META-DELTA: EVOLVING KNOWLEDGE THROUGH GLOBAL REVIEW FOR LLM-DRIVEN OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM-driven evolutionary optimization accumulates experience about which changes improve a candidate solution. As search progresses, this experience must be interpreted in context: lessons may have limited scope, accumulated guidance may conflict, and repeated refinement may leave alternatives unexplored. We introduce **Meta-Delta**, a framework that jointly evolves candidate solutions and the knowledge guiding their generation. A **Delta Bank** stores structured **Delta Cards** with search lessons, applicability conditions, and supporting evidence. Routine local updates incorporate what each experiment teaches, while Global Review examines the Bank and broader search history to reconsider existing guidance and propose directions for subsequent testing. The updated Bank guides new experiments, whose outcomes further refine its knowledge. Empirically, Meta-Delta achieves the best or tied-best result on 10 of 11 tasks across mathematical, scientific, and systems optimization. On scientific law discovery, it achieves the highest mean numeric-fit score on SCILAWS-REAL, ties for the highest scientific-validity score at displayed precision, and achieves the highest mean structure score on SCILAWS-PARALLEL. Global Review ablations provide evidence that broader knowledge revision helps sustain progress, and case studies show how revised knowledge changes subsequent search and the structure of discovered scientific laws. These results support treating search knowledge as an evolving component of long-horizon optimization.

1 INTRODUCTION

In LLM-driven evolutionary optimization, each evaluated candidate is both a possible solution and evidence for what to try next. Repeated generation and evaluation can improve solutions beyond a single attempt (Romera-Paredes et al., 2024; Novikov et al., 2025), and the accumulated history informs subsequent search through elite solutions (Novikov et al., 2025), semantic modifications (Jiang et al., 2026), or evolving search strategies (Liu et al., 2026). As this history grows, the agent must decide which lessons to retain, when they apply, and how new results change their interpretation. These decisions shape the experiments that follow.

Search lessons are provisional. A successful modification may apply only within a particular solution family; a failed attempt may reflect an implementation error or a limitation of the strategy. Individually reasonable lessons can give conflicting guidance, and repeatedly refining a productive direction can leave alternatives unexplored. As the candidate solutions and evidence change, the agent must reconsider what its accumulated experience implies for the next experiment. This motivates a knowledge state whose claims, applicability conditions, and relationships can be revised throughout search.

Meta-Delta maintains this knowledge at two complementary scales. A Delta Bank stores structured Delta Cards containing lessons, conditions, evidence, and follow-up directions. After each experiment, local curation asks what the observed outcome teaches about the attempted modification. It updates relevant Cards using the latest evaluated and audited evidence. Global Review asks what the accumulated experience suggests trying next, considering the Bank and broader search history together. These two views let the agent learn from individual experiments while reconsidering how its knowledge should guide future search.

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.

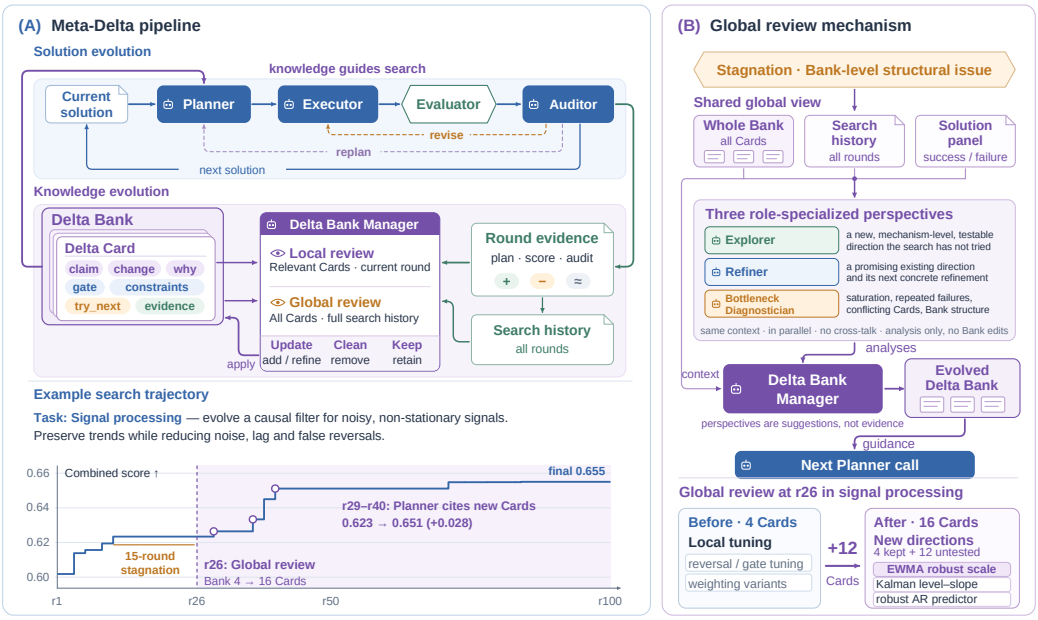


Figure 1: Overview of Meta-Delta. **(A)** Candidate solutions evolve under the guidance of a Delta Bank, which the Delta Bank Manager updates from each audited round. **(B)** On stagnation or a Bank-level issue, Global Review uses three role-specialized perspectives to revise the Bank and open new directions.

When progress stalls or a Bank-level issue arises, Global Review uses independent analyses of the shared history and Bank to help the Manager reconcile conflicting guidance, revisit earlier lessons, and identify directions for further testing. The Planner uses relevant Cards to propose the next modification. New directions remain exploratory until tested, and their outcomes return to the Bank to support, qualify, or reject the hypothesis. This closes the loop between the two levels of evolution: knowledge changes which experiments the agent pursues, and experiments change the knowledge available for subsequent planning.

Meta-Delta achieves the best or tied-best result on 10 of 11 tasks spanning mathematical, scientific, and systems optimization. On the 118-problem SCILAWS-REAL benchmark (Huang et al., 2026), it achieves the highest mean numeric-fit score (S_N) and ties for the highest mean scientific-validity score (S_V) at displayed precision; on SCILAWS-PARALLEL, it achieves the highest mean structure score (S_S) (Table 1; Section 5.1). Comparisons with Global Review disabled show that broader knowledge revision adds value even when routine Delta Bank maintenance remains active (Section 5.2). Card-level case studies connect these outcomes to changes in search guidance and, on SCILAWS-REAL, to the structure of submitted laws (Sections 5.3 and 5.4). These findings suggest that revisiting accumulated guidance can help sustain progress as long-horizon search settles into existing directions.

2 Related Work

LLM-driven evolutionary search. LLM-driven evolutionary search improves programs and algorithmic solutions through repeated generation and evaluation. AlphaEvolve (Novikov et al., 2025) maintains evaluated programs for parent and inspiration sampling, while OpenEvolve (Sharma, 2025) provides an open-source implementation. Recent work improves search control: ShinkaEvolve (Lange et al., 2026) combines improved parent sampling, novelty filtering, and adaptive model selection; AdaEvolve (Cemri et al., 2026) adapts search behavior to observed progress; and EvoX (Liu et al., 2026) jointly evolves candidate solutions and search strategies. Other work changes what experience is retained: DeltaEvolve (Jiang et al., 2026) stores semantic parent-to-child deltas, while PACEvolve (Yan et al., 2026) manages and prunes accumulated context and backtracks when progress stalls. Related guidance appears as verbal reflections (Ye et al., 2024), concept libraries (Grayeli et al.,

2024), cycle summaries (Wan et al., 2025), or mutation advice (Volkov et al., 2026); AEvo (Zhang et al., 2026b) revises the evolution procedure, while TTT-Discover (Yuksekgonul et al., 2026) adapts model weights at test time. Meta-Delta instead represents evaluated experience as revisable claims with applicability conditions and supporting evidence, combining routine maintenance with broader reconsideration of the Delta Bank.

Meta-agents and automated self-improvement. Meta-agents improve the systems that perform a task by modifying different components of the agent itself. ADAS (Hu et al., 2025) and the Darwin Gödel Machine (Zhang et al., 2026a) evolve agent designs or code, while GEPA (Agrawal et al., 2026) and Promptbreeder (Fernando et al., 2023) adapt prompts from execution experience. Meta-Harness (Lee et al., 2026) searches over information-processing harnesses, and ALMA (Xiong et al., 2026) meta-learns agent memory designs expressed as code. These methods target agent architectures, code, prompts, harnesses, or memory mechanisms. Meta-Delta instead revises the problem-specific knowledge accumulated during optimization, changing the claims and directions that guide subsequent search.

Agent memory and evolving knowledge. Agents reuse experience as self-reflections (Shinn et al., 2023) or trajectory-derived insights (Zhao et al., 2024; Ouyang et al., 2026). ACE (Zhang et al., 2026c) curates an evolving context playbook, while AlphaOPT (Kong et al., 2026) refines insights with applicability conditions. Most closely related, KSI (Wang et al., 2026) treats an evidence-backed knowledge base, rather than the agent, as the persistent object of improvement across tasks. Periodic reflection over memory streams (Park et al., 2023) and meta-review of role-specialized agents (Gottweis et al., 2026) provide precedents for Global Review. Meta-Delta brings these ideas into single-problem optimization, grounding Bank claims in evaluated outcomes and treating review perspectives as suggestions rather than evidence.

3 METHOD

3.1 OVERVIEW

Meta-Delta couples a candidate-solution loop with a knowledge-update loop (Figure 1A). The candidate loop plans, executes, evaluates, and audits modifications. The knowledge loop maintains a Delta Bank whose Cards inform subsequent plans. Routine curation updates relevant knowledge using the current round's evidence. When search stagnates or a Bank-level issue arises, Global Review considers the accumulated Bank and search history together. Both processes update the guidance available to the Planner, and subsequent experiments provide evidence for further revision. Algorithm 1 summarizes the overall procedure.

3.2 SOLUTION EVOLUTION LOOP

Planning. At round k, the Planner inspects a selected parent solution and retrieves relevant Cards from the current Delta Bank $B^{(k)}$. It chooses one modification direction and produces a concrete natural-language plan. Delta Cards are treated as evidence rather than fixed instructions: the Planner may use the Cards that are relevant to the current solution, ignore those that do not apply, or proceed without any Card. It also records which Cards materially informed the proposed plan.

Execution and evaluation. The Executor implements the plan and produces a new candidate solution. We separate planning from execution so that strategy selection remains with the Planner, while the Executor focuses on faithfully realizing the proposed change. The candidate is then passed to the task-specific Evaluator, which returns its objective score and any available evaluation feedback.

