, 0.34 and 0.52 with ties on the two where it plateaus, so the BaSE-reconciliation side holds where the model has room and the same-floor side where it does not.

Stage 3D (recorded 2026-09-23, before any 3D cell; first recorded for `microsoft/phi-4` on the cluster GPU, then switched to `gpt-4.1-mini` the same afternoon when the GPU server failed to receive requests, before any result from either model had been seen). Design: best-of-N, FunSearch, Evolve on the four Stage 3C tasks, 10 seeds, N=100. D1: on MAX-3-SAT and $C(28, 6, 3)$ the evolutionary advantage and the between/within ratio lie between Qwen-7B's and DeepSeek's. D2: the two plateau tasks stay ties. D3: fewer failed best-of-N cells than Qwen's 3 of 40. Outcomes: D1 holds on $C(28, 6, 3)$ (Evolve $+1.1$ seed-SD, ratio 0.59, between 0.22 and 0.73) and fails as written on MAX-3-SAT, where gpt-4.1-mini's sampling champion equals DeepSeek's (6858 clauses) and evolution ties rather than falling between; the ordering by sampling-champion quality predicts both outcomes, and the text reports that ordering rather than a between-ness claim. D2 holds. D3 holds (0 of 40).

E ADDITIONAL TABLES AND POOLED STATISTICS

Pooled across the nine main-grid tasks in seed-SD units (normal approximation over task-level bootstrap intervals), Evolve, FunSearch and Hedge sit at -0.04, -0.05 and -0.09 with half-widths of

about 0.2, and the reflection arms at -0.23 and -0.25 with upper limits at zero; over all fifteen tasks FunSearch pools to $+0.17$ $[-0.01, +0.36]$ and Evolve to $+0.07$ $[-0.11, +0.24]$. These figures are secondary to the per-task results.

Table 4: N=100, 20 seeds, main grid: mean difference from best-of-N in the task's score units, 95% bootstrap interval, bold where $p < 0.05$.

Task	FunSearch	Evolve	Reflect$_{16}$	Reflect$_{32}$	Hedge
Sidon set	+0.18 [-0.18, +0.55]	**+0.45 [+0.15, +0.70]**	+0.10 [-0.35, +0.65]	-0.25 [-0.55, +0.05]	+0.20 [-0.20, +0.55]
LABS	-0.04 [-0.19, +0.11]	-0.11 [-0.24, +0.03]	+0.13 [-0.03, +0.28]	+0.09 [-0.08, +0.28]	-0.03 [-0.19, +0.14]
Ramsey K_4	**-9.25 [-20.00, -0.75]**	+1.29 [-1.57, +4.47]	+0.50 [-2.85, +3.75]	+1.95 [-0.95, +4.90]	-0.10 [-2.30, +2.15]
Circle packing	+0.05 [-0.04, +0.17]	+0.06 [-0.03, +0.18]	-0.22 [-0.54, +0.03]	-0.16 [-0.44, +0.06]	-0.10 [-0.27, +0.07]
$A_2(21, 10)$	+0.55 [-1.60, +2.60]	+0.35 [-2.30, +3.05]	+0.55 [-2.05, +3.20]	-0.45 [-2.90, +2.00]	-0.70 [-2.35, +0.90]
$A_3(13, 8)$	-0.10 [-1.15, +1.00]	-1.05 [-2.10, +0.00]	**-1.60 [-3.00, -0.20]**	-0.15 [-1.35, +1.00]	-0.25 [-1.25, +0.75]
$C(13, 6, 3)$	**+0.40 [+0.10, +0.65]**	+0.10 [-0.20, +0.40]	+0.30 [+0.00, +0.60]	+0.05 [-0.25, +0.35]	-0.20 [-0.55, +0.15]
$C(20, 12, 3)$	-0.05 [-0.15, +0.00]	-0.10 [-0.25, +0.00]	-0.15 [-0.30, +0.00]	-0.15 [-0.30, +0.00]	+0.00 [+0.00, +0.00]
$C(24, 6, 3)$	-3.00 [-9.10, +2.00]	-3.35 [-9.90, +1.85]	**-15.15 [-22.20, -7.60]**	**-13.20 [-21.25, -5.15]**	-0.05 [-5.00, +4.65]

Table 5: N=300, 10 seeds: mean difference from best-of-N, 95% interval, bold where $p < 0.05$.

