Recursive Epistemic Engines for Verifiable Open-Ended Scientific Agents

David Scott Lewis [1] **Enrique Zueco** [1] **Haley Yi** [1]

Abstract

Current language-model research agents can generate ideas, write code, run experiments, and draft reports, but their unit of progress is still too often a fluent research narrative rather than a verified scientific object. We formalize this gap as the *Novelty Horizon*: the boundary beyond which an autonomous system cannot jointly generate, execute, verify, and reuse non-derivative hypotheses. We propose *Recursive Epistemic Engines* (REEs), a template-free architecture that couples large-language-model proposal priors to typed artifact languages, quality-diversity archives, theorem provers, runtime/statistical checkers, and guarded self-modification. The central object is a proof-carrying, experiment-carrying claim package whose value is measured by verified compression gain, expected information gain, transfer, and lineage-auditable reuse. We instantiate the framework with three mechanisms: a verification-gated quality-diversity prover loop; a categorical abstraction layer for compositional scientific objects; and a guarded self-evolution loop for improving verifiers, descriptors, and campaign policies. Three reproducible synthetic experiments isolate the architecture's core claims. Verification-gated QD expands diverse admissible archives while avoiding off-spec artifacts; proof checking without semantic-alignment gates admits many vacuous or misformalized statements; and guarded self-modification preserves heldout utility and verifier strength under Goodhart pressure better than visible-score adoption. The result is a concrete research program for scientific agents that evolve not by producing more plausible papers, but by recursively expanding the space of machine-checkable hypotheses, experiments, and abstractions.

[1]AIXC Research, Zaragoza, Spain. Correspondence to: David Scott Lewis <reports@aixcresearch.com>.

The 3rd *AI for Math Workshop* at the 43rd International Conference on Machine Learning (ICML), Seoul, South Korea, 2026.
Copyright 2026 by the author(s).

1. Introduction

Automated research agents are crossing an important engineering threshold. Recent systems orchestrate literature search, idea generation, code editing, experiment execution, result analysis, and manuscript drafting (Lu et al., 2024; Yamada et al., 2025; Schmidgall et al., 2025). In parallel, LLM-guided program search has produced nontrivial mathematical and algorithmic discoveries (Romera-Paredes et al., 2024; Novikov et al., 2025; Lehman et al., 2023), while formal theorem-proving systems show that neural proposal mechanisms can be grounded in proof assistants (Yang et al., 2023; Song et al., 2024; Hubert et al., 2025). These developments suggest a plausible path toward autonomous mathematical and scientific discovery.

The missing layer is epistemic rather than merely agentic. A research loop that ends in a plausible report may still contain unverified code, a misformalized theorem, a data leak, a vacuous specification, or an evaluation artifact that rewards legibility rather than truth. Surveys and critiques of autonomous and LLM-mediated science repeatedly identify hallucination, brittle execution, template reliance, weak reproducibility, and evaluation mismatch as structural bottlenecks (Musslick et al., 2025; Zhang et al., 2025a; Wei et al., 2025; Cornelio et al., 2025; Zenil et al., 2026; Kulkarni et al., 2025). The deeper problem is that current agents usually search inside a pre-given grammar: known paper forms, known benchmark APIs, known coding idioms, known verifier stacks, and known notions of contribution.

We therefore change the unit of automation. A scientific agent should not optimize directly for report-shaped outputs. It should optimize for *verified expansion of reusable explanatory power*. The primitive object is a claim package

$$a = (\Sigma, h, s, c, \Omega, p, e, r, \ell), \tag{1}$$

where Σ is a typed signature, h an informal hypothesis, s a formal specification, c executable code, Ω proof and runtime obligations, p proof objects or counterexample certificates, e an experiment protocol, r results with uncertainty, and ℓ a provenance lineage. A package can be accepted as a theorem, verified program, empirical regularity, falsified hypothesis, or reusable abstraction. This object is small enough to be checked and large enough to represent scientific progress.

Contributions. We make four contributions. First, we define the Novelty Horizon as a joint generation–verification boundary, not a semantic-similarity threshold. Second, we introduce REEs, an architecture in which LLMs are stochastic proposal priors over typed artifacts, while acceptance is controlled by formal, statistical, causal, and replay gates. Third, we give a mathematical intrinsic-value functional based on verified MDL compression, expected information gain, transfer, and archive expansion. Fourth, we provide three reproducible experiments that isolate verification-gated QD search, semantic-alignment leakage, and guarded self-modification.

2. The Novelty Horizon

Let an agent state at cycle t be

$$\Pi_t = (\mathcal{L}_t, \mathcal{G}_t, \mathcal{A}_t, \mathcal{V}_t, \mathcal{E}_t, \mathcal{M}_t, \pi_t), \tag{2}$$

where $\mathcal{L}_t$ is the typed representation language and library, $\mathcal{G}_t$ the proposal distribution, $\mathcal{A}_t$ the accepted archive, $\mathcal{V}_t$ the verifier stack, $\mathcal{E}_t$ the experimental environment, $\mathcal{M}_t$ memory/world model, and π_t the campaign policy. The system's verified reach is

$$\begin{aligned}\mathcal{H}_t(\epsilon, \rho, B) = \{a : P_{\mathcal{G}_t}(a \mid \mathcal{M}_t, \mathcal{A}_t) > \epsilon, \\ P[\mathcal{V}_t(a) = 1] > \rho,\ \mathrm{Cost}(a) \le B\}.\end{aligned} \tag{3}$$

The *Novelty Horizon* is the frontier of $\mathcal{H}_t$: objects beyond it may be speakable but not verifiable, or verifiable but not generable. Paradigm-level discovery requires expansion of the language, verifier, environment, and policy, not only sampling more candidates:

$$\Delta\mathcal{H}_t \approx \Delta\mathcal{L}_t + \Delta\mathcal{V}_t + \Delta\mathcal{E}_t + \Delta\pi_t. \tag{4}$$

This view separates text novelty from epistemic novelty. A random theorem statement has high description length but no proof, compression, or transfer. A benchmark tweak may be valid but scientifically low-value. A useful abstraction compresses evidence, supports falsifying tests, transfers across contexts, and enters the reusable library.

Verified interestingness. For data D, current library $\mathcal{L}$, and artifact a, define a two-part description length $C_{\mathcal{L}}(D) = L(M_{\mathcal{L}}) + L(D \mid M_{\mathcal{L}})$. Let

$$\Delta_{\mathrm{MDL}}(a; D \mid \mathcal{L}) = C_{\mathcal{L}}(D) - C_{\mathcal{L}\cup\{a\}}(D) \tag{5}$$

be the compression gain after admitting a. Let $\widehat{K}(a \mid \mathcal{L})$ approximate conditional algorithmic complexity by compressed size, typed derivation length, or proof term length. Let $T(a)$ measure transfer to heldout worlds, $QD(a)$ archive-cell improvement, and $U_{\mathrm{sem}}(a)$ semantic uncertainty. We define

$$\begin{aligned}\mathcal{I}_t(a) = G_t(a)[\alpha\Delta_{\mathrm{MDL}}(a) + \beta\,\mathrm{EIG}(a) + \gamma T(a) \\ + \eta QD(a) + \zeta V_{\mathrm{new}}(a) \\ - \lambda\widehat{K}(a \mid \mathcal{L}) - \mu\,\mathrm{Cost}(a) - \nu U_{\mathrm{sem}}(a)].\end{aligned} \tag{6}$$

where $G_t(a) \in \{0, 1\}$ is a hard admissibility gate. For formal claims, G_t requires type/proof checking or an explicit conjectural status; for empirical claims, it requires a registered protocol, replayable code, and error-controlled statistical evidence. Equation 6 borrows the spirit of MDL and algorithmic information theory (Grünwald, 2007; Li & Vitányi, 1998; Schmidhuber, 2010), Bayesian experimental design and active inference (Friston et al., 2015; 2017; Parr et al., 2022; Beck et al., 2018), and quality-diversity search (Pugh et al., 2016; Lehman & Stanley, 2011; Mouret & Clune, 2015).

3. Design Requirements

The Novelty Horizon implies four requirements that are easy to miss when building agent pipelines. *First*, the system must distinguish proposal fluency from admissibility. LLMs are useful as abductive priors over candidate ideas, proof sketches, and code transformations, but they should not own the acceptance criterion. Every accepted artifact must have a machine-checkable status and a replay path from claim to evidence. This is why the archive stores proof terms, runtime traces, data hashes, and failed obligations rather than only natural-language summaries.

Second, the system must preserve stepping stones without preserving errors. Standard best-first optimization is a poor model of scientific progress because many important discoveries are intermediate instruments, lemmas, counterexamples, or representational shifts whose value appears only after later reuse. QD archives address this by maintaining diverse elites, but ungated QD can become a museum of invalid artifacts. The archive therefore admits only candidates that satisfy the relevant formal, empirical, and alignment gates, while rejected candidates remain available as repair data.

Third, the system must represent abstraction explicitly. A discovered equation, theorem, or algorithm is not only a string; it is a typed morphism between assumptions, measurements, computations, and conclusions. If the code, theorem, and experiment do not commute as a diagram, the artifact is epistemically incoherent even when each local component appears plausible. The categorical layer is not decorative mathematics; it is a discipline for making claim-to-evidence alignment checkable.

Fourth, self-evolution must improve the machinery of verification, not merely the machinery of persuasion. A self-

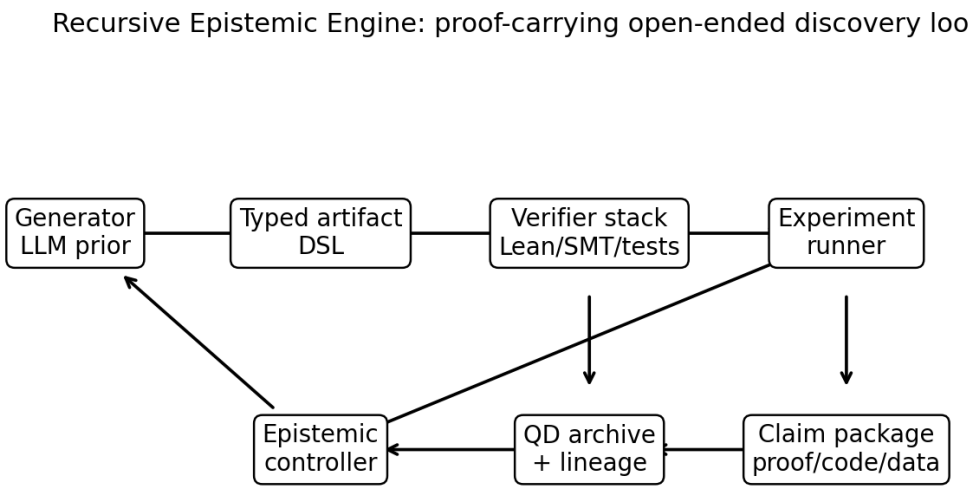

Figure 1. A Recursive Epistemic Engine treats proofs, programs, experiments, and provenance as first-class artifacts. LLMs propose typed candidates, but archive admission is gated by formal and empirical verification. Failed checks return structured repair signals rather than becoming self-confirming narratives.

modifying agent can raise its visible score by weakening validators, narrowing test distributions, or exploiting benchmark conventions. A recursive scientific agent must instead be rewarded for verifier improvement, hidden transfer, and falsification yield. This motivates guarded adoption rules, sealed holdouts, and evidence budgets.

A useful operational test is whether a system can explain why a candidate is *not yet* knowledge. A failed proof obligation, a counterexample, a violated unit constraint, or a non-replayable trace should produce structured information that alters future search. In contrast, a conversational self-critique that merely says a result is uncertain does not update the epistemic substrate. This is the reason for maintaining rejected artifacts with diagnostic labels: they form a curriculum for repairing the generator and strengthening validators.

4. Architecture

An REE has six coupled modules: (i) a proposal prior, typically an LLM or ensemble; (ii) a typed artifact DSL; (iii) a verifier stack; (iv) an experiment runner; (v) a QD archive with lineage; and (vi) a metacognitive controller. Figure 1 shows the loop.

QD-Prover loop. The first instantiation is a MAP-Elites-like archive over descriptors $b(a)$ such as claim type, proof depth, experiment class, causal role, abstraction reuse, and novelty mode. Variation operates over typed artifacts, not raw text: rewrite a lemma, compose two verified modules, generalize an invariant, or add a falsifying experiment. The update rule is

$$z = b(a), \qquad \mathcal{I}_t(\varnothing) = -\infty,$$
$$\mathcal{A}[z] \leftarrow \begin{cases} a, & \mathrm{Accept}(a) \wedge \mathcal{I}_t(a) > \mathcal{I}_t(\mathcal{A}[z]), \\ \mathcal{A}[z], & \text{otherwise.} \end{cases} \tag{7}$$

This imports stepping-stone preservation from QD and open-ended evolution (Wang et al., 2019; Stock & Gorochowski, 2024; Kumar et al., 2025; Faldor & Cully, 2024) but changes admission from behavioral novelty alone to proof- and experiment-carrying novelty.

Categorical abstraction layer. The second instantiation represents scientific claims as compositional diagrams. Let $\mathcal{C}_t$ be an epistemic category whose objects include datasets, hypotheses, proof states, programs, experiments, and claims. Morphisms are certified transformations: formalization, compilation, measurement, proof, execution, or counterexample construction. A claim is a diagram $D : J \to \mathcal{C}_t$; coherence requires designated diagrams to commute. Functors $F_{\mathrm{proof}} : \mathcal{C}_t \to \mathbf{Prop}$, $F_{\mathrm{exec}} : \mathcal{C}_t \to \mathbf{Prog}$, and $F_{\mathrm{data}} : \mathcal{C}_t \to \mathbf{Meas}$ map informal objects into proof obligations, executable programs, and measurable quantities. In deployment, the categorical layer can be realized as a typed

Table 1. Design requirements induced by the Novelty Horizon. The goal is to make each failure mode observable as a failed gate or audit trace rather than a plausible but unchecked narrative.