Auditing. The Auditor determines whether the candidate is a valid and informative test of the proposed modification. It can ACCEPT the experiment, request a REVISE when the strategy remains valid but its implementation should be corrected, or request a REPLAN when the strategy itself should be reconsidered. A REVISE is routed back to the Executor, while a REPLAN returns to the Planner. The Auditor also filters invalid or exploitative candidates and records whether the executed change actually implements the proposed plan. After the round's audit and revision process concludes, the

Algorithm 1 Meta-Delta: co-evolving solutions and search knowledge.

Require: task with evaluator E, initial solution x_0, number of rounds K
solution archive $\mathcal{P} \leftarrow \{x_0\}$; Delta Bank $B^{(1)} \leftarrow \emptyset$; history $\mathcal{H} \leftarrow \emptyset$
for $k = 1, \ldots, K$ **do**
    Level 1 | candidate-solution evolution
    $x \leftarrow \textsc{SelectParent}(\mathcal{P})$; $C \leftarrow \textsc{RetrieveCards}(B^{(k)}, x)$; $\pi \leftarrow \textsc{Planner}(x, C)$
    $\text{execFeedback} \leftarrow \emptyset$
    while the per-round attempt budget remains **do**
        $x' \leftarrow \textsc{Executor}(x, \pi, \text{execFeedback})$; $(\text{score}, \text{evalFeedback}) \leftarrow E(x')$
        $(\text{action}, \text{auditFeedback}) \leftarrow \textsc{Auditor}(x, \pi, x', \text{score}, \text{evalFeedback})$
        if $\text{action} = \textsc{Accept}$ **then break**;
        else if $\text{action} = \textsc{Revise}$ **then** $\text{execFeedback} \leftarrow \text{auditFeedback}$
        else $\pi \leftarrow \textsc{Planner}(x, C, \text{auditFeedback})$; $\text{execFeedback} \leftarrow \emptyset$
    end while
    update $\mathcal{P}$ and $\mathcal{H}$ with valid candidates and audited evidence_k
    Level 2 | Delta Bank knowledge evolution
    $L_k \leftarrow \textsc{LocalView}(B^{(k)}, \text{evidence}_k)$
    $\tilde{B} \leftarrow \textsc{BankManager}_{\text{routine}}(L_k)$; $B^{(k+1)} \leftarrow \tilde{B}$
    $\text{bankRequest} \leftarrow \textsc{ReviewDecision}(\tilde{B}, \mathcal{H})$; $\text{stagnated} \leftarrow \textsc{Stagnated}(k)$
    if $(\text{bankRequest} \wedge \textsc{ReviewEligible}(k)) \vee \text{stagnated}$ **then**
        $G \leftarrow \textsc{GlobalContext}(\tilde{B}, \mathcal{H}, \mathcal{P})$
        $(p_{\text{exp}}, p_{\text{ref}}, p_{\text{bot}}) \leftarrow \textsc{Parallel}[\textsc{Explorer}, \textsc{Refiner}, \textsc{BottleneckDiagnostician}](G)$
        $B^{(k+1)} \leftarrow \textsc{BankManager}_{\text{global}}(G, p_{\text{exp}}, p_{\text{ref}}, p_{\text{bot}})$
    end if
end for
return the best evaluated solution in $\mathcal{P}$

plan, implementation, evaluation, and audit are collected as round evidence. Both successful and unsuccessful attempts can provide useful evidence about a search strategy. This evidence is passed to the Delta Bank Manager for routine curation before the next round of solution evolution.

3.3 Delta Bank and Delta Cards

Knowledge representation. The Delta Bank stores reusable search knowledge as structured Delta Cards. Supported Cards capture lessons distilled from evaluated experience; exploratory Cards record hypotheses for subsequent testing. Each Card records a proposed search direction together with the conditions and evidence that qualify its reuse. Figure 2 shows how real Cards are revised during Global Review and subsequently updated by downstream evidence.

Evidence-driven maintenance. After each round, the Delta Bank Manager receives the audited round evidence and performs a local curation step. It may add a new Card when the experiment reveals a distinct reusable lesson, update an existing Card when new evidence refines an earlier conclusion, remove knowledge that has become redundant or contradicted, or leave the Delta Bank unchanged when the current result does not justify an update. The Delta Bank Manager does not re-evaluate the candidate itself; its role is to maintain the knowledge extracted from evaluated search outcomes. The updated Delta Bank is then available to the Planner in the next round. The Planner considers each Card in relation to the current parent and its applicability conditions when deciding how it should influence the next modification. Testing an exploratory Card can support the hypothesis, narrow its scope, or provide evidence against it; the Manager incorporates that outcome during local curation.

3.4 Global Review: Evolving the Knowledge Layer

Trigger and scope. Global Review reconsiders guidance across the accumulated Bank and search history (Figure 1B). It is triggered when search stagnates or the Bank Manager identifies a structural issue that routine local curation cannot adequately resolve. Its shared context includes the current Bank and the broader search history, with representative successful and unsuccessful solutions, allowing relationships among lessons to be examined together.

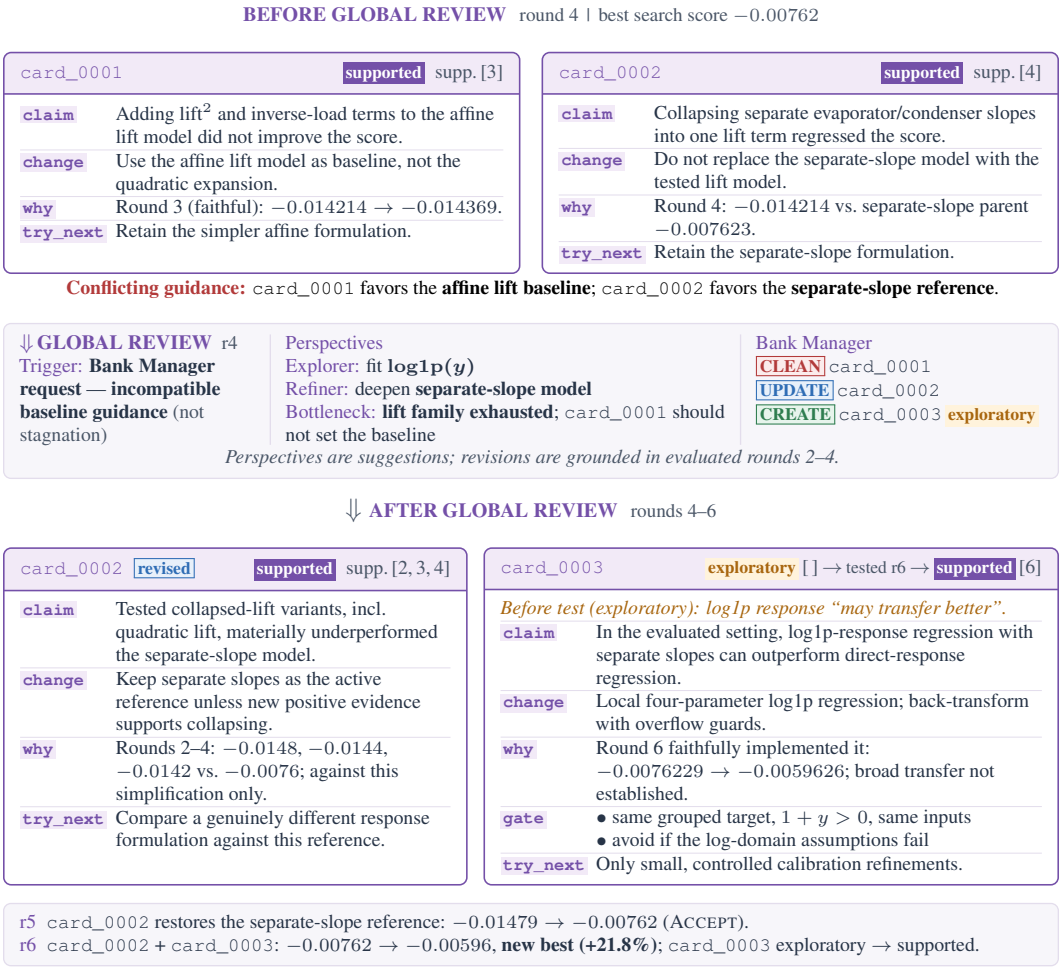


Figure 2: Knowledge revision in a Chiller compressor-work run (SCILAWS-REAL, seed 0). Global Review resolves conflicting Cards and introduces an exploratory log-response Card later promoted by downstream evidence. Card text is abridged; supp. denotes supporting iterations; scores are training-side, not held-out S_N.

Complementary analyses. Three LLM calls independently examine this context under complementary briefs. The *Explorer* seeks a new, mechanism-level, testable direction that the search has not meaningfully explored. The *Refiner* identifies a promising existing direction and proposes its next concrete refinement. The *Bottleneck Diagnostician* identifies saturation, repeated failures, misleading or conflicting knowledge, and Bank structure that should be reorganized. These parallel calls cannot see one another's outputs or modify the Bank; their analyses are passed to the same Manager that performs routine curation.

Bank revision and subsequent testing. The Manager can retain useful knowledge, refine claims and applicability conditions, remove misleading or redundant guidance, reorganize Cards jointly, and introduce exploratory directions. The perspectives provide suggestions; revisions to established knowledge remain grounded in evaluated history, and untested directions remain exploratory until supported by evaluation. The updated Bank $B^{(k+1)}$ then informs subsequent Planner calls. The Planner selects a direction in the context of the current solution, and execution, evaluation, and auditing supply evidence for the next Bank update. This links the reconsideration of accumulated guidance to experiments that can support, qualify, or reject it.

4 EXPERIMENTAL SETUP

Tasks. We evaluate Meta-Delta on 11 optimization tasks spanning mathematics, scientific modeling, and systems optimization, and on both settings of SCILAWS-BENCH. The 11-task set is adapted from benchmark tasks used in recent LLM-driven evolutionary discovery work (Liu et al., 2026; Qu et al., 2026); Symbolic Regression is drawn from LLM-SRBench (Shojaee et al., 2025). The set includes Heilbronn Triangle, Circle Packing, Erdős Min Overlap, Third Autocorrelation Inequality, two Minimizing-Max-Min-Distance (MMD) instances, Signal Processing, Symbolic Regression, EPLB, Cloudcast, and Transaction Scheduling. We further evaluate all 118 problems in SCILAWS-REAL and SCILAWS-PARALLEL (Huang et al., 2026). SCILAWS-REAL is the fixed-data setting, while SCILAWS-PARALLEL is active experimentation: each task hides a newly synthesized structural variant of a published law that the solver must infer by querying a simulator.