Task	FunSearch	Evolve	Reflect$_{16}$
Sidon set	+0.40 [-0.20, +1.00]	+0.40 [-0.10, +0.90]	+0.70 [-0.10, +1.70]
LABS	-0.10 [-0.31, +0.11]	**-0.22 [-0.43, -0.02]**	-0.14 [-0.34, +0.07]
Ramsey K_4	-2.20 [-6.90, +2.40]	+2.40 [-2.20, +7.00]	-10.10 [-23.90, +0.80]
Circle packing	+0.06 [-0.00, +0.15]	-0.25 [-0.79, +0.07]	-0.09 [-0.40, +0.11]
$A_3(13, 8)$	**+1.00 [+0.60, +1.40]**	-1.10 [-2.30, +0.10]	+0.20 [-0.40, +0.90]
$C(13, 6, 3)$	+0.20 [+0.00, +0.50]	+0.20 [+0.00, +0.50]	+0.00 [-0.30, +0.30]
$C(24, 6, 3)$	-2.40 [-7.80, +0.60]	+0.10 [+0.00, +0.30]	-9.00 [-18.00, +0.00]

Table 6: Verdict of a k-seed study, pooled over tasks, with the full-seed analysis as reference. Inconsistent win: full-seed tie declared a win at k. Missed: full-seed win declared a tie at k.

k seeds	N=100 (ref. 20 seeds)		N=300 (ref. 10 seeds)	
	inconsistent win	missed win	inconsistent win	missed win
3	11%	81%	14%	56%
5	9%	70%	10%	42%
8	8%	60%	9%	38%
10	7%	55%	–	–

Table 7: Solver tasks, N=100. Seeds are best-of-N/method.

Task	method	diff vs best-of-N [95% CI]	effect (σ)	seeds
MAX-3-SAT	FunSearch	**+5.28 [+1.53, +8.93]**	+0.93	15/20
	Evolve	+2.21 [-1.99, +6.18]	+0.39	15/20
	Hedge	+1.28 [-2.87, +5.34]	+0.23	15/20
Planted clique	FunSearch	-2.12 [-18.34, +11.43]	-0.11	15/20
	Evolve	-2.83 [-19.32, +11.02]	-0.15	15/20
	Hedge	-3.38 [-20.48, +11.91]	-0.18	15/20

Table 8: Covering-design ladder. Best-of-N blocks: mean champion. Gap: distance above the record. FunSearch, Evolve: advantage over best-of-N in within-task seed-SD, bold where $p < 0.05$; a superscript gives the seed count where below 10.

	N=100				N=300			
Instance	best-of-N blocks	gap	FunSearch	Evolve	best-of-N blocks	gap	FunSearch	Evolve
$C(13, 6, 3)$ (record 21)	21.6	3%	**+0.83**	+0.21	21.2	1%	+1.42	+1.42
$C(16, 6, 3)$ (record 38)	40.5	7%	**+1.03**	+0.85	38.2	1%	+0.26	+0.00
$C(20, 6, 3)$ (record 71)	82.1	16%	**+1.01**	+0.51	80.6	14%	**+2.13**	**+1.48**
$C(24, 6, 3)$ (record 116)	123.7	7%	-0.31	-0.35	122.0	5%	-0.83	+0.03
$C(28, 6, 3)$ (record 185)	240.3	30%	-0.44	-0.18	234.8	27%	**-0.86**	-0.45
$C(32, 6, 3)$ (record 300)	365.0	22%	+0.68	-0.09	354.8	18%	-0.75	-0.24

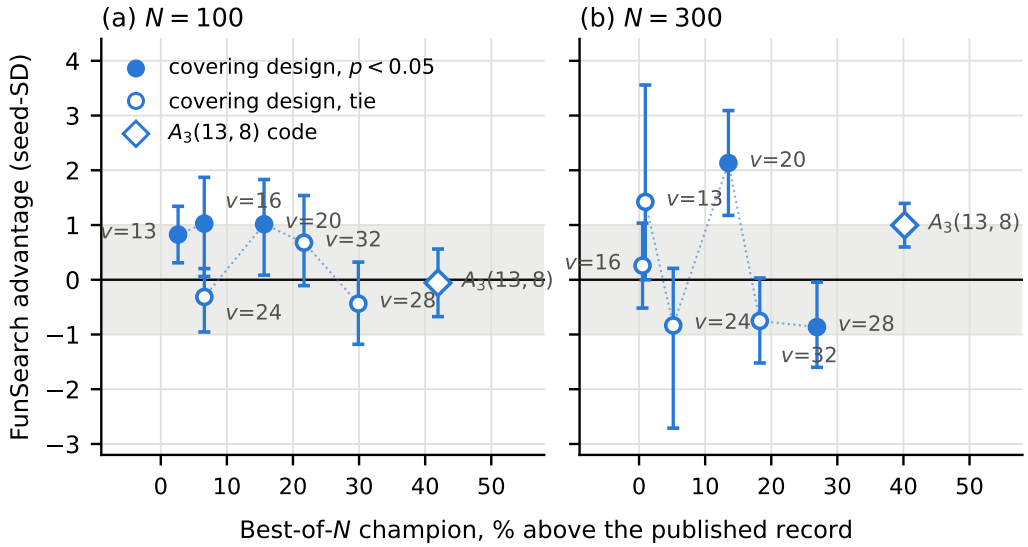


Figure 2: FunSearch's advantage over best-of-N (seed-SD, 95% interval) against how far the best-of-N champion sits above the published record, six $C(v, 6, 3)$ instances at two budgets, with $A_3(13, 8)$ as an open marker. Shaded: ± 1 seed-SD. No monotone relation.