Requirement	Failure if absent	REE mechanism	Measured by
Executable provenance	claims detach from code, data, or proof objects	content-addressed claim package and replay traces	auditability, replay rate
Semantic formalization	true theorem encodes wrong intent	equivalence probes, anti-vacuity, counterexample search	false-pass rate
Open-ended diversity	search collapses to one local idiom	descriptor archive over claim, proof, experiment, and abstraction roles	verified coverage, archive hypervolume
Falsification discipline	predictive fit masquerades as causal evidence	explicit nulls, interventions, stopping rules, and e-values	falsification yield, calibration
Guarded self-evolution	agent improves visible score by corrupting validators	hidden validators, invariants, and adoption budgets	regression rate, Goodhart gap
Reusable abstraction	artifacts solve one instance but transfer nowhere	typed interfaces, categorical composition, heldout worlds	transfer gain, reuse count

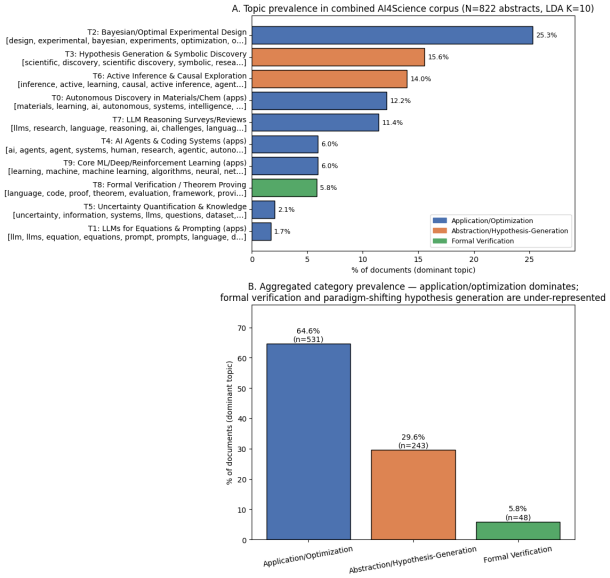

Figure 2. Literature synthesis motivating the gap: research concentration on application/optimization leaves comparatively little emphasis on formal verification and open-ended abstraction. The proposed architecture targets this under-instrumented intersection.

artifact graph: nodes are content-addressed claim fields, edges are checked transformations, and each commuting square becomes an executable contract or proof obligation. This layer draws on applied category theory and sheaf-based local-to-global consistency (Spivak, 2014; Fong & Spivak, 2019; Bodnar et al., 2022; Shiebler et al., 2021). It is designed to catch inconsistencies such as "the implementation tests a different construct than the hypothesis" before execution.

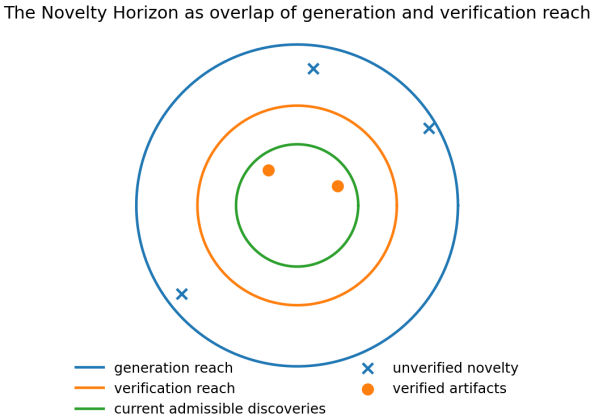

Figure 3. The Novelty Horizon as overlap between generation reach and verification reach. An REE expands the overlap, not only the generative frontier.

Guarded self-evolution. The third instantiation evolves not only artifacts, but also descriptors, verifiers, simulators, and campaign policies. A proposed self-modification $m : \Pi_t \mapsto \Pi'_t$ is adopted only if it passes invariants, hidden holdouts, regression tests, and statistical certificates:

$$\Pi_{t+1} = \begin{cases} \Pi'_t, & G_{\mathrm{inv}}(m)G_{\mathrm{stat}}(m) \\ & G_{\mathrm{holdout}}(m)G_{\mathrm{safety}}(m) = 1, \\ \Pi_t, & \text{otherwise.} \end{cases} \tag{8}$$

This preserves the open-ended archival spirit of Gödel machines, PowerPlay, Darwin Gödel Machines, and hyperagents (Schmidhuber, 2003; 2013; Zhang et al., 2025b; 2026; Yin et al., 2024; Wu et al., 2025), but makes scientific validity rather than benchmark score the quantity being recursively improved.

Algorithm 1 Recursive Epistemic Engine loop

Require: Library $\mathcal{L}_t$, archive $\mathcal{A}_t$, verifier stack $\mathcal{V}_t$, evidence D_t

for campaign cycle $t = 1, 2, \ldots$ **do**
  Sample parent artifacts and open archive niches from $\mathcal{A}_t$
  Propose typed candidate $a = (\Sigma, h, s, c, \Omega, p, e, r, \ell)$ using $\mathcal{G}_t$
  Elaborate a into proof, runtime, statistical, and replay obligations
  Run $\mathcal{V}_t(a)$; collect proofs, counterexamples, traces, and failed goals
  if $\mathrm{Accept}(a)$ and $\mathcal{I}_t(a) > \theta$ **then**
    Insert or replace $\mathcal{A}_t[\mathrm{desc}(a)]$ using verified interestingness
  else
    Store failure as structured repair signal, not as accepted knowledge
  end if
  Propose meta-actions over descriptors, verifiers, tools, and policy
  Adopt only modifications passing guarded self-evolution gates
end for

Ensure: Lineage-auditable archive of proof- and experiment-carrying artifacts

5. Evaluation Principles

We evaluate scientific agents by claim packages, not by narrative fluency. The primary scalar is a normalized verified novelty index

$$\mathrm{VNI}(a; D, \mathcal{L}) = \frac{\max(0, \mathcal{I}_t(a))}{\log(1 + |D|)}. \tag{9}$$

Complementary axes are: proof-realization ratio; proof-carrying-code rate; executable verification score; falsification discipline; transferable abstraction gain; open-ended archive growth; auditability; and Goodhart resistance. The benchmark design is inspired by full-cycle robot-scientist and autonomous-discovery environments (King et al., 2004; Sparkes et al., 2010; Vriza et al., 2025; Anonymous Authors, 2025; Garikaparthi et al., 2026) while adding formal proof, hidden validators, and self-modification governance.

Adjacent governance and agent-evaluation tools help specify the deployment interface. EviBound-style evidence gates require queryable run artifacts; Pramāṇa-style attestations provide a typed verification record; AAR-style DAG navigation tasks expose non-linear tool-use failures; and T2-style modular adaptation suggests how verifiers and tools can evolve without rewriting the frozen generator (Chen, 2025; Kadaboina, 2026; Kim et al., 2026; Jiang et al., 2025). In REEs, these are not competing systems but boundary conditions on the claim package: every accepted artifact should carry evidence, attestation, nonlinear replay stress, and guarded tool evolution.

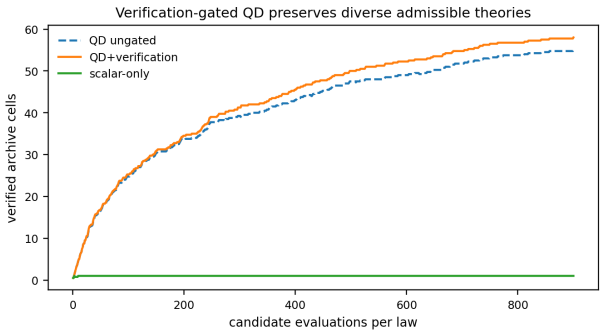

Figure 4. Experiment 1. Verification-gated QD preserves diverse admissible symbolic theories; scalar search collapses to one elite, and ungated QD stores invalid high-fit candidates.

6. Experiments

The experiments are controlled, synthetic, and reproducible. They isolate mechanisms that are difficult to attribute cleanly in full-stack agent studies: archive admission under verification, semantic leakage through formalization, and governance of self-modification.

6.1. Experiment 1: Verification-Gated QD

We created four hidden symbolic-law worlds: an even quadratic, an odd saturating rational law, an even damped oscillator, and an odd oscillatory law. Candidate artifacts are small programmatic equation skeletons drawn from polynomial, trigonometric, exponential, rational, logarithmic, radical, and mixed bases. Each candidate is fit by least squares, evaluated on mild extrapolation, assigned a descriptor by basis family and complexity, and checked against invariant obligations such as evenness or oddness. We compare scalar best-only search, ungated QD, and QD with verification-gated archive admission.

Figure 4 shows that the scalar policy finds a high-scoring artifact but covers only one niche. Ungated QD obtains broad coverage but admits off-spec artifacts that fit training data while violating hidden invariant tests. QD+verification reaches a mean of 58.0 verified cells with zero off-spec archive entries at the final budget. This supports the architectural claim that open-endedness must be paired with hard admissibility gates.

6.2. Experiment 2: Formalization Leakage

Formal theorem provers certify truth of the submitted formal statement, not faithfulness to an informal intent. We emulate autoformalization by sampling theorem statements

Table 2. Three experiments isolate core architectural claims. Results are means over hidden laws or random seeds; full protocols, ablations, and additional plots are in the appendix.

Experiment	Comparison	Primary metric	Best baseline	REE mechanism
Verification-gated QD	scalar-only vs. QD archives	verified archive cells	1.0	58.0
Verification-gated QD	ungated QD vs. gated QD	off-spec accepted artifacts	22.3%	0.0%
Semantic formalization	proof-only vs. semantic gate	false-pass among accepted	56.2%	0.0%
Guarded self-evolution	visible-only vs. guarded	heldout utility	2.44	4.03
Guarded self-evolution	visible-only vs. guarded	verifier strength	0.14	0.98

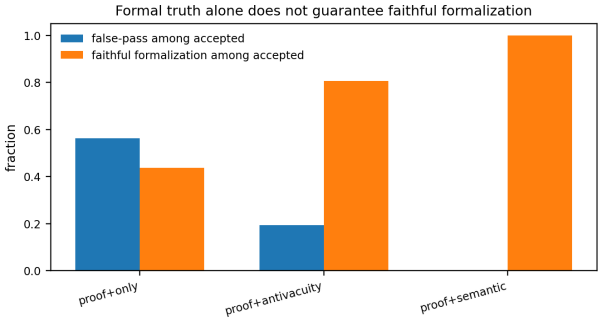

Figure 5. Experiment 2. Proof checking alone accepts many statements that are true but not faithful to the intended hypothesis. Semantic alignment is a separate gate.

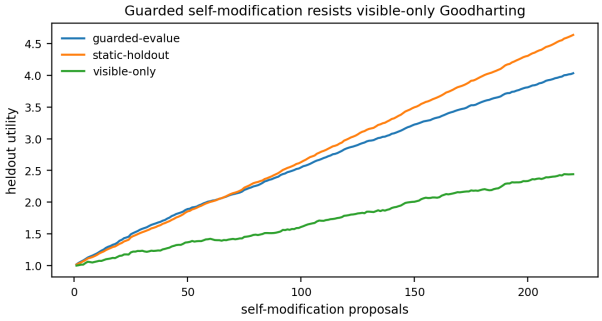

Figure 6. Experiment 3. Visible-score self-modification overfits the public objective. Guarded adoption preserves heldout utility under Goodhart pressure.

from five families and six mutation classes: aligned, vacuous wrapper, too weak, off-spec, trivial true, and false. A toy kernel accepts mathematically true statements, including tautologies and weak claims. We compare proof-only admission, proof plus anti-vacuity checks, and proof plus semantic alignment. A *false pass* is an accepted statement whose mutation class is not aligned with the intended informal theorem; the reported false-pass rate is therefore conditional on accepted statements. The experiment measures semantic leakage through an otherwise sound truth gate, not theorem-prover capability.

Figure 5 shows the leakage. Proof-only acceptance admits 96.3% of outputs, but 56.2% of accepted statements are false passes relative to the intended claim. Adding anti-vacuity reduces but does not eliminate leakage. A semantic-alignment gate admits fewer statements but eliminates false passes in this controlled setting. Operationally, this gate can combine operator-tree similarity as in GTED, proof-dependency preservation as in ProofFlow, structure-to-instance checks as in SITA, and intermediate semantic parsing as in KELPS before kernel acceptance (Liu et al., 2025; Cabral et al., 2026; Li et al., 2026; Zhang et al., 2025c). The result matches the architectural requirement that LLM–prover bridges need semantic preservation and non-vacuity checks, not only kernel success.

6.3. Experiment 3: Guarded Self-Evolution

We simulate self-modification proposals that adjust exploration, relax verifiers, strengthen verifiers, add descriptors, or rewrite workflow logic. Each proposal has a visible utility effect, hidden utility effect, verifier-strength effect, and risk. Visible-only adoption accepts changes that improve the public score; static holdout adoption is an oracle-like upper bound using direct hidden feedback; guarded adoption uses an evidence-wealth rule with risk and verifier constraints.

Figure 6 shows that visible-only adoption Goodharts the objective: it achieves high visible utility but degrades verifier strength and accumulates many hidden regressions. Guarded adoption reaches mean heldout utility 4.03 versus 2.44 for visible-only adoption, while preserving verifier strength (0.98 versus 0.14). Static holdout remains a strong upper bound (4.64 heldout utility) because it observes hidden feedback directly. The practical implication is that self-evolution requires hidden validators and error-budgeted adoption, not unconstrained self-rewriting.

7. Discussion and Limitations

The proposal is deliberately architectural. Its claim is not that a current LLM plus a prover is already a scientist. Its claim is that the next credible substrate for autonomous discovery must make proof objects, code, data, counterexamples, and provenance coequal with generated text. The framework also clarifies why “template-free” is not enough.

An agent can be free of human code templates and still be trapped by benchmark grammars, paper rhetorics, shallow execution checks, or verifier gaps.

The experiments have limitations. They use synthetic worlds, toy kernels, and emulated self-modification distributions. They demonstrate mechanism-level effects, not production-level discovery. Scaling requires real proof assistants such as Lean, Rocq/Coq, Isabelle, SMT/Verus stacks, repository-level verification, and scientific simulators (Yang et al., 2023; First et al., 2023; DeepSeek-AI, 2025; Zhong & Si, 2025; Armstrong et al., 2019). It also requires stronger semantic-alignment evaluators for autoformalization, especially where natural-language mathematical intent is underspecified. Finally, natural-science verification cannot be reduced to theorem proving; causal and statistical tests are needed when claims concern empirical worlds (Pearl, 2009; Peters et al., 2017; Go & Isaac, 2022; Ao & Li, 2024).