Baselines. Across all three settings, we compare against Parallel Sampling (Best-of-N), Greedy Refine, OpenEvolve (Sharma, 2025), DeltaEvolve (Jiang et al., 2026), and ShinkaEvolve (Lange et al., 2026). Parallel Sampling returns the best of independent samples, while Greedy Refine repeatedly revises the current best solution using evaluator feedback. The evolutionary baselines respectively use AlphaEvolve-style program search, semantic parent-to-child deltas, and population-based search with diverse inspiration sampling. We use the official ShinkaEvolve implementation and SkyDiscover's native OpenEvolve port; because no official DeltaEvolve implementation was publicly available at the time of our experiments, we reproduced it from the original paper.

Implementation Details. Unless otherwise specified, all methods use GPT-5.6-Luna with medium reasoning effort; Global Review uses three independent calls with the same setting. We use three seeds (0, 1, 2), with 100 rounds on the 11-task set and 30 rounds on both SCILAWS settings. On the 11-task set, all methods share the same initial solution and fixed task-specific evaluator. On SCILAWS-REAL, each run instead generates its initial solution from the task description and training data and constructs a training-side search evaluator that may be revised during evolution. The selected solution is scored only afterward by the official held-out evaluator, which is identical across methods and never exposed during search. SCILAWS-PARALLEL starts without observations: all methods use the same method-independent acquisition procedure and fixed query budget, followed by 30 rounds of search with no further simulator access. The generating law is never exposed, and S_S is computed only after search. Selected prompt templates and further task details are provided in the appendix.

Evaluation Metrics. For the 11-task set, we report each task's native objective using best@3 over three independent runs, with $\uparrow$ / $\downarrow$ indicating whether higher or lower is better. For the two MMD tasks, all methods reach the same best-known numerical plateau up to floating-point precision; later strict improvements are only 10^{-14}–10^{-16}-scale. We therefore report round-to-plateau, the first round at which the best-so-far ratio squared matches the run's final plateau to six decimal places, using best@3 across seeds.

For SCILAWS-REAL, we use the benchmark's official numeric-fit (S_N) and scientific-validity (S_V) scores. For SCILAWS-PARALLEL, we use the structure score (S_S), which compares the selected and generating laws at five levels: 0 (invalid or unrelated), 0.25 (relevant variables or trends), 0.5 (published base form), 0.75 (base form with most added terms), and 1 (complete hidden structure). SCILAWS scores are reported as percentages, and the main table reports the mean of run-level task aggregates across seeds 0, 1, and 2. The 11-task set follows prior best-of-three reporting for comparability, while SCILAWS uses task-averaged evaluation over 118 problems to assess broader robustness.

5 RESULTS AND ANALYSIS

5.1 OVERALL OPTIMIZATION PERFORMANCE

11-task set. Meta-Delta achieves the best or tied-best result at the displayed precision on 10 of the 11 tasks, with strong results across mathematical, scientific, and systems problems (Table 1). On the

Table 1: Overall optimization results. The 11-task set reports native objectives using best@3 over three seeds; MMD entries are round-to-plateau. SCILAWS-REAL (S_N, S_V) and SCILAWS-PARALLEL (S_S) report pooled All means (%) over three runs. ↑/↓ indicate optimization direction; bold marks the best displayed value. DE/OE/SE/PS denote DeltaEvolve/OpenEvolve/ShinkaEvolve/Parallel Sampling; Greedy denotes Greedy Refine; Full denotes Meta-Delta and w/o GR its variant without Global Review.

Task / Metric	Full	w/o GR	DE	OE	SE	PS	Greedy
11-task set							
Signal Proc. ↑	**0.6550**	0.5415	0.5969	0.6078	0.5982	0.5596	0.6062
Heilbronn Triangle ↑	**0.0365**	0.0331	0.0359	0.0339	0.0352	0.0341	**0.0365**
Circle Packing ↑	**2.6360**	2.6302	**2.6360**	2.6343	2.6343	2.5414	**2.6360**
Symbolic Reg. ↓	2.4e-5	1.3e-4	1.1e-4	1.3e-4	**1.3e-5**	1.4e-3	3.6e-5
EPLB ↑	**0.1470**	0.1438	0.1300	0.1299	0.1280	0.1275	0.1284
Cloudcast ↓	**618.0699**	627.0956	618.0734	618.0817	**618.0699**	689.4941	627.0956
Transaction Sched. ↑	**4366.81**	4291.85	4329.00	4166.67	4347.83	3906.25	4347.83
Erdős Min Overlap ↓	**0.3809**	**0.3809**	0.3810	0.3810	**0.3809**	**0.3809**	**0.3809**
Third Autocorr. ↓	**1.4547**	1.4563	1.4582	1.4594	1.4586	1.4599	1.4577
MMD-16-2 ↓	**r1**	r2	r2	r3	**r1**	**r1**	**r1**
MMD-14-3 ↓	**r1**	r3	r4	r3	**r1**	r11	r3
SCILAWS-REAL							
S_N All ↑	**49.30**	47.81	49.12	48.77	48.71	46.68	47.00
S_V All ↑	**84.24**	**84.24**	79.52	79.59	84.20	81.57	69.37
SCILAWS-PARALLEL							
S_S All ↑	**48.02**	47.81	47.53	45.55	47.25	47.53	42.58

two MMD tasks, where all methods reach the same plateau, Meta-Delta reaches it in the first round on both, as does ShinkaEvolve.

SCILAWS-REAL. On the pooled 118-task benchmark, Meta-Delta achieves the highest mean numeric-fit score, with S_N All of 49.30, compared with 49.12 for the strongest baseline, DeltaEvolve. It also ties for the highest mean scientific-validity score at displayed precision, with S_V All of 84.24, matched by the variant without Global Review (84.24) and followed by ShinkaEvolve (84.20). Disabling Global Review lowers S_N All from 49.30 to 47.81, while S_V All remains tied at 84.24. Split-level results are reported in Appendix A.2.

SCILAWS-PARALLEL. Meta-Delta achieves the highest mean S_S on SCILAWS-PARALLEL (48.02%), followed by the variant without Global Review (47.81%) and by DeltaEvolve and Parallel Sampling (47.53% each); the remaining baselines score between 42.58% and 47.25%.

5.2 CONTRIBUTION OF GLOBAL REVIEW

We compare the full Meta-Delta framework with a variant in which Global Review is disabled for the entire search run ("w/o GR" in Table 1); routine Delta Bank curation remains enabled, and all other components and settings are unchanged.

Gains beyond local curation. As shown in Table 1, enabling Global Review leads to better results on ten of the eleven tasks: a better final score on eight tasks and an earlier plateau on both MMD tasks, with Erdős Min Overlap tied at the displayed precision.

Sustaining progress later in search. Beyond the final scores, the trajectories show a difference in late-stage search behavior. Across the reconstructed trajectories, the median round at which the search reaches its final meaningful plateau is 35 with Global Review, compared with 13 when Global Review is disabled. Moreover, 39% of the full-framework trajectories still find meaningful improvements after round 50, compared with 18% without Global Review. This pattern is particularly

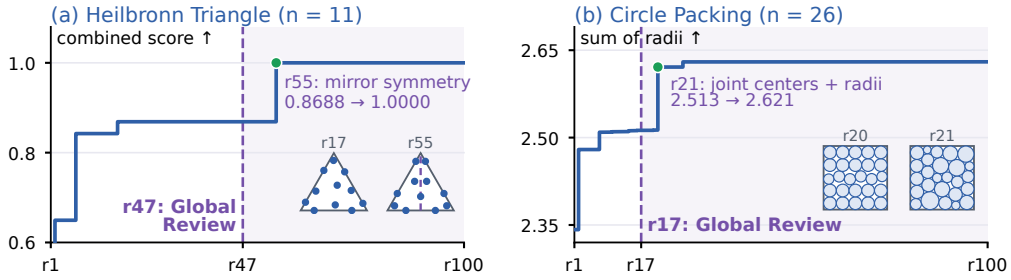


Figure 3: Best-so-far trajectories of two 11-task runs; the dashed line marks Global Review and the green circle the first new best whose plan cites a Card it created or revised. Insets show solutions before and after (mirror axis in (a)); 1.0000 is at displayed precision.

visible on tasks such as Signal Processing, Heilbronn Triangle, and Symbolic Regression, where some Meta-Delta runs continue to improve well after the corresponding ablated runs have stagnated. These trajectories suggest that Global Review helps sustain progress after search has begun to settle into existing directions.

5.3 Knowledge Revision in Optimization Search

We trace how Bank changes made by Global Review alter the Planner's subsequent decisions in two 11-task runs (Figure 3); fuller descriptions and a further case are given in Appendix A.3.

Revising existing knowledge. On Heilbronn Triangle, search stalls at 0.8688 while all eight Cards concern settings of the same differential-evolution/SLSQP pipeline. Global Review adds no Cards but revises five existing ones; the Card on the optimizer budget now directs follow-up work toward symmetry-reduced or alternative parameterizations instead of further budget tuning. The Planner instantiates this guidance as mirror symmetry, reducing the point-coordinate search space from 22 to 11 variables, and a plan citing only the revised Card raises the combined score to 1.0000 at displayed precision (Figure 3a).

Changing the search structure. On Circle Packing, the construction stays close to a fixed grid and the sum of radii stalls near 2.513. Global Review narrows the existing Card to local polishing and adds three exploratory directions: exact radius allocation by linear programming, joint optimization of all 26 centers and radii, and an alternative contact topology. A plan citing only the linear-programming and joint-optimization Cards replaces the grid with an irregular contact arrangement and raises the sum of radii from 2.513 to 2.621 (Figure 3b). In each case, the review changes the guidance available to the Planner, and subsequent experiments determine which directions are retained.

5.4 Knowledge Revision in Scientific-Law Discovery

We follow two SCILAWS-REAL runs in which a Global Review changes the Bank and the resulting change appears in the submitted law. Figure 4 summarizes the structural change in each law, and Figure 5 shows when the review occurs and how the reviewed direction propagates through the run; two further cases are given in Appendix A.4.

Resolving conflicting guidance. On baseflow recession, two supported Cards give conflicting guidance for the same event-anchored fitting setting. The Bank Manager requests a review, which reorganizes the Bank: it revises both Cards and adds an exploratory Card, proposed by the Explorer, that derives recession from a nonlinear reservoir, $dQ/dt = -kQ^p$. The submitted law comes from a plan citing this Card together with a revised one. Integrated per event, it remains in the same hyperbolic family as the law it replaces, but its recession rate now depends on each event's initial discharge Q_0 rather than only on fixed per-group constants ($S_N = 50.2$, $S_V = 100$).