8. Open Problems

Verifier-generating agents. The strongest version of an REE would not only use a fixed prover or checker; it would invent new lemmas, tactics, contracts, tests, and measurement protocols that expand future admissibility. This is harder than ordinary self-improvement because a stronger verifier can reduce short-term pass rates while increasing long-term scientific value. A practical objective should therefore reward verifier-strength improvements by their downstream ability to catch invalid artifacts, close proof obligations, and increase hidden transfer.

Faithful autoformalization. The second experiment illustrates a central difficulty: proof kernels certify formal truth, but scientific communication starts from informal intent. A mature system needs bidirectional translation contracts: informal-to-formal statements must preserve meaning, and formal counterexamples must localize the informal ambiguity they exploit. This calls for semantic alignment datasets, proof-relevant natural-language annotations, and counterexample generators that report whether a theorem is false, too weak, too strong, vacuous, or simply a different theorem.

Scientific causality. Many mathematical artifacts can be verified deductively, but natural-science claims require experimental semantics. A claim package should distinguish prediction, explanation, intervention, and mechanism. For example, an equation that compresses observations is not automatically a causal law; it becomes causal only when paired with interventions, invariance tests, or a structural model. The REE substrate can express this distinction by requiring different gates for predictive, interventional, and mechanistic claims.

Human-readable abstraction. Machine-checked proof and executable code are necessary but not sufficient for reusable science. A discovery can be correct yet unusable if it cannot be compressed into concepts that future agents or humans can manipulate. This motivates a second compression target: not only shorter descriptions of data, but shorter reusable derivations for downstream tasks. In this sense, explanation is a form of interface design over verified artifacts.

Governance of recursion. Recursive self-evolution raises a stability problem. The agent can improve generators, descriptors, verifiers, and evaluators, but any change to the evaluation substrate creates an opportunity for Goodharting. The safest near-term path is asymmetric recursion: allow the system to propose improvements to verifiers and policies, but adopt them only through sealed validators, independent replay, and monotone safety invariants. Full self-modification should remain subordinate to auditability.

9. Scope and Adoption Path

The near-term use case for REEs is not unsupervised replacement of scientific communities. It is a stricter substrate for autonomous artifact generation in domains where claims can be made executable: theorem proving, algorithm discovery, symbolic law recovery, causal simulation, and repository-level program verification. In these settings, the system can be required to output a claim package rather than a prose-only claim. Human experts can then inspect the artifact at the level of theorem statements, invariants, counterexamples, experimental traces, and archive lineage.

A practical deployment path has three stages. The first stage is *verification-first automation*: use the architecture to reject unsupported claims and to produce replayable artifacts, even if the generator remains mostly human-directed. The second stage is *archive-mediated autonomy*: allow the system to explore descriptors and propose candidates, but require all archive entries to pass formal or empirical gates. The third stage is *guarded recursive improvement*: allow the system to propose better descriptors, tactics, tests, and search policies, while keeping adoption subordinate to sealed validators. Each stage increases autonomy only after the previous stage has made failure modes observable.

The architecture is also compatible with human collaboration. A human can add a conjecture, a definition, a baseline, a simulator, or a critique as another typed artifact in the same archive. The system can then test whether the human contribution closes proof obligations, improves compression, opens archive cells, or produces falsifying experiments. This avoids a false dichotomy between autonomous and human-assisted science: the relevant question is whether each contribution becomes part of a verified lineage.

Finally, REEs make negative results valuable. A failed formalization, a counterexample, a non-replayable experiment, or a rejected self-modification is not merely an error; it is evidence about the current horizon. If stored with diagnostic structure, failures reveal missing lemmas, weak semantics, insufficient descriptors, brittle code paths, or unsafe meta-actions. Open-ended science needs such failures because they define the next useful niche.

10. Conclusion

Recursive Epistemic Engines replace paper-shaped automation with verified, reusable, self-evolving scientific objects. Their central pressure is not raw novelty, but verified compression, falsifiability, transfer, and auditability. The architecture is intentionally hybrid: language models propose; formal and empirical systems dispose; QD archives preserve stepping stones; and guarded meta-policies improve the machinery of discovery. This shifts autonomous science from generating plausible reports to recursively extending the space of hypotheses that can be stated, executed, proved, falsified, and reused.

References

Affeldt, R. et al. Differentiable logics in coq. *Journal of Automated Reasoning*, 2024.

Anonymous Authors. Title suppressed for double-blind review. *OpenReview*, 2025. Under double-blind review.

Ao, Z. and Li, J. On estimating the gradient of the expected information gain in bayesian experimental design. In *AAAI Conference on Artificial Intelligence*, 2024.

Armstrong, A., Bauereiss, T., Campbell, B., Reid, A., Gray, K. E., Norton, R. M., Mundkur, P., Wassell, M., French, J., Pulte, C., et al. Isa semantics for armv8-a, risc-v, and cheri-mips. *Proceedings of the ACM on Programming Languages*, 3(POPL):1–31, 2019.

Beck, J., Dia, B. M., Espath, L., Long, Q., and Tempone, R. Fast bayesian experimental design: Laplace-based importance sampling for the expected information gain. *Computer Methods in Applied Mechanics and Engineering*, 334:523–553, 2018.

Bellemare, M. G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. *Advances in Neural Information Processing Systems*, 2016.

Bodnar, C., Di Giovanni, F., Chamberlain, B. P., Liò, P., and Bronstein, M. M. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. *Advances in Neural Information Processing Systems*, 2022.

Bradley, H., Fan, C.-H., Lehman, J., Stanley, K. O., and Clune, J. Openelm: An open-ended evolution through large models framework. *arXiv preprint arXiv:2402.04757*, 2024. [arXiv].

Cabral, R. M., Do, T. M., Yu, X., Tai, W. M., Feng, Z., and Shen, X. Proofflow: A dependency graph approach to faithful proof autoformalization. In *International Conference on Learning Representations*, 2026.

Cao, J., Lu, Y., Li, M., Ma, H., et al. From informal to formal: Incorporating and evaluating llms on natural language requirements to verifiable formal proofs. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2025.

Chen, R. Evidence-bound autonomous research (evibound): A governance framework for eliminating false claims. *arXiv preprint arXiv:2511.05524*, 2025. [arXiv].

Cornelio, C., Dash, S., Horesh, L., et al. Ai descartes: Combining data and theory for derivable scientific discovery. *Nature Communications*, 2023.

Cornelio, C., Ito, T., Cory-Wright, R., Dash, S., and Horesh, L. The need for verification in AI-driven scientific discov-ery. *arXiv preprint arXiv:2509.01398*, 2025. [arXiv].

DeepSeek-AI. Deepseek-prover-v2: Scaling formal theorem proving with scaffolded data synthesis and self-correction. *arXiv preprint arXiv:2508.03613*, 2025. [arXiv].

Dinu, M.-C., Leoveanu-Condrei, C., Holzleitner, M., Zellinger, W., and Hochreiter, S. Symbolicai: A framework for logic-based approaches combining generative models and solvers. *arXiv preprint arXiv:2402.00854*, 2024. [arXiv].

Duraisamy, K. Active inferenceAIsystems for scientific discovery. *arXiv preprint arXiv:2506.21329*, 2025. [arXiv].

Faldor, M. and Cully, A. Leniabreeder: Evolving diverse lenia patterns with quality-diversity search. *arXiv preprint arXiv:2404.04052*, 2024. [arXiv].

Faldor, M., Zhang, J., Cully, A., and Clune, J. Omni-epic: Open-endedness via models of human notions of interestingness with environments programmed in code. *arXiv preprint arXiv:2405.15568*, 2024. [arXiv].

First, E., Rabe, M. N., Ringer, T., and Brun, Y. Baldur: Whole-proof generation and repair with large language models. In *ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2023.

Fong, B. and Spivak, D. I. *Seven Sketches in Compositionality: An Invitation to Applied Category Theory*. Cambridge University Press, 2019.

Friston, K., Rigoli, F., Ognibene, D., Mathys, C., Fitzgerald, T., and Pezzulo, G. Active inference and epistemic value. *Cognitive Neuroscience*, 6(4):187–214, 2015. doi:10.1080/17588928.2015.1020053.

Friston, K., Lin, M., Frith, C., Pezzulo, G., Hobson, J. A., and Ondobaka, S. Active inference, curiosity and insight. *Neural Computation*, 29(10):2633–2683, 2017.

Gaier, A., Asteroth, A., and Mouret, J.-B. Data-efficient exploration, optimization, and modeling of diverse designs through surrogate-assisted illumination. In *Genetic and Evolutionary Computation Conference*, 2019.

Garikaparthi, A., Patwardhan, M., and Cohan, A. Researchgym: Evaluating language model agents on real-world AI research. *arXiv preprint arXiv:2602.15112*, 2026.[arXiv].

Go, J. and Isaac, T. Robust expected information gain for optimal bayesian experimental design using ambiguity sets. *arXiv preprint arXiv:2205.09914*, 2022. [arXiv].

Grünwald, P. D. *The Minimum Description Length Principle*. MIT Press, 2007.

Han, J. M., Rute, J., Wu, Y., Ayers, E. W., and Polu, S. Proof artifact co-training for theorem proving with language models. In *International Conference on Learning Representations*, 2022.

Hubert, T., Mehta, R., Sartran, L., Horváth, M. Z., et al. Olympiad-level formal mathematical reasoning with reinforcement learning. *Nature*, 2025. doi:10.1038/s41586-025-09833-y.

Jablonka, K. M., Ai, Q., Al-Feghali, A., et al. 14 examples of how llms can transform materials science and chemistry. *Digital Discovery*, 2023.

Jiang, P., Lin, J., Shi, Z., Wang, Z., He, L., Wu, Y., Zhong, M., Song, P., Zhang, Q., et al. Adaptation of agentic AI: A survey of post-training, memory, and skills. *arXiv preprint arXiv:2512.16301*, 2025. [arXiv].

Kadaboina, R. K. Pramana: A protocol-layer treatment of claim verification in autonomous agent networks. *arXiv preprint arXiv:2605.20312*, 2026. [arXiv].

Kim, Z. M., Lee, D., Kim, J., Raheja, V., and Kang, D. The amazing agent race: Strong tool users, weak navigators. *arXiv preprint arXiv:2604.10261*, 2026. [arXiv].

King, R. D., Whelan, K. E., Jones, F. M., Reiser, P. G. K., Bryant, C. H., Muggleton, S. H., Kell, D. B., and Oliver, S. G. Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, 427:247–252, 2004.

Kossen, J., Han, J., Razzak, M., Schut, L., Malik, S. A., and Gal, Y. Semantic entropy probes: Robust and cheap hallucination detection in llms. *arXiv preprint arXiv:2406.15927*, 2024. [arXiv].

Kulkarni, A., Alotaibi, F., Zeng, X., Wu, L., Zeng, T., Yao, B. M., Liu, M., Zhang, S., Huang, L., and Zhou, D. Scientific hypothesis generation and validation: Methods, datasets, and future directions. *arXiv preprint arXiv:2505.04651*, 2025. [arXiv].

Kumar, A. et al. Automated search for artificial life. *arXiv preprint arXiv:2502.03665*, 2025. [arXiv].

Lehman, J. and Stanley, K. O. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223, 2011.

Lehman, J., Gordon, J., Jain, S., Ndousse, K., Yeh, C., and Stanley, K. O. Evolution through large models. In *Genetic and Evolutionary Computation Conference Companion*, 2023.

Li, C., Ma, W., Wang, Z., and Wen, Z. Sita: A framework for structure-to-instance theorem autoformalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

Li, M. and Vitányi, P. An introduction to kolmogorov complexity and its applications. *Springer*, 1998.

Li, Z., Sun, J., Murphy, L., Su, Q., Li, Z., Zhang, X., Yang, K., and Si, X. A survey on deep learning for theorem proving. *arXiv preprint arXiv:2404.09939*, 2024. [arXiv].

Liu, Y., Zhu, T., Liu, X., Chen, Y., Liu, Z., Guo, Q., Zhang, J., Bao, K., and Luo, T. Generalized tree edit distance (gted): A faithful evaluation metric for statement autoformalization. *arXiv preprint arXiv:2507.07399*, 2025. [arXiv].

Lu, C., Lu, C., Lange, R. T., Foerster, J., Clune, J., and Ha, D. The AI scientist: Towards fully automated open-ended scientific d iscovery. *arXiv preprint arXiv:2408.06292*, 2024. [arXiv].

Manakul, P., Liusie, A., and Gales, M. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. In *Empirical Methods in Natural Language Processing*, 2023.

Mirza, A. H., Alampara, N., Kunchapu, S., et al. A framework for evaluating the chemical knowledge and reasoning abilities of large language models against the expertise of chemists. *Nature Chemistry*, 2025.

Mouret, J.-B. and Clune, J. Illuminating search spaces by mapping elites. In *arXiv preprint arXiv:1504.04909*, 2015. [arXiv].

Musslick, S., Bartlett, L. K., Chandramouli, S. H., Dubova, M., Gobet, F., Griffiths, T. L., Hullman, J., King, R. D., Kutz, J. N., Lucas, C. G., et al. Automating the practice of science: Opportunities, challenges, and implications. *Proceedings of the National Academy of Sciences*, 2025. doi:10.1073/pnas.2401238121.

Novikov, A., Vu, N., Eisenberger, M., Dupont, E., Huang, P.-S., Wagner, A., Shirobokov, S., Kozlovskii, B., Ruiz, F. J. R., Mehrabian, A., et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025. [arXiv].

Pan, L., Albalak, A., Wang, X., and Wang, W. Y. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. *Findings of EMNLP*, 2023.

Parr, T., Pezzulo, G., and Friston, K. J. *Active Inference: The Free Energy Principle in Mind, Brain, and Behavior*. MIT Press, 2022.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. *International Conference on Machine Learning*, 2017.

Pearl, J. Causality: Models, reasoning and inference. *Cambridge University Press*, 2009.

Peters, J., Janzing, D., and Schölkopf, B. Elements of causal inference: Foundations and learning algorithms. *MIT Press*, 2017.