(a) Baseflow recession: conflicting guidance

Bank	Two supported Cards give conflicting guidance.
Review	**REORGANIZE** (Bank request); new Card: $dQ/dt = -kQ^p$
Before	$Q = Q_0\,(1 + at)^{-b}$
After	$Q = Q_0\,[1 + (p-1)\,k\,Q_0^{p-1}\,t\,]^{-1/(p-1)}$ *recession rate depends on each event's Q_0*
Submitted	$S_N = 50.2 \mid S_V = 100$

(b) Heligman–Pollard: new explanatory axis

Bank	Eight Cards on variants of one age–period model.
Review	**UPDATE** (stagnation); new Card: birth-cohort (Lexis) axis
Before	period–age interaction $t\,x/50$
After	cohort term q^2, $q = (\text{year} - \text{age} - 1950)/50$ *cohort term persists into the submitted law*
Submitted	$S_N = 69.8 \mid S_V = 100$

Figure 4: Global Review changes the structure of submitted laws in two SCILAWS-REAL runs; trajectories are in Figure 5.

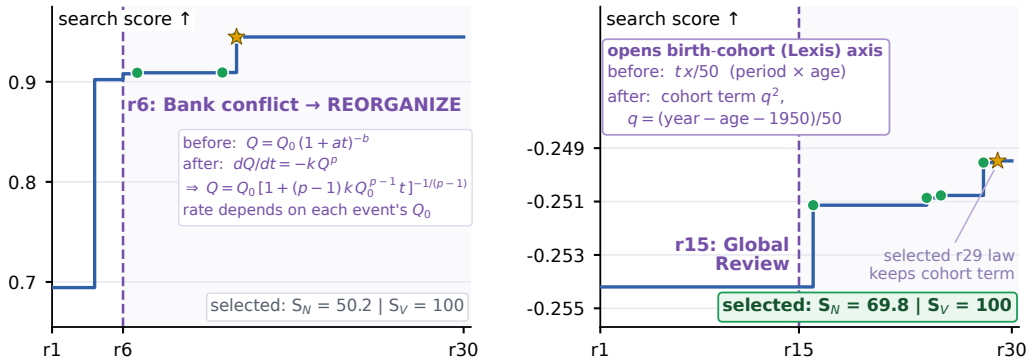


Figure 5: Best-so-far trajectories for the two SCILAWS-REAL cases in Figure 4. Left: Baseflow recession; right: Heligman–Pollard mortality. Dashed lines mark the highlighted Global Review, green circles accepted new bests citing Cards created or revised by that review, and stars the submitted programs. Later round-29 reviews are omitted.

Opening a new explanatory axis. On Heligman–Pollard mortality, no candidate improves on the initial age–period model before the review, and the eight Cards record negative or neutral results about variants of that model. The stagnation-triggered review adds an exploratory Card proposing a birth-cohort (Lexis) component based on year minus age. The next plan replaces the period–age interaction $t\,x/50$ with a cohort term q^2, $q = (\text{year} - \text{age} - 1950)/50$, giving the run's first improvement, and the cohort term persists into the submitted law ($S_N = 69.8$, $S_V = 100$). Because the cohort Card is always cited together with older Cards, we attribute the introduction of the cohort direction, not every later gain, to Global Review.

6 Conclusion and Discussion

Meta-Delta treats search knowledge as an evolving object alongside candidate solutions. Delta Cards make reusable lessons explicit, while Global Review reconsiders accumulated guidance when search begins to settle into existing directions. Across mathematical, scientific, and systems optimization, our results support the value of this second level of evolution for sustaining long-horizon progress. The case studies further show how revising knowledge can change the experiments pursued next and, in scientific-law discovery, the structure of submitted laws, while the outcomes of those experiments further refine the Delta Bank. This feedback loop enables agents to improve both their solutions and the knowledge used to generate them.

Meta-Delta still depends on the quality of evaluated search outcomes: despite auditing and evidence grounding, systematic evaluator errors or biases may propagate into the Delta Bank and reinforce misleading knowledge. Our evaluation spans mathematical, scientific, and systems optimization, including SCILAWS-REAL and SCILAWS-PARALLEL, but broader task domains are needed to assess generalization. Finally, substantially longer runs may produce larger Delta Banks and place greater demands on retrieval, organization, and repeated Global Review.

AI USE STATEMENT

In this work, we used generative AI tools for research ideation or execution. We have not used generative AI tools to draft sections of the paper, and generating synthetic datasets and proving mathematical claims are not applicable to this work. Additionally, we used generative AI tools to aid or polish writing and for retrieval and discovery (e.g., finding related work). We have reviewed all AI-assisted work. AI-assisted research outputs and code were reviewed and tested by the authors, and retrieved references and literature-based claims were manually verified. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Meta-Delta is designed for automated optimization in settings with executable evaluators. A potential risk is that errors or biases in an evaluator may be propagated through the Delta Bank, or that the search may exploit evaluator weaknesses rather than produce genuinely better solutions. We mitigate these risks by auditing candidate changes, grounding supported Delta Cards in evaluated evidence, and treating Global Review outputs as hypotheses until they are tested. These safeguards do not eliminate evaluator-induced failure modes; applications in consequential settings should therefore use independent validation and appropriate human oversight.

REPRODUCIBILITY STATEMENT

Anonymous code, configurations, prompt templates, and reproduction scripts are available at `https://anonymous.4open.science/r/meta-delta-release-C818`. The repository contains the Meta-Delta implementations used for the 11-task set, SCILAWS-REAL, and SCILAWS-PARALLEL, together with the corresponding launch configurations and benchmark adapters. SCILAWS-BENCH data and benchmark-side simulation and evaluation infrastructure are not bundled; setup instructions for connecting an external benchmark checkout are provided in the repository. Real runs require access to the corresponding LLM API and may incur API costs.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, et al. Gepa: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 8479–8565, 2026.

Mert Cemri, Shubham Agrawal, Akshat Gupta, Shu Liu, Audrey Cheng, Qiuyang Mang, Ashwin Naren, Lutfi Eren Erdogan, Koushik Sen, Matei Zaharia, et al. Adaevolve: Adaptive llm driven zeroth-order optimization. *arXiv preprint arXiv:2602.20133*, 2026.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*, 2023.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, Artiom Myaskovsky, Grzegorz Glowaty, Felix Weissenberger, Alessio Orlandi, Dan Popovici, et al. Accelerating scientific discovery with co-scientist. *Nature*, pp. 1–3, 2026.

Arya Grayeli, Atharva Sehgal, Omar Costilla-Reyes, Miles Cranmer, and Swarat Chaudhuri. Symbolic regression with a learned concept library. *Advances in Neural Information Processing Systems*, 37: 44678–44709, 2024.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

Yiming Huang, Ziche Liu, Zhuohang Wu, Yiqian Wang, Junxia Cui, Xinkai Zou, Linjun Mao, Nan Huang, Naicheng Yu, Kaijie Zhu, et al. Can llms discover scientific laws in real and parallel worlds? *arXiv preprint arXiv:2609.01552*, 2026.

Jiachen Jiang, Tianyu Ding, and Zhihui Zhu. Deltaevolve: Accelerating scientific discovery through momentum-driven evolution. *arXiv preprint arXiv:2602.02919*, 2026.

Minwei Kong, Ao Qu, Xiaotong Guo, Wenbin Ouyang, Chonghe Jiang, Han Zheng, Yining Ma, Dingyi Zhuang, Yuhan Tang, Junyi Li, et al. Alphaopt: Formulating optimization programs with self-improving llm experience library. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 2389–2400, 2026.

Robert Lange, Yuki Imajuku, and Edoardo Cetin. Shinkaevolve: Towards open-ended and sample-efficient program evolution. In *International Conference on Learning Representations*, volume 2026, pp. 74026–74078, 2026.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026.

Shu Liu, Shubham Agarwal, Monishwaran Maheswaran, Mert Cemri, Zhifei Li, Qiuyang Mang, Ashwin Naren, Ethan Boneh, Audrey Cheng, Melissa Z Pan, et al. Evox: Meta-evolution for automated discovery. *arXiv preprint arXiv:2602.23413*, 2026.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, et al. Reasoningbank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations*, volume 2026, pp. 94327–94354, 2026.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Ao Qu, Han Zheng, Zijian Zhou, Yihao Yan, Yihong Tang, Shao Yong Ong, Fenglu Hong, Kaichen Zhou, Chonghe Jiang, Minwei Kong, et al. Coral: Towards autonomous multi-agent evolution for open-ended discovery. *arXiv preprint arXiv:2604.01658*, 2026.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

Asankhaya Sharma. Openevolve: an open-source evolutionary coding agent, 2025. URL `https://github.com/algorithmicsuperintelligence/openevolve`.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Parshin Shojaee, Ngoc-Hieu Nguyen, Kazem Meidani, Amir Barati Farimani, Khoa D Doan, and Chandan K Reddy. Llm-srbench: A new benchmark for scientific equation discovery with large language models. *arXiv preprint arXiv:2504.10415*, 2025.

Viktor Volkov, Valentin Khrulkov, Andrey V Galichin, Danil Sivtsov, Nikita Glazkov, Olga Volkova, Konstantin Pchelin, Iaroslav Bespalov, Dmitry V Dylov, Petr Anokhin, et al. Evomem: Memory-augmented evolution for code optimization. *arXiv preprint arXiv:2608.10795*, 2026.

Chunhui Wan, Xunan Dai, Zhuo Wang, Minglei Li, Yanpeng Wang, Yinan Mao, Yu Lan, and Zhiwen Xiao. Loongflow: Directed evolutionary search via a cognitive plan-execute-summarize paradigm. *arXiv preprint arXiv:2512.24077*, 2025.

Xuefei Julie Wang, Lauren Hyoseo Yoon, Chengrui Qu, Amanda Zichang Wang, Atharva Sehgal, Eric Mazumdar, and Yisong Yue. Knowledge-centric self-improvement. *arXiv preprint arXiv:2607.19592*, 2026.

Yiming Xiong, Shengran Hu, and Jeff Clune. Learning to continually learn via meta-learning agentic memory designs. *arXiv preprint arXiv:2602.07755*, 2026.

Minghao Yan, Bo Peng, Benjamin Coleman, Ziqi Chen, Zhouhang Xie, Shuo Chen, Zhankui He, Noveen Sachdeva, Isabella Ye, Weili Wang, et al. Pacevolve: Enabling long-horizon progress-aware consistent evolution. *arXiv preprint arXiv:2601.10657*, 2026.

Haoran Ye, Jiarui Wang, Zhiguang Cao, Federico Berto, Chuanbo Hua, Haeyeon Kim, Jinkyoo Park, and Guojie Song. Reevo: Large language models as hyper-heuristics with reflective evolution. *Advances in neural information processing systems*, 37:43571–43608, 2024.

Mert Yuksekgonul, Daniel Koceja, Xinhao Li, Federico Bianchi, Jed McCaleb, Xiaolong Wang, Jan Kautz, Yejin Choi, James Zou, Carlos Guestrin, et al. Learning to discover at test time. *arXiv preprint arXiv:2601.16175*, 2026.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pp. 104223–104294, 2026a.

Jiayi Zhang, Yongfeng Gu, Jianhao Ruan, Maojia Song, Yiran Peng, Zhiguang Han, Jinyu Xiang, Zhitao Wang, Caiyin Yang, Yixi Ouyang, et al. Harnessing agentic evolution. *arXiv preprint arXiv:2605.13821*, 2026b.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, et al. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations*, volume 2026, pp. 86069–86100, 2026c.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

A Appendix

A.1 Task Details

Table 2: Summary of the 11-task evaluation set.

Task	Direction	Brief Description
Signal Processing	Higher	Optimizes a causal sliding-window filter to suppress noise while preserving signal dynamics, tracking accuracy, and low lag.
Heilbronn Triangle	Higher	Places 11 points in an equilateral triangle to maximize the minimum area formed by any three points.
Circle Packing	Higher	Places 26 non-overlapping circles in a unit square to maximize the sum of their radii.
Symbolic Regression	Lower	Fits a symbolic equation for the LLM-SRBench `phys_osc/PO0` problem by minimizing training-set NMSE after coefficient fitting.
EPLB	Higher	Optimizes expert replication and GPU placement to balance MoE load while reducing rebalancing time.
Cloudcast	Lower	Routes broadcasts across five cloud scenarios to minimize summed egress and VM-time cost.
Transaction Scheduling	Higher	Orders transactions in three workloads to minimize their summed makespan, for which the evaluator reports a reciprocal score.
Erdős Min Overlap	Lower	Finds a discretized step function that minimizes the upper bound C_5 on the Erdős minimum overlap constant.
Third Autocorrelation Inequality	Lower	Finds a discretized function that minimizes the upper-bound ratio C_3 in the third autocorrelation inequality.
MMD-16-2	Lower	Places 16 points in two dimensions to minimize the squared ratio of maximum to minimum pairwise distance.
MMD-14-3	Lower	Places 14 points in three dimensions to minimize the squared ratio of maximum to minimum pairwise distance.

A.2 SCILAWS-REAL Split Results

Table 3 breaks down the SCILAWS-REAL scores in Table 1 by task split, using the same mean across three independent runs. **Single** contains the 66 single-group tasks, which use a single global law and coefficient set; **Multi** contains the 52 multi-group tasks, which share a functional form with a small number of group-specific coefficients; **All** pools the 118 tasks.

Table 3: SCILAWS-REAL split results, as means across three independent runs (seeds 0, 1, and 2). For each seed, each score is aggregated over that seed's tasks in the given split, and the three seed-level aggregates are then averaged. All pools the 118 tasks; with complete observations it is the 66/52 task-count-weighted combination of Single and Multi. The few cells for which validity judging produced no score are excluded from the corresponding S_V aggregate. Best values at the displayed precision are in bold.

Metric	Meta-Delta Full	w/o Review	DE	OE	SE	PS	Greedy
S_N Single ↑	**49.15**	47.79	48.90	48.67	48.66	46.07	46.09
S_N Multi ↑	**49.48**	47.84	49.41	48.90	48.77	47.45	48.16
S_N All ↑	**49.30**	47.81	49.12	48.77	48.71	46.68	47.00
S_V Single ↑	86.92	**87.31**	81.25	81.71	86.75	80.52	64.51
S_V Multi ↑	80.84	80.34	77.31	76.88	80.96	**82.92**	75.76
S_V All ↑	**84.24**	**84.24**	79.52	79.59	84.20	81.57	69.37

A.3 Case Studies of Meta-Delta on the 11-Task Set

We select representative trajectories for mechanistic clarity rather than the highest-scoring run on each task. In each case, a stagnation-triggered Global Review changes the Delta Bank, and a later

Planner call that cites the revised or newly created Cards yields a substantial accepted improvement. Heilbronn Triangle and Circle Packing are shown in Figure 3; Figure 6 shows Cloudcast. We focus on the pre-review guidance, the Bank revision, and the subsequent Card-guided change.

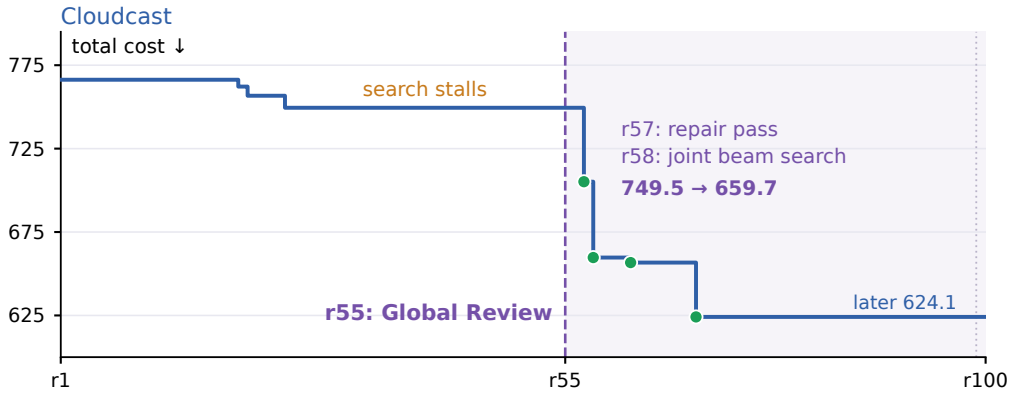


Figure 6: Best-so-far trajectory over all 100 rounds of the Cloudcast run (lower cost is better). The dashed purple line marks the highlighted Global Review, the dotted grey line marks another review in the same run, and green circles mark accepted new bests whose Planner plan cites a Card created or revised by the highlighted review.

Revising existing knowledge: symmetry reduction (Heilbronn Triangle). Before the round-47 review, search stalls at 0.8688; the review adds no Cards but revises five of the eight, all of which concern settings of the same differential-evolution/SLSQP pipeline. In particular, the Card on the optimizer budget broadens its follow-up guidance to test symmetry-reduced or alternative parameterizations as structural changes rather than further budget tuning. The Planner instantiates this direction as mirror symmetry about the triangle's vertical axis: five mirrored point pairs plus one on-axis point, reducing the point-coordinate search space from 22 to 11 variables while leaving the optimizer unchanged. The first implementation, at round 53, contains a variable-count error caught by the Auditor; this is an implementation failure rather than evidence against the idea. At round 55, the Planner cites only the revised Card, and the accepted candidate raises the combined score from 0.8688 to 1.0000 at displayed precision; later rounds only numerically polish this solution.

Opening a global geometric strategy: joint center and radius optimization (Circle Packing). Before the review, the construction remains close to a fixed grid and the sum of radii stalls near 2.5125. At round 17, Global Review narrows the existing Card to local polishing and adds three exploratory directions: exact radius allocation by linear programming (LP) for fixed centers, joint optimization of all 26 centers and radii, and an alternative contact topology. The LP Card gives a small gain at round 20. At round 21, the Planner cites only the LP and joint-optimization Cards. The Auditor returns the first implementation because a boundary repair incorrectly shrinks the radii; the corrected second attempt is accepted and increases the sum of radii from 2.513 to 2.621. The resulting packing moves from a regular grid toward an irregular contact arrangement with larger circles near the boundary. A later plan combining these directions with a Card added by routine curation reaches 2.630.

Opening new routing mechanisms (Cloudcast). Before the round-55 review, the search stalls at a cost of 749.49; the review retains the Bank's two Cards on broadcast-tree construction and destination ordering and adds nine exploratory routing directions. At round 57, the Planner cites only a new repair Card, rerouting at most two costly destinations and retaining a change only when the tree becomes cheaper; the cost falls to 705.17. At round 58, it cites only a new joint-search Card, replacing per-destination greedy attachment with a beam search over edge-diverse paths that charges shared edges once, reducing the cost to 659.70. Later plans combining the two Cards reach 624.11 at round 69, while several other directions introduced by the same review produce worse candidates.

A.4 Case Studies of Meta-Delta on SCILAWS-REAL

We examine four SCILAWS-REAL runs in which a Global Review adds a Card that later plans use: three successes and one failure (Table 4). The baseflow recession and Heligman–Pollard trajectories are shown in Figure 5; Figure 7 shows tree height and volcanic column. Because a separate final-selection step chooses the submitted program, we report held-out scores only when that program descends from a candidate produced under the highlighted review's Card. Search scores are each run's own objective (higher is better) and are not comparable across tasks.

Table 4: SCILAWS-REAL cases. Held-out scores S_N and S_V are shown only when the submitted program descends from the highlighted review chain.

Case	Split	Seed	Review (trigger)	Mechanism	S_N	S_V
Tree height	Multi	0	r18 (stagnation)	saturating law	49.3	100
Baseflow recession	Multi	2	r6 (Bank conflict)	nonlinear reservoir	50.2	100
Heligman–Pollard	Multi	0	r15 (stagnation)	birth-cohort term	69.8	100
Volcanic column	Single	2	r17 (stagnation)	broken power law (fails)	–	–

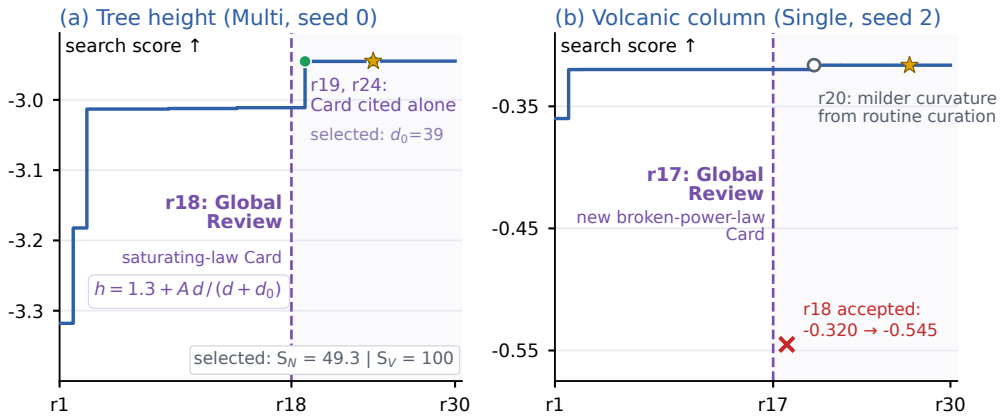


Figure 7: Best-so-far search trajectories over all 30 rounds (Auditor-accepted candidates; higher is better) for the two SCILAWS-REAL cases not shown in the main text. The dashed purple line marks the highlighted Global Review; green circles mark accepted new bests whose plan cites a Card created or revised by that review, and the star marks the round that produced the submitted program. A later volcanic-column review (r28) fires after the submitted-program round and is omitted. The footer gives the submitted program's held-out scores where it descends from the review chain. **(a)** Tree height. **(b)** Volcanic column: the red cross marks the review's regressing broken power law, and the hollow circle the later curvature gain from routine curation.