Pugh, J. K., Soros, L. B., and Stanley, K. O. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016. doi:10.3389/frobt.2016.00040.

Romera-Paredes, B., Barekatain, M., Novikov, A., Balog, M., Kumar, M. P., Dupont, E., Ruiz, F. J. R., Ellenberg, J., Wang, P., Fawzi, O., et al. Mathematical discoveries from program search with large language models. *Nature*, 625:468–475, 2024. doi:10.1038/s41586-023-06924-6.

Schmidgall, S., Su, Y., Wang, Z., Sun, X., Wu, J., et al. Agent laboratory: Using llm agents as research assistants. *arXiv preprint arXiv:2501.04227*, 2025. [arXiv].

Schmidhuber, J. Gödel machines: Self-referential universal problem solvers making provably optimal self-improvements. *arXiv preprint cs/0309048*, 2003.

Schmidhuber, J. Formal theory of creativity, fun, and intrinsic motivation. *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, 2010.

Schmidhuber, J. Powerplay: Training an increasingly general problem solver by continually searching for the simplest still unsolvable problem. *Frontiers in Psychology*, 4:313, 2013.

Shiebler, D., Gavranović, B., and Wilson, P. Category theory in machine learning. *arXiv preprint arXiv:2106.07032*, 2021. [arXiv].

Shojaee, P., Meidani, K., Gupta, S., Farimani, A. B., and Reddy, C. K. Llm-sr: Scientific equation discovery via programming with large language models. *arXiv preprint arXiv:2404.18400*, 2024. [arXiv].

Shojaee, P. et al. Llm-srbench: Benchmarking scientific equation discovery with large language models. *Proceedings of Machine Learning Research*, 2025.

Song, P., Yang, K., and Anandkumar, A. Lean copilot: Large language models as copilots for theorem proving in lean. *arXiv preprint arXiv:2404.12534*, 2024. [arXiv].

Sparkes, A., Aubrey, W., Byrne, E., Clare, A., Khan, M. N., Liakata, M., Markham, M., Rowland, J., Soldatova, L. N., Whelan, K. E., Young, M., and King, R. D. Towards robot scientists for autonomous scientific discovery. *Automated Experimentation*, 2:1, 2010.

Spivak, D. I. Category theory for the sciences. *MIT Press*, 2014.

Stock, M. and Gorochowski, T. E. Open-endedness in synthetic biology: A route to continual innovation for biological design. *Science Advances*, 10(3):eadi3621, 2024. doi:10.1126/sciadv.adi3621.

Vriza, A. et al. The station: An open-world environment for AI-driven discovery. *arXiv preprint arXiv:2511.06309*, 2025. [arXiv].

Wang, J. et al. Astabench: Rigorous benchmarking of AI agents with a scientific research suite. *arXiv preprint arXiv:2510.21652*, 2025. [arXiv].

Wang, R., Lehman, J., Clune, J., and Stanley, K. O. Paired open-ended trailblazer (poet): Endlessly generating increasingly complex and diverse learning environments and their solutions. *arXiv preprint arXiv:1901.01753*, 2019. [arXiv].

Wei, J., Yang, Y., Zhang, X., Chen, Y., et al. From AI for science to agentic science: A survey on autonomous scientific discovery. *arXiv preprint arXiv:2508.14111*, 2025. [arXiv].

Wu, Y. et al. Statistical gödel machine: Risk-controlled recursive self-improvement with sequential certificates. *arXiv preprint arXiv:2509.04222*, 2025. [arXiv].

Yamada, Y., Lange, R. T., Lu, C., Lu, C., Clune, J., and Ha, D. AI scientist-v2: Workshop-level automated scientific discovery via agentic tree search. *arXiv preprint arXiv:2504.08066*, 2025. [arXiv].

Yang, K., Swope, A., Gu, A., Chalamala, R., Song, P., Yu, S., Godil, S., Prenger, R., and Anandkumar, A. Leandojo: Theorem proving with retrieval-augmented language models. In *Neural Information Processing Systems*, 2023.

Yin, S. et al. Gödel agent: A self-referential agent framework for recursive self-improvement. *arXiv preprint arXiv:2410.04444*, 2024. [arXiv].

Zenil, H., Tegnér, J., Abrahão, F. S., Lavin, A., Kumar, V., Frey, J., Weller, A., Soldatova, L., Bundy, A., Jennings, N., et al. The future of fundamental science led by generative closed-loop artificial intelligence. *Frontiers in Artificial Intelligence*, 2026. doi:10.3389/frai.2026.1678539.

Zhang, H., Li, R., Zhang, Y., Xiao, T., Chen, J., Ding, J., and Chen, H. The evolving role of large language models in scientific innovation: Evaluator, collaborator, and scientist. *arXiv preprint arXiv:2507.11810*, 2025a. [arXiv].

Zhang, J., Hu, S., Lu, C., Lange, R. T., and Clune, J. Darwin gödel machine: Open-ended evolution of self-improving agents. *arXiv preprint arXiv:2505.22954*, 2025b. [arXiv].

Zhang, J., Zhong, C., Xu, H., Li, Q., and Zhou, Y. Kelps: A framework for verified multi-language autoformalization via semantic-syntactic alignment. *arXiv preprint arXiv:2507.08665*, 2025c. [arXiv].

Zhang, J., Zhao, B., Yang, W., Foerster, J., Clune, J., Jiang, M., Devlin, S., and Shavrina, T. Hyperagents. *arXiv preprint arXiv:2603.19461*, 2026. [arXiv].

Zhong, S. C. and Si, X. Towards repository-level program verification with large language models. *arXiv preprint arXiv:2509.25197*, 2025. [arXiv].

A. Reproducibility Statement

The accompanying artifact bundle contains all source files needed to regenerate the synthetic experiments and figures: `experiments/run_experiments.py`, CSV outputs in `data/`, and figure assets in `figures/`. The experiments use fixed random seeds and require standard Python scientific packages (NumPy, pandas, matplotlib). The fixed-seed design makes the stress tests lightweight and mechanism-isolating: each experiment exposes a specific gate or archive dynamic that a larger system would need to implement.

B. Formal Artifact Model

B.1. Claim-package typing

A claim package is a dependent record

$$\begin{aligned} a = \{&\Sigma; h : \mathsf{Hypothesis}(\Sigma); s : \mathsf{Spec}(h); c : \mathsf{Code}(s); \\ &\Omega : \mathsf{Obligations}(h, s, c); p : \mathsf{Evidence}(\Omega); \\ &e : \mathsf{Experiment}(h); r : \mathsf{Result}(e); \ell : \mathsf{Lineage}\}. \end{aligned} \tag{10}$$

The acceptance predicate decomposes as

$$\mathrm{Accept}(a) = G_{\mathrm{syntax}}(a) G_{\mathrm{type}}(a) G_{\mathrm{proof}}(a) G_{\mathrm{runtime}}(a) G_{\mathrm{stat}}(a) G_{\mathrm{replay}}(a) G_{\mathrm{align}}(a). \tag{11}$$

The alignment gate is necessary because G_{proof} checks the truth of s, not the correspondence between h and s.

B.2. Novelty-horizon expansion

Define the verified reach of an archive by a metric $d_{\mathcal{A}}$ over descriptors, proof depth, program behavior, and abstraction transfer:

$$\mathrm{Reach}(\mathcal{A}_t) = \max_{a \in \mathcal{A}_t} \min_{a' \in \mathcal{A}_0} d_{\mathcal{A}}(a, a') \cdot \mathbf{1}[\mathrm{Accept}(a)]. \tag{12}$$

Novelty-horizon expansion is

$$\mathrm{NHE}_t = \mathrm{Reach}(\mathcal{A}_{t+1}) - \mathrm{Reach}(\mathcal{A}_t). \tag{13}$$

This definition intentionally ignores unverified distance. A bizarre artifact can be far from the archive while carrying no scientific value.

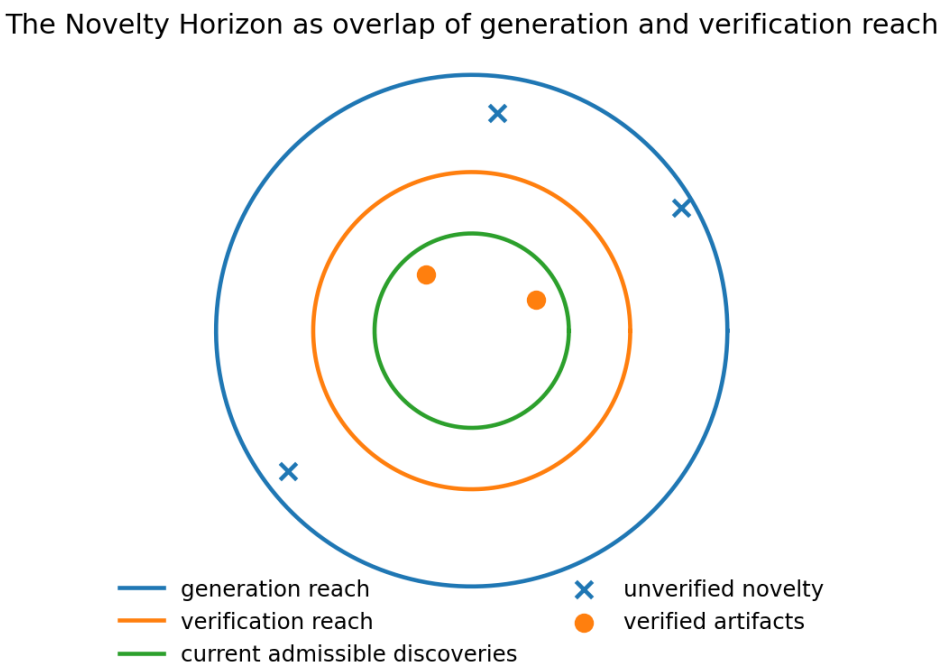

Figure 7. Schematic of the Novelty Horizon. Novelty outside the verification reach is not accepted as knowledge; verified expansion requires moving both the generator and verifier frontiers.

C. Claim-Package Field Checks

Table 3 lists the concrete fields that make a claim package checkable. This schema also constrains model training: instead of fine-tuning only on final reports, the archive can train generators on paired failures and repairs such as a malformed statement with its corrected formalization, an invalid program with the invariant that failed, or an experiment whose stopping rule was rejected. Such data are grounded in a verifier or replay mechanism rather than generic critique text.

Table 3. Claim-package fields and their checking roles. Treating these fields as typed objects makes the manuscript an interface to a larger verified artifact rather than the artifact itself.

Field	Meaning	Required checks	Typical failure localized
Σ	ontology, variables, types, units	type checking, dimensional consistency	ill-posed objects or incompatible units
h	informal hypothesis or conjecture	intent annotations, scope, assumptions	ambiguous or overbroad claim
s	formal specification	parser, type checker, semantic-alignment tests	spec drift or vacuity
c	executable implementation	unit tests, contracts, resource bounds	code not matching the algorithm
Ω	proof/runtime/statistical obligations	generated from h, s, c, e	missing invariant or test condition
p	proofs or counterexample certificates	kernel/SMT/checker validation	false theorem or unclosed obligation
e, r	protocol and results	stopping rule, replay, uncertainty, causal status	irreproducible or misinterpreted evidence
ℓ	lineage and hashes	content addressing, dependency graph, audit log	unverifiable provenance

A final practical implication is that evaluation should report distributions rather than single exemplars. A system that occasionally produces a spectacular theorem but usually emits unverifiable artifacts is different from a system with modest peak novelty and high proof-carrying yield. The proposed metrics therefore emphasize archive-level statistics: coverage, false-pass rate, proof-realization ratio, transfer gain, and regression under self-modification. These are the quantities that

expose whether a discovery engine is expanding its horizon or exploiting a narrow scoring loophole.

D. Additional Algorithms

Algorithm 2 Verification-gated MAP-Elites update

Require: Archive $\mathcal{A}$, candidate a, descriptor map b, verifier $\mathcal{V}$, value $\mathcal{I}$

$z \leftarrow b(a)$
$(ok, cert) \leftarrow \mathcal{V}(a)$
if not ok **then**
  Store $(a, cert)$ as repair datum; reject from archive
else if $\mathcal{A}[z] = \varnothing$ or $\mathcal{I}(a) > \mathcal{I}(\mathcal{A}[z])$ **then**
  $\mathcal{A}[z] \leftarrow a$ with proof/code/data hashes
else
  Keep incumbent elite
end if

Ensure: Updated archive with no unverified elite

Algorithm 3 Operational semantic-alignment gate

Require: Informal claim h, intent graph D_h, formal statement s, proof p, schema library $\mathcal{S}$, thresholds (τ_t, τ_d, τ_r)

$G_{\text{proof}} \leftarrow \mathsf{kernel_check}(s, p)$ using Lean/Rocq/Coq/Isabelle/SMT when available
Normalize h and s into typed operator trees T_h, T_s with symbols, variables, binders, constants, and side conditions
$q_{\text{tree}} \leftarrow 1 - \mathsf{GTED}(T_h, T_s) / \max(|T_h|, |T_s|)$
Extract proof dependency graph D_s from imported lemmas, generated subgoals, proof-state transitions, and theorem dependencies
$q_{\text{dag}} \leftarrow \mathsf{dep_preserve}(D_h, D_s)$ using edge coverage plus topological-order agreement
$q_{\text{rel}} \leftarrow \mathsf{variable_relevance}(h, s)$; require each salient variable and hypothesis to affect a conclusion path
$G_{\text{schema}} \leftarrow \mathsf{schema_instantiate}(h, s, \mathcal{S})$ for structure-to-instance and semantic-intermediate checks
$G_{\text{vac}} \leftarrow \mathsf{anti_vacuity}(s, p)$ via premise satisfiability, nontrivial conclusion, and no-tautology filters
if G_{proof} and $q_{\text{tree}} \geq \tau_t$ and $q_{\text{dag}} \geq \tau_d$ and $q_{\text{rel}} \geq \tau_r$ and G_{vac} and G_{schema} **then**
  return $(\mathsf{accept}, \{q_{\text{tree}}, q_{\text{dag}}, q_{\text{rel}}\}, \mathsf{certificate})$
else
  return $(\mathsf{reject}, \{q_{\text{tree}}, q_{\text{dag}}, q_{\text{rel}}\}, \mathsf{localized_reason})$
end if