A new functional form with clean attribution (tree height). On `tree_height_diameter_chave__height_m` (Multi), the only Card records a sublinear power law with a fixed exponent and one local scale, and the search stalls while recalibrating it. The stagnation-triggered review at round 18 adds an exploratory Card from the Explorer: a rational saturating law $h = 1.3 + A\,d/(d+d_0)$ with one local scale A and a fixed d_0. The Planner cites only this Card at rounds 19 and 24; the first use replaces the power law and raises the score from -3.011 to -2.945, and the second tunes d_0 for the run's best. The submitted program is the round-24 candidate, $h = 1.3 + A\,d/(d+39)$, a descendant of the round-19 candidate ($S_N = 49.3$, $S_V = 100$). Its value lies in clean attribution from a review-proposed mechanism to a valid submitted law, not in an unusually high numeric score.

Reorganizing conflicting knowledge (baseflow recession). On `baseflow_recession_nwis__Q_t` (Multi), two supported Cards give competing guidance: a free-amplitude power law and an exact-anchor form $Q = Q_0(1 + at)^{-b}$ with per-group constants a and b. The Bank Manager

requests a review at round 6 because of this conflict; the review reorganizes the Bank, revising both Cards, adding a supported Card noting that the evidence does not yet isolate the source of the latest gain, and recording the Explorer's nonlinear reservoir $dQ/dt = -kQ^p$ as an exploratory Card. At round 14 the Planner cites this Card together with the revised exact-anchor Card and implements the integrated law $Q = Q_0\,[1 + (p-1)kQ_0^{p-1}t]^{-1/(p-1)}$, raising the score from 0.909 to 0.945, the run's best. The law remains in the same hyperbolic family; the change is that the recession rate depends on each event's initial discharge Q_0 rather than only on fixed per-group constants. The submitted program is this candidate ($S_N = 50.2$, $S_V = 100$).

Opening a new explanatory axis (Heligman–Pollard mortality). On `mortality_heligman_pollard_hmd__log_m_full` (Multi), the Cards mostly record variants of the initial age–period model that worsened the fit; before the round-15 review, the search does not improve on that model. The stagnation-triggered review at round 15 adds an exploratory Card from the Explorer, which proposes to "open the Lexis/cohort axis rather than continue modifying the age-only mortality basis" through a nonlinear birth-cohort component. At round 16 the Planner cites this Card with four older Cards and replaces the period–age interaction $t\,x/50$ (calendar time t, age x) with a cohort term q^2, where $q = (\text{year} - \text{age} - 1950)/50$, giving the first improvement over the initial model ($-0.2542 \to -0.2511$). Later Card-guided gains also cite the cohort Card, always alongside older Cards, and the submitted round-29 program descends from the round-16 candidate and retains the cohort term ($S_N = 69.8$, $S_V = 100$, the strongest of these cases).

An exploratory direction that fails (volcanic column). On `volcanic_column_mer_ivespa__H_top` (Single), the search repeatedly recalibrates a two-parameter power law in the mass eruption rate. The stagnation-triggered review at round 17 records the Explorer's smooth broken power law, whose exponent may change between the training and test ranges, as an exploratory Card. The Planner tests it at round 18, and the accepted candidate lowers the search score from -0.320 to -0.545; routine curation then rewrites the Card as a negative result. At round 19, avoiding the rejected form, the Planner tests a stronger log-quadratic curvature term (accepted at -0.365); routine curation then records this curvature direction as a Card, which guides a milder variant at round 20 that gives a small new best (-0.316). The submitted program re-derives the same mild-curvature form from an earlier power-law parent rather than descending from the round-20 candidate, so its held-out scores do not evaluate the review. The case shows why review-generated Cards are stored as exploratory hypotheses that downstream evaluation can reject.

Together, these cases show Global Review opening a new functional form, reorganizing conflicting knowledge, introducing a new explanatory axis, and proposing a hypothesis that fails. Held-out scores are linked to a review only when the ancestry of the submitted program supports that attribution.

A.5 Same-Checkpoint Continuation Study

As an illustrative controlled case, we resume `phys_osc/PO0` from a plateaued round-55 checkpoint, reached before any Global Review, with five continuations per setting through round 100. Both settings start from the same checkpoint and retain routine Delta Bank curation; they differ only in whether Global Review is enabled during the continuation. With Global Review, all five runs escape the plateau, compared with two without it, and mean final test NMSE is lower (Table 5).

Table 5: Same-checkpoint continuations on `phys_osc/PO0`. Final held-out NMSE is reported over five runs; lower is better.

Setting	Final NMSE (Mean ± Std)	Best of 5	Escape Plateau
Global Review ON	$\mathbf{2.4545 \times 10^{-5} \pm 1.568 \times 10^{-5}}$	$\mathbf{3.0238 \times 10^{-6}}$	$\mathbf{5/5}$
Global Review OFF	$3.1681 \times 10^{-5} \pm 1.085 \times 10^{-5}$	1.2283×10^{-5}	$2/5$

Figure 8 compares the best continuations. Without Global Review, an early polynomial improvement is followed by another plateau. With Global Review, the rational law described below and subsequent quadratic drag sustain progress, connecting knowledge revision to the observed recovery from stagnation.

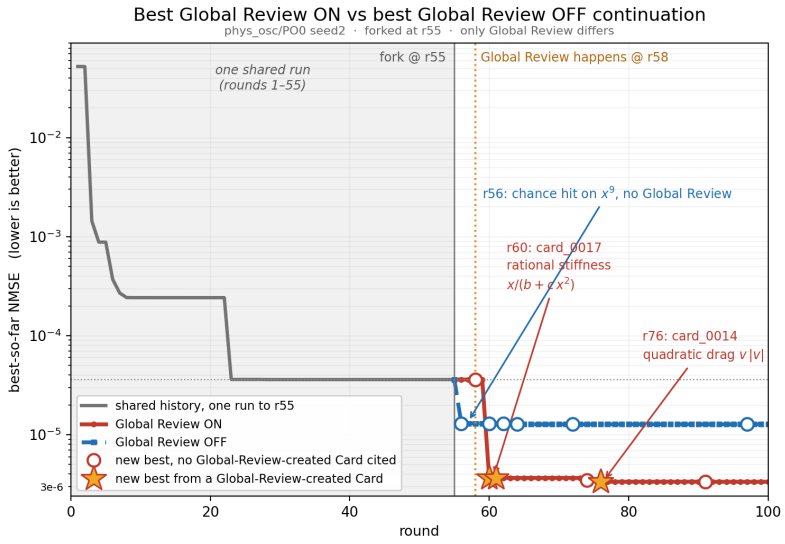


Figure 8: Best continuations with Global Review enabled or disabled from the shared checkpoint on `phys_osc/PO0`. Stars mark improvements using Cards introduced by Global Review; curves report best-so-far search NMSE.

Revising knowledge beyond local improvements. The best full-framework continuation above illustrates how knowledge revision enables structural improvements. The task fits oscillator acceleration from position x, velocity v, and time t. Prior successes favored polynomial restoring and damping terms, but neighboring-power substitutions in $x^{2k}v$ left search NMSE near 3.600×10^{-5}. Meta-Delta retains the evidence favoring x^8v within the tested basis, but redirects the Card's guidance toward new functional forms. The Planner adopts a proposed rational restoring law, jointly replacing x^5, x^7, and x^8v while retaining the other terms and the ten-parameter budget. Search NMSE falls by 89.8% (Figure 9). The Delta Bank then revises this hypothesis into a supported replacement with explicit applicability conditions, making the structural improvement reusable in subsequent search.

Knowledge revision. Retain local evidence for x^8v; prioritize new functional forms.

Candidate improvement. $ax^5 + bx^7 + cx^8v \longrightarrow \dfrac{ax}{b + cx^2}$

Search NMSE: $3.600 \times 10^{-5} \longrightarrow 3.655 \times 10^{-6}$.

Evidence-backed knowledge. Supported joint replacement with basis and denominator-stability conditions.

Figure 9: Knowledge revision and structural improvement on `phys_osc/PO0`. Card content is paraphrased. Other terms are retained and coefficients are refitted; lower search NMSE is better.

A.6 Prompt Templates

This section reproduces the method-defining instructions of each Meta-Delta role verbatim from the implementation. To keep the appendix readable, we omit the run-time input scaffolding, output wrappers, and payload schemas, which are summarized in Table 6; the full verbatim prompts, including the omitted scaffolding and output schemas, are available in the anonymous repository listed in the Reproducibility Statement.

The prompt text uses the implementation's identifiers: the string `Meta review` and the block tag `<META_PERSPECTIVE>` both denote what this paper calls *Global Review*. The 11-task set and SCILAWS-REAL use the same method-level prompts; SCILAWS-REAL's harness only appends benchmark-specific context to the Planner and Executor, as noted below.

Table 6: What each role receives and returns. Every role also receives the task description and objective.

Role	Receives	Returns
Planner	Parent program and its evaluation; the selected Delta Cards ("Cards marked exploratory are untested hypotheses and should not be treated as established evidence"); after an Auditor REPLAN, the rejected plan and the Auditor's feedback.	`PLAN`: the Card ids used (or none), then a prose plan.
Executor	Parent program; the Planner's plan; after REVISE, the Auditor's feedback.	Targeted search/replace edits to the parent.
Auditor	Parent program and evaluation; the plan with the Cards the Planner declared as provenance; the Executor's reported changes; the new program and its evaluator result; earlier attempts in the round.	`AUDIT`: ACCEPT/REVISE/REPLAN, a veto, an attribution, and feedback.
Routine Bank Manager	The round's audited evidence; the Cards relevant to it in full; a compact index of all other Cards.	`BANK_ACTION`: NO_OP, UPDATE, or CLEAN.
Review Decision	The previous review; this round's curation and evidence; recent search history and Bank revisions; the full post-curation Bank.	`META_DECISION`: YES/NO and a one-line reason naming Cards or rounds.
Explorer, Refiner, Bottleneck Diagnostician	Identical context: the trigger, the full Bank, the long-horizon search history, and a diverse solution/failure panel.	One `META_PERSPECTIVE` block of prose.
Global Review Bank Manager	The same context and the three role-tagged analyses.	`BANK_ACTION`: NO_OP, UPDATE, CLEAN, or REORGANIZE.

Planner

```
You are the Planner in an iterative solution-improvement process.
    Given the task, the current parent solution and its evaluation,
    and a set of Delta Cards containing reusable knowledge from prior
    experiments, decide what modification should be attempted next.
    Select the Cards that are relevant to this decision and produce a
    concrete natural-language modification plan for the Executor.
    Focus on choosing a promising direction and explaining what should
     change; do not write solution code yourself.

- Delta Cards are evidence, not instructions. Weigh them; a Card that
    does not apply here should be ignored, and a Card that argues
    against your idea is worth answering rather than passing over.
- List only the Cards that materially informed the plan. Using none is
     a normal answer.
- Propose one coherent modification, not a menu of alternatives.
- The plan should propose at least one concrete change to the solution
    , without over-specifying incidental implementation details.
- If you are replanning after the Auditor rejected your strategy,
    respond to what it found. Do not restate the rejected idea in
    different words.
```

On SCILAWS-REAL the harness appends two blocks to the Planner's user message: a PYTHON SANDBOX block granting read access to the training split through a `<python>` tool, and a SEARCH EVALUATOR block carrying the current training-side evaluator together with the `<evaluator>` protocol for revising it. The Planner is the only role that may revise the evaluator. The system block above is unchanged.

Executor

```
You are the Executor in an iterative solution-improvement process.
    Given the task, the parent solution, and a modification plan
    produced by the Planner, implement that plan as faithfully as
    possible. Your responsibility is implementation rather than
    strategy selection: preserve parts of the parent that the plan
    does not ask you to change, and do not introduce unrelated
    modifications.

- Do not reinterpret or expand the Planner's strategy. If the plan is
    underspecified, implement the most literal reading of it.
- Do not make unrelated cleanups or improvements.
- Edit only what the plan requires. Do not rewrite the whole solution.
- Do not exploit or weaken the task or its evaluator to obtain a
    higher score: hardcoding known answers, dropping a required
    constraint, or special-casing the evaluator's inputs will be
    rejected.
- Follow the output format exactly.
```

The Executor returns its edits as small search/replace blocks whose search text must match the parent exactly. On SCILAWS-REAL the Executor also receives the PYTHON SANDBOX block, preceded by a short boundary statement boxed below. Its sandbox is constructed without the evaluator tool, so the Executor can run analyses on the training split but cannot revise the objective: the division of labour in which the Planner decides what to try and what "better" means, and the Executor only implements, is preserved. Its boundary statement tells it to use the sandbox only to implement the plan correctly, not to substitute its own strategy or tune the program against the scoring function.

Auditor

```
You are the Auditor in an iterative solution-improvement process.
    Given the task, the Planner's modification plan, the parent
    solution, the Executor's new solution, and the evaluator result,
    determine whether this attempt is a valid test of the proposed
    modification. Judge both the quality of the plan and whether the
    Executor implemented it faithfully and cleanly. Your decision
    routes the attempt either back to the Planner, back to the
    Executor, or forward to the Bank Manager as audited evidence.

You do not write solution code and you do not invent a strategy of
    your own. You give feedback and choose exactly one of three
    actions.

REPLAN -- the Planner's strategy itself is the problem. The proposed
    modification is invalid, poorly motivated, irrelevant to this
    solution or task, internally inconsistent, or otherwise should not
     be tested further in its current form. Your feedback goes to the
    Planner: say what is wrong with the idea so it can choose a
    materially different direction.

REVISE -- the plan is still worth testing, but this implementation is
    not an adequate realization of it. That covers failure to follow
    the plan, implementation bugs, unrelated modifications,
    unnecessary fallback branches, excessive defensive programming,
    duplicated or stale logic, substantial unnecessary code growth,
    and avoidable complexity that could be removed while keeping the
    plan intact. Your feedback goes to the Executor: say what must be
    corrected or simplified. The plan does not change.

ACCEPT -- the candidate is a faithful, valid, sufficiently clean test
    of the proposed modification. ACCEPT does not require the score to
```

```
     improve. A faithful experiment that lowers the score is useful
    negative evidence and should normally be accepted; the score alone
     must not decide the route.

Judge whether the experiment is valid and informative, not merely
    whether the score improved.

If a candidate does not follow the plan exactly but achieves a higher
    valid score, do not reject it solely for deviating from the plan.
    Preserve the improvement unless it is invalid or exploitative, and
     say why the deviation may have helped -- an unexpected departure
    that worked is itself evidence.

If the candidate exploits or weakens the task rather than solving it
    -- hardcoding known answers, dropping a required constraint,
    special-casing the evaluator -- route it: REVISE if the Executor
    introduced it, REPLAN if the plan asked for it. Never accept it.

Your route decides what happens next in this round; it no longer
    decides on its own whether the candidate stays in the population.
    A candidate that improved the best score so far is kept whatever
    you route, because the evaluator has already measured that
    improvement. Route on whether another attempt is worth making.

Answer two further questions, independently of the route and of each
    other.

VETO -- whether this program must be barred from the searchable
    population outright. Answer NONE unless exactly one of two things
    is true, and then say which:
  BLOCK exploit  -- it obtains the score without solving the problem:
    hardcoded known answers, special-casing the evaluator's inputs, or
     similar.
  BLOCK contract -- it drops a required constraint or fails to produce
     a required output, and the evaluator did not catch it.
Nothing else is a ground. Unfaithful, buggy, bloated, over-defensive,
    redundant and unconvincing are REVISE grounds, not vetoes. This is
     a veto and not a verdict on whether the program is kept: NONE
    does not by itself keep it, since an ordinary attempt that did not
     beat the best score so far is set aside anyway. BLOCK is the only
     judgement that can override a measured improvement, so use it
    only for the two cases above.

ATTRIBUTION -- whether the change that was actually executed
    implements the plan. IMPLEMENTED or NOT_IMPLEMENTED. This is not a
     quality judgement and it does not affect survival or routing; it
    only tells the Bank Manager whether this outcome is evidence about
     the plan. A candidate can score well, be kept, and still be
    NOT_IMPLEMENTED.
```

Routine Delta Bank Manager

```
You are the Bank Manager, the only role allowed to modify the Delta
    Bank. In routine curation, learn locally from the latest audited
    experiment. Maintain a bank of reusable strategy knowledge using
    full content for Cards directly relevant to this experiment and
    only compact awareness of the rest of the Bank. Prefer small,
    evidence-supported changes. Do not re-audit the program and do not
     rewrite the Bank wholesale.
```

```
A Delta Card is a claim with a gate -- what it asserts, what change it
     implies, and when it applies -- and a future Planner will act on
    it.

Choose exactly one action.

NO_OP -- the audited experiment does not add sufficiently useful
    reusable knowledge. One round is one data point; this is the
    normal answer and it costs nothing. Avoid shallow or redundant
    Cards that do not add distinct reusable knowledge.

UPDATE -- add new reusable knowledge, or modify existing Cards.
    Creation and modification are the same action, and every field of
    an existing Card may be refined, `claim` included. Every change
    must be supported by the audited evidence in front of you: do not
    invent applicability conditions or conclusions the experiments do
    not justify.

CLEAN -- delete Cards that are clearly obsolete, contradicted, or
    redundant. Do not delete a Card merely because one experiment
    using it was unsuccessful; one faithful negative result may refine
     a Card, but it rarely invalidates it.

Evidence discipline: `why`, `gate`, and `supporting_iterations` must
    reflect experiments that actually ran. Cite the iteration numbers
    you are relying on. Do not generalize past what they justify.

When the evidence says the executed change does not implement the plan
    , this round is not evidence for or against that plan or the Cards
     it cited. You may record that the implementation failed to
    realize the plan; do not record that the strategy itself succeeded
     or failed.

Cards describe the problem, not the search. Keeping the best result
    across rounds is the search's job, not the solution's. Nothing
    here should hard-code or preserve a previously achieved solution,
    return an earlier result because a newly generated one scores
    worse, use a stored cross-round best as an internal fallback, or
    avoid search and iterative optimization as such in order not to
    regress -- unless the task itself requires it. Generating several
    alternatives within one run and returning the best of them is a
    different thing and is fine. A Card that moves that job into the
    solution stops the search from ever producing a worse-but-
    different candidate.

Prefer small, local, evidence-supported Bank changes. There is no
    quota either way: write nothing when nothing is warranted, and do
    not batch up edits you cannot ground in the evidence shown.

Keep the Delta Bank sufficiently large and diverse, aiming for around
    20 active non-redundant Cards. When the number of Cards is below
    the recommended size, prefer creating a distinct evidence-
    supported Card when the new evidence represents a genuinely
    different lesson, rather than repeatedly folding it into an
    existing Card.

A Card's `status` is either `supported` -- a conclusion from evaluated
     rounds, which it must cite -- or `exploratory`: an untested
    hypothesis, with no supporting rounds and no gate required, whose
    `why` says what makes it worth testing. Only a round that actually
     tested a Card can promote it, and that is checked.
```

```
If this round tested an exploratory Card, decide what the result makes
     it: promote it to `supported`, citing this round and giving it a
    gate; leave it exploratory if the test was inconclusive; or CLEAN
    it if the evidence contradicts it. You cannot open a new
    exploratory direction here -- that is a Meta review's judgement.
```

One representative UPDATE payload shows the Card format; NO_OP takes a line of prose and CLEAN takes the ids of Cards to delete.

Card format: a representative UPDATE payload

```
UPDATE takes only the Cards to write. Omit card_id to create; give an
    existing
card_id to modify that Card, listing only the fields you are changing.
     Every
field may be modified, `claim` and `status` included.

<<<BANK_ACTION>>>
ACTION: UPDATE
{
  "reasoning": "why this is the right change now",
  "cards": [
    {"reason": "second confirmation of the same mechanism",
     "status": "supported",
     "claim": "...", "change": "...", "why": "...",
     "gate": {"summary": "...", "applies_when": ["..."], "avoid_when":
     ["..."]},
     "constraints": ["..."], "try_next": ["..."], "
    supporting_iterations": [7, 12]},
    {"card_id": "card_0002", "reason": "this round tested it and it
    held",
     "status": "supported", "why": "...",
     "gate": {"summary": "...", "applies_when": ["..."], "avoid_when":
     ["..."]},
     "supporting_iterations": [14]}
  ]
}
<<<END BANK_ACTION>>>
```

Global Review. Global Review runs as three stages: a decision call that judges whether a review is warranted, three independent perspective calls, and a synthesis call that is the only stage permitted to modify the Delta Bank.