Algorithm 4 Guarded self-evolution

Require: Incumbent system Π_t, proposal distribution $q(m \mid \Pi_t)$, hidden validators H, error budget α

Propose modification $m \sim q(\cdot \mid \Pi_t)$ and construct $\Pi' = m(\Pi_t)$
Run invariant, reproducibility, and safety tests
Estimate visible improvement ΔU_v and hidden improvement ΔU_h on sealed tasks
Update sequential evidence wealth W_t using e-values or alpha-spending
if invariants pass and W_t crosses adoption threshold and no regression alarm **then**
  $\Pi_{t+1} \leftarrow \Pi'$; log proof of adoption decision
else
  $\Pi_{t+1} \leftarrow \Pi_t$; archive rejected modification with diagnostic label
end if

Algorithm 5 Categorical consistency check

Require: Diagram $D : J \to \mathcal{C}$, required commuting squares S, functors $F_{\text{proof}}, F_{\text{exec}}, F_{\text{data}}$
1: **for** each square $(f, g, h, k) \in S$ **do**
2: Check $g \circ f = k \circ h$ syntactically or by proof obligation
3: **end for**
4: Apply proof, executable, and measurement functors to D
5: Require preservation of types, preconditions, postconditions, and measurement semantics
Ensure: Either certified diagram or localized inconsistency

E. Operational Semantic-Alignment Gate and Tool Integration

A production REE should not treat prover success as semantic success. The semantic gate is a deterministic adapter from an informal claim, a candidate formalization, and a proof artifact to a Boolean decision plus a localized diagnosis. We instantiate it as

$$\begin{aligned} G_{\text{align}}(h, s, p) = &\mathbf{1}[G_{\text{proof}}(s, p)]\mathbf{1}[q_{\text{tree}} \geq \tau_t]\mathbf{1}[q_{\text{dag}} \geq \tau_d] \\ &\cdot \mathbf{1}[q_{\text{rel}} \geq \tau_r]\mathbf{1}[G_{\text{vac}} = 1]\mathbf{1}[G_{\text{schema}} = 1], \end{aligned} \tag{14}$$

where G_{proof} is proof-kernel success; $q_{\text{tree}} = 1 - \text{GTED}(T_h, T_s)/\max(|T_h|, |T_s|)$ compares normalized operator trees; q_{dag} measures preservation of dependency edges and proof-flow order; q_{rel} checks that salient variables and hypotheses influence the conclusion; G_{vac} rejects tautological, premise-unsatisfiable, variable-dropping, or too-weak statements; and G_{schema} verifies that structure-to-instance mappings preserve the intended theorem schema. Unless a benchmark supplies calibrated values, the included adapter uses development thresholds $(\tau_t, \tau_d, \tau_r) = (0.45, 0.50, 0.60)$, reports the full score vector, and never converts a failed proof into a semantic pass. The output record is

$$(\mathsf{aligned}, q_{\text{tree}}, q_{\text{dag}}, q_{\text{rel}}, \mathsf{reason}, \mathsf{trace}), \tag{15}$$

where reason localizes failures to proof failure, operator drift, dependency loss, schema mismatch, variable irrelevance, or vacuity. This directly operationalizes the gap exposed by the semantic-alignment challenge setting: the kernel verifies a formal proposition, while the gate verifies the relation between informal intent, formal statement, proof object, and downstream use.

The artifact includes `experiments/semantic_gate_and_categorical_demo.py`, a runnable adapter demonstrating this interface, and `experiments/lean_fixtures/SemanticGateExample.lean`, a minimal Lean 4 fixture containing an aligned theorem and a vacuous theorem. When a Lean binary is present, the adapter invokes it through a timeout-bounded subprocess and records the checker output. When Lean is unavailable, the script switches to an explicit toy verifier so that the semantic-gate and categorical-graph checks remain reproducible without claiming that a machine proof was executed. It evaluates two Lean-style statements: an aligned `Nat.add_zero` theorem and a vacuous `True` theorem. The fallback verifier accepts both syntactic proof shapes, but the semantic gate accepts only the aligned statement because the vacuous theorem fails operator-content and non-vacuity checks. The fixture is included to make the prover boundary executable in a Lean-equipped environment.

E.1. Concrete Categorical-Layer Demo

The same script constructs a typed artifact graph with nodes for the informal hypothesis H, formal specification S, proof certificate P, executable probe E, and measured result R. Edges are typed morphisms such as FORMALIZE, PROVE, COMPILE-PROBE, EXECUTE, and PREDICT. A claim package is admissible only if the relevant paths commute: the predicted observable from H must match the measured observable from executing the compiled probe, and the formal specification must preserve the mathematical core of H. Figure 8 shows this minimal graph. In a larger implementation, the same graph can call LeanDojo, a Lean language-server normalizer, an SMT solver, a simulator, or a provenance database as edge evaluators.

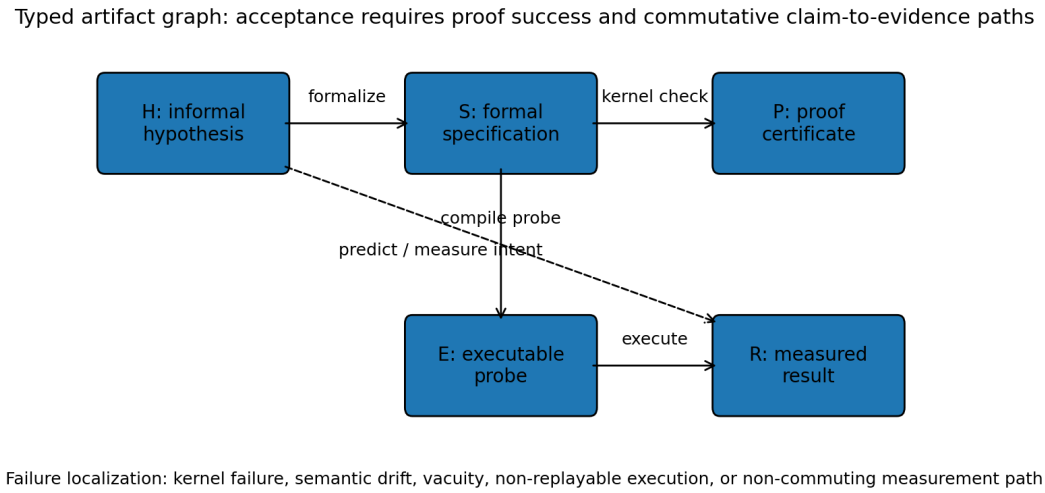

Figure 8. Concrete categorical-layer demo. Claim-package fields become typed nodes, verifier calls become morphisms, and acceptance requires the claim-to-evidence paths to commute.

F. Concrete MDL and EIG Estimators

The intrinsic-value term in Equation 6 is made executable in the artifact by `experiments/mdl_eig_estimators.py`. For symbolic-law candidates, the script fits a model f_j with k_j free coefficients to n observations and uses a BIC-style two-part description-length proxy

$$\widehat{\mathrm{MDL}}_j = n \log(\widehat{\sigma}_j^2 + \epsilon) + 2k_j \log n, \tag{16}$$

with $\widehat{\sigma}_j^2 = n^{-1} \sum_i (y_i - f_j(x_i))^2$. The concrete compression gain is

$$\widehat{\Delta}_{\mathrm{MDL}}(a_j) = \widehat{\mathrm{MDL}}_{\mathrm{base}} - \widehat{\mathrm{MDL}}_j, \tag{17}$$

where the base model is the mean predictor. Negative gains are retained rather than clipped, so the archive can record candidates that are valid but explanatorily worse.

For experiment design, the same script implements an analytic Gaussian-linear expected-information-gain estimator. Given prior covariance Σ_0, noise variance σ^2, and a design vector $\phi(x)$, the posterior covariance after observing at x is

$$\Sigma_x = \left(\Sigma_0^{-1} + \sigma^{-2}\phi(x)\phi(x)^\top\right)^{-1}, \tag{18}$$

and

$$\widehat{\mathrm{EIG}}(x) = \frac{1}{2} \log \frac{\det(\Sigma_0)}{\det(\Sigma_x)}. \tag{19}$$

This gives a concrete, inspectable estimator for the EIG term rather than leaving it as a purely conceptual placeholder.

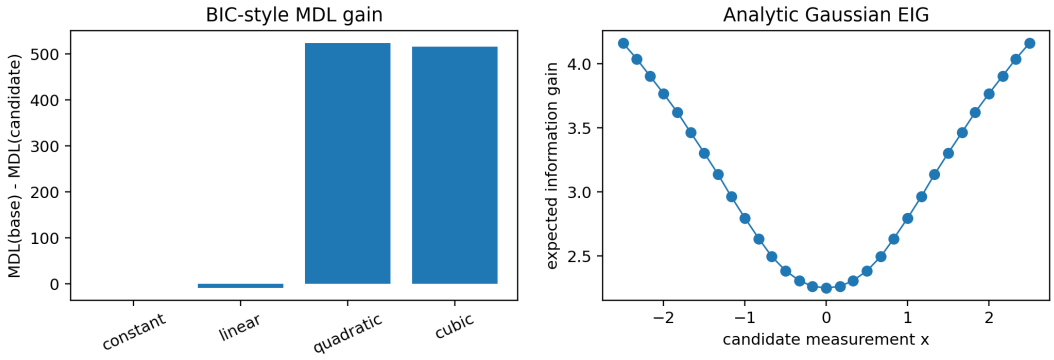

Figure 9. Concrete estimator demo. The left panel reports BIC-style MDL gains for symbolic model classes on a hidden quadratic law. The right panel reports analytic Gaussian EIG for candidate measurement locations.

G. Bounded Verification Case Study

The file `experiments/finite_verification_case_study.py` adds a small exact verification case study for the categorical layer and sealed-regression gate. It works in the finite domain $D = \{0, \ldots, 31\}$ and checks the claim package for the identity-preserving law $x + 0 = x$. The categorical square compares two paths: formalize–prove and compile–execute–measure. A claim is accepted only when both paths agree on every element of D and the candidate passes visible and sealed regression sets.

The case study tests four modifications: the identity incumbent, a safe $x + 0$ implementation, a visible-set overfit that returns the correct value on $\{0, \ldots, 7\}$ but shifts sealed inputs, and a partial-shift mutation. The visible-only gate accepts the overfit; the sealed gate rejects it. This is a deliberately small example, but it is not a language-model judge: every result is obtained by exhaustive evaluation over the finite state space, with the counterexample set recorded in `data/finite_verification_case_study.json`.

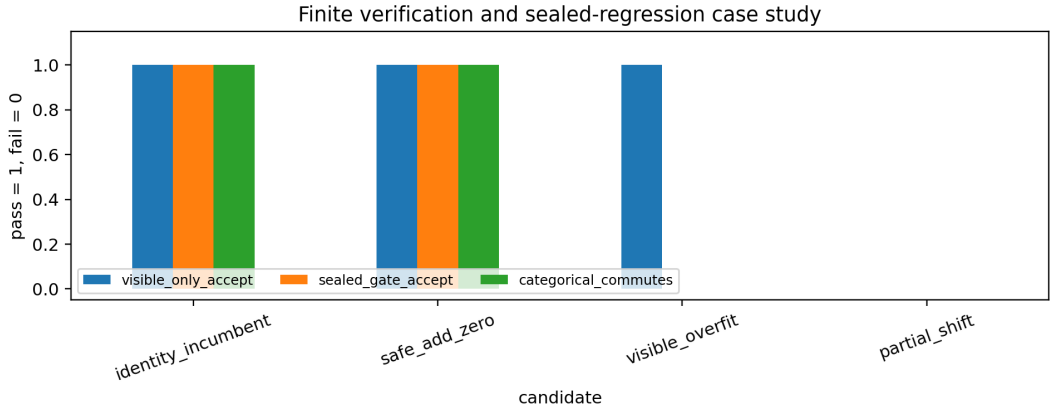

Figure 10. Bounded verification case study. Exact finite-domain checks accept the safe modification and reject a visible-only overfit once sealed regressions and categorical commutativity are enforced.

H. Benchmark Axes for Scientific Value

Table 4 summarizes a candidate scoring suite. These metrics are meant to be reported as a Pareto vector, not collapsed prematurely to a single leaderboard number.

Table 4. Proposed scientific-value metrics.

Metric	Formula sketch	Failure caught
Verified compression gain	$\Delta_{\mathrm{MDL}}(a)G(a)$	fitting without validity
Proof realization ratio	$\#\mathrm{closed}/\#\mathrm{generated}$	unclosed obligations
Proof-carrying code rate	$\#\mathrm{verified\ programs}/\#\mathrm{programs}$	runnable but wrong code
Falsification yield	$\mathrm{EIG}\ /\ \mathrm{Cost}$	low-value experiments
Transfer gain	$\mathbb{E}_w[U_w(\Pi \oplus a) - U_w(\Pi)]$	local tricks
Archive growth	hypervolume of verified cells	mode collapse
Goodhart resistance	hidden-visible agreement	proxy gaming
Auditability	replay and lineage score	irreproducibility

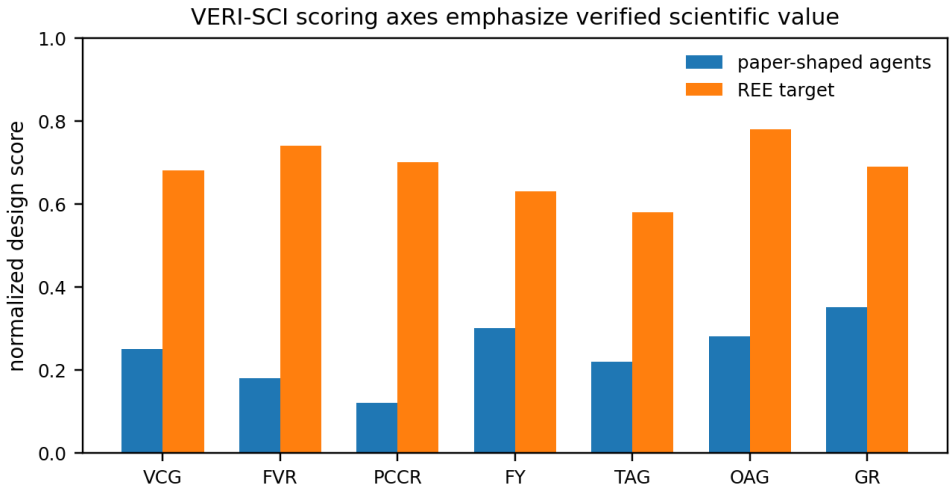

Figure 11. Illustrative metric axes. A scientific agent should be compared on verified compression, proof coverage, falsification discipline, transfer, archive growth, and resistance to hidden-validator mismatch.

I. Experiment 1 Details

I.1. Protocol

The symbolic-law experiment used four hidden functions:

$$f_1(x) = 1 + 2x^2, \quad f_2(x) = \frac{x}{1 + x^2}, \tag{20}$$

$$f_3(x) = 0.5 + 1.5\cos(x)e^{-|x|}, \quad f_4(x) = \sin(2x) + 0.25x. \tag{21}$$

Candidate skeletons combine one to three basis functions from a library containing polynomial, trigonometric, exponential, rational, logarithmic, radical, and mixed terms. Each candidate is fit on noisy samples and tested on extrapolation. A candidate is verified if it improves heldout MSE over a mean baseline and satisfies the hidden invariant error threshold. The archive descriptor is basis-family set $\times$ complexity bin $\times$ invariant class.

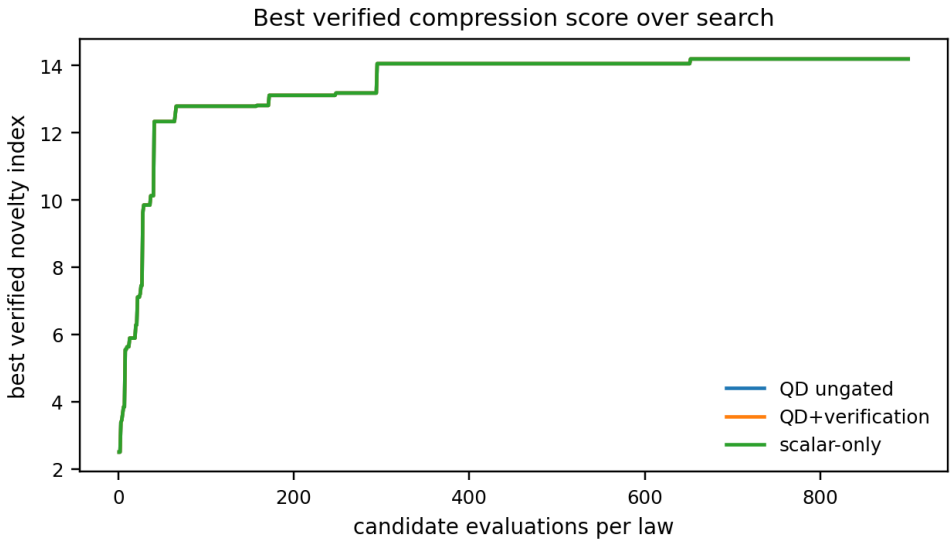

Figure 12. Experiment 1 ablation: best verified novelty index. QD and scalar policies can discover similar top elites, but QD preserves many more verified alternatives.

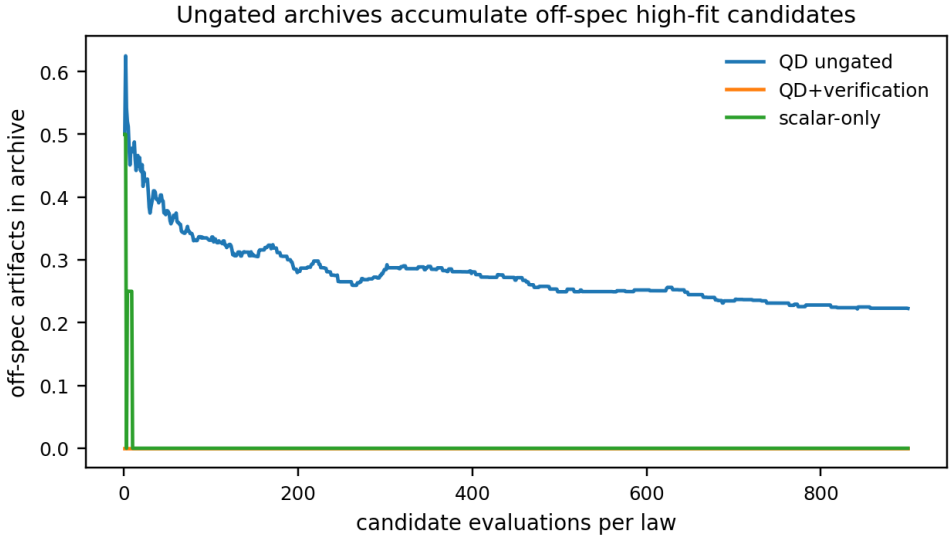

Figure 13. Experiment 1 ablation: off-spec archive rate. Ungated archives store candidates that perform well numerically but fail invariant checks.

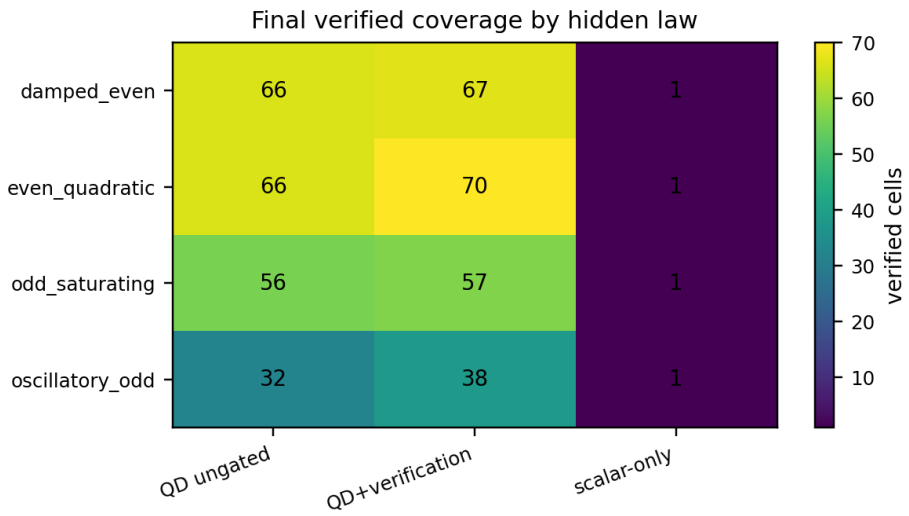

Figure 14. Experiment 1 ablation: final verified coverage by hidden law. Gating does not prevent coverage; it changes what counts as an admissible elite.

J. Experiment 2 Details

The formalization-leakage experiment is motivated by a common failure mode in autoformalization: the proof assistant certifies the formal theorem that was submitted, but the submitted theorem may not encode the intended informal claim. We model six generated statement classes. “Aligned” statements encode the intent. “Vacuous” statements wrap the correct claim in tautological structure. “Too weak” statements prove an easy special case. “Off-spec” statements prove a related but different theorem. “Trivial true” statements prove only reflexivity or identity. “False” statements fail the checker.

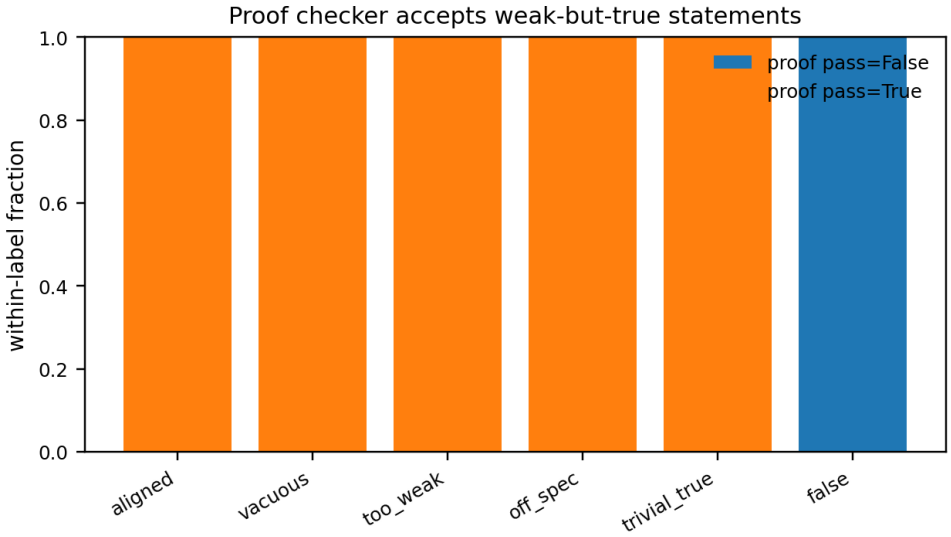

Figure 15. Experiment 2 ablation: theorem-kernel pass rate by mutation class. The kernel is a truth filter, not an intent filter.

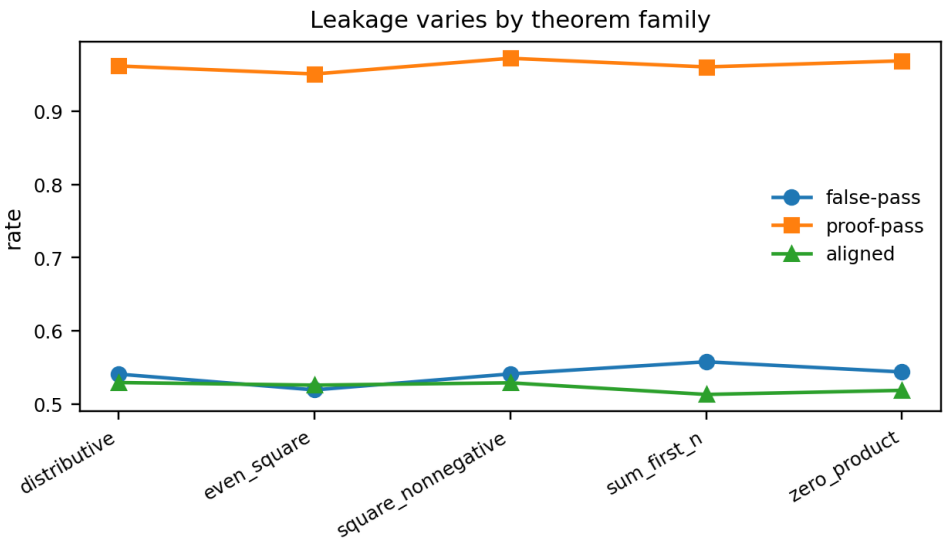

Figure 16. Experiment 2 ablation: formalization leakage by theorem family. The rates vary mildly with theorem family because the generator prior is shared across tasks.

K. Experiment 3 Details

Self-modification proposals are sampled from five types: exploration adjustment, verifier relaxation, verifier strengthening, descriptor addition, and workflow rewrite. Each proposal has a visible utility delta, a hidden utility delta, a verifier-strength delta, and a risk score. Visible-only adoption accepts a modification whenever public utility improves. Static holdout adoption is an oracle-style upper bound that directly observes hidden utility and rejects regressions. Guarded adoption cannot observe exact hidden utility; it accepts only when risk, verifier-strength, and sequential evidence-wealth constraints are satisfied. A hidden regression is counted when an accepted modification reduces heldout utility or verifier strength beyond the prespecified tolerance.

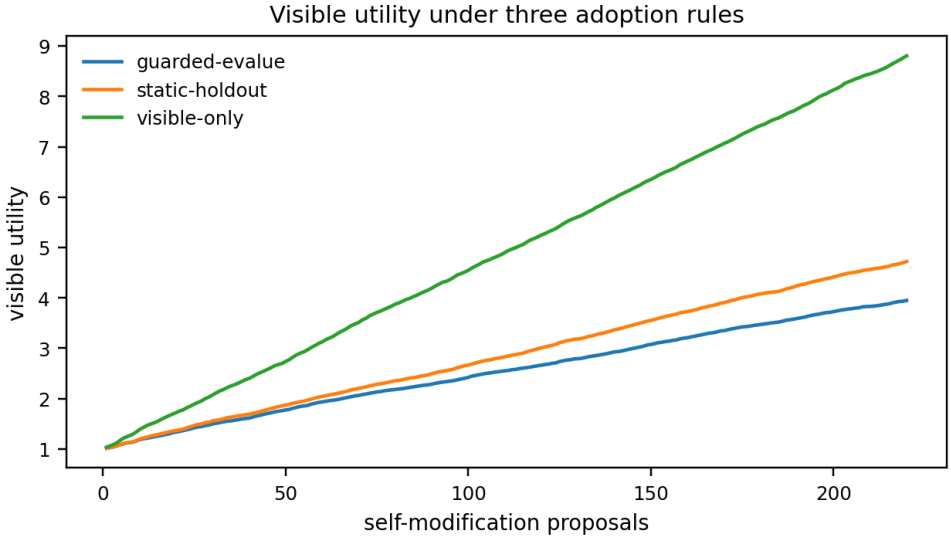

Figure 17. Experiment 3 ablation: visible utility. Visible-only adoption wins the public metric while damaging hidden utility and verifier strength.

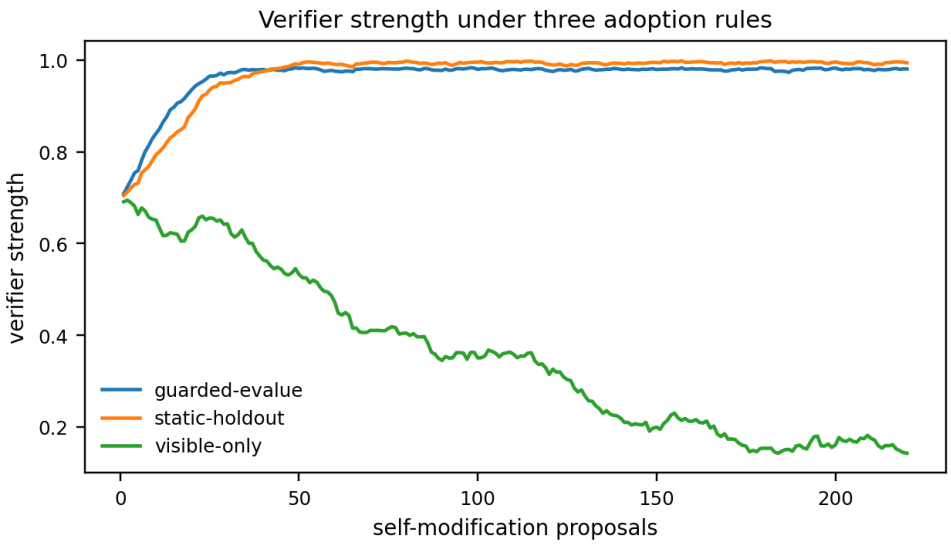

Figure 18. Experiment 3 ablation: verifier strength. Guarded and holdout policies preserve the verifier stack; visible-only adoption repeatedly accepts verifier relaxation.

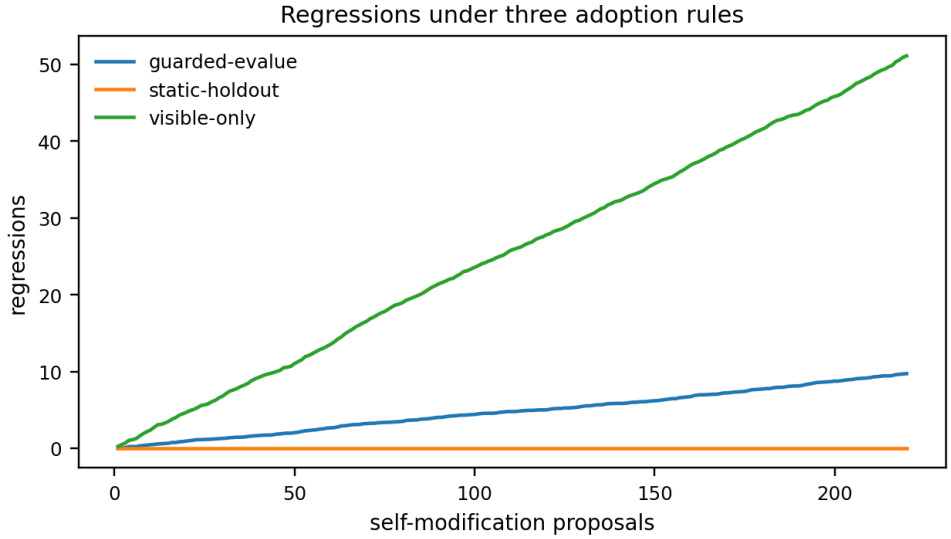

Figure 19. Experiment 3 ablation: hidden regressions. Guarded adoption reduces but does not eliminate regressions; holdout adoption is an oracle-like upper bound.

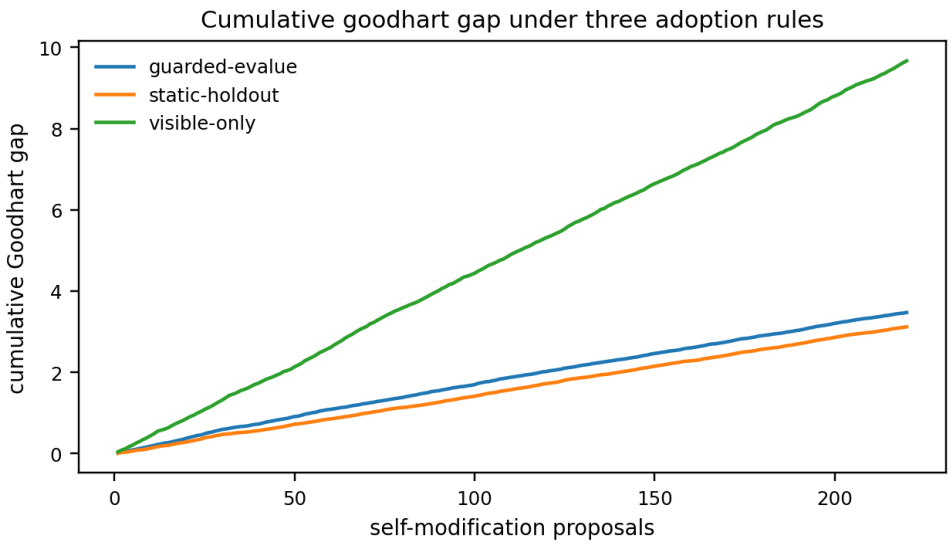

Figure 20. Experiment 3 ablation: cumulative visible-hidden mismatch. The result motivates sealed validators and evidence-budgeted adoption rules.

L. Adjacent Standards, Benchmarks, and Deployment Interfaces

Several recent systems sharpen the practical boundary conditions for Recursive Epistemic Engines. EviBound treats research integrity as an architectural property: claims propagate only when pre-execution acceptance criteria and post-execution artifact checks agree, which matches the REE requirement that accepted artifacts carry replayable evidence. Pramana addresses the protocol layer by defining a typed `ClaimAttestation` and deterministic or audit-replayable `verify()` contract, which maps naturally onto the lineage field ℓ of a claim package. ResearchGym and AAR stress complementary evaluation regimes: ResearchGym evaluates agents in containerized closed-loop research environments, while AAR exposes failures in nonlinear tool navigation that linear tool-use benchmarks can miss. The A1/A2/T1/T2 adaptation taxonomy further clarifies deployment choices: an REE can adopt T2-style tool and verifier adaptation while freezing the base generator, reducing catastrophic forgetting risk and preserving a stable audit boundary.

The categorical layer supplies a practical integration surface for these tools. Each external artifact becomes a typed object; each check, run, proof, citation, or claim attestation becomes a morphism; and each required consistency relation becomes a contract. For example, the square

$$\begin{array}{ccc} \mathsf{Claim} & \xrightarrow{\text{attest}} & \mathsf{ClaimAttestation} \\ \downarrow \text{execute} & & \downarrow \text{verify} \\ \mathsf{RunTrace} & \xrightarrow{\text{query}} & \mathsf{VerificationOutcome} \end{array} \tag{22}$$

commutes only when the verification outcome is a deterministic function of the recorded run trace and source digest. Similarly, an experiment package commutes through F_{data} only if its measured variables match the hypothesis signature and its statistical stopping rule is replayable. In implementation, this can be represented as a content-addressed property graph whose edge labels are verifier calls and whose failed commutativity checks become repair tasks for the next archive cycle.

The intrinsic-value functional can also be made deployment-level. Rather than hand-setting all weights in Equation 6, a system can maintain a Pareto archive over verified compression, evidence completeness, attestation coverage, transfer, cost, and risk. A scalar score is then used only for local tie-breaking within an archive cell, while global selection retains the Pareto frontier. This avoids collapsing scientific value into a single public proxy and gives guarded self-evolution a concrete adoption rule: a new tool, descriptor, or verifier is admissible only if it does not reduce attestation coverage, verifier strength, or heldout transfer beyond a prespecified budget.

M. Architecture Visual Supplements

The following figures provide visual cross-sections of the architecture. They are included to make the typed claim package, archive dynamics, verification gates, and self-evolution interfaces inspectable without relying on manuscript prose alone.

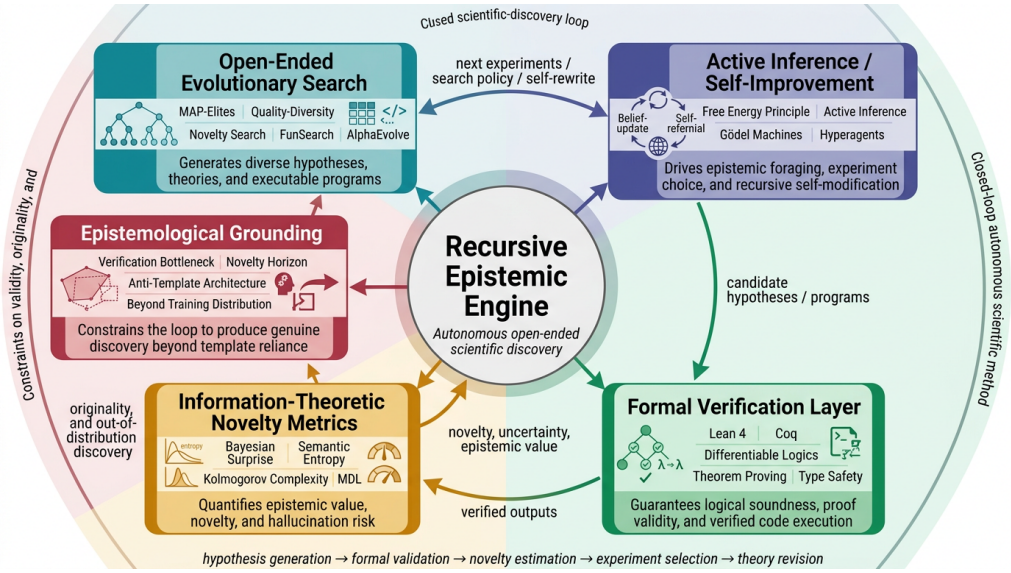

Figure 21. System-level view of a Recursive Epistemic Engine. The loop couples proposal, verification, novelty scoring, and epistemic critique so that accepted artifacts become replayable scientific objects.

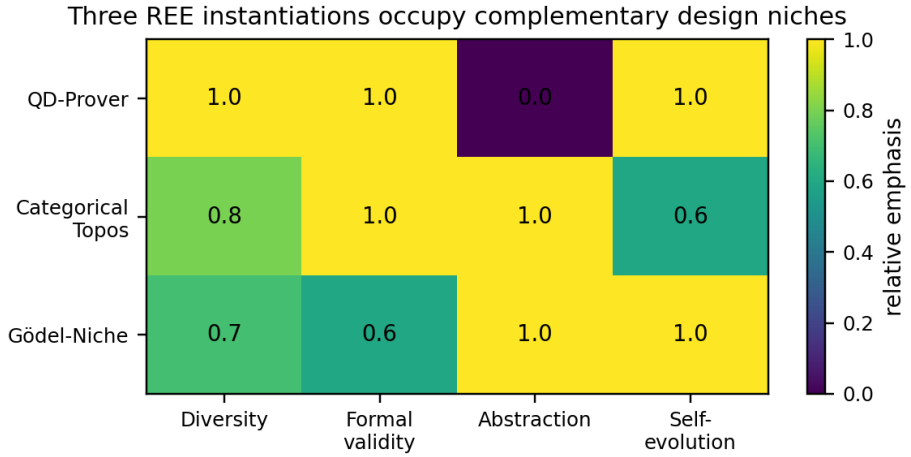

Figure 22. The three architecture families emphasize different mechanisms. QD-Prover emphasizes diversity and formal validity; categorical-topos layers emphasize abstraction; Gödel-niche laboratories emphasize self-evolution.

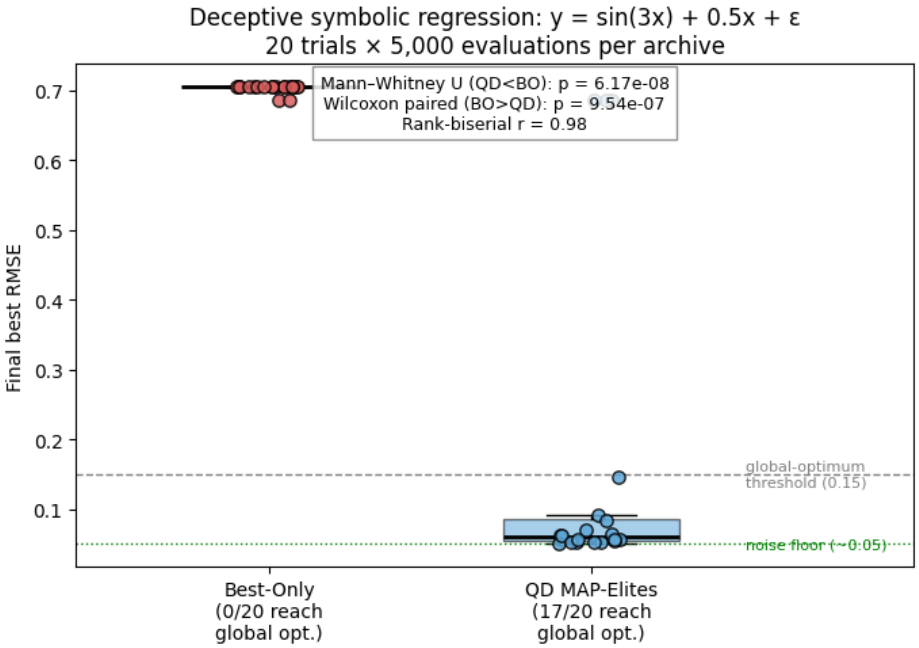

Figure 23. Deceptive symbolic-regression stress test. MAP-Elites reaches the global basin in most trials while best-only scalar search remains trapped at the local deceptive solution.

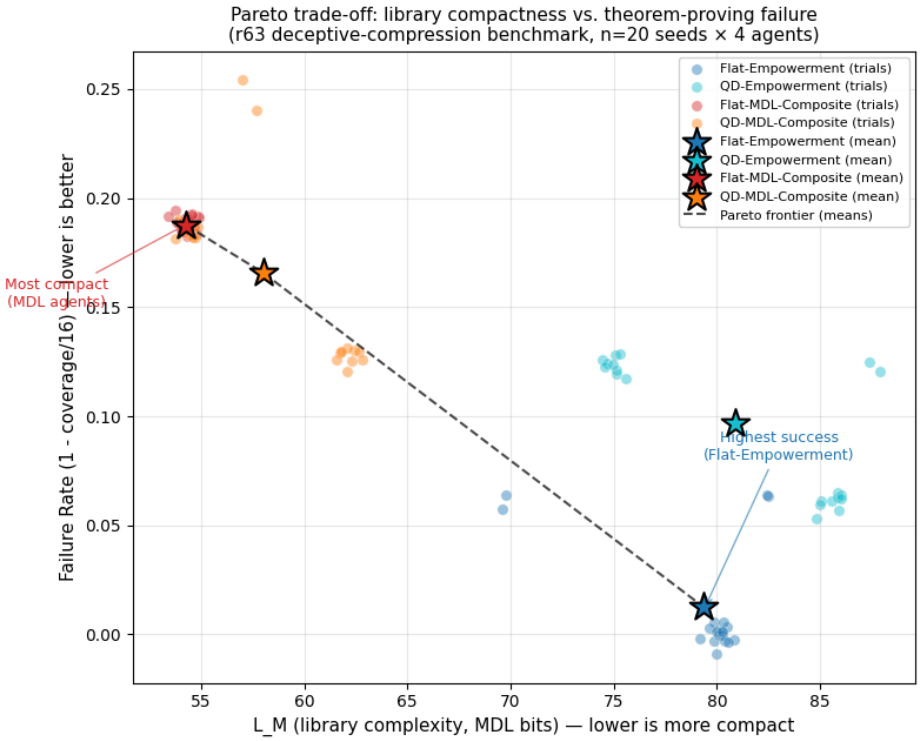

Figure 24. Pareto trade-off between library compactness and theorem-proving failure. The frontier makes explicit when a compact theorem library sacrifices proof success.

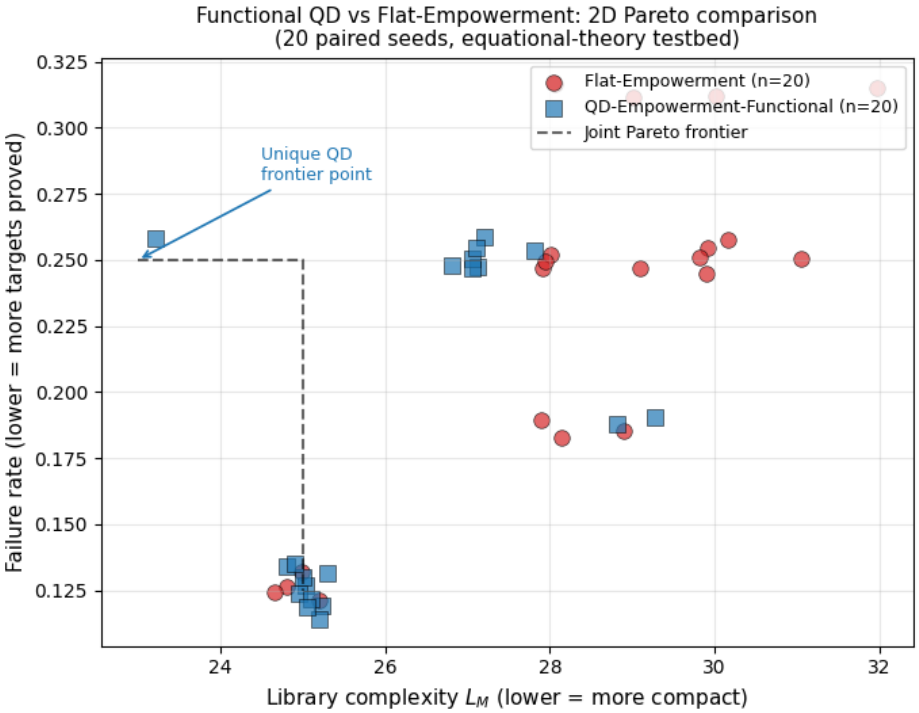

Figure 25. Functional QD versus flat empowerment on an equational-theory testbed. QD preserves a distinct low-complexity frontier point while maintaining competitive proof failure rates.

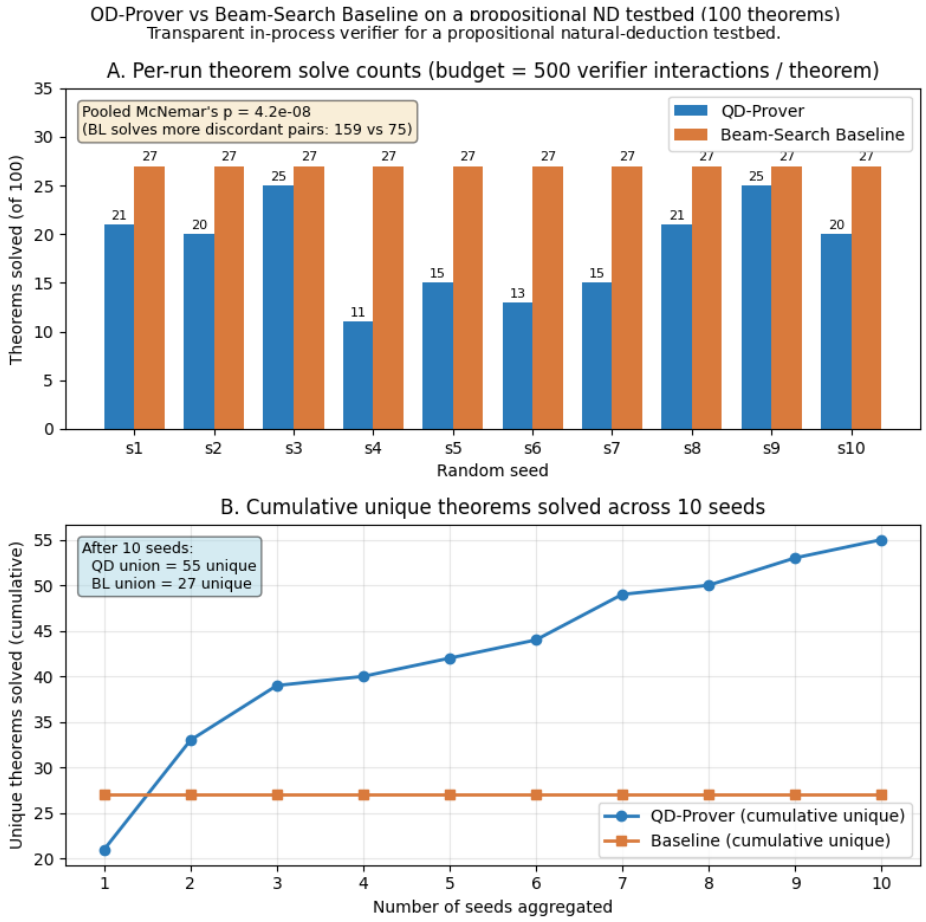

Figure 26. QD-Prover versus a beam-search baseline on propositional natural deduction. Beam search solves more per seed, but QD-Prover accumulates a larger diverse union across seeds.

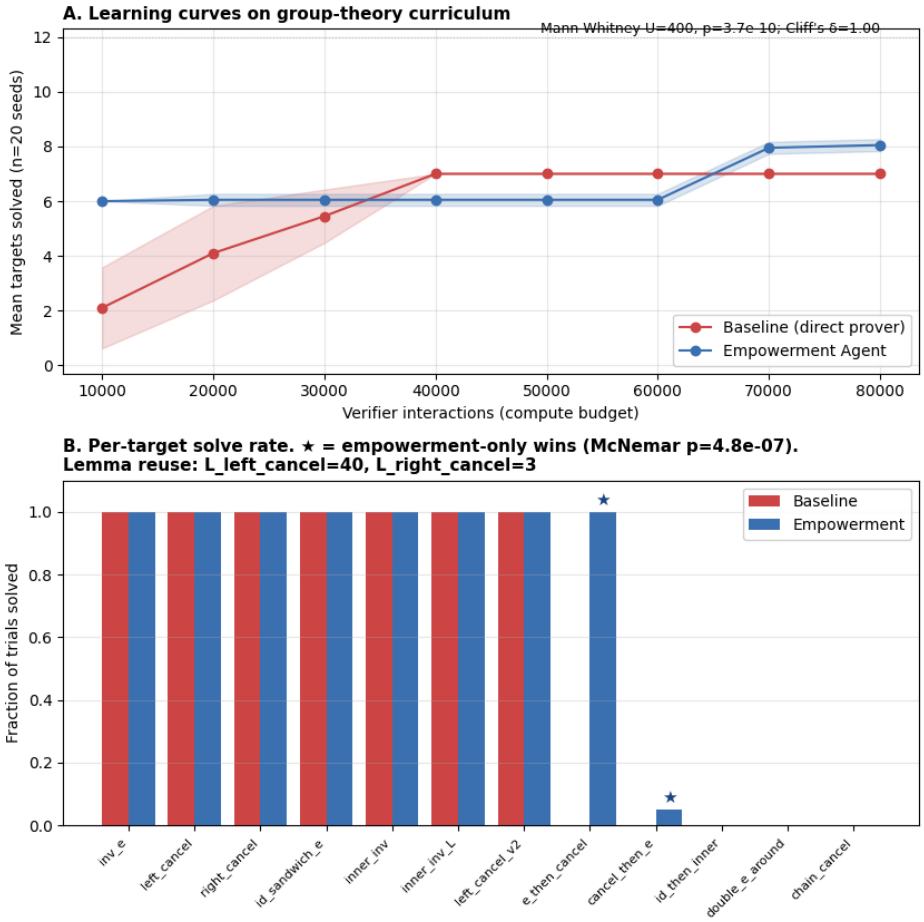

Figure 27. Group-theory curriculum results. The learning curves and per-target solve rates show where empowerment-style search changes theorem coverage relative to a direct prover baseline.

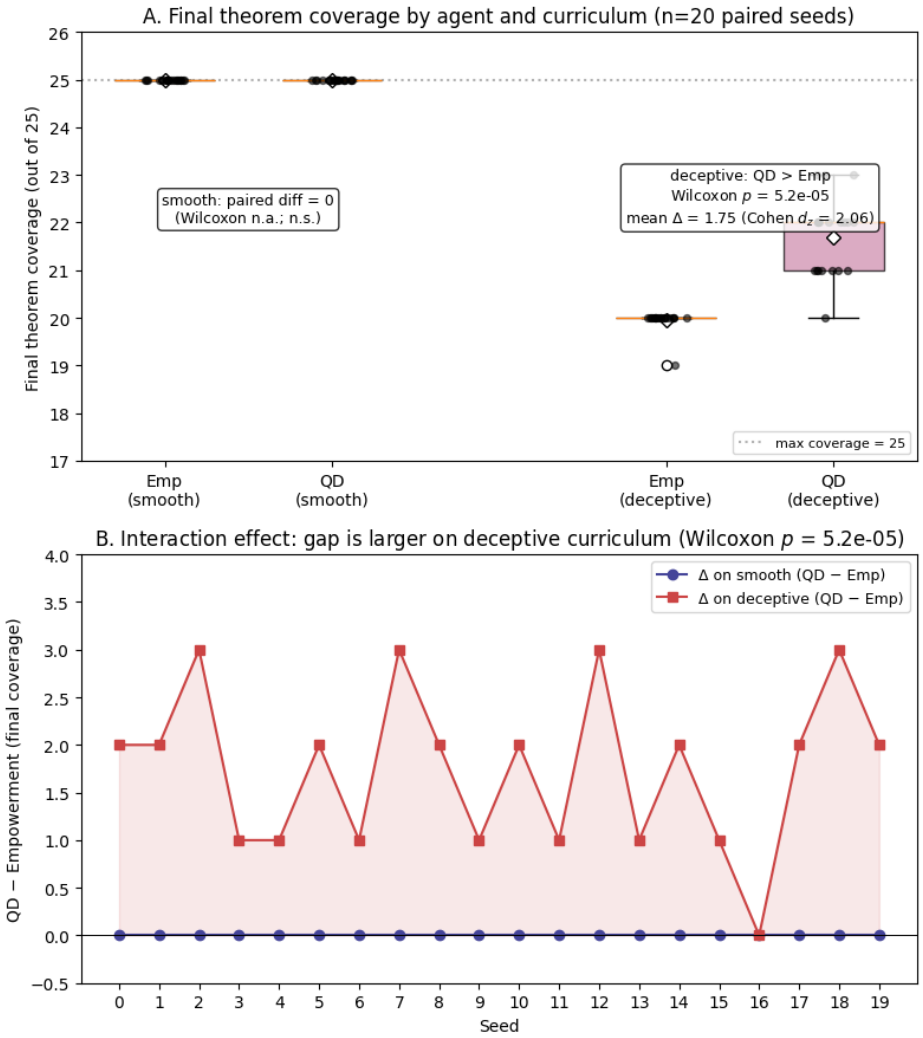

Figure 28. Curriculum interaction for theorem coverage. QD and empowerment match on smooth targets, while QD improves coverage on deceptive targets where diversity preserves useful stepping stones.

N. Failure Modes and Mitigations

Table 5 summarizes common failure modes and the specific architectural controls that localize or block them. The goal is not to make every proposed artifact valid, but to make invalidity observable before it contaminates the archive.

Table 5. Failure modes addressed by the architecture.

Failure mode	Symptom	Mitigation
Paper-shaped hallucination	report contains claims unsupported by executed artifacts	claim-package lineage and replay gates
Spec drift	formal theorem is true but not the intended theorem	semantic-alignment and anti-vacuity gates
Archive pollution	high-scoring but invalid candidates become stepping stones	verification-gated QD admission
Verifier overfitting	self-modification weakens checks to improve visible score	hidden validators and adoption budgets
Mode collapse	search repeatedly mutates a single idiom	descriptor-based QD and novelty-horizon metrics
Causal misinterpretation	predictive fit is reported as intervention evidence	causal protocol and falsification discipline

O. Additional Related Work

Open-ended discovery has roots in novelty search, QD, POET, artificial life, and self-generated curricula (Lehman & Stanley, 2011; Pugh et al., 2016; Mouret & Clune, 2015; Wang et al., 2019; Schmidhuber, 2013). LLM-guided program evolution provides recent evidence that foundation models can serve as high-level mutation priors when paired with external evaluators (Romera-Paredes et al., 2024; Lehman et al., 2023; Novikov et al., 2025; Bradley et al., 2024). Formal methods provide the complementary correctness substrate: theorem-proving environments, proof repair, proof-artifact co-training, repository-level verification, and formal semantics for executable systems (Yang et al., 2023; Han et al., 2022; First et al., 2023; Song et al., 2024; Zhong & Si, 2025; Armstrong et al., 2019). Information-theoretic and active-inference principles justify curiosity, expected information gain, compression progress, and robust experimental design as intrinsic objectives (Friston et al., 2015; 2017; Beck et al., 2018; Go & Isaac, 2022; Ao & Li, 2024; Grünwald, 2007). The proposed REE architecture is the intersection: divergent search over typed artifacts, formal and empirical gates, and guarded recursion over the discovery machinery itself.