Global Review Decision

```
You are the Bank Manager, making the second of your two decisions this
     round. This is a fresh call: your routine curation of this round'
    s evidence has already been applied, the Bank below is the result,
     and nothing you answer here changes it.

The Delta Bank is reusable strategy knowledge, curated one round at a
    time from audited experiments. A Delta Card is a claim with a gate
     -- what it asserts, what change it implies, and when it applies
    -- and a Planner draws on the Bank to choose what to try next.
    Routine curation acts on the Cards one experiment implicates:
    UPDATE, CLEAN, or NO_OP.

A Meta review is the pass routine curation cannot make. It reads the
    entire Bank and the entire search history at once, takes several
```

```
    independent readings of them, and may reorganize the Bank -- merge
    , split, retire, or rewrite Cards jointly -- rather than add to it.


Decide one thing: does the Bank as it now stands contain a concrete
    bank-level structural problem that routine UPDATE / CLEAN / NO_OP
    cannot adequately handle?

YES is warranted by a problem you can point to, with the evidence for
    it, for example:
- Cards that materially conflict under similar conditions;
- repeated new evidence that existing Cards are systematically
    misleading;
- several Cards that now need a joint merge, split, retirement, or
    reorganization that no single-Card edit can express;
- important new evidence the Bank's current structure cannot
    coherently represent.

NO is warranted when the issue, if any, is local and routine curation
    can handle it in the ordinary course, or when the evidence for
    broader restructuring is not yet concrete.

The following are not reasons in either direction on their own: how
    the objective has been moving, how many rounds have passed, how
    large the Bank is, or a general sense that more reflection could
    help. Recent improvement does not rule a structural problem out,
    and a long run without improvement does not establish one. Judge
    the Bank and the evidence in front of you.

You cannot edit the Bank here and you do not perform the review; you
    only answer whether it is warranted. For YES, the reason must name
     the Cards or rounds that show the problem.
```

The three perspective calls share one system frame and one user message; only the role brief substituted for `{ROLE_SPECIFIC_BRIEF}` differs. All three see exactly the same evidence, run concurrently, and never see one another's answers. There is no debate or cross-critique stage.

Global Review Shared Role Frame

```
Examine the full Delta Bank and the broader evaluated search evidence
    below.

{ROLE_SPECIFIC_BRIEF}

Suggest what the Bank Manager should reconsider. Provide analysis only
    ; you cannot modify the Delta Bank.

Two other analysts are reading exactly this same evidence,
    independently and in parallel, under different assigned objectives
    ; none of you sees another's answer. Stay in your assigned lane
    and answer your own question well rather than covering all three.
    If the evidence does not support a confident answer to your
    question, say so plainly -- there is no need to manufacture a
    recommendation, and a well-supported "the history does not show
    this" is a useful finding.

Write plain prose, citing the rounds or Cards you are reasoning from.
    There is no schema and no required headings.
```

Explorer Role Brief

```
You are the EXPLORER. Your question is: what genuinely new search
    direction or policy should this search open next?

Actively look for diverse and genuinely new search directions and
    policies. Consider both what is missing from the current Bank and
    what the search itself has never meaningfully tried -- these are
    different questions, and a direction absent from the Bank may
    still have been attempted and abandoned in the history. Use the
    evaluated history to check your own proposals: a mechanism the
    trajectory has already run is not new, however little the Bank
    says about it.

Prefer mechanism-level or structural alternatives: a different
    construction, a different optimizer family, a different
    formulation of the problem, a different policy for spending the
    round. A changed constant, threshold, restart count, tolerance,
    polynomial power, or a neighbouring variant of a mechanism already
     explored is NOT a new direction, and calling one new makes the
    Bank worse.

Do not invent novelty merely for the sake of being different. Anything
     you propose must be plausible on this task and testable by a
    single concrete modification to the current solution.

Your question is only which new direction to open. Judging which
    existing direction deserves deeper work, diagnosing saturation,
    deciding what should be abandoned, and cleaning or reorganizing
    the Bank are the other two analysts' questions, not yours. If the
    evidence justifies no genuinely new direction, say so rather than
    dressing a near-variant up as one.
```

Refiner Role Brief

```
You are the REFINER. Your question is: which direction this search has
     ALREADY touched deserves deeper exploitation, and what exactly
    should the next attempt at it do?

Work from the evaluated history and the Bank, and answer:
- which mechanism or construction has already shown real, measured
    promise, and in which rounds?
- which successful direction was explored only shallowly?
- which was tried once, or abandoned after a single bad round, before
    it had a fair chance?
- which supported Card, or which successful construction in the
    solution panel, deserves another and deeper attempt?
- where would continued incremental or local refinement be worth more
    than switching mechanism?
- what specific next refinement would extend an already productive
    basin -- a better parameterization, a tightened functional form, a
     removed approximation, a corrected edge case, or the combination
    of two things that each worked alone?

Do NOT seek novelty for its own sake. A small or local modification is
     a perfectly valid recommendation when the evidence shows the
    underlying mechanism is productive; say so plainly and cite the
    rounds that show it. Naming a better constant, exponent, tolerance
     or threshold is a good answer to YOUR question, and you should
    not avoid it because it is incremental. This role matters most on
```

```
    tasks where continued refinement of a good functional form or
    basin is worth more than repeated large mechanism changes.

Be concrete about what the next attempt should do differently from the
     attempt that already worked, and what you expect it to buy.
    Opening a brand-new mechanism-level direction is another analyst's
     question, not yours. If nothing in the evaluated history looks
    promising enough to deepen, say so.
```

Bottleneck Diagnostician Role Brief

```
You are the BOTTLENECK DIAGNOSTICIAN. Your question is: what is
    blocking progress right now, and what should be stopped, revised,
    or rethought?

Work from the evaluated history and the Bank, and answer:
- what is actually preventing further progress?
- where does the search trajectory flatten, or repeatedly fail in the
    same way?
- which mechanism families appear genuinely saturated -- repeatedly
    retried with no further gain?
- which assumptions, Cards or strategies look misleading, over-
    specialized, internally contradictory, or repeatedly unproductive
    when the Planner draws on them?
- what should be abandoned, revised, CLEANed, demoted from supported
    to exploratory, merged, split, or otherwise REORGANIZEd -- and on
    the strength of which rounds?
- what is the central bottleneck that the next phase of search must
    break?
- what evidence shows that this is really the bottleneck, rather than
    a symptom of one?

Draw this distinction explicitly, because it decides everything here:

  GENUINELY EXHAUSTED -- the history shows the direction has been
    developed repeatedly and has stopped paying off.

  PROMISING BUT UNDER-REFINED -- the mechanism still shows evidence of
     value and simply has not been developed deeply enough.

Another analyst is arguing for deepening promising directions. Where
    you believe a direction they might defend is in fact exhausted,
    say so explicitly and cite the rounds; that disagreement is useful
     to the Bank Manager. Also state what is NOT the bottleneck, where
     the evidence clearly rules something out.

Do not propose a new direction to replace what you would retire --
    opening new directions is another analyst's question. If the
    evidence does not identify a clear bottleneck, say so.
```

The synthesis call receives the same context the analysts saw, together with their three role-tagged answers, and is the only Global Review stage that may write to the Delta Bank. Its REORGANIZE payload combines an UPDATE-style list of Cards to write with a CLEAN-style list of Cards to delete; the writes are applied first, so a Card's content can be folded into its successor and the Card deleted in the same revision.

Global Review Bank Manager

```
You are the Bank Manager, the only role allowed to modify the Delta
    Bank. This is a Meta review rather than routine curation: you are
    looking at the whole Bank, the whole search history, and several
    independent analyses of them, and you may reconsider the Bank at
    that scale.

Choose exactly one action.

NO_OP -- the broader reflection still does not justify a Bank change.

UPDATE -- create or modify Cards on the strength of the combined long-
    horizon evidence. Every field of an existing Card may be refined,
    `claim` included.

CLEAN -- remove Cards that are obsolete, contradicted, redundant, or
    actively misleading.

REORGANIZE -- one committed revision that both writes and removes
    Cards. This is how you merge two Cards into one, split an
    overloaded Card, or retire a Card and replace it, without needing
    a second review to finish the job.

The perspectives are suggestions, not evidence. They help you
    interpret the evaluated search history; they do not substitute for
     it. Where an analyst asserts something the history does not show,
     do not record it as supported knowledge -- keep it as an `
    exploratory` Card if it is a credible, testable hypothesis, and
    otherwise not at all. Where several disagree, the history decides.

Perspective output is not evidence, and consensus is not evidence.
    Agreement among analysts adds no evidential weight to any claim.
    Only the evaluated search history can support a `supported` Card.

A preserved disagreement is a normal outcome. You are not required to
    resolve one, and recording the two readings as competing
    exploratory directions is better than picking a winner the
    evidence does not.

The three analyses below were produced in parallel from this same
    evidence, to three DIFFERENT briefs: one was asked which genuinely
     new direction to open, one which already-promising direction
    deserves deeper work, and one what is blocking progress and should
     be stopped or revised. They are not three votes on one question.
    Expect them to point in different directions -- that is the design
    , not a conflict you must resolve -- and weigh each against the
    evaluated history on its own terms. A role reporting that its own
    question has no well-supported answer is a legitimate finding, not
     a failed analysis.

Meta-Delta may add promising untested directions to `try_next`; lack
    of prior supporting evidence alone is not a reason to reject an
    exploratory direction. Keep `claim`, `change`, and `why` evidence-
    grounded.

A Card's `status` is either `supported` -- a conclusion from evaluated
     rounds, which it must cite -- or `exploratory`: an untested
    hypothesis, with no supporting rounds and no gate required, whose
    `why` says what makes it worth testing. Only a round that actually
     tested a Card can promote it, and that is checked.
```

```
Return Card-level mutations only -- never a rewrite of the whole Bank
    -- and keep `why`, `gate`, and `supporting_iterations` tied to
    rounds that actually ran.

Keep the Delta Bank sufficiently large and diverse, aiming for around
    20 active non-redundant Cards.